

Процедурне програмування. Абстракція даних.

У лекції розглядаються елементи мови (ідентифікатори, ключові слова, коментарі). Дається поняття змінної, базових типів даних, операцій з даними.

Мета: Засвоєння основ мови програмування C++, типів даних, операцій з даними.

Лексика мови C++

Алфавіт мови це:

- Великі та малі латинські букви.
- Цифри.
- Спеціальні знаки: “ { } , [] # ! < = ? > ; & ` * / + - () % ~ : ^ _ .

При компіляції програми із символів алфавіту формуються лексичні елементи (лексеми) мови. Для виділення лексем використовуються пробільні роздільники (пробіли, символи табуляції, перехід на новий рядок). До лексем мови відносять:

- Ідентифікатори.
- Зарезервовані (ключові) слова.
- Константи.
- Знаки операцій.
- Знаки пунктуації (роздільники).

Ідентифікатори мови C++

Ідентифікатори — це імена, якими в програмі можна позначати всі елементи програми: змінні, константи, типи, функції і мітки.

Ідентифікатор — це послідовність латинських букв, цифр, символів підкреслення, причому *першою повинна бути буква чи символ підкреслення*.

Ідентифікатори можуть мати **довільне число символів**.

У C++ **розрізняються заголовні та малі літери**. Це значить, що компілятор C++ розглядає букви верхнього і нижнього регістрів як різні символи.

Наприклад, компілятор вважає змінні *Count*, *count* і *COUNT* (*Rate*, *rate*, *RATE*) трьома унікальними ідентифікаторами.

Зарезервовані слова (ключові слова)

Зарезервовані слова (ключові слова)— це визначені ідентифікатори, які мають особливе значення для компілятора C/C++, їх можна використовувати тільки відповідно до опису. Хоча згідно стандарту, програмні ідентифікатори можуть збігатися з ключовими словами, якщо їх визначити в іншому просторі імен. Зарезервовані слова наведені у таблиці.

| | | | | | | | |
|-------|----------|--------|--------|-----------|--------|----------|----------|
| asm | class | double | friend | new | return | switch | union |
| auto | const | else | goto | operator | short | template | unsigned |
| break | continue | enum | if | private | signed | this | virtual |
| case | default | extern | inline | protected | sizeof | throw | void |
| catch | delete | float | int | public | static | try | volatile |
| char | do | for | long | register | struct | typedef | while |

Якщо ви працюєте у стандартному просторі імен, ідентифікатор не може виглядати так само, як зарезервоване (ключове) слово мови.

Коментарі мови C++

Коментарі в мові C++ – це написана програмістом примітка, яка призначена для пояснення деяких аспектів коду програми. Компілятор ігнорує коментарі.

Коментар в один рядок довжиною починається із двох похилих //.

// Коментар довжиною в один рядок

Коментар довільної довжини починається із символів /* і закінчується символами */.

/* Коментар

довільної довжини !!!!!!!

****/***

Змінні мови C++

Для збереження даних програма має знати:

- де зберігаються данні, тобто ім'я для звернення до даних або адресу комірки пам'яті;
- який вид (тип) даних зберігається;
- яке значення там зберігається.

Змінна - це ділянка пам'яті, яка має ім'я і у якій зберігається значення певного типу, яке може бути зміненим у програмі.

Оголошення змінної

Всі змінні перед їх використанням мають бути оголошеними.

При кожному оголошенні визначається таке:

- *вид* (тип) даних, які мають зберігатися у змінній.
- *ім'я змінної*, яке є символічним представленням імені змінної, міткою області пам'яті.

У змінних можуть зберігатися числа, букви та інші символи. Число (або данні іншого типу), яке зберігається у змінній називають її **значенням**.

Вид даних, які зберігаються у змінній, називаються її типом. Загальна форма оголошення має такий вигляд:

Ім'я_типу список_змінних;

список змінних це:

ім'я_змінно1, ім'я_змінної2,... ім'я_змінноїK;

Типи даних

Під типом розуміємо множину значень, які може приймати змінна а також множину операцій, які можна виконувати над змінною.

Тип визначає формат внутрішнього представлення даних в пам'яті комп'ютера.

Тип змінної визначається при її оголошенні і не може бути зміненим в процесі виконання програми.

Змінна може використовуватися з допустимими для неї операціями.

В мові C++ існує дві групи даних: **базові (основні)** та **похідні**, які є похідними від основних типів даних.

Похідні можна поділити на:

Скалярні (вказівники, посилання, перелічення).

Структуровані (масиви, структури, класи, об'єднання).

Базові (основні) типи даних

Розглянемо базові типи даних:

1. Цілочисельний – **int**
2. Дійсний (із плаваючою точкою) - **float**
3. Дійсний (із плаваючою точкою) подвійної довжини - **double**
4. Символьний - **char, wchar_t**
5. Бульовий - **bool**
6. Невизначений - **void**

Тип `int`

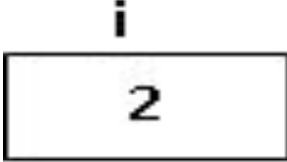
Тип `int` — цілі числа, що використовуються для рахунка. Можуть мати довжину 32 розряду (4 байти) і діапазон значень від

-2 147 483 648 до 2 147 483 647 ($2^{32}=4\ 294\ 967\ 295$).

Цілочисельний тип даних може мати специфікаторі.

- `signed`, `unsigned`
- `long`
- `short`

```
int i;  
i=2;
```



A diagram showing a rectangular box representing a memory cell. Above the box is the label 'i'. Inside the box is the number '2'.

```
unsigned int d1;
```

Тип `int`

Цілочисельні константи мови C++ можуть записуватися в трьох системах числення: **десятковій, восьмиричній, шістнадцятиричній**. Для визначення до якої системи числення відноситься числова константа в C++ використовується одна або дві перші цифри константи.

- Якщо перша цифра знаходиться у діапазоні 1 – 9, тоді число є десятковим, наприклад: 214 або 2567.
- Якщо перша цифра є 0, а друга знаходиться у діапазоні від 1 до 7, тоді число є восьмиричним (число 043 – це восьмиричне число еквівалентне десятковому числу 35).
- Якщо перші два символи є **0x** або **0X**, тоді це число є шістнадцятиричним, наприклад 0x43 – це число еквівалентне десятковому числу 67. В шістнадцятиричних числах символи a-f та A-F означають відповідні шістнадцятиричні цифри 10-15. Наприклад 0xF – це шістнадцятиричне число еквівалентне десятковому числу 15. 0xAF – це шістнадцятиричне число еквівалентне

Тип **char**

Тип **char**— призначений для представлення окремих символів, наприклад букв або цифр або розділових знаків. Основний набір символів може містити 256 символів, тобто мати довжину 8 розрядів чи 1 байт і приймати значення в діапазоні від 0 до 255 – це мінімально можливий діапазон ($2^8=256$).

wchar_t — призначений для представлення двобайтових символів. В мовах програмування для представлення символів використовується числовий код.

```
char c1;  c1='M';
```

```
wchar_t dob;  dob=L'!';
```

Тип `char`

Спеціальні символні константи. Деякі неграфічні символи, наприклад символ переходу на новий рядок, вимагають спеціального представлення. Іноді їх називають – ESC- або управляючими послідовностями. Такі символи укладаються в одинарні лапки і починаються із оберненої похилої лінії \. Наведемо деякі із цих символів:

- `'\n'` – перехід на новий рядок
- `'\0'` – нульовий символ
- `'\t'` – горизонтальна табуляція
- `'\v'` – вертикальна табуляція

Константа символний рядок. Символьний рядок – це послідовність символів, які розміщуються у подвійних лапках. Наприклад, **“Мова програмування C++”** - це символний рядок.

```
cout<<“Мова програмування C++”;
```

Тип **float, double**

В мові C++ є три типи дійсних даних з плаваючою точкою: **float, double, long double**. Використовується американський тип нотації при записі дробових чисел, тобто ціла частина відокремлюється від дробової точкою, наприклад **3.14159**. В комп'ютері ці числа зберігаються у вигляді двох складових частин. Одна частина являє собою деяке значення, а друга частина степінь цього значення.

Тип **float**— числа, що мають дробову частину, довжина 32 розряди, тобто **4 байти**, а діапазон від $3.4E-38$ до $3.4E+38$.

Тип **float, double**

Тип **double**— числа з плаваючою точкою подвійної довжини довжиною 64 розряди, тобто 8 байтів, а діапазон — від $1.7E-308$ до $1.7E+308$. Довгі числа подвійної точності long double можуть мати довжину 80 бітів чи 10 байт і діапазон — від $\pm 1.18E-4932$ до $1.18E+4932$.

Існує два способи запису таких чисел: **стандартний запис (5.8)** та **експоненціальна форма запису** ($7.563E6$ або $7.563e6$). Цей запис означає, що число 7.563 множиться на 1000000 ; $E6$ означає 10 в 6-му степені, при цьому 6 називають експонентою, а 7.563 — мантисою.

Тип **bool**

Тип `bool` мають змінні, які можуть приймати два значення: **true**(істина) **false**(хибність). Можна використовувати визначені літерали `true` та `false` для задавання бульового типу, наприклад,

```
bool isready= true;
```

Літерали `true` та `false` можуть бути перетворені в данні типу `int`, при цьому `true` перетворюється в 1, а `false` – в 0. Окрім цього, будь-яке ненульове значення перетворюється в значення `true`, а нульове – в значення `false`, наприклад,

```
bool start= -100;
```

```
bool stop= 0;
```


Модифікатор доступу до змінної **const**

Змінна, до якої в оголошенні застосований модифікатор `const`, не може змінювати своє значення, її можна тільки ініціалізувати, тобто визначити її значення на початку виконання програми. Наприклад, в оголошенні

```
const int k2=10;
```

створюється змінна з ім'ям "k2", причому їй присвоюється початкове значення 10, що надалі в програмі змінити не можна.

Довжина і діапазон значень основних (базових) типів даних

| Тип | Типова довжина | Мінімальний діапазон значень |
|---------------|----------------|-------------------------------------|
| char | 1 байт | від -120 до 127 |
| unsigned char | 1 байт | від 0 до 255 |
| signed char | 1 байт | від -120 до 127 |
| int | 4 байти | від -2 147 483 648 до 2 147 483 647 |
| short int | 2 байти | від -32 768 до 32 767 |
| long int | 4 байти | від -2 147 483 648 до 2 147 483 647 |
| unsigned int | 4 байти | від 0 до 4 294 967 295 |
| unsigned | 2 байти | від 0 до 65 535 |
| unsigned long | 4 байти | від 0 до 4 294 967 295 |
| float | 4 байти | від 3.4E-38 до 3.4E+38 |
| double | 8 байта | від 1.7E-308 до 1.7E+308 |
| long double | 10 байт | від 1.1E-4932 до 1.1E+4982 |

Глобальні та локальні змінні

Програма написана мовою програмування C++ являє собою набір функцій. Причому виконання програми починається з функції **main ()**. Оголошення змінної може бути розташоване в трьох місцях: **усередині функції**, при визначенні параметрів функції і **поза усіма функціями**. Це — місця оголошень відповідно локальних змінних, формальних параметрів функцій і глобальних змінних.

Оголошення_глобальних_змінних

```
void main ( )           // заголовок головної функції
{                       //початок блоку
    Оголошення_локальних_змінних;
} //закриття блоку
```

Змінні, оголошені усередині блоку (функцій), називаються **локальними змінними**.

Операції

Існує чотири основних класи операцій:

- ▣ *арифметичні,*
- ▣ *логічні,*
- ▣ *порівняння (відношення) та*
- ▣ *порозрядні(з бітами).*

Крім них, є також деякі спеціальні операції, наприклад, операція присвоювання.

Операція присвоювання

Аналогічно іншим операціям C++, результат операції присвоювання є деяким значенням, що теж можна присвоювати. Оператор присвоювання може бути присутнім у будь-якій виразі. Загальна форма оператора присвоювання:

```
ім'я_змінної = значення; // int s; s=45;
```

Значення може бути просто константою, функцією, змінною або виразом.

Вираз з операцією присвоювання можна використовувати в довгих виразах, подібних наступному:

```
int value1, value2;
```

```
value1 = 8*(value2=5);
```

Множинне присвоювання.

```
x = y = z = 0;
```

Арифметичні операції. Пріоритет операцій.


| Оператор | Операція |
|----------|---------------------------------|
| - | Віднімання, також унарний мінус |
| + | Додавання |
| * | Множення |
| / | Ділення |
| % | Залишок від ділення |
| -- | Декремент, чи зменшення на 1 |
| ++ | Інкремент, чи збільшення на 1 |

Найвищий ++ --

- (унарний мінус)

/ % *

Щонайнижчий + -



```
int a=3,b=10,c=0,d;
```

```
d=b+a;
```

```
d=b-a;
```

```
d=b*a;
```

```
d=b/a;
```

```
d=b%a;
```

```
d = a % b;
```

```
d = b % c;
```

```
a++; b--;
```

Операції відношення (порівняння) та логічні операції

| Операції відношення | |
|---------------------|-----------------------|
| > | Більше ніж |
| >= | Більше або дорівнює |
| < | Менше ніж |
| <= | Менше або дорівнює |
| != | Не дорівнює |
| == | Дорівнює (тотожність) |
| Логічні оператори | |
| && | AND (логічне І) |
| | OR (Або) |
| ! | NOT (Ні, заперечення) |

Необхідно пам'ятати, що результатом будь-якої операції порівняння чи логічної операції є false (0) чи true (1).

Найвищий !
> >= < <=
== !=
&&
Найнижчий ||

!(0 && 0) || 0 .

!0 && 0 || 0

Операції з бітами

| | |
|----|--------------------------|
| & | AND (І) |
| | OR (Або) |
| ^ | Виключаюче XOR (Або) |
| ~ | NOT (Ні) доповнення до 1 |
| >> | Зсув вправо |
| << | Зсув вліво |

Ці операції застосовуються до комірок пам'яті, в яких зберігаються данні типу `char` та `int`. Порозрядні операції виконуються над окремими розрядами (бітами) операндів.

Вирази

Вирази складаються з операцій, констант, функцій і змінних. У мові C++ виразом є будь-яка правильна послідовність цих елементів.

Більшість виразів у мові C++ за формою дуже схожі на алгебраїчні. Однак тут необхідно бути уважним і враховувати специфіку виразів та пріоритет операцій у мові C++.

Пріоритети операцій

| | |
|-----------|--|
| Найвищий | <code>() [] -></code> <code>! ~ ++ -- (type) * & sizeof</code> <code>* / %</code> <code>+ -</code> <code>>> <<</code> <code>< <= > >=</code> |
| | <code>== !=</code> <code>&</code> |
| | <code>^</code> <code> </code> <code>&&</code> <code> </code> |
| | <code>?:</code> <code>= += .= *= /=</code> |
| Найнижчий | <code>,</code> |

Перетворення типів

В мові C++ перетворення типів можуть виконуватися автоматично або явно задаватися програмістом. Перетворення типів виконується автоматично:

- Коли данні одного типу присвоюються змінній іншого типу.
- Коли вираз містить данні різних типів.

Перетворення типів при присвоюваннях

Якщо в операції зустрічаються змінні різних типів, відбувається *перетворення типів*. В операторі присвоювання діє просте правило: значення виразу з **правої частини** перетвориться до типу об'єкта **в лівій частині**. При цьому може бути втрачена частина даних. Наприклад, перетворення цілих у символи вилучає відповідні старші бінарні розряди.

Перетворення типів

Перетворення типів у виразах

Якщо у виразі зустрічаються змінні і константи різних типів, вони перетворюються до одного типу. Компілятор перетворить "менший" тип у "більший". Перерахуємо базові типи від „найменшого” до „найбільшого” : bool, char, signed char, unsigned char, short, unsigned short, int, unsigned int, long, unsigned long, float, double, long double. Цей процес називається *перетворенням типів (type promotion)*. Таким чином при обчисленні виразів типи даних bool, char, signed char, unsigned char, short, unsigned short перетворюються в тип int. При цьому значення true перетворюється в 1 а значення false – в 0.

Явне перетворення типів

Програміст може "примусово" перетворити значення виразу до потрібного йому типу, використовуючи **операцію приведення типів**. Загальна форма використання операції явного приведення типу:

(тип) вираз

Тут *тип* — це будь-який підтримуваний тип даних.

Наприклад,

```
int x=7;
```

```
cout<<(float) x/2; //Результат 3.5
```

Якщо б не застосовувалося операція приведення типів, тоді б виконувалося цілочисельне.

Операція приведення типу є унарною (тобто має один операнд) і має той же пріоритет, що й інші унарні операції.

Приклад програми мовою C++

```
#include <iostream>
using namespace std;
// оголошення глобальних змінних
void main ( )
{
//оголошення локальних змінних
Інструкція_1;
Інструкція_2;
...
Інструкція_N;
}
```

Схема простої програми мовою C++

```
#include <iostream>
using namespace std;
void main ( )
{
    setlocale(LC_ALL, "");
    int integer1, integer2, sum;    // оголошення
    cout << "Введіть перше ціле число ";    // запрошення
    cin>> integer1;    // введення цілого
    cout << " Введіть друге ціле число " << endl;    // запрошення
    cin>> integer2;    // введення цілого
    sum = integer1 + integer2;    // додавання
    cout << "Сума дорівнює " << sum << endl;    // виведення
        //результату
}
```