Семинар 3.
Указатели,
ссылки,
массивы,
адресная арифметика

1. Указатели

Специальными объектами в языках C и C++ являются указатели. Различают указатели-переменные и указатели-константы. Значениями указателей служат адреса участков памяти, выделенных для объектов конкретных типов.

1. Указатели

Определение и описание указателя имеет вид:
type *имя_указателя

type – обозначение типа
имя_указателя – идентификатор
* - унарная операция обращения/доступа по адресу (раскрытия ссылки, разыменования)

1. Указатели

В совокупности имя типа и символ * перед идентификатором воспринимаются как обозначение особого типа данных «указатель на объект данного типа».
Определение int *i1p, *i2p, *i3p, i;
вводит три указателя на объекты целого типа и одну переменную i целого типа.

1. Указатели

В совокупности имя типа и символ * перед идентификатором воспринимаются как обозначение особого типа данных «указатель на объект данного типа».
Определение int *i1p, *i2p, *i3p, i; вводит три указателя на объекты целого типа и одну переменную i целого типа.

1. Указатели

При определении указателя можно выполнить его инициализацию:
type *имя_указателя инициализатор;
В качестве инициализатора могут быть:

- явно заданный адрес участка памяти
- указатель, уже имеющий значение
- выражение, позволяющее получить адрес объекта с помощью операции &

1. Указатели

Пример. Знакомство с указателями.

```
char cc = 'd';
char *pc = &cc;
char *ptr(0);
char *p;
cout << "\n cc = " << cc << " and *pc=" << *pc;
ptr = &cc; // эквивалентно ptr = pc
cout << "\n *ptr= " << *ptr;
*ptr = 'y';
cout << "\n cc = " << cc;
return 0;
```

1. Указатели

Возможные операции над указателями:

- операция разыменования (*)
- приведение типов
- присваивание
- получение адреса
- сложение, вычитание, инкремент, декремент
- операции сравнения

2. Ссылки

Ссылка – другое имя уже существующего объекта. Основные достоинства ссылок проявляются при работе с функциями. Определение ссылки:
type & имя_ссылки инициализатор;

2. Ссылки

Пример. Знакомство со ссылками.

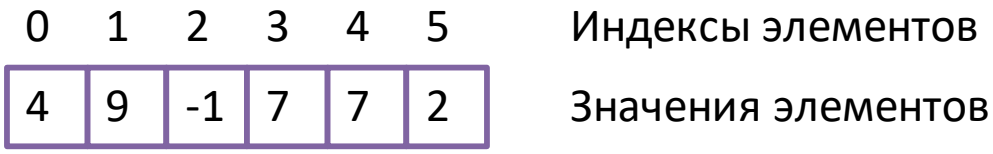
```
int L = 80;
int &RL = L;
cout << "RL = " << RL << endl;
return 0;
```

2. Ссылки

Имена переменной и ссылки, «настроенной» на эту переменную, полностью равноправны и соотносятся с одним и тем же участком памяти. Присваивание значения переменной приводит к изменению значения, связанного со ссылкой, и наоборот.

3. Массивы

Массив – один из агрегатных типов языка Си ++. Массив отличается тем, что все его элементы имеют один и тот же тип и что все его элементы расположены в памяти подряд.



3. Массивы

Примеры определения массивов:

```
char CH[] = {'A', 'B', 'C', 'D'};
int IN[6] = {10, 20, 30, 40};
char STR[] = "ABCD "; // 5 элементов!

double B[4] = {1, 2, 3, 4, 5, 6};
// ошибка инициализации
```

3. Массивы

Операция имя_массива[индекс] называется операцией индексации элементов массива. Пример.

```
int arr[] = {10, 20, 30, 40};
cout << "arr[0] = " << arr[0] << endl;
cout << "arr[1] = " << arr[1] << endl;
cout << "arr[2] = " << arr[2] << endl;
cout << "arr[3] = " << arr[3] << endl;
return 0;
```

3. Массивы

Ещё один пример индексации массивов.

```
int arr[4];
arr[0]=10;
arr[1]=20;
arr[2]=30;
arr[3]=40;
cout << "summa = " << arr[0]+arr[1]+arr[2]+arr[3];
return 0;
```

3. Массивы

Важно! Первый элемент массива в Си++ имеет индекс 0. Индекс = расстояние от начала массива. Последний элемент массива arr[n] имеет индекс (n-1).

3. Массивы

С помощью операции sizeof(имя_массива) можно определить размер массива в байтах. Так как все элементы массива имеют одинаковый размер, то частное sizeof(имя_массива) / sizeof(имя_массива[0]) определяет количество элементов в массиве

3. Массивы

Пример. Ввод и вывод массива.

```
#include "stdafx.h"
#include <iostream>
using namespace std;
# define N 5

int _tmain(int argc, _TCHAR* argv[])
{
  int arr[N];
  for(int i = 0; i < N; i++){ // Цикл для ввода элементов массива
    cout << "Enter element #" << i << ": ";
    cin >> arr[i];
  }
  for(int i = 0; i < N; i++) // Цикл для вывода элементов массива
    cout << "\nEnter element #" << i << " = " << arr[i];
  return 0;
}
```

3. Массивы

Пример. Ввод и вывод массива.

```
#include "stdafx.h"
#include <iostream>
using namespace std;
# define N 5

int _tmain(int argc, _TCHAR* argv[])
{
  int arr[N];
  for(int i = 0; i < sizeof(arr)/sizeof(arr[0]); i++){
    cout << "Enter element #" << i << ": ";
    cin >> arr[i];
  }
  for(int i = 0; i < sizeof(arr)/sizeof(arr[0]); i++)
    cout << "\nEnter element #" << i << " = " << arr[i];
  return 0;
}
```

3. Массивы

Многомерный массив в Си++ является массивом массивов. Определение многомерного массива:
type имя_массива[K1][K2]…[KN];

3. Массивы

Например,
int arr3[4][3][6];
Это массив из 4 элементов, каждый из которых является двумерным массивом 3x6. В памяти массив arr3 размещается в порядке возрастания самого правого индекса. Младший элемент arr3[0][0][0], далее – arr3[0][0][1], последний – arr3[3][2][5].

3. Массивы

Например,
int arr3[4][3][6];

0	0	0
0	0	1
0	0	2
0	0	3
0	0	4
0	0	5
0	1	0
0	1	1
0	1	2
0	1	3
0	1	4
0	1	5
0	2	0
0	2	1
…	…	…
3	2	4
3	2	5

3. Массивы

Пример. Ввод и вывод двумерного массива.

```
#define N 5 // количество строк
#define M 3 // количество столбцов
…
 int arr[N][M];
 for(int i = 0; i < N; i++)
   for(int j = 0; j < M; j++){
     cout<<"Enter element ["<<i<<"]["<<j<<"]:";
     cin >> arr[i][j];
   }
 for(int i = 0; i < N; i++){
   for(int j = 0; j < M; j++)
     cout<<arr[i][j]<<'\t';
   cout << endl;
 }
 return 0;
```

3. Массивы

Пример. Поиск максимального элемента в двумерном массиве.

```
int max = arr[0][0];
for(int i = 0; i < N; i++)
   for(int j = 0; j < M; j++)
      if(max<arr[i][j]) max = arr[i][j];
cout << "max = " << max << endl;
```

3. Массивы

Задания.

1. В одномерном массиве из N элементов найти и вывести на экран индексы двух соседних элементов, разница между которыми по модулю максимальна. Если таких пар несколько, вывести первую.
2. Дан массив целых чисел. Проверить, есть ли в массиве одинаковые элементы.
3. Дан массив целых чисел. Найти количество элементов, превосходящих среднее значение элементов массива.
4. Создать одномерный массив из N элементов, заполнить массив простыми числами (простое число – это натуральное число, которое делится без остатка только на 1 и на самого себя): в первом элементе – 2, во втором – 3, в третьем – 5, в четвёртом – 7 и т.д..
5. В двумерном массиве N на M элементов вычислить сумму элементов в каждой строке и в каждом столбце.
6. В двумерном массиве N на M различных целочисленных элементов найти максимальный элемент и вывести на экран строку, содержащую максимальный элемент и столбец, содержащий максимальный элемент.

3. Массивы

Динамическое распределение памяти – способ выделения оперативной памяти для объектов, при котором выделение памяти осуществляется во время выполнения программы. Для создания динамических массивов в Си++ необходимо использование операции new. Для освобождения памяти – операцию delete.

3. Массивы

Только первый (самый левый) размер массива может быть задан с помощью переменной.

3. Массивы

Пример. Создать динамический массив, присвоив в его элементы значения от 1 до n. Вывести полученный массив на экран.

```
int n;
cout << "Enter n:";
cin >> n;
double *matr; //Указатель для массива указателей
matr = new double [n]; //Массив указателей double *
for(int i = 0; i<n; i++)
  matr[i] = i + 1;
for(int i = 0; i<n; i++)
  cout << matr[i] << '\t';
delete [] matr;
return 0;
```

3. Массивы

Пример. Единичная матрица с изменяемым порядком.

```
int n;
cin >> n;
double **matr; //Указатель для массива указателей
matr = new double *[n]; //Массив указателей double *
for(int i = 0; i<n; i++){
  matr[i] = new double[n];
  for(int j = 0; j<n; j++)
    matr[i][j] = (i != j ? 0 : 1);
}
cout << "Result:\n";
for(int i = 0; i<n; i++){
  for(int j = 0; j<n; j++)
    cout << '\t' << matr[i][j];
  cout << endl;
}
for(int i = 0; i < n; i++)
  delete [] matr[i];
delete [] matr;
return 0;
```

3. Массивы

Пример. Создать динамический массив размерности 3 и в каждый элемент этого массива присвоить сумму индексов этого элемента.

```
int n;
cin >> n;
double ***matr;
matr = new double ** [n];
for(int i = 0; i<n; i++){
  matr[i] = new double *[n];
  for(int j = 0; j<n; j++){
    matr[i][j] = new double [n];
    for(int k = 0; k<n; k++)
      matr[i][j][k] = i + j + k;
  }
}
for(int i = 0; i<n; i++)
  for(int j = 0; j<n; j++)
    for(int k = 0; k<n; k++)
      cout<<"matr["<<i<<"]["<<j<<"]["<<k<<"]="<<matr[i][j][k]<<'\n';
```

3. Массивы

Задания.

1. Освободить память, выделенную для динамического массива matr (на предыдущем слайде).
2. Создать двумерный динамический целочисленный массив размера N на M элементов. В элементы массива присвоить значения, равные произведению номера строки на номер столбца.

4. Адресная арифметика

Значение выражения *указатель зависит не только от значения указателя, но и от типа.

Арифметические операции с адресами заключаются в том, что при увеличении адреса на 1 результатом является адрес соседнего «справа» блока памяти длины sizeof(тип данных), а при уменьшении на 1 – адрес соседнего «слева» блока памяти.

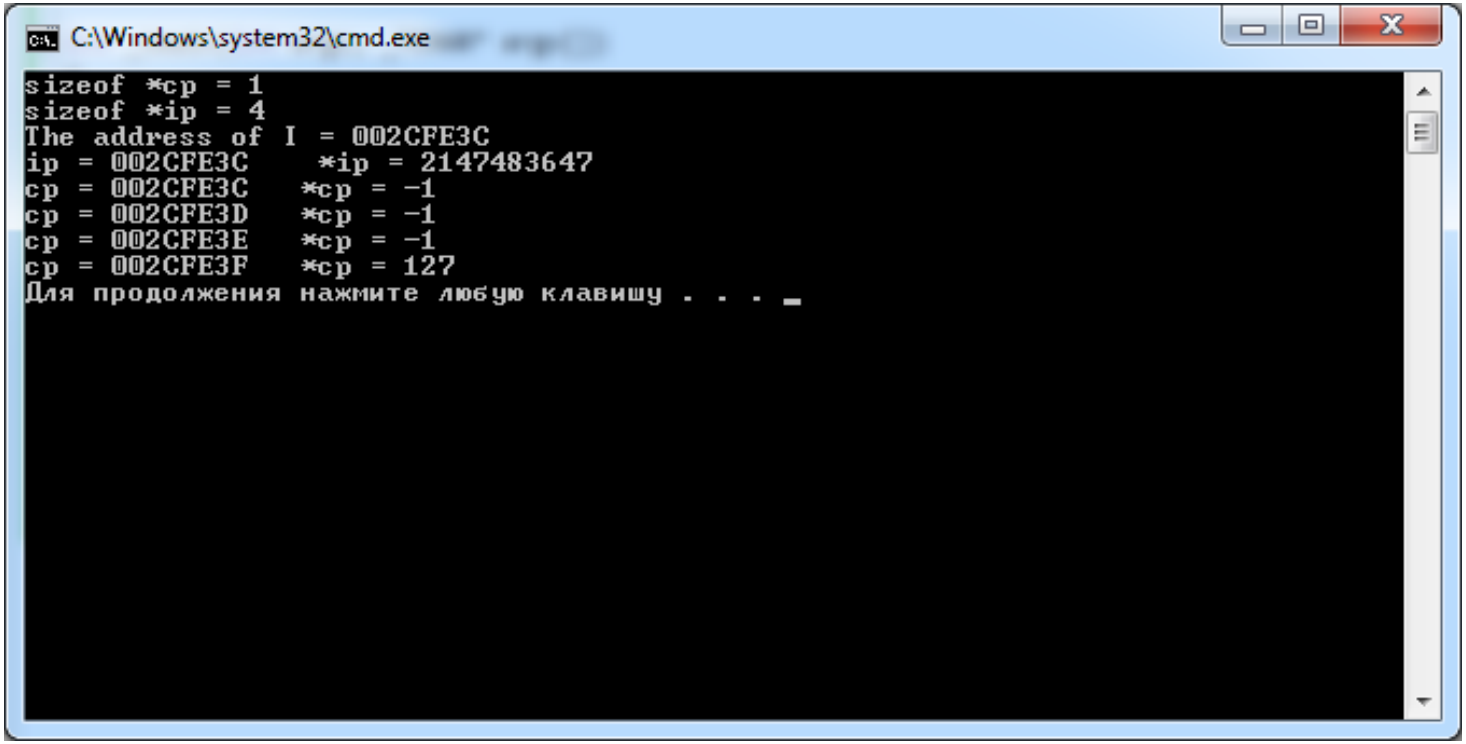
4. Адресная арифметика

Пример. Целое число: что внутри?

```
int I = INT_MAX;
char *cp = (char*)&I;
int *ip = &I;
cout << "sizeof *cp = " << sizeof * cp << endl;
cout << "sizeof *ip = " << sizeof * ip << endl;
cout << "The address of I = " << &I << endl;
cout << "ip = " << (void*) ip << "\t *ip = " << *ip << endl;
for(; cp < (char *)ip + 4; cp++)
   cout << "cp = " << (void *)cp << "\t*cp = " << (int)*cp << endl;
return 0;
```

4. Адресная арифметика

Пример. Целое число: что внутри?



4. Адресная арифметика

В языке Си++ истинно:

имя_массива
== &имя_массива
== &имя_массива[0]

Имя массива – константный указатель того типа, к которому отнесены элементы массива.

4. Адресная арифметика

Пример. Адресная арифметика и массивы. Вывод элементов символьного массива на экран с помощью операции разыменования (*).

```
char x[] = "QWERTY";
int i = 0;
while(*(x+i)!='\0')
cout << *(x + i++) << endl;
return 0;
```

4. Адресная арифметика

Задания.

1. Ввести и вывести элементы одномерного целочисленного массива размера N без операции [] (применять адресную арифметику и операцию разыменования).
2. Вывести все отрицательные элементы одномерного целочисленного массива размера N без операции [].
3. Вывести те элементы одномерного целочисленного массива размера N на экран, которые имеют чётный индекс.

5. Сортировка пузырьком

```
int a[N] = {6, 3, 9, 10, 0, 12, 5, -1, 4, 9};
int vspom;
for(int i = 0; i < N - 1; i++)
  for(int j = 0; j < N - i - 1; j++)
    if(a[j] > a[j + 1]){
      vspom = a[j];
      a[j] = a[j + 1];
      a[j + 1] = vspom;
    }
for(int i = 0; i < N - 1; i++)
  cout << a[i] << '\t';
return 0;
```