ЎЗБЕКИСТОН АЛОҚА ВА АХБОРОТЛАШТИРИШ АГЕНТЛИГИ
ТОШКЕНТ АХБОРОТ ТЕХНОЛОГИЯЛАРИ УНИВЕРСИТЕТИ

«Ахборот технологиялари» факультети
«Информатика ва КГ » кафедраси

«Информатика»
фани бўйича маърузалар матни

С/С++ дастурлаш тили
2-қисм

Тошкент -2011

1-маъруза
C / C++ тилида массивлар.

Мақсад: C / C++ тилида массивлар устида амаллар бажариш кўникмасини ҳосил қилиш.
Калит сўзлар: массив, элемент, индекс, сатр, устун, символли массив.

Режа:

1. Массивлар хакида тушунча.
2. C / C++ тилида 1улчовли ва 2 улчовли массивлар билан ишлаш.
3. Массив элементларини киритиш ва чиқариш.
4. Массивлар устида амаллар бажариш.
5. Символли ва сўзлар массивларини ташкил этиш.
6. Кўрсаткичли массивлар билан ишлаш асослари.

1.1. Бир ўлчовли массивлар.

Массивларни таърифлаш.Бир ўлчовли массивлар.

Массив бу бир типли номерланган маълумотлар жамланмасидир. Массив индексли узгарувчи тушунчасига мос келади. Массив таърифланганда типи, номи ва индекслар чегараси курсатилади. Мисол учун long int a[5]; char w[200];double f[4][5][7]; char[7][200]. Массив индекслар хар доим 0 дан бошланади. Си тили стандарти буйича индекслар сони 31 тагача булиши мумкин, лекин амалда бир улчовли ва икки улчовли массивлар кулланилади. Бир улчовли массивларга математикада вектор тушунчаси мос келади. Массивнинг int z[3] шаклдаги таърифи, int типига тегишли z[0],z[1],z[2] элементлардан иборат массивни аниклайди.

Массивлар таърифланганда инициализация килиниши, яъни бошлангич кийматларлари курсатилиши мумкин. Мисол учун:

float C[]={1,-1,2,10,-12.5};

Бу мисолда массив чегараси автоматик аникланади. Агар массив инициализация килинганда элементлар чегараси курсатилган булса , руйхатдаги элементлар сони бу чегарадан кам булиши мумкин, лекин ортик булиши мумкин эмас.

Мисол учун int A[5]={2,-2}. Бу холда a[0] ва a[1] кийматлари аникланган булиб, мос холда 2 ва –2 га тенг.

Массивда мусбат элеменлар сони ва суммасини хисоблаш.

```
# include <iostream.h>;
# include <conio.h>;
Main() {
```

```
Int  x[]={-1;2;5;-4;8;9};
Clrscr();
For (int s=0,int k=0, int I=0; I<6; I++) {
If (x[I]<=0) continue;
k++;s++;  }
Cout<<k;
Cout<<s;
getch( ); };
```

Массивнинг энг катта, энг кичик элементи ва урта кийматини аниклаш:

```
 #include <iostream.h>
Void main( )
{
Int I,j,n;
Float a,b,d,x[100];
While(1)
{
Cout<<("\n n="); Cin>>n;
If ( n>0 && n < = 100 ) break;
Cout<<"\n Хато  0<n<101 булиши керак";
}
Cout<<"\n элементлар кийматларини киритинг:\n";
For (i=0;i<n;i++)
{ Cout<<"x["<<i<<"]="; Cin>>x[i];}
max=x[0];min=x[0];
For (s=0,i=0;i<n;i++)
{ s++;
  If (max<x[i]) max=x[i];
  If (min>x[i]) min=x[i]; }
s/=n;
Cout<<"\n max="<<max;
Cout<<"\n min="<<min;
Cout<<"\n урта киймат="<<s;  }
```

1.2. Икки ўлчовли массивлар. Жадваллар билан ишлаш

Икки улчовли массивлар математикада матрица еки жадвал тушунчасига мос келади. Жадвалларнинг инциализация килиш коидаси, икки улчовли массивнинг элементлари массивлардан иборат булган бир улчовли массив таърифига асослангандир.

Мисол учун икки катор ва уч устундан иборат булган хакиикий типга тегишли d массив бошлангич кийматлари куйидагича курсатилиши мумкин:

float d[2][3]={(1,-2.5,10),(-5.3,2,14)};

Бу езув куйидаги киймат бериш операторларига мосдир:

d[0][0]=1;d[0][1]=-2.5;d[0][2]=10;d[1][0]=-5.3;d[1][1]=2;d[1][2]=14;

Бу кийматларни битта руйхат билан хосил килиш мумкин:

float d[2][3]={1,-2.5,10,-5.3,2,14};

Инициализация ердамида бошлангич кийматлар аникланганда массивнинг хамма элементларига киймат бериш шарт эмас.

Мисол учун: int x[3][3]={(1,-2,3),(1,2),(-4)}.

Бу езув куйидаги киймат бериш операторларига мосдир:

x[0][0]=1;x[0][1]=-2;x[0][2]=3;x[1][0]=-1;x[1][1]=2;x[2][0]=-4;

Инициализация ердамида бошлангич кийматлар аникланганда массивнинг биринчи индекси чегараси курсатилиши шарт эмас, лекин колган индекслар чегаралари курсатилиши шарт.

Мисол учун:

Double x[][2]={(1.1,1.5),(-1.6,2.5),(3,-4)}

Бу мисолда автоматик равишда каторлар сони учга тенг деб олинади.

Куйидаги курадиган мисолимизда жадвал киритилиб хар бир каторнинг максимал элементи аникланади ва бу элементлар орасида энг кичиги аникланади:

```
#include <iostream.h>
void main()
{ double a[4,3]; double s,max=0.0,min=0.0;
int i,j;
for(i=0;i<4;i++) {
for(j=0;j<3;j++)
{ Cout<<“a[“<<i<<”][“<<j<<”]=”;Cin>>s;a[i,j]=s;
if (max<s) max=s;
};
Cout<<’\n’;
if (max<min) min=max;  }
Cout<<“\n min=”,min;  }
```

Массив элементларига сон қиймат беришда компьютер хотирасидаги тасодифий бутун сонлардан фойдаланиш ҳам мумкин. Бунинг учун стандарт кутубхонанинг rand ( ) функциясини ишга тушириш керак. rand ( ) функцияси ёрдамида 0 ÷ 32767 ораликдаги ихтиёрий сонларни олиш мумкин. Бу кийматлар умуман тасодифийдир. (псевдо – тасодифий дегани).

Агар дастур кайта-кайта ишлатилса, айти тасодифий кийматлар такрорланаверади. Уларни янги тасодифий кийматлар килиш учун srand ( ) функциясини дастурда бир марта эълон килиш керак. Дастур ишлаши жараёнида эхтиёжга караб rand ( ) функцияси чакирилаверади. Тасодифий кийматлар билан ишлаш учун <stdlib.h> файлини эълон килиш зарур. srand ( ) функциясидаги кийматни автоматик равишда узгарадиган холатга келтириш учун srand (time (NULL)) ёзиш маъкул, шунда компьютер ичидаги соатнинг киймати time ( ) функцияси ёрдамида урнатилади ва srand га параметр сифатида берилади. NULL ёки 0 деб ёзилса, киймат секундлар куринишида берилади. Вакт билан ишлаш учун <time.h> ни эълон килиш керак.

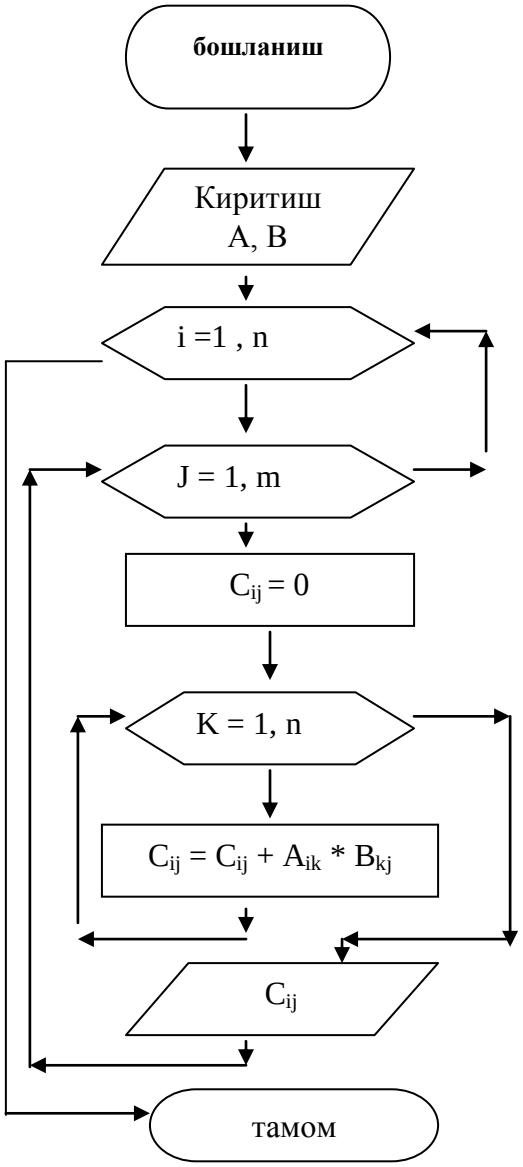
```
# include <iostream.h>
# include <conio.h>
```

```
# include <stdlib.h>
# include <time.h>
void main ( )
{ srand ( time (0));
int a[5], b[5], i;
for (i = 0; i < 5;  i++)     a[i] = rand ( );
for (i = 0; i < 5;  i++)
{  b[i] = a[i] + 64;
cout << "b="<<b[i]<<endl; }    getch ( ); }
```

Изох: тасодифий сонлар ичида манфий сонларнинг хам катнашишини ихтиёр этсак,

a[i] = 1050 - rand ( ); ёки a[i] = rand ( )-1000; деб ёзиш хам мумкин.

2-мисол. 2та матрица берилган. Уларни ўзаро кўпайтириб янги матрица ҳосил қилинг. Бу ерда 1-матрицанинг устунлар сони 2-матрицанинг сатрлар сонига тенг бўлиши керак.




```
# include <iostream.h>
# include <conio.h>
# include <stdlib.h>
# include <time.h>
void main ( )
{
{ srand ( time (0));
int a[3][3], b[3][3],c[3][3], i, j, k;
for (i=0; i<3; i++)
for (j=0; j<3; j++)
a[i][j] = rand ( );
for (i=0; i<3; i++)
for (j=0; j<3; j++)
b[i][j] = rand ( );
for (i=0; i<3; i++)
{ for (j=0; j<3; j++)
{  c[i][j] = 0;
for (k=0; k<3; k++)
c[i][j] = c[i][j] + a[i][k]*b[k][j];
cout <<"c="<<c[i][j]<<"\t"; }
 cout << endl; }
getch ( );}
```

1.3. Символли массивлар.

Си тилида сатрлар символли массивлар сифатида таърифланади. Символли массивлар куйидагича тасвирланиши мумкин: Char pas[10];

Символли массивлар куйидагича инициализация килинади:

Char capital[]=”TASHKENT”; Бу холда автоматик равишда массив элементлари сони аникланади ва массив охирига сатр кучириш ‘\n’ символи кушилади.

Юкоридаги инициализацияни куйидагича амалга ошириш мумкин:

Char capital[]={‘T’,’A’,’S’,’H’,’K’,’E’,’N’,’T’,’\n’};

Бу холда суз охирида ‘\n’ символи аник курсатилиши шарт.

Мисол учун палиндром масаласини куриб чикамиз. Палиндром деб олдидан хам охиридан хам бир хил укиладиган сузларга айтилади. Мисол учун нон. Дастурда киритилган суз палиндром эканлиги аникланади:

```
#include <iostream.h>
void main()
{
  gets(a);
for( int j=0, a[j]!=’\0’;j++);
I=0;
while(I<j)
if (a[I++]!=a[j--]) break;
if ((j-I)>1) Cout<<(“Палиндром эмас”) else Cout<<(“Палиндром”);
```

Кейинги мисолимизда киритилган суздан берилган харф олиб ташлаш дастури берилган:

```
#include <iostream.h>
void main()
{

char s[100];
int c;
cin>>s;
int i, j;

for ( i = j = 0; s[i] != '\0'; i++)
if ( s[i] != c )
s[j++] = s[i];
s[j] = '\0';

    cout<<s;
    }
```

Хар гал 'c' дан фаркли символ учраганда , у J позицияга езилади ва факат шундан сунг J киймати 1 га ошади. Бу куйидаги езувна эквивалент:

```
if ( s[i] != c )
        s[j] = s[i];
        j++;
```

Сузлар массивлари.

Си тилида сузлар массивлари икки улчовли символли массивлар сифатида таърифланади. Мисол учун:

Char Name[4][5].

Бу таъриф ердамида хар бири 5 та харфдан иборат булган 4 та сузли массив киритилади. Сузлар массивлари куйидагича инициализация килиниши мумкин:

Char Name[3][8]={“Анвар”,”Миркомил”,”Юсуф”}.

Бу таърифда хар бир суз учун хотирадан 8 байт жой ажратилади ва хар бир суз охирига ‘\0’ белгиси куйилади.

Сузлар массивлари инициализация килинганда сузлар сони курсатилмаслиги мумкин. Бу холда сузлар сони автоматик аникланади:

Char comp[][9]={“компьютер”,”принтер”,”картридж”}.

Куйидаги дастурда берилган харф билан бошланувчи сузлар руйхати босиб чикарилади:

```
#include <iostream.h>
void main()
{ char a[10][10];
   char c;
for (int i=0;i<10;i++) cin>>a[i];
cin>>c;
for (i=0;i<10;i++) if (a[i][0]==c) cin>>a[i];
}
```

Куйидаги дастурда фан номи, талабалар руйхати ва уларнинг бахолари киритилади. Дастур бажарилганда икки олган талабалар руйхати босиб чикарилади:

```
#include <iostream.h>
void main()
{ char a[10][10];
  char s[10];
  int k[10];
cin>>s;
for (int i=0;i<10;i++) cin>>a[i];
```

```
for (i=0;i<10;i++) {Cin>>k[i]};
for (int i=0;i<10;i++)  if (k[i]==2) cout<<a[i];
}
```

1.4. Курсаткичлар массивлари.

Курсаткичлар массивлари куйидагича таърифланади

<тип> *<ном>[<сон>]

Мисол учун int *pt[6] таъриф int типидаги объектларга олти элементли массивни киритади.

Курсаткичлар массивлари сатрлар масссивларини тасвирлаш учун кулайдир.

Мисол учун фамилиялар руйхатини киритиш учун икки улчовли массивдан фойдалани керак. char fam[][20]={“Олимов”,”Рахимов”,”Эргашев”}

Хотирада 60 элементдан иборат булади, чунки хар бир фамилия гача 0 лар билан тулдирилади.

Курсаткичлар массиви ердамида бу массивни куйидагича таърифлаш мумкин.

Char *pf[]= {“Олимов”,”Рахимов”,”Эргашев”}.

Бу холда руйхат хотирада 23 элементдан иборат булади, чунки хар бир фамилия охирига 0 белгиси куйилади

Курсаткичлар массивлари мураккаб элеменларни содда усулда тартиблашга имкон беради.

Куйидаги мисолда матрица сатрлари биринчи элементлари усиши тартибида чикарилади. Бу мисолда ердамчи курсаткичлар массиви яратилиб шу массив тартибланади ва массив асосида матрица элементлари чикарилади.

```
# include <iostream.h>
void main()
{int n=2;
 int m=3;
array[][3]={(1,3,5),(3,1,4),(5,7,1)};
int *pa[n];
for (i=0;i<n;i++) pa[I]=(int *)&a[i];

for (i=0;i<n-1;i++)
{for (int k=i=1;k<n;k++)
if a[i][1]>a[k][1]
{ int *pp=pa[i];
pa[i]=pa[k];pa[k]=pp;};

for (i=0;i<n;i++)
{Cout<<’\n’<<i+1;
for (int j=0;j<magistr;j++)
```

```
Cout<<pa[i][j];}
};
```

Курсаткичлар массивлари функцияларда матрицалар кийматларини узгартириш учун мумкин. Куйидаги мисолда матрицани транспонирлаш функцияси ишлатилади.

```
Void trans(int n,double *p[]);
{ double x;
for (int i=0;i<n-1;i++)
for (int j=i+1;j<n;j++)
{x=p[i][j];
p[i][j]=p[j][i];
p[j][i]=x;
}
};
void main()
{double a[3,3]={11,12,13,21,22,23,31,32,33};
double ptr={(double*)&a[0], (double*)&a[1], (double*)&a[2]};
int n=3;
trans(n,ptr);
for (int i=0;i<n;i++)
{Cout<<'\n'<<,i+1;
for (int j=0;j<n;j++)
Cout<<'\n'<<a[i][j]);
};
};
```

Назорат саволлари

1. Массив деб нимага айтилади?
2. c/c++ тилида массивларни кандай ифода этилади?
3. 1 улчовли массив элементларини киритиш ва чикариш усуллари.
4. 2 улчовли массив элементларини киритиш ва чикариш усуллари.
5. Массив элементларига сон киймат беришда тасодифий кийматлардан фойдаланиш
6. Массив элементларини натижа сифатида чикариш усуллари.
7. 1 улчовли массив элементларини кушиш алгоритми.
8. 1 улчовли массив элементларини купайтириш алгоритми.
9. 2 улчовли массив элементларини кушиш алгоритми.
10. 2 улчовли массив элементларини купайтириш алгоритми.
11. Массив элементларини йигиш ёки купайтириш алгоритми.
12. 1 улчовли массив элементларини саралаш алгоритми.
13. 2 улчовли массив элементларини саралаш алгоритми.
14. 2 улчовли массив элементларини сатр буйича саралаш алгоритми.
15. 2 улчовли массив элементларини устун буйича саралаш алгоритми.
16. Массивнинг изини хисоблаш алгоритми.
17. Символлардан иборат массивларни ҳосил қилиш.
18. Сўзлардан тузилган массивлар ва улар устида амаллар бажариш

Адабиётлар

1. Т.Х.Холматов ва бошқалар. “Информатика”, Тошкент, 2003
2. Р.Каримов ва бошқалар. “Дастурлаш”, Тошкент, 2003
3. Ш.Ш.Шохамидов. “Амалий математика элементлари”, Тошкент, 1997
4. Ашарина И.В. “Основы программирования на языках С и С++”, Москва, 2002
5. Павловская Т.А. «С /С++ программирование на языке высокого уровня», С.Петербург, 2001
6. В.В.Подбельский, С.С.Фомин. Программирование ва Си. Москва, 2004

ТЕСТЛАР

1. Массив элементларидан фойдаланиш нима оркали бажарилади:
   A) FIFO йуналиш
   B) LIFO йуналиш
   C) нукта операцияси
   D) элемент исми
   +E) элемент индекси

2. Массив нотугри таърифини курсатинг:
   A) int a[20];
   B) int a[]={1,2,3,4};
   C) int a[4]={1,2,3,4};
   +D) int a[2]={1,2,3,4};
   E) хаммаси тугри

3. Квадрат (n,n) матрицанинг кайси кисмини берилган фрагмент коди тулдиради?

```
for(i=0;i<n;++i)
for(j=n-i;j<n;++j)
a[i][j]=i+j+1;
```

   A) асосий диагональ элементларини
   B) асосий диагональ устидаги элементларни
   +C) асосий диагональ остидаги элементларни
   D) кушимча диагональ элементлари
   E) кушимча диагональ устидаги элементлари

4. Экранга нима чикади?

```
#include <string.h>
#include <iostream.h >
void main()
{
char a[] = "123";
cout<<strlen(a);
}
```

   A) 6
   +B) 3
   C) 4
   D) 8
   E) 8

5. Массив деб нимага айтилади?
+A). Элементларнинг тартибланган структураси

B). Элементларнинг тартибланмаган структураси
C) Элементларнинг ихтиёрий структураси
Д) Тўғри жавоб берилмаган.

7. C/C++ тилида неча ўлчовгача массивлар ишлатилиши мумкин?
A) 7тагача
+B) 31тагача
C) 3тагача
D) 2 тагача

8. C/C++ тилида массив индекси нечадан бошланади?
A) 1 дан
B) 7 дан
+C) 0 дан
Д) 2дан

9. Массив элементларига тасодифий кийматлар бериш учун кайси директива ишлатилади?
A) iostream.h
B) conio.h
C) math.h
+Д) stdlib.h

10. Массив элементларига тасодифий кийматлар бериш учун кайси функция ишлатилади?
A) sin ( )
B) srand ( )
+C) rand ( )
Д) NULL ( )

2-маъруза
Функциядан фойдаланиш асослари.

Мақсад: C / C++ тилида функциялар яратиш ва улардан фойдаланиш кўникмасини ҳосил қилиш.

Калит сўзлар: функция, соҳта параметр, ҳақиқий параметр, функцияга қиймат қайтариш, C / C++ тилида функция-процедура.

Режа:

1. Фойдаланувчининг функциясини яратиш асослари.
2. Функцияга параметрларни узатиш.
3. Функция параметрлари қийматини ўзгартириш. Процедуралар ҳосил қилиш.
4. Иловалардан фойдаланиш.
5. Рекурсия холати.

2.1 Фойдаланувчи Функциялари.

Функцияларни таърифлаш ва уларга мурожаат килиш. Функция таърифида функция номи, типи ва формал параметрлар руйхати курсатилади. Формал параметрлар номларидан ташкари типлари хам курсатилиши шарт. Формал параметрлар руйхати функция сигнатураси деб хам аталади.

Функция таърифи умумий куриниши куйидагичадир:

Функция типи функция номи(формал_параметрлар_таърифи)

Формал параметрларга таъриф берилганда уларнинга бошлангич кийматлари хам курсатилиши мумкин.

Функция кайтарувчи ифода киймати функция танасида **return** <ифода> ; оператори оркали курсатилади. Мисол:

```
Float  min(float, float b)
{ if (a<b) return a;
      return b; }
```

Функцияга мурожаат килиш куйидагича амалга оширилади:

Функция номи (хакикий параметрлар руйхати)

Хакикий параметр ифода хам булиши мумкин. Хакикий параметрлар киймати хисобланиб мос формал параметрлар урнида ишлатилади.

Мисол учун юкоридаги функцияга куйидагича мурожаат килиш мумкин:

Int x=5,y=6,z; z=min(x,y) еки int z=Min(5,6) еки int x=5; int z=min(x,6)

Функция таърифида формал параметрлар инициализация килиниши, яъни бошлангич кийматлар курсатилиши мумкин.

Функцияга мурожаат килинганда бирор хакикий параметр курсатилмаса, унинг урнига мос формал параметр таърифида курсатилган бошлангич киймат ишлатилади.

Мисол учун:

```
Float  min(float a=0.0, float b)
{ if (a<b) return a;
      return b; }
```

Бу функцияга юкорида курсатилган мурожаат усулларидан ташкари куйидагича мурожаат килиш мумкин:

Int y=6,z; z=min(,y) еки int z=Min(,6);

Агар функция хеч кандай киймат кайтармаса унинг типи **void** деб курсатилади.

Мисол учун:

```
Void print;
{ Cout<<("\n Salom!);
};
```

Бу функцияга Print; шаклида мурожжат килиш экранга Salom! Езилишига олиб келади.

Киймат кайтармайдиган функция формал параметрларга эга булиши мумкин:

```
Void Pint_Baho(Int baho);
{
Switch(baho)
{case 2:Cout<<("\n  емон");break;
case 3:Cout<<("\n  урта");break;
case 4:Cout<<("\n  яхши");break;
case 5:Cout<<("\n  аъло");break;
default: Cout<<("\n  бахо нотугри киритилган");
};
```

Бу функцияга Print_Baho(5) шаклида мурожаат килиш экранга аъло сузи езилишига олиб келади.

Агар программада функция таърифи мурожаатдан кейин берилса, еки функция бошка файлда жойлашган булса, мурожжатдан олдин шу функциянинг прототипи жойлашган булигши керак. Прототип функция номи ва формал параметрлар типларидан иборат булади. Формал параметрлар номларини бериш шарт эмас.

Мисол учун y=min(a,b)+2*max(c,d) ифодани хисоблашни курамиз:

```
#Include <iostream.h>
int max(int a,int b)
{if (a<b) return a;else return b};
void main()
```

```
{int a,b,c,d,y;
 int min(int ,int);
 Cin>>a>>b>>c>>d;
y=min(a,b)+2*max(c,d);
Cout<<("\n %f",y);
};
int min(int a,int b)
{if (a<b) return b;else return a};
```

Функцияга параметрлар узатиш. Функцияга параметрлар киймат буйича узатилади ва куйидаги боскичлардан иборат булади:

1. Функция бажаришга тайерланганда формал параметрлар учун хотирадан жой ажратилади, яъни формал параметрлар функцияларнинг ички параметрларига айлантирилади. Агар параметр типи float булса double типидаги объектлар хосил булади, char ва shortint булса int типидаги объектлар яратилади.
2. Хакикий параметрлар сифатида ишлатилган ифодалар кийматлари хисобланади.
3. Хакикий параметрлар ифодалар кийматлари формал параметрлар учун ажратилган хотира кисмларига езилади. Бу жараенда float типи double типига, char ва shortint типлари int типига келтирилади.
4. Функция танаси ички объектлар – параметрлар ердамида бажарилади ва киймат чакирилган жойга кайтарилади.
5. Хакикий параметрлар кийматларига функция хеч кандай таъсир утказмайди.
6. Функциядан чикишда формал параметрлар учун ажратилган хотира кисмлари бушатилади.

С тилида чакирилган функция чакирувчи функциядаги узгарувчи кийматини узнартира олмайди. У факат узининг вактинчалик нусхасини узгартириши мумкин холос.

Киймат буйича чакириш кулайлик тугдиради. Чунки функцияларда камрок узгарувчиларни ишлатишга имкон беради. Мисол учун шу хусусиятни акс эттирувчи POWER функцияси вариантини келтирамиз:

```
 power(x,n)
int x,n;
int p;

for (p = 1; n > 0; --n)
  p = p * x;
return (p);
```

Аргумент N вактинчалик узгарувчи сифатида ишлатилади. Ундан то киймати 0 булмагунча бир айрилади. N функция ичида узгариши функцияга мурожжат килинган бошлангич кийматига таъсир килмайди.

2.2 Функция параметрлари кийматларини узгартириш. Процедуралар.

Курсаткичлардан фойдаланиш. Функцияга параметрлар кийматлари узатилиши хакикий параметрлар кийматларини функция танасида узгартириш имконини бермайди. Бу муаммони хал килиш учун курсаткичлардан фойдаланиш мумкин.

Мисол учун туртбурчак юзи ва периметрини берилган томонлари буйича хисоблаш функциясини куйидагича тасвирлаш мумкин.

```
void pr(float a,float b, float* s, float* p)
{ *p=2(a+b);
  *s= a*b;
}
```

Бу функцияга куйидагича мурожаат килиниши мумкин pr(a,b,&p,&s). Функцияга p ва s узгарувчиларнинг адреслари узатилади. Функция танасида шу адреслар буйича 2*(a+b) ва a*b кийматлар езилади.

Кейинги мисолда берилган икки узгарувчининг кийматларини узаро алмаштириш функциясидан фойдаланилади:

```
Include <iostream.h>
Void swap(float* b,float* c);
{float  e;
e=*b;
*b=*c;
*c=e;
}
void main()
{float x,z;
Cin>>x;
Cin>>z;
swap(&x,&z);
Cout<<'\n'<<x<<'\n'<<z;  }
```

Иловалардан фойдаланиш. Иловалардан функция параметрлари сифатида фойдаланиш хакикий параметрлар кийматларини узгартиришга имкон беради. Курсаткичлардан устунлиги шундан иборатки адрес буйича киймат олиш автоматик бажарилади. . Натижада юлдузча (*) амалидан фойдаланилмайди.

Мисол учун туртбурчак юзи ва периметрини берилган томонлари буйича хисоблаш функциясини иловалардан фойдаланиб куйидагича тасвирлаш мумкин.

```
void pr(float a,float b, float & s, float & p)
{ p=2(a+b);
  s= a*b;  }
```

Бу функцияга куйидагича мурожаат килиниши мумкин pr(a,b,p,s).

Кейинги мисолда берилган икки узгарувчининг кийматларини узаро алмаштириш функциясидан фойдаланилади:

```
Include <iostream.h>
Void swap_values(float* b,float* c);
{float  e;
e=b;
b=c;
c=e;
}
void main()
{float x,z;
Cin>>x;
Cin>>z;
swap_values(x,z);
Cout<<'\n'<<x<<'\n'<<z;  }
```

2.3 Рекурсия.

Рекурсив функциялар. Рекурсив функция деб узига узи мурожжат килувчи функцияга айтилади. Мисол учун факториални хисоблаш функциясини келтирамиз:

```
Long fact(int k)
{if (k<0) return 0;
if (k==0) return 1;
return k*fact(k-1);
}
```

Манфий аргумент учун функция 0 киймат кайтаради. Параметр 0 га тенг булса функция 1 киймат кайтаради. Акс холда параметр киймат бирга камайтирилган холда функциянинг узи чакирилади ва узатилган параметрга купайтирилади. Функциянинг уз узини чакириш формал параметр киймати 0 га тенг булганда тухтатилади.

Кейинги мисолимизда ихтиерий хакикий соннинг бутун даражасини хисоблаш рекурсив функциясини келтирамиз.

```
Double expo(double a, int n)
```

```
{ if (n==0) return 1;
if (a==0.0) return 0;
if (n>0) return a*expo(a,n-1);
if(n<0) return expo(a,n+1)/a;
}
```

Мисол учун функцияга expo(2.0,3) шаклда мурожаат килинганда рекурсив равишда функциянинг иккинчи параметри камайган холда мурожжатлар хосил булади:

Expo(2.0,3),expo(2.0,2),expo(2.0,1),expo(2.0,0). Бу мурожаатларда куйидага купайтма хисобланади: 2.0*2.0*2.0*1 ва керакли натижа хосил килинади.

Шуни курсатиб утиш керакки бу функциямизда ноаниклик мавжуддир яъни 0.0 га тенг соннинг 0 чи даражаси 0 га тенг булади. Математик нуктаи назардан булса бу холда ноаниклик келиб чикади. Юкоридаги содда мисолларда рекурсиясиз итератив функциялардан фойдаланиш максадга мувофикдир.

Масалан даражани хисоблаш функцияни куйидагича тузиш мумкин:

```
Double expo(double a, int n)
{ if (n==0) return 1;
if (a==0.0) return 0;
int k=(n>0)?n:-n;
for(double s=1.0,int i=0;i<k;i++,s*=a);
if (n>0) return s else return 1/s;
}
```

Рекурсияга мисол сифатида сонни сатр шаклида чикариш масаласини куриб чикамиз. Сон ракамлари тескари тартибда хосил булади. Биринчи усулда ракамларни массивда саклаб сунгра тескари тартибда чикаришдир.

Рекурсив усулда функция хар бир чакирикда бош ракамлардан нусха олш учун уз узига мурожаат килади, сунгра охирги ракамни босиб чикаради.

```
printd(n)
int n;
 { int i;
  if (n < 0)  cout<<'-';
       n = -n;
  if ((i = n/10) != 0)
       printd(i);
  cout<<n % 10;   }
```

PRINTD(123) чакирикда биринчи функция PRINTD N = 123 кийматга эга. У 12 кийматни иккинчи PRINTD га узатади, бошкариш узига кайтганда 3 ни чикаради.

Назорат саволлари

1. Функция деб қандай структурага айтилади?
2. Функция нималардан ташкил топади?
3. Функциядан қандай чиқилади?
4. Функциянинг сохта параметрлари қандай аниқланади?
5. Функцияга мурожаат қилиш, унинг ҳақиқий параметрларини аниқлаш
6. Функцияларни эълон қилиш қоидалари.
7. Функция ичида параметрлар қийматларини ўзгартириш.
8. Функция – процедуралар хосил қилиш ва уларга мурожаат қилиш.
9. Рекурсия холати. Рекурсияни тўхтатиш.

Адабиётлар

1. Т.Х.Холматов ва бошқалар. “Информатика”, Тошкент, 2003
2. Р.Каримов ва бошқалар. “Дастурлаш”, Тошкент, 2003
3. Ш.Ш.Шохамидов. “Амалий математика элементлари”, Тошкент, 1997
4. Ашарина И.В. “Основы программирования на языках С и С++”, Москва, 2002
5. Павловская Т.А. «С /С++ программирование на языке высокого уровня», С.Петербург, 2001

ТЕСТЛАР

1. Агар continue оператори цикл оператори ичида келса, у холда:

+A) у бошкарувни циклнинг келаси итерациясининг бошланишига узатади

B) у бошкарувни циклнинг олдинги итерациясининг охирига узатади

C) у бошкарувни белгидан кейин келган цикл итерациясига узатади

D) у бошкарувни циклдан кейинги операторга узатади

E) у бошкарувни программа бошига узатади

2. Блокнинг ичида ифодаланган узгарувчи качон куринади?

A) ифодаланган нуктадан дастур охиригача

B) ифодаланган нуктадан функция охиригача

+C) ифодаланган нуктадан блок охиригача

D) блок ичида

D) блок ташкарисида

3. Дастурда кайси функция булиши шарт?

A) local()

+B) main()

C) friend()

D) global()

E) inline()

4. Библиотекали exit() функцияси нимадан чикиш учун мулжалланган:

A) узи жойлашган циклдан

B) узи жойлашган блокдан

C) узи жойлашган функциядан

+D) узи жойлашган дастурдан

E) узи жойлашган файлдан

5. Кийматни олувчи функция уз ичида кайси сузларни олиши шарт?

A) delete

B) new

C) void

D) break

+E) return

6. Кийматни олмаган функция кайси суз оркали ифодаланади?

A) delete

B) new

+C) void

D) break

E) return

7. Кушимча юкланган функциялар:

A) битта исмга эга булган функиялар топлами

B) улар бир хил аргументлар ва типларга эга

C) дастурлаш жараёнини осонлаштиради

D) огирликни кутара олмаслиги мумкин

+E) хамма жавоблар тугри

8. Бутун натижа кайтариб хакикий аргумент кабул килувчи функция сарлавхаси аниклансин.

+A) int func1 (float a);
B) int func1 (int a);
C) float (float a);
D) float func1 (float a);
E) long func1 (float a).

9. Бутун натижа кайтариб бутун аргумент кабул килувчи функция сарлавхаси аниклансин.

A) int func1 (float a);
+B) int func1 (int a);
C) float (float a);
D) float func1 (float a);
E) long func1 (float a).

3-маъруза
C/C++ да графика

Мақсад: C / C++ тилининг график имкониятларидан фойдаланиш.

Калит сўзлар: адаптер, драйвер, график режим, чизмалар, динамик хотира

Режа:

1. C/C++ тилининг график режимида ишлаш асослари.
2. Чизмаларни ҳосил қилувчи функциялар ва процедуралар
3. Бўяш ва чизиш усуллари ҳамда стиллари ҳақида
4. График режимида турли шрифтлардан фойдаланиш
5. Динамик хотира ва ундан фойдаланиш
6. Чизмаларни экран бўйлаб харакатлантириш

3.1. C/C++ тилининг график режимида ишлаш асослари

C/C++да график режимида ишлаш учун махсус graphics.h файли мавжуд. Бу директива узгармаслар, узгарувчилар ва турли кисм дастурлардан ташкил топган булиб, улар ёрдамида турли график адаптерлар билан хар хил тасвирлар чизиш мумкин. Адаптер компьютерда graphics.h файли билан ишлаш имкониятини яратадиган махсус курилмадир. График режимига утилганда экран алохида-алохида нукталарга булинади. Хар бир нукта уз координатасига эгадир.

Энг куп ишлатиладиган адаптерлар:

1. CGA - color graphics Adapter
2. MCGA - multi color graphics array
3. EGA - enhenced graphics Adapter
4. VGA - video graphics array .

Драйверларни курсатиш учун куйидаги узгармаслар ишлатилади:

Detect = 0 CGA = 1; MCGA = 2; EGA=3; VGA=9.

Матн режимидан график режимига утиш учун махсус процедурадан фойдаланилади: initgraph (&gd, &gm, “ path “); бу ерда:

gd - драйвер номи;

gm - режим номи;

Path - керакли драйвер файлининг йули. Купинча gd=0 деб олинади.

Драйверлар .bgi файлларида сакланади. Агар драйвер ишчи каталогнинг узида жойлашган булса, у холда Path = " " (буш белгиси) булади.

График режимидан яна матн режимига утиш керак булса, closegraph( ) функцияси ишлатилади.

3.2. Чизмаларни хосил килиш учун ишлатиладиган процедура ва функциялар

1. putpixel (x, y, color) - x ва y координатадаги нуктани color рангда чизиш;
2. getpixel (x, y) - x ва y координатадаги нуктанинг рангини аниклайди;
3. line (x1, y1, x2, y2) - x1 ва y1 координатадаги нуктадан x2 ва y2 координатадаги нуктагача кесма чизиш;
4. circle (x, y, r) - маркази x ва y координатада ва радиуси R булган айлана чизиш;
5. rectangle (x1, y1, x2, y2) - юкори чап нуктаси x1 ва y1 координатада, унг пастки нуктаси x2 ва y2 координатада булган тугритуртбурчакни чизиш;
6. setbkcolor (color) - орка фонга ранг бериш;
7. setcolor (color) - чизиш рангини урнатиш (рангли калам); Бу ерда color - ранг номери ёки номи. Агар ранг номи ёзиладиган булса, уни катта харфларда ёзилади.
8. bar (x1, y1, x2, y2) - жорий ранг ва чизиклар ёрдамида ичи буялган тугритуртбурчак чизиш;
9. fillellpse (x, y, xr, yr) - маркази x ва y да, xr кенгликда ва xr баландликда ичи буялган рангли эллипс чизади;
10. getmaxx - жорий режим ва драйверлар учун нукталар сонини аниклаш; getmaxy - жорий режим ва драйверлар учун вертикал нукталар сони. Бу процедура ёрдамида компьютернинг узи экрандаги максимал нукталар сонини аниклайди.
11. linerel (x, y) - x ва y координатали нуктадан жорий нуктагача кесма чизиш;

lineto (x, y) - жорий нуктадан x ва y координатали нуктагача кесма чизиш;

12. bar3D (x1, y1, x2, y2, h, top) - параллелопипед чизади. Бу ерда h - параллелопипеднинг узунлиги; top - юкори кисмини чизиш учун керак. Агар topon - булса томи бор, агар topoff - булса томи йук.
13. arc (x, y, a, b, r) - ёй чизиш учун. Бу ерда x ва y - марказнинг координаталари, a - бош бурчак, b - охириги бурчак, r - ёй радиуси. Бурчаклар градусда кабул килинади.
14. ellipse (x, y, a, b, xr, yr) - худди шу тартибда эллипс ёйини чизади.
15. drawpoly (n, p) - купбурчак чизиш учун. Бу ерда n - купбурчакнинг учлари сони; p -. Купбурчак учларининг координаталари.

3.3. Бўяш ва чизиш усуллари ҳамда стиллари ҳақида

1. setfillstyle (style, color) - буяш усул ва рангни урнатиш. Бу ерда style - узгармас катталик булиб, у куйидагича булиши мумкин:

0 - сохани фон ранги билан тулдириш;

1- сохани ранг билан узлуксиз тулдириш;

2 - калик горизонтал чизиклар
3 - ингичка огма чизиклар
4 - йугон огма чизиклар
5 - йугон огма чизиклар (бошка стил)
6 - огма йуллар
7 - туртбурчакли чизиклар
8 - огма туртбурчаклар
9 - зич огма шртихлар
10 - сийрак нукталар (у ер - бу ерда)
11 - зич нукталар билан

2. floodfill (x, y, color) - жорий ранг ва усулдан фойдаланган холда чегараланган сохани буяш. Бу ерда x ва y - шу сохага тегишли булган бирор нукта координатаси. Аввал ранг, кейин стили курсатилади. Масалан:

```
setcolor (4);            {кизил  рангли калам, чегара ранги}
setfillstyle (1, 2);     {1-стиль  билан яшил ранг билан буяш}
circle (50, 50, 35);     {радуси 35 булган айлана чизиш}
floodfill (50, 50, 4); {айлана ичига ранг тукиш, буяладиган чегара ранги рангли калам билан бир хил булиши керак}
```

3. setlinestyle (s, a, b) – турли стилдаги чизикларни чизиш учун; Бу ерда s- style номери; a –фойдаланувчи стилини яратиши мумкин булган параметр, одатда a=1 деб олинади; b- чизикнинг калинлигини курсатадиган параметр

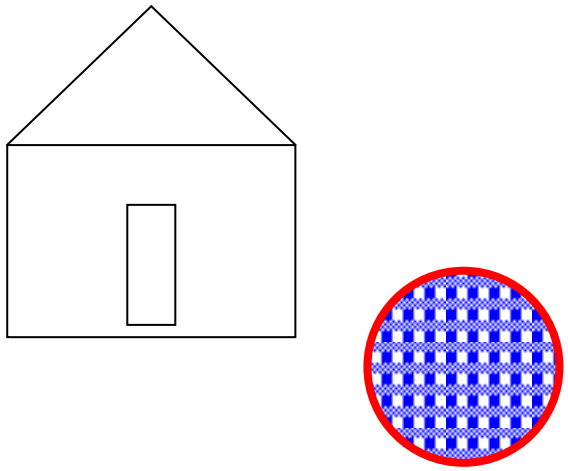
0 – оддий чизик;
1 – майда пунктир чизик;
2 – калин ва узунчок пунктир чизик
3 – юпка ва узунчок пунктир чизик;
4 – сийрак нуктали чизик.

1- Мисол:

```
# include <graphics.h>
# include <conio.h>
void  main ( )
{  int  i, j, gd, gm ;
gd= 0;
initgraph (&gd, &gm, " ");
setcolor (14);                                  //  сарик калам
for  ( i=0; i<=20; i++)
for  ( j=0; j<=20; j++)
circle (i*50, j*30, 55);                        //  сарик рангли айланалар
rectangle (0, 0, getmaxx, getmaxy);   //экран буйлаб тугри туртбурчак
setcolor (11);                                  //  тук феруза рангли калам
bar3d(200, 300, 100, 150, 30, topon);  //  параллелопипед, ичи ок
setcolor (CYAN);                                // оч феруза рангли калам
fillellipse (350, 360, 135, 90);             //эллипс, ичи ок рангда
getch( );
closegraph( );  }
```

2-мисол.

```
…….
void main ( )
{  gd=0;
initgraph (&gd, &gm, ' ');
   setbkcolor (BLUE);
   setcolor (14);
 rectangle (120, 130, 240, 250);
setcolor (6);
line (120, 130, 180, 80);
setcolor (2);
line (180, 80, 240, 130);
setcolor (14);
rectangle (160, 160, 200, 250);
setcolor (4);
setfillstyle(7, 9);
circle( 300, 300, 50);
floodfill (300, 300, 4);
getch( ); closegraph ( );   }
```



3.4. График режимида шрифтлар

График режими холатида турли шрифтлардан фойдаланиб матнларни хам ёзса булади. Шрифтлар .chr кенгайтмали файлларда сакланади. Улар .bgi файллари билан битта каталогда сакланиши шарт.

1. outtextxy (x, y, 'matn'); - матнни ёзиш; бу ерда x ва y матн бошланадиган нукта координаталари; масалан: outtextxy (10, 10, 'Mirzaev K. 212-07 Aty’);
2. settextstyle (sh, n, r); мант шрифтини урнатиш; бу ерда sh - шрифт номери (0 - векторли шрифт, 1 - стандарт шрифт); n - шрифт йуналиши (0 - чапдан унгга, 1 - куйидан юкорига ёзиш); r - шрифт размери (оддий шрифтда 1, векторли шрифтда 4 деб олинади);
3. settextjustify (h, v) - ёзилган каторни текислайди. У outtextxy процедурасидан кейин ёзилади. Бу ерда h - горизонтал текислаш; v - вертикал текислаш; Горизонтал текислаш учун: 0 - чапга; 1- марказга; 2 - унгга. Вертикал текислаш учун: 0 - пастга; 1 - марказга; 2 - юкорига.
4. setusercharsize - вектор шрифтлари учун бир хил символларнинг эни ва буйини урнатади. Масалан: setUserCharSize(x1, y1, x2, y2);

3-мисол. Функцияларнинг графикларини чизиш.

```
# include <graphics.h>
# include <conio.h>
# include <math.h>
void  main ( )
```

```
{  int  i, j, gd, gm ;  float x, y;
gd=0;   initgraph (&gd, &gm, "  ");
setcolor (14);
line (320, 0, 320, 480);
line (0, 240, 640, 240);
line (480, 0, 480, 235);
line (325, 120, 635, 120);
line (160, 245, 160, 475);
line (0, 360, 315, 360);
line (480, 245, 480, 475);
line (325, 360, 635, 360);

x =-10;  outtextxy(10, 20, ' y=sin(x) grafigi');
do
{ y = sin(x);
putpixel (160 + 10*x, 120 - y, 5);
 x = x+0.001;   }
while  (x<=10);

x1 = -10;  outtextxy(10, 20, ' y=cos(x) grafigi');
do
{ y = cos(x1);
putpixel (480 + 20*x1, 120 - 20*y1, 6);
 x1 = x1+0.001;   }
while  (x1 < =10);

x2 = -10;  outtextxy(10, 20, ' y=exp(x) grafigi');
do
{ y2 = exp(x2);
putpixel (160 + 10*x2, 360 - 20*y2, 7);
 x2 = x2+0.001;   }
while  (x2 <=10);

x3 = -10;  outtextxy(10, 20, ' y=ln(x) grafigi');
do
{ y3 = ln(x3);
putpixel (480 + 10*x3, 360 - y3, 8);
 x3 = x3+0.001;  }
while  (x3 <=10);  getch( );  closegraph( );   }
```

3.5. Динамик хотира ҳақида

include <grahpics.h> директивасининг яна шундай процедуралари мавжудки, улар ёрдамида чизмаларни экран буйлаб харакатга келтириш

мумкин. Фигураларни харакатга келтиришнинг бир неча усуллари бор. Улардан бири харакатни такрорланиш буйруги оркали ташкил килишдир. Иккинчи усул экранда чизилган чизма жойлашган сохани массив куринишида эслаб колиб, уни махсус процедура ёрдамида экраннинг керакли нуктасига кучиришдир. Бунда динамик хотирадан фойдаланилади.

Катта микдордаги маълумотлар ишлатиладиган масалаларни ечишда, компьютернинг график имкониятларидан фойдаланганимизда хотира хажми етишмаслиги мумкин. Бундай холларда динамик хотира жуда кул келади. Динамик хотира бу компьютернинг дастурга маълумотлар сегментидан ташкари юклатилган тезкор хотирадир. Бу хотира тахминан 200-300 Кбни ташкил килади. Динамик хотирадан фойдаланиш учун курсаткичлар ишлатилади. Бу узгарувчиларни (курсаткичларни) хотирада жойлаштиришни компилятор амалга оширади. Курсаткич шундай узгарувчики, унинг киймати узгарувчи кийматига эмас, балки шу узгарувчи жойлашган хотира адресига тенгдир.

Динамик хотира сохасидан жой ажратиш учун **new** оператори ишлатилади. Бу суздан кейин хотирага жойлаштириладиган объект типи аникланади. Масалан: new int; деб ёзсак, динамик хотирадан 2 байт жой ажратган буламиз. Масалан: int *p;

p = new int;

ёки int *p = new int ;

Ажратилган хотира сохасига бирор кийматни жойлаштириш мумкин: *p = 750 ;

Бу ёзувни куйидагича укилади: « p курсаткичида адреси сакланаётган хотирага 750 сонини ёзинг ».

Динамик хотира сохаси чегараланган, у тулиб колганда new оператори оркали жой ажратиш хатоликка олиб келади. Бу холни биз хотиранинг тулиб кетиши ёки окиб кетиши деймиз (утечка памяти). Шунинг учун хотира бошка керак булмаса уни бушатиш зарурдир. Буни **delete** оператори ёрдамида бажарилади. Масалан: delete p;

3.6. Экранда чизмаларни харакатлантириш

Экранда чизмаларни харакатлантириш учун керак буладиган процедуралар:

1.i**magesize (x1, y1, x2, y2)** – экраннинг чап юкори нуктаси ва унг пастки нуктаси координаталаридан тугритуртбурчакли сохани саклаш учун керак буладиган хотиранинг улчами (байтларда олинади);

2. g**etimage (x1, y1, x2, y2, p)** – динамик хотиранинг берилган p майдонида тугритуртбурчакли тасвирни саклаш. Бу ерда p – тасвир сакланадиган жойнинг адресини саклайдиган узгарувчи, яъни курсаткич.

3. **putimage (x, y, p, m)** – экраннинг берилган жойига тасвирни чикариш; бу ерда x ва y – хотиранинг p майдонидаги тасвирдан нусха кучириладиган экран майдонининг чап юкори нуктаси; m – тасвирни экранга чикариш режими. Агар:

m = 0 (NormalPut) - тасвирни кучириш. Бунда эскичи учиб, янгиси пайдо булади (худди юриб кетаётгандек)

m = 1 (XorPut)

m = 2 (Orput) –

m = 3 (AndPut)

Масалан: қуйидаги дастурда квадрат ичидаги бўялган айлана экран бўйлаб харакатланади.

```
# include < graphics.h >
# include <conio.h >
# include <dos.h >
void main ( )
{  int  gd = 0, gm, I, j, s;  int  *a;
  initgraph(&gd,&gm,"");
  setcolor ( 4 );
  circle ( 30, 30, 20 );  putpixel ( 30, 30, 2);
  rectangle ( 10, 10, 50, 50);
  s = imagesize ( 9, 9, 51, 51);
   *a = new  int;   *a = s;
  getimage (9, 9, 51, 51, a);
  for ( i = 0; i < = 585; i + +)
  { putimage ( i, 10, a, 0);  sound (20);   delay (10); nosound ( ); }
  for ( j = 10; j < = 420;  j + +)
  { putimage(585, j, a, 0 );  sound ( 30 );  delay (10);  nosound( );}
  for (i = 585;  i > = 10; i - -)
  { putimage(i, 420, a, 0);  delay ( 10 ); }
  for (j = 420;  j >10; j - -)
  { putimage(10, j, a, 0);  delay( 10 ); }
  delete  a;
  getch();  //closegraph( );
  }
```

Назорат саволлари.

1. C/C++ тилида график режимида ишлаш учун қандай директивалар керак бўлади?
2. Кўп ишлатиладиган адаптерлар ҳақида маълумот беринг.
3. Драйвер нима ва унинг вазифаси
4. Оддий чизмаларни ҳосил қилиш учун қайндай функция ва процедуралар мавжуд?
5. Нуқта, тўғри чизиқ, айлана, тўғри тўртбурчак, ичи бўялган тўртбурчак, эллипс, ёй, параллелопипед ва шу кабиларни чизиш структуралари
6. Соҳани бўяш структураси.
7. Чизиш стилларини ўрнатиш
8. Графикада матнларни ишлатиш асослари.
9. Функциянинг графигини чизиш.
10. Динамик хотира нима ва унинг вазифаси.
11. Динамик хотира учун жой ажратиш.
12. Динамик хотирани бўшатиш.
13. Экранда чизмаларни харакатлантириш учун керакли функция ва процедуралар.

Адабиётлар

1. Т.Х.Холматов ва бошқалар. “Информатика”, Тошкент, 2003
2. Р.Каримов ва бошқалар. “Дастурлаш”, Тошкент, 2003
3. Ш.Ш.Шохамидов. “Амалий математика элементлари”, Тошкент, 1997
4. Ашарина И.В. “Основы программирования на языках С и С++”, Москва, 2002
5. Павловская Т.А. «С /С++ программирование на языке высокого уровня», С.Петербург, 2001

ТЕСТЛАР

1. Нуқтани чизувчи функция:
A) circle (x,y);
B) line (x1,y1,x2,y2);
+C) putpixel (x,y,color);
Д) getpixel (x,y);

2. Айланани чизувчи функция:
+A) circle (x,y);
B) line (x1,y1,x2,y2);
C) putpixel (x,y,color);

Д) getpixel (x,y);

3. Кесмани чизувчи функция:
A) circle (x,y);
+B) line (x1,y1,x2,y2);
C) putpixel (x,y,color);
Д) getpixel (x,y);

4. Нуқтанинг рангини аникловчи функция:
A) circle (x,y);
B) line (x1,y1,x2,y2);
C) putpixel (x,y,color);
+Д) getpixel (x,y);

5. Буяш усули ва рангини урнатувчи функция:
+A) setfillstyle (a, b);
B) linerel (x, y);
C) lineto (x, y);
Д) floodfill (x,y);

6. Чегараланган сохани буяш функцияси:
A) setfillstyle (a, b);
B) linerel (x, y);
C) lineto (x, y);
+Д) floodfill (x,y, color);

7. График режимида матнларни киритиш функцияси:
A) drawpoly (n, p);
Б) setcolor (r);
C) arc (x, y, a,b, r);
+Д) outtextxy (x,y,m);

4-маъруза
Структуралар.

Мақсад: C / C++ тилида структуралар устида амаллар бажариш кўникмасини ҳосил қилиш.

Калит сўзлар: структура, рўйхат, структура элементи, дастурчининг типи.

Режа:

1. C/C++ тилида аралаш типли ўзгарувчилар – структураларни ҳосил қилиш. Структурали типлар.
2. Структуралар ва массивлар.
3. Структуралар ва кўрсаткичлар.
4. Структуралар ва функциялар.

4.1 Структурали типлар ва структуралар.

Структурали тип.

Структура бу турли типдаги маълумотларнинг бирлаштирилган типдир. Структура хар хил типдаги элементлар-компонеталардан иборат булади..

Структуралар куйидагича таърифланиши мумкин:

Struct струтурали_тип_номи

{Элементлар_таърифлари}

Мисол учун омбордаги молларни тасвирловчи структурани курамиз. Бу структура куйидаги компоненталарга эга булиши мумкин:

- Мол номи (char*)
- Сотиб олиш нархи (long)
- Устига куйилган нарх, фоизда (float)
- Мол сони (int)
- Мол келиб тушган сана (char[9])

Бу структура дастурда куйидагича таърифланади:

```
 struct goods
 {
char* name;
long price;
float percent;
int vol;
char date[9];
}
```

Конкрет структуралар ва структурага курсаткичлар бу тип ердамида куйидагича таърифланиши мумкин:

goods food, percon; **goods** *point_to;

Мисол учун:

```
struct complex
{
double real;
double imag;
}
```

Бу мисолда комплекс сонни тасвирловчи структурали тип complex киритилган булиб, комплекс сон хакикий кисмини тасвирловчи real ва мавхум кисмини тасвирловчи imag компоненталаридан иборатдир.

Конкрет структуралар бу холда куйидагича тасвирланади:

Complex sigma,alfa;

Куйидаги мисолда каср сонни тасвирловчи numerator –сурът ва denominator-махраж компоненталаридан иборат структура таърифи келтирилган.

```
struct  fraction;
{
  int numerator;
  int denominator;
}
```

Бу холда конкрет структуралар куйидагича тасвирланиши мумкин:

fraction beta;

Конкрет структураларни тасвирлаш.

Структуралар таърифланганда конкрет структуралар руйхатини киритиш мумкин:

Struct струтурали_тип_номи
{Элементлар_таърифлари}
Конкрет_структуралар_руйхати.

Мисол:

```
Struct student
{
char name[15];
char surname[20];
int year;
} student_1, student_2, student_3;
```

Бу холда student структурали тип билан бирга учта конкрет структура киритилади. Бу структуралар студент исми (name[15]), фамилияси (surname[20]), тугилган йилидан (year) иборат.

Структурали тип таърифланганда тип номи курсатилмай, конкрет стьруктуралар руйхати курсатилиши мумкин:

Struct
{Элементлар_таърифлари}
Конкрет_структуралар_руйхати.

Куйидаги таъриф ердамида учта конкрет структура киритилади, лекин структурали тип киритилмайди.

```
struct {
  char processor [10];
  int frequency;
int memory;
int disk;
} IBM_486, IBM_386, Compaq;
```

Структуралар учун хотирадан жой ажратиш.

Структурали тип киритилиши бу тип учун хотирадан жой ажратилишига олиб келмайди. Хар бир конкрет структура (объект) таърифланганда, шу объект учун элементлар типларига караб хотирадан жой ажратилади. Хотирадан жой зич ажратилганда структура учун ажратилган жой хажми хар бир элемент учун зарур булган хотира хажмлари йигиндисига тенг булади. Шу билан бирга хотирадан жой зич ажратилмаслиги хам мумкин яъни элементлар орасида буш жойлар хам колиши мумкин. Бу буш жой кейинги элементни хотира кисмларининг кабул килинган чегаралари буйича текислаш учун колдирилади. Мисол учун бутун типдаги элементлар жуфт адреслардан бошланса, бу элементларга мурожаат тезрок амалга оширилади. Конкрет структураларни жойлашувига баъзи компиляторларда #pragma препроцессор директиваси ердамида таъсир утказиш мумкин. Бу директивадан куйидаги шаклда:

Pragma pack(n)

Бу ерда n киймати 1,2 еки 4 га тенг булиши мумкин.

Pack(1) – элементларни байт чегаралари буйича текислаш;

Pack(2) – элементларни сузлар чегараларига караб текислаш;

Pack(4) – элементларни иккиланган музлар чегараларига караб текислаш.

Структура учун ажратилган жой хажмини куйидаги амаллар ердамида аниклаш мумкин:

Sizeof (структурали_тип_номи);

Sizeof (структура_номи);

Sizeof структура_номи.

Охирги холда структура номи ифода деб каралади. Ифоданинг типи аникланиб, хажми хисобланади.

Мисол учун:

Sizeof (struct goods)

Sizeof (tea)

Sizeof coat

Структураларга мурожаат.

Конкрет структуралар таърифланганда массивлар каби инициализация килиниши мумкин. Масалан

complex sigma ={1.3;12.6};

goods coats={“пиджак’,40000,7.5,220,”12.01.97”);

Бир хил типдаги структураларга киймат бериш амалини куллаш мумкин:

Complex alfa; alfa=sigma;

Лекин структуралар учун солиштириш амаллари аникланмаган.

Структуралар элементларига куйидагича мурожаат килиш мумкин:

Структура номи.элемент_номи.

‘Нукта амали’ структура элементига мурожаат килиш амали дейилади. Бу амал кавс амаллари билан бирга энг юкори устиворликка эгадир.

Мисол:

Complex alfa={1.2,-4.5},betta={5.6,-7.8),sigma;

Sigma.real=alfa.real+betta.real;

Sigma.imag=alfa.imag+betta.imag;

Конкрет структуралар элементлари дастурда алохида киритилиши ва чикарилиши зарурдир. Куйидаги мисолда икки комплекс сон кийматлари киритилиб, йигиндиси хосил килинади:

```
#include  <iostream.h>
typedef struct {
double real;
double imag;
} complex;
void main()
{
complex x,y,z;
Cout<<’\n’; Cin>>x.real;
Cout<<’\n’;Cin>>x.imag;
Cout<<’\n’;Cin>>y.real;
Cout<<’\n’;Cin>>y.imag;
z.real=x.real+y.real;
z.imag=x.imag+y.imag;
Cout<<’\n’<<z.real;
Cout<<’\n’<<z.imag;
}
```

4.2 Структуралар ва массивлар.

Массивлар структуралар элементлари сифатида.

Массивларни структуралар элементи сифатида ишлатилиши хеч кандай кийинчилик тугдирмайди. Биз юкорида символли массивлардан фойдаланишни курдик.

Куйидаги мисолда фазода берилган нуктавий жисмни тасвирловчи компоненталари жисм массаси ва координаталаридан иборат структура киритилган булиб, нуктанинг координаталар марказигача булган масофаси хисобланган.

```
 Include <iostream.h>
#include <math.h>
void main()
{  struct
 {  double mass;
float coord[3]
} point={12.3,{1.0,2.0,-3.0}};
int i;
float s=0.0;
for (i=0;i<3; i++)
s+=point.coord[i]*point.coord[i];
Cout<<''\n масофа="<<sqrt(s);
}
```

Бу мисолда point структураси номсиз структурали тип оркали аникланган булиб, кийматлари инициализация ердамида аникланади.

Структуралар массивлари.

Структуралар массивлари оддий масивлар каби тасвирланади. Юкорида киритилган структурали типлар асосида куйидаги структуралар массивларини киритиш мумкин:

```
Struct goods list[100];
Complex set [80];
```

Бу таърифларда list ва set структуралар номлари эмас, элеменлари структуралардан иборат массивлар номларидир. Конкрет структуралар номлари булиб set[0],set[1] ва хоказолар хизмат килади. Конкрет структуралар элементларига куйидагича мурожаат килинади: set[0].real– set массиви биринчи элементининг real номли компонентасига мурожаат.

Куйидаги мисолда нуктавий жисмларни тасвирловчи структуралар массиви киритилади ва бу нукталар системаси учун огирлик маркази

координаталари (xc,yc,zc) хисобланади. Бу координаталар куйидаги формулалар асосида хисобланади:

m=∑mi; xc =(∑ximi)/m; yc =(∑yimi)/m; zc =(∑zimi)/m;

```
#Include <iostream.h>
struct particle {
double mass;
float coord [3];
};

struct particle mass_point[]={ 20.0, {2.0,4.0,6.0}
                               40.0, {6.0,-2.0,6.0}
                               10.0, {1.0,3.0,2.0}
                                                  };
int N;
struct particle center ={ 0.0, {0.0,0.0,0.0}
                                              }
int I;
N=sizeof(mass_point)/sizeof(mass_point[0]);
For (I=0;I<N;I++)
{
center.mass+=mass_point[I].mass
center.coord[0]+=  mass_point[I].coord[0]* mass_point[I].mass;
center.coord[1]+=  mass_point[I].coord[1]* mass_point[I].mass;
center.coord[2]+=  mass_point[I].coord[2]* mass_point[I].mass;
}
Cout<<“\n Масса маркази координаталари:”;
for (I=0;I<3;I++)
{
center.coord[I]/=center.mass;
Cout<<“\n Координата ”<<(I+1)<<center.coord[I]);
}
}
```

4.3 Структуралар ва курсаткичлар.

Структурага курсаткичлар.

Структурага курсаткичлар оддий курсаткичлар каби тасвирланади:

Complex *cc,*ss; goods *p_goods;

Структурага курсаткич таърифланганда инициализация килиниши мумкин. Мисол учун экрандаги рангли нуктани тасвирловчи куйидаги структурали тип ва структуралар массиви киритилади. Структурага курсаткич кийматлари инициализация ва киймат бкриш оркали аникланади:

```
Struct point
{int color;
  int  x,y;
} a,b;
 point *pa=&a,pb; pb=&b;
```

Курсаткич оркали структура элементларига икки усулда мурожаат килиш мумкин. Биринчи усул адрес буйича киймат олиш амалиг асосланган булиб куйидаги шаклда кулланилади:

(* структурага курсаткич).элемент номи;

Иккинчи усул махсус стрелка (->) амалига асосланган булиб куйтдаги куринишга эга:

структурага курсаткич->элемент номи

Структура элементларига куйидаги мурожаатлар узаро тенгдир:

(*pa).color==a.color==pa->color

Структура элементлари кийматларини курсаткичлар ердамида куйидагича узгартириш мумкин:

```
(*pa).color=red;
pa->x=125;
pa->y=300;
```

Дастурда нуктавий жисмни тасвирловчи particle структурали типга тегишли m_point структураси аникланган булсин. Шу структурага pinta курсаткичини киритамиз:

```
Struct particle * pinta=&m_point;
```

Бу холда m_point структура элементларини куйидагича узгартириш мумкин:

```
Pinta->mass=18.4;
For (I=0;I<3;I++)
Pinta->coord[I]=0.1*I;
```

Структураларга курсаткичлар устида амаллар.

Структураларга курсаткичлар устида амаллар оддий курсаткичлар устида амаллардан фарк килмайди. Агар курсаткичга структуралар массивининг бирор элементи адреси киймат сифатида берилса, массив буйича узлуксиз силжиш мумкин булади.

Мисол тарикасида комплекс сонлар массиви суммасини хисоблаш масаласини куриб чикамиз:

```
#include <iostream.h>
void main()
{
sruct complex
 {float x;
   float y;} array[]={1.0,2.0,3.0,-4.0,-5.0,-6.0,-7.0,-8.0};
```

```
struct complex summa={0.0,0.0};
struct complex *point=&array[0];
int k,I;
k=sizeof(array)/sizeof(array[0]);
for(i=0;i<k;i++)
{
summa.x+=point->x;
summa.y+=point->y;
point++;
}
Cout<<“\n Сумма: real=”<< summa.x <<” imag=”<<summa.y;
}
```

Дастур бажарилиши натижаси:
Сумма: real=-8.000000, imag=-16.000000

4.4 Структуралар ва функциялар.

Структуралар функциялар аргументлари сифатида еки функция кайтарувчи киймат келиши мумкин. Бундан ташкари иккала холда хам структурага курсаткичлардан фойдаланиш мумкиндир.

Мисол учун комплекс сон модулини хисоблаш дастурини келтирамиз:

```
Double modul(complex a)
{return sqrt(a.real*a.real+a.imag*a.imag}
```

Икки комплекс сон йигиндисини хисоблаш функцияси:

```
Complex add(complex a, complex b)
{ complex c;
c.real=a.real+b.real;
c.imag=a.imag+b.imag;
return c;   }
```

Бу функцияни курсаткичлар ердамида куйидагича езиш мумкин

```
Complex* add(complex* a, complex* b)
{ complex* c; c=(complex*)malloc(sizeof(complex));
c->real=(*a).real+(*b).real;
c->imag=(*a).imag+(*b).imag;
return c;   }
```

Бу функция complex типидаги динамик объект яратиб адресини кайтаради. Дастурда бу объект учун ажратилган жойни озод килиш максадга мувофик. Бу функцияга дастурда куйидагича мурожаат килиш мумкин:

```
Complex a={0.1,-0.3},b={0.2,-0.5};
Complex* pa; pa=add(&a,&b);
```

Назорат саволлари

1. Структура деб нимага айтилади?
2. Хар хил тоифадаги ўзгарувчиларни қандай бирлаштириш мумкин?
3. Структура таркибида нималар бўлиши мумкин?
4. Структура қандай таърифланади?
5. Структура элементларига мурожаат қилиш асослари.
6. Структураларни ҳосил қилишда дастурчининг типидан фойдаланиш
7. Структура таркибида массивлар хам ишлатилиши қоидалари.
8. Структура таркибида кўрсаткичларнинг ишлатилиши.
9. Структура таркибида функцияларнинг ишлатилиши.

Адабиётлар

1. Т.Х.Холматов ва бошқалар. “Информатика”, Тошкент, 2003
2. Р.Каримов ва бошқалар. “Дастурлаш”, Тошкент, 2003
3. Ш.Ш.Шохамидов. “Амалий математика элементлари”, Тошкент, 1997
4. Ашарина И.В. “Основы программирования на языках С и С++”, Москва, 2002
5. Павловская Т.А. «С /С++ программирование на языке высокого уровня», С.Петербург, 2001
6. В.В.Подбельский, С.С.Фомин. Программирование ва Си. Москва, 2004

ТЕСТЛАР

1. Кайси суз ёрдамида структура таърифланади?
   A) switch
   B) throw
   C) public
   +D) struct
   E) for

2. Кайд этиш (перечисление) кайси калитли суздан бошланади?
   +A) enum
   B) struct
   C) typedef
   D) union
   E) class

3. Бирлашма (объединение) кайси калитли суздан бошланади?

A) enum
B) struct
C) typedef
+D) union
E) class

4. Структура майдонига мурожат этганда (.) дан чапда кайси операнд жойлашади?

A) структура майдони
B) структура исми
+C) структурали узгарувчи
D) структуранинг калитли сузи
E) структурага курсаткич

4. Фойдаланувчининг типини яратиш кайси функция оркали бажарилади?
A) float
B) double
C) long double
+Д) typedef

5. Структураларда кандай типлар жамланади?
A) факат бутун типлар
+B) ихтиёрий типлар
C) бутун ва хакикий типлар
Д) факат хакикий типлар

5-маъруза
Файллар

Мақсад: C/C++ тилида файллар яратиш ва улардан фойдаланиш кўникмасини ҳосил қилиш
Калит сўзлар: файл, файл оқими, файл типи, маълумот алмашиш, директива яратиш.

Режа

1. C/C++ тилида файллар ҳосил қилиш ва улар устида амаллар бажариш
2. Фойдаланувчининг директива файлини яратиш қоидалари
3.

5.1. C/C++ да файллар билан ишлаш

Олдиндан берилган катталикларни - объектларни киритиш чикариш
C++ тилида киритиш-чикариш окимларининг синфлари мавжуд булиб, улар киритиш-чикариш стандарт кутубхонасининг объектга мулжалланган эквивалентидир. Улар куйидагилар:
istream - киритиш окими
ostream - чикариш окими
iostream - киритиш/чикариш окими

Сатрли окимлар хотирада жойлаштирилган сатрли буферлардан маълумотларни киритиш-чикариш учун хизмат килади.
istrstream - сатрли киритиш
ostrstream - сатрли чикариш
strstream - сатрли киритиш/чикариш

Куйидаги файлли окимлар файллар билан ишлаш учун хизмат килади.
ifstream - файлли киритиш
ofstream - файлли чикариш
fstream - файлли киритиш/чикариш

Одатда бу окимлар include < …..> сифатида ёзилади.
ifstream, ofstream ва fstream окимлари дастурда файллар хосил килиш, улардаги маълумотлардан фойдаланиш учун ишлатилади. Уларнинг кулланилиши куйидагича:
ofstream name (" path\ file_name"); - маълумотли файл хосил килиш, яъни маълумотлар базаси учун очиш;
Масалан: ofstream farruh("c:\ tcpp\bin\d11.dat");
ofstream alibek ("nnn.txt");
Бу ерда

name - ихтиёрий ном (лотинча);яъни оким номи. Кейинчалик файлдаги маълумотларни ёзиш ёки укиш учун шу номдан фойдаланамиз.
d11.dat ва nnn.txt биз хосил килган файл номлари булиб, улар оким номлари билан боглангандир.
Энди хосил булган маълумотлардан фойдаланиш учун уни очишни курамиз:
ifstream name ("path");
Масалан: ifstream farruh ("c:\ tcpp\bin\d11.dat");
ifstream alibek("nnn.txt");

Очилган файлларни албатта ёпиш керак! Бу жараённи куйидагича амалга оширилади: name.close( );
Масалан, farruh.close( ); ёки alibek.close( );

Демак, farruh билан d11.dat, alibek билан nnn.txt номлари узаро маълумот алмашинувини таъминлайди.

Масалан:
2та бутун сонни ва уларнинг йигиндисини узида сакловчи файл хосил килинг ва ундан кейинги дастурда фойдаланинг.
Сонларни a, b, йигиндини s, файлни ttt.dat, оким номини jasur деб атаймиз.

```
# include <iostream.h>
# include <fstream.h>
# include <conio.h>
void main ( )
{
int a=12, b=13, s;
ofstream jasur ("ttt.dat");
s = a + b;
cout <<”s=” << s << endl;
jasur << s <<endl;
jasur.close ( );
getch ( );
}
```

Энди ундан фойдаланамиз:

```
// # include <iostream.h>
# include <fstream.h>
# include <conio.h>
# include <math.h>
void main ( )
{
int   s; float s1;
ifstream jasur ("ttt.dat");
jasur >>s;
s1 = sin (s);
cout <<"s1="<<s1<<endl;
jasur.close ( );
getch( );   }
```

2-мисол. Матрица ва векторлар берилган. Сон кийматлари ихтиёрий. Ушбу кийматлардан фойдаланиб, матрицани векторга купайтириш, матрицанинг изини хисоблаш ва векторнинг йигиндисини хисоблаш дастурини тузинг.

Аввал матрица ва векторларнинг сон кийматларини узида сакловчи файл хосил киламиз.

Сунгра бу маълумотлардан фойдаланамиз.

```
# include <iostream.h>
# include <fstream.h>
# include <stdlib.h>
# include <time.h>
void main ( )
{ srand (time (0));
int  a [3][3], b[3], i, j;
ofstream said ("akbar.txt");
for ( i=0; i<3; i++)
{ for (j=0; j<3; j++)
{ a[i][j] = rand( );
said <<a[i][j]; } }
for (i=0; i<3; i++)
{ b[i] = rand( );
said << b[i];  }
said.close ( );
}
# include <iostream.h>
# include <fstream.h>
void main ( )
{
int a[3][3], b[3], i, j, c[3], s1=0, s2=0;
ifstream said ("akbar.txt");
for ( i=0; i<3; i++)
for (j=0; j<3; j++)
said >>a[i][j];
for ( i=0; i<3; i++)
said >> b[i];
for ( i=0; i<3; i++)
{  c[i] = 0;
for (j=0; j<3; j++)
c[i] = c[i] + a[i][j] * b[i];
cout << "c="<<c[i]<<endl; }
for ( i=0; i<3; i++)
s1 = s1 + a[i][i];
for ( i=0; i<3; i++)
s2=s2 + b[i];
cout << "s1="<<s1<<"   s2="<<s2<<endl;
```

}

5.2. Фойдаланувчининг директиваларини (include) яратиш

C/C++ тилида фойдаланувчи узининг include ини яратиши ва ундан керакли вактларда кенг фойдаланиши имконияти берилган. Бунинг учун керакли операциялар алгоритми функция ва процедура сифатидаги дастури блокнотга ёзилади. Сунгра бу дастурга ихтиёрий ном (лотинча) берилади, файл номидан сунг .h кенгайтмаси берилади. h - header - сарлавха деган маънони билдиради. Бу файлни INCLUDE папкасига саклаш шарт! Фойдаланувчи бу include да аввалдан мавжуд булган include лардан, исталган узгарувчи, узгармасдар, функция ва процедуралардан фойдаланиши мумкин. Дастурчи узининг шахсий include ини ишлатиши учун асосий дастурда (C/C++ мухитида) уларни эълон килиши ва ундаги функция ва процедураларнинг номлари хамда хакикий параметрларининг урнини билиши керак. Шахсий include ларни чакиришда < > ёки " " белгилари ишлатилиши мумкин.
Масалан: arccos x = arctg ( ) формуласи ёрдамида ташкил этувчи include яратамиз ва ундан фойдаланамиз.
Блокнотдаги дастури:

```
# include < math.h >
float   acos (float x)
{ float  y;
y = atan( sqrt ( 1-x*x) / x );
return y;
}
```

Бу файлга ихтиёрий ном берамиз, масалан: farruh . h унгра ундан фойдаланиб дастур тузамиз: (C/C++ мухитида)

```
# include  < iostrream.h >
# include < conio.h >
# include < farruh.h >
void main ( )
{ float  x, y;
cin >> x;
y = acos (x);
cout << "y="<< y  << endl;
getch ( );
}
```

Одатда яратилаётган include ларга барча керакли функциялар гурухлаб жойлаштирилади ва бир йула чакириб ишлатилади. Масалан:

```
# include  < math. h >
float acos ( float  x)
{  ……     }

float   asin ( float  x )
{ …….  }
```

```
float  sh ( float  x )
{  ……..  }
```
ва х.к.

2-мисол. Учбурчак ва бешбурчак чизадиган процедура учун include яратинг ва ундан фойдаланинг.

Блокнотда: (номи chiz . h булсин)
```
# include  < graphics. h >
int  gd=0, gm;  initgraph (&gd, &gm, " ");
void  uchb ( int  x1, int  y1, int  x2, int  y2, int  x3, int  y3)
{ line (x1, y1, x2, y2);
   line  (x1, y1, x3, y3);
   line  (x2, y2, x3, y3 ); }

void  besh ( int a[ ] )
{ drawpoly ( 6, a );}
```

Энди бу include дан фойдаланамиз:
```
# include  < chiz. h >
# include  < conio. h >
{  void  main ( )
setcolor (4);
uchb ( 10, 20, 100, 80, 250, 150);
int a[ ] = { 10, 200, 30, 50, 70, 50, 90, 200, 50, 350, 10, 200);
besh (a);
getch ( );
}
```

Назорат саволлари

1. C/C++ тилида файллар нима учун ишлатилади?
2. C/C++ тилида файллар қандай ҳосил қилинади?
3. Маълумотлар базасини очиш йўлларини кўрсатинг.
4. C/C++ тилида файллардан қандай фойдаланилади?
5. Очилган файл нима учун ёпилиши зарур?
6. C/C++ тилида маълумотлар қандай алмашинилади?
7. Маълумотлар базасидаги массивлардан қандай фойдаланиш мумкин?
8. Фойдаланувчининг директиваси нима учун керак?
9. Фойдалаувчининг директиваси қандай яратилади?
10. Фойдаланувчининг директивасидан қандай файдаланиш мумкин?

Адабиётлар

1. Т.Х.Холматов ва бошқалар. “Информатика”, Тошкент, 2003
2. Р.Каримов ва бошқалар. “Дастурлаш”, Тошкент, 2003
3. Ш.Ш.Шохамидов. “Амалий математика элементлари”, Тошкент, 1997
4. Ашарина И.В. “Основы программирования на языках С и С++”, Москва, 2002
5. Павловская Т.А. «С /С++ программирование на языке высокого уровня», С.Петербург, 2001
6. В.В.Подбельский, С.С.Фомин. Программирование на Си. Москва, 2004

ТЕСТЛАР

1. Файллар билан ишлаш учун дастурга кайси сарлавхали файлни кушиш лозим?

A) stdio.h
B) iostream.h
C) conio.h
+D) fstream.h
E) string.h

2. Файл охирини аниклаш учун кандай функция ишлатилади?

A) bad()
B) fail()
C) flush()
D) tellg()
+E) eof()

3. Файл билан ишлашда хатони аниклаш учун кандай функция ишлатилади?

A) bad()
+B) fail()
C) flush()
D) tellg()
E) eof()

4. Файлни кушиш режимида очиб файл курсаткичини файл охирига жойлаштирувчи очиш режими кийматини курсатинг:
+A) ios::app
B) ios::ate
C) ios::in
D) ios::nocreate
E) ios::noreplace

5. Истисноларни (исключительные ситуации) генерация килиш кайси суз ёрдамида амалга оширилади?
A) switch
+B) throw
C) public
D) struct
E) class

6. Истисноларни (исключительные ситуации) кайта ишлаш блогини эълон килиш кайси суз ёрдамида амалга оширилади?
A) operator
+B) catch
C) friend
D) try
E) throw

7. Истиснолар (исключительные ситуации) хосил килиши мумкин бўлган блокни эълон килиш кайси суз ёрдамида амалга оширилади?
A) operator
B) catch
C) friend
+D) try
E) operator

8.Агар кайд этиш(enum) куйидагича берилган булса, унинг элементи BLUE киймати аниклансин Enum COLOR { WHITE, BLACK=100, RED, BLUE, GREEN=300};
A) 4
B) 3
+C) 102
D) 101
E) 11

6-маъруза
C/C++ тилида объектлар

Мақсад: C/C++ тилида объектлар билан ишлаш кўникмасини хосил қилиш

Калит сўзлар: объект, синф, динамик хотира, инкапсуляция, меросхўрлик, полиморфизм, синф аъзолари, усуллари.

Режа:

1. **C++** тили ва объектларга мулжалланган дастурлаш
2. Синф усулларининг аникланиши.
3. Инструктор ва деструкторлар

6.1. C++ тили ва объектларга мулжалланган дастурлаш

Шу вактгача тузган дастурларимиз берилган маълумотлар устида бирор-бир амалларни бажарувчи процедуралар кетма-кетлигидан иборат эди. Процедура ва функциялар хам узида аникланган кетма-кет командалар тупламидан иборатдир.

Объектга йуналтирилган дастурлашнинг асосий максади берилганлар ва улар устида амал бажарувчи процедураларни ягона объект деб карашдан иборатдир. Масалан: нон, автомобиль, одам … объектлари.

C++ тили Объектга йуналтирилган дастурлаш принципларини куллаб кувватлайди. Бу принциплар куйидагилардан иборат:

1. Инкапсуляция
2. Меросхурлик
3. Полиморфизм

Инкапсуляция. Агар инженер ишлаб чикаришда диод, триод еки резисторни ишлатса, у бу элементларни янгитдан ихтиро килмайди, балки дукондан сотиб олади. Демак, инженер уларнинг кандай тузилганлигига эътиборини каратмайди, бу элементлар яхши ишласа етарли. Айнан шу ташки конструкцияда ишлайдиган яширинлик ёки автономлик хоссаси инкапсуляция дейилади.

Меросхурлик. Янги объект яратилаётган булса, иккита вариантдан бири танланади: мутлако янгисини яратиш ёки мавжуд моделнинг конструкциясини такомиллаштиришдир. Купинча 2-вариант танланади, демак, баъзи хусусиятлари узгартирилади холос. Бу нарса меросхурлик принципига асос солади. Янги синф олдин мавжуд булган синфни кенгайтиришдан хосил булади. Бунда янги синф олдинги синфнинг меросхури деб аталади.

Полиморфизм. Поли – куп, морфе – шакл деган маънони билдиради. C++ тили бир хил номдаги функция турли объектлар томонидан ишлатилганда турли амалларни бажариш имкониятини таъминлайди. Полиморфизм – шаклнинг куп хиллигидир.

Дастурда ишлатиладиган хар бир узгарувчи уз типига эга ва у куйидагиларни аниклайди:

1. хотирадаги улчовини;
2. унда сакланаётган маълумотларни;
3. унинг ёрдамида бажарилиши мумкин булган операцияларни.

С++ тилида дастурчи узига керакли ихтиёрий типни хосил килиши мумкин. Бу янги тип ички типларнинг хоссалари ва уларнинг функционал имкониятларини узида ифодалайди. Янги тип синфни эълон килиш оркали тузилади. Синф бу – бир-бири билан функционал богланган узгарувчилар ва усуллар (функциялар) тупламидир.

Масалан: Мушук номли синф тузмокчимиз. Бу ерда унинг ёши, огирлиги каби узгарувчилар ва миёвлаш, сичкон тутиш каби функциялардан иборат. Ёки Машина синфи гилдирак, эшик, уриндик, ойна каби узгарувчилар ва хайдаш, тухтатиш каби функциялардан иборат.

Синфдаги узгарувчилар – синф аъзолари ёки синф хоссалари дейилади.

Синфдаги функциялар одатда узгарувчилар устида бирор бир амал бажаради. Уларни синф усуллари (методлари) деб хам аталади.

Синфни эълон килиш учун **class** сузи , { } ичида эса щу синфнинг аъзолари ва усуллари келтирилади. Масалан:

```
class  non
{  int  ogirlik ;
   int  baho ;
   void  yasash ( );
   void  yopish ( );
   void  eyish ( );
}
```

Синфни эълон килишда хотира ажратилмайди. Синф эълон килинганда компилятор факат шундай (non) синф борлигини, хамда унда кандай кийматлар (ogirlik, baho) сакланиши мумкинлигини, улар ёрдамида кандай амалларни (yasash, yopish, eyish) бажариш мумкинлиги хакида хабар беради. Бу синф объекти хаммаси булиб 4 байт жой эгаллайди (2та int).

Объект синфнинг бирор бир нусхаси хисобланади.

С++ тилида типларга киймат узлаштирилмайди, балки узгарувчига узлаштирилади. Шунинг учун тугридан тугри int = 55 деб ёзиб булмаганидек non.baho=1200 деб хам булмайди. Узлаштиришда хатоликка йул куймаслик учун олдин non синфига тегишли patir объектини хосил киламиз кейин эса унга керакли кийматларни берамиз.

Масалан:

```
int  a;          // бутун типли а узгарувчиси, объекти
non  patir;  //
```

Энди non синфининг реал объекти аникланганидан сунг унинг аъзоларига мурожаат килиш мумкин. Буни нукта оператори (.) оркали бажарилади.

```
patir.baho = 1200;
```

```
patir.ogirlik = 500;
patir.yasash ( ) ;
```

Синфни эълон килишда куйидагилардан фойдаланилади:
public - очик
private – ёпик

Синфнинг барча усул ва аъзолари бошлангич холда автоматик равишда ёпик булади.ёпик аъзоларга эса факат шу синфнинг усуллари оркалигини мурожаат килиш мумкин. Объектнинг очик аъзоларига эса дастурдаги барча функциялар мурожаат килиши мумкин. Лекин синф аъзоларига мурожаат килиш анча мушкул иш хисобланади. Агар тугридан тугри:

```
non  patir;
patir.baho = 1200;
patir.ogirlik = 500;
```

деб ёзсак хато булади.

Аъзоларга мурожаат килишдан олдин уни очик деб эълон килиш керак:

```
# include < iostream.h >
class  non
{  public :
   int  baho;
   int  ogirlik;
   void yasash ( );    };
void  main ( )
{ non  patir;
patir.baho = 1200;   patir.ogirlik = 500;
cout <<"men olgan patir "<<patir.baho <<" so'm"<<endl;
cout <<" uning og'irligi ="<<patir.ogirlik <<endl ;     }
```

6.2. Синф усулларининг аникланиши.

Синф усулини аниклаш учун синф номидан сунг иккита икки нукта (::) белгиси, функция номи ва унинг параметрлари курсатилади.

Масалан: non синфининг patir объектидаги усулларни аниклаш дастури:

```
# include < iostream.h >
class  non
{  public :
   int  baho;
   int  ogirlik;
   void yasash ( );   };
void  non :: yasash ( );
{  cout << "hamir qoriladi, zuvala tutiladi, doira holiga keltiriladi"<< endl;  }
void  main ( )
{ non  patir;
patir.baho = 1200;   patir.ogirlik = 500;
cout <<"men olgan patir "<<patir.baho <<" so'm"<<endl;
cout <<" uning og'irligi ="<<patir.ogirlik <<endl ;
```

```
cout <<” u quyidagich yasaladi:”;
patir . yasash ( );   }
```

6.3. Конструктор ва деструктор

Конструкторлар. Конструкторлар бу синф компонента функциялари булиб, объектларни автоматик инициализация қилиш учун ишлатилади.

Конструкторлар кўриниши қуйидагича бўлиши мумкин:

Синф номи (формал параметрлар руйхати)

{конструктор танаси}

Бу компонента функция номи синф номи билан бир хил бўлиши лозим.

Мисол учун complex синфи учун конструкторни қуйидагича киритиш мумкин:

```
complex (double re = 0.0; double im = 0.0 )
  {real=re; imag=im;}
```

Товарлар синфи учун конструкторни қуйидагича киритиш мумкин.

```
Goods(char* new _ name, float new _ price)
{name= new _ name; price= new _ price; }
```

Конструкторлар учун кайтарилувчи типлар, хатто void типи хам кўрсатилмайди. Дастурчи томонидан кўрсатилмаган холда хам объект яратилганда конструктор автоматик равишда чакирилади.

Масалан cc объект Copmlex cc; шаклида аникланган бўлса, конструктор автоматик чакирилиб real ва imag параметрлари автоматик равишда 0.0 қийматларига эга бўлади.

Кўрсатилмаган холда параметрсиз конструктор ва қуйидаги типдаги нусха олиш конструкторлари яратилади: T :: T (const T&)

Мисол учун

```
Class F
{…..
        public : F(const T&)
        …..
        }
```

Синфда бир нечта конструкторлар бўлиши мумкин, лекин уларнинг фақат биттасида параметрлар қийматлари олдиндан кўрсатилган бўлиши керак.

Конструктор адресини хисоблаш мумкин эмас. Конструктор параметри сифатида ўз синфининг номини ишлатиш мумкин эмас, лекин бу номга кўрсаткичдан фойдаланиш мумкин.

Конструкторни оддий компонента функция сифатида чақириб бўлмайди. Конструкторни икки хил шаклда чақириш мумкин :

Синф_номи ,Объект_номи (конструктор_хақиқий_параметлари)

Синф_номи (конструктор_хақиқий_параметлари)

Биринчи шакл ишлатилганда хақиқий параметрлар руйхати буш булмаслиги лозим. Бу шаклдан янги объект таърифланганда фойдаланилади:

Complex SS(10.3; 0.22)

// real=10.3; SS.imag= 0.22;

Complex EE (2.3)

// EE . real= 2.3;

EE.imag= 0.0;

Complex D( ) // хато

Конструкторни иккинчи шаклда чақириш номсиз объект яратилишига олиб келади. Бу номсиз объектдан ифодаларда фойдаланиш мумкин.

Мисол учун :

Complex ZZ= complex (4.0;5.0);

Бу таъриф орқали ZZ объект яратилиб , унга номсиз объект қийматлари(real= 4.0; imag= 5.0) берилади;

Конструктор номи синф номи билан бир хил бўлиши лозимдир. Мисол учун сиз employee синфдан фойдалансангиз, конструктор хам employee номга эга бўлади. Агар дастурда конструктор таърифи берилган бўлса объект яратилганда автоматик чакирилади. да вы создаете объект. Қуйидаги CONSTRUC.CPP номли дастурда employee номли синф киритилгандир:

```
class employee
{
public:
  employee(char *, long, float); //Конструктор
  void show_employee(void);
  int change_salary(float);
  long get_id(void);
private:
  char name [64];
  long employee_id;
  float salary;
};
```

Конструктор таърифи:

```
employee::employee(char *name, long employee_id, float salary)
{
  strcpy(employee::name, name) ;
  employee::employee_id = employee_id;
  if (salary < 50000.0)
  employee::salary = salary;
  else // Недопустимый оклад
  employee::salary = 0.0;
}
```

Конструктордан фойдаланилганда объект таърифиланганда параметр узатиш мумкин: employee worker("Happy Jamsa", 101, 10101.0);

Агар дастурда employee типидаги объектлар мавжуд бўлса хар бирини қуйидагича инициализация килигш мумкин

employee worker("Happy Jamsa", 101, 10101.0);

employee secretary("John Doe", 57, 20000.0);

employee manager("Jane Doe", 1022, 30000.0);

)

Конструкторлар ва кўзда тутилган қийматлар. Конструкторларда кўзда тутилган қийматлардан хам фойдаланиш мумкиндир. Мисол учун қуйидаги конструктор employee оклада қийматини дастурда кўрсатилмаган бўлса 10000.0 тенг килиб олади.:

```
employee::employee(char *name, long employee_id, float salary = 10000.00)
    {
  strcpy(employee::name, name);
  employee::employee_id = employee_id;
  if (salary < 50000.0)
    employee::salary = salary;
  else // Недопустимый оклад
    employee::salary = 0.0;
}
```

Деструкторлар. Синфнинг бирор объекти учун ажратилган хотира объект йукотилгандан сўнг бушатилиши лозимдир.

Синфларнинг махсус компоненталари деструкторлар , бу вазифани автоматик бажариш имконини яратади.

Деструкторни стандарт шакли қуйидагича :

~ синф_номи ( ) {деструктор танаси}

Деструктор параметри ёки кайтарилувчи қийматга эга бўлиши мумкин эмас. (хатто void типидаги)

Дастур объектни учирганда деструктор автоматик чакирилади.

```
# include < string.h >
# include < iostream.h >
class stroka
{ char *ch;
int  len;
public:
stroka (int  N=80) : len(0)
{ ch= new char [N+1];
ch[0]='\0'
}
```

```
stroka (const char  *ar)
{ len= strlen (ar); ch=new  char[len+1];
strcpy (ch, arch);
}
int len_str (void)
{return len}
char * string (void)
{return ch;}
```

Назорат саволлари

1. Объектга мўлжалланган дастурлашнинг асосий мақсади.
2. Объектга мўлжалланган дастурлашнинг асосий принциплари
3. Инкапсуляция хоссасини айтиб беринг
4. Меросхўрлик хоссасини айтинг.
5. Полиморфизм тушунчаси нимани билдиради?
6. Синф деб нимага айтилади?
7. Синфнинг ўзгарувчилари - аъзолари.
8. Синф усуллари ва уларни ташкил этиш.
9. Объектга қийматларнинг берилиши.
10. Синфни эълон қилиш усуллари.

Адабиётлар

1. Т.Х.Холматов ва бошқалар. “Информатика”, Тошкент, 2003
2. Р.Каримов ва бошқалар. “Дастурлаш”, Тошкент, 2003
3. Ш.Ш.Шохамидов. “Амалий математика элементлари”, Тошкент, 1997
4. Ашарина И.В. “Основы программирования на языках С и С++”, Москва, 2002
5. Павловская Т.А. «С /С++ программирование на языке высокого уровня», С.Петербург, 2001
6. В.В.Подбельский, С.С.Фомин. Программирование на Си. Москва, 2004

ТЕСТЛАР

1. Ворислик бу:

A) бир объектга бошка объект нусхасини кушиш

B) бир объектга бошка объектга илова кушиш

+C) бир синфга бошка синф функционаллигини кушиш

D) синф усулларини кайта таърифлаш

E) ягона объектда маълумотлар ва функцияларни жамлаш

2. Инкапсуляция бу:

+A) ягона объектда маълумотлар ва функцияларни жамлаш

B) бир объектга бошка объект нусхасини кушиш

C) ягона объектда маълумотлар ва шу маълумотларга курсаткичларни жамлаш

D) ягона объектда маълумотлар ва шу маълумотларга иловаларни жамлаш

E) бир синф усулларини бошкасида кайта таърифлаш

3. Хамма объектлар учун умумий булган синф аъзолари кайси суз ёрдамида таърифланади?

+A) static

B) protected

C) private

D) friend

E) public

4. Синфдан ташкарида таърифланган функцияга синф ёпик элементларига мурожаат хукуки кайси суз ёрдамида берилади?

A) static

B) protected

C) private

+D) friend

E) try

5. Кайси суз ёрдамида факат синф ичида ёки унинг авлодларида синифнинг аъзоларидан эркин фойдаланиш хукукини бериш мумкин?

A) static

+B) protected

C) private

D) friend

E) public

6. Кайси суз ёрдамида факат синф ичида синфнинг аъзоларидан эркин фойдаланиш хукукини бериш мумкин?

A) static

B) protected

+C) private
D) friend
E) public

7. Синф аъзосига синф ичида ва ташкарисида мурожаат хукукини бериш кайси суз ёрдамида амалга оширилади?

A) switch
B) throw
+C) public
D) struct
E) protected

8. Синф компонентасига синф номи оркали мурожаат килиш мумкин бўлиши учун у кандай таърифланиши лозим?

A) static ва protected
+B) static ва public
C) static ва private
D) friend ва public
E) static ва friend

9. Объектни инициализация килиш учун ишлатиладиган усулни курсатинг:

+A) конструктор

B) деструктор

C) статик

D) бош

E) инициализатор

10. Кайси жавобда конструктор хоссаси тугри курсатилган?

A) конструктор хеч кандай типдаги киймат кайтармайди

B) конструкторга курсаткич таърифлаш мумкин эмас

C) конструктор адресини олиш мумкин эмас

D) конструкторлар ворисликка утмайди

+E) хамма жавоблар тугри