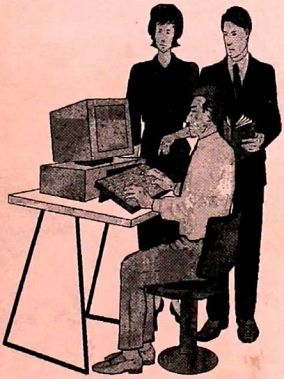
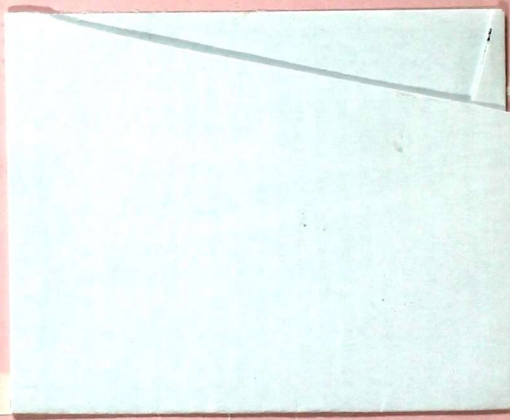
А.О. КЫБЫРАЕВ, Н.С. БЕДЕЛОВА

ПАСКАЛЬ ТИЛИ БОЮНЧА ЛАБОРАТОРИЯЛЫК ПРАКТИКУМ



Ош 2005



А.О. КЫБЫРАЕВ, Н.С. БЕДЕЛОВА

ПАСКАЛЬ ТИЛИ БОЮНЧА ЛАБОРАТОРИЯЛЫК ПРАКТИКУМ

Ош-2005

УДК 51
ББК 22.1
К- 97

Рецензенттер:

Сатыбаев А.Ж., И. Разаков атындагы КӨУнин «Колдонмо жана эсептөө математикасы» кафедрасынын башчысы, ф.-м.и.к., доцент
Сагындыков М.К., Ош МУнун Информатика кафедрасынын башчысы, ф.-м.и.к., доцент

Кыбыраев А.О., Беделова Н.С.
К-97 Паскаль тили боюнча лабораториялык практикум.-Ош: Ош МУнун басма борбору, 2005. – 106 б.

ISBN 9967-08-109-0
К 1602000000-05

Окуу колдонмо Паскаль программалоо тилинин жардамы менен ар кандай типтеги маселелерди чечүүнүн алгоритмин түзүүгө, программалоого ык-машыгууларды калыптандырууга арналган. Мында тиешелүү лабораториялык иштердин баяндамалары, аларды аткаруунун максаттары, өз алдынча иштөө үчүн тапшырмалар, ар бир ишгин кыскача теориялык материалдары, 1-2 мисалдын чыгарылыштарынын алгоритмдеринин жана программаларынын үлгүлөрү, аткаруу үчүн берилген тапшырмалардын варианттары, кайталоо үчүн тапшырмалар, көрсөтмөлөр берилген.

Окуу колдонмонун материалдары «Алгоритмдик тилдер жана программалоо», «Программалоо тилдери жана трансляциялоо методдору», «ЭЭМде практикум», «Программалоо технологиялары» ж.б.у.с. предметтерди окутууда колдонулушу мүмкүн.

Бул колдонмо жогорку жана орто окуу жайларынын студенттери үчүн арналган. Ошондой эле Паскаль тилинде программалоону өз алдынча үйрөнүүгө кызыккандар да колдонсо болот.

Ош МУнун Окумуштуулар Кеңешинин сунушу менен басмадан чыгарылган

ISBN 9967-08-109-0
К 1602000000-05

ББК 22.1

Мазмуну

КИРИШ СӨЗ

Азыркы мезгилдеги программалоо технологияларында жана андагы колдонулуучу усулдарга өтө жакын болгон программалоо тилдеринин бири - Паскаль тили болуп эсептелет.

Паскаль тилинин идеологиясы толугу менен структуралык программалоонун негизги талаптарына, идеяларына жооп берет жана ошол себептен ал структуралык тилдердин классына кирет. Паскаль тилинин башкаруучу конструкциялары жана ар түрдүү берилгендердин структуралары менен иштей алышы – бул тилдин негизги өзгөчөлүк сапаты болуп саналат. Көпчүлүк маселелерди чечүүдө алардын ар түрдүү деталдаштыруу деңгээлдеринде атайын блок-схема көрүнүшүндөгү алгоритмдерине жана программалык проекттерге кайрылып олтурбастан эле тупадан -туура программаларын жазууга мүмкүнчүлүк берет.

Паскаль программалоо тилин үйрөнүүдө, ар түрдүү типтеги маселелерди чечүүчү программаларды иштеп чыгууда практикалык ыкмаларды көндүмдөргө ээ болуу, программалоо сырларын терең өздөштүрүү чоң мааниге ээ жана квалификациялуу программист болууну каалагандар үчүн негизги сапаттардын жана максаттардын бири. Ошондуктан ушул таризде жазылган «Паскаль программалоо тили боюнча лабораториялык практикум» окуу колдонмосу колдонуучу үчүн жогорудагы сапаттарга ээ болууда пайдалуу окуу куралы болуп эсептелет.

Авторлор тилдин толук баяндамасын жазууну максат кылышкан жок. Себеби теориялык материалдар лекциялык сабактарда, окуу китептеринде толугураак берилет. Ошентсе да теориялык мүнөздөгү кыскача маалыматтар менен камсыз кылуу менен практикалык жана лабораториялык көнүгүүлөргө (Турбо-Паскаль тилинин базасында) басым жасалды. Практикалык жана лабораториялык жумуштардын структурасы төмөндөгүдөй түзүлгөн:

- Иштин максаты;
- Өз алдынча даярдоо үчүн тапшырмалар;
- Теориялык материалдын кыскача баяндамасы;
- Мисал же маселенин чыгарылышынын үлгүсү;
- Тапшырмалардын варианттары(20 вариант);
- Кайталоо үчүн суроо-тапшырмалар.

Бул практикалык жана лабораториялык окуу колдонмону алгоритмдик тилдердин жардамында программалоого байланышкан дисциплиналарга жакын негизги курстарда (тандоо курсу, программалоо тилдери, программалоо технологиялары, ж.б.) колдонууга болот.

Лабораториялык практикумду жакшыртуу максатында авторлор тарабынан бардык сын-пикирлер, каалоолор ыраазычылык менен кабыл алынат.

Лабораториялык иш №1

Тема: ***Паскаль тилинин программалык структурасы.***

Иштин максаты - Паскаль тилиндеги программанын структурасын билүү жана анын жазылыш иретин үйрөнүү.

Өз алдынча иштөө үчүн тапшырмалар

1. Программанын структурасынын жалпы форматын билүү.
2. Анын жазылыш тартибин сактоо.
3. Бул тилдеги программанын структурасын башка программалоо тилдеринен айырмалай билүү.

Паскаль тилиндеги программанын структурасы 3 бөлүктөн турат:

- **программаның бөркү;**
- **баяндоо бөлүгү;**
- **операторлук бөлүгү;**

б.а.

program <аты>;
баяндоо бөлүгү;
begin
операторлук бөлүк;
end.

Бөрк(орусча – заголовок) **program** кызматчы сөзүнөн жана программанын атынан(колдонуучу тарабынан берилген) жана кашаанын ичиндеги жазылышы шарттуу эмес болгон кийирүү-чыгаруу (input, output) параметрлеринен турат. Бөрк « ; »(үтүрлүү чекит) символу менен аяктайт.

Баяндоо бөлүгү программада кездешүүчү бардык берилгендерди жарыялоо үчүн арналат. Мында берилгендердин аттары(б.а. өзгөрүлмөлөрдүн) жана алардын типтери көрсөтүлүшү керек. Бул бөлүк өз учурунда меткалардын, турактуулардын, типтердин, өзгөрүлмөлөрдүн, процедуралардын жана функциялардын баяндалышын кармап турат. Ар бир баяндоодон кийин « ; » коюлат.

Операторлор бөлүгү негизги программа жазыла турган бөлүк. Ал **begin**(баштоо) жана **end**(аяктоо) кызматчы сөздөрү (операторлук кашаалар) менен жазылып, **end** тен кийин чекит коюлат. Операторлор бөлүгүндө аткарылуучу операторлордун удаалаштыгы жазылат. Ар бир оператор аткаруу зарыл болгон аракетти чагылдырат. Алар бири-биринен үтүрлүү чекит менен ажыратылат.

Турбо-Паскаль тилиндеги программа төмөнкүдөй структурада (форматта) жазылат:

program ***программанын аты;***
label
меткалар;
const
турактууларды баяндоо;
type
берилгендердин тибин аныктоо;
var
өзгөрүлмөлөрдү баяндоо;

процедуралар;
функциялар;

begin

негизги программанын телосу (программалык бөлүк);

end.

Эскертүү: **end** кызматчы сөзүнөн мурда үтүрлүү чекитти (;) койбой койсо да болот. Label, const, type, var, процедуралар жана функциялар баяндоолорунун ар бири программада катышуусу жана атайын иреттелип жайгашуусу зарыл эмес.

Турбо-Паскаль тилинде операторлор жөнөкөй жана татаал болуп бөлүнүшөт. Жөнөкөй операторлорго ыйгаруу, өтүү, кийирүү-чыгаруу операторлору, ал эми татаал операторлорго шарттуу, составдык, тандоо операторлору жана циклдик операторлор киришет.

Текшерүү үчүн суроолор

1. Турбо-Паскаль трансляторун кантип жүктөйбүз?
2. Паскалда программанын структурасы кандай?
3. Программанын структурасы канчага бөлүнөт жана ал бөлүктөрдүн кызматтары кандай?

Лабораториялык иш №2.

Тема: ***Арифметикалык туюнтмалардын жана стандарттык функциялардын Турбо-Паскаль тилинде жазылышы.***

Иштин максаты: Турбо-Паскаль (кыскача ТР) тилинде арифметикалык туюнтмалардын жана стандарттык функциялардын жазылышын мурда окуп үйрөнүлгөн Бейсик тили менен салыштырып үйрөнүү (Таблица 1).

Өз алдынча иштөө үчүн тапшырмалар

1. Арифметикалык функцияларды жазуу эрежелери.
2. Стандарттык функциялардын жазылышы.
3. Кээ бир функциялардын келтирилип чыгарылышы.

Таблица 1

к-н	Математикалык жазылышы	Бейсикте жазылышы	Паскалда жазылышы
1.	sinx	SIN(X)	SIN(X)
2.	cosx	COS(X)	COS(X)
3.	tgx	TAN(X)	SIN(X)/COS(X)
4.	ctgx	1/TAN(X)	COS(X)/SIN(X)
5.	$\|x\|$	ABS(X)	ABS(X)
6.	e^x	EXP(X)	EXP(X)
7.	x^2	X^2	SQR(X)
8.	$\sqrt{x}$	SQR(X)	SQRT(X)
9.	lnx	LOG(X)	LN(X)
10.	lgx	LOG(X)/LOG(10)	LN(X)/LN(10)

11.	$\log_a x$	LOG(X)/LOG(A)	LN(X)/LN(A) {(A>0, A<>0)}
12.	аргументтин бүтүн бөлүгү	INT(X)	INT(X)
13.	π	PI	PI
14.	arctgx	ATN(X)	ARCTAN(X)
15.	arcsinx	ATN(X/SQR(1-X^2)))	ARCTAN(X/SQRT(1-SQR(X))) {(X<1)}
16.	arccosx	ATN(SQR(1-X^2))/X)	ARCTAN(SQRT(1-SQR(X))/X) {(X<1, X<>0)}
17.	arcctgx	ATN(1/X)	ARCTAN(1/X) {(X<>0)}
18.	кокустук сан	RND(X)	RANDOM(X)
19.	a^b	A^B	EXP(B*LN(A))
20.	x^3	X^3	EXP(3*LN(X))

Градуста берилген маанилерди радианга которуу үчүн төмөнкү формуланы колдонууга болот: <радиан>=<градус>PI/180^0,

мында PI=3,1415... - туракту иррационалдык саны.

Мисал -1. Төмөнкү туюнтмаларды Турбо-Паскалда жазгыла:

1. 5, 28cosx+$\frac{1}{x}$-lnx туюнтмасынын TPда жазылышы: 5.28*COS(X)+1/X-LN(X);
2. sin(cos(x))-e^x+x^4 туюнтмасы үчүн: SIN(COS(X))-EXP(X)+EXP(4*LN(X));
3. arcsinx+e^{x2}-lgc туюнтмасынықы ARCTAN(X/SQRT(1-SQR(X)))+EXP(SQR(X))-ln(C)/LN(10) болуп жазылат.

Варианттар

1. y=tg(a+b+c)+e^{a+b+c}
2. z=$\log_3 x^2$-e^{3x}+$\sqrt{x}$
3. y=e^{x+a}+a^{b-x}
4. t=arcsin(ln(x+3)-1)
5. m=x^{y+z}-c^{lnx}+$\log_n$(abc)
6. h=2rsin(x^2+5)+ln|x − 1|
7. r=lg(a+b)-arccos(a-b)
8. y=arcctg(a+1)-e^{a+x}-$\sqrt{a+x}$
9. L=e^{3x}-cos^{2}x+$\sqrt[3]{\cos x - 1}$
10. z=3ab^2+tg$\frac{c}{2}$
11. u=sin(arccos$\frac{1}{a}$)+x^3
12. f=$\log_2$x+e^{2x}-|x|
13. t=$e^{\sin ac}$+e^{ax}
14. z=0,1x^2-tg$\left|\frac{x}{2}\right|$+ln|x^2 − 5|
15. v=$\frac{1}{\cos^2 x}$ − $tg^2 x$ + 0,001
16. y=tgx+10^x-0,7
17. z=$\frac{\sin a + \cos x}{|x|}$ + $\frac{tga}{1-\cos a}$
18. x=ln(a^2+c^2)+ $e^{a^2+b^2+c^2}$
19. z=lg(b-a)-ax^2-tg(abx)
20. q=$\left(\frac{a}{bc} - e^{x^2}\right)^3$ − $\log_n$ x

Текшерүү үчүн суроолор

1. Программалоо тилинде кандай чоңдуктардын тиби колдонулат?
2. Чыныгы жана бүтүн типтердин маанилеринин диапазонун көрсөткүлө.
3. Программада өзгөрүлмөлөрдүн кандай аттарын берүүгө болот?
4. Туюнтмаларда кашаалардын колдонулушунун ролу кандай?

Лабораториялык иш №3

Тема: ***Сызыктуу структурадагы алгоритмдерди программалаштыруу.***

Иштин максаты: Сызыктуу структурадагы эсептөө процессин программалаштыруу жана аны иштеп чыгууну практикалаштыруу.

Өз алдынча иштөө үчүн тапшырмалар

1. Өзгөрүлмөлөрдүн, стандарттык функциялардын, константалардын жазылышын үйрөнүү;
   -арифметикалык функциялардын жазылуу эрежеси;
   -ыйгаруу оператору;
   -жөнөкөй берилгендердин киийрүү-чыгаруусун уюштуруу.
2. Тапшырма менен дал келген чечимдин алгоритмин иштеп чыгуу.
3. Маселенин чечиминин программасын түзүү.
4. Программанын аткарылышын компьютерден анализдеп берүү.

Жөнөкөй алгоритмдик структура болуп (б.а. сызыктуу алгоритм) жазылган тартиби боюнча биринин артынан бири удаалаш аткарылуучу операциялардын удаалаштыгы эсептелет. Мындай алгоритмдик структуранын программалоо тилинде жазылышын **сызыктуу программа** деп атайбыз. Сызыктуу алгоритмдер жана аларга тиешелеш болгон сызыктуу программалар көбүнчө кандайдыр бир шарттарды, альтернативаларды тандоо же кайталоо каралбаган жөнөкөй маселелерди чечүү үчүн арналган.

Сызыктуу структурадагы программа эч кандай шартты кармабайт жана мындай программа жазылыш удаалаштыгы боюнча аткарылат.

Ыйгаруу оператору- каалаган программалоо тилинин негизги оператору болуп эсептелет жана ал кандайдыр бир туюнтманын маанисин өзгөрүлмөгө ыйгаруу үчүн колдонулат. Оператордун жазылышынын жалпы көрүнүшү(форматы) төмөндөгүдөй:

T:=X;

мында Т-өзгөрүлмөнүн аты, «:=»-ыйгаруу белгиси, Х-туюнтма.

Эскертүү: Өзгөрүлмө жана туюнтманын мааниси бир типке таандык болушу зарыл.

Турбо-Паскаль тилинде берилгендерди киийрүү жана чыгаруу үчүн процедуралар деп аталган атайын программалар каралган жана алар адатта «операторлор» деп эле аталышат. Киийрүү процедурасы **read** киийрүү операторунун жардамында, чыгаруу процедурасы **write** чыгаруу операторунун жардамында чакырылышат. Киийрүү оператору программанын аткарылуу процессинде берилгендерди киийрүү үчүн кызмат кылат. Аны менен киийрилген берилгендер өзгөрүлмөлөргө ыйгарылат.

Киийрүү оператору программанын аткарылуу процессинде берилгендерди өзгөрүлмөлөргө клавиатурадан киийрүү үчүн кызмат кылат. Берилгендер өз алдынча жолчолорго бөлүнүшү мүмкүн. Жолчонун аягынын белгиси болуп <enter> клавишасы эсептелет.

Кийирүү үчүн колдонулуучу операторлор:

1) read(x1, x2,..., xn) - ар бир кийирилген маани тиешелеш түрдө удаалаш x1, x2, ...,xn өзгөрүлмөлөрүнө ыйгарылат.
2) readln(x1, x2,..., xn) - ар бир кийирилген маани жаңы жолчого өткөндөн кийин удаалаш x1, x2, ...,xn өзгөрүлмөлөрүн алат деп айтсак да болот.
3) readln – берилгендерди кийирүүдө жаңы жолчого өтүү.

Турбо-Паскаль тилинде бүтүн, чыныгы жана символдук берилгендерди кийирүүгө болот, ал эми логикалык берилгендерди кийирүүгө жол берилбейт. Бир нече берилгендерди өзгөрүлмөлөргө ыйгарууда read жана readln операторлорунун аткарылыштарында берилгендердин маанилеринин арасы пробел менен ажыратылат.

Чыгаруу оператору информацияларды(берилгендерди) ЭЭМдин эсинен экранга чыгарат. Информацияларды чыгаруу үчүн төмөнкү операторлор колдонулат:

1) write(x1, x2,...,xn) - x1, x2, ... , xn дердин маанилерин удаалаш экранга чыгарат;
2) writeln(x1, x2,...,xn) - x1, x2, ... , xn дердин маанилерин удаалаш чыгарат жана кийинки чыгаруу үчүн жаңы жолчого өткөрөт;
3) writeln-берилгендерди чыгарууда жаңы жолчого өтүүнү камсыз кылат.

Чыгаруу операторунда жазылган өзгөрүлмөлөрдүн аты бүтүн, чыныгы, символдук, логикалык типке же башка структуралашкан типке таандык болушу мүмкүн.

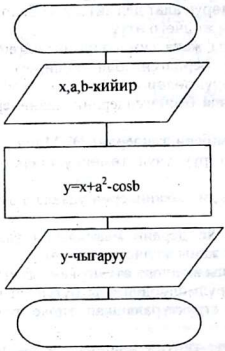
Фиксирленген чекити менен чыныгы типтин маанилерин чыгарууда сандын бөлчөк бөлүгүн да келтирип чыгаруучу талаанын туурасы көрсөтүлөт. Операторду жазуунун жалпы көрүнүшү:

write(y: k:n); же writeln(y:k:n);

мында k – чыгарылуучу у чоңдугунун мааниси үчүн жалпы позициялардын саны(үтүр үчүн бир позиция кошо эсептелгенде); n - сандын бөлчөк бөлүгү үчүн(үтүрдөн кийин) ажыратылган позициялардын саны.
write(X:8:3); Бул учурда Хтин маанисине 8 позиция бөлүнөт, 3өөсү бөлчөк бөлүгүнө чыгат. Эгерде чыныгы маанилерди чыгарууда позициясы көрсөтүлбөсө, анда жыйынтык ондук тартип менен нормалдаштырылган көрүнүштө алынат.

Мисал - 1. Берилген y=x+a²-cosb функциясынын маанисин эсептеткиле жана аны печатка чыгаргыла.

Алгоритми:



Программасы:

```
program m_1;
var
   a,b,y:real;

begin
      read(x,a,b);
y:=x+sqr(a)-cos(b);
write('y=',y:8:3);
readln;
end.
```

Варианттар

1. $y=e^{\sin^2 x}+\arccos(a+1)/tg^2(ax)$
2. $x=y^3-2ab+e^{yab}+c$
3. $t=\frac{x+1}{3}-2(x+1)^2$
4. $y=\sin(\arccos x)-\ln(ax^2+bx+c)$
5. $m=5x^3+3(x^2-1)$
6. $m=x^3-2ab+e^{xab}+c$
7. $y=\left|2x^2-tg(a+b)\right|-\ln|x|$
8. $z=(\cos(a-1)-e^x)Y\sqrt{\arcsin ax-a^2}$
9. $y=\cos^2(arctgx)$
10. $z=\cos ax^2-\left|x^2-a^2\right|$
11. $t=(2x)^{x^2}+\lg x^2-\ln\left|\frac{1}{x}\right|$
12. $r=\left|x^2-tg(a+x)\right|$
13. $y=\frac{e^{x^2}-\sqrt[3]{\cos^2 x-1}}{\lg x+0{,}1x^2}$
14. $k=\frac{x^2-5x-0{,}6}{ax^2+bx+c}-\lg(ax^2)$
15. $f=2^{\sin x}+\frac{|ax+by|}{\sqrt{x^2+y^2}}$
16. $y=\ln\left|(b-\sqrt{a})(a-\frac{b}{c+a^2}\right|$
17. $y=\frac{(x+1)^2+2(x+1)}{4}$
18. $z=x^3+2x^4+6$
19. $z=\frac{x}{3}+\left(\frac{x}{3}\right)^2+1$
20. $y=\frac{x^2+1}{2}-\frac{27}{x^2}$

Текшерүү үчүн суроолор

1. Программалоо тилинде кандай чондуктардын тиби колдонулат?
2. Чыныгы жана бүтүн типтеринин маанилеринин диапазонун көрсеткүлө.
3. Программада өзгөрүлмөлөрдүн кандай аттарын берүүгө болот?
4. Кийирүү-чыгаруу, ыйгаруу операторлору кайсылар?

Лабораториялык иш №4

Тема: ***Бутактануучу структурадагы алгоритмдерди программалаштыруу.***

Иштин максаты: Бутактануучу структурадагы эсептөө процессин программалаштыруу жана аны практикада колдонууну үйрөнүү.

Өз алдынча иштөө үчүн тапшырмалар

1. Шарттуу жана шартсыз өтүү операторлорунун, бутактануучу структурадагы эсептөө процессинин реализацияланышы үчүн программалоо тилинин мүмкүнчүлүгүн үйрөнүү;
2. Тапшырмага ылайык маселенин чечимин табуучу алгоритмди иштеп чыгуу.
3. Маселенин чечимин табуунун программасын түзүү.
4. Программанын аткарылышын компьютерде анализдөө.

Паскалда бутактануучу структурадагы программа кээ бир шарттан көз каранды болгон бир нече удаалаштыктагы операторлордун бирөөсүн тандоону карайт. Программада бутактануу уюштуруу үчүн **шарттуу, тандоо, өтүү операторлору** колдонулат.

Шарттуу оператор аныкталган шарттын аткарылышынан же аткарылбашынан көз каранды болгон түрдүү жол менен эсептөө жүргүзүлгөн учурда колдонулат.

Шарттуу оператордун жалпы форматы:

If <логикалык туюнтма> then <оператор1> else <оператор2>;

Мында if(эгерде), then(анда), else(антпесе) - кызматчы сөздөр,

оператор1, оператор2 – Паскаль тилинин жөнөкөй же составдык операторлору.

Шарттуу операторлун аткарылышы:

Эгерде логикалык туюнтма чын болсо, анда оператор1 аткарылат, антпесе, б.а. логикалык туюнтма жалган болсо, анда оператор2 аткарылат. Операторлор катары өз кезегинде шарттуу операторлор болуп калышы да мүмкүн. Шарттуу оператор татаал операторго кирет.

Кээде шарттуу операторлун башка жазылышы да колдонууга ынгайлуу болот жана структуралык программалоо талаптарына туура келет.

if<логикалык туюнтма>
then <оператор 1>
else <оператор 2>;

Шарттуу операторлун аракетин составдык операторду колдонуп кенейтүүгө болот. Бул учурда then жана else сөздөрүнөн кийин составдык оператор болушу мүмкүн. Мисалы:

If <логикалык туюнтма> then
begin

<оператор11>…<оператор1n>
end
else
begin
<оператор21>;…<оператор2n>
end;

Else кызматчы сөзүнөн мурда үтүрлүү чекит коюлбайт. Составдык оператордун ичинде жөнөкөй жана составдык операторлорду кармап турган шарттуу оператор болушу мүмкүн. Камтылуунун терендиги түрдүү тилдин версияларында чектелиши мүмкүн.

Турбо-Паскаль тилинде шарттуу операторду кыскартылган формасы да колдонулат:

if **<логикалык туюнтма> then<оператор >;**

Эгерде логикалык туюнтма чын болсо, анда оператор аткарылат, антпесе программада if шарттуу операторунан кийин жазылган оператор аткарылат. Мисалы,

… ; if x>5 then y:=x+11; k:=k+1; …

Эгерде x>5 шарты чын болсо y:=x+1 ыйгаруу оператору аткарылат, антпесе шарттуу оператордон кийин турган k:=k+1 ыйгаруу оператору аткарылат. Шарттуу операторду кыска формасын колдонуудан этият болуу керек. Себеби камтылган шарттуу операторлордо бардык структура бузулушу да мүмкүн. Кыска формасынын ордуна дайыма толук формасын колдонуу сунуш кылынат. Демек, каралган мисалга төмөнкү конструкцияны түзүү жакшыраак:

if x>5 then y:=x+11;
else;
k:=k+1;

Мында else сөзүнөн кийин эч нерсе коюлбай, бош оператор колдонулат.Ушул эле мисалды then операторунан кийин бош операторду колдонуп жазсак да болот:

if x>5 then
else y:=x+11;
k:=k+1;

Шартсыз өтүү оператору . Программалоо практикасында кээде операторлордун удаалаш аткарылыш тартибин бузуу зарылчылыгы келип чыгат. Ал үчүн төмөнкү форматка ээ болгон **шартсыз өтүү** же жөн эле **өтүү** оператору делген оператор колдонулат:

Жазылышы:

goto <метка>;

Мында **goto** (баруу) – кызматчы сөз, **метка** – метканы баяндоо бөлүгүндө жарыяланган белги.

Аткарылышы: goto операторуну башкарууну программанын **метка** турган чекитине өткөрөт.

Өтүү операторуну жөнөкөй операторго кирет, себеби анын составына башка операторлор кирбейт. Метка программанын баяндоо бөлүгүндө жарыяланышы керек. Ал үчүн жарыялоо төмөнкү көрүнүшкө ээ болот:

label<метка>;

Жөнөкөй программаларда өтүү операторуну кыйынчылык туудурбайт, ал эми чоң көлөмдөгү татаал программаларда операторлордун аткарылуусун көзөмөлдөөдө жана карап чыгууда бир топ кыйынчылыкты

туудурат. Өтүү операторун көп колдонуу менен программанын айкындыгы жана түшүнүктүүлүгү бузулушу мүмкүн. Ошондуктан өтүү операторун мүмкүн болушунча колдонбоо сунуш кылынат.

Мисалы, программанын төмөнкү ыңгайсыз болгон фрагментин

```
…; if x>y then goto 1;
x:=x-y; goto 2;
1: x:=x+y; 2: z:=t;…
```

Паскаль тилинин төмөнкү конструкциясы менен алмаштырса болот:

```
…; if x>y then x:=x+y else x:=x-y;
z:=t;…
```

Тандоо оператору(вариантты тандоо) - кандайдыр бир туюнтманын маанисинен көз каранды болгон учурда бир нече удаалаштыктагы операторлордун бирин аткаруу зарыл болгон учурда колдонулат. Тандоо операторун татаал операторго кирет.

Жазылышы:

```
case <туюнтма> of
     <константа 1 >:<оператор 1 >;
     …
     <константа n >:<оператор n >
end;
```

Мында case(учурда), of(дан), end(аягы) - кызматчы сөздөр, константа1, …., константа n дер турактуулар же турактуулардын тизмеси, ал эми оператор1, …., оператор n – Паскаль тилинин операторлору.

Аткарылышы: Эгерде туюнтманын мааниси константалардын бирине барабар болсо(дал келсе), анда ага туура келүүчү оператор аткарылат, андан кийин башкаруу тандоо операторунан кийинки турган операторго берилет. Эгерде туюнтманың мааниси константалардын бири менен да дал келбесе, анда башкаруу тандоо операторунан кийин турган операторго өткөрүлөт.

Туюнтманын тиби чыныгы(real) типтен башка бардык стандарттык типке ээ болушу талап кылынат. Мунун негизинде константа чыныгы типте болбошу керек жана константанын тиби менен туюнтманын тиби дал келүүсү зарыл.

Мисал, case n+1 of

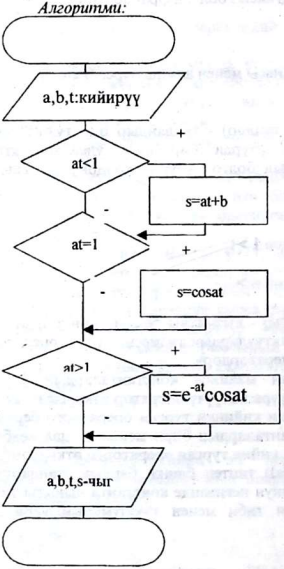
```
        1: z:=sqr(x);
        2: y:=sqrt(x);
        3: z:=abs(y)
end;
```

Эгерде n+1 туюнтмасынын мааниси 1 ге барабар болсо, анда y:=sqr(x) ыйгаруу оператору аткарылат жана башкаруу end сөзүнөн кийин жазылган операторго берилет.Ушул сыяктуу эле n+1 – дин мааниси 2 ге, 3 кө барабар болсо, анда аларга туура келүүчү операторлордун бири аткарылат жана андан кийин башкаруу тандоо операторунан кийинки (end ден кийинки) операторго берилет.

Мисал-1. a,b,t – лардын чыныгы маанилеринде берилген функциянын маанисин эсептегиле:

$$S=\begin{cases} at+b, & \text{эгерде } at<1 \text{ болсо;} \\ cosat, & \text{эгерде } at=1 \text{ болсо;} \\ e^{-at}\,cosat, & \text{антпесе (б.а. } at>1 \text{ болсо).} \end{cases}$$

Алгоритми:



Программасы:

```
program m_2;
var
   a,b,t,s:real;
begin
  writeln('a,b,t лары    кийиргиле:');
  read(a, b, t);
  if a*t<1 then s:=a*t+b
     else
  if a*t=1 then s:=cos(a*t)
     else s:=exp(-a*t)*cos(a*t);
 writeln('a=',a:8:3,'b=',b:8:3,'t=',t:8:3);
 writeln('жыйынтыгы=', s:8:3');
 readln;
 end.
```

Мисал-2. Эки санды салыштыруунун программасын түзгүлө.
Чыгаруу:

```
program two_numbers;
var
      first_number, second_number:real;
begin
      writeln('биринчи санды кийиргиле:');
      readln(first_number);
      writeln('экинчи санды кийиргиле:');   readln(second_number);
if first_number < second_number then
      writeln('кичинеси болуп биринчи сан эсептелет')
else if first_number = second_number then
```

```
    writeln('кийирилген маанилер барабар')
    else writeln('кичинеси болуп экинчи сан эсептелет');
readln;
end.
```

Варианттар

1. $y=\begin{cases}\cos x+\ln x, эгердех>1\\ e^{x+1}, эгердех\le 1\end{cases}$

2. $u=\min\{x,\min\{y,z\}\}$

3. $y=\max\{a,\min\{b,c\}\}$

4. $M=\begin{cases}(x-1)^2, x>1\\ x^2-3x+4, x\le 1\end{cases}$

5. $z=\begin{cases}|x|, -2\le x\le 2\\ x^2+1, x>2\\ x^3+4, x<-2\end{cases}$

6. $t=\min\{x,y,z\}$

7. $u=\begin{cases}e^x+\ln|x-8|, x<8\\ x^3+4x-5, x\ge 8\end{cases}$

8. $y=\begin{cases}\ln x-e^{2x}, эгердех\ge 1\\ tgx, эгердех<1\end{cases}$

9. y=max $\{a,b,c\}$

10. $y=\begin{cases}x^3-x^2-x+1, x\ge 0\\ \frac{1}{x^2}+\frac{1}{x^3}+2, x<0\end{cases}$

11. $k=\min\{\max\{a,b\},c\}$

12. $M=\begin{cases}(x+5)^2+10, x\ge 1\\ x^2+9, x<1\end{cases}$

13. $z=\begin{cases}e^x, эгерх>1\\ \cos^2 x, эгерх\le 1\end{cases}$

14. $y=\begin{cases}x^2-1, эгер-1<x<0\\ \ln x, эгерх>1\\ \cos x, эгерх<-1\end{cases}$

15. $z=\begin{cases}\sin(x-1), эгерх>1\\ tgx-ctg^2 x, эгерх\le 0\end{cases}$

16. $y=\begin{cases}\sqrt{x+3}, эгерх\ge -3\\ x^2-1, эгерх<-3\end{cases}$

17. $z=\begin{cases}x^2, эгерх<0\\ x^3, эгер0\le x\le 1\\ x^4, эгерх>1\end{cases}$

18. $y=\begin{cases}\ln(x^2+5)+\ln x, x>2\\ e^x+4, x\le 2\end{cases}$

19. $r=\begin{cases}0{,}2+e^{2x}, x>1\\ 0{,}3+\ln x, x>0\end{cases}$

20. $y=\begin{cases}0{,}5\ln x, x>2\\ |x|, x\le 2\end{cases}$

Текшерүү үчүн суроолор.

1. Шарттуу операторду аткарууда ишке ашкан аракеттерди эсептегиле.
2. Өтүү оператору менен кандай аракеттер аткарылат?
3. Бутактануучу структурадагы эсептөө процесси деген эмне?
4. а) эки бутактануу; b) үч бутактануу болгондо бутактанууну кандай уюштурууга болот?
5. 2- мисалдын программасына туура келген алгоритмди түзгүлө.

Лабораториялык иш №5

Тема: ***Whil е операторунун жардамында циклдик структурадагы алгоритмдерди программалоо.***

Иштин максаты: Циклдик структурадагы эсептөө процессин программалаштыруу жана аны иштеп чыгууну практикалаштыруу. While операторунун жардамында циклдик программаларды түзүү.

Өз алдынча иштөө үчүн тапшырмалар

1. Циклде берилген кайталоо саны менен циклдик структурадагы эсептөө процессинин реализацияланышы үчүн программалоо тилинин мүмкүнчүлүктөрүн үйрөнүү.
2. Тапшырма менен дал келген чечимдин алгоритмин иштеп чыгуу.
3. Маселенин чечиминин программасын түзүү.
4. Программанын аткарылышын компьютерден анализдеп берүү.

Көп маселелерди чечүүдө эсептөө процесси циклдик мүнөзгө ээ. Программада циклдин колдонулушу машинаны эффективдүү колдонууга мүмкүндүк берет, программанын узундугун кыскартат жана анын түзүлүшү үчүн убакыттын үнөмдөлүшүн камсыз кылат.

Турбо-Паскалда циклди уюштуруучу 3 түрдүү операторлор бар:

I. Шарты алдын-ала келүүчү оператор;

II. Шарты кийин келүүчү оператор;

III. Параметрлүү цикл.

Циклдерди жазуу үчүн операторлор татаал б.э., алардын составына башка операторлор да кирет.

Бардык циклдик операторлор үчүн төмөнкү өзгөчөлүктөр мүнөздүү.Кайталануучу эсептөөлөр бир гана жолу жазылат.Циклге кирүү башынан гана жүргүзүлөт.Циклдик оператордун өзгөрүлмөлөрү циклдик бөлүмгө киргенге чейин эле аныкталышы керек.Циклден чыгууну абайлап кароо зарыл: же кадимки анын аягынан чыгуу же өтүү оператору боюнча.Эгерде аны карабаса циклдик эсептөө чексиз кайталанат.

Шарты алдын-ала келүүчү цикл циклдин кайталоо саны алдын-ала белгисиз болгон учурда колдонулат.Жалпы форматы:

while <логикалык туюнтма> do
begin
    <оператор 1>;
    …
    <оператор n >
end

Мында while(азырынча) жана do(аткаруу)-кызматчы сөздөр. Кыска **while b do s** десек, мында b-логикалык туюнтма, б.а. шарт, s-циклдин телосу, б.а.<оператор 1> … <оператор n >ге чейин.

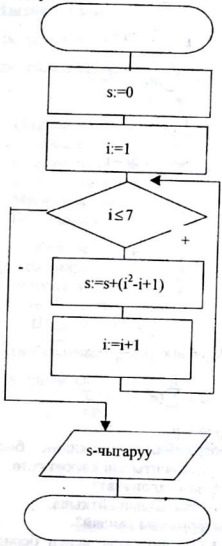
Циклдин оператору төмөнкүдөй аракеттенет. Логикалык туюнтманын мааниси алдын-ала текшерилет. Ал чын болгон учурда циклдик бөлүмдүн операторлору аткарылат, ал эми ал жалган болгондо циклден чыгат.Эгерде башынан эле логикалык туюнтманын мааниси жалган болсо, анда циклдик бөлүмдүн операторлору бир да жолу аткарылбайт.Эгерде циклдик бөлүмдө бир гана оператор турса, анда оператордук кашааларды көрсөтпесө деле болот жана төмөнкүдөй жазылат:

while <логикалык туюнтма> do <оператор >;

While операторунун жардамында циклдик программаны түзгүлө.

Мисал(1). $S = \sum_{i=1}^{7} i^2 - i + 1$ суммасын эсептөөнүн программасын түзгүлө.

Алгоритми:



Программасы:

```
program m_1;
 var
     i,s:integer;
 begin
     s:=0;
     i:=1;
 while i<=7 do
 begin
     s:=s+(sqr(i)-i+1);
     i:=i+1;
 end;
 write('s=', s:8:3);
 readln;
 end.
```

Мисал(2). $P = \prod_{n=1}^{5} n^2 - 11$ көбөйтүндүнү эсептөөнүн программасын түзгүлө.

```
program m_2;
var
      n:integer;
      p:real;
begin
      p:=1;  n:=1;
while n<=5 do
begin
      p:=p*sqr(n)-11;
      n:=n+1;
end;
write('p=', p:8:3);readln;
end.
```

Варианттар

1. $S = \sum_{n=1}^{10} (n^3 - 1)(n+1)$
2. $p = \prod_{i=1}^{5} (e^i + 1)$
3. $S = \sum_{n=1}^{5} \frac{9n}{n^2 + n + 1}$
4. $P = \prod_{i=1}^{5} (\cos i - i^2)$
5. $S = \sum_{i=1}^{10} \frac{i}{(i+1)^2}$
6. $P = abc - \prod_{n=1}^{5} n^2$
7. $S = \sum_{i=1}^{10} (i^2 + 1)$
8. $y = \ln x + \sqrt{x-1}$, мындах $x = \overline{1,10}$
9. $S = \sum_{k=1}^{7} \frac{k}{k^3 - k}$
10. $P = \prod_{n=1}^{6} (e^n + \cos^2 n)$
11. $S = \sum_{n=1}^{10} \frac{n^2 - 1}{lgn - n}$
12. $P = \prod_{n=1}^{8} (e^n + \ln n)$
13. $S = \frac{1}{5} \sum_{n=1}^{4} (\ln n + \sqrt{n})$
14. $P = \prod_{i=1}^{5} e^{i^2} + 1$
15. $S = \sum_{i=1}^{4} \frac{i^2 + 2i - 1}{i}$
16. $P = \prod_{n=1}^{10} \frac{n+1}{n^2 + 1}$
17. $S = \sum_{n=1}^{100} (1 + \frac{1}{n^2})$
18. $z = \sum_{a=1}^{9} \frac{1}{a^{2a}}$
19. $y = \ln x^2 + \frac{x^2 - 1}{2}, x = \overline{1,5}$
20. $S = \sum_{n=1}^{11} (n^3 - 1)(n+3)$

Текшерүү үчүн суроолор

1. Кайталоо саны менен берилген программанын циклдик бөлүгү уюштурулганда аткарылган аракеттердин удаалаштыгын көрсөткүлө.
2. Шарты алдын-ала келүүчү цикл кайсы учурда колдонулат?
3. Циклдин уюштурулушунун эрежесин жана арналышын айткыла.
4. Шарты алдын-ала келүүчү циклдин жалпы форматы кандай?
5. Өздөштүргөн программалоо тилинерде кайталоо саны менен берилген циклдин уюштурулушунун мүмкүн болгон ыкмаларын эсептеп бергиле.

Лабораториялык иш №6

Тема: ***Repeat операторунун жардамында циклдик структурадагы алгоритмдерди программалоо.***

Иштин максаты- Repeat операторунун жардамында циклдик программаларды түзүү. Циклдик структурадагы эсептөө процессин программалаштыруу жана аны практикада колдонуу.

Өз алдынча иштөө үчүн тапшырмалар

2. Циклде берилген кайталоо саны менен циклдик структурадагы эсептөө процессинин реализацияланышы үчүн программалоо тилинин мүмкүнчүлүктөрүн үйрөнүү.
5. Тапшырмага ылайык алгоритмди иштеп чыгуу.
6. Маселени чечүүнүн программасын түзүү.
7. Программанын аткарылышын компьютерде анализдөө.

Көп маселелерди чечүүдө эсептөө процесси циклдик мүнөзгө ээ. Программада циклдик операторлун колдонулушу ЭЭМди эффективдүү колдонууга мүмкүндүк берет, программанын узундугун кыскартат жана аны түзүүдө убакыттын үнөмдөлүшүн камсыз кылат.

Шарты кийин келүүчү циклдик оператор, эреже катары, циклдин кайталоо саны алдын-ала белгисиз болгон учурда колдонулат жана төмөнкү көрүнүштө жазылат:

repeat

<оператор 1>;

...

<оператор n>

until <логикалык туюнтма>;

мында repeat(кайталоо), until(чейин) - кызматчы сөздөр; оператор1, ..., оператор n – циклдин телосу, кайталануучу бөлүк же циклдик бөлүм; логикалык туюнтма – бульдук типтеги туюнтма.

Ошондой эле кыскача

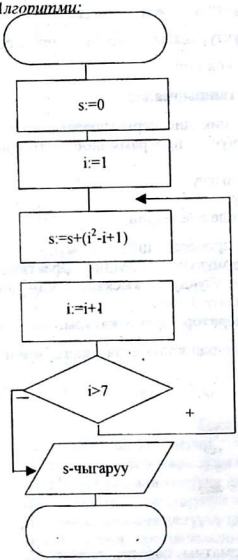
repeat s until b;

деп да жазууга болот, мында s-циклдин телосу, b-логикалык туюнтма(шарт).

Аткарылышы: Шарты кийин келүүчү циклдик оператор төмөнкүдөй аракеттенет: циклдик бөлүмдүн операторлору логикалык туюнтманын мааниси жалган болгонго чейин кайталана берет. Циклдик эсептөөнүн токтошунун шарты болуп логикалык туюнтманын чын мааниси эсептелет. Демек, алгач циклдин телосу аткарылып, анан шарт текшерилет.

Мисал-1. $S=\sum_{i=1}^{7} i^2 - i + 1$ суммасын эсептөөнүн программасын түзгүлө.

Алгоритми:



Программасы:

```
program m_1;
var
      s, i:integer;
begin
      s:=0;
      i:=1;
repeat
      s:=s+(sqr(i)-i+1);
      i:=i+1;
until i>7;
write('s=', s:8:3);
readln;
end.
```

Мисал - 2. $P=\prod_{n=1}^{5} n^2 - 11$ көбөйтүндүнү эсептөөнүн программасы берилген. Программа боюнча блок-схема түрүндөгү алгоритмин түзгүлө.

```
program m_2;
   var
      n:integer;
      p:real;
begin
      p:=1;  n:=1;
  repeat
      p:=p*sqr(n)-11;
      n:=n+1;
  until n>5;
  write('p=', p:8:3);
   readln;
end.
```

Варианттар

1. $S=\sum_{n=1}^{10}(n^3-1)(n+1)$
2. $p=\prod_{i=1}^{5}(e^i+1)$
3. $S=\sum_{n=1}^{5}\frac{9n}{n^2+n+1}$
4. $P=\prod_{i=1}^{5}(\cos i-i^2)$
5. $S=\sum_{i=1}^{10}\frac{i}{(i+1)^2}$
6. $P=abc-\prod_{n=1}^{5}n^2$
7. $S=\sum_{i=1}^{10}(i^2+1)$
8. $y=\ln x+\sqrt{x-1}$, *мында* $x=\overline{1,10}$
9. $S=\sum_{k=1}^{7}\frac{k}{k^3-k}$
10. $P=\prod_{n=1}^{6}(e^n+\cos^2 n)$
11. $S=\sum_{n=1}^{10}\frac{n^2-1}{tgn-n}$
12. $P=\prod_{n=1}^{8}(e^n+\ln n)$
13. $S=\frac{1}{5}\sum_{n=1}^{4}(\ln n+\sqrt{n})$
14. $P=\prod_{i=1}^{5}e^{i^2}+1$
15. $S=\sum_{i=1}^{4}\frac{i^2+2i-1}{i}$
16. $P=\prod_{n=1}^{10}\frac{n+1}{n^2+1}$
17. $S=\sum_{n=1}^{100}(1+\frac{1}{n^2})$
18. $z=\sum_{a=1}^{9}\frac{1}{a^{2n}}$
19. $y=\ln x^2+\frac{x^2-1}{2}$, $x=\overline{1,5}$
20. $S=\sum_{n=1}^{11}(n^3-1)(n+3)$

Текшерүү үчүн суроолор

1. Кайталоо саны менен берилген программанын циклдик бөлүгү уюштурулганда аткарылган аракеттердин удаалаштыгын көрсөткүлө.
2. Шарты кийин келүүчү цикл кайсы учурда колдонулат?
3. Циклдин уюштурулушунун эрежесин жана арналышын айткыла.
4. Шарты кийин келүүчү циклдин жалпы форматы кандай?
5. Шарты кийин келүүчү цикл менен шарты мурда келген циклдин айырмачылыгы кандай?

Лабораториялык иш №7

Тема: ***For … операторунун жардамында циклдик структурадагы алгоритмдерди программалоо.***

Иштин максаты- Параметрлүү циклдин жардамында циклдик программаларды түзүү. Циклдик структурадагы эсептөө процессин программалаштыруу жана аны иштеп чыгууну практикалаштыруу.

Өз алдынча иштөө үчүн тапшырмалар

1. Циклде берилген кайталоо саны менен циклдик структурадагы эсептөө процессинин реализацияланышы үчүн программалоо тилинин мүмкүнчүлүктөрүн үйрөнүү.
2. Тапшырма менен дал келген чечимдин алгоритмин иштеп чыгуу.
3. Маселенин чечиминин программасын түзүү.
4. Программанын аткарылышын компьютерден анализдеп берүү.

Параметрлүү циклдик оператор циклдин саны канча жолу кайталанышы белгилүү болгон учурда колдонулат.

Жазылышы:

for i:=e1 to e2 do

b;

ында **for** (үчүн), **to**(чейин), **do**(аткаруу) - кызматчы сөздөр, i - циклдин параметри, e1-циклдин параметринин баштапкы маанеси, e2 - циклдин параметринин акыркы маанеси; b – оператор, циклдин телосу.

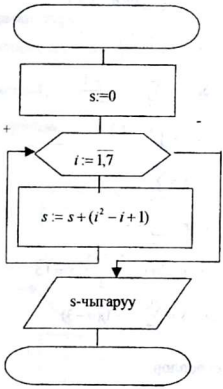
Циклдин телосу циклдин параметри i нин ар бир маанеси үчүн e1 ден e2 ге чейин +1 кадам(ошондой эле -1 кадам) менен өзгөргөнгө чейин кайталанат. Циклдин параметри катары өзгөрүлмө колдонулат, ал эми e1, e2 нин ордуна туюнтма болушу да мүмкүн. Көбүнчө циклдин параметри I ни бүтүн типтин өзгөрүлмөсү катары колдонушат, ал эми өзгөрүү кадамы +1 ге же –1 ге барабар болот. Эгерде циклдин параметринин мааниси өссө, анда кадамы +1 ге өзгөрөт. Эгерде циклдин параметринин мааниси кемисе, анда кадамы –1 ге өзгөрөт жана for операторунда to кызматчы сөзүнүн ордуна downto кызматчы сөзү колдонулат, б.а. төмөндөгүдөй жазылат:

for i:=e1 downto e2 do b;

Параметрлүү циклдик операторду i циклдин параметри циклдик бөлүмдүн ичинде аныкталышы керек эмес. Эгерде параметрдин өзгөрүү

кадамы +1 ге(-1 ге) барабар болсо жана e1>e2 (e1<e2) болсо, анда циклдик бөлүм бир да жолу аткарылбайт.

Мисал-1. $S=\sum_{i=1}^{7} i^2-i+1$ суммасын эсептөөнүн алгоритмин жана программасын түзгүлө.




```
program m_1;
var
      s, i:integer;
begin
      s:=0;
      for i:=1 to 7 do
      s:=s+(sqr(i)-i+1);
      write('s=', s:8:3);
      readln;
end.
```

Мисал(2). $P=\prod_{n=1}^{5} n^2-11$ көбөйтүндүнү эсептөөнүн программасын түзгүлө.

```
program m_2;
var
      n:integer;
      p:real;
begin
      p:=1;
      for n:=1 to 5 do
      p:=p*sqr(n)-11;
      write('p=', p:8:3);
readln;
end.
```

Варианттар

1. $S=\sum_{n=1}^{10}(n^3-1)(n+1)$
2. $p=\prod_{i=1}^{5}(e^i+1)$
3. $S=\sum_{n=1}^{5}\frac{9n}{n^2+n+1}$
4. $P=\prod_{i=1}^{5}(\cos i-i^2)$
5. $S=\sum_{i=1}^{10}\frac{i}{(i+1)^2}$
6. $P=abc-\prod_{n=1}^{5}n^2$
7. $S=\sum_{i=1}^{10}(i^2+1)$
8. $y=\ln x+\sqrt{x-1}$, *мында* $x=\overline{1,10}$
9. $S=\sum_{k=1}^{7}\frac{k}{k^3-k}$
10. $P=\prod_{n=1}^{6}(e^n+\cos^2 n)$
11. $S=\sum_{n=1}^{10}\frac{n^2-1}{tgn-n}$
12. $P=\prod_{n=1}^{8}(e^n+\ln n)$
13. $S=\frac{1}{5}\sum_{n=1}^{4}(\ln n+\sqrt{n})$
14. $P=\prod_{i=1}^{5}e^{i^2}+1$
15. $S=\sum_{i=1}^{4}\frac{i^2+2i-1}{i}$
16. $P=\prod_{n=1}^{10}\frac{n+1}{n^2+1}$
17. $S=\sum_{n=1}^{100}(1+\frac{1}{n^2})$
18. $z=\sum_{n=1}^{9}\frac{1}{a^{2n}}$
19. $y=\ln x^2+\frac{x^2-1}{2}$, $x=\overline{1,5}$
20. $S=\sum_{n=1}^{11}(n^3-1)(n+3)$

Текшерүү үчүн суроолор

1. Кайталоо саны менен берилген программанын циклдик бөлүгү уюштурулганда аткарылган аракеттердин удаалаштыгын көрсөткүлө.
2. Параметрлүү цикл кайсы учурда колдонулат?
3. Циклдин уюштурулушунун эрежесин жана арналышын айткыла.
4. Параметрлүү циклдин жалпы форматы кандай?
5. Өздөштүргөн программалоо тилиңерде кайталоо саны менен берилген циклдин уюштурулушунун мүмкүн болгон ыкмаларын эсептеп бергиле.

Лабораториялык иш №8

Тема: ***Камтылуучу циклдер.***

Иштин максаты- камтылуучу циклдер менен эсептөө процессин уюштуруу, программалаштыруу жана аны иштеп чыгууну практикалаштыруу.

Өз алдынча иштөө үчүн тапшырмалар

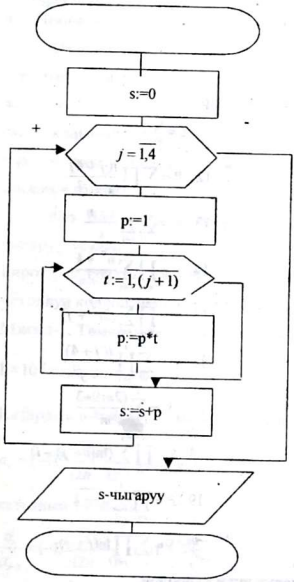
1. Камтылуучу циклдерди чыгарууну кароо.
2. Тапшырма менен дал келген чечимдин алгоритмин иштеп чыгуу.
3. Маселенин чечиминин программасын түзүү.

4. Программанын аткарылышын компьютерден анализдеп берүү.

Циклдер бири-бирине камтылган болушу мүмкүн жана андай циклдерди татаал циклдер деп атайбыз. Камтылуучу циклдерди колдонууда программаны ички цикли толугу менен ички циклдик бөлүмгө кире тургандай кылып түзүү зарыл. Ички цикл өз учурунда башка ички циклди кармап турушу да мүмкүн. Камтылуучу циклдердин структурасы төмөнкү мисалдарда каралат.

Мисал-1. $S = \sum_{j=1}^{4}(j+1)!$ суммасын эсептегиле.

Алгоритми:



Программасы:

```
program m_1;
   var
      j, t:integer;
      s, p: integer;
begin
      s:=0;
      for j:=1 to 4 do
      begin
            p:=1;
      for t:=1 to (j+1) do
      p:=p*t;
      s:=s+p;
      end;
      write('s=', s:8:3);
      readln;
end.
```

Мисал-2. $P=\prod_{i=1}^{7}\sum_{j=1}^{5}\frac{j}{i+1}$ көбөйтүндүсүнүн программасы боюнча алгоритмди жазгыла.

```
program m_2;
var
      i, j:integer;
      s, p:real;
begin
      p:=1;
      for i:=1 to 7 do
      begin
            s:=0;
            for j:=1 to 5 do
            s:=s+(j/(i+1));
      end;
      p:=p*s;
      write('p=', p:8:3);
readln;
end.
```

Варианттар

1. $P=\prod_{i=1}^{4}(i+2)!$
2. $S=\sum_{r=1}^{5}\frac{1+(R!)}{R^2}$
3. $P=\sum_{i=1}^{6}\sum_{j=1}^{4}(i^2+j)$
4. $S=\sum_{i=1}^{5}\prod_{j=1}^{4}\frac{1}{i+j^2}$
5. $P=\sum_{m=1}^{5}m!$
6. $M=\sum_{i=1}^{8}\sum_{j=1}^{6}\frac{i-j+1}{i+j}$
7. $Z=\prod_{i=1}^{7}\sum_{j=1}^{5}\frac{i}{j+1}$
8. $K=\sum_{i=1}^{9}\sum_{j=1}^{7}\sin(i^3+j^4)$
9. $P=\prod_{k=1}^{4}\sum_{i=1}^{6}\cos(i+k)$
10. $S=\sum_{i=1}^{7}\prod_{j=1}^{4}\frac{j}{2j+i}$
11. $Z=\sum_{i=1}^{6}(i+4)!$
12. $P=\sum_{i=1}^{4}\prod_{j=1}^{4}\frac{i(j+4)}{i+j^2}$
13. $K=\sum_{n=1}^{7}\sum_{k=1}^{7}\frac{4+n}{k}$
14. $P=\prod_{n=1}^{4}\sum_{k=1}^{5}\frac{n^2+k}{k+n}$
15. $S=\prod_{i=1}^{4}\sum_{j=1}^{5}(2i^2+j)$
16. $P=\sum_{i=1}^{5}\prod_{j=1}^{4}\frac{i(j+4)}{i+j^2}$
17. $K=\sum_{m=1}^{5}\frac{(2m)!+3}{m^2}$
18. $S=\prod_{i=1}^{4}\sum_{j=1}^{5}(\ln(i+j)-i)$
19. $P=\prod_{k=1}^{4}\sum_{i=1}^{3}\sqrt{i+k}$
20. $S=\sum_{i=1}^{3}\prod_{j=1}^{2}\ln(i+j)$

Текшерүү үчүн суроолор.

1. Камтылуучу циклдерди уюштуруунун негизги эрежелерин көрсөткүлө.
2. Ички циклден анын толук бүтүшүнө көз карандысыз чыгып кетүүгө болобу?
3. Сырткы жана ички цикл деп эмнени айтабыз?

Лабораториялык иш №9

Тема: ***Итерациялык циклдик структурадагы алгоритмдерди программалаштыруу.***

Иштин максаты - итерациялык циклдик структурадагы алгоритмдерди программалаштырууну үйрөнүү. Анын кайталоо саны менен берилген циклден айырмасын ажырата билүү.

Өз алдынча иштөө үчүн тапшырмалар

1. Итерациялык циклдерди уюштырууну, мындай циклдерди уюштуруу үчүн программалоо тилинин мүмкүнчүлүктөрүн үйрөнүү.
2. Тапшырма менен дал келген чечимдин алгоритмин иштеп чыгуу.
3. Маселенин чечиминин программасын түзүү.
4. Программанын аткарылышын компьютерден анализдеп берүү.

Итерациялык циклдин жардамы менен эсептөө көпчүлүк убактарда берилген тактык менен удаалаш жакындашуу жолу аркылуу жакындаштырып эсептөөлөрдө ишке ашырылат. Бул учурда циклдердин саны алдын-ала белгисиз болот. Ошондуктан for операторун колдонуу мүмкүн эмес. Мисалы чексиз катардын мүчөлөрүнүн суммасын эсептөө маселеси итерациялык циклди уюштырууну талап кылат.

Кээ бир учурда катардын мүчөсүн эсептөөдө туундуну келтирип чыгаруу усулун колдонуу максатка ылайыктуу жана убакытты үнөмдөйт. Бирок рекуренттик катыш аркылуу эсептөө усулу да эсептөөнү тездетет. Бул методдун колдонулушун мисалда көрөбүз.

Мисал-1. Төмөнкү чексиз катардын мүчөлөрүнүн суммасын x=0,1 чекитинде $E=10^{-4}$ тактык менен эсептөгиле: $S = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + ... + (-1)^n \frac{x^{2n-1}}{(2n-1)!} + ...$

Катардын n- жана n-1 - мүчөлөрүн алабыз:

$a_n = (-1)^n \frac{x^{2n-1}}{(2n-1)!}$, $a_{n-1} = (-1)^{n-1} \frac{x^{2n-3}}{(2n-3)!}$. Андан кийин n- жана n-1- мүчөлөрүнүн катышын табабыз:

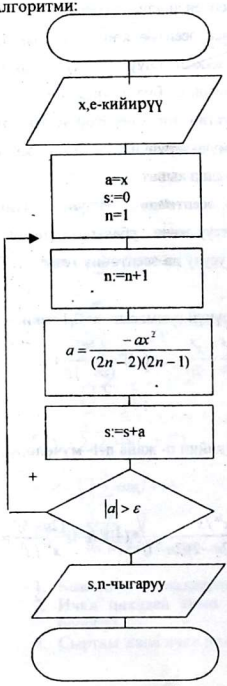
$$\frac{a_n}{a_{n-1}} = (-1)^n \frac{x^{2n-1}}{(2n-1)!} * (-1)^{n-1} \frac{(2n-3)!}{x^{2n-3}} = (-1)^n \frac{x^{2n}/x}{(2n-3)!(2n-2)(2n-1)} * (-1)^n / (-1) \frac{(2n-3)!}{x^{2n}/x^3} =$$

$$= -\frac{x^2}{(2n-2)(2n-1)}.$$

Мындан $a_n = a_{n-1}\frac{-x^2}{(2n-2)(2n-1)}; a_1 = x, n = 2,3,4,...$ экендигин алабыз. Ушул барабардык рекуренттик байланыш же рекуренттик формула деп аталат. Алгоритмди жазууда адатта индекстик өзгөрүлмөлөрдөн качуу үчүн катардын биринчи мүчөсүнүн маанисин циклге чейин a=x ыйгаруу оператору менен эсептеп, ал эми калган мүчөлөрүн циклде $a = -a\frac{x^2}{(2n-2)(2n-1)}$ рекуренттик катышы менен эсептөө ыңгайлуу.

Төмөндө катардын суммасын эсептөөнүн алгоритми жана программасы келтирилет.

Алгоритми:



Программасы:

```
program metod_1;
var
      x, eps, a, s:real;
      n:integer;
begin
      writeln('x, eps ду кийиргиле');
      read(x, eps);
      a:=x; s:=0; n:=1;
      while abs(a)>=eps do
       begin s:=s+a;
            n:=n+1;
      a:=-a*x*x/((2*n-2)*(2*n-1));
      end;
writeln('катардын суммасы s=', s:8:3);
writeln('катардын мүч-нүн саны n=', n:4);
readln;
end.
```

Варианттар

Тапшырма: x($x \neq 0$) бүтүн саны берилген, $\varepsilon = 10^{-4}$ тактыгы менен эсептөө талап кылынат.

1. $S = \sum_{n=0}^{\infty} (-1)^n \frac{(2x)^{2n}}{(2n)!}$
2. $S = \sum_{n=0}^{\infty} (-1)^{n+1} \frac{x^{2n+1}}{4n^2+1}$
3. $S = \sum_{n=0}^{\infty} \frac{x^{2n}}{(2n)!}$
4. $S = \sum_{n=0}^{\infty} (-1)^n \frac{1}{(2n+1)x^{2n+1}}$
5. $S = \sum_{n=0}^{\infty} \frac{x^{2n+1}}{(2n+1)!}$
6. $S = \sum_{n=0}^{\infty} (-1)^{n+1} \frac{x^{2n+1}}{4n^2-1}$
7. $y = \sum_{n=0}^{\infty} \frac{x^{2n}}{2^n R!}$
8. $K = \sum_{n=0}^{\infty} \frac{(-1)^n x^{2n-5}}{(2n+5)!}$
9. $S = \sum_{n=0}^{\infty} \frac{(-1)^n x^{3n+1}}{(n+1)!(3n+1)}$
10. $p = \sum_{n=0}^{\infty} \frac{(-1)^n x^{5n}}{(3n)!5n}$
11. $S = \sum_{n=0}^{\infty} \frac{(-x)^{2n}}{2n!}$
12. $M = \sum_{n=0}^{\infty} \frac{(-1)^n (n+1) x^n}{3n!}$
13. $T = \sum_{n=0}^{\infty} \frac{(-1)^n x^{n+2}}{(n+1)(n+2)!}$
14. $S = \sum_{n=0}^{\infty} \frac{(-1)^n x^n}{(2n+1)!}$
15. $p = \sum_{n=0}^{\infty} \frac{(-1)^n x^{3n}}{(n+1)(3n)!}$
16. $y = \sum_{n=0}^{\infty} \frac{(n+4)x^{n+3}}{(n+2)(n+3)!}$
17. $M = \sum_{n=1}^{\infty} \frac{(-1)^n x^{n+7}}{n(n+7)!}$
18. $S = \sum_{n=0}^{\infty} \frac{(-1)^n x^{3n+2}}{(3n)!(3n+2)}$
19. $y = \sum_{n=0}^{\infty} \frac{(-1)^n x^{2n+1}}{(2n+1)(n+4)!}$
20. $M = \sum_{n=0}^{\infty} \frac{(-1)^n x^{3n-1}}{(3n-1)(n+1)!}$

Текшерүү үчүн суроолор

1. Итерациялык циклдик процесс деген эмне?
2. Итерация методунун маңызы эмнеде?
3. Чексиз катардын суммасынын маанисин эсептөөдө циклден чыгуунун кандай шарты бар?
4. Программада чексиз катардын мүчөлөрүнүн суммасын эсептөөдө кайсы операторлор циклди уюштурат?

Лабораториялык иш №10

Тема: ***Массивдер. Бир ченемдүү массивдер менен иштөө.***

Иштин максаты - массивдер менен иштөөнү үйрөнүү, аларды кийирүү-чыгаруунун өзгөчөлүктөрүн билүү.

Өз алдынча иштөө үчүн тапшырмалар

1. Программалоо тилинде массивдердин өлчөмдөрүн баяндоонун ыкмаларын үйрөнүү.
2. Массивдерди кийирүү-чыгаруунун ыкмалары.
3. Тапшырма менен дал келген чечимдин алгоритмин иштеп чыгуу.
4. Маселенин чечиминин программасын түзүү.
5. Программанын аткарылышын компьютерден анализдеп берүү.

Массив катары бир типтеги жалпы атка ээ болгон иреттелген берилгендердин тобун түшүнүүгө болот. Массив деп аталуучу объектилер математикадагы вектор, удаалаштык, матрица, таблица түшүнүктөрү менен дал келет. Бул объектилер бир (мисалы, x_i) жана эки индекстүү (x_{ij}) өзгөрүлмөлөр менен белгиленгенин биз математикадан билебиз. Ошондуктан информатика курсунда, программалоо тилдеринде вектор, удаалаштык объектилери бир ченемдүү массивдер, ал эми матрица, таблица объектилери эки ченемдүү массивдер деп аталышат. Массивдин ар бир элементи индекстүү массивдин аты менен дайындалат. Массивдин элементтери индекстин мааниси боюнча иреттелген.

Паскаль тилинде индекс квадраттык кашаага жазылат. Эгерде программада массив колдонулса, анда ал var өзгөрүлмөлөр бөлүгүндө же type типтер бөлүгүндө баяндалышы керек.

Өзгөрүлмөлөр бөлүгүндө баяндоо.

Өзгөрүлмөлөрдү баяндоо бөлүгүндө бир ченемдүү массив төмөндөгүдөй баяндалат:

var

<массивдин аты>:array [n1 .. n2] **of <тип>;**

Мында array(массив), of(дан)-кызматчы сөздөр; n1..n2 - индекстердин диапазону(интервалдык же диапазондук тип дессек да болот); **тип** - Паскаль тилинде мүмкүн болгон массивдин элементтеринин тиби.

М., var a:array [1..7] of real;

Мында a-элементтеринин базалык тиби real болгон массивдин аты; индекси 1 ден 7 ге чейин өзгөрөт, б.а. 7 элементтен турган a деген аталышка ээ болгон бир ченемдүү массив баяндалды.

Эскертүү! Индекстин тиби стандарттык бүтүн жана чыныгы тип болбогондуктан төмөнкүдөй жазуу ката болуп эсептелет:

```
var
    b:array[6] of integer;
же var
    c:array[integer] of real;
```

Массивдерди туура жазуунун мисалдары:

```
1) var
    b:array[1..6] of real;
2) var
    c:array[1..100] of real;
```

Эгерде бир нече массивдер индекстердин бирдей диапазонуна (бир тибине) жана бирдей базалык типке ээ болушса, анда баяндоодо массивдерди бир тизмеге бириктирүүгө болот. Мисалы:

```
var a, b, c:array[1..5] of real;
```

Массивди типтер бөлүгүндө баяндоо:

type <типтин аты>=array [m1] **of m2;**

var <массивдин аты>:<типтин аты>;

Мында m1-индекстин тиби, m2-массивдин элементтеринин базалык тиби.

Мисалы, типтин атын massiv деп, 4 элементтен турган чыныгы типтеги a массивин колдонсок, анда аны баяндоо төмөндөгүдөй болот:

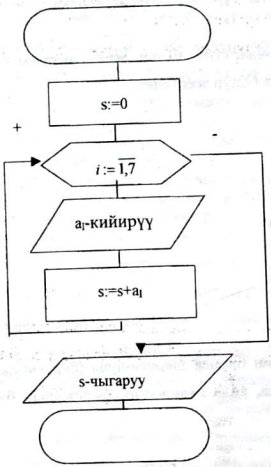
```
type massiv=array[1..4]  of real;
var a:massiv;
```

Эгерде программада бир нече массив болуп калса, анда өзгөрүлмөлөр бөлүгүндөй эле үтүр менен ажыратып, бир тизмеге бириктирип жазабыз:

```
var a,b,c:massiv;
```

Мисал-1. n элементтен турган бир ченемдүү a массивинин суммасын таап, жыйынтыгын печатка чыгаргыла, б.а. $s=a_1+a_2+a_3+\ldots+a_n$ суммасын эсептеп жыйынтыгын экранга чыгаргыла.

Алгоритми:



Программасы:

```
program massiv;
const n=7;
var
      i : integer;
      s : real;
      a : array[1..n] of real;
begin
      s:=0;
      for i:=1 to n do
      begin
            read(a[i]);
            s:=s+a[i];
      end;
  write('s=', s:8:3);
  readln;
  end.
```

Мисал - 2. Бир өлчөмдүү массивдин максималдык элементин жана анын индексин(катар номерин) издөөчү программа берилген. Бул программаны жазгыла.

```
program maximum_search;
const
     max=100;
var
      random_array:array[1..max] of integer;
      max_element, index, index_max:integer;
begin
      randomize;
for index:=1 to max do
      random_array[index]:=random(100);
max_element:=random_array[1];
for index:=2 to max do
if max_element<random_array[index] then
   begin
      max_element:=random_array[index];
      index_max:=index;
   end;
writeln('максималдык элементтин мааниси: ', max_element );
```

```
writeln( 'максималдык элементтин индекси:', index_max);
writeln('иштти аяктоо үчүн <Enter>ди баскыла);
readln;
end.
```

Вариаттар

Эсептегиле:

1. $S = \sum_{i=1}^{100} X_i$
2. $P = \prod_{j=1}^{80} A_j$
3. $y = \sum_{k=1}^{70} X_k$
4. $K = \sum_{n=1}^{50} B_n$
5. $Y = \sum_{i=1}^{5} \left(\sqrt{i} + Z_i\right)^3$
6. $Z = \sum_{i=1}^{12} \frac{x_i * y_i}{x_i + y_i};\ x_i, y_i \in R,\ x_i \neq -y_i$
7. $p = \prod_{i=1}^{6} (i - y_i)^3$
8. $Y = \sum_{k=1}^{7} a_k b_k$
9. $s = \prod_{i=1}^{5} \sqrt{a_i + b_i};\ a_i + b_i \geq 0$
10. $Y = \sum_{n=1}^{6} \frac{a_n}{b_n - 1};\ b_n \neq 1$
11. $C_k = \sum_{k=1}^{20} a_k + b_k$
12. $M = \sum_{n=1}^{30} X_n$
13. $D_i = \sum_{i=1}^{8} \frac{a_i + b_i}{c_i}$
14. $Y = \prod_{i=1}^{3} (a_i + c_i)$
15. $F = \frac{x_j^{j+1}(j+2)}{j}$
16. $M = \sum_{i=1}^{20} (x_i + i)$
17. $Y = \sum_{n=1}^{5} \frac{a_n}{n!}$
18. $S = \sum_{i=1}^{6} \ln(a_i);\ a_i > 0$
19. $Y = \prod_{i=1}^{5} (x_i - i);\ x_i \neq i$
20. $Y = \sum_{k=1}^{5} \frac{a_k + x_k}{a_k \cdot x_k};\ a_k \neq 0, x_k \neq 0$

Текшерүү үчүн суроолор

1. Массив колдонулган программанын өзгөчөлүгүн көрсөткүлө.
2. Массивдерди баяндоо үчүн тилдин кандай кызматчы сөздөрүн колдонууга болот?
3. Массивдерди кайра иштетүүдө циклди уюштуруунун өзгөчөлүгү эмнеде?
4. Массивдерди кийирүү жана чыгаруунун өзгөчөлүктөрүн көрсөткүлө.

Лабораториялык иш №11

Тема: ***Эки өлчөмдүү массивдер менен иштөө.***

Иштин максаты-эки өлчөмдүү массивдер менен иштөөнү үйрөнүү, аларды кийирүү-чыгаруунун өзгөчөлүктөрүн билүү.

Өз алдынча иштөө үчүн тапшырмалар

1. Программалоо тилинде массивдердин өлчөмдөрүн баяндоонун ыкмаларын үйрөнүү.
2. Эки өлчөмдүү массивдерди кийирүү-чыгаруунун ыкмалары.
3. Тапшырма менен дал келген чечимдин алгоритмин иштеп чыгуу.
4. Маселенин чечиминин программасын түзүү.
5. Программанын аткарылышын компьютерден анализдеп берүү.

Мурунку лабораториялык иште элементи бир гана индекстен турган массивди, б.а. бир өлчөмдүү массивди карадык. Эки өлчөмдүү массив катары матрицаны(таблицаны) түшүнүүгө болот. Мында деле массив бир атка ээ болот, бирок ар бир элемент өз-өзүнчө индекске ээ. Мисалы, ар түрдүү белгилеништеги a_{ij} же a(i, j) матрицасында биринчи индекс i – элемент жайланышкан жолчонун номерин, ал эми экинчи индекс j - мамычанын номерин түшүндүрөт.

Эки өлчөмдүү массив бир өлчөмдүү массив сыяктуу эле var өзгөрүлмөлөр бөлүгүндө же type типтер бөлүгүндө баяндалышы керек.

Өзгөрүлмөлөр бөлүгүндө баяндоо.

Өзгөрүлмөлөр бөлүгүндө эки өлчөмдүү массивди төмөндөгүдөй баяндоого болот:

var

<массивдин аты>: array [m1..m2, n1..n2] **of tip;**

Мында **array**(массив), of(дан) - кызматчы сөздөр; m1..m2 - жолчолордун диапазону(номерлери), n1..n2 – мамычалардын диапазону (номери); tip - Турбо-Паскаль тилинде мүмкүн болгон массивдин элементтеринин тиби.

Мисалы,

var a:array[1..7, 1..5] of real;

Мында 7 жолчодон, 5 мамычадан турган, элементтеринин базалык тиби real болгон, массивдин аты a болгон матрица баяндалды.

Эгерде бир нече массивдер индекстердин бир тибине жана бирдей базалык типке ээ болушса, анда баяндоодо массивдерди бир тизмеге бириктирүүгө болот. Мисалы:

var a,b,c:array[1..5, 1..5] of real;

Типтер бөлүгүндө баяндоо.

Эки ченемдүү массив типтер бөлүгүндө төмөндөгүдөй баяндалат:

type <типтин аты>=array [m1..m2, n1..n2] **of tip;**

var <массивдин аты>:<типтин аты>;

Мында m1..m2 - жолчонун номерлери, n1..n2 - мамычасынын номерлери, tip - массивдин элементтеринин базалык тиби. Мисалы:

1) type matrix =array[1..3, 1..4] of integer;

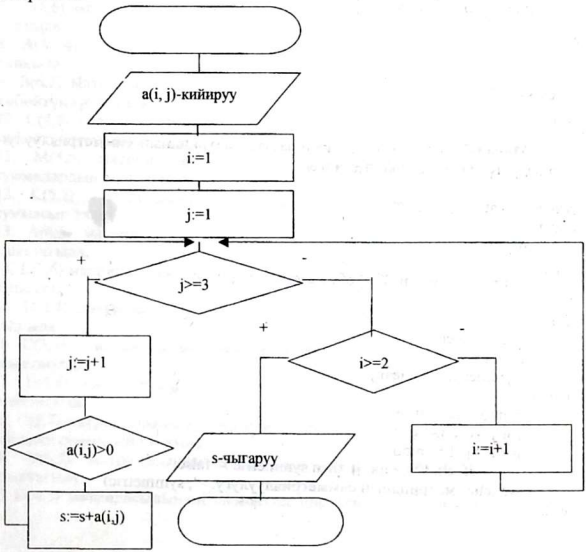
var a: matrix; же

2) type matrix =array[1..3] of array[1..4] of integer;

var a:matrix;

Мисал - 1. 2x3 өлчөмдөгү A массивинин оң элементтеринин суммасын тапкыла.

Алгоритми:



Программасы:

```
program massiv;
var
   i, j:integer;
   s:real;
   a:array[1..2, 1..3] of real;
begin
      for i:=1 to 2 do
begin
      for j:=1 to 3 do
      begin
      read(a[i, j]);
      if a[i, j]>0 then s:=s+a[i, j];
end;
end;
write('s=', s:8:3);
end.
```

Мисал-2. Бүтүн типтеги квадраттык матрицанын симметриялуулугун текшерүүчү программа келтирилген.

```
program test_for_symmetry;
const
      n=10;
type
      matrix=array[1..n, 1..n] of integer;

var
      j, k : integer;
      a : matrix;
      symmetric : boolean;
begin
      symmetric := true;
      for j:=1 to n-1 do
      for k:=j+1 to n do
            if a[j, k]<>a[k, j] then symmetric := false;
      writeln(' матрицанын симметриялуулугу: - ', symmetric)
end.
```

Варианттар

1. $C_i = \sum_{j=1}^{4} a_{ij} (i = 1,2,3)$ формуласын пайдаланып, а матрицасынын элементтеринин жолчолор боюнча суммасын тапкыла.

2. $X_c = (\sum_{i=1}^{n} m_i x_i)/(\sum_{i=1}^{n} m_i)$, $y_c = (\sum_{i=1}^{n} m_i y_i)/(\sum_{i=1}^{n} m_i)$ болсо, анда тегиздикте жайгашкан n материалдык чекиттин системасынын оордук борборлорунун координаталарын эсептөөнүн программасын түзгүлө.

3. a_1, a_2, a_3 бүтүн сандары берилген. $b_{ij}=a_i-3a_j$ боло тургандай $\{b_{ij}\}$ (i,j=1,2,3) матрицасын алуунун программасын түзгүлө.

4. $a_{ij} = \begin{cases} \sin(i+j), & \text{эгерде } i < j \\ 1, & \text{эгерде } i \geq j \end{cases}$ боло тургандай $\{a_{ij}\}$ $(i,j=\overline{1,n})$ матрицасын алуунун программасын түзгүлө.

5. n натуралдык саны жана $n \times 9$ элементтүү чыныгы матрицасынын жуп номерге ээ болуучу ар бир мамычанын арифметикалык орточосун табуунун алгоритмин түзгүлө.

6. A(7,6) матрицасынын ар бир мамычасынын максималдык элементтеринин көбөйтүндүсүн аныктагыла.

7. C(6,6) матрицасынын ар бир мамычасынын элементтеринин суммасын тапкыла.

8. A(4, 4) матрицасынын ар бир жолчосунун максималдык элементин тапкыла.

9. B(6,7) матрицасынын ар бир жолчосунун минималдык элементтеринин көбөйтүндүсүн тапкыла.

10. C(4,5) матрицасынын максималдык жана минималдык элементтеринин көбөйтүндүсүн аныктагыла.

11. M(5,5) матрицасынын ар бир жолчосунун суммасын жана бул суммалардын эң чоңун тапкыла.

12. K(5,7) матрицасынын ар бир мамычасынын көбөйтүндүсүн жана суммасын тапкыла.

13. A(6,6) матрицасынын минималдык жана максималдык элементтерин аныктагыла.

14. L(7,6) матрицасынын ар бир жолчосунун нөл эмес элементтеринин санын тапкыла.

15. D(5,4) матрицасынын ар бир жолчосундагы оң элементтеринин санын тапкыла.

16. C(3,4) матрицасынын элементтеринин суммасын жана көбөйтүндүсүн аныктагыла.

17. B(6,6) матрицасынын оң жана терс элементтеринин жалпы санын аныктагыла.

18. C(5,7) матрицасынын ар бир мамычасынын минималдык элементин жана алардын суммасын тапкыла.

19. B(6,7) матрицасынын терс жана нөлдүк элементтеринин санын аныктагыла.

20. D(4,5) матрицасынын оң элементтеринин суммасын тапкыла.

Текшерүү үчүн суроолор

1. Эки өлчөмдүү массив колдонулган программанын өзгөчөлүгүн көрсөткүлө.
2. Эки өлчөмдүү массивдерди баяндоо үчүн тилдин кандай операторлорун колдонууга болот?
3. Эки өлчөмдүү массивдерди кайра иштетүүдө циклди уюштуруунун өзгөчөлүгү эмнеде?
4. Эки өлчөмдүү массивдерди кийирүү жана чыгаруунун өзгөчөлүктөрүн көрсөткүлө.

Лабораториялык иш №12

Тема: ***Процедуралар.***

Иштин максаты: Түрдүү көрүнүштөгү жардамчы камтылуучу программаларды колдонуу менен программалоо жана алгоритмдештирүү ыкмаларына машыгуу. Камтылуучу программалардын жазылышын жана аларга кайрылуунун, параметрлерин тандоо ыкмаларын билүү. Колдонуучуларды өз алдынча жардамчы алгоритмдерди жана алардын программаларын түзүүгө үйрөтүү.

Өз алдынча иштөө үчүн тапшырмалар

1. Түрдүү көрүнүштөгү камтылуучу программалардын жазылышын жана аларга кайрылуунун жазылыш эрежеси.
   - Камтылуучу программага параметрлерди берүү ыкмасы.
   - Камтылуучу программа колдонулган программанын жазылуу эрежеси.
   - Камтылуучу программа колдонулган программанын аткарылыш эрежеси.
2. Берилген маселенин чечилишинин алгоритмин иштеп чыгуу.
3. Маселенин чечилишинин программасын түзүү.
4. Программанын жыйынтыгын компьютерден анализдеп берүү.

Кээ бир маселелердин программаларын түзүүдө бир эле операторлордун удаалаштыгын (курама оператор сыяктуу) бир нече жолу кайталап жазууга туура келет. Ошондуктан ыңгайсыз болгон кайталоолордун ордуна операторлордун удаалаштыгын(курама операторду) бир программалык бирдик катары кароого жана ага негизги программалык бөлүктөн(курама операторго ат берип жана аты боюнча) кайрылууга болот. Мындай өз алдынча болгон программалык бирдик

программалоо тилдеринде жардамчы (камтылган) программа деп аталат. Паскаль тилинде жардамчы программалар катары процедуралар жана функциялар колдонулушат.

Процедуралардын үч түрү бар:

1. Параметрсиз процедура;
2. Параметрлери-өзгөрүлмө процедура;
3. Параметрлери -маани процедура.

Процедура негизги программанын структурасына окшош структурага ээ. Айырмачылыгын төмөнкү баяндоолордон жеңил эле байкоого болот (процедуранын бөркүн(заголовка) жазууда program деген кызматчы сөздүн ордуна procedure кызматчы сөзү жазылат). Программада процедуранын атын оператор катары колдонуу ***процедурага кайрылуу*** же ***процедуранын оператору*** деп аталат.

1. Параметрсиз процедуралар.

Параметрсиз процедуранын форматы:

procedure <аты>;

баяндоо бөлүгү;

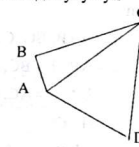
begin

оператордук бөлүк

end;

Параметрсиз процедуранын баяндалышын жана колдонулушун төмөндөгү мисалдан көрсөк болот.

Мисал-1. Жактары AB, BC, CD, AD жана диагоналы AC болгон томпок ABCD төрт бурчтугунун аянтын эсептөө талап кылынсын.



Чыгаруу:

Томпок төрт бурчтукту анын диагоналы эки үч бурчтукка бөлөт. Ошондуктан төрт бурчтуктун аянты эки үч бурчтуктун аянттарынын суммасынан турат. Ал эми үч бурчтуктун аянты Геррондун формуласы боюнча эсептелет:

$$S = \sqrt{p(p-a)(p-b)(p-c)}\ ,$$

мында a, b, c- үч бурчтуктун жактарынын узундуктары, p=(a+b+c)/2 – жарым периметр. Демек, үч бурчтуктун аянтын эсептөөчү процедураны түзүү максатка ылайыктуу. Төмөндөгү процедура үч бурчтуктун аянтын эсептейт:

```
procedure str1;
  var
    a, b, c, p, s : real;
begin
            p:=(a+b+c)/2;
            s:=sqrt(p*(p-a)*(p-b)*(p-c));
end;
```

Төрт бурчтуктун аянтын эсептөөнүн программасынын бир вариантын жазабыз:

```
program S_ABCD1;
var
      AB, BC, CD, DA, AC,  s1,s2,s,a,b,c:real;
procedure str1;
      var p: real;
      begin
            p:=(a+b+c);
            s:=sqrt(p*(p-a)*(p-b)*(p-c));
      end;
begin
      read(AB, BC, CD, DA, AC);
      a:=AB;  b:=BC; c:=AC;  str1;  s1:=s;
      a:=DA; b:=AC; c:=CD; str1;  s1:=s;
      s:=s1+s2;
      writeln(' төрт бурчтуктун аянты =', s:10:3)
end.
```

Программада str1 процедурасына эки жолу кайрылуу болду. Үзгүлтүксүз ар бир кайрылуунун алдында a, b, c өзгөрүлмөлөрүнө маанилерди берүү ыйгаруу операторлору аркылуу ишке ашырылды. Ар бир кайрылуу процедуранын аткарылышын камсыз кылат. Процедурага кайрылгандан кийин s өзгөрүлмөсүнүн мааниси үч бурчтуктун аянтына барабар болгон мааниге ээ болот. Программада дагы р өзгөрүлмөсү бар

жана ал процедураның ичинде жардамчы болуп эсептелет. Жардамчы өзгөрүлмөлөр негизги программанын өзгөрүлмөлөрү менен дал келбеши керек. Процедурада баяндалган өзгөрүлмө негизги программага карата локалдык деп эсептелет. «Локалдык» деген сөз арналышына карата жергиликтүү дегенди билдирет жана р өзгөрүлмөсү локалдык өзгөрүлмө деп аталат. Процедура аткарылып бүткөндө локалдык өзгөрүлмөнүн мааниси унутулат б.а. «жашоосун токтотот» деп эсептелет. Ошондуктан негизги программалык бөлүктөн локалдык өзгөрүлмөлөр менен иш алып барганда этият болуш керек.

*2. **Параметрлери-өзгөрүлмөлөр процедуралары.***

Параметри-өзгөрүлмө процедураның форматы:

procedure <аты>(var q_1:T_1; var q_2:T_2; …; var q_n:T_n);

<камтылуучу программалардын жана

локалдык параметрлердин баяндалышы жана аныктоо бөлүгү>

begin

P_1;

P_2;

.

. (оператордук бөлүк)

.

P_m

end;

мында, q_i-формалдуу параметрлердин тизмеси; T_i –параметрлердин тиби (i=1,…,n), P_j (j=1,…,m)–камтылуучу программанын телосунун операторлору.

Мисал катары параметрсиз процедуранын ордуна параметри-өзгөрүлмө процедураны төмөндөгүдөй баяндасак болот:

```
procedure str2(var a,b,c,s:real);
   var
      p:real;
begin
   p:=(a+b+c)/2;
   s:=sqrt(p*(p-a)*(p-b)*(p-c))
end;
```

Процедуранын атынан кийин кашаанын ичинде көрсөтүлгөн a, b, c, s параметрлери - формалдуу параметрлер. Бул процедурага негизги программадан **str2(AB,BC, AC, s1)** – деп кайрылууга мүмкүн. Бул кайрылуу AB, BC, AC жактары боюнча үч бурчтуктун аянтын эсептөөнү камсыз кылат жана s1 өзгөрүлмөсүнө аянттын маанисин ыйгаруу ишке ашырылат. Мында AB, BC, AC, s1 – чыныгы(фактические) параметрлер болушат. Программанын аткарылышында процедурага кайрылуу убагында формалдык параметрлер чыныгы параметрлер менен алмаштырылат: биринчи формалдуу параметр тиешелеш түрдө биринчи чыныгы параметр менен алмаштырылат, экинчи формалдуу параметр тиешелеш түрдө экинчи чыныгы параметр менен алмаштырылат ж.б.у.с. Алмаштыруу бүткөндөн кийин процедура аткарылат. Жогорудагы айтылгандарды төрт бурчтуктун аянтын эсептөөнүн программасынын дагы бир вариантынан көрсөк болот:

```
program S_ABCD2;
var
      AB, BC, CD, DA, AC,s1,s2:real;
procedure str2(var a,b,c,s:real);
      var
            p:real;
      begin
            p:=(a+b+c)/2;
            s:=sqrt(p*(p-a)*(p-b)*(p-c))
      end;
begin
      read(AB, BC,  CD, DA, AC);
      str3(AB, BC, AC, s1);
      str3(CD,  DA, AC, s2);
      writeln(s1+s2)
end.
```

Мында, процедурага биринчи кайрылуу төмөнкү операторлорунун аткарылышына алып келет:

```
begin
      p:=(AB+BC+AC)/2;
      s1:=sqrt(p*(p-AB)*(p-BC)*(p-AC))
```

```
end;
экинчи кайрылууда
 begin
      p:=(CD+DA+AC)/2;
      s2:=sqrt(p*(p-CD)*(p-DA)*(p-AC))
end;
```

операторлорунун аткарылышын камсыз кылат.

3. Параметрлери-маанилер процедуралары.

Процедуранын бөркү (заголовкасы) кээ бир формалдуу параметрлердин группаларында var кызматчы сөзү колдонулбай түзүлүшү мүмкүн.Мисалы, procedure str4(a, b, c:real; var s:real); procedure prim(r:integer; var p:integer); ж.б.у.с. **«var»** кызматчы сөзүн кармабаган группага кошулуучу формалдуу параметрлер ***формалдуу параметрлүү-маанилер*** деп аталат. Биринчи мисалда a, b, c-бул формалдуу параметрлүү-маанилер, ал эми S-бул формалдуу өзгөрүлмө-параметр. Экинчи мисалда r-бул формалдуу маани-параметр, ал эми p-бул формалдуу өзгөрүлмө-параметр.

Мындай формалдуу параметр-мааниге туура келүүчү фактылык параметр (real, integer же char тиби көрсөтүлсө) өзгөрүлмө эле болбостон, аларга туура келүүчү фактылык параметрлер каалагандай туюнтма болушу мүмкүн. Мисалы жогоруда каралган str2 процедурасына кайрылуу төмөндөгүдей параметрлер менен ишке ашырылышы мүмкүн:

7str2(3.14, x, sqrt(10+sqr(x)), y).

Эгерде str3 процедурасынын жарыяланышы болуп төмөнкү кызмат кылса:

```
procedure str3(a, b,  c:real; var s:real);
var
      p:real;
begin
      p:=(a+b+c)/2;
      s:=sqrt(p*(p-a)*(p-b)*(p-c))
end;
```

анда str4(3.14, x, sqrt(10+sqr(x)), y) кайрылуусу төмөндөгү курама оператордун аткарылышына алып келет:

```
begin
```

```
    a:=3.14; b:=x; c:=sqrt(10+sqr(x));
    p:=(a+b+c)/2;
    y:=sqrt(p*(p-a)*(p-b)*(p-c))
end;
```

Параметрлери-өзгөрүлмө процедура үчүн келтирилген мисалдын программасын жогорудагы айтылгандарды эске алып төмөндөгүдөй кылып оңой эле өзгөртүп жазууга болот:

```
program S_ABCD3;
var
    AB, BC, CD, DA, AC,s1,s2:real;
procedure str3( a,b,c: real; var s : real);
    var
      p:real;
  begin
      p:=(a+b+c)/2;
      s:=sqrt(p*(p-a)*(p-b)*(p-c))
  end;
begin
    read(AB, BC, CD, DA, AC); x:=BC;
    str3(3.14, x, sqrt(10+sqr(x)), s1);
    str3(CD+5, DA*AC, AC, s2);
    writeln(s1+s2)
end.
```

Вариаттар

1. $s,t \in R$.

$g(a,b)=\dfrac{a^2+b^2}{a^2+3ab+3b^2+4}$ болгондо $g(1.2-s)+g(t,s)-g(2s-1,st)$ - ны алуунун программасын жазгыла.

2. $s,t \in R$. $f(a,b,c)=\dfrac{2a-b-\sin c}{5+|c|}$ болгондо f(t, -2c, 1.17)+f(2.2, t, s-t) -ти алуунун программасын жазгыла.

3. $y \in R$. $t(x) = \dfrac{\sum\limits_{r=0}^{10} \dfrac{x^{2r+1}}{(2r+1)!}}{\sum\limits_{r=0}^{10} \dfrac{x^{2r}}{(2r)!}}$ болгондо $\dfrac{1.7t(0.25)+2t(1+y)}{6-t(y^2-1)}$ -ти алуунун программасын жазгыла.

4. $a,b,c \in R$. Эсептегиле: $\dfrac{\max(a,a+b)+\max(a,b+c)}{1+\max(a+bc,1.15)}$.

5. $a,b \in R$. Тапкыла: $\upsilon = \min(a,b)$, $\vartheta = \min(ab,a+b)$, $\min(\upsilon+\vartheta^2, 3.14)$.

6. $n,m \in N$, $a_1,\ldots, a_n$, $b_1,\ldots, b_m$, $c_1,\ldots, c_{30}$ бүтүн сандары берилген. Эсептегиле:

$$l = \begin{cases} \min(b_1,\ldots,b_m)+\min(c_1,\ldots,c_{30}), & \text{эгерде } \left|\min(a_1,\ldots,a_n)\right| > 0 \\ 1+(\max(c_1,\ldots,c_{30}))^2, & \text{тескери учурда} \end{cases}$$

7. $r,l,m \in N$, $x_1,\ldots, x_n$, $y_1,\ldots, y_l$, $z_1,\ldots, z_m$ бүтүн сандары берилген. Эсептегиле:

$$t = \begin{cases} (\max(y_1,\ldots,y_l)+\max(z_1,\ldots,z_m))/2, & \text{эгерде } \max(x_1,\ldots,x_r) \geq 0 \\ 1+(\max(x_1,\ldots,x_r))^2, & \text{тескери учурда} \end{cases}$$

8. s, t чыныгы сандары берилген. $h(a,b) = \dfrac{a}{1+b^2} + \dfrac{b}{1+a^2} - (a-b)^3$ болгондо h(s, t)+max(h^2(s-t, st), h^4(s-t, s+t))+h(1, 1) – туюнтмаларынын маанисин эсептегиле.

9. $a_0,\ldots, a_6$ чыныгы сандары берилген. $p(y)=a_6y^6+a_5y^5+\ldots+a_0$ болгон учурда x=1, 3, 4 үчүн p(x+1)-p(x) – тин маанисин табуу керек.

10. s, t, $a_0,\ldots, a_{12}$ чыныгы сандары берилген. $p(x)=a_{12}y^{12}+a_{11}y^{11}+\ldots+a_0$ болгон учурда $p\text{-}p(t)+p^2(s\text{-}t)\text{-}p^3$ - туюнтмасынын маанисин эсептегиле.

11. n натуралдык саны берилген. n , n+1, …, 2n сандарынын ичинен айырмасы экиге барабар болгон жөнөкөй сандарды, б.а. түгөйлөрдү аныктагыла.

12. Процедураны колдонуп, A(8, 10) матрицасынын ар бир мамычасынын орточо геометриялык оң элементтерин эсептөө үчүн программасын түзгүлө.

13. $n \in N$. $x,y,a_n,b_n,a_{n-1},b_{n-1},\ldots,a_0,b_0$.Комплекстик коэффициент менен көп мүчөнүн маанисин Горнердин схемасы боюнча эсептегиле.

$(a_n+ib_n)(x+iy)^n+(a_{n-1}+ib_{n-1})(x+iy)^{n-1}+\ldots+(a_0+ib_0)$.(Процедураны комплекстик сандардын үстүнөн жүргүзүлгөн амалдардын аткарылышы менен аныктагыла).

14. Натуралдык n, чыныгы $a_1,\ldots, a_{30}$. Эгерде $x=a_1{}^*a_2{}^*\ldots a_n$,

$y=a_{n+1}*a_{n+2}*\ldots a_{2n}$,

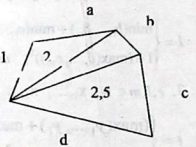
$z=a_{2n+1}*a_{2n+2}*\ldots a_{3n}$.

болсо $x+y^2+z^3$ эсептегиле.

15. x_1, y_1, x_2, y_2, …, x_{10} , y_{10} чыныгы сандары берилген. Чокулары тиешелүү (x_1, y_1), (x_2, y_2), …, (x_{10} , y_{10}) координаттарына ээ болгон он бурчтуктун периметрин тапкыла. Өзүнүн координаттары берилген эки чекиттин арасындагы аралыкты эсептөөнү процедурада баяндагыла.

16. c, d чыныгы сандары берилген. u=max(c, d), v=max(cd, c+d), max(u+v^2, 3.14) табуу керек.

17. a, b, c, d чыныгы сандары берилген. Сүрөттө көрсөтүлгөн беш бурчтуктун аянтын тапкыла. Үч бурчтуктун аянтын үч жагы боюнча эсептөөнүн процедурасын аныктагыла.



18. n натуралдык саны берилген. n, n+1, … , 2n сандарынын ичинен айырмасы 2ге барабар болгон сандарды тапкыла. Жөнөкөй сандарды табуунун процедурасын аныктагыла.

19. n натуралдык саны x_1, y_1, x_2, y_2, …, x_n , y_n чыныгы сандары берилген. Чокулары (x_1, y_1), (x_2, y_2), …, (x_n , y_n) координаттарына ээ болгон n-бурчтуктун аянтын тапкыла. Үч бурчтуктун чокулары боюнча аянтын эсептөөнүн процедурасын аныктагыла.

20. Жыйынтыгы чыныгы мааниге ээ болгон процедураны түзгүлө, эгерде процедурага кайрылгандагы символ тамга болсо(тескери учурда жалган мааниге ээ болот).

Текшерүү үчүн суроолор

1. Камтылуучу программага параметрлерди берүү ыкмасы.
2. Параметрсиз камтылуучу программаны кантип уюштурууга болот?
3. Камтылуучу программадан чыгуу кантип жана кайда ишке ашырылат?
4. Камтылуучу программанын түрдүү көрүнүштөрүнүн айырмасы кандай?

Лабораториялык иш №13

Тема: **Функциялар.**

Иштин максаты: Түрдүү көрүнүштөгү камтылуучу программаларды колдонуу менен программалоого жана алгоритмдештирүүнүн ыкмаларына машыгуу. Камтылуучу программалардын жазылышын жана аларга кайрылуунун, параметрлерин тандоо ыкмаларын билүү. Колдонуучуну өз алдынча жардамчы алгоритмдерди жана функцияны түзө билүүгө үйрөтүү.

Өз алдынча иштөө үчүн тапшырмалар

5. Түрдүү көрүнүштөгү камтылуучу программалардын жазылышын жана аларга кайрылуунун жазылыш эрежеси.
   - Камтылуучу программага параметрлерди берүү ыкмасы.
   - Камтылуучу программа колдонулган программанын жазылуу эрежеси.
   - Камтылуучу программа колдонулган программанын аткарылыш эрежеси.
6. Берилген маселенин чечилишинин алгоритмин иштеп чыгуу.
7. Маселенин чечилишинин программасын түзүү.
8. Программанын чыгарылышын компьютерден анализдеп берүү.

Функциялар бир маанини эсептөө үчүн колдонулат. Аны жазуунун жалпы форматы:

function f (q_1:t_1; q_2:t_2;…):t;

<камтылуучу программалардын жана локалдык параметрлердин баяндалышы жана аныктоо бөлүгү>

begin

p_1;

p_2;

.

.

.

p_n

end;

мында, f-функциянын аты, q_i-формалдуу параметрлердин аттары; t-функциянын атынын тиби, t_i –параметрлердин тиби, p_i –функциянын телосунун операторлору.

Функцияга кайрылуу **f(b_1, b_2,…)** көрүнүшүндөгү процедуранын оператору менен ишке ашырылат, ыйгаруу операторунун оң жагына

жазылат, мында f-камтылуучу программанын аты, b_i-фактылык параметрлердин аттары.

Мисал, функцияны колдонуп n!, m!, (n-m)! факториалдарынын маанисин аныктоо үчүн программасын түзгүлө.

```
program factl(input, output);
var
      nf, mf, nmf:integer;
function fact(k:integer):integer;
var
      pf, i:integer;
begin
if k<0 then fact:=0
      else if k=0  then fact:=1
      else
            begin
                  pf:=1;
                  for i:=2 to k  do
                  pf:=pf*i;
                  fact:=pf
            end;
begin
      read(n, m);
      nf:=fact(n);
      mf:=fact(m);
      nmf:=fact(n-m);
      write('nf=',nf,' ':4, 'mf=','mf,' ':4,  'nmf=',nmf)
end.
```

Программада ыйгаруу операторлорунда функцияга 3 жолу кайрылуу жазылган: fact(n), fact(m), fact(n-m).

Паскалда процедура менен катар анын конструкциясына окшош функция бар. Функцияга кайрылуу анын маанисин эсептөөгө алып келет-real, integer же char тибиндеги объекттер. Маанинин тиби функциянын жарыяланышында фиксирленет. Эгерде кээ бир функциянын мааниси (мисалы f(a, b)) real же integer тибине ээ болсо, анда бул функцияны колдонуу

ыкмасы sin, cos, round ж.б. функцияларының колдонуу ыкмаларына аналогиялуу.

Программада ыйгаруу операторлору кезигиши мүмкүн, мисалы x:=x+f(x,y), y:=sin(f(x, x)/2) же циклдүү оператор while f(x, y)<f(y, x) do x:=f(x, x).

Эгерде кээ бир функциянын мааниси (мисалы h(a) функциясы) char тибине ээ болсо, анда бул функция программада тырмакчага алынган конкреттүү символдор менен барабар колдонулушу мүмкүн. Программада мисалы, ыйгаруу оператору колдонулушу мүмкүн s:=h(i), шарттуу оператор If s=h(i) then s:='a' else s:=h(i) ж.б.

Функция процедурадан айырмаланып, баяндоонун эки өзгөчөлүгүнө ээ:биринчи өзгөчөлүк бөрк менен байланышкан: ал function-функция деген кызматчы сөзү менен башталышы керек жана функциянын мааниси ээ болгон тип менен аякташ керек.Мисалы,

```
function f(a:real; var b:t):real;
function g(var a, b:integer):integer;
function h(a:integer):char; ж.б.
```

Экинчи өзгөчөлүк бөрктөн кийин функциянын жарыяланышында жайгашкан составдуу оператор, локалдуу меткалардын же өзгөрүлмөлөрдөн кийин « := » дын сол жагында функциянын аты жайгашкан сөзсүз түрдө өз ичинде ыйгаруу операторун кармаш керек.

```
f:=3.14
g:=a+2*b
if a mod 2=0 then h:='*' else h:='!' ж.б.
```

Мындай ыйгаруу операторлордон бир топ болушу мүмкүн, бирок ар бир конкреттүү функцияга кайрылууда алардын ичинен бирөө гана «иштеши» керек. Бул ыйгаруу функциянын мааниси аныктайт. Салыштырмалуу функциянын параметри процедуранын параметри менен толук аналогиялуу. Функциянын баяндалышы процедуранын баяндалышындай эле программада өзгөрүлмөлөрдүн удаалаштыгы жарыяланган кийин жайгашат.

Процедурада каралган төрт бурчтуктун аянтын табууну функциянын жардамында төмөндөгүдөй эсептөөгө болот.

```
program F3(input, output);
var
    AB, BC, CD, DA, AC:real;
```

```
function triangl(a,b,c:real):real;
var p:real;
begin
    p:=(a+b+c)/2;
    triangl:=sqrt(p*(p-a)*(p-b)*(p-c))
end;
begin
read(AB, BC, CD, DA, AC);
write(triangl(AB, BC, AC)+triangl(CD, DA, AC))
end.
```

Варианттар

1. $s,t \in R$. $g(a,b) = \dfrac{a^2+b^2}{a^2+3ab+3b^2+4}$ болгондо $g(1.2-s)+g(t,s)-g(2s-1,st)$ ны алуу керек.

2. $s,t \in R$. $f(a,b,c) = \dfrac{2a-b-\sin c}{5+|c|}$ болгондо f(t, -2c, 1.17)+f(2.2, t, s-t) ти алуу керек.

3. $y \in R$. $f(x) = \dfrac{\frac{x}{1!}+\frac{x^3}{3!}+\frac{x^5}{5!}+\frac{x^7}{7!}}{\frac{x^2}{2!}+\frac{x^4}{4!}+\frac{x^6}{6!}+\frac{x^8}{8!}}$ болгондо $\dfrac{1.7f(0.25)+2f(1+y)}{6-f(y^2-1)}$ ти табуу керек.

4. $a,b,c \in R$. Төмөндөгүнү алуу керек: $\dfrac{\max(a,a+b)+\max(a,b+c)}{1+\max(a+bc,1.15)}$

5. $a,b \in R$.Табуу керек: $\upsilon = \min(a,b)$, $\vartheta = \min(ab,a+b)$, $\min(\upsilon+\vartheta^2,3.14)$.

6. Ачка жүрүү дарылануусунан 30 күндө пациенттин салмагы 96 кгдан 60 кг га төмөндөгөн. Күнүмдүк салмакты жоготуу телонун салмагына пропорционалдуу экендиги аныкталган. 2-3-,…, 29-күндөрү пациенттин салмагы канчага барабар болгондугун аныктоо керек.

Көрсөтмө: ln функциясынын жардамында пропорционалдуулук коэффициентин аныктап, андан кийин exp функциясын колдонуу керек.

7. f1, f2, f3, f4 төрт символдук файлдары берилген.Компоненттеринин саны көбүрөөк болгон файлды чыгаруу керек.(эгерде мындай файлдар бирден көп болсо, анда алардын бирөөсүн гана чыгаруу керек).Программада мааниси

файлдын компонентинин санына барабар болгон L(f) функциясын баяндоо керек.

8. $s = \sqrt{x^2 + y^2 + \sin^2 xy} + \sqrt{y^2 + z^2 + \sin^2 yz} + \sqrt{z^2 + x^2 + \sin^2 zx}$ функциясын колдонуп, функциянын маанисин эсептөөнүн программасын түзгүлө.

9. Функцияны камтылуучу программа катары колдонуу менен

$a = \frac{\sqrt{sz^3 + qz^2 + tz + t}}{1 + e^{\pi^2 + z - 1}}$; $b = \frac{t^2 \sin\alpha + t\cos\beta + 3{,}5}{z^2 + 2r - t}$ функциялардын маанисин эсептөөнүн программасын түзгүлө.

10. $n! = \begin{cases} 0, & эгер \quad n < 0; \\ 1, & эгер \quad n = 0; \\ n!, & эгер \quad n > 0. \end{cases}$ функциясын камтылуучу программа катары колдонуп,

с нын маанисин эсептегиле. $c = \frac{n!}{m!(n-m)!}$

11. Функцияны камтылуучу программа катары колдонуу менен X(60), Y(75), Z(80) массивдеринин орточо арифметикалык оң элементтерин тапкыла.Массивдер оң элементтерге ээ.

12. $SUM = \sum_{i=r}^{l} mas_i$ функциясын колдонуп, $y=ax^2+bx+d$ функциясынын маанисин эсептегиле. $a = \sum_{i=1}^{n_r} t_i; b = \sum_{i=n_r+1}^{100} t_i; d = \sum_{i=1}^{20} q_i$

13. а) катеттеринин узундугу б) гипотенузанын жана катеттин узундугу менен берилген тик бурчтуу үч бурчтуктун тик бурчунун маанисин тапкыла.

Көрсөтмө: **arctan** функциясын колдонгула.

14. $primer(a) = \begin{cases} 1, & эгер \quad a - жоноко\ddot{и} \quad сан \\ 0, & тескери \quad учурда \end{cases}$ функциясын колдонуп сандардын арасынан түгөйлөрдү издөөнүн программасын түзгүлө.

15. Катеттердин жана гипотенузанын узундугу берилген тик бурчтуу үч бурчтуктун тик бурчтарынын маанисин тапкыла. Arctan функциясын пайдаланғыла.

16. $x_1, y_1, x_2, y_2, \ldots, x_{10}, y_{10}$ чыныгы сандары берилген. Чокулары тиешелүү (x_1, y_1), (x_2, y_2), …, (x_{10}, y_{10}) координаттарына ээ болгон он бурчтуктун периметрин тапкыла. x1, y1 жана x2, y2 координатасына ээ болгон эки чекиттин арасындагы аралыкты d(x1, y1, x2, y2) функциясын колдонуп эсептегиле.

17. Бүтүн 10x10 өлчөмдүү матрицаны жолчосу боюнча сандардын маанилерин өсүү тартибинде сорттогула. Алгач матрицаны 0 дөн 100 гө чейинки диапазондо бүтүн кокустук сандар менен толтургула жана функцияны колдонуп чыгаргыла.

18. Так 0дөн 9 га чейинки бардык сандардын факториалдарынын суммасын эсептөөнү функцияны колдонуп чыгаргыла.

19. X, Y, Z, T сандары берилген-төрт бурчтуктун жактары. Анын аянтын функцияны колдонуп эсептегиле, эгерде X жана Y узундуктагы жактарынын ортосундагы бурч түз болсо.

20. A(N) массиви берилген (N-жуп). А массивинде катар турган элементтеринин жубу орточо арифметикалык сандар болсо B(M) массивинин элементтерин тапкыла. Мисал, А массиви 1, 3, 5, -2, 0, 4, 0, 3 болсо, В массивинин элементтери 2; 1.5; 2; 1, 5 болот.

Текшерүү үчүн суроолор

5. Функциянын процедурадан айырмачылыктары.
6. Камтылуучу программага параметрлерди берүү ыкмасы.
7. Камтылуучу программадан чыгуу кантип жана кайда ишке ашырылат?
8. Камтылуучу программанын түрдүү көрүнүштөрүнүн айырмасы кандай?

Лабораториялык иш № 14

Тема: ***Файлдар менен иштөө.***

Иштин максаты-Турбо-Паскалда файлдар менен иштөөнүн өзгөчөлүктөрүн билүү. Студенттерге Паскалда файлдар менен иштөөнүн бир топ мүмкүнчүлүктөрүн үйрөтүү.

Өз алдынча иштөө үчүн тапшырмалар

1. Типтештирилген файлдар.
2. Типтештирилбеген файлдар.
3. Тексттик файлдар.

Файлдар менен иштөө үчүн файлдык өзгөрүлмөлөр колдонулат. Файлдык өзгөрүлмөлөр 3 типте болушат: **типтештирилген, типтештирилбеген, тексттик.**

Типтештирилген өзгөрүлмөлөр керек, эгерде файлды бир жана ошол эле типтеги компоненттердин удаалаштыгы катары кароо ыңгайлуу болсо. Типтештирилген файлдык өзгөрүлмөлөрдү баяндоо үчүн «file of» деп жазуу зарыл жана файлдын компонентинин тибин көрсөтүү керек.

Мисалдар:

```
x:file of real;
```

y:file of char;
z:file of string[15];

Типтештирилбеген өзгөрүлмөлөр файлды берилгендердин жалпы патогу катары кароо ыңгайлуу болгон учурда талап кылынат. Типтештирилбеген файлдык өзгөрүлмөлөрдү баяндоо үчүн «file» кызматчы сөзү жазылат.Мисалдар:

```
picture:file;
file1,file2:file;
```

Тексттик өзгөрүлмөлөр тексттик файлдар менен иштөө үчүн колдонулушу мүмкүн, б.а.аныкталбаган узундуктагы жолчолордон турган файлдар.Тексттик өзгөрүлмөлөрдү баяндоо үчүн «text» кызматчы сөзү жазылат.Мисалдар:

```
book:text;
txt1,txt2:text;
```

Файлдык өзгөрүлмөлөрдү колдонуунун алдынан аны **Assign** процедурасынын жардамында дисктик файл менен байланыштыруу зарыл.Assign процедурасын чакыруруда параметр катары өзгөрүлмөнүн аты жана файлдын толук аты көрсөтүлөт.Мисалы:

```
Assign(picture, 'c:\bp\bin\mydata')
Assign(book, 'd:\tp\readme');
Assign(chapter,'e:\book\5.pas');
```

Файлдык өзгөрүлмө конкреттүү файл менен байланышкандан кийин файл Reset же Rewrite процедураларынын жардамында ачылышы керек.

```
Reset(Picture);
Rewrite(Txt1);
Reset(Chapter);
```

Reset процедурасы бар файлды ачат. **Rewrite** процедурасы Assign процедурасын чакырууда көрсөтүлгөн ат менен жаңы файл түзөт. Эгерде мындай аттуу файл бар болсо, анда ал жокко чыгарылат да, анын ордуна жаңы бош файл түзүлөт.

Эгерде мурда ачылган файл программада колдонулбаса, анда аны **Close** процедурасы менен жабуу талап кылынат.

```
Close(Picture);
Close(Txt1);
Close(Chapter);
```

Файлдар менен иштөө үчүн төмөнкү фундаменталдык операцияларды аткаруу керек:

- берилгендердин блогун файлдын берилген ордунан оперативдүү эске эсептөө керек;
- берилгендердин блогун оперативдүү эстен файлдын берилген ордуна жазуу.

Бул идеология көбүнчө типтештирилбеген файлдар менен иштөөдө жакшы байкалат.

Типтештирилбеген файлдар менен иштөө үчүн процедуралар жана функциялар.

BlockRead процедурасы файлдан жазуулардын берилген санын оперативдүү эстин көрсөтүлгөн ордуна окуйт.

Баяндалышы: **BlockRead(var F; var V; N:word);**

Мында, F-файлдык өзгөрүлмө, V-эсептелген информация жайгаша турган эстеги адрес, N-эсептелиши керек болгон жазуулардын саны.

Жазуунун узундугу файлды ачууда аныкталат.
BlockWrite процедурасы оперативдүү эстин көрсөтүлгөн ордунан жазуулардын берилген санын файлга жазат.
Баяндалышы: **BlockWrite(var F:file; var V; N:word);**
Мында F-файлдык өзгөрүлмө, V-информация жайгаша турган эстеги адрес, N-файлга жазылышы керек болгон жазуулардын саны.
FilePos функциясы жазууларда файлдын учурдагы позициясын кайтарып берет. Баяндалышы: **FilePos(var F:file):LongInt;**
FileSize функциясы жазууларда файлдын өлчөмүн кайтарып берет.
Баяндалышы: **FileSize(var F:file):LongInt;**
Жазуунун узундугу файлды ачууда аныкталат.
Reset процедурасы дискте бар болгон файлды ачат. Учурдагы позиция файлдын башына орнотулат. Баяндалышы: **Reset(var F:file[; Size:word])**
F-файлдык өзгөрүлмө, Size-файл ачылуучу жазуунун узундугу. Эгерде жазуунун узундугу көрсөтүлбөсө 128 байтка барабар деп кабыл алынат.
Rewrite процедурасы файлды ачат жана түзөт. Эгерде файл бар болсо, анда ал жокко чыгарылат да, жаңыдан түзүлөт.
Баяндалышы: **Rewrite(var F[; Size:word]);**
F-файлдык өзгөрүлмө, Size-файл ачылуучу жазуунун узундугу. Эгерде жазуунун узундугу көрсөтүлбөсө, анда ал 128 байтка барабар деп кабыл алынат.
Seek процедурасы файлда көрсөткүчтү берилген жазууга позициялайт.
Баяндалышы: **Seek(var F; L:LongInt);**
F-файлдык өзгөрүлмө, L-көрсөткүч позицияланган жазуунун узундугу.
Мисал. 60 Кбайтка чейинки файлдарды көчүрүү.

```
program m_1;
var
      Buf:array[1..60000] of byte;
      FIn, FOut:file;
begin
Assign(FIn,ParamStr(1));
Assign(FOut, ParamStr(2));
Reset(FIn, 1);
Rewrite(FOut,1);
BlockRead(FIn,Buf,FileSize(FIn));
BlockWrite(FOut,Buf,FileSize(FIn));
Writeln('файл', ParamStr(1),'файлга көчүрүлгөн', ParamStr(2));
Close(FIn);
Close(FOut);
readln;
end.
```

Типтештирилген файлдар менен иштөө үчүн процедуралар жана функциялар.
Read процедурасы файлдан көрсөтүлгөн өзгөрүлмөлөрдүн маанисин окуйт.
Баяндалышы: **Read(F; X1[,X2[,X3[..XN]]]);**
Мында F-типтештирилген файлдык өзгөрүлмө. X1,...XN-файлдын компоненттери катары ошол эле типтин өзгөрүлмөлөрү.
Write процедурасы файлга көрсөтүлгөн өзгөрүлмөнүн маанисин жазат.
Баяндалышы: **Write(F, X1[,X2[,X3[...XN]]]);**
Мында, типтештирилген файлдык өзгөрүлмө, X1,..XN-файлдын компоненти катары ошол эле типтин өзгөрүлмөлөрү.

FilePos функциясы файлдын учурдагы компонентинин иреттүү номерин кайтарып берет. Баяндалышы: **FilePos(F):LongInt;**

FileSize функциясы файлда компоненттердин санын кайтарып берет. Баяндалышы: **FileSize(F):LongInt;**

Reset процедурасы дискте бар файлды ачат. Учурдагы позиция файлдын башында орнотулат. Баяндалышы: **Reset(F);**

Мында F-файлдык өзгөрүлмө.

Rewrite процедурасы файлды ачат жана түзөт. Эгерде файл бар болсо, анда ал жокко чыгарылат жана жаңыдан түзүлөт. Баяндалышы: **Rewrite(F);**

F-файлдык өзгөрүлмө.

Seek процедурасы файлдын берилген компонентине көрсөткүчтү позициялайт. Баяндалышы: **Seek(var F; L:LongInt);**

F-файлдык өзгөрүлмө, L-көрсөткүч позициялануучу компоненттин номери.

Тексттик файлдар менен иштөө үчүн процедуралар жана функциялар.

Read процедурасы файлдан көрсөтүлгөн өзгөрүлмөлөрдүн маанисин окуйт. Баяндалышы: **Read(var F:text; X1[,X2[,X3[..XN]]]);**

Мында, F-типтештирилген файлдык өзгөрүлмө, X1,..XN-файлдын компоненти катары ошол эле типтин өзгөрүлмөлөрү.

Readln процедурасы файлдан көрсөтүлгөн өзгөрүлмөлөрдүн маанисин окуйт. Баяндалышы: **Readln(var F:text; X1[,X2[,X3[..XN]]]);**

Мында F-типтештирилген файлдык өзгөрүлмө, X1,..XN-файлдын компоненттери сыяктуу ошол эле типтин өзгөрүлмөлөрү.

Write процедурасы файлга көрсөтүлгөн өзгөрүлмөлөрдүн маанисин жазат. Баяндалышы: **Write(var F:text; X1[,X2[,X3[..XN]]]);**

Мында, F-типтештирилген файлдык өзгөрүлмө, X1,..XN-файлдын компоненттери сыяктуу ошол эле типтин өзгөрүлмөлөрү.

Writeln процедурасы файлга көрсөтүлгөн өзгөрүлмөлөрдүн маанисин жазат. Баяндалышы: **Writeln(var F:text; X1[,X2[,X3[..XN]]]);**

Мында, F-типтештирилген файлдык өзгөрүлмө, X1,..XN-файлдын компоненттери сыяктуу ошол эле типтин өзгөрүлмөлөрү.

Reset процедурасы дисктеги бар файлды ачат. Учурдагы позиция файлдын башына орнотулат. Баяндалышы: **Reset(var F:text);**

Мында, F-файлдык өзгөрүлмө.

Rewrite процедурасы файлды ачат жана түзөт. Эгерде файл бар болсо, анда ал жокко чыгарылат жана жаңыдан түзүлөт. Баяндалышы: **Rewrite(F:text);**

Мында, F-файлдык өзгөрүлмө.

Мисал.Студенттердин сессиясын ташырганды жөнүндө маалыматты кармап турган файлды түзүүнүн программасын түзгүлө. Жазуунун структурасы мындай талааларды кармап турат: группанын индекси, студенттин фамилиясы, беш экзамен боюнча баалар.

Талаанын жарыяланышы	Өзгөрүлмөнүн аты	Берилгендердин тиби	Мисал
Группанын индекси	INDEX	6 символ	ИУ2-11
Фамилия	FAM	20 символ	Асанов А.
Беш экзамен боюнча баалар	Массив MARKER(5)	ар бир элемент бир символдуу	54323

Идентификаторлор жана талаалардын узундугу таблицада келтирилген.

```
program zapis;
type
      zap=record
            index:string[6];
            fam:string[20];
            marcer:packed array[1..5] of char;
      end;
var
      sessya:file of zap;
      k:integer;
      flag:boolean;
      x:zap;
begin
      k:=0; flag:=true;
rewrite(sessya);
repeat
      writeln('учурдагы жазууну кийиргиле: группанын индекси, фамилиясы
жана');
      writeln('экзамендер боюнча беш баа');
      writeln('ишти аяктоо үчүн индекстин ордуна кийиргиле');
      writeln('группалар символдор ######');
read(x,index,x.fam,x.marcer[1],x.marcer[2],x.marcer[3], x.marcer[4], x.marcer[5]);
if x.index<> '######' then
                  begin
                        k:=k+1;
                        write(sessya, x);
                  end
                  else
                        flag:=false
until flag;
writeln('файлда ',' жазуу бар');
assign(sessya,'sessya');
reset(sessya);
while not eof(sessya) do
      begin
            read(sessya,x);
            writeln(x.index,x.fam,x.marcer[1],      x.marcer[2],x.marcer[3],
x.marcer[4], x.marcer[5]);
      end;
writeln('файлдын аягы');
readln;
end.
```

Варианттар

1. ONE файлын түзгүлө, анын ар бир элементин пробел аркылуу көбөйтүп эсептегиле.
2. TWO файлында «а» тамгасын эки жолу көбөйтүп эсептегиле.
3. Сандардан турган файлдын элементтеринин суммасын тапкыла.
4. Символдордон турган файл берилген. Латын алфавитинин "А" тамгасы бул файлда канча жолу кезигерин тапкыла.
5. Жолчолордон турган файл берилген, андагы эң узун жолчонун позициясын тапкыла.
6. Жолчолордон турган файл берилген, андагы эң кыска жолчонун позициясын тапкыла.
7. Файл түзгүлө, "." символу бул файлда канчанчы позицияда жайгашкандыгын таап, биринчи «.»тен кийинки тексттерди өчүргүлө..
8. ONE файлын окуп, андагы кездешкен цифралардын суммасын тапкыла.
9. Символдордон турган файл берилген. "." символдорунун ордуна "!" символдоруна алмаштыргыла.
10. Бүтүн сандардын файлы берилген, андагы эң чоң санды тапкыла.
11. Бүтүн сандардын файлы берилген, андагы эң кичине санды тапкыла.
12. Жазуулардан турган файл түзгүлө, жазууда студенттин Ф.А.А., туулган жылы, телефону болсун.
13. Жазуулардан турган файл түзгүлө, жазууда ОшМУнун факультеттери жана кафедралары болсун.
14. Сандардан турган файл берилген. Бул сандарды файлга өсүү тартибинде сорттоп жазгыла.
15. Сандардан турган файл берилген. Бул сандарды файлга кемүү тартибинде сорттоп жазгыла.
16. Жазуулардан турган файл түзүлсүн. Жазууда ИТАС кафедрасынын окутуучуларынын Ф.А.А. жазылсын.
17. Символдордон турган файл берилген, андагы акыркы « , » символунун позициясын тапкыла.
18. Типтештирилген файл берилген. Андагы жазуулардын санын тапкыла.
19. Студенттер жөнүндө маалыматтардан турган файл түзгүлө.
20. Эки жолчолук файлдар берилген. Бул файлдардагы окшош жолчолорду тапкыла.

Текшерүүчү суроолор

1. ТРда файл деген эмне ?
2. Файлдын канча түрү бар жана жалпы жазылыштары кайсылар ?
3. Типтештирилген, типтештирилбеген, тексттик файлдар кандай түзүлөт ?
4. Типтештирилген файлдык өзгөрүлмө кайсы учурда колдонулат?
5. Типтештирилбеген файлдык өзгөрүлмө кайсы учурда колдонулат?
6. Тексттик файлдык өзгөрүлмө кайсы учурда колдонулат?

Лабораториялык иш №15

Тема: ***Көптүктөр.***

Иштин максаты-көптүктөр менен иштөөнү үйрөнүү, Паскалда көптүктөрдүн үстүнөн жүргүзүлүүчү амалдарды билүү.

Өз алдынча иштөө үчүн тапшырмалар

1. Көптүктөрдү баяндоонун ыкмаларын үйрөнүү.
2. Тапшырма менен дал келген чечимдин алгоритмин иштеп чыгуу.
3. Маселенин чечиминин программасын түзүү.
5. Программанын аткарылышын компьютерден анализдеп берүү.

Турбо-Паскалда берилгендердин тиби жөнөкөй жана татаал болуп бөлүнүшөт. Жөнөкөй типке стандарттык, эсептелүүчү, чектелген, ал эми татаал типке массивдер, көптүктөр, жазуулар жана файлдар кирет. Татаал типти структураланган тип деп да айтабыз. Татаал типтин элементтери болуп жөнөкөй типтер, о.э.өз учурунда татаал типтер да болушу мүмкүн. Татаал типке киришүү программалоо тилин бир топ кубаттуу кылат жана жогорку маселелер классында эффективдүү программаларды түзүүгө мүмкүндүк берет.

def: **Көптүк** деп, бири-бири менен логикалык байланышта болгон чектелген, иретсиз бир типтеги түрдүү элементтердин чогуусун айтабыз.

Көптүктүн массивден, жазуудан айырмасы: элементтеринин санынын туруктуу эместиги жана ал элементтер түрдүү типке таандыгы. Көптүктөр var өзгөрүлмөлөр бөлүгүндө, type типтер бөлүгүндө жарыяланышы керек.

Көптүктү өзгөрүлмөлөр бөлүгүндө жарыялоо:

var

<көптүктүн аты>:set of <базалык тип>;

M., var

жыл:set of 2000...2004

c:set of char;

Көптүктү типтер бөлүгүндө жарыялоо:

type

<типтин аты>=set of <базалык тип>;

var <көптүктүн аты>:<типтин аты>;

M., type

number char=set of '0'...'9';

number=set of 0..9;

end;

var

S1, S2, S3:number char;

S4, S5, S6:number;

S1:=['1', '2','3'];

S2:=['2','3'];

S3:=['3','2','1'];

S5:=['1','2','3','4','5'];

Set-көптүк д.т. Көптүккө кирген элементтердин тиби **базалык тип** д.а. Базалык тип катары жөнөкөй типти колдонууга болот, чыныгы, чектелген, эсептелүүчү типтеринен башка каалаган ТРдын типтери кирет. Көптүккө кирген элементтердин саны 256дан ашпас керек.

Элементтеринин саны 0 болгон көптүк бош көптүк д.а.
Көптүктөр тууңду обьекттердин жыйындысы. Турбо Паскалда көптүктөрдүн максималдык элементи 256 элементтен турат, ал эми минималдык элемент-бош көптүк.
Көптүктөрдүн үстүнөн жүргүзүлүүчү амалдар(биригүү, кесилишүү, айырма).
1) **Бириктирүү амалы** C=A∪B(C көптүгүнүн ар бир элементи же А көптүгүнүн элементи же В көптүгүнүн элементи б.э.).
2) **Кесилишүү амалы** C=A∩B(C көптүгүнүн ар бир элементи бир эле учурда А көптүгүнүн да В көптүгүнүн да элементи б.э.).
3) **Айырма** C=A/B(C көптүгүнүн ар бир элементи А көптүгүнүн элементи б.э., бирок В көптүгүнүн элементи болуп эсептелбейт).
Көптүктөрдүн элементтери иреттелген эмес, ошондуктан {1, 5, 4} {4, 5, 1} {5, 4,1} бул көптүктөр бирдей б.э. Бардык көптүккө бир бүтүн ат берилет.

Турбо Паскаль тилинде көптүктөрдүн үстүнөн төмөнкү операциялар жүргүзүлөт:
a) +-көптүктөрдүн биригүүсү;
b) *-көптүктөрдүн кесилишүүсү;
c) --көптүктөрдүн айырмасы;
d) =, <>-көптүктөрдү барабардыкка жана барабарсыздыкка текшерүү.

А көптүгү В көптүгүнө барабар болот, эгерде А көптүгүнүн ар бир элементи В көптүгүнүн элементи б.э. жана тескерисинче А жана В көптүктөрү бири-бирине барабар эмес болушат.
e)<=, >=-көптүктөрдүн камтылышын текшерүү. А көптүгү В көптүгүнө камтылган (A<=B же B>=A), эгерде А көптүгүнүн бардык элементтери В көптүгүнүн элементтери болуп эсептелишсе. Жыйынтыгы чын(true) болот.
IN-элементтин көптүктө жатарын текшерүү. Бул амал (x in Y) x базалык тибинин элементи Y көптүгүндө жатарын текшерүү үчүн колдонулат.
Мисалы,

```
uses crt;
var
      ch:char;
      X:set of char;
begin
      repeat
      ch:=readkey;
      write(ch);
      include(ch,X);
      until ch=#27;
      if « A» in X then writeln('кирет') else writeln('кирбейт');
readln;
end.
```

Мисал. Латын алфавити менен көптүктүн кокустук көптүкчөсүн түзүүнүн программасы.

```
program random_subsets;
uses crt;
type
      char_set=set of char;
var
      S:char_set;
      i, j, k:integer;
```

```
        first, last, ch, stop:char;
procedure print(name:string; S:char_set);
var
        ch:char;
begin
        if name =' ' then write('[') else write(name. '=[ ');
        for ch:=#0 to #255 do
        begin
        if ch in S then
        begin
                if (ch<=' ') or (ch=#255) then write('Chr('.ord(ch).')') else write(ch);
                S:=S-[ch];
                if S=[ ] then break;
                write('.');
        end;
end;
writeln(']');
end;
begin
        first:='A';     last:='Z';
        S:=[first..last];
        print('көптүк S:=', S);
        Randomize; k:=5;
        writeln(',k,' элементтен туруучу', 'S кокустук көптүкчөсү');
        for i:=1 to 20 do
        begin
        stop:=last;
        for j:=1 to k-1 do dec(stop);
        j:=ord(stop)-ord(first)+1;
        S:=[ ];
        while stop <=last do
        begin
                ch:=Chr(ord(first)+random(j));
                if ch in S then S:=S+[stop] else S:=S+[ch];
                inc(stop); inc(j);
        end;
        print(' ',S);
end;
writeln('ишти аяктоо үчүн <Enter>ди баскыла');
readln; end.
```

Варианттар

1. Символдордун көптүгү берилген. Бул көптүктө чоң «A» латын тамгасы бар же жок экендигин аныктагыла.
2. X, Y көптүктөрү берилген. Алардын биригүүсүн табуу керек.
3. A жана B көптүктөрү берилген. Бул көптүктөрдүн кесилишүүсүн тапкыла.
4. C жана D көптүктөрү берилген. Көбөйтүндүсүн тапкыла.
5. X, A көптүктөрү берилген. A көптүгү X көптүгүнө камтылуучу көптүк экендигин текшергиле.
6. Z1, Z2,...,ZN символдуу көптүк берилген. Бул символдордун арасынан канчасы пробел экендигин аныктагыла.
7. Y1, Y2,..., YN символдуу көптүк берилген. Символдордун арасынан цифраларды бөлүп алып, аларды санга айландырып, өсүү тартибинде жайгаштыргыла.
8. X1, X2,...,XN символдуу көптүк берилген, ал көптүктүн элементтеринин арасынан «c» тамгасынын санын тапкыла.
9. A1, A2,...,AN символдуу көптүк берилген, ал көптүктүн элементтеринин арасынан «d» символун таап, анын кодун тапкыла.
10. C1, C2,...,CN сандуу көптүк. Бул көптүктү тең экиге бөлүп, биринчи бөлүгүнөн «1» санынын жана экинчи бөлүгүнөн «7» санынын кездешүүсүн тапкыла.
11. A жана B көптүктөрүн салыштыргыла.
12. C1, C2,...,CN сандуу көптүк берилген. C санынын бул көптүккө тиешелүүлүгүн текшергиле.
13. D1, D2,...,DN символдуу көптүк берилген. Бул көптүктүн элементтеринин арасынан «.»(чекит) символунун санын тапкыла.
14. Y1, Y2, Y3, Y4 көптүктөрү берилген. Төмөнкүнү аткаргыла: (Y1+Y4)*(Y3-Y2)<=Y1
15. A, B, C, D, E көптүктөрү берилген. Төмөнкүнү аткаргыла:
(A+C)-C*(D-A)+E
16. X, Y, Z, T көптүктөрү берилген. Алардын арасынан кайсынысы бош көптүк экендигин текшергиле.
17. M1, M2, M3 көптүктөрү берилген. Анда «b» тамгасы бар же жок экендигин текшергиле.
18. X1, X2, X3, X4, X5 көптүктөрү берилген. X1*X2-X5+x4*X3<>X5 аткаргыла.
19. D1, D2,..., DN символдуу көптүк берилсе, анын элементтеринин арасынан «K» тамгасынын бар же жок экендигин текшергиле.
20. A, B, C, D, E, F, L көптүктөрү берилген. Төмөнкүнү аткаргыла:
(B*C-A)*E+(D-F)<=L

Текшерүү үчүн суроолор

1. Көптүк деген эмне?
2. Көптүктү өзгөрүлмөлөр бөлүгүндө кантип жарыялайбыз?
3. Көптүктү типтер бөлүгүндө кантип жарыялайбыз?
4. Көптүктүн массивден, жазуудан айырмасы эмнеде?

Лабораториялык иш №16

Тема: ***Жазуулар.***

Иштин максаты-Паскалда жазуулар менен иштөөнүн мүмкүнчүлүктөрүн үйрөнүү.

Өз алдынча иштөө үчүн тапшырмалар

1. Жазууларды баяндоонун ыкмаларын үйрөнүү.
2. Тапшырма менен дал келген чечимдин алгоритмин иштеп чыгуу.
3. Масселенин чечиминин программасын түзүү.
5. Программанын аткарылышын компьютерден анализдеп берүү.

Турбо-Паскалда берилгендердин тиби жөнөкөй жана татаал болуп бөлүнүшөт. Жөнөкөй типке стандарттык, эсептелүүчү, чектелген, ал эми татаал типке массивдер, көптүктөр, жазуулар жана файлдар кирет. Татаал типти структураланган тип деп да айтабыз. Татаал типтин элементтери болуп жөнөкөй типтер, о.э.өз учурунда татаал типтер да болушу мүмкүн. Татаал типке киришүү программалоо тилин бир топ кубаттуу кылат жана жогорку маселелер классында эффективдүү программаларды түзүүгө мүмкүндүк берет.

Берилгендердин группасы менен иштөө үчүн Паскалда жазуу түшүнүгү киргизилген.

Def: **Жазуу** -бул талаа деп аталуучу, фиксирленген сандагы, түрдүү типтеги элементтерден турган компоненттерди кармаган берилгендердин структурасы аталат.

Жазуулар var өзгөрүлмөлөр бөлүгүндө, type типтер бөлүгүндө жарыяланышы керек.

Жазууну өзгөрүлмөлөр бөлүгүндө жарыялоо:

```
VAR
     <жазуунун аты>:RECORD
          <1-элементтин аты>:<тиби>;
          ...
          <n-элементтин аты>:<тиби>;
END;
```

M, var stud:record
    name:string[10];
    gr:boolean;
    nz:93100..93500;
  end;

Жазууну типтер бөлүгүндө жарыялоо:

```
TYPE
     <типтин аты>=RECORD
          <1-элементтин аты>:<тиби>;
          ...
          <n-элементтин аты>:<тиби>;
     END;
VAR
     <жазуунун аты>:<типтин аты>;
```

Record-жазуу д.т. Бул кызматчы сөз ачылуучу кашаанын ролун аткарат, end-жабылуучу кашаа б.э.

```
M,type tab=record
      a,b:real;
   end;
   var
      c:tab;
      d:record
      e,f:integer;
  end;
```

Фиксирленген жазуулар-бул фиксирленген элементтердин санынан турган туруктуу түзүлүүчү структура. Мында жазуунун элементтери каалаган типте болушу мүмкүн, ошонун ичинде көптүктөр, файлдар, башка жазуулар да болуп калышы ыктымал.

Жазууну баяндоонун формасы: **type t=record талаалардын тизмеси end;**

талаалардын тизмеси-бул жазуунун элементтерин аныктоонун удаалаштыгы; талааларды аныктоо идентификатор же үтүр менен ажыратылган талаанын идентификаторлорунун тизмеси болушу мүмкүн.

Мисалы, а) type tab=record
```
                a,b:real;
              end;
b) var c:tab;
     d:record
       e,f:integer;
       end;
```

With-бириктирүү оператору. Бириктирүү оператору жазуунун элементтерине кайрылуунун жөнөкөйлөтүү үчүн арналган, тагыраак айтканда жазуунун элементтеринин жөнөкөй аттарынын жардамында жазуунун элементтерине кайрылуунун мүмкүнчүлүктөрүн камсыздоо үчүн арналган.

With операторунун формасы: **With жазуулардын аттарынын тизмеси do S;**

Мында, S-жөнөкөй же составдык оператор.

With(бирге) көбүнчө жазуунун бир же бир нече талааларынын үстүнөн бир нече аракеттерди аткарган учурда колдонулат.

М., with Y[i] do if den>15 then write('айдын экинчи жарымы'); Бул төмөнкү жазууга эквиваленттүү if Y[i].den>15 then write('айдын экинчи жарымы');

Варианттуу жазуулар. Паскалда варианттуу жазууларды колдонууга болот-варианттуу талаалары менен өзгөрүлмө структурадагы жазуулар. Мындай жазуунун талааларынын тизмесинде фиксирленген талаалар жана бир гана өзгөрүлмө(варианттуу) талаа болушу мүмкүн. Жазуунун варианттуу бөлүгү баяндоонун аягында, б.а. акыркы элемент катары болушу керек. Жазуунун баяндалышында жазуунун өзгөрүлмө талаасынын бардык мүмкүн болгон варианттары эсептелет. Жазуу менен иштөөдө жазуунун селекторунан(ачкычынан) маанисинен көз каранды жазуунун өзгөрүлмө талааларынан бир варианты тандалат. Жазуунун өзгөрүлмө бөлүгү төмөнкү эки форманын биринде көрсөтүлөт.

а) type sel=тип;...жазуунун фиксирленген бөлүгүнүн баяндалышы
```
case sel of
      метка-1:(талаалардын тизмеси);
      метка-2:(талаалардын тизмеси);...
      end;
```
b) type sel=тип;...жазуунун фиксирленген бөлүгүнүн баяндалышы
```
case pr:sel of
```

```
        метка-1:(талаалардын тизмеси);
        метка-2:(талаалардын тизмеси);...
        end;
```

Sel-рг ачкычынын тибинин аты жана вариантты тандоо үчүн меткалар.

рг-вариантты тандоо селекторунда өзгөрүлмөнүн аты, ал жазуунун варианттык бөлүгүн колдонгонго чейин эле маани алышы керек. SEL тиби өзгөрүлмөнүн эсептелүүчү тибин аныкташы керек.

Эскертүү! 1) Жазуунун баяндалышында бир гана варианттык бөлүк болушу керек; 2) бир жазуунун талааларынын тизмесинде бардык идентификаторлор түрдүү болушу керек, бирок бир жазуунун талааларынын камтылуусунун түрдүү даражаларында жана түрдүү жазууларда аттар кайталанышы мүмкүн; 3) жазуунун варианттык бөлүгү анын акыркы элементи болушу керек.

Мисал(1). Жеке адамдын жашын, фамилиясын, жынысын жазууну колдонуп чыгаргыла.

```
program exemple_1;
type
        person=record
        name:string[30];
        sex:char;{жынысы}
        age:byte; {жашы}
end;
var
        man:person;
        number:byte;
begin
        repeat
        writeln('вариантты тандагыла:');
        writeln('1-берилгендерди кийирүү');
        writeln('2-берилгендерди чыгаруу');
        writeln('3-программаны бүтүрүү');
        readln(number);
case number of
1: begin
        write('Ф.А.А.:');    readln(man.name);
repeat
        write('жынысы(эркек, аял):');    readln(man.sex);
        until man.sex in['М','м','Ж','ж'];
        write('жашы(жашта):');
        readln(man.age);
end;
2:begin
writeln;writeln('Ф.А.А.:', man.name);
write('жынысы:');
if man.sex in ['М','м'] then writeln('мужской') else writeln('женский');
writeln('жашы:',man.age,' жашта');
writeln;
end;
end;
until not (number in[1,2])
readln;  end.
```

Варианттар

1. Абитуриенттер тапшырган 3 кирүү экзаменди жазуунун структурасын колдонуп түзгүлө. Экзаменден өтүү үчүн 12 балл топтоо керек болчу. Бардык кирүү экзамендеринин жыйынтыгын эсептөөчү жана экранга төмөнкү маалыматты чыгаруучу программаны түзгүлө:
   а) үч экзаменди тең «5» ке тапшырган абитуриенттердин тизмеси;
   б) экзаменден «2» алган абитуриенттердин тизмеси;
   в) үч экзаменди тең «4» кө тапшырган абитуриенттердин тизмеси.
2. Товардын аты, коду, баасы жөнүндө маалыматты кармаган жазуунун программасын түзгүлө. Жазууну аныктаган өзгөрүлмөнү Tovar деп атагыла.
   - товардын коду(integer тибинде)
   - товардын аталышы(string тибинде)
   - баасы(real тибинде).
3. Жазууну аныктоочу өзгөрүлмөнү Inf деп атап, студенттердин маалыматтар базасын кармаган Baza тибиндеги ББнан студенттин маалыматтарын төмөнкү талаалар боюнча аныктоонун программасын түзгүлө.
   - студенттин жеке номери(integer тибинде);
   - ФАА(string тибинде);
   - туулган жылы(integer тиби);
   - адрес(string тиби).
4. Информатика боюнча өзүнөрдүн группанардагы студенттердин алган бааларын аныктоочу жазуунун программасын түзгүлө. Жазууну Geometr тиби менен төмөнкү талаалары боюнча аныктагыла. Жазууну аныктоочу өзгөрүлмөнү Dig деп атагыла.
   - ФАА(string тибинде);
   - 9 ай ичинде алган баалар, айын max 20 баадан.
5. Sport аттуу тип менен, эң жакшы спортсмендер тууралуу информацияны кармаган программаны түзгүлө:
   - спорттун түрү(string тиби);
   - рекордсмендин фамилиясы(string тиби);
   - рекордду орнотуу датасы(Day, Month, Year талааларынан турган Dat жазуусу);
   - жыйынтыгы тууралуу маалымат(real).

   Жазууну аныктоочу өзгөрүлмөнү Rec деп атагыла.
6. Data аттуу тип менен 30 күн үчүн үйдүн орточо температурасы жөнүндө маалыматты кармаган жазууну түзүүнүн программасын түзгүлө.
   - айдын номери(integer тиби);
   - температура(real тиби);

Жазууну аныктоочу өзгөрүлмөнү Zamer деп атагыла. Мисалы, июль айы жана биринчи күн үчүн температура-9, 5 ^{0}C.

7. 40 чекиттен графика түзүү үчүн зарыл болгон берилгендерден турган Graf аттуу тип менен жазууну түзүүнүн программасын түзгүлө:
   - графиканын аталышы(string тиби);
   - 40 маани (integer тиби).

Жазууну аныктоочу өзгөрүлмөнү X деп атагыла. With операторунун жардамында жазуунун талааларына төмөнкү мааниларди ыйгаргыла: графиканын аталышы 'Y:=f(T)', биринчи үч чекиттин мааниси: 5, 7, 9.

8. Оорулуулардын базасында стационардык дарыланган оорулуулар тууралуу маалыматты кармаган Boln аттуу тип менен жазууну баяндоонун программасын түзгүлө.
- ФАА(string тиби);
- жашы(integer тиби);
- адрес(string тиби);
- ооруканага жаткан датасы(string тиби);

9. Силердин шаарыңардагы электропоезддин кыймылы жөнүндө маалыматты кармаган Rasp аттуу тип менен жазууну баяндоонун программасын түзгүлө:
- багыты(string тиби);
- электропоезддин жөнөө убактысы(real тиби).

Жазууну аныктоочу өзгөрүлмөнү R деп атагыла.

10. Почталык берилгендер базасында газета жана журналга жазылуучулар жөнүндө информацияны кармаган жазууну Post аттуу тип менен баяндоонун программасын түзгүлө.
- ФАА(string тиби);
- адрес(string тиби);
- газета жана журналдардын аталышы менен 10 жолчо;

Жазууну аныктоочу өзгөрүлмөнү G деп атагыла.

11. Төмөнкү талаалардан турган Karta аттуу тип менен жазууну баяндагыла.
- өлчөө бирдиги(integer тиби);
- мааниси(real тиби);

Жазууну аныктоочу өзгөрүлмөнү Z деп атагыла.

12. Doc типтүү аттуу менен төмөнкү талаалардан турган жазууну түзгүлө:
- документтин жолчосунун номери(integer тиби);
- жолчонун тексти(string тиби);

Жазууну аныктоочу өзгөрүлмөнү S деп атагыла.

13. Systema аттуу тип менен планеталар жөнүндө информацияны кармаган жазууну баяндагыла:
- планетанын аталышы(string тиби);
- көлөмү(real);
- диаметр(real);
- Жерден алыстыгы(real тиби).

Жазууну аныктоочу өзгөрүлмөнү Planeta деп атагыла.

14. Жазуунун структурасын колдонуп, студенттердин группасы деген жазууну баяндагыла(фамилия, туулган жылы, туулган айы, туулган датасы). Группадагы студенттердин тизмеси жана айлар боюнча туулган күндөрүн экранга чыгаруучу программасын түзгүлө, мисалы:

январь 12 Үсөнов Асан
23 Таабалдиев Курманбек
25 Ташбалтаев Төлөгөн
февраль 5 Молдобаев Эмил
11 Жээнбеков Акыл

15. Жазуунун структурасын колдонуп мектептик класс жазуусун түзгүлө. Предметке катышкан, баа алган окуучулардын фамилиясы жөнүндө маалыматты сакташ үчүн талааларды жазууда көрсөткүлө. Класстын жетишкендиги тууралуу берилгендерди эсептөөчү жана экранга

класстын отличниктерин жана орто окуган окуучулары тууралуу маалыматты чыгаруучу программаны түзгүлө.

16. Химиялык элементтердин таблицасын баяндаган программаны төмөнкү маалыматтарды чагылдырып түзгүлө: аталышы, символикалык белгиси, атомдун массасы, атомдук ядронун заряды, негизги химиялык касиеттердин саны. Программа көрсөтүлгөн символикалык белгиси боюнча химиялык элемент жөнүндө берилгендерди аткарышы керек.
17. Телефондук справочник деген жазуулардын массивин баяндоочу программаны түзгүлө. Фамилия боюнча телефондун номерин издөө, эсептөө жана «компьютердик оюндарды ойноо» критериясы боюнча бардык абоненттердин тизмесин чыгаруучу программаны түзгүлө. Жазууда ар бир абонент боюнча төмөнкү маалыматтар кармалат: фамилиясы, аты, телефону, хоббиси.
18. Китептер жөнүндө жазууларды кармаган файл берилген. Ар бир китептин маалыматтары: китептин аталышы, автордун фамилиясы, чыккан датасы. Чыккан датасы 2000-жылдан кем болгон китептердин санын чыгаруунун программасын түзгүлө.
19. Өзүңөрдүн досуңар жөнүндө жазуулар сакталган программаны түзгүлө. Клавиатурадан кийирилген номери боюнча маалыматты чыгарууну уюштургула. {Мисалы, жазуунун номери 3:

    Фамилиясы: Асанов
    Аты: Үсөн
    Атасынын аты: Султанбекович
    Туулган жылы: 1977
    Туулган айы: 12
    Туулган числосу: 22}
20. Түрдүү даталарды кармаган файл түзгүлө. Ар бир дата-бул число, айы жана жылы. Клавишанын басылышы боюнча экранга чыгарууну уюштургула:

    F1-жыл;
    F2-жаз айындагы даталар;
    F3-кыш айындагы даталар;
    F10-чыгуу.

Текшерүү үчүн суроолор

1. Жазуу деген эмне?
2. Жазууну өзгөрүлмөлөр бөлүгүндө кантип жарыялайбыз?
3. Жазууну типтер бөлүгүндө кантип жарыялайбыз?

Лабораториялык иш №17

Тема: Модулдар.

Түрдүү программаларда берилгендердин тибин, процедура-функциялардын наборун колдонуу керек болот. Паскалда мындай маселени модулду түзүп жана компиляциялоо менен чечүүгө болот. Өз учурунда Турбо-Паскалдын түзүүчүлөрү IBM ПКнин бүт мүмкүнчүлүктөрүн колдонуу

үчүн берилгендердин тибин, константаларды, керектүү процедура жана функцияларды модуль көрүнүшүндө түзүшкөн. Төмөнкү ушул модулдардын тизмеси:

System модулу-стандарттык процедура жана функцияларды кармап турат, б.а "классикалык" Паскалдын процедура жана функциялары. System модулу автоматтык түрдө бардык программалар үчүн иштейт.

Dos модулу-MS-DOS ОСнын каражаттарын колдонууга мүмкүн болгон процедура жана функцияларды кармап турат.

Crt модулу-экран, клавиатура, IBM компьютеринин динамиги менен иштөө үчүн процедуралардын жыйындысы.

Graph модулу-(түрдүү графикалык CGA, EGA, Hercules, ATT 400, MCGA, 3270 PC, VGA тибиндеги адаптерлери колдогон) компьютердин графикалык мүмкүнчүлүгүн колдонууга мүмкүндүк берүүчү программалардын кеңири набору.

Printer модулу-бул модул принтер менен иштөөнү жеңилдетет.

Graph3-Турбо-Паскалдын 3.0 версиясы үчүн графикалык программанын толук наборун кармап турат.

Turbo3-Graph3 сыяктуу 3.0 версиясы менен биригиши үчүн кийирилген.

Модулду колдонуу үчүн программанын бөркүнөн(program программанын аты;) кийин ***uses модулдун аты;*** жазуу жетиштүү.

Эгерде программада бир канча модул колдонулса, анда төмөндөгүдөй жазабыз:

uses 1-модулдун аты, 2-модулдун аты,..., N-модулдун аты;

Лабораториялык иш №18

Тема: Өздүк модулдар.

Турбо-Паскалда ***өздүк модулдарды*** түзүүгө мүмкүндүк бар.

Модулдун структурасы:

```
unit модулдун_аты;
Interface
----------
{ачык баяндоо бөлүгү-интерфейстик секция}
----------
Implementation
{жабык баяндоо бөлүгү}
begin
----------
{инициализациялоо секциясы }
----------
end.
```

Модуль **unit** сөзү менен башталып, анан модулдун аты жазылат. Андан кийин сексиянын башталышын билдирген **interface** сөзү жазылат. Берилген модуль interface ачкычтык сөзүнөн кийин uses кызматчы сөзү жана колдонулган модулдардын тизмесин жайгаштыруу зарыл.

Интерфейс бөлүгү-interface жана implementation сөздөрүнүн ортосундагы модулдагы бөлүк. Бул разделде константаларды, берилгендердин типтери, өзгөрүлмөлөр, процедуралар жана функцияларды аныктоого мүмкүн. Берилген модулду колдонуучу бардык программалар

жана модулдар үчүн көрсөтмөлүү болот. Бул процедура жана функциялардын телосу implementation сөзүнөн кийин башталуучу реализация сексиясында болот. Реализация сексиясында берилген модуль колдонулган программалар жана модулдар үчүн көрүнбөс болгон өздүк баяндалышы болушу мүмкүн. Интерфейс сексиясында баяндалган константалар, типтер, өзгөрүлмөлөр, процедуралар жана функциялар реализация секциясында көрүнүктүү б.э. Интерфейс сексиясындагы баяндалган процедуралар жана функциялар реализация сексиясында дагы бир жолу жарыяланат, алардын бөркү интерфейс секциясында көрсөтүлгөндөй, өзүндөй болушу керек.

Инициализация сексиясы begin жана end сөздөрүнүн ортосунда жайгашат. Эгерде begin сөзү жок болсо, анда инициализация сексиясы да жок болот. Инициализация сексиясында негизги программага чейинки башкарууну аткаруучу операторлор жайгашышат. Бул операторлор көбүнчө ишке программаны даярдоо үчүн колдонулат.

Мисалы, инициализация сексиясында өзгөрүлмөлөр инициализацияланышы мүмкүн, керектүү файлдар ачылат. Программаны аткарууда кээ бир колдонулуучу модуль-бул модулдун инициализация сексиясы негизги программанын телосу жүктөлгөнгө чейин чакырылат. Бир нече модулдарды колдонууда алардын инициализация сексиясы uses операторунда көрсөтүлгөн тартипте чакырылат.

Мисал катары кичинекей модулду алалы, анда Min(X, Y) функциясы эки сандын минимумун, Max(X, Y) функциясы эки сандын максимумун кайтарып берүүчү функциялар баяндалган.

```
unit Study;
Interface {ачык баяндоолор бөлүгү-интерфейстик сексия}
    function Min(X, Y:integer):integer;
    function Max(X, Y:integer):integer;
    Implementation
        {жабык баяндоолор бөлүгү}
    function Min(X, Y:integer):integer;
    begin
    if X<=Y then Min:=X else Min:=Y;
    end;
    function Max(X, Y :integer):integer;
    begin
     if X>=Y then Max:=X else Max:=Y;
    end;
end.
```

Бул модулду жазып, жыйынтыгы Study.tpu аты менен файлды компиляциялоо керек. Компиляцияланган модуль сөзсүз study.tpu аттуу болушу керек, себеби модулдун биринчи жолчосунда "unit study" деп жазылган, эми колдонулуучу модуль « uses study» деп жазылат.

Tpu-файлдын аты бул жолчодон эмес, модулдун учурдагы тексти сакталган файлдын атынан алынып жатат. Эгерде учурдагы текстин модулу сакталган файл Myfile.pas деп аталса, анда компиляциялоонун жыйынтыгында myfile.tpu файлын алабыз.

Программада Min жана Max функцияларын Study модулунан колдонот.

2) {М массивинде myfile.int файлынан биринчи 100 бүтүн сандар эсептелет. Программа массивде минималдык жана максималдык сандарды табат жана аларды экранга чыгарат} .

```
uses crt study;
var
      M: array [1..100] of integer;
      F:file of integer;
Min_, Max_,I:integer;
begin
      Assign (F,'myfile.int');
      Reset(F) ;{массивди инициализациялоо}
      For I:=1 to 100 do
      Read(F, M[i]);
{максимум жана минимумду табуу}
Min_:=M[1];
Max_:=M[1];
for I:=1 to 100 do
begin
Min_=Min(Min_, M[i]);
Max_=Max(Max_, M[i]);
end;
{үн сигналы}
Sound(300);
Delay (100);
Nosound ;
writeln('минимум=' , Min_, 'максимум=', Max_);
readln;
end.
```

Лабораториялык иш №19

Тема: System модулунун процедуралары жана функциялары.

System модулунун процедуралары жана функциялары автоматтык түрдө бардык программалар үчүн доступный жана system модулунда uses деп жазуу шарт эмес.

Программанын шитешин башкаруучу процедуралар.

Exit процедурасы учурдагы блоктон тез чыгууга мүмкүндүк берет.

Баяндалышы: **Exit;**

Мисал,

```
program test_prog;
function test:integer;
begin
      test:=5;
      exit;
{кийинки оператор аткарылбайт жана Test функциясы 10ду эмес 5 ти
кайтарып берет}
      test:=10;
end;
begin
```

```
    writeln(test);
    {«5» печатталат}
end.
```

Halt процедурасы программанын аткарылышын токтотот жана башкарууну операциялык системага кайтарып берет.

Баяндылышы: **Halt[(ExitCode:word)];**

ExitCode шарттуу эмес параметри программаны аяктоонун кодун кайтарып берет. Эгерде ExitCode параметри чакырууда калып калса, анда программаны аяктоонун коду 0гө барабар.

Мисал,

```
    begin
    Halt;
    {программанын аткарылышы Halt процедурасы менен бузулду жана кийинки оператор аткарылбайт}
    writeln('*****');
end.
```

RunError процедурасы программанын аткарылышын токтотот жана аткарылуу убактысынын катасын генерациялайт.

Баяндалышы: **RunError[(ErrorCode:word)];**

Процедураны пайдалануунун жыйынтыгында экранга ErrorCode параметринин номери менен берилген ката тууралуу маалымат чыгат.

Типтерди келтирип чыгаруучу функциялар.

Chr- символду кайтарат, ASCII таблицасында берилген бүтүн санга барабар болгон иреттүү номер.

Баяндалышы: **Chr(N:byte):char;** N-символдун иреттүү номери.

Мисал,

```
var
    I:integer;
begin
    writeln('ASCIIнин берилгендери печатталат');
    for I:=0 to 255 do
        write(Chr(I));
    readln;
end.
```

Ord-эсептелүүчү типтин мааниси боюнча иреттүү санды кайтарат.

Баяндалышы: **Ord(X):longint;** X-эсептелүүчү типтин мааниси.

```
uses crt;
var
    Ch:char;
begin
    {Esc клавишасы басылганга чейин кайталанат}
    repeat
        ch:=Readkey;
        write(',', Ord(Ch),'.');
    until Ch=#27;
readln;
end.
```

Round- чыныгы бүтүн типтин маанисин узун бүтүн типке ээ болгонго чейин тегеректейт.

Баяндалышы: **Round(X:real):longint;**

```
begin
        writeln('5.6 болжол менен барабар', Round(5.6));
end.
```

Trunc- чыныгы бүтүн типтин маанисин узун бүтүн типке ээ болгонго чейин кыркып(кесип) алат.

Баяндалышы: **Trunc(X:real):longint;**

```
begin
        writeln('5.6 ны кыркуу', Trunc(5.6));    { жообу: 5}
end.
```

Арифметикалык функциялар.

Abs-аргументтин абсолюттук маанисин(модуль) кайтарып берет.

Баяндалышы: **Abs(X);**

ArcTan-аргументтин арктангенсин кайтарат.

Баяндалышы: **ArcTan(X:real):real;**

Мисал,

```
begin
        writeln(ArcTan(Pi/4);
end.
```

Cos-аргументтин косинусун кайтарат.

Баяндалышы: **Cos(X:real):real;**

Мисал,

```
var
        X:real;
begin
        writeln('Бурч үчүн Пифагордун теоремасы текшерилет=');
        readln(X);
        if    sin(x)*sin(x)+cos(x)*cos(x)-1<0.01    then    writeln('туура!')    else
        writeln('туура эмес!');
end.
```

Exp-аргументтин экспонентасын кайтарып берет.

Баяндалышы: **Exp(X:real):real;**

Frac-аргументтин бөлчөк бөлүгүн кайтарат.

Баяндалышы: **Frac(X:real):real;**

Int-аргументтин бүтүн бөлүгүн кайтарат.

Баяндалышы: **Int(X:real):real;**

Ln-аргументтин натуралдык логарифмин кайтарып берет.

Баяндалышы: **Ln(X:real):real;**

Pi-Pi санынын маанисин берет.

Баяндалышы: **Pi:real;**

Sin-аргументтин синусун кайтарат.

Баяндалышы: **Sin(X:real):real;**

Sqr-аргументти квадратта кайтарат.

Баяндалышы: **Sqr(X);**

Sqrt-аргументтин квадраттыктамырын кайтарат.

Баяндалышы: **Sqrt(X:real):real;**

Эсептелүүчү типтин процедуралары жана функциялары.

Dec процедурасы өзгөрүлмөнүн маанисин кичирейтет.

Баяндалышы: **Dec(var x[;n:longint]);**

n параметри аргументтин мааниси кичирейген санды кармап турат, аргумент 1ге кичиреет.

```
var
      x:integer;
begin
      x:=3;
      dec(x,2);
      writeln(x);
end.
```

Inc процедурасы өзгөрүлмөнүн маанисин 1ге чоңойтот.

Баяндалышы: **Inc(var x [;n:longint]);**

n параметри аргументтин маанисин чоңойткон санды кармап турат, аргумент 1 ге чоңоет.

Odd процедурасы аргумент так же жуп экендигин текшерет.

Баяндалышы: **Odd(x:longint);boolean;**

```
var
      i:integer;
begin
      readln(i);
if Odd(i) then writeln('так') else writeln('жуп');
readln;
end.
```

Pred процедурасы аргументтин алдынкы маанисин кайтарып берет.

Баяндалышы: **Pred(x);**

```
begin
      writeln('8дин алдындагы', pred(8));
end.
```

Succ процедурасы анын кийинки маанисин кайтарып берет.

Баяндалышы: **Succ(x);**

```
begin
      writeln('10дон кийинки ', succ(10));
end.
```

Лабораториялык иш №20

Тема: ***Жолчолук процедуралар жана функциялар.***

Иштин максаты-жолчолук процедуралар жана функциялар менен иштөөнү үйрөнүү, алардын колдонулуш максатын билүү.

Өз алдынча иштөө үчүн тапшырмалар

1. Жолчолук процедуралар
2. Жолчолук функциялар

Concat функциясы жолчолордун удаалаштыгын конкатенацияloону(кошууну) аткарат.

Баяндалышы: **Concat(S1[,S2,...,SN]:string):string;**

S1 жолчосу параметр катары көрсөтүлгөн жолчолорду удаалаш бириктирет.

Мисал,

```
var
      a,c:string;
begin
      a:='Компьютердик технологиялар';
      c:='факультети';
```

```
        writeln(concat(a,c));
readln;
end.
```
Ж:='Компьютердик технологиялар факультети';

Copy функциясы жолчодон ички жолчону кайтарып берет.

Баяндалышы: **Copy(S:string; index, count:integer):string;**

S-учурдагы жолчо, index-Sтен бөлүнүүчү ички жолчонун символунун номери, count-ички жолчодогу символдордун саны.

Мисал,
```
var
        a,b:string;
begin
        a:='математика';
        b:=copy(a,3,4);
        writeln(b);
readln;
end.
```
Ж:='тема';

Delete процедурасы жолчодон ички жолчону өчүрөт.

Баяндалышы: **Delete(var S:string;index:integer;count:integer);**

S-учурдагы жолчо, index-Sтен өчүрүлүүчү ички жолчонун символунун номери, count-ички жолчодогу символдун саны.

Мисал,
```
var
        a:string;
begin
        a:='китепкана';
        Delete(a,6,4);
        writeln(a);
readln;
end.
```
Ж:='китеп';

Insert процедурасы жолчого ички жолчону кошот.

Баяндалышы: **Insert(S1:string;var S2:string;index:integer);**

S2-учурдагы жолчо, S1-кошулуучу ички жолчо, index-S1 ички жолчосу кошулуучу S2 жолчосунун символунун номери.

Мисал,
```
var
        s:string;
begin
        s:='Ош Университети';
        Insert(' Мамлекеттик',s,4);
        writeln(s);
readln;
end.
```
Ж:'Ош Мамлекеттик Университети';

Length функциясы жолчонун динамикалык узундугун кайтарып берет.

Баяндалышы: **Length(S:string):integer;**

Мисал,
```
var
```

```
        s:string;
        i:integer;
begin
        s:='информатика';
        i:=length(s);
readln;
end.
```

Ж: 11

Pos функциясы жолчодон ички жолчону издейт.

Баяндалышы: **Pos(Substr,S:string):byte;**

substr-S жолчосунда издөө жүргүзүлүүчү ички жолчо. Pos функциясы Sten табылган substrдин символунун номерин кайтарып берет. Эгерде substr табылбаса, анда 0 болот.

Мисал,

```
var
        S:string;
        I:integer;
begin
        S:='МИКРОПроцесСОР';
        I:=Pos('PO',S);
        writeln(I);
readln;
end.
```

Ж: I:=4;

Str процедурасы сандык маанини жолчолук мааниге айландырат.

Баяндалышы: **Str(x[:width[:decimals]]:var S:string);**

width параметри-S жолчосунун узундугу, decimals-үтүрдөн кийинки белгилердин саны.

Мисал,

```
var
        s:string;
        x:real;
begin
        x:=545, 678;
        Str(x:2:2,s);
        writeln(s);
readln;
end.
```

Ж:='545, 678';

Val процедурасы жолчолук маанини сандык көрүнүшкө айландырат.

Баяндалышы: **Val(S:string;var V; var code:integer).**

S-учурдагы жолчо, V-Stин сандык көрүнүшү жайгашышы керек болгон сандык өзгөрүлмө, code-бүтүн өзгөрүлмө. Эгерде Sти санга өзгөртүү мүмкүн эмес болсо, анда Val процедурасы аткарылгандан кийин code өзгөрүлмөсү нөлдүк маанини алат.

Мисал,

```
var
        s:string; x:real; c:integer;
begin
        s:='125,67';
```

```
    Val(s,x,c);
    writeln(x:2:2);
readln;
end.
```

Вариаттар

1. «Osh University» жолчосуна «State» ички жолчосун кошуп, «Osh State University» жолчосун түзгүлө.
2. N натуралдык саны жана a1, a2, ...an символдору берилген. a1, a2, ...an удаалаштыгында а тамгасы канча жолу кезигээрин эсептегиле.
3. «Азыр сабак болуп жатат» сүйлөмүнөн «болуп жатат» сөздөрүн алып таштагыла.
4. N натуралдык саны жана a1, a2, ...an символдору берилген. a1, a2, ...an удаалаштыгында "б" тамгасы менен башталган бардык сөздөрдү тапкыла.
5. «Студенттик өмүр» жолчосу экранда бир пайда болуп, бир жок болуп туруучу программаны түзгүлө.
6. N натуралдык саны жана a1, a2, ...an символдору берилген. a1, a2, ...an удаалаштыгында эң кыска сөздүн узундугун тапкыла.
7. «1, 2., 3, 4, 5, 6, 7» сандарын ирети менен бирден экранга чыгаруучу программаны түзгүлө.
8. N натуралдык саны жана a1, a2, ...an символдору берилген. a1, a2, ...an удаалаштыгындагы "тарак" сөзүн "терек" сөзүнө алмаштыргыла.
9. Фамилияңды, атыңды, атаңдын атын кошуунун программасын түзгүлө.
10. Берилген тексттеги C1 сөзүн C2 сөзү менен алмаштыргыла.
11. N натуралдык саны жана a1, a2, ...an символдору берилген. a1, a2, ...an удаалаштыгындагы биринчи жана акыркы символдору дал келген сөздөрдүн санын тапкыла.
12. «1978» жолчолук маанисин сан маанисинө өзгөрткүлө.
13. Тексттеги сөздөрдүн санын аныктагыла.
14. «ABC789» жолчосунан «ABC» ички жолчосун алып таштагыла.
15. «Программалоо» жолчосунан «программа» ички жолчосун алуунун программасын түзгүлө.
16. 789, 123 сандарын жолчолук мааниге өзгөрткүлө.
17. Тексттеги сөздөрдүн арасында катар кеткен үч «е» тамгасы бар экендигин текшергиле.
18. Паскаль тилиндеги программанын структурасын экранга чыгаруучу программаны түзгүлө.
19. N натуралдык саны жана a1, a2, ...an символдору берилген. a1, a2, ...an удаалаштыгында "кафе" деген тамгалардын группасын "кафедра" деген тамгалардын группасына өзгөрт.
20. N натуралдык саны жана a1, a2, ...an символдору берилген. a1, a2, ...an удаалаштыгында жанаша жайгашкан символдордун бирдей экендигин аныктагыла.

Лабораториялык иш №21

Тема: ***CRT модулу. CRT модулунун процедуралары жана функциялары.***

План

1. ***Экран менен иштөөчү процедуралар жана функциялар***
2. ***Клавиатура менен иштөөчү функциялар***
3. ***Үйдү башкаруучу процедуралар***
4. ***Ар түрдүү арналыштагы процедуралар***

Crt (Cathode-Ray Tube-электрондук-нурлуу түтүкчө) модулу Turbo Pascalдын библиотекалык файлына камтылып, анын процедура жана функциялары төмөндөгү иш-аракеттер үчүн арналган: клавиатураны башкаруу, тексттик экранды башкаруу, анын ичинде: терезени формулировкалоо, курсордун абалын башкаруу, экрандагы символдордун, фондун түсүн аныктоо, үн сигналдарын башкаруу(динамик менен иштөө) үчүн арналган. Аны программада колдонуу үчүн Uses Crt; оператору программада болушу керек.

Экран менен иштөө үчүн процедуралар.

Window процедурасы бул экранда тексттик терезени аныктайт, терезе өзүнчө экран катары эсептелет.

Жалпы форматы: **Window (x1, y1, x2, y2:Byte) ;**

Мында X1-терезенин сол жогорку бурчунун мамычасы, Y1-терезенин сол жогорку бурчунун жолчосу, X2- терезенин оң төмөнкү бурчунун мамычасы, Y2- терезенин оң төмөнкү бурчунун жолчосу.

TextBackGround процедурасы фондун түсүн берет.

Жалпы форматы:**TextBackGround(Color:byte);** (Col-керектүү түс)

TextColor процедурасы тексттин түсүн орнотот.

Жалпы форматы: **TextColor (Color: byte**); (Color-керектүү түс).

ClrEol процедурасы курсордун позициясынан баштап жолчонун аягына чейин символду тазалайт. Жалпы форматы: **ClrEol;**

ClrScr процедурасы экранды тазалайт жана курсорду экрандын сол жогорку бурчуна жайгаштырат. Жалпы форматы: **ClrScr;**

```
1) program pr_1;
uses crt;
var
   I: integer;
begin
      TextBackGround(2);
       TextColor(14);
Clrscr;
for I:=1 to 50 do
Write (' Азыр сабак болуп жатат');
Window (5, 5, 25, 15);
TextBackGround (blue);
TextColor (15);
Clrscr;
for I:=1 to 10 do
write ('жаңы терезе');
readln;
readln;
```

end.

DelLine процедурасы курсор турган жолчону өчүрөт. Бардык жолчолор бир жолчо жогору жылат жана акыркы жолчо тазаланып калат.

Жалпы форматы: **DelLine** ;

GOTOXY процедурасы берилген жолчо жана мамычага курсордун позицияланышын аткарат. Жалпы форматы: **GOTOXY (X, Y:Byte);**

InsLine процедурасы курсор жайгашкан жерге бош жолчо коет.

Жалпы форматы: **InsLine;**

2)uses crt ;
```
var
  i:integer;
begin
      TextBackGround (0);
   TextColor (white);
   ClrScr;
     for I:=1 to 5 do
     begin
         GotoXY(2*i,  i);
         write ('программалоо тили', I);
     end;
     TextBackGround(1);
     Window (50, 10, 70, 17);
    ClrScr;
    for i:=1 to 5 do
    begin
        GOTOXY(2*I, I);
       write ('программалоо тили',I);
   end;
 readln;
end.
```
3) uses crt;
```
   var
     I:integer;
  begin
     TextBackGround(15);
    ClrScr;
    for I:=3 to 10 do
    begin
        TextBackGround(I);
        Window (3*I, I, 80-3*I, 22-I);
        ClrScr;
   end;
end.
```

Экран менен иштөөчү функциялар

Where X функциясы курсордун учурдагы X координатасын берет, б.а. учурдагы терезеге жараша курсордун позициясынын мамычасынын номерин берет. Жалпы форматы: **Where X:Byte;**

Where Y функциясы курсордун учурдагы Y координатасын берет, б.а. учурдагы терезеге жараша курсордун позициясынын жолчосунун номерин берет. Жалпы форматы: **Where Y: Byte;**

Түстөрдү башкаруу процедуралары

HingVideo процедурасы экранга чыгарылуучу символдордун жогорулатылган жарыктыгын орнотот. (0-7ге чейинки түстөрдү 8-15ке чейинки түстөргө алмаштырат).
Жалпы форматы: **HingVideo;**
LowVideo процедурасы символдордун жарыктануусун төмөндөтөт(8-15ке чейинки түстөрдү 0-7ге чейинки түстөргө алмаштырат).
Жалпы форматы: **LowVideo;**
NormVideo процедурасы символдордун жарыктануусун нормалдаштырат.
Жалпы форматы: **NormVideo;**

Клавиатура менен иштөөчү функциялар

Keypressed функциясы басылган клавишаны текшерет, эгерде клавиша клавиатурадан басылган болсо True маанисин берет, False маанисин тескери учурда алат.
Жалпы форматы: **Keypressed: Boolean;**

4) uses crt;
```
  begin
  {экранда кандайдыр бир клавиша басылганга чейин экранда тик бурчтук
  өчүп-жанып чыга берет}
     repeat
     TextBackGround (Random(8));
     Window (30, 8, 50, 16);
     ClrScr;
     Delay(300);
 until KeyPressed;
end.
```

Readkey функциясы басылган клавишанын символун окуйт.
Жалпы форматы: **Readkey : Char;**

5)uses Crt;
```
var
      ch:Char;
begin
      ClrScr;
{Esc клавишасы басылганга чейин кайталайт}
      repeat
      ch:=ReadKey
      If ch=#0 then
begin
      ch:= ReadKey;
      Write('~');
end;
write(Ord(ch),'_');
until ch=#27;
readln;
end.
```

Үндү башкаруучу процедуралар.

Sound процедурасы берилген жыштыкта үн чыгарат.
Жалпы форматы: **Sound(H:word);** мында H- үндүн жыштыгы.

Nosound процедурасы үнду өчүрөт.
Жалпы форматы: **Nosound;**
М: 6)uses crt;
begin
repeat
Sound(Random(100)+100);
until KeyPressed;
Nosound;
end.
7)uses crt;
var
i:integer;
begin
for i:=1 to 5 do
begin
Sound(100); Delay (200); Nosound;
Delay (500); Sound(100); Delay(100); Nosound;
Delay(500);
end;
end.
8)uses crt;
begin
repeat
sound(262);
delay(8000);
sound(294);
delay(6000);
sound(330);
delay(7000);
sound(349);
delay(5000);
sound(392);
delay(4000);
sound(440);
delay(3000);
sound(494);
delay(7000);
sound(524);
delay(9000);
until keypressed;
nosound;
end.

Ар түрдүү арналыштагы процедуралар

Delay(H:word); процедурасы программанын аткарылышын мл|с га токтотуп турат.
AssignCrt процедурасы Crtнын орнотулушу үчүн тексттик файлды берет.
Жалпы форматы: **AssignCrt(var f:Text);**

TextMode процедурасы текстгик режимди орнотот, учурдагы окнону толук экранга чоңойтот.
Жалпы форматы: **TextMode(Mode:word);** (mode-талап кылынган текстгик режим).

Түстөрдүн константалары

```
Black=0; {кара}
Blue=1; {көк}
Green=2; {жашыл}
Cyan=3; {бирюзовый}
Red=4; {кызыл}
Magenta=5; {малина өңдүү}
Brown=6; {күрөң}
LightGray=7; {ачык боз}
DarkGray=8; {ток боз}
LightBlue=9; {ачык көк}
LightGreen=10; {ачык жашыл}
LightCyan=11; {ярко-бирюзовый}
LightRed=12; {ачык-кызыл}
LightMagenta=13; {ачык малина өңдүү}
Yellow=14; {сары}
White=15; {ак}
Blink=128; {бүлбүлдөө(мерцание)}
```

0-7 ге чейин символ катары, фон катары колдонсо болот, калгандары, мерцание коду жалаң символ катары колдонулат.
Экранга ирмелүү текстин чыгаруу үчүн алдын-ала TextColor(Color+Blink); процедурасын чакырып алуу керек. Мында Color текстгин түсү. Төмөнкү программа ирмелүү көздөрү менен бет түзүлүшүн чагылдырат.

```
9)uses crt;
begin
    TextBackGround (black);
    ClrScr;
        TextBackGround(brown);
        Window (36, 11, 44, 14);
        ClrScr;
        GOTOXY(1, 2);
        TextColor (blink+blue);
        Write ('* *');                    {ирмелүү көздөрү}
        TextColor (white);
        GOTOXY(5,3);
        Write ('^');                      {мурууну}
        GOTOXY(1,4);
        Write (' # # # # ');              {оозу}
end.

10)uses crt;
const
  text='Азыр информатика сабагы';
  textlen=7;
  MinLen=textlen+6;
  pause=1000;
```

```
 var
   x1,y1,x2,y2:word;
   background:word;
   color:word;
   setblink:byte;
   freg:word;
 procedure DoubleFrame;
 var
  i:byte;
 begin
   gotoxy(2,1);
   write(' ');
   for i:=3 to x2-2 do
   write('=');
   write('»');
 for i:=2 to y2-1 do
 begin
   gotoxy(2,i);   write('є');
   gotoxy(x2-1,i);  write('є');
 end;
 gotoxy(x2,y2);write('И');
 for i:=3 to x2-2 do
 write('=');write('j');
 end;
 begin
 textbackground(black);
 clrscr;randomize;
 while not keypressed do
 begin
 x1:=1+random(80-MinLen);
 x2:=minlen+random(80-x1-minlen);
 y2:=round(x2*25Y80)-1;
 y1:=1+random(24-y2);
 window(x1,y1,x1+x2-1,y1+y2-1);
 delay(1000);
 background:=random(8);
 color:=random(10);
 setblink:=random(2);
 textbackground(background);
 clrscr;
 textcolor(color+blink+setblink);
 gotoxy((x2-textlen) div 2+1, (y2-1) div 2+1);
 write(text);
 DoubleFrame;
 BackGround:=(TextAttr and $70);
 case background of
 0:freg:=262;
 1:freg:=294;
 2:freg:=330;
 3:freg:=349;
```

```
4:freg:=392;
5:freg:=440;
6:freg:=494;
7:freg:=524;
end;
sound(freg); delay(pause); nosound;
end;
window(1,1,80,25);
textbackground(black);
textcolor(lightgray);
clrscr;
end.
```

Варианттар

1. Экранга терезе чыгарып, «Enter» клавишасы басылган сайын терезе түрдүү түскө алмашып туруучу программаны түзгүлө.
2. Кесилишкен квадратты чыгаруунун программасын түзгүлө.
3. Музыка же үн пайда боло турган программаны түзгүлө
4. Экрандын жалпы фону жашыл түстө болгон терезенин ичине «информатика» текстин кызыл түстө жазгыла.
5. Клавишаны басканда анын кодун окуй турган программаны түзгүлө.
6. Өчүп-жанып чыгып туруучу тик бурчтук түзгүлө.
7. Терезелерди кокустук абалда чыгаруунун программасын түзгүлө.
8. Жети нотанын үнүн чыгаруучу программаны түзгүлө.(до-262, ре-294, ми-330, фа-349, соль-392, ля-440, си-494).
9. Латын алфавитинин символдорунун кодун чыгаруучу программаны түзгүлө.
10. Экранды тазалоочу программаны түзгүлө.
11. Символдордун жарыктануусун өзгөртүүнүн программасын түзгүлө.
12. Жолчолорду өчүрүп, текстгин жаңы жолчосун экранга чыгаруучу программаны түзгүлө.
13. Үй түзүүнүн программасын түзгүлө.
14. Символдордун жарыктануусун нормалдаштыруунун программасын түзгүлө.
15. X жана Y координаталарын берип, ошол координата боюнча курсорду жайгаштыруунун программасын түзгүлө.
16. KeyPressed процедурасын колдонуп программа түзгүлө.
17. Терезелерди ар түрдүү түс менен кокустук абалда чыгаруунун программасын түзгүлө.
18. Фондун түсүн кара түстө, текстгин түсүн көк түстө чыгаруунун программасын түзгүлө.
19. Кубду чыгаруунун программасын түзгүлө.
20. Курсордун позициясынан баштап жолчонун аягына чейин символду тазалоонун программасын түзгүлө.

Лабораториялык иш №22

Тема: **Dos модулу.Dos модулунун процедуралары жана функциялары.**

FindFirst процедурасы учурдагы же берилген каталогдон атрибуттар менен берилген биринчи файлды издейт.
Баяндалышы: **FindFirst(Path:string, Attr:word; var S:SearchRec);**
Path-файлдын толук аты жана файлдын атынын мүнөздүк белүгү.
Мисалы, *.pas-учурдагы каталогдо жайгашкан .pas кенейтилишиндеги файл.
Attr-файлдын атрибуттары, S-SearchRec тибиндеги өзгөрүлмө.

```
type
      SearchRec=record
      fill:array[1..21] of byte;
      attr:byte;
      time, size:longint;
      name:string[12];
end;
```

Мисал,

```
uses dos;
var
      fileinfo: SearchRec;
begin
FindFirst('*.pas', archive, fileinfo);
If fileinfo.name<>' ' then
writeln('.pas кенейтилишиндеги файл катышпайт');
readln;
end.
```

FindNext процедурасы FindFirst процедурасынын шартындай эле кийинки файлды издейт.
Баяндалышы: **FindNext(var S: SearchRec);**
S- SearchRec тибиндеги өзгөрүлмө.
Мисал,

```
uses   Dos;
var
      fileinfo: SearchRec;
      N:integer;
begin
      N:=0;
      FindFirst('C:\Dir\*.pas', archive, fileinfo);
while DosError=0 do
begin
      N:=N+1;
      writeln(fileinfo.name);
      FindNext(fileinfo);
end;
writeln('C:\Dir\ каталогунда' , N ' .pas кенейтилиши менен архивдик файл табылды');
readln;
end.
```

Эсептөө процессин башкаруу процедуралары.

Exec(Path, LmdLine:string); процедурасы берилген командалык жолчодон параметрлер менен берилген программаны аткарат.

LmdLine-командалык жолчо.

```
uses dos;
var
      ProgramName, LmdLine:string;
begin
      write('program to exec(full path):');
      readln(ProgramName);
      write('command line to pass to', ProgramName, ':');
      readln(CmdLine);
      writeln('about to exec...');
      SwapVectors;
      Exec(ProgramName, CmdLine);
      SwapVectors;
      writeln('...back from Exec');
      if DosError<>0 then
      writeln('DosError#', DosError) else
      writeln('Exec successful.', Child process exit code=', DosExitCode);
readln;
end.
```

Бул программа Dostогу .exe файлдарды чыгарат.

GetIntVec(IntNo:byte; var Vector:Pointer); процедурасы үзгүлтүккө учуроону кайра иштетүүчү программанын адресин берет. IntNo-үзүлүү чекити(0...255), Vector-Pointer тибиндеги өзгөрүлмө, Intr-үзгүлтүккө учураган программаны аткартат.

Intr(IntNo:byte; Regs:Registers);

IntNo-үзүлүү чекити, Regs-Registers тибиндеги өзгөрүлмө.

MsDos(var Regs:Registers); MsDos функциясынын аткарылышы.

Reep(ExitCode:word); процедурасы программанын аткарылышын токтотот жана эске аны сактайт. Мында ExitCode-аяктоо коду.

SetIntVec(IntNo:byte;Vector:Pointer); процедурасы программанын иштелишинин бөлүнүү адресин жүктөйт.

SwapVectors; процедурасы бөлүнүү векторун кайра аныктайт.

Эсептөө процессин башкаруу функциялары.

DosExitCode:word; аткаруучу процесстин чыгуу кодун берет.

EnvCount:integer; Ms-Dosто жолчонун санын берет.

EnvStr(index:integer):string; операциялык системада көрсөтүлгөн жолчону берет.

GetEnv(Env var:string):string; Ms-Dosто көрсөтүлгөн өзгөрүлмөнүн маанисин берет.

Убакыттар жана даталар менен иштөөчү процедуралар.

GetDate(var Year, Month, Day, Day of week:word); процедурасы Ms-Dos то түзүлгөн учурдагы датаны берет.

```
uses dos, crt;
const
days:array[0..6] of string[9]=('Sunday', 'Monday', 'Tuesday', 'Wednesday',
'Thursday', 'Friday', 'Saturday');
var
```

```
        y, m, d,dow::word;
begin
        GetDate(y, m, d, dow);
        Writeln('today is', 'days[dow], ',', m:0, '/', d:0, '/', y:0);
end.
```

Today is Monday, 23.05.05

GetFTime(var F;var Time:LongInt); процедурасы файлдын акыркы жаңыланган датасын жана убактысын берет. Мында, F-файл.

GetTime(var Hour, Minute, Second, Sec100:word); процедурасы учурдагы убакытты берет. Hour-саат.

```
Uses dos, crt;
var
        h, m, s, hund:word;
function LeadingZero(w:word):string;
var
        S:string;
begin
        Str(w:0, S);
if length(S)=1 then S:='0'+S;
LeadingZero:=S;
end;
begin
        GetTime(h, m, s, hund);
        writeln('it is now', leadingzero(h), ':'), leadingzero(m), ':', leadingzero(S), '.',
leadingzero(hund));
end.
```

Now, it is 16:50:21.09

PackTime(var DT:DateTime; var Time:LongInt); процедурасы берилген дата жана убакытты (упаковкалайт).

DataTime-өзгөрүлмөлүү тип, Time-өзгөрүлмө.

SetDate(Year, Month, Day:LongInt); процедурасы операциялык системага учурдагы датаны берет. Мында, Year-жыл, Month-айы, Day-күн.

SetFTime(var F; Time:LongInt); процедурасы түзүлгөн файлга дата-убакытты упаковка кылат.

SetTime(Hour, Minute, Second, Sec100:word); процедурасы операциялык системага учурдагы убакытты жүктөйт.

UnPackTime; процедурасы упаковкаланган дата-убакытты распаковкалайт.

Файлдар жана дисктер менен иштөөчү функциялар.

FExpand(Path:PathStr):PathStr; файлдын атын толукталганга чейин кеңейтет.

FSearch(Path:PathStr; DirList:string):PathStr; DirList-каталогдун тизмеси.

Файлдар менен иштөөчү процедуралар.

FSplit(Path:PathStr; var Dir:DirStr; Name:NameStr; var Ext:ExtStr); файлдын атын 3кө бөлөт-1) жол көрсөтүү, 2) файлдын аты, 3) файлдын кеңейтилиши.

GetFAttr(F; Attr:word); файлдын атрибуттарын берет.

SetFAttr(var F; Attr:word); файлдын атрибуттарын установка кылат. F – файл, Attr-атрибуттар боюнча параметр.

Файлдар менен иштөө үчүн башка процедура жана функциялар.

ChDir учурдагы каталогду өзгөртөт.

ChDir(NewDir:string);

NewDir-учурдагы таблица турган каталогдун аты.

```
var
    DirName:string;
Begin
     ChDir(ParamStr(1));
      If IoResult<>0 then writeln('көрсөтүлгөн каталог жок');
      GetDir(0, DirName);
      writeln('учурдагы каталог жок', DirName);
readln;
end.
```

Eof функциясы true маанисин берет, эгер файлдын аягына жетсе, эгер жетпесе false маанисин алат.

Баяндалышы: **Eof(var F):boolean;**

```
var
      FLn, Fout:file of byte;
      B:byte;
begin
      Assign(FLn, ParamStr(1));
      Reset(FLn);
      Assign(FOut, ParamStr(2));
      Reset(FOut);
repeat
      Read(FLn, B);
      Write(Fout, B);
until Eof(FLn);
readln;
end.
```

DiskSize функциясы берилген дисктин өлчөмүн берет.

Баяндалышы: **DiskSize(D:byte):longint;**

Мында, Drive-түзүүчүнүн шарттуу номери.

Эгер D=0 болсо, анда учурдагы диск каралат, эгер D=1 болсо, анда диск А, D=2 болсо диск В, D=3 болсо диск С ж.б. Эгерде көрсөткүчү жашабаса DiskSize –1ди берет.

1)
```
uses dos, crt;
begin
writeln(disksize(3) div 1024, 'kbytes capacity');
readln;
end.
```

2)
```
uses dos;
var
      Drive:integer;
begin
      writeln('учурдагы каталогдун өлчөмү', DiskSize(0), 'байт');
Drive=3;
while DiskSize(Drive)<>-1 do
begin
writeln('дисктин өлчөмү', Chr(Ord('A')+Drive-1), ':', DiskSize(Drive), 'байт');
Drive:=Drive+1;
end;
```

```
readln;
end.
```
DiskFree функциясы берилген дисктен бош мейкиндиктин өлчөмүн берет.
Баяндалышы: **DiskFree(D:byte):longint;**
Эгер D=0 болсо, анда учурдагы диск, D=1-диск A, D=2-B, D=3-C болот.
```
1) uses dos, crt;
begin
      writeln(diskfree(3) div 1024, 'kbytes free');
readln;
end.
2) uses dos;
var
      Drive:integer;
begin
      writeln('учурдагы дисктеги бош',DiskFree'(0), 'байт');
Drive:=3;
while DiskFree(Drive)<>-1 do
begin
writeln('дискте', Chr(Ord('A')+Drive-1), ':бош', DiskFree(Drive), 'байт');
Drive:=Drive-1;
end;
readln;
end.
```
Кийирүү-чыгарууну башкаруучу процедуралар.
Append(var F:Text); процедурасы тексттик файлды ачат.
Assign(F; Name:string); процедурасы ички файл менен файлдык өзгөрүлмө Fтин байланышын түзөт.
BlockRead(var F:file; var Buf; N:word[;var Result:word]); процедурасы файлдан компонентти окуйт.
F-файлдык өзгөрүлмө, N-компоненттердин саны.
BlockWrite(var F:file;var Buf;N:word[;var Result:word]); процедурасы файлга компонент жазат.
ChDir(S:string); процедурасы учурдагы каталогду өзгөртөт.
Close(var F); процедурасы ачык файлды жабат.
Erase(var F); процедурасы файлды алып таштайт.
Flush(var F:Text); процедурасы тексттик файлга буфер бошотот.
GetDir(D:byte;var S:string); процедурасы издөөчүнүн учурдагы каталогун алуу.
MkDir(S:string); процедурасы каталог түзөт.
Read(var F, v1, [, v2,..., vn]); процедурасы типтештирилген файлдагы информацияны окуйт.
Readln([var F:text;] [v1, v2,..., vn]); процедурасы тексттик файлдагы информацияны окуйт.
Rename(var F; New Name:string); процедурасы ички файлдын атын өзгөртөт.
Reset(var F; [Sizeiword]); процедурасы мурда бар файлды ачат.
RmDir(S:string); процедурасы бош каталогду алып таштайт.
Seek (var F; Nom:longint); процедурасы файлдын керектүү компонентин настройкалайт.
Nom-номер компоненти, **F**-ар кандай өзгөрүлмө файл.

SetTextBuf (var F:Text; var Buf[;Size:word]); процедурасы тексттик файл үчүн чыгуу-кирүү буферин түзөт.

Truncate (var F;) процедурасы учурдагы позициядан башталган файлдын бөлүгүн алып таштайт.

Write ([var F:Text;][v1, v2,…, vn]); процедурасы файлга информация жазат.

Writeln ([var F:Text;][v1, v2,…, vn]); процедурасы тексттик файлга информация жазат.

Кийирүү-чыгарууну башкаруучу функциялар.

Eof [(var F)]:boolean; функциясы файлдын аягына чейин карап чыгат.

EOln[(var F:Text)]:boolean; функциясы тексттик файлдын акыркы жолчосу.

FilePos(var F):longint; функциясы файлдын учурдагы компонентинин номерин берет.

FileSize(var F):longint; функциясы учурдагы файлдын өлчөмүн берет.

LoResult:word; функциясы кийирүү-чыгаруу акыркы операциясынын жыйынтыгы.

SeekEof[(var F:Text)]:boolean; файлдын аягы.

SeekEOln[(var F:Text)]:boolean; тексттик файлдын жолчосунун аягы.

Динамикалык эс менен башкарылуучу процедуралар.

Dispose(var P[, <дестуктор>]); процедурасы динамикалык эсти бошотот.

FreeMem(var P; Size:word); процедурасы динамикалык өзгөрүлмөнү берилген өлчөмдө алып таштайт.

GetMem(var P; Size:word); процедурасы жаңы динамикалык өзгөрүлмө түзөт.

Mark(var P); процедурасы динамикалык эстин абалын фиксирлейт.

New(var P[, <конструктор>]); процедурасы жаңы динамикалык өзгөрүлмө түзөт.

Release(var P); процедурасы берилген абалга динамикалык эсти кайтарып берет.

Динамикалык эс менен башкарылуучу жана адрестик функциялар.

Assigned(var P:<указатель>):boolean; функциясы маанинин көрсөткүчүнүн барабардыгын текшерет.

CSeg:word; функциясы CS регистринин учурдагы маанисин берет.

DSeg:word; функциясы DS регистринин учурдагы маанисин берет.

MaxAvail:longint; функциясы динамикалык эстеги максималдык блоктун өлчөмүн берет.

MemAvail:longint; функциясы динамикалык эстин бардык бош областынын өлчөмүн берет.

Ofs(x):word; функциясы аргументтин сегментинин жайгашышы.

Ptr(Seg, Ofs:word):Pointer; функциясы көрсөткүчтүн адресин өзгөртөт.

Seg(x):word; функциясы аргументтин сегментинин адресин көрсөтөт.

Ар түрдүү арналыштагы процедуралар.

GetCBreak(Break:boolean); Ctrl+Break клавишаларынын комбинациясы менен иштөөнүн ыкмасын жүктөө.

GetVerify(var Verify:boolean); Ms-Dosто дискке жазуу операцияларын текшерүүдө желекченин абалын кайтарып берет.

Варианттар

1. Жалаң «окуу үчүн гана мүмкүн болгон» файлдарды көрүүнүн программасын түзгүлө.

2. Дисктеги бош орунду табуунун программасын түзгүлө.
3. «Негизги каталогду» табуунун программасын түзгүлө.
4. С дискинин өлчөмүн табуунун программасын түзгүлө.
5. А дискинин бош мейкиндигин аныктоочу программаны түзгүлө.
6. Берилген каталогдон 1-файлды издөөнүн программасын түзгүлө.
7. Түзүлгөн файлга дата жана убакытты берүүнүн программасын түзгүлө.
8. «Архивдик» файлдарды чыгаруунун программасын түзгүлө.
9. «Каалаган» файлдарды издеп табуунун программасын түзгүлө.
10. «Системалык» кийинки файлдарды издөөнүн программасын түзгүлө.
11. D дискиндеги бардык каталогдорду чыгаруунун программасын түзгүлө.
12. «.PAS» кеңейтилишиндеги бардык файлдарды эсептегиле.
13. Дисктеги 1-файлды издөөнүн программасын түзгүлө.
14. Винчестердин өлчөмүн көрсөтүүчү программаны түзгүлө.
15. Учурдагы датаны чыгаруунун программасын түзгүлө.
16. Учурдагы убакытты чыгаруунун программасын түзгүлө.
17. DOS тогу .EXE файлдарды чыгаруунун программасын түзгүлө.
18. Учурдагы каталогду өзгөртүүнүн программасын түзгүлө.
19. «Жашыруун(скрытый)» файлдарды табуунун программасын түзгүлө.
20. «Системалык» бардык файлдарды чыгаруунун программасын түзгүлө.

Лабораториялык иш №23

Тема: ***Экранды графикалык режимге алып келүү***

Кадимки режим экран үчүн тексттик б.э. Ал эми экранды графикалык режимге алып келүү үчүн Graph модулунун InitGraph процедурасын колдонобуз.

InitGraph(GD,GM,Path)-экранды графикалык режимге алып келет. GD-драйвердин номери, GM-режимдин номери, Path-керектүү драйверди кармап турган файлга жол ачуу. Эгер Path өзгөрүлмөсү бош жолчону кармаса (Path=' '), анда драйвер учурдагы каталогдон изделет. GD жана GM өзгөрүлмөлөрү параметрлер деп аталышат. Эгер InitGraph чыгарылганда GD=0 болсо, анда керектүү драйвер жана оптималдуу графикалык режим бул драйвер үчүн автоматтык түрдө аныкталат. **CloseGraph** процедурасы графикалык режимди жабат.

```
M:      uses Graph;
        var
        GDriver, GMode:integer;
        begin
              GD:=Detect; {Detect=0}
        InitGraph(GD,GM,'c:\bp\bgi');
        readln;
        CloseGraph;
        end.
```

InitGraph процедурасы кээде себептер менен өзүнүн жумушун нормалдуу аткара албай калышы мүмкүн. М, эгер керектүү драйверди кармап турган файлга жол туура эмес көрсөтүлсө же оперативдик эс

графика үчүн жетишпесе. Ал үчүн бул учурда GraphResult функциясын колдонобуз. Текшерүүдө каталардын номери чыгат, ошол номерлер боюнча кандай ката экенин билип алууга болот.

0-катасы жок; 2-графикалык режим орнотулган эмес; 3- мындай драйверлүү файл табылган жок; 4- туура эмес драйвер; 5- графика үчүн оперативдик эс жетишпейт дегенди түшүндүрөт.

```
uses Graph;
var
Gd, Gm,ErrCode:integer;
begin
Gd:=Detect;
InitGraph(Gd, Gm, '       ');
ErrCode:=GraphResult;
IfErrCode <> 0 then
writeln (' графиканын инициализацияланышындагы катанын
номери', ErrCode) else {катасы жок}
CloseGraph;
End.
x=639; y=479
```

Чекит, түз сызык, түс.

Graph модулунда 80ге жакын процедура жана функция бар. Алардын жардамында чекит, эллипс, тик бурчтук, көп бурчтук ж.б. ар түрдүү фигураларды сызууга, боёого жана жылдырууга болот. Координаталар системасынан баштайбыз. Экрандын ар бир чекити өзүнүн координатасына ээ.

Төмөнкү сол бурч (0;0) координатасындагы чекит; X координатасы солдон оңго өсөт, Y координатасы төмөндөн жогору карай өсөт.

PutPixel(x,y,color) процедурасы (x, y) координатасы боюнча түс менен берилген чекитти тартат. Экрандын ортосунун координатасы (320,240);

Ал эми **GetPixel(x,y)** функциясынын жардамында тескерисинче, берилген координатта чекит кандай түстө экендиги аныкталат.

GetPixel (x, y) функциясы (x, y) координатасы менен берилген чекиттин түсүнүн маанисин берет.

Line (x1, y1, x2, y2) процедурасы (x1, y1) чекитинен (x2, y2) чекитине түз сызык сызат.

Circle (x, y, radius) – борбору (x, y) болгон, радиусу менен берилген айлана сызат.

Rectangle (x1, y1, x2, y2) – тик бурчтук сызат.

Setcolor (color) – фигураны тартууда учурдагы түстү аныктайт.

Эгер SetColor процедурасында түсү берилбесе, анда учурдагы түс ак түстө болот.

FillEllipse (X, Y, XRadius, YRadius) штрих, түс менен толтурулган эллипс сызат.

Төмөндө айлана сызуунун программасы көрсөтүлгөн.

```
uses Graph;
var
     Gd, Gm, i, j :integer;
begin
     Gd:=detect;
     InitGraph (Gd, Gm, 'c:\bp\bgi');
```

```
        for i:=1 to 10 do
        for j:=1 to 20 do
        circle (200,300,150);          {circle(i*40, j*30,64)}
        readln;
        closegraph;
end.
```

GetMaxX(горизонталдык) жана **GetMaxY**(вертикалдык) функцияларын колдонобуз. Бул функциялар программада режимден көз каранды эмес.

```
uses Graph;
var
        Gd, Gm, i, j:integer;
begin
        Gd:=detect;
        Initgraph(Gd,Gm,'d:\tp\bgi');
        Rectangle(0,0,GetMaxX,GetMaxY);
        for i:=1 to 10 do
        for i:=1 to 20 do
        Rectangle(0, 0, GetMaxX, GetMaxY);
        readln;
        CloseGraph;
        end.
```

Лабораториялык иш №24

Тема: **Текст.**

Графикалык режимде растрдык шрифтти, андан сырткары бир нече вектордук шрифттерди колдонууга болот. Растрдык шрифт чекиттердин матрицасы менен, ал эми вектордук шрифт символдорду түзүүчү векторлордун катары менен берилет.

Растрдык символду чоңойтууда аны түзгөн чекиттерди ажыратууга болот, ал эми вектордук символду чоңойтууда чагылуунун сапаты өзгөрбөйт жана экрандын мүмкүндүк берүү ыкмасынан көз каранды.

Шрифттердин файлы **.CHR** кеңейтилишине ээ. Шрифтти жүктөөдө дал келүүчү файл же **.BGI** графикалык драйвер (InitGraph процедурасында берилет) жайгашкан каталогдо жайгашышы керек же учурдагы каталогдо болушу керек.

Шрифтти тандоо жана масштабдоо SetTextStyle процедурасынын жардамында ишке ашырылат.

SetTextStyle(Font, Direction, Size) – учурдагы шрифтти, тексттин чыгуу багытын жана символдун өлчөмүн(размерин) орнотот.

Font-шрифти, Direction – тексттин чыгуу багыты(солдон онго же төмөндөн жогору), Size – шрифттин размерин аныктайт.

Растрдык шрифттерди жана тексттин чыгуу багытын көрсөтүү үчүн төмөнкү константалар аныкталган:

```
Const
                                {шрифттер}
DefaultFont=0{8x8 стандарттык растрдык шрифт}
TriplexFont=1{вектордук шрифт}
SmallFont=2{ вектордук шрифт }
```

```
SansSerifFont=3{ вектордук шрифт }
GothicFont=4{ вектордук шрифт }
                    {тексттин багыты}
HorizDir=0{солдон оңго}
VertDir=1{төмөндөн жогору}
```

Растрдык шрифт үчүн нормалдуу размер Size=1, ал эми вектордук шрифт үчүн Size=4.

OutTextXY(X,Y,TextString) процедурасы (X, Y) чекитинен баштап TextString жолчосун чыгарат. TextString (X,Y) чекитинен башталат. Жолчо учурдагы шрифт, учурдагы багыты жана символдордун өлчөмү менен чыгарылат.

SetTextJustify(Horiz, Vert) процедурасы OutTextXY жана OutText процедуралары менен кийин колдонулуучу текстти автоматтык түрдө барабарлоону орнотот. Horiz – горизонталдык, Vert – вертикалдык барабарлоо.

Барабарлоону көрсөтүү үчүн константалар аныкталган:

```
Const
{горизонталдык барабарлоо үчүн колдонулат}
LeftText=0{солдон барабарлоо}
CenterText=1{ортосунан(борборунан) барабарлоо}
RightText=2 {ондон барабарлоо}
                {вертикалдык барабарлоо үчүн колдонулат}
BottomText=0{төмөндөн барабарлоо}
CenterText=1{ортосунан барабарлоо}
TopText=2 {жогорудан барабарлоо}
Program stud;
Uses
      Graph;
var
      GD,GM:integer;
begin
      Gd:=Detect;
Initgraph(GD,GM,'d:\tp\bgi');
SetTextJustify(CenterText, CenterText); {барабарлоону аныктоо}
SetTextStyle(TriplexFont, HorizDir,8); {шрифти, багыты жана өлчөмү}
{Окшош эле жазуу ар түрдүү түс менен бир аз өзгөрүп жайгашып чыгат}
SetColor(White);
OutT extXY(GetMaxX div 2, GetMaxY div 2,'student');
SetColor(LightBlue);
OutTextXY(GetMaxX div 2+2, GetMaxY div 2,'student');
SetColor(LightRed);
OutTextXY(GetMaxX div 2+3, GetMaxY div 2, 'student');
SetColor(LightRed);
OutTextXY(GetMaxX div 2+4, GetMaxY div 2, 'student');
SetColor(White); Readln; CloseGraph;
End.
```

Эскертүү! TriplexFont, SmallFont, SansSerifFont жана GothicFont вектордук шрифттери орусча символдорду албайт, кээде Borland фирмасынын «BGI Toolkit» программасында жаңы вектордук шрифттерди түзүүгө мүмкүндүк берүүчү шрифттердин редактору бар.

Варианттар

1. 8x8 өлчөмүндөгү стандарттык растрдык шрифт менен **«Ош мамлекеттик университети»** деген текстти түзгүлө.
2. Вертикалдык жогору багытталган **«50 жыл»** текстин вектордук кичинекей шрифт менен түзгүлө.
3. Солдон оңго карай ориентацияланган **«ПОВТАС»** текстин вектордук шрифт менен экрандын тең ортосунда 4 өлчөмүндө чыгаргыла.
4. **«ФКТ»** текстин сол жактан барабарлап, GothicFont шрифти менен чыгаргыла.
5. **«Студент»** текстин SansSerifFont шрифти боюнча экранда горизонталдык багыт боюнча оң тарапка чыгаруунун программасын түзгүлө.
6. Фамилия, атыңды жана атаңдын атын үч бурчтук формасында экрандын ортосуна чыгаргыла.
7. Айлананын ичине адресиңди чыгаруунун программасын түзгүлө.
8. **«Бактылуу балалык»** текстин 8 өлчөмүндө, экрандын оң жагынан TripLexFont вектордук шрифти боюнча түзгүлө.
9. **«Ата-эне»** текстин эллипстин ичине вертикалдык багытта экрандын тең ортосуна чыгаруунун программасын түзгүлө.
10. Горизонталдык сол багыттан ориентацияланган **«Ырыс алды ынтымак»** текстин чыгаргыла.
11. **«Заман»** текстин айлананын жаасы сыяктуу экрандын ортосуна, горизонталдык багыт боюнча чыгаруунун программасын түзгүлө.
12. Тик бурчтуктун диагоналына **«Студенттик күндөрүм»** текстин чыгаруунун программасын түзгүлө.
13. Имарат чийип, анын ичине **«Бул менин үйүм»** текстин чыгаруунун программасын түзгүлө.
14. SmallFont шрифтин пайдаланып, **«ОшМУда окуганыма сыймыктанам»** текстин горизонталдык багыт боюнча чыгаргыла.
15. **«Ош-2005»** текстин DefaultFont шрифти боюнча горизонталдык багытта экрандын төмөн жагына тик бурчтуктун ичине чыгаргыла.
16. **«Нооруз-2005»** текстин китепченин ичине жайгаштыруунун программасын түзгүлө.
17. Беш жылдыздын ичине **«Студенттик күндөрүм-унутулгус өмүрүм!»** текстин чыгаруунун программасын түзгүлө.
18. Алты бурчтуктун ичине **«Компьютердик класс»** текстин чыгаруунун программасын түзгүлө.
19. Титулдук барак түзүүнүн программасын түзгүлө.
20. **«Ош-3000»** текстин айлананын ичине жайгаштыруунун программасын түзгүлө.

Экрандын областтары.

GetImage, PutImage процедураларын жана ImageSize функциясын колдонуп экранга сүрөттөлүштөрдүн тик бурчтуу областтарын сактоо жана чыгаруу мүмкүнчүлүгүнө ээ болобуз.

ImageSize(X1,Y1,X2,Y2) функциясы экрандын берилген тик бурчтуу фрагментин сактоо үчүн керектүү эстин өлчөмүн байтта кайтарат.
(X1,Y1) фрагменттин сол жогорку бурчу, (X2,Y2) фрагменттин оң төмөнкү бурчу.

GetImage(X1,Y1,X2,Y2,Area) процедурасы Area эсинин областында сүрөттөлүштүн тик бурчтуу фрагментин сактайт. (X1,Y1) фрагменттин сол жогорку бурчу, (X2,Y2) оң төмөнкү бурчу.

PutImage(X,Y,Area,Mode) процедурасы сүрөттөлүштүн фрагментин экрандын берилген ордуна чыгарат. (X, Y)-Area эсинин областынан сүрөттөлүш көчүрүлүшү керек болгон экрандын областынын сол жогорку бурчу. Mode – экранга чагылдуунун чыгуу режими. Сүрөттөлүш экрандын берилген областына мурдагы кармалган нерселердин ордуна жөн эле бастырылат же алар менен өз ара байланышат.

Түрдүү чыгуу режимин баяндоо үчүн төмөнкү константалар колдонулат:

```
Const
      {PutImage процедурасы үчүн константалар}
      NormalPut=0; {катышкан сүрөттөлүштү алмаштыруу}
      XorPut=1; {исключающее ЖЕ (XOR)}
      OrPut=2; {логикалык ЖЕ (OR)}
      AndPut=3; {логикалык ЖАНА (AND)}
      NotPut=4; {логикалык тануу (NOT)}
```

NormalPut режиминде экранга чыгарылуучу ар бир чекит сүрөттөлүш ал областка бастырылышы керек болгон ошол чекитти жабат. Калган режимдерде бастырылуучу чекиттин түсүнүн экилик чагылдырылышын ар бир бити ошол чекиттин жуп бити менен өз ара байланышат. Жыйынтыгы болуп учурдагы эки чекиттен айырмаланган түс келип чыгат.

```
Uses
      Crt,Graph;
Var
      GD,GM: integer;
      P: Pointer;
      i,size: integer;
begin
      GD:=Detect;
      InitGraph(GD, GM,'d:\tp\bgi');
      SetFillStyle(1,14);
      Bar(0,0,GetMaxX,GetMaxY);
      {(0,0,40,40) тик бурчтугунда сүрөттөлүш түзүлөт}
      SetColor(1);
      for i:=1 to 10 do
            Rectangle(20-2*i, 20-2*i, 20+2*i, 20+2*i);
      OutTextXY(80, 20, 'This is the image');
      {(0,0,40,40) фрагментин эстөө үчүн керектүү эстин өлчөмүн аныктайт}
      size:=ImageSize(0,0 ,40,40);
      {талап кылынуучу эстин областы бөлүнөт жана ага P көрсөткүчү
      адрестелет}
      GetMem(P,Size);
      {P көрсөткөн областка  (0,0,40,40) фрагменти сакталат}
      GetImage(0,0,40,40,p^);
      {NormalPut режиминде чыгаруу}
      for i:=1 to10 do
            PutImage(i*20,60,p^,NormalPut);
      OutTextXY(250,80,'NormalPut');
      {XorPut режиминде чыгаруу}
```

```
    for i:=1 to 10 do
    PutImage(i*20,100,p^,XorPut);
    OutTextXY(250,120,'XorPut');
    {OrPut режиминде чыгаруу }
    for i:=1 to 10 do
    PutImage(i*20,140,P^,OrPut);
    OutTextXY(250,160,'OrPut');
    {AndPut режиминде чыгаруу }
    for i:=1 to 10 do
        PutImage(i*20,180,P^,AndPut);
    OutTextXY(250,200,'AndPut');
    Readln;
    CloseGraph;
end.
```

Палитра.

Палитра-бул экранда реалдуу түрдө пайда болгон түстөрдүн жана түстөрдүн номеринин ортосундагы дал келүүчүлүк.

SetPallette(Col1, Col2) процедурасы Col2де көрсөтүлгөн Col1 номери менен палитранын түсүн орнотот.

```
uses graph;
var
    gd, gm:integer;
begin
    gd:=detect;
    InitGraph(gd,gm,'d:\tp\bgi');
    SetPallette(black,blue);
    readln;
    SetPallette(black, red);
    readln;
    closegraph;
end.
```

SetAllPaliette(Pallette) процедурасы бир убакытта палитранын бардык түстөрүн орнотот. Pallette адреси боюнча палитраны баяндоочу областы жайгашышы керек, биринчи байтта палитранын узундугу көрсөтүлүп изи боюнча түстөр жайгашышат. Palette өзгөрүлмөсүн PaletteType тибинде аныктоо,ыңгайлуу.

```
type
    PalletteType=record
    Size:Byte;
    Colors:Array[0..MaxColors] of ShortInt;
end;
```

MaxColors-15 ке барабар болгон аныкталган константа.

Төмөнкү программада SetAllPallette процедурасынын жардамында түстөр циклдүү түрдө бири-бирине өтөт.

```
uses crt, graph;
var
    gd,gm,i:integer;
    P:PaletteType;
begin
    gd:=detect;
```

```
    initgraph(gd,gm,'c:\bp\bgi');
    Bar(0,0,GetMaxX,GetMaxY);
    {палитра үчүн түстөрдү орнотот}
    for I:=0 to MaxColors do P.Colors[i]:=i;
    P.Size:=MaxColors;
    {актан сырткары бардык түстөрдү удаалаш сүрөттөйт}
    for i:=0 to MaxColors-1 do
    begin
        SetFillStyle(1,i);
        Bar(50+i*10,0,50+(i+1)*10,GetMaxY);
    end;
    {түстөр циклдүү алмаштырылат}
    repeat
        for i:=0 to MaxColors-1 do
        P.Colors[i]:=(P.Colors[i]+1) mod MaxColors;
        SetAllPalette(P);
    until keypressed;
    readln;
    closegraph;
end.
```

SetRGBPalette(Col,R,G,B) процедурасы Col номери менен туура келүүчү R, G жана Вга кызыл, жашыл жана көк түстөрүн өзгөртөт. Бул палитра VGA жана IBM 8514 адаптерлеринде иштейт.

```
uses graph;
var
    gd,gm:integer;
begin
    gd:=detect;
    initgraph(gd,gm,'c:\bp\bgi');
    {кара экран}
    readln;
    {ак-кызгымтыл түс}
    SetRGBPalette(black,55,45,45);
    readln;
    {кырмызы түс-көк жана кызыл түстү элестетет}
    SetRGBPalette(black,30,5,30);
    readln;
    closegraph;
end.
```

Лабораториялык иш №25

Тема: **Graph модулунун процедуралары жана функциялары.**

Arc (X, Y:integer; StrAng, EndAng, Radius : word); процедурасы айлананын жаасын сызат. Мында, X,Y- айлананын борборунун координаттары; StAng, EndAng- баштапкы жана акыркы бурч; Radius- радиусу.

Мисал:

```
    uses crt, graph;
    var
```

```
        Gd, Gm, I:integer;
begin
        Gd:=Detect;
        InitGraph(Gd, Gm, 'd:\tp\bgi');
        SetBkColor (LightGray);
        for i:=1 to 200 do
        begin
            SetColor (I div 15);
            Arc (GetMaxX div 2, GetMaxY div 2, I, I+300, I+10);
        end;
        readln;
        CloseGraph;
end.
```

Bar3D (X1, Y1, X2, Y2:integer; Depth: word; Top: Boolean) процедурасы учурдагы штрихтөөнү жана учурдагы толтуруу түсүн колдонуп параллелепипед сызат. x1, y1, x2, y2- координаталары, Depth өзгөрүлмөсү параллелепипеддин терендигин аныктайт. Top логикалык өзгөрүлмөсү жогорку гранды чийүү же чийбөө керектигин аныктайт. Эгерде Top өзгөрүлмөсү true(чын) маанисин алса, анда жогорку грань чийилет, эгерде Top өзгөрүлмөсү false(жалган) маанисин алса, анда жогорку грань чийилбейт.

```
uses crt, graph;
const
        Heigth=10;
        Width=10;
        Depth=20;
var
        Gd:=Detect;
begin
        Gd;=Detect;
        InitGraph (Gd, Gm, 'd:\bp\bgi');
        SetColor(LightRed);
        for i:=1 to 50 do
    Bar3D (I*Width, I*Heigth, (I+I)*Width, (I+I)*Heigth, Depth, TopOn);
        readln;
        CloseGraph;
end.
```

CloseGraph- процедурасы графикалык режимден чыгат.

DrawPoly (Num Points:word; var PolyPoints)- процедурасы көп бурчтук сызат. NumPoints- көп бурчтуктун чокусунун саны, PolyPoints- чекиттин координаталары эсептелген жерде массив болушу мүмкүн.

```
uses Graph ;
conts
        N=100; {көп бурчтуктун чокуларынын саны}
var
        Gd,Gm,I:integer;
        Poly: array [1..100] of PointType;
begin
        Gd:=Detect;
        InitGraph (Gd, Gm, 'c:\bp\bgi');
```

```
        for i:=1 to n do
    begin
        Poly[I].X:= Random(GetMaxX);
        Poly[I].Y:= Random(GetMaxY);
    end;
        DrawPoly (n, Poly);
    readln; CloseGraph;
end.
```

Ellipse (X, Y:integer, StAngle, EndAngle:word; XRadius, YRadius:word); процедурасы эллипстин жаасын сызат.

X, У- борборунун координаталары, StAngle, EndAngle- жаанын баштапкы жана акыркы бурчтары; XRadius жана YRadius- бийиктиги жана туурасы.

```
    uses crt, graph;
    var
        Gd, Gm, X, Y:integer;
    begin
        Gd:=Detect;
        InitGraph (Gd, Gm, 'c:\bp\bgi');
        SetColor (blue);
        X:=GetMaxX div 2;
        Y:=GetMaxY div 2;
        repeat
        X:=X+Random(5)-2;
        Y:=Y+Random(5)-2;
        Ellipse (X,Y,340, 200, 50, 40);
        until Keypressed;
        readln; CloseGraph;
    end.
```

FillPoly (NumPoins:word; var PolyPoints); процедурасы боелгон көп бурчтук сызат.

FloodFill (X, Y:integer; Border: word); процедурасы чектелген областка түс берет. X, Y- координаталары, Border- кайсы түс чек ара экендигин көрсөтөт.

```
    uses Graph;
    var
        Gd, Gm :integer;
    begin
        Gd:=Detect;
        InitGraph (Gd, Gm, 'd:\bp\bgi');
        SetColor(LightRed);
        Circle(100, 100, 80);
        Circle (200, 100, 80);
        SetFillStyle(I, Cyan);
        FloodFill (150, 100, LightRed);
        readln; CloseGraph;
    end.
```

GetArcCoords (var ArcCoords:ArcCoodsType); процедурасы Arc процедурасынын акыркы координатасын сызат.

```
type
```

```
        Arc CoordsType=record.
        X, Y, XStart, YStart, XEnd, YEnd: integer;
        end;
```
X, Y- айлананын борборунун координаталары, XStart, YStart- баштапкы координаталары, XEnd, YEnd- жаанын акыркы чекити.
```
        uses Graph;
        var
              Gd, Gm, I, J, Angle:integer;
              ArcCoord: ArcCoordsType;
        begin
              Gd:=Detect;
              InitGraph (Gd, Gm, 'c:\bp\bgi');
              Angle:=30;
              for I:=1 to 12 do
              for J:=1 to I do
        begin
              Arc (GetMaxX div 2, GetMaxY div 2, I*Angle, J*Angle,100);
              GetArcCoords (ArcCoord);
  Line(ArcCoords.XStart,ArcCoord.YStart,ArcCoord.XEnd, ArcCoord.YEnd);
        end;
        readln; CloseGraph;
end.
```
GetColor: word; функциясы учурдагы түстү берет.
GetGraphMode: integer; функциясы учурдагы графикалык режимди берет.
GetImage (x1, y1, x2, y2:integer; var Area) процедурасы экранда берилген областтын образын сактайт.
PutImage (x, y, Area, Mode); процедурасы фрагменттин сүрөттөлүшүн экрандын берилген ордунда чыгарат. Mode- чыгуу режими.
```
        uses crt, graph;
        var
              Gd, Gm,Size:integer;
              P:Pointer;
        begin
              Gd:=Detect;
              InitGraph (Gd, Gm, 'd:\tp\bgi');
              SetFillStyle (10, LigthGreen);
              Bar (0, 0, 40, 40);
              Rectangle (0, 0, 40, 40);
              Size:=Imaje Size (0, 0, 40, 40);
              GetMem (P, Size);
              GetImage (0, 0, 40, 40, P^);
repaet
        PutImage (Random(GetMaxX), Random(GetMaxY), P^, NormalPut);
        until Keypressed;
        readln;
        CloseGraph;
end.
```
GetMaxColor:word; функциясы SetColor процедурасына берилүүчү түстүн максималдуу маанисин берет.
GetMaxX:integer; функциясы горизонталь боюнча чекиттин санын берет.

GetMaxY:integer; функциясы вертикаль боюнча чекиттин санын берет.
PutPixel(X, Y, Color); функциясы (X, Y) координатасы менен берилген түс менен толтурулган чекитти берет.

```
uses Graph;
var
      Gd,Gm:integer;
begin
      Gd:=Detect;
      InitGraph (Gd, Gm, 'd:\tp\bgi');
      PutPixel(GetMaxX div 2, GetMaxY div 2, LightGreen);
readln;
CloseGraph;
end.
```

GetPixel(X, Y:integer):word; функциясы берилген чекиттин түсүн берет.

```
uses crt, graph;
var
      Gd, Gm, X, Y:integer;
      J:longint;
begin
      Gd:=0;
      InitGraph(Gd, Gm, 'd:\tp\bgi');
      SetTextStyle(DefaulFont, HorizDir,7);
      SetColor(DarkGray);
      OutTextXY(50, 50, 'жадырап');
      OutTextXY(50, 150,'жаз келди');
      for I:=1 to 50000 do
begin
      X:=Random(GetMaxX);
      Y:=Random(GetMaxY);
      If GetPixel(X, Y)=DarkGray then PutPixel (X,Y,14);
      end;
      readln;
      CloseGraph;
end.
```

GraphErrorMsg(Code:integer):string; функциясы ката боюнча жолчону көрсөтөт.
ImageSize(x1, y1, x2, y2:integer):word; функциясы байттын санын берет.
InitGraph(var Driver, Mode:integer;Path:string); процедурасы графикалык режимди инициализациялайт.
LineRel(DX, DY:integer); процедурасы учурдагы көрсөткүчтүн жардамында берилген чекитке чейин кесинди сызат.

```
uses crt, graph;
var
      Gd, Gm, I:integer;
begin
      Gd:=0;
      InitGraph(Gd, Gm, 'd:\tp\bgi');
      repeat
      MoveTo(Random(GetMaxX), Random(GetMaxY));
      SetColor(random(16));
```

```
        for I:=1 to 100 do
        LineRel(Random(5)-2, Random(5)-2);
        until keypressed;
        readln;
        CloseGraph;
end.
```

LineTo(X, Y:integer) процедурасы учурдагы көрсөткүчтөн берилген чекитке чейин түз сызык сызат.

MoveRel(Dx, Dy:integer); процедурасы учурдагы көрсөткүчтү берилген позициядан баштап жайгаштырат.

MoveTo(X, Y:integer); процедурасы учурдагы көрсөткүчтөн берилген чекитке көчүрөт.

OutText(S:string); процедурасы учурдагы көрсөткүчтөн баштап жолчону сызат.

OutTextXY(S:string); берилген чекитген баштап жолчону чыгарат.

```
uses crt, graph;
var
        Gd, Gm, I:integer;
begin
        Gd:=detect;
        InitGraph(Gd, Gm, 'c:\bp\bgi');
        SetTextJustify(CenterText, CenterText);
        SetColor(LightCyan);
        SetTextStyle(SmallFont, HorizDir, I);
        SetWriteMode(XorPut);
        repeat
        for I:=1 to 30 do
begin
        SetUserCharSize(I, 2, I, 2);
        OutTextXY(GetMaxX div 2, GetMaxY.div 2+I, 'Бактылуу');
        Delay(20);
        OutTextXY(GetMaxX div 2, GetMaxY div 2+I, 'студенттик');
end;
        for I:=30  downto  1 do
begin
        SetUserCharSize(I, 2, I, 2);
        OutTextXY(GetMaxX div 2, GetMaxY div 2+I, 'күндөр');
        Delay(20);
        OutTextXY(GetMaxX div 2, GetMaxY div 2+I, '!');
end;
        until keypressed;
        readln; Closegraph;
end.
```

PieSlice(X, Y:integer; StAng, EndAng, Radius:word); сектор сызат.

```
uses crt, graph;
var
        Gd, GM, X, Y, Dx, Dy,W, H:integer;
```

{x, у борборунун координаталары Dx, Dy-горизонталдык жана вертикалдык кадамдар, W, H-бийиктигинин жана туурасынын жарымы}

```
        procedure Click;
```

```
begin
      Sound(1000);
      Delay(5);
      Nosound;
end;
begin
      Gd:=detect;
      InitGraph(Gd, Gm,'c:\bp\bgi');
      SetWriteMode(XorPut);
      X:=100; Y:=4; Dx:=4; Dy:=4; W:=15; H:=15;
      repeat
      X:=X+Dx; Y:=Y+Dy;
      If (X>GetMaxX) or (X<0) then begin click;  Dx:=-Dx; end;
      If (Y>GetMaxY) or (Y<0) then begin click;  Dy:=-Dy; end;
      Rectangle(X-W, Y-H, X+W, Y+H);
      PutPixel(X, Y, yellow);
      until KeyPressed;
      readln; CloseGraph;
end.
```

RestoreCRTMode; экрандын режимин кайра түзөт.

SetAllPallette(var Pallette); процедурасы палитранын бардык түсүн бир убакта өзгөртөт.

SetBkColor(Color:word); фондун учурдагы түсүн берет.

SetGraphMode(Mode:integer); процедурасы экранды тазалайт жана берилген графикалык режимди түзөт.

SetColor(Color:word); процедурасы учурдагы тартылган түс.

SetUserCharSize(x1, y1, x2, y2:word); процедурасы символдун бийиктигин жана туурасын вектордук шрифт үчүн түзөт.

SetWriteMode(Mode:integer); процедурасы сызык сызуунун режимин түзөт.

TextHeight(s:string):word; функциясы тексттин бийиктигин берет.

TextWidth(s:string):word; функциясы жолчонун туурасын берет.

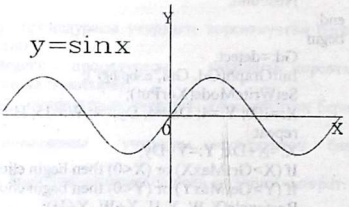
Мисал:

```
uses crt, graph;
var
      Gd, Gm, I:integer;
begin
      Gd:=detect; InitGraph(Gd, Gm, 'c:\bp\bgi');
      SetTextJustify(CenterText, CenterText);
      OutTextXY(GetMaxX div 2, GetMaxY div 2+I, 'жыл');
      Delay(1000); SetColor(LightCyan);
      SetTextStyle(SmallFont, HorizDir, 1);
      for i:=20  downto 15 do
      begin  SetUserCharSize(I, 2, I, 2);
            OutTextXY(GetMaxX div 2, GetMaxY div 2+I, 'мезгилдери');
      end;  readln; closegraph;
end.
```

Функциялардын графигин түзүү.

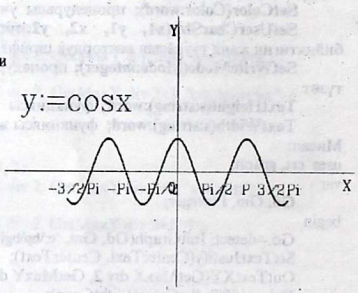
1) y=sinx функциясынын графиги

```
program sinusoida;
uses graph;
var
  Gd,Gm:integer;
  e,i,x,y:real;
begin
    Gd:=Detect;
    InitGraph(Gd,Gm,'c:\bp\bgi');
x:=-2*Pi;
SetBkColor(0); SetColor(15);
Line(0,240,640,240);
Line(320,0,320,470);
OutTextXY(630,245,'X');
OutTextXY(305,10,'Y');
OutTextXY(305,245,'0');
SetColor(14);
OutTextXY(60,50,'y=sinx');
while x<2*Pi do
begin
y:=sin(x);
PutPixel(Round(x*50)+320,round(-y*80)+240,4);
x:=x+0.01;
end;
readln;CloseGraph;
end.
```



2) y=cosx функциясынын графи

```
program gr_cos;
uses graph,dos,crt;
var
  a,b:pointer;
  gd,gm,k,l,x,y,n,size,i,t:integer;
begin
gd:=detect;
initgraph(gd,gm,'c:\bp\bgi');
setcolor(2); setfillstyle(1,1);
bar(0,0,640,480); setcolor(15);
line(320,0,320,480);
line(0,240,640,240);
x:=-200;
repeat
y:=trunc(cos(0.05*x)*50);
circle(x+320,-y+240,1); x:=x+1;
delay(10);
until x=200;
setcolor(red); settextjustify(1,1); settextstyle(8,0,3);
outtextxy(320,260,'-3/2Pi -Pi -Pi/2  Pi/2 P 3/2Pi');
outtextxy(200,180,'y:=cosx'); readln; closegraph;
end.
```



Варианттар

1. а) Концентрикалык айланаларды чыгаруунун программасын түзгүлө.
   б)$y=x^2$ функциясынын графигин тургузуунун программасын түзгүлө.
2. а) Трапеция сызуунун программасын түзгүлө.
   б)y=sinx функциясынын графигин тургузуунун программасын түзгүлө.
3. а) Кесилишкен эллипс сызуунун программасын түзгүлө.
   б)y=cosx функциясынын графигин тургузуунун программасын түзгүлө.
4. а) Айлананын жаасын чийүүнүн программасын түзгүлө.
   б) y=tgx функциясынын графигин тургузуунун программасын түзгүлө.
5. а) Тик бурчтукту чийүүнүн программасын түзгүлө.
   б) y=ctgx функциясынын графигин тургузуунун программасын түзгүлө.
6. а) Боелгон көп бурчтук чийүүнүн программасын түзгүлө.
   б) $y=x^4$ функциясынын графигин тургузуунун программасын түзгүлө.
7. а) Кубду түзүүнүн программасын түзгүлө.
   б) $y=x^2-2x+3$ функциясынын графигин тургузуунун программасын түзгүлө.
8. а) Квадрат чийүүнүн программасын түзгүлө.
   б) $y=a^x$ функциясынын графигин тургузуунун программасын түзгүлө.
9. а) Ромб сызуунун программасын түзгүлө.
   б) $y=\sqrt{x}$ функциясынын графигин тургузуунун программасын түзгүлө.
10. а) Параллелограмм чийүүнүн программасын түзгүлө.
    б) $y=\sqrt[3]{x}$ функциясынын графигин тургузуунун программасын түзгүлө.
11. а) Алты бурчтук сызуунун программасын түзгүлө.
    б) $y=|x|$ функциясынын графигин тургузуунун программасын түзгүлө.
12. а) Параллелопипед чийүүнүн программасын түзгүлө.
    б) y=sin2x функциясынын графигин тургузуунун программасын түзгүлө.
13. а) Сектор сызуунун программасын түзгүлө.
    б) y=cos3x функциясынын графигин тургузуунун программасын түзгүлө.
14. а) Пирамиданы түзүүнүн программасын түзгүлө.
    б) $y=\log_a x$ (a>1) функциясынын графигин тургузуунун программасын түзгүлө.
15. а) Тең капталдуу үч бурчтук сызуунун программасын түзгүлө.
    б) $y=\log_a x$ (0<a<1) функциясынын графигин тургузуунун программасын түзгүлө.
16. а) Конус чийүүнүн программасын түзгүлө.
    б) $y=x^3$ функциясынын графигин тургузуунун программасын түзгүлө.
17. а) Цилиндр чийүүнүн программасын түзгүлө.
    б) y=sin3x функциясынын графигин тургузуунун программасын түзгүлө.
18. а) Тең жактуу үч бурчтук чийүүнүн программасын түзгүлө.
    б) y=kx(k>0) функциясынын графигин тургузуунун программасын түзгүлө.
19. а) Үй чийүүнүн программасын түзгүлө.
    б) y=kx(k<0) функциясынын графигин тургузуунун программасын түзгүлө.
20. а) Эллипстин жааларын кокустук абалда ар түрдүү түс менен чыгаруунун программасын түзгүлө.
    б) $y=x^n$(n-жуп) функциясынын графигин тургузуунун программасын түзгүлө.

Колдонулган адабияттар

1. Файсман А. В. Профессиональное программирование на Турбо-Паскале Ташкент: «Инфомекс Корпорэйшн», 1992
2. Абрамов В.Г., Трифонов Н. П., Трифонова Г. Н. Введение в язык Паскаль. -М.: «Наука», 1988.
3. Фаронов В.В. Турбо Паскаль 7.0 –М.:2000., 416 стр.
4. Епанешников М.А., Епанешников В.А. Turbo Pascal 7.0 –М.:-2000., 367 стр.
5. Климова Л.М. Pascal 7.0 Практическое программирование. Решение типовых задач. -М.:2000., 528 стр.
6. Пильщиков В.Н. Сборник задач по языку Паскаль. –М.: «Наука», 1989.
7. Савельев А.Я. Языки программирования (Паскаль, ПЛҮМ).-М: «Высшая школа» 1987.
8. Грогоно П. Программирование на языке Паскаль. -М.: «Мир» 1982.
9. Вирт Н. Алгоритмы +структура данных=программы. -М.: «Мир» 1985.
10. Грехем Р. Практический курс языка Паскаль для микро ЭВМ. – М.: «Радио и связь», 1986.
11. Керниган Б., Плоджер Ф. Инструментальные средства программирования на языке Паскаль: Пер. с англ. –М.: «Радио и связь», 1985.
12. Уилсон И.Р., Эддимон М.А. Практическое введение В Паскаль: Пер.с англ. – М: «Радио и связь», 1983
13. Марченко А.И., Марченко Л.А. Программирование в среде Turbo Pascal 7.0 -Киев: «ВЕК», 2000.

Басууга берилди: 20.06.2005
Форматы: 60x84 1/16 Көлөмү: 7 б.т.
Буюртма № Нускасы: 300 экз.

« Кагаз ресурстары» ЖЧК
Ош шаары, Курманжан датка көчөсү 287, тел.: 2-52-50