УДК 004.42(076)

БОЛАШАҚ IT-МАМАНДАРЫН ДАЙЫНДАУДА С++ БАҒДАРЛАМАЛАУ ТІЛІНДЕ МАССИВТЕР МЕН ФУНКЦИЯЛАРДЫ ҮЙРЕНУДІҢ ӘДІСТЕМЕЛІК ЖОЛДАРЫ

Г.Г.Бегаришева

В статье описывается передача одномерных и многомерных массивов, строк в функцию как параметров. Приведены примеры, которые закрепляют теоретический материал для проведения операции с массивами и строками на языке программирования С++ .

In article transfer of one-dimensional and multidimensional files, lines to function as parametres is described. Examples which fix a theoretical material for carrying out of operation with files and lines in the programming language With ++ are resulted.

Функциялар және массивтер

Болашақ IT-мамандарын дайындауда С++ бағдарламалау тілінде көптеген әдебиеттер бар, бірақ қазақ тілінде олар санаулы және кейбір тақырыптар қосымша ақпарат талап етеді, мысалы «С/С++ программалау тіліне кіріспе» оқу-әдістемелік құралында [1].

Си/Си++ тілінде, негізгі типтен басқа негізгі типтердің негізінде алынған туынды типтерді енгізуге және қолдануға рұхсат берілген. Тіл стандарты туынды типтерді алудың үш тәсілін анықтайды:

- берілген типтің массив элементтері;
- берілген типтің объектті көрсеткіші;
- берілген типтің мәнін қайтаратын, функция.

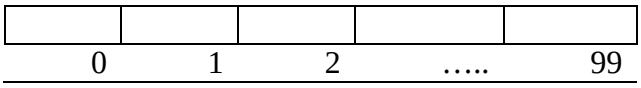
Массив – бұл бір типтің айнымалыларының реттелген тізбегі. Массивтің әр элементіне жадының бір ұяшығы беріледі. Бір массивтің элементтері жадының тізбектей орналасқан ұяшықтарын алады. Барлық элементтердің бір ат беріледі – массив аты және индекстермен айрықшаланады – массивтің реттік номерлері. Массивтегі элементтер саны оның өлшемі деп аталады. Жадыда массивті орналастыруға қажет ұяшықтар санын бөлу үшін алдын ала оның өлшемін білу керек. Жадыны массив үшін алдын ала бөлу бағдарламаның компиляция сатысында орындалады.

Массивті анықтау

int a[100]; // бүтін типті 100 элементтен тұратын массив

sizeof(a) операциясының нәтижесі 400 болады, яғыни әр элементі 4 байтты алатын 100 элемент.

Массив элементтері әрқашанда 0 –ден бастап номерленеді.



Массив элементімен жұмыс жасау үшін массив атын және массивтегі элемент номерін(индекс) көрсету қажет:

a[0] – индекс тұрақты сияқты беріледі,

a[55] – индекс тұрақты сияқты беріледі,

a[I] – индекс айнымалы сияқты беріледі,

a[2*I] – индекс формула сияқты беріледі.

Массив элементтерін оны анықтаған кезде беруге болады:

int a[10]={1,2,3,4,5,6,7,8,9,10} ;

sizeof(a) операциясының нәтижесі 40 болады, яғыни әр элементі 4 байтты алатын 10 элемент.

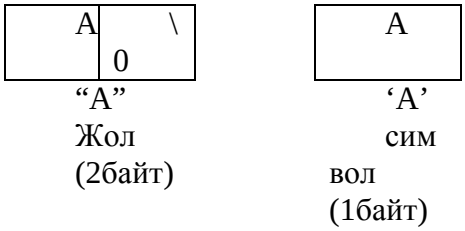
int a[10]={1,2,3,4,5};

sizeof(a) операциясының нәтижесі 40 болады, яғыни әр элементі 4 байтты алатын 10 элемент. Егер бастапқы мәндер саны берілген массив ұзындығынан аз болса, онда массивтің бастапқы элементтері тек қана бірінші элементтерді алады.

int a[]={1,2,3,4,5};

sizeof(a) операциясының нәтижесі 20 болады, яғыни әр элементі 4 байтты алатын 5 элемент. Массив ұзындығы инициализация кезінде саналған мәндер саны бойынша компилятормен есептеледі.

Си++ - де **жол** - бұл символдар массиві, нөл-символға – ‘\0’ (нөл-терминаторға) аяқталады. Нөл-терминатор жағдайы фактикалық жолдар ұзындығын анықтайды. Мұндай массивте элементтер саны жолдардың бейнеленуіне қарағанда 1-ге артық.



Сур.2. Жолдар мен символдарды көрсету

Меншіктеу операторының көмегімен жолға мән меншіктеуге болмайды. Массивке жолды енгізудің немесе инициализациялаудың көмегімен орналастыруға болады.

Мысал

```
void main()
{
        char s1[10]="string1";
        int k=sizeof(s1);
        cout<<s1<<"\t"<<k<<endl;
        char s2[]="string2";
        k=sizeof(s2);
        cout<<s2<<"\t"<<k<<endl;
        char s3[]={‘s’,’t’,’r’,’i’,’n’,’g’,’3’}
        k=sizeof(s3);
        cout<<s3<<"\t"<<k<<endl;
        char *s4="string4";//жол көрсеткіші, оны өзгертуге болмайды
        k=sizeof(s4);
        cout<<s4<<"\t"<<k<<endl;
}
```

Нәтижелер:

string1 10 – 10 байт бөлінген

string2 8 – 8 байт (7+1) бөлінген

string3 8 – 8 байт (7+1) бөлінген
string4 4 – көрсеткіш өлшемі

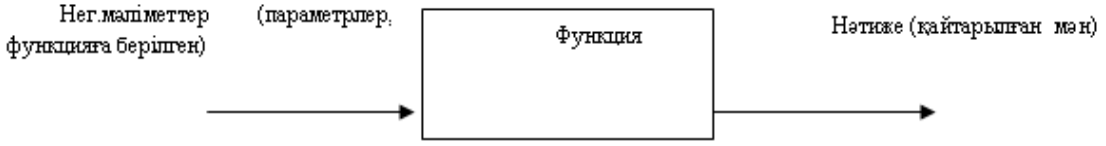
Функциялар

Бағдарлама көлемінің үлкеюіне байланысты барлық элементтерді жадыда ұстау мүмкін болмайды. Бағдарламаның күрделілігін азайту үшін, оны бөлшектерге бөледі. Си++ -те тапсырмалар функциялар көмегімен аса қарапайым тапсырмаларға бөлінуі мүмкін. Тапсырманың функцияларға бөлінуі сондай-ақ код салмақтығын азайтуға мүмкіндік береді, себебі функцияны бір рет жазып, бірнеше рет шақырамыз. Функциядан құралған бағдарламаны бақылаған жеңіл. Жиі қолданылатын функцияларды кітапханаларға орналастыруға болады.

Функцияларды анықтау және хабарлау

Функция – бұл аяқталған әрекетті орындайтын атаулы әрекеттер мен операторлардың тізбегі, мысалы массивтердің құрылуы, массивтерді шығару т.б..

Функция, біріншіден, СИ++-дің кез-келген типтерінің бірі болып саналады, ал, екіншіден, бағдарламаның ең кіші орындалушы модулі.



Кез келген функция хабарландырылған және анықталған болуы керек. Функцияның хабарлануы(прототип, тақырыпша) функцияның атын береді, қайтарылған мән типі және берілетін параметрлер тізімі. Функцияның анықталуы хабарландырудан басқа, әрекеттер және операторлардың тізбегінен тұратын функция денесін қамтиды.

тип аты_функция([формальдыпараметрлер тізімі])
{ функция _ денесі }

Функция_денесі – бұл блок немесе құрама оператор. Функцияның ішінде басқа функцияны анықтауға болмайды. Функция денесінде функцияның алынған мәнін шақырылған нүктеге қайтаратын оператор болуы керек. Ол екі формада болуы мүмкін:

1. Return формула;
2. return;

Бірінші форма нәтижені қайтару үшін қолданылады, сондықтан формула типі анықтамадағы функция типімен сәйкес келуі керек. Екінші форма функция мәнді қайтармаса қолданылады, яғыни void типі бар. Бағдарламашы бұл операторды функция денесінде анық түрде қолданбауы мүмкін, компилятор оны автоматты түрде функция соңына қосады.

Қайтарылған мән массив және функциядан басқа кез келген тип болуы мүмкін, бірақ массив немесе функцияға көрсеткіш болуы мүмкін.

Формальды параметрлер тізімі – бұл функцияға беруді қажет ететін шамалар. Тізім элементтері үтірлермен бөлінеді. Әр параметр үшін типі және аты көрсетіледі. Хабарлауда атын көрсетпеуге болады.

Функция денесінде жазылған операторлар орындалуы үшін функция шақырылуы қажет. Шақырылғанда функция аты және нақты параметрлер көрсетіледі. Нақты параметрлер формальды параметрлерге функция денесіне операторларының орындалу барысында ауысады. Нақты және формальды параметрлер саны және типі бойынша сәйкес келуі керек.

Мәтінде функция шақырылуынан бұрын хабарлануы керек, себебі компилятор шақырудың дұрыстығын тексеруді жүзеге асыру үшін. Егер функция void емес типте болса, онда оның шақырылуы өрнек операндасы болуы мүмкін.

Мысал:

Үшбұрыш жақтарының координаталары берілген. Егер мұндай үшбұрыш бар болса, онда оның ауданын табу.

1. Математикалық модель:

l=sqrt(pow(x1-x2,2)+pow(y1-y2,2)); //үшбұрыш жағы ұзындығы
p=(a+b+c)/2;
s=sqrt(p*(p-a)*(p-b)*(p-c)); // Герон формуласы

үшбұрыштың бар болуын тексеру

(a+b>c&&a+c>b&&c+b>a)

2. Алгоритм:

Үшбұрыш жақтарының координаталарын енгізу (x1,y1),(x2,y2),(x3,y3);

Жақтардың ұзындығын есептеу ab, bc, ca;

Мұндай жақтармен үшбұрыш бар ма тексеру. Егер бар болса, онда ауданын есептеп және нәтижесін шығар.

Егер жоқ, онда хабарламаны шығару.

Егер барлық координаталар 0-ге тең болса, онда соңы, әйтпесе п.1 –ге қайту.

```
#include <iostream.h>
#include <math.h>
double line(double x1,double y1,double x2,double y2)
{
//функция координаталары x1,y1 және x2,y2 берілген кесіндінің ұзындығын қайтарады,
return sqrt(pow(x1-x2,2)+pow(y1-y2,2));
}
double square(double a, double b, double c)
{
//функция қабырғаларының ұзындығы a,b,c арқылы берілген үшбұрыш ауданын қайтарады,
double s, p=(a+b+c)/2;
return s=sqrt(p*(p-a)*(p-b)*(p-c));// Герон формуласы
}
bool  triangle(double a, double b, double c)
{
// true қайтады, егер үшбұрыш бар болса
if(a+b>c&&a+c>b&&c+b>a) return true;
else return false;
}
```

```
void main()
{
double x1=1,y1,x2,y2,x3,y3;
double point1_2,point1_3,point2_3;
do
{
cout<<"\nEnter koordinats of triangle:";
cin>>x1>>y1>>x2>>y2>>x3>>y3;
point1_2=line(x1,y1,x2,y2);
point1_3=line(x1,y1,x3,y3);
point2_3=line(x2,y2,x3,y3);
if(triangle(point1_2,point1_3,point2_3)==true)
cout<<"S="<<square(point1_2,point2_3,point1_3)<<"\n";
else cout<<"\nTriagle doesnt exist";
}
while(!(x1==0&&y1==0&&x2==0&&y2==0&&x3==0&&y3==0));
}
```

Функция прототипі

Функцияға хабарласуды жүзеге асыру үшін сол файлда функция анықтамасы немесе сипаттамасы(прототипі) орналасуы керек.

```
double line(double x1,double y1,double x2,double y2);
double square(double a, double b, double c);
bool  triangle(double a, double b, double c);
double line(double ,double ,double ,double);
double square(double , double , double );
bool  triangle(double , double , double );
```

Прототип бар болған жағдайда шақырылған функция шақырушы функциямен бір файлда орналасуы міндетті емес, жеке модульдер түрінде сипатталуы мүмкін және объектті модулдер кітапханасында компиляцияланған түрде сақталады.Бұл стандартты модульдердегі функцияларға да қатысты. Бұл жағдайда кітапханалар функциялардың анықталуы трансляцияланғанжәне объектті модулдер түрінде сипатталынған, компилятор кітапханасында орналасқан, ал функция сипаттамасын бағдарламаға қосымша қосу қажет. Бұл қайтапроцессорлық бұйрығы include< файл аты> көмегімен жүзеге асады.

Файл_аты – функцияның берілген компиляторы үшін стандартты топтар протатипін қамтитын тақырыптық файлды анықтайды. Мысалы,барлық бағдарламаларда ағындық енгізу - шығару обьектілерін бейнелеу үшін біз #include <iostream.h> командасын және оларға сәйкес әрекеттерді қолдандық.

Көп мөлшерлі және әртүрлі модульдерде орналасқан функциялардан тұратын өз бағдарламаларымызды құрастыру кезінде, функция протатипі және ішкі обьектілер бейнесі (тұрақтылар, айнымалылар, массивтер) бөлек файлға орналасады.

Функция параметрлері

Параметрлер механизмі шақырылған және шақырған функциялардың арасындағы ақпарат алмасудың негізгі әдісі болып табылады. Функцияға параметрді берудің екі тәсілі бар: адрес бойынша және мәні бойынша [2].

Мән бойынша беруде келесі әрекеттер орындалады:

- нақты параметрлер орнында тұрған өрнектің мәні есептеледі;
- функцияның формальді параметрлеріне стекте жады белгіленеді;
- әрбір нақты параметрге формальды параметр мәні меншіктеледі, сонымен қатар типтер сәйкестігі тексеріледі және қажеттілігіне қарай олардың түрлендіруі орындалады.

Мысал:

```
double square(double a, double b, double c)
{
//функция қабырғаларының ұзындығы a,b,c арқылы берілген үшбұрыш ауданын қайтарады, double s, p=(a+b+c)/2;
return s=sqrt(p*(p-a)*(p-b)*(p-c)); // Герон формуласы
}
1) double s1=square(2.5,2,1);
2) double a=2.5,b=2,c=1;
double s2=square(a,b,c);
3) double x1=1,y1=1,x2=3,y2=2,x3=3,y3=1;
double s3=square(sqrt(pow(x1-x2,2)+pow(y1-y2,2)), // 1және2 арасындағы арақашықтық
 sqrt(pow(x1-x3,2)+pow(y1-y3,2)), // 1 және 3 арасындағы арақашықтық
sqrt(pow(x3-x2,2)+pow(y3-y2,2))); // 2 және3 арасындағы арақашықтық
```

Стек

A	.5
B	
C	
S	
P	

P және S – жергілікті айнымалылар.

Сонымен стекке нақты параметрлер көшірмесі кіргізіледі және функция операторлары осы көшірмелермен жұмыс жасайды. Функцияның нақты параметрлердің өзімен жұмыс жасайтын мүмкіндігі жоқ, демек, оларды өзгертуге болмайды.

Адрес бойынша беруде стекке параметрлердің адрестік көшірмесі енгізіледі, яғни функцияның нақты параметрлер орналасқан жады ұяшығына рұқсаты болады және ол оны өзгерте алады.

Мысал.

```
void Change(int a,int b)//мән бойынша беру
{int r=a;a=b;b=r;}
int x=1,y=5;
Change(x,y);
```

	1	5
	5	1
		1

```
cout<<"x="<<x<<"y="<<y;
```

шығады: x=1y=5

```
void Change(int *a,int *b)//адрес бойынша беру
```

{int r=*a;*a=*b;*b=r;}
int x=1,y=5;
Change(&x,&y);

	&x	5
	&y	1
		1

cout<<”x=”<<x<<”y=”<<y;
шығады: x=5y=1

Адрес бойынша беру үшін сондай-ақ сілтемелер де қолданылуы мүмкін. Сілтеме бойынша беруде функцияға параметрді шақыру кезіндегі көрсетілген параметрдің адресі беріледі ,ал функцияның ішінде параметрге хабарласудың барлығы айқын емес түрде ат өзгертіледі .

void Change(int &a,int &b)
{int r=a;a=b;b=r;}
int x=1,y=5;
Change(x,y);

	&x	5
	&y	1
		1

cout<<”x=”<<x<<”y=”<<y;
шығады: x=5y=1

Көрсеткіштердің орнына сілтемелерді қолдану бағдарлама оқылуын жақсартады, сондықтан ат өзгерту операциясын қолданудың қажеті жоқ. Мән бойынша берудің орнына сілтемелерді қолдану да тиімті болып табылады, себебі параметрлерді көшіру міндетті емес. Егер функция ішінде параметрді өзгертуге тиым салу міндеттелсе, онда const модификаторы қолданылады. Const-ты барлық параметрлердің алдына қою ұсынылады (өзгертулер функцияда қарастырылмаған).

Бірөлшемді массивтерді функцияға беру

Массивті функция параметрі сияқты қолдануда, функцияға оның бірінші элементіне көрсеткіш беріледі, яғни массив әрқашан адрес бойынша беріледі. Мұндайда массивтегі элементтер саны туралы ақпарат жоғалады, сондықтан массив өлшемін жек параметр ретінде беру ұсынылады. Функцияға массивтің басына көрсеткіш берілгендіктен (адрес бойынша беру), массив функция денесінің операторлары есебінен өзгеруі мүмкін[2].

Мысал 1:
Массивтен барлық жұп элементтерді жою

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

int form(int a[100])
{
int n;
        cout<<"\nEnter n";
        cin>>n;

        for(int i=0;i<n;i++)
```

```
                a[i]=rand()%100;
        return n;
}
void print(int a[100],int n)
{
        for(int i=0;i<n;i++)
                cout<<a[i]<<" ";
        cout<<"\n";
}

void Dell(int a[100],int&n)
{
        int j=0,i,b[100];
        for(i=0;i<n;i++)
        if(a[i]%2!=0)
        {
        b[j]=a[i];j++;
        }
        n=j;
        for(i=0;i<n;i++)a[i]=b[i];

}

void main()
{
        int a[100];
        int n;
        n=form(a);
        print(a,n);
        Dell(a,n);
        print(a,n);
}
```

Мысал 2

Массивтен бірінші элементпен сәйкес келетін барлық элементтерді жадының динамикалық бөлінуін қолдану арқылы жою.

```
#include <iostream.h>
#include <stdlib.h>
int* form(int&n)
{
        cout<<"\nEnter n";
        cin>>n;
        int*a=new int[n];   // динамикалық жады облысына көрсеткіш

        for(int i=0;i<n;i++)
                a[i]=rand()%100;
        return a;
}
void print(int*a,int n)
```

```
{
        for(int i=0;i<n;i++)
                cout<<a[i]<<" ";
        cout<<"\n";
}

int*Dell(int *a,int&n)
{
        int k,j,i;
        for(k=0,i=0;i<n;i++)
        if(a[i]!=a[0])k++;
        int*b;
        b=new int [k];
        for(j=0,i=0;i<n;i++)
        if(a[i]!=a[0])
        {
        b[j]=a[i];j++;
        }
        n=k;
        return b;
}

void main()
{
        int *a;
        int n;
        a=form(n);
        print(a,n);
        a=Dell(a,n);
        print(a,n);
}
```

Жолды функцияға беру

Жолдарды функцияға беруде char* типті көрсеткіштер ретінде немесе char типті бірөлшемді массив сияқты берілуі мүмкін. Кәдімгі массивтерден айырмашылығы функцияда жолдар ұзындығы көрсетілмейді, себебі жолдар соңында жолдар соңы /0 белгісі бар.

Мысал: Жолдан берілген символды іздеу функциясы

```
int  find(char *s,char c)
{
for (int I=0;I<strlen(s);I++)
if(s[I]==c) return I;
return –1
}
```

Бұл функцияның көмегімен жолдағы дауысты әріптер санын санаймыз.

```
void main()
{
char s[255];
```

```
gets(s)
char gl=”aouiey”;
for(int I=0,k=0;I<strlen(gl);I++)
if(find(s,gl[I])>0)k++;
printf(“%d”,k);
}
```

Көпөлшемді массивтерді функцияға беру

Көпөлшемді массивтерді функцияға беру кезінде барлық өлшемдер параметрлер ретінде берілуі керек [3]. Егер біз массивті бірнеше индекстермен сипаттасақ, мысалы, int mas[3][4] массиві, онда біз элементтері int[4] бірөлшемді массивіне көррсеткіш болатын бірөлшемді mas массивін сипаттадық.

Мысал: Квадратты матрицаларды тасымалдау

Егер функция тақырыбын анықтасақ: void transp(int a[][],int n){…..} – онда біздің функцияға белгісіз өлшеммен массивті бергіміз келетіні анықталады. Анықтама бойынша массив бірөлшемді және оның элементтері бірдей ұзындықта болуы керек. Массивті беруде элементтер өлшемі туралы ештеңе айтылмаған, сондықтан компилятор қатені береді.

Бұл мәселені шешудің ең қарапайым әдісі функцияны келесі түрде анықтау: void transp(int a[][4],int n), онда әр жолдың ұзындығы 4, ал массив көрсеткіштердің өлшемі есептелетін болады.

```
#include<iostream.h>
const int N=4;// ауқымды айнымалы
void transp(int a[][N],int n)
{
int r;
for(int I=0;I<n;I++)
for(int j=0;j<n;j++)
if(I<j)
{
r[a[I][j];a[I][j]=a[j][I];a[j][I]=r;
}
}
void main()
{
int mas[N][N];
for(int I=0;I<N;I++)
for(int j=0;j<Nlj++)
cin>>mas[I][j];
for(I=0;I<N;I++)
{
for(j=0;j<N;j++)
cout<<mas[I][j]
cout<<”\n”;
}
transp(N,mas);
```

```
for(I=0;I<N;I++)
{
for(j=0;j<N;j++)
cout<<mas[I][j]
cout<<”\n”;
}
}
```

Бұл мақала жоғары оқу орындарында 050704 «Есептеуіш техникасы және бағдарламалық қамтамасыз ету», 050703 «Ақпараттық жүйе» мамандықтарының студенттеріне С++ бағдарламалау тілінде массивтер мен функцияларды үйренуде қажет құрал болады деген ойдамын.

Әдебиеттер:

1. Рашбаев Ж.М. С/С++ программалау тіліне кіріспе: оқу-әдістемелік құрал.- Атырау. Х.Досмұхамедов атындағы Атырау мемлекеттік университеті, 2007. -184 б.
2. А.Я.Архангельский. Программирование в С++ Builder 6.–М.: ЗАО «Издательство БИНОМ», 2004 г.- 1152 с.: ил.
3. С/С++. Структурное программирование: Практикум/ Т.А. Павловская, Ю.А.Щупак.-СПб.:, Питер, 2002.-240с.:ил.