

# Лекция 4 Особенности использования указателей и ссылок

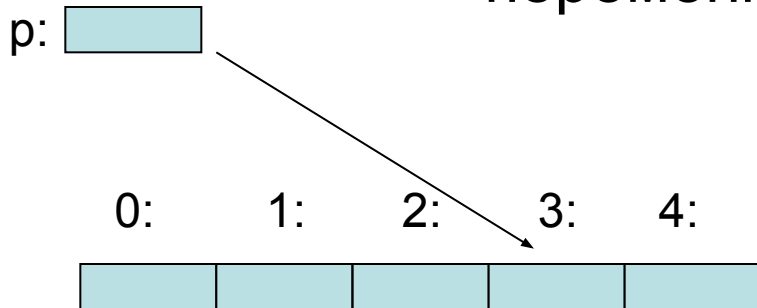
О.С. Трушин  
Зав. лаб. ЯФ ФТИАН РАН,  
Доцент кафедры нанотехнологии в  
электронике

# План

- Особенности работы с указателями
- Особенности использования ссылок
- Различные способы передачи данных в функцию
- Массивы указателей
- Указатели на функции
- Перегрузка функций
- Шаблоны функций
- Рекурсия
- Примеры обработки символов и строк

# Указатели

Указатель это специальная  
переменная для хранения адреса памяти



`char* p;`

Пример:

`char v[5];`

`char* p=&v[3]`

\* - операция «взять содержимое»

& - операция «взять адрес»

# Разыменование указателей

Операция разыменования указателя – это получение данных, на которые он ссылается

`x = *y;`

Пример:

```
int    i=20;  
int*   iptr = &i;  
int    j;  
int    k=50;
```

```
j = *iptr; // j присваивается значение i  
*iptr = k; // i присваивается значение k
```

# Арифметические операции с указателями

Для указателей участвующих в выражениях определены специальные правила выполнения арифметических операций: СЛОЖЕНИЯ, ВЫЧИТАНИЯ, ИНКРЕМЕНТА, ДЕКРЕМЕНТА

Пример:

```
int* p;  
int n;
```

```
p=p+n;
```

Указатель смещается на  $n$  – позиций  
в массиве целых чисел

# Инициализация указателя с помощью оператора new

Оператор new – обеспечивает динамическое выделение памяти «из кучи»

Пример:

```
void main(){  
  
char* word;  
  
word=new char;  
  
cin >> word;  
  
}
```

# Ссылки

Ссылки используются в качестве альтернативных имен переменных

Пример:

```
int i=20;
```

```
int &r=i;
```

```
r++; // увеличивает  
i на единицу
```

# Передача аргументов в функцию по значению

**При передаче аргументов в функцию по значению их величины копируются**

Пример объявления функции:

```
Int squareByValue(int a)
{
    return a * a;
}
```

Пример вызова функции:

```
void main()
{
    int x=2;
    int z=squareByValue(x);
    cout << " x=" << x << " z=" << z ;
}
```

Ответ: x=2 z=4



# Передача аргументов в функцию по ссылке

**При передаче аргумента в функцию по ссылке ей передается только адрес переменной**

Пример объявления функции:

```
Int squareByReference(int &a)
{
    return a *= a;
}
```

Пример вызова функции:

```
void main()
{
    int x=2;
    int z=squareByReference(x);
    cout << " x=" << x << " z=" << z ;
}
```

Ответ: x=4 z=4

# Передача аргументов в функцию с помощью указателя

**При передаче аргумента в функцию с помощью указателя  
ей передается только адрес переменной**

Пример объявления функции:

```
Int squareByPointer(int *nPtr)
{
    return *nPtr *= *nPtr;
}
```

Пример вызова функции:

```
void main()
{
    int x=2;
    int z=squareByPointer(&x);
    cout << " x=" << x << " z=" << z ;
}
```

Ответ: x=4 z=4

# Массивы указателей

Массивы могут содержать указатели. Типичный пример обработка массива строк.

Пример1 инициализация при объявлении:

```
char* suit[3]={“Черви”, “Бубны”, “Пики”}
```

Пример2 инициализация с помощью оператора new:

```
void main(){  
    ifstream file(“test.txt”);  
    char* line[3];  
  
    for(int i=0; i<3; i++){  
        line[i]=new char;  
        file.getline(line[i],80,‘\n’);  
        cout << line[i] << endl;  
    }  
}
```

# Указатели на функции

Указатель на функцию содержит адрес функции в памяти.  
Их можно передавать функциям, возвращать из функций,  
хранить в массивах и присваивать другим указателям на функции.

Пример объявления функций:

```
void bubbleSorting( int *work, int size, int (*compare) (int,int) )  
{  
    ...  
    if( (*compare)(work[count], work[count+1])  
        swap(&work[count], &work[count+1]);  
}
```

```
int ascending(const int a, const int b)  
{ return b<a; }  
int descending(const int a, const int b)  
{ return b>a; }
```

Пример вызова функции:

```
bubble(a,size,ascending);
```

# Перегрузка функций

C++ позволяет определить несколько функций с одним и тем же именем. Эта особенность называется перегрузкой функций.

Пример:

Сигнатура функции => комбинация имени и типа параметров
--

```
#include <iostream>
```

```
int square(int x) { return x*x;}
```

```
double square(double y) { return y*y};
```

```
void main()  
{  
    cout << square(2) << endl;  
    cout << square(2.5);  
}
```

# Шаблоны функций

Шаблоны средства генерации кода

Пример объявления шаблона:

```
template <class T>
T maximum( T x1, T x2, T x3)
{
    T max= x1;
    if( x2 > max) max=x2;
    if( x3 > max) max=x3

    return max;
}
```

Пример использования:

```
void main()
{
    int x1=1,x2=2,x3=3;
    cout << maximum(x1,x2,x3);

    double y1=1.1,y2=1.2,y3=1.3;
    cout << maximum(y1,y2,y3);

    char z1='a',z2='b',z3='c';
    cout << maximum(z1,z2,z3);

}
```

# Рекурсия

Рекурсивная функция – это функция, которая вызывает сама себя.

Пример: вычисление факториала

$$N! = N * (N-1)!$$

```
int factorial( int x)
{
    if(x<=1)return 1;
    else
    return x*factorial(x-1);
}
```

```
void main()
{
    cout << factorial(10);
}
```

# Пример1 Подсчет числа вхождений данной буквы

```
void main()
{
char* line=" Text for work \n";
char x='r';

int count=0;

while(*line != '\n')
{
    if( *line == x )count++;
    ++line;
}

cout << " count=" << count << endl;

system("pause");
}
```



# Пример2 Обработка строки с использованием указателя

```
void main()
{
ifstream file1("text.txt");
int count=0;
char* line = new char[80];

file1.getline( line ,80, '\n');
char x = 'a';

while(*line != '\0')
{
    if( *line == x )count++;
    ++line;
}

cout << " count=" << count << endl;

system("pause");
}
```

## Пример3 Работа с массивом строк

```
void main() {  
    int count=0;  
    char x='e';  
    ifstream file1("text.txt");  
    char line[80];  
    int i;  
    while( file1.getline( line ,80, '\n') ) {  
        i=0;  
        while(line[i] != '\0') {  
            if( line[i] == x1 )count1++;  
            if( line[i] == x2 )count2++;  
            if( line[i] == x3 )count3++;  
            i++;  
        }  
    }  
    cout << x << " count=" << count << endl;  
    system("pause");  
}
```

# Составные элементы данных: структуры struct

struct Имя

{

Тело

};

Пример объявления:

```
struct Persona
```

```
{
```

```
char name[10];
```

```
char surname[20];
```

```
int age;
```

```
};
```

Пример создания  
экземпляра

```
void main()
```

```
{
```

```
Persona stud[20];
```

```
cout << stud.name;
```

```
}
```

# Примеры использования объектов типа struct

```
struct Persona
{
    char Name[10];
    char FName[15];
    char Surname[20];
    int Age;
    char City[10];
};

#include "persona.h"
using namespace std;

void main()
{
    Persona student[20];
    ifstream myfile("spisok.txt");

    int i=0;
    while( myfile >> student[i].Name >> student[i].FName >>
student[i].Surname >> student[i].Age >> student[i].City )
    {
        cout << " i=" << i << student[i].Name << " "<<
student[i].FName << " " << student[i].Surname << " "<<
student[i].Age << " " << student[i].City << endl;
        i++;
    }
    system("pause");
}
```