

Тема 4. Оператори мови C ++.

Умовні оператори, оператори вибору switch, оператори циклу (for, while, do while), оператори переходу (break, continue, return, goto), порожній оператор.

Оператори вибору

У мові програмування C ++ існує два оператора вибору:

- 1) Оператор вибору **if**
- 2) Оператор вибору **switch**

Оператори вибору дозволяють прийняти програмі рішення, ґрунтуючись на істинності або хибності умови. Якщо умова є істиною (true), оператор в тілі if виконується, після чого виконується наступний по порядку оператор. Якщо умова хибна (false), оператор в тілі if не виконується (ігнорується або пропускається) і відразу ж виконується наступний оператор.

```
a=5; b = 7;  
if (a > b){  
    // код невиконується  
}
```

Оператори вибору

Також іноді зручно використати конструкцію else if.

Приклад:

```
a=5; b = 7;
```

```
if (a < 4){  
    // код невиконується  
}else if(a > b){  
    // код невиконується  
}else if(a > 4){  
    // код виконується  
}
```

Оператори вибору

// синтаксис оператора вибору switch

switch (a){

case 0:

//код

break;

case 1:

//код

break;

. . .

default:

//код

}

a – змінна

break; - оператор виходу

case 0: - випадок, якщо $a = 0$.

case 1: - випадок, якщо $a = 1$.

Оператори циклу з передумовою

// синтаксис оператора циклу while

```
while (a>b){ //якщо умова виконюється (a>b) – правда, то код виконується  
    //код  
}
```

Приклад. Обчислити факторіал числа k.

```
int k = 5; //факторіал 5
```

```
int i= 1;
```

```
int result = 1;
```

```
while(i <= k){
```

```
    result *= i;
```

```
    i++;
```

```
}
```

```
//result містить значення факторіала
```

Оператори циклу з післяумовою

```
// синтаксис оператора циклу do {} while();  
do{  
    //код  
}while (a>b); //якщо умова виконюється (a>b) – правда, то цикл виконується
```

Приклад. Обчислити факторіал числа k.

```
int k = 5; //факторіал 5  
int i= 1;  
int result = 1;  
do{  
    result *= i;  
    i++;  
} while(i <= k);
```

```
//result містить значення факторіала
```

Оператори покрокового циклу

// синтаксис оператора циклу

```
for(int i = 0; i < n; i++) {  
    //код  
}
```

Вираз збільшення

Умовний вираз

Початковий вираз

Приклад. Обчислити суму перших цілих чисел до k. З Edit1 зчитується значення k, у Edit2 виводиться результат.

```
int k = StrToInt(Edit1->Text);  
int result = 0;          //початкове значення суми  
for(int i = 1; i <= k; i++){  
    result += i;  
}  
Edit2->Text = IntToStr(result);
```

Оператори циклу

Тело оператора **for** выполняется до тех пор, пока *условное выражение* не станет ложным (равным 0). *Начальное выражение* и *выражение приращения* обычно используются для инициализации и модификации параметров цикла и других значений. *Начальное выражение* вычисляется один раз до первой проверки *условного выражения*, а *выражение приращения* вычисляется после каждого выполнения *оператора*. Любое из трех выражений заголовка цикла, и даже все три могут быть опущены (не забывайте только оставлять точки с запятой). Если опущено *условное выражение*, то оно считается истинным, и цикл становится бесконечным.

Оператор пошагового цикла в языке C++ является гибкой и удобной конструкцией, поэтому оператор цикла с предусловием **while** используется в языке C++ крайне редко, т.к. в большинстве случаев удобнее пользоваться оператором **for**.

Оператор розриву **break**;

Оператор разрыва прерывает выполнение операторов **while**, **do**, **for** и **switch**. Он может содержаться только в теле этих операторов. Управление передается оператору программы, следующему за прерванным. Если оператор разрыва записан внутри вложенных операторов **while**, **do**, **for**, **switch**, то он завершает только непосредственно охватывающий его оператор.

Приклад. Скільки випадкових чисел потрібно згенерувати поки не випаде задане число k . З Edit1 зчитується значення k , у Edit2 виводиться результат.

```
int k = StrToInt(Edit1->Text);
int result = 0;
while(1){ // нескінчений цикл
    result++;
    if(k == random(100)) break; // розрив оператора while, якщо умова виконана
}

Edit2->Text = IntToStr(result);
```

Оператор продолжения `continue`;

Оператор продолжения передает управление на следующую итерацию в операторах цикла **while**, **do**, **for**. Он может содержаться только в теле этих операторов. В операторах **do** и **while** следующая итерация начинается с вычисления условного выражения. В операторе **for** следующая итерация начинается с вычисления выражения приращения, а затем происходит вычисление условного выражения.

Оператор продовження continue;

```
int k = StrToInt(Edit1->Text);
int max_step = 0;
int S1 = 0, S2 = 0;
do {
    max_step++;
    k = random(100);
    if(k > 50){
        S1 += k;
        continue;
    }
    if(k <= 50){
        S2 += k;
    }
} while(max_step < k);
```

```
Edit2->Text = IntToStr(S1);
```

```
Edit3->Text = IntToStr(S2);
```

Приклад. Підрахувати окремо суми випадкових $k=1000$ чисел, більших $k>50$ та менших $k\leq 50$.

З Edit1 зчитується значення k , у Edit2 та Edit3 виводяться суми.

Оператор повернення `return`;

return вираз ;

Оператора возврата заканчивает выполнение функции, в которой он содержится, и возвращает управление в вызывающую функцию. Управление передается в точку вызывающей функции, непосредственно следующую за оператором вызова. Значение *выражения*, если она задано, вычисляется, приводится к типу, объявленному для функции, содержащей оператор возврата, и возвращается в вызывающую функцию. Если *выражение* опущено, то возвращаемое функцией значение не определено.

Оператор повернення return;

//функція

```
double f(double r){  
    double S = 3.1415926538*r*r;  
    return S;  
}
```

```
void __fastcall TForm1::Button1Click(TObject  
*Sender)  
{  
    double r = StrToFloat(Edit1->Text);  
    Edit2->Text = FloatToStr( f(r) ); //виклик функції  
}
```

Приклад. Написати функцію, яка обчислює вираз $S = \pi r^2$, де r є аргументом функції. З Edit1 зчитується значення r , у Edit2 виводяться S .

Тема 5. Масиви.

Об'ява масиву, завдання типу елементів масиву та їх кількості. Операції з масивами. Робота з рядками.

Массивы

Массивы — это группа элементов одинакового типа (double, float, int и т.п.).

Из объявления массива компилятор должен получить информацию о типе элементов массива и их количестве.

Объявление массива имеет два формата:

1) спецификатор-типа описатель [константное_выражение];

Пример: `int M[10];`

2) спецификатор-типа описатель [];

Пример: `int M[];`

Описатель — это идентификатор массива.

Спецификатор-типа задает тип элементов объявляемого массива.

Элементами массива не могут быть функции и элементы типа void.

Константное-выражение в квадратных скобках задает количество элементов массива.

Константное-выражение при объявлении массива может быть опущено в следующих случаях:

- при объявлении массив инициализируется,
- массив объявлен как формальный параметр функции,
- массив объявлен как ссылка на массив, явно определенный в другом файле проекта.

Массивы

Приклады:

1) `int a[2][3];`

`/* представлено в виде матрицы`

`a[0][0] a[0][1] a[0][2]`

`a[1][0] a[1][1] a[1][2]`

`*/`

2) `double b[10]; /* вектор из 10 элементов имеющих тип double */`

3) `int w[3][3] = { { 2, 3, 4 },`
`{ 3, 4, 8 },`
`{ 1, 0, 9 } };`

Динамічні масиви

Приклад:

```
String S = "Hello C++";  
int len = S.Length();
```

```
char *M = new char [len];
```

```
for(int i = 0; i < len; i++){  
    M[i] = S.operator [](i+1);  
}
```

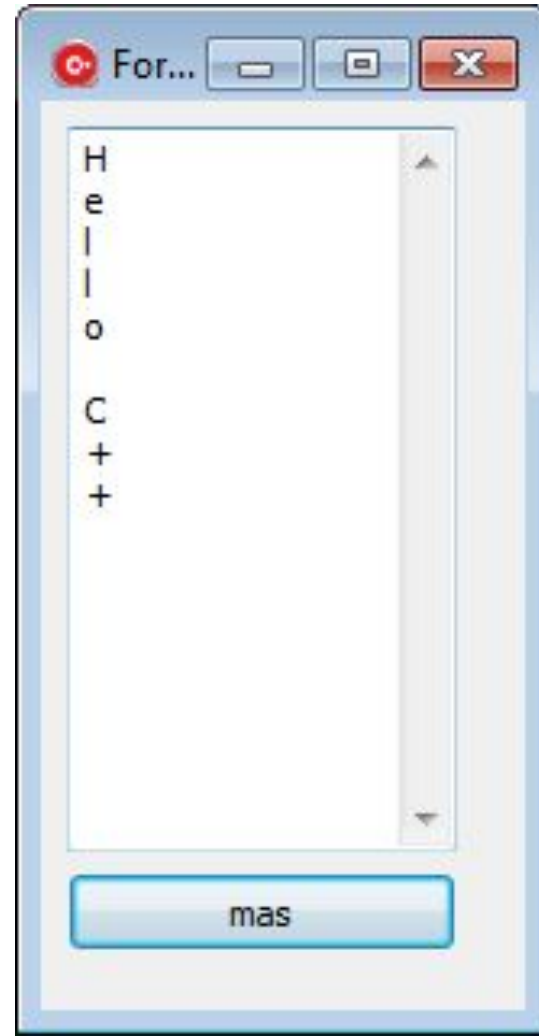
```
Memo1->Clear();
```

```
Memo1->Lines->Add(M[i]);
```

```
for(int i = 0; i < len; i++){  
    Memo1->Lines->Add(M[i]);  
}
```

```
delete[] M;
```

«Розібрати» строку «Hello C++» на символи та вивести символи в стовпчик у Мемо.



Динамічні масиви

Заповнити квадратну матрицю NxN випадковими числами, тільки головну діагональ заповнити одиницями.

```
String S = ""; // Объява змінної типу рядок
```

```
int n = StrToInt(Edit1->Text);
```

Вказівник на вказівник

```
int **M = new int* [n];
```

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

```
    M[i] = new int [n];
```

Можливе оголошення змінної, котра містить адресу іншої змінної, котра, у свою чергу, також є вказівником.

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

```
    for(int j = 0; j < n; j++){
```

```
        if(i == j) M[i][j] = 1;
```

```
        else M[i][j] = random(100);
```

```
    }
```

```
}
```

Динамічні масиви

Заповнити квадратну матрицю $N \times N$ випадковими числами, тільки головну діагональ заповнити одиницями.

```
Memo1->Clear();
```

```
for(int i = 0; i < n; i++){  
    S = "";  
    for(int j = 0; j < n; j++){  
        S += IntToStr(M[i][j])+" " ;  
    }  
    Memo1->Lines->Add(S);  
}  
delete[] M;
```

