Указатели към функции

```
int f(int);    // прототип на функция
```

```
int (*pf) (int);  //pf - указател към функция
pf=&f;   // задаване на стойност на указателя
pf=f;  // опростено
извикване:
m = f(5);
m = pf(5);
/*(опростяване на стандартното извикване:
m=(*pf)(5); )  */
```

ако указателя към функция се използва като параметър:

```
int func( int(*pf)(int) );

m=func(&f);  или  m=func(f); // опростено
```

Пример- стандартна библиотечна функция qsort(): сортиране на масив с елементи от произволен тип.

```
void qsort(void *base, size_t num_elements, size_t element_size,  int (*compare)(void const *, void  const *));
```

base – указател към първия елемент на сортирания масив (void *)
compare – функция за сравняване на два елемента – резултат

<0 – ако първата стойност е по-малка от втората
=0 – ако са равни
>0 – ако първата е по-голяма от втората

За масив от цели числа:

```
int compareInt(void const *i, void const *j){
   return  *((int*)i)- *((int *)j);;
}
```

За масив от низове:

```
int compareStr(void const *s1, void const *s2){
   return strcmp((char*)s1,(char *)s2);
}
```

```
#include <string.h>
#include <stdio.h>
#include <stdlib.h>
#define SIZE 8
#define STRSIZE 10
int compareInt(void const *i, void const *j){
   return  *((int*)i)- *((int *)j);
}
int compareStr(void const *s1, void const *s2){
   return strcmp((char*)s1,(char *)s2);
}
int main(){
  int i, arr[SIZE] = {8,16,1,43,34,17};
  char arStr[SIZE][STRSIZE] = {"Rosa","Anabel","Kiro", "Simeon","Zora"};
  qsort((void *)arr,SIZE,sizeof(int),compareInt);   // или &compareInt
  for(i=0;i<SIZE;i++)printf("%d  ",arr[i]);
  qsort((void *)arStr,SIZE,STRSIZE,compareStr);     // или &compareStr
  puts("");
  for(i=0;i<SIZE;i++)printf("%s  ",arStr[i]);
  return 0;
}
```

Масиви от функции

```
#include <stdio.h>
#include <stdlib.h>

int sum(int , int );
int subtract(int , int );
int mul(int , int );
int dv(int , int );

int main(int argc, char *argv[]){
  int result;
  int i, j, op;
  int (*p[4]) (int, int);

  p[0] = sum; /* address of sum()   */
  p[1] = subtract; /* address of subtract() */
  p[2] = mul; /* address of mul() */
  p[3] = dv; /* address of dv() */
// или  int(*p[4])(int,int) ={sum,subtract,mul,div};          инициализация при декларирането на масива
  printf("Enter two numbers\n number 1: ");
  scanf("%d", &i);
  printf (" number 2: ");
  scanf("%d", &j);
  do{
    printf("0: Add, 1: Subtr, 2: Mult, 3: Div, 4: End\n");
    do {
      printf("Enter number of operation (0 - 4): ");
      scanf("%d", &op);
    } while(op<0 || op>4);
    if(op==4)break;

    result = (*p[op]) (i, j);      //или    result = (p[op]) (i, j);  съкратен запис
    printf("The result is %d\n", result);
  }while(1);
  return 0;
}

int sum(int a, int b){ return a + b; }
int subtract(int a, int b) {return a - b; }
int mul(int a, int b) {  return a * b;}
int dv(int a, int b){
  if(b)   return a / b;
  else    return 0;
}
```