Лабораторно упражнение № 3

Работа с едномерни масиви. Управляващи конструкции for и foreach

I. Теоретична част

1. Дефиниране на масиви. Достъп до елементи на масив

Масивът е несортирана последователност от елементи от един и същи тип, за разлика от класовете, при които полетата са от различен тип. Елементите на масива се съхраняват в последователен блок от паметта и са достъпни чрез индекс, който е цяло число и показва мястото на елемента в масива.

1.1. Деклариране на масиви и създаване на инстанции от тип масив.

Променлива от тип масив се декларира като се посочи името й, типа на елементите и квадратни скоби. Общият вид на декларацията на едномерен масив е следния:
тип [] имеМасив;
Пример:

```
int[] array;
```

Необходимо е да се отбележи, че броя на елементите не е част от декларацията и че синтаксисът на език C# изисква скобите да са преди името на масива.
Декларирането на двумерен масив има следният общ вид:
тип [,] имеДрумеренМасив;
Аналогично, тримерен масив ще бъде деклариран като:
тип [,,] имеТримеренМасив;
Масивите са адресни типове, независимо от типа на техните елементи. Това означава, че името на масива се съхранява в стека, а в динамичната памет се създава инстанция от тип масив, в която се съхраняват стойностите на самите елементи т.е. самият масив. Поради тази причина, в декларацията на масива не се посочва броя на елементите.
Създаването (дефиниране) на масив е едва след като се създаде инстанция (екземпляр) чрез ключова дума new.
Общият вид на дефиниране на едномерен масив е:
имеМасив = new тип [брой елементи];
Пример:

```
int[] array;
array = new int[4];
```

С първия програмен ред се декларира името array в стека. Първоначално, както всяка променлива от референтен тип, array е инициализирана със стойност null. С вторият програмен ред в динамичната памет се заделя място, необходимо да всичките елементи и адресът в стека се инициализира така че да сочи паметта, където е съхранен масивът.
Дефинирането на масив може да е и с един програмен ред:

```
int[] array = new int[4];
```

В език C# масивите са от динамичен тип и броя на елементите може да се определи и по време на изпълнение на програмния код. Например:

```
int n = Int32.Parse(Console.ReadLine());
double[] mA = new double[n];
```

Синтаксисът на дефиниране на двумерни и тримерни масиви е следния:
имеДвумеренМасив = new тип [бройРедове, бройСтълбове];
имеТримеренМасив = new тип [граница1, граница2, граница3];

Примери за създаване на многомерни масиви са:
```
int[,] matrix = new int[5, 4];
double [,] twoDimentionalArray = new double [3,2];
double [,,] threeDimentionalArray = new double [2,3,2];
```

1.2. Инициализиране на масиви

Когато се създаде обект (инстанция) от тип масив всички елементи на масива се инициализират със стойности по подразбиране в зависимост от техния тип. Възможно е още при дефинирането на масив, неговите елементи да бъдат инициализирани съа зададени стойности.
Общият вид на инициализацията е:
имеМасив = new тип [бройЕлементи] {списък стойности};
Пример:
```
int[] mB = new int[5] {1,2,3,4,5};
```

Не е необходимо стойностите във фигурните скоби непременно да бъдат константи, те могат да се изчислят и по време на изпълнение. Пример:
```
Random r = new Random();
int[] array = new int[4] { r.Next() % 10, r.Next() % 10,
                      r.Next() % 10, r.Next() % 10 };
```

В случая се използва генератор на случайни числа, с които да се инициализират елементите на масива.
Броят на стойностите между фигурните скоби задължително трябва да съответства на размера на масива. Например, посочените по-долу програмни редове ще предизвикат грешка по време на компилиране:
```
int[] array = new int[4] { 1, 2, 3 };
int[] array = new int[3] { 1, 2, 3, 4 };
```

При инициализиране на масиви изразът new и размерът на масива могат да бъдат пропуснати. Компилаторът изчислява размера на масива според броя на стойностите.
Пример:
```
int [] array ={1,2,3,4,5,6};
```

В случая е създаден масив от 6 елемента от тип int.

Инициализацията на многомерните масиви е подобна на едномерните.
Примери:
```
int[,] matrix = new int[2, 3] { { 1, 3, 5 }, { 2, 4, 6 } };

int[, ,] array3d = new int[,,] { { { 1, 2, 3 }, { 4, 5, 6 } },
```

```
                                        { { 7, 8, 9 }, { 10,
11, 12 } } };
```

1.3. Достъп до елементите на масив

За да се получи достъп до елемент на масив е необходимо да се зададе индексът, посочващ елемента.
Пример:

```
int a;
a = array[3];
```

В случая, на променливата a се присвоява стойността на елемент от масива с индекс 3.

Индексите на масивите в език C# започват от 0, както е и в програмни езици C и C++. При всяко индексиране (адресиране на елемент на масив) се проверява дали индексът попада в границите на масива. Ако се използва индекс по-малък от 0, равен или по-голям от дължината на масива, компилаторът генерира изключение IndexOutOf RangeException.

II. Задачи за изпълнение:

1. Да се създаде конзолно приложение на C#, с което се дефинира едномерен масив от n реални елемента с двойна точност. Да се въведат стйности на елементите и се изведат в прав и обратен ред. Да се изведат поотделно елементите с четен и нечетен индекс.

Упътване: Където е възможно да се използва конструкция foreach вместо конструкция for

2. Да се създаде конзолно приложение на C#, с което се дефинира едномерен масив от n целочислени елемента. Да се въведат стойности на елементите и да се създаде втори масив, в който да се копират само положителните елементи от първия масив. Да се изведат на екрана и двата масива.
3. Да се дефинира двумерен масив от имената на месеците и броя на дните във всеки месец. Да се изведе информация всеки месец какъв брой дни има.

Упътване: Да се използват масиви от тип string. Информацията за броя на дните да е представена като низове, а не като числови стойности.