

Особенности работы функций

1. Если функция возвращает значение через return, то ее можно использовать в выражениях и операторе вывода, например

```
n=a+b+sum(x,y)*5;  
printf(“%d”, sum(x,y));
```

2. Не следует возвращать из функции адреса локальных переменных, так как по завершении работы функции их значения уничтожаются и память освобождается.

3. Если параметр передается по значению, то соответствующим ему фактическим параметром может быть константа, переменная или выражение, например

```
s=sum(a, 3);  
s=sum(b, a*3);
```

Двумерные массивы. Ввод и вывод

```
void vv_matr(int a[][10], int *n, int *m) //или int (*a)[10]
{ //передаем массив указателей на строки, n и m – по адресу
int i, j;
printf("n = "); scanf("%d", n);
printf("m = "); scanf("%d", m);
printf("Введите матрицу размера %d на %d \n", *n, *m);
for(i=0; i<*n; i++)
    for(j=0; j<*m; j++)
        scanf("%d", a[i]+j);
}
```

//передаем n и m по значению

```
void viv_matr(int a[][10], int n, int m) {int i, j;
for(i=0; i<n; i++)
{
for(j=0; j<m; j++)
    printf("%4d", a[i][j]);
printf("\n");
}
}
```

```
int main()
{int a[10][10], n, m;
vv_matr(a, &n, &m);
viv_matr(a, n, m);
return 0;
}
```

Задача 1. Вычислить максимальный среди отрицательных элементов целочисленной матрицы A , не встречающихся в массиве B .

Ввод и вывод матрицы, ввод массива и вычисления оформить в виде отдельных функций.

Формальные параметры функции `maxot()` (для вычислений):

Вход:

n – количество строк в матрице A ,

m – количество столбцов матрицы A ,

a – указатель на массив указателей на строки матрицы A ,

k – количество элементов в массиве B ,

b – указатель на первый элемент массива B ,

Выход:

`flag` – признак существования `max` (`=0`, если `max` не найден).

Значение, возвращаемое функцией `maxot()` – искомый элемент.

```
#include "iostream"
#include "stdio.h"
#include "limits.h"
using namespace std;
void vv_matr(int a[][10], int *n, int *m);
void viv_matr(int a[][10], int n, int m);
int vv_mas(int b[]);
int maxot(int n, int m, int a[][10], int k, int b[], int *flag)
{
    int i,j,t,max = - LONG_MAX;
    *flag = 0;
```

```
for (i = 0; i < n; i++)
    for (j = 0; j < m; j++)
        if (a[i][j] < 0)
            {
                for (t = 0; t < k && a[i][j] != b[t]; t++);
                if (t == k)
                    if (a[i][j] >= max)
                        {
                            max = a[i][j];
                            *flag = 1;
                        }
            }
return max;
}
```

```
int main()
{
    setlocale(LC_ALL,"RUS");
    int n,m,a[10][10],k,b[10],max,flag;
    vv_matr(a, &n, &m);
    viv_matr(a, n, m);
    k = vv_mas(b);
    max = maxot(n,m,a, k, b, &flag) ;
    if (flag)
        printf("max = %7d", max);
    else
        printf("нет max");
    return 0;
}
/* здесь должны располагаться функции vv_matr, viv_matr,
vv_mas*/
```

/* в результате выполнения проверочной работы должна получиться примерно такая функция*/

```
int vv_mas(int b[])
```

```
{int nb, i;
```

```
printf(" Введите длину массива:");
```

```
scanf("%d", &nb);
```

```
printf(" Введите массив из %d элементов:\n");
```

```
for(i=0; i<nb; i++)
```

```
    scanf("%d", &b[i]);
```

```
return nb;
```

```
}
```

Задача 3. Определить адреса двух первых четных элементов массива A с использованием функции

```
#include "stdio.h"
#include "iostream"
void adres(int a[ ], int na,int **u1,int **u2)
{
    int *ua;
    *u1=*u2=NULL;
    for(ua=a;ua<na+a&&*u2==NULL;ua++)
    if (*ua %2==0)
    { if (*u1==NULL)
      *u1=ua;
      else *u2=ua;
    }
}
```



```
int main()
{
    setlocale(LC_ALL,"RUS");
int a[10],na,i,*u1,*u2;
//здесь ввод массива
    adres(a,na,&u1,&u2);
if (u1==NULL)
printf("Нет первого четного");
else { printf("Адрес первого четного %0X\n значение
    %d\n",u1,*u1); //16-ричный адрес
if (u2==NULL)
printf("Нет второго четного");
else {
        printf("Адрес второго четного %u\n значение
    %d\n",u2,*u2); //10-чный адрес
    }
}
return 0;
}
```