Упражнение №7

МАСИВИ

Масив е група от данни от един и същи **тип** с общо име (**идентификатор**).

Масивът се дефинира по следния начин:

<тип> <идентификатор> [<размер>];

като **<размер>** е константен израз (целочислена или символна константа), определя броя на елементите в масива.

Достъпът до елемент на масив се осъществява чрез операциите:

– индексиране **[]** (директен достъп) – представлява относително отместване спрямо началото на масива;

<идентификатор> [<индекс>]

като **<индекс>** е целочислен израз, определящ мястото на елемента спрямо началото на масива; първият елемент на масива е с индекс **0**, а последният – с индекс **<размер>-1**.

– намиране на стойност по адрес ***** (косвен достъп) – абсолютно адресиране, по-бързо от индексиране; името на масива е указател към първия елемент на масива.

***(<указател> + <индекс>)**

Предаване на масив като параметър на функция се осъществява по адрес – предава се адресът на първия елемент от масива.

Пример 1. За едномерен целочислен масив да се съставят функции за въвеждане, извеждане, намиране на индекса на първия отрицателен елемент, намиране на индекса на последния отрицателен елемент, извеждане на всички отрицателни елементи и техните индекси, намиране на средноаритметичната стойност от всички елементи, намиране на средноаритметичната стойност от елементите, чиито индекси са в зададен интервал, и намиране на средноаритметичната стойност от елементите, чиито стойности са в зададен интервал. В програмата да се въведе едномерен целочислен масив с максимален брой 50 елемента, да се извикат функциите и се отпечатат получените резултати.

```
#include <stdio.h>
#define N 50                  //максимална размерност

int read_arr(int a[]);
void read_arr(int a[], int *n);
void write_arr(int a[], int n);
int negative(int a[], int n);
int rv_negative_1(int a[], int n);
int rv_negative_2(int a[], int n);
void all_negative(int a[], int n);
double avg(int a[], int);
double avg_index_bound( int a[], int, int inf, int sup);
double avg_value_bound( int a[], int, int inf, int sup);

int main()
{
      int arr[N];
      int n;
      read_arr( arr, &n );    // или n = read_arr( arr );
      write_arr( arr, n );
      int index = negative( arr, n );
```

```
	if( index >= 0 )
	{
		printf("\nIndeks na pyrvia otritzatelen element: %d\n", index);
	}
	else if (index == -1)
	{
		printf("\nNe sa namereni otritzatelni elementi!\n");
		return 0;
	}
	index = rv_negative_1( arr, n );    //или index=rv_negative_2(arr,n);
	if( index >= 0 )
	{
		printf("\nPIndeks na poslednia otritzatelen element: %d\n",
			index);
	}
	else if (index == -1)
	{
		printf("\nNe sa namereni otritzatelni elementi!\n");
	}
	all_negative( arr, n );
	double av = avg( arr, n );
	printf("\nSrednoaritmetichna stojnost ot vsichki elementi: %lf\n",
		av);
	int inf, sup;
	printf("\nVavedi dolna granitza za indeks v [dolna, gorna):");
	scanf("%d", &inf);
	printf("\nVavedi gorna granitza za indeks v [dolna, gorna):");
	scanf("%d", &sup);
	av = avg_index_bound( arr, n, inf, sup );
	printf("\nSrednoaritmetichna stojnost ot elementite s indeksi v ");
	printf("intervala [%d, %d): %lf\n", inf,sup, av);
	printf("\nVavedi golna granitza za stojnosti v [dolna, gorna]:");
	scanf("%d", &inf);
	printf("\nVavedi gorna granitza za stojnosti v [dolna, gorna]:");
	scanf("%d", &sup);
	av = avg_value_bound( arr, n, inf, sup );
	printf("\nSrednoaritmetichna stojnost ot elementite sys stojnosti ");
	printf("v intervala [%d, %d): %lf\n", inf, sup, av);
	return 0;
}
/* Въвежда елементите на едномерен целочислен масив a и връща броя на
елементите му – версия 1 */
int read_arr(int a[])
{
	int j;
	int dim;
	do
	{
		printf("\nVavedi razmernostta na masiva: ");
		scanf("%d", &dim);
	}
	while( dim < 1 || dim > N);
	for( j = 0; j < dim; j++ )
	{
		printf("a[%d]: ", j);
		scanf("%d", &a[j]);
	}
	return dim;
}
```

```
/* Въвежда елементите на едномерен целочислен масив a и записва броя на
елементите му в аргумента n – вариант 2 */
void read_arr(int a[], int *n)
{
	int j;
	do
	{
		printf("\nVavedi razmernostta na masiva: ");
		scanf("%d", n);
	}
	while( *n < 1 || *n > N);
	for( j = 0; j < *n; j++ )
	{
		printf("array[%d]: ", j);
		scanf("%d", &a[j]);
	}
}

/* Извежда на екрана елементите на едномерен целочислен масив a с n
елемента */
void write_arr( int a[], int n)
{
	int j;
	printf("\n(");
	for( j = 0; j < n; j++ )
	{
		printf(" %d", a[j]);
	}
	printf(")\n");
}

/* Връща индекса на първия отрицателен елемент на едномерен целочислен
масив a с n елемента или -1, ако в масива няма отрицателен елемент */
int negative(int a[], int n)
{
	int j;
	for( j = 0; j < n; j++ )
	{
		if( a[j] < 0 )
			return j;
	}
	return -1;
}

/* Връща индекса на последния отрицателен елемент на едномерен целочислен
масив a с n елемента или -1, ако в масива няма отрицателен елемент –
вариант 1*/
int rv_negative_1(int a[], int n)
{
	int j;
	for( j = n-1; j >= 0; j-- )
	{
		if( a[j] < 0 )
			return j;
	}
	return -1;
}
```

```
/* Връща индекса на последния отрицателен елемент на едномерен целочислен
масив a с n елемента или -1, ако в масива няма отрицателен елемент –
вариант 2*/
int rv_negative_2(int a[], int n)
{
	int j;
	int neg = -1;
	for( j = 0; j < n; j++)
	{
		if(a[j] < 0)
			neg = j;
	}
	return neg;
}

/* Извежда на екрана всички отрицателни елементите и техните индекси на
едномерен целочислен масив a с n елемента */
void all_negative( int a[], int n)
{
	int j;
	printf("\n");
	for( j = 0; j < n ; j++ ) {
		if( a[j] < 0 )
		{
			printf("a[%d] = %d < 0\n", j, a[j] );
		}
	}
}

/* Връща средноаритметичната стойност от елементите на едномерен целочислен
масив a с n елемента */
double avg(int a[], int n)
{
	int j, sum = 0;
	double res;
	for( j = 0; j < n; j++ )
	{
		sum += a[j];
	}
	res = (double)sum/n;	// n елемента от a[0] до a[n-1]
	return res;
}

/* Връща средноаритметичната стойност от елементите на едномерен целочислен
масив a с n елемента, чиито индекси са в зададен интервал [inf,sup) */
double avg_index_bound( int a[], int n, int inf, int sup)
{
	int j, sum = 0;
	double res;
	if ( inf >= sup || inf < 0 || sup >= n)
	{
		printf("\nNekorekten interval!\n");
		return 0.0; // NOT exit()...
	}
	for( j = inf; j < sup; j++ )
	{
		sum += a[j];
	}
	res = (double)sum/(sup - inf);		// !=0
	return res;
}
```

```
/* Връща средноаритметичната стойност от елементите на едномерен целочислен
масив a с n елемента, чиито стойности са в зададен интервал [inf,sup] */
double avg_value_bound( int a[], int n, int inf, int sup)
{
	int j, sum = 0, count = 0;
	double res;
	if ( inf > sup )
	{
		printf("\nNekorekten interval!\n");
		return 0.0; // NOT exit()...
	}
	for( j = 0; j < n; j++ ) {
		if( a[j] >= inf && a[j] <= sup )
		{
			sum += a[j];
			++count;		//брой на елементите в интервала
		}
	}
	if( count == 0 )
	{
		printf("\nNiama takiva elementi!\n");
		return 0.0;
	}
	res = (double)sum/count;
	return res;
}
```