Любая функция имеет физическое местоположение в памяти.

Адрес функции является входной точкой в тело функции.

Этот адрес может быть присвоен указателю.

Следовательно, указатель на функцию может быть использован для вызова функции.

В общем виде указатель на функцию определяется следующим образом:

тип функции (*имя указателя функции)(список параметров);

Например, int (*fprt)(int);

Определяется указатель fprt, на функцию с целочисленным параметром, возвращающую целочисленное значение.

Пример.

Определим функцию f1, вычисляющую сумму двух чисел, и функцию f2, вычисляющую разность этих же чисел.

```
double SUM (double a, double b)
{
   return a + b;
}
double RAZ (double a, double b)
{
   return a - b;
}
void main ( )
{
  double q =-44, w = 70;
  double d;
  double (*ptr_f) (double, double);       /* объявили указатель на функцию*/

  ptr_f = SUM;                             /* ptr_f присвоили адрес SUM */
  d = (*ptr_f) (q, w);                    /* вызвали SUM по адресу  */


  ptr_f = RAZ;                            /* присвоили адрес RAZ */
  d = (*ptr_f) (q, w);                    /* вызвали RAZ */
}
```

Следует отметить, что вызов функции можно было записать следующим образом:

```
ptr_f (q, w);
```

Указатель на функцию может быть параметром для некоторой функции.

```
.    .    .

int ff(int a ,int b, int (*f)(int,int))
{
    return f(a,b);
}
int  main()
{

  int (*f_p)(int,int);

.    .    .

  cout<<endl<<endl<<"указатель на функцию-параметр"<<endl;

  cout<<ff(3,4,sum)<<endl;
  cout<<ff(3,4,mult)<<endl;
```

Массивы указателей.

В языке C/C++ можно создавать массивы указателей. Объявим массив целочисленных указателей из 4 элементов:

```
int *m[4];      /*  Пока еще элементы массива не имеют значений.*/
```

Рассмотрим следующий простейший пример

```
int x, a=3, b=10, c=-4, d=-6;

int *m[4];

m[0]=&a;
m[1]=&b;
m[2]=&c;
m[3]=&d;
```

Для того чтобы получить **значение** какой-либо переменной, например a, можно написать ***m[0]** или указать имя переменной a.

Допустим, увеличим значение переменной a в два раза и положим полученное значение в переменную x.

Возможны следующие операторы:

```
x = *m[0]*2;
или    x = a*2;
```

Массив указателей можно передавать в функцию в качестве параметра.

Указатели на указатели.

В случае обычных указателей, указатель содержит адрес некоторого участка памяти, содержащего некоторое значение.

В языке C/C++ можно создать указатель «на указатель». В этом случае первый указатель содержит адрес второго, который в свою очередь содержит адрес участка памяти.

```
int a = 7;
int *ptr = &a;
int **ptr_1 = &ptr;
```

```
. . .
const int  n = 4;
  int **a;
  int i, j;
  a = new int* [n];
  for (i = 0; i<n; i++ )  {
    a[i] = new int[n];
     for (j = 0; j<n; j++)
       a[i][j] = i + j + 1;
   }
   for ( i = 0;i<n;i++)  {
     for (j = 0;j<n;j++)
       cout<<a[i][j]<<"  " ;
     cout<<endl;
   }
```

```
. . .
for (i = 0; i<n; i++ )   {
     a[i] = new int[i];
      for (j = 0; j<=i; j++)
        a[i][j] = i + j + 1;
   }
   for ( i = 0;i<n;i++)   {
      for (j = 0;j<=i;j++)
        cout<<a[i][j]<<"   " ;
      cout<<endl;
   }
```

Освобождение памяти

```
delete [ ] a;
```

```
#define n 6
void vvod(int *m)
{
    for(int i = 0;i<n;i++)
      cin >> m[i];
}
void print(int *m)
{
    for(int i = 0;i<n;i++)
      cout << m[i]<<"   ";
    cout<<endl;
}
```

```
void main()
{
    int m[n],*p_min[n],*temp;

    clrscr();
    vvod(m);
    print(m);
    for (int i = 0;i<n;i++)
      p_min[i] = &m[i];
```

```
// сортировка
    for(i = 0;i<n-1;i++)
     for (int j = i+1;j<n;j++)
    if (*p_min[i]<*p_min[j]) {
        temp = p_min[i];
        p_min[i] = p_min[j];
        p_min[j] = temp;
    }
 // ПЕЧАТЬ ЭЛЕМЕНТОВ МАССИВА
//ОСВОБОЖДЕНИЕ ПАМЯТИ
    getchar();
}
```

```
#include <iostream.h>
#include <conio.h>
#include <stdio.h>
void main()
{
    char str[] = " ASDFG";
    int i;
    char *s;
    s = new  char [sizeof(str)];
    for(i = 0;i<sizeof(str);i++)
        *(s + i) = *(str + i);
    cout<<s<<endl;
```

```
char * f(char  *str, int k)
{
    int i;
    char *s;

    s= new char[k];
    for( i = 0; i<k;i++)
      *(s + i) = *( str + i);
    return s;
}
```

```
char * f(char *str, int k);
void main()
{
    char str[] = "ASDFGH";
    int k;
    char *s;

    clrscr();
    k = sizeof(str)+1;
    s = f(str,k);
    cout<<"s = "<<s<<endl;
  getchar();
}
```