Масиви

Вправи

1. Створити статичний одномірний масив заданої розмірності. Ввести значення елементів масиву з консолі. Вивести зазначення елементів на консоль. Підрахувати середнє арифметичне всіх елементів масиву. Підрахувати окремо середні арифметичні додатних і від'ємних елементів масиву.
2. Створити одномірний динамічний масив, розмірності заданої користувачем. Заповнити його випадковими числами.
    a. Знайти будь-який елемент масиву, значення якого потрапляють у заданий інтервал [a,b].
    b. Знайти всі елементи масиву, значення яких потрапляють у заданий інтервал [a,b].
    c. Знайти найбільший і найменший елемент масиву.
3. Створити статичний двовимірний масив заданої розмірності. Ввести значення елементів масиву з консолі. Вивести зазначення елементів на консоль. Створити вектор середніх арифметичних значень кожного рядка масиву.
4. Розібрати способи створення двовимірних динамічних масивів за кодами, наведеними у додатку.
5. Комплексну матрицю Z зобразимо парою <X,Y> дійсних матриць Z=X+*i*Y. Написати програму обчислення добутку двох комплексних матриць <A,B> і <C,D>. За визначенням добуток має вигляд X+*i*Y = (A+*i*B)*(C+*i*D) = (AC-BD)+*i*(AD+BC). Його можна обчислити іншим способом, знайшовши три матриці
   T = (A+B)*(C-D); R = A*D; S= B*C
   та зауваживши, що
   X = T + R – S; Y = A*D+B*C.
   Який з двох способів економніший?
6. *Магічний квадрат.* Заповніть двомірну матрицю *M* порядку *n* (*n* непарне) натуральними числами 1,2,...,n^2 так, щоб
   $$\sum_{k=1}^{n} M_{ik} = \sum_{k=1}^{n} M_{ki} = C$$
   та
   $$\sum_{k=1}^{n} M_{kk} = \sum_{k=1}^{n} M_{k,n-k+1} = C$$
   де $$C = \frac{n(n^2+1)}{2}$$.
   Вказівка. Почніть з $$i = \frac{n+1}{2}$$, *j*=*n*. Послідовно присвоюйте значення 1,2,...,n^2 протягом *n-1* кроків, збільшуючи *i* та *j* на 1 (за модулем *n*). На кожному *n*–му кроці залишайте *i* без змін, а *j* зменшуйте на 1.

Додаток 1. Створення двовимірного масиву

```
int main()
{
	const int ncol = 4;
	int nrow =5;
	int (*m)[ncol] = new int [nrow][ncol];

	for (int i=0; i<nrow; i++)
		for (int j=0; j<ncol; j++)
			*(*(m+i)+j)=10*i+j;
	for ( i=0; i<nrow; i++)
	{
```

```
            for (int j=0; j<ncol; j++)
                  cout<<m[i][j]<<' ';
            cout<<endl;
      }
      return 0;
}
```

Додаток 2. Способи створення двовимірного масиву

```
const int sizex=3, sizey=4;
void show(double c[sizex][sizey]);
void show(double c[][sizey], int m);
//The following does not work:
//void show(double c[][], int m, int n);
void show(double **c, int m, int n);
void show(double *c, int m, int n);

int main()
{
      int m=sizex, n=sizey;
//1. Two-dinensional array
      double a[sizex][sizey];
      cout<<"Two-dimentional array a["<<sizex<<"]["<<sizey<<"] was
created"<<endl;
      for ( int i=0; i<sizex; i++)
            for (int j=0; j<sizey; j++)
            {
                  a[i][j]=10*i+j;
            }
      show(a);
      show(a, sizex);

//    Access to a[m][n] via *p
      cout<<"Access to a[m][n] via double *p"<<endl;
      double *p;
      p=(double*)a;
      show(p,m,n);

//2.Creating two-dimentional array using pointer

      cout<<"A two-dimentional array created using a pointer"<<endl;
      int mn=m*n;
      p= new double [mn];
      for ( i=0; i<m; i++)
            for (int j=0; j<n; j++)
            {
                  p[i*n+j]=10*i+j;
            }
      show(p,m,n);
      delete []p;

//2. Handler (pointer to pointers)
      cout<<"A two-dimentional array created using a handler (pointer to
pointers)"<<endl;
      double **pp;

      pp = new double* [m];
      for (i=0; i<m; i++)
            pp[i]=new double [n];

      for ( i=0; i<m; i++)
            for (int j=0; j<n; j++)
            {
                  pp[i][j]=10*i+j;
            }
```

```
	show(pp, m,n);

	for (i=0; i<m; i++)
		delete []pp[i];
	delete []pp;

	return 0;
}

void show(double c[sizex][sizey])
{
	cout<<"We can know its dimensions from the global context"<<endl;
	long unsigned int addr;
	for (int i=0; i<sizex; i++)
	{
		addr=(long)&c[i];
		cout<<addr<<'('<<&c[i]<<") c["<<i<<"]"<<endl;
		for (int j=0; j<sizey; j++)
		{
			addr=(long)&c[i][j];
			cout<<addr<<'('<<&c[i][j]<<")
c["<<i<<','<<j<<"]="<<c[i][j]<<endl;
		}
		cout<<endl;
	}
	cout<<endl;
}
void show(double c[][sizey], int m)
//void show(double c[][], int m, int n) does not work
{
	cout<<"We can pass a second dimension but not both as a
parameter"<<endl;
	long unsigned int addr;
	for (int i=0; i<m; i++)
	{
		addr=(long)&c[i];
		cout<<addr<<'('<<&c[i]<<") c["<<i<<"]"<<endl;
		for (int j=0; j<sizey; j++)
		{
			addr=(long)&c[i][j];
			cout<<addr<<'('<<&c[i][j]<<")
c["<<i<<','<<j<<"]="<<c[i][j]<<endl;
		}
		cout<<endl;
	}
	cout<<endl;
}
void show(double *c, int m, int n)
{
	cout<<"Array is passed as a pointer and both dimensions as parameters
too"<<endl;
	long unsigned int addr;
	for (int i=0; i<m; i++)
	{
		for (int j=0; j<n; j++)
		{
			addr=(long)&c[i*n+j];
			// c[i][j] does not work
			cout<<addr<<'('<<&c[i*n+j]<<")
c["<<i<<','<<j<<"]="<<c[i*n+j]<<endl;
		}
		cout<<endl;
	}
	cout<<endl;
```

```
}
void show(double **c, int m, int n)
{
	cout<<"Array is passed as a handler and both dimensions as parameters
too"<<endl;
	long unsigned int addr;
	for (int i=0; i<m; i++)
	{
		addr=(long)&c[i];
		cout<<addr<<'('<<&c[i]<<") c["<<i<<"]"<<endl;
		for (int j=0; j<sizey; j++)
		{
			addr=(long)&c[i][j];
			cout<<addr<<'('<<&c[i][j]<<")
c["<<i<<','<<j<<"]="<<c[i][j]<<endl;
		}
		cout<<endl;
	}
	cout<<endl;
}
```