

# **ВАРИАНТЫ ОПИСАНИЯ И ИСПОЛЬЗОВАНИЯ ФУНКЦИЙ.**

### 3. Функция возвращает одно значение или несколько значений через свои аргументы.

#### при описании

- a) должен быть указан тип `void`;
- b) аргументы – как указатели `void fname (<тип> *ptx, ...)`
- c) внутри функции переменная должна быть описана –  
как указатель `*ptx=...;`

## вызов функции

**a)** является самостоятельным оператором.

**b)** в качестве возвращаемого параметра - адрес переменной

`fname (... , &x , ...)`

## Пример:

составить функцию, которая определяет сумму и разность двух величин  $s = a + b$ ,  $r = a - b$ .

```
void fsr (float a, float b, float *pts, float *ptr )  
{  
*pts = a+b;  
*ptr = a - b;  
}
```

```
void main()  
{  
float c =17, d= 12, sum, razn;  
fsr1(c, d, &sum, &razn);  
printf("\n sum =%f__ razn = %f", sum, razn);  
}
```

## Пример

Задан одномерный массив, найти  $(\max + \min) / 2$ ;

С помощью одной функции:

- сформировать массив,
- распечатать массив,
- найти  $\max$  и  $\min$ .

В основной программе найти и распечатать  $(\max + \min) / 2$ ;

```
void par(float *ptmax, float *ptmin)
{
int i;
float x[20];
*ptmax=-1000; *ptmin=1000;
for (i=0;i<20;i++)
{
x[i]=0.002*rand();
printf(" %6.1f \n", x[i]);
if (x[i]< *ptmin) *ptmin=x[i];
if (x[i]> *ptmax) *ptmax=x[i];
}
}
```

```
int main()
{
float maxr, minr, rez;
par (&maxr, &minr);
rez=(maxr+minr)/2;
printf("rez= %6.1f \n",rez);
}
```

## 4. В качестве аргумента функции используется массив.

*Возможны 3 варианта использования массива:*

1. размер массива указывается фиксировано:

```
< тип> namefun (< тип> x[50], ...)
```

```
x[i]
```

```
namefun ( d, ... ) ;
```



## 2. размер массива не указывается :

---

```
< тип> namefun (< тип> x[], int n...)
```

```
x[i]
```

```
namefun ( d, 20, ...);
```

---

Пример: Найти среднее арифметическое в массиве чисел, размером 20.

---

Составить функцию, в которой:

- сформировать массив,
- найти среднее арифметическое
- вернуть результат через имя.

В основной программе распечатать массив и результат.

```
/* функция формирует массив, находит среднее  
арифметическое и возвращает результат через имя  
функции */
```

```
float form_sr (float x[], int n)  
{  
    int i;  
    float s=0, sr;  
    for (i=0; i<n; i++)  
    {  
        x[i]=0.01*rand();  
        s=s+x[i];  
    }  
    sr= s/n;  
    return sr;  
}
```

```
void main()  
{  
float  a[20],sra;  
int i;  
sra= form_sr (a,20);  
for(i=0; i<n; i++)  
printf ("%5.1f",a[i]);  
printf ("sra=%5.2f",sra);  
}
```

3. массив объявляется через указатель :

```
< тип> namefun (< тип> *ptr, int n, ...)
```

```
* (ptr+i)
```

```
namefun ( &d[0], 20, ... ) ;
```

Пример.

Имеется два массива  $a[15]$ ,  $b[20]$ , элементы которых формируются по следующим формулам:

$$a_i = 1.5 * i * \sin(2.7 * i)$$

$$b_i = 3.9 * i * \sin(5.3 * i)$$

Найти в каждом массиве максимальный и минимальный элементы

- Массивы сформировать и распечатать с помощью одной функции;
- Найти  $\max$  и  $\min$  с помощью другой функции, результат вернуть через аргументы.
- В основной программе вывести результаты.

Общий вид формулы:  $x_i = k1 * i * \sin(k2 * i)$

```
/* функция формирует и возвращает массив */
```

```
void form_mas( float * x, int n, float k1,  
float k2)
```

```
{
```

```
    int i;
```

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

```
    {
```

```
        *(x+i)=k1*i*sin(k2*i);
```

```
        printf("%6.1f" , *(x+i));
```

```
    }
```

```
}
```

```
/* функция находит максимальное и минимальное  
значения и возвращает результат через  
аргументы*/
```

```
void max_min(float *x, int n,  
             float *pt_max, float *pt_min)  
{  
int i;  
*pt_max= *pt_min = *x;
```



```
for (i=0; i<n; i++)
{
if    (* (x+i) < *pt_min)
        *pt_min = * (x+i);

if    (* (x+i) > *pt_max)
        *pt_max = * (x+i);
}
}
```

```
int main()
{
    float  a[15], b[20]];
    float  amax, amin,    bmax, bmin;
    form_mas(&a[0], 15, 1.5, 2.7);
    max_min(&a[0], 15, &amax, &amin);
    printf("amax=%.2f  __  amin=%.2f \n",
           amax, amin);

    form_mas(&b[0],20, 3.9, 5.3);
    max_min( &b[0],20, &bmax, &bmin);
    printf("bmax=%.2f  __  bmin=%.2f\n",
           bmax, bmin);
}
```