Лабораторна робота №1

Тема: Арифметичні та фундаментальні типи C++. Структури.

Мета: Навчитись працювати з арифметичними та фундаментальними типами мовою програмування C++.

Навчальні матеріали

Арифметичні типи

Стандартні типи даних часто отримують назву "арифметичними", оскільки їх можна використовувати в арифметичних операціях. Для опису основних типів визначено ключові слова: цілий, символьний, логічний, дійсний.

Цілий тип (ціле число) є одним з основних типів даних в багатьох програмних мовах, включаючи C++. Він призначений для представлення цілих чисел, тобто чисел без десяткової частини або експоненти. У мові програмування C++, цілий тип позначається ключовим словом int.

Дійсний тип (або тип з плаваючою комою) є одним з основних типів даних в багатьох програмних мовах, включаючи C++. Він призначений для представлення чисел з плаваючою точкою, тобто тих, що можуть мати десяткову частину або експоненту. У мові програмування C++, дійсний тип позначається ключовими словами float, double, та long double.

Логічний тип (булевий тип) є одним з основних типів даних в багатьох програмних мовах, включаючи C++. У мові програмування C++, логічний тип позначається ключовим словом bool. Логічний тип може мати два значення: true (істина) або false (неправда).

Символьний тип (char) є одним з основних типів даних в багатьох програмних мовах, включаючи C++. У мові програмування C++, символьний тип призначений для представлення символів і використовується для зберігання одного символу.

Операції порівняння є ще одним типом операцій над арифметичними типами. Це способи порівнювати два значення чи об'єкти для визначення, чи вони рівні, менші, більші чи в чому-то іншому схожі. Результатом операції порівняння є логічне значення, таке як істина (true) або неправда (false).

Фундаментальні типи

Фундаментальні типи в C++ вказує на базові типи даних, які не будуються з інших типів, а вже є визначеними в мові. Ці типи використовуються для представлення простих значень, таких як цілі числа, дійсні числа, символи та логічні значення.

Основні типи мови C++:

Категорія	Тип	Опис

Цілі числа	*char*	це цілочисельний тип, зазвичай містить члени кодування виконання (в Microsoft C++ це кодування ASCII).
		Компілятор C ++ обробляє змінні типу *char*, *signed char* і *unsigned char* як змінні різних типів. Змінні типу *char* підвищуються до *int*, якщо вони мають тип *signed char* за замовчуванням, і не використовується параметр компіляції */J*. У цьому випадку вони розглядаються як тип *unsigned char* і підвищуються до типу *int* без розширення знака.
	bool	це цілочисельний тип, який може мати одне з двох значень: true або false. Його розмір не визначений.
	short	*short int* (або просто *short*) - це цілочисельний тип, розмір якого більше або дорівнює розміру типу *char* і менше або дорівнює розміру типу *int*.
		Об'єкти типу *short* можуть оголошуватися як об'єкти типу *signed short* і *unsigned short*. *Signed short* - синонім *short*.
	int	це цілочисельний тип, розмір якого більше або дорівнює розміру типу *short int* і менше або дорівнює розміру типу *long*.
		Об'єкти типу *int* можуть бути оголошені в *unsigned int* або *signed int*. *Signed int* - синонім *int*.
	*__int***n**	Ціле значення заданої розмірності, де **n** - розмір цілого числа в бітах. Значення **n** може бути 8, 16, 32 або 64.
	long	Тип *long* (або *long int*) - це цілочисельний тип, який більше або дорівнює розміру типу *int*.
		Об'єкти типу *long* можуть оголошуватися як об'єкти типу *signed long* і *unsigned long*. *Signed long* - синонім *long*.
	long long	Більше, ніж беззнаковий *long*.
		Об'єкти типу *long long* можуть оголошуватися як об'єкти типу *signed long long* і *unsigned long long*. *Signed long long* - синонім *long long*.
Числа з плаваючою	*float*	це тип з плаваючою комою найменшого розміру.

комою		
	double	це тип з плаваючою комою, розмір якого більше або дорівнює розміру типу *float*, але менше або дорівнює розміру типу *long double*.
	long double*1*	це тип з плаваючою комою, розмір якого дорівнює розміру типу double.
Розширені символи	*__wchar_t*	Змінна *__wchar_t* позначає тип розширених символів або багатобайтних символів. За замовчуванням тип wchar_t є власним типом, але можна використовувати */Zc: wchar_t-*, щоб зробити *wchar_t* визначенням типу для *unsigned short*. Щоб вказати константу розширеного символьного типу, перед символьною або рядковою константою слід використовувати префікс L.
Інші	*void*	Вказує на відсутність значення

Розміри основних типів:

Тип	**Розмір**
bool	1 байт
char, unsigned char, signed char	1 байт
short, unsigned short	2 байта
int, unsigned int	4 байта
__int **n**	8, 16, 32, 64 або 128 біт в залежності від значення **n**. *__int* **n** відноситься тільки до систем Microsoft.
long, unsigned long	4 байта
float	4 байта
double	8 байт
long double1	8 байт
long long	Аналогічно параметру __int64.

Кожен студент має за допомогою С++ виконати 1 завдання по номеру у списку.

Завдання

1. Реалізуйте алгоритм відповідно до Вашого варіанту.
2. Опишіть логіку заповнення структури з клавіатури.
3. Виведіть значення структури на екран.

Варіанти завдань

1. Створіть програму, яка використовує цілі числа для розв'язання математичної задачі обчислення суми арифметичної прогресії.
2. Розробіть програму, що використовує дійсні числа для вирішення задачі обчислення площі кола за його радіусом.
3. Створіть програму, яка обчислює периметр трикутника за введеними значеннями довжин сторін.
4. Створіть програму, яка перевіряє чи є введений символ є літерою чи цифрою.
5. Створіть програму, яка обчислює площу та периметр прямокутника за введеними значеннями довжин сторін.
6. Створіть програму, яка обчислює площу трикутника за введеними значеннями довжин сторін a, b та c.
7. Створіть програму, яка за двома введеними числами визначає більше та виводить це число на екран.
8. Створіть програму, яка знаходить середнє геометричне двох чисел a і b.
9. Створіть програму, яка вирахує суму, різницю, добуток і частку від ділення двох введених чисел.
10. Створити програму, яка за кількістю введених секунд виведе кількість хвилин.

Перерахований тип

Перерахований тип (enumeration або просто enum) є способом визначення набору іменованих констант, які можуть бути використані для представлення набору можливих значень. Enum дозволяє створювати псевдоніми для цілих чисел, щоб зробити код більш читабельним і зрозумілим.

В мові C++ для створення перерахування використовується ключове слово *enum*. Загальна форма перерахування має наступний вигляд:

enum ім'я_перерахування {список_імен} список_змінних

Приклад оголошення:

```
enum Color {
   RED,
   GREEN,
```

```
    BLUE
};
```

Значення за замовчуванням:

При використанні такого перерахованого типу, елементам буде привласнюватися цілочисельні значення від 0 вгору. У прикладі вище RED отримає значення 0, GREEN - 1, BLUE - 2.

Задання значень:

Можна явно вказати значення для елементів перерахованого типу:

```
enum Color {
    RED = 5,
    GREEN = 10,
    BLUE = 15
};
```

Використання:

```
Color myColor = RED;
```

Зручність читання коду:

Використання перерахованих типів може покращити зрозумілість коду, особливо коли маємо справу з набором конкретних значень.

Перераховані типи допомагають уникнути використання незрозумілих чисел у коді, зробити його більш читабельним і підвищити його підтримуваність.

Структури C++

Структури в програмуванні - це конструкція, яка дозволяє об'єднати різні типи даних під однією назвою. Вони дозволяють групувати різні змінні під один загальний тип, що дозволяє легше та більш організовано працювати з даними.

Основні характеристики структур включають:

- Поля (члени): Структури мають поля або члени, які можуть бути різних типів даних, наприклад, цілі числа, дійсні числа, рядки, інші структури тощо.
- Групування: Структури дозволяють групувати пов'язані дані в один об'єкт. Це допомагає в структуризації та організації коду.

- Оголошення: Структури оголошуються за допомогою ключового слова struct в C++. Екземпляри структур можуть бути створені, і до їх полів можна звертатися за допомогою оператора крапки (.).
- Вкладені структури: Структури можуть містити інші структури, утворюючи так звані вкладені структури. Це дозволяє створювати складні структури даних.

Структури часто використовуються для моделювання об'єктів реального світу або групування пов'язаних даних. Вони є одним із засобів організації та структуризації програмного коду в мові програмування C++.

В мові програмування C++, структури - це конструкція, яка дозволяє об'єднувати різні типи даних під однією назвою. Структури дозволяють створювати власні типи даних, які можуть містити поля з різними типами даних. Ось основні аспекти використання структур в C++:

Оголошення структур:

```
// Оголошення структури
struct Person {
    // Поля структури
    std::string name;
    int age;
    double height;
};

// Створення екземпляра структури
Person person1;
```

Ініціалізація та доступ до полів структури:

```
// Ініціалізація полів структури
person1.name = "John Doe";
person1.age = 25;
person1.height = 1.75;
```

```
// Доступ до полів структури
std::cout << "Name: " << person1.name << "\n";
std::cout << "Age: " << person1.age << "\n";
std::cout << "Height: " << person1.height << " meters\n";
```

Структури в функціях:

```
// Передача структури в функцію за значенням
void printPerson(Person p) {
    std::cout << "Name: " << p.name << "\n";
    std::cout << "Age: " << p.age << "\n";
    std::cout << "Height: " << p.height << " meters\n";
}

int main() {
    // Створення екземпляра структури
    Person person1;
    person1.name = "John Doe";
    person1.age = 25;
    person1.height = 1.75;

    // Виклик функції та передача структури
    printPerson(person1);

    return 0;
}
```

Вкладені структури:

```
// Оголошення вкладеної структури
struct Address {
    std::string city;
    std::string street;
```

```
    int zipCode;
};

// Структура, яка містить вкладену структуру
struct Employee {
    std::string name;
    int employeeId;
    Address address;
};

int main() {
    // Створення екземпляра структури з вкладеною структурою
    Employee employee1;
    employee1.name = "Alice";
    employee1.employeeId = 1001;
    employee1.address.city = "New York";
    employee1.address.street = "Broadway";
    employee1.address.zipCode = 10001;

    return 0;
}
```

Завдання

4. Реалізуйте структуру відповідно до Вашого варіанту.
5. Опишіть логіку заповнення структури з клавіатури.
6. Виведіть значення структури на екран.

Варіанти завдань

1. Створіть структуру User, яка представляє користувача програми. Кожен користувач має ім'я (перерахований тип), вік (ціле число), та першу букву електронної адреси (символ).
2. Створіть структуру Film, яка представляє інформацію про фільм. Кожен фільм має назву (перерахований тип), режисера (перерахований тип), рік випуску (ціле число) та рейтинг (дробове число).

3. Створіть структуру Computer, яка містить дані про комп'ютер. Кожен комп'ютер описується маркою (перерахований тип), процесором (перерахований тип), об'ємом оперативної пам'яті (ціле число) та розміром екрану (дробове число).
4. Створіть структуру Store, що містить інформацію про магазин. Кожен магазин характеризується назвою (перерахований тип), розташуванням (перерахований тип), площею (ціле число) та кількістю працівників (ціле число).
5. Створіть структуру Country для представлення інформації про країну. Кожна країна має назву (перерахований тип), столицю (перерахований тип), населення (ціле число) та площу (дробове число).
6. Створіть структуру Album, яка містить дані про музичний альбом. Кожен альбом включає назву (перерахований тип), виконавця (перерахований тип), рік випуску (ціле число) та кількість треків (ціле число).
7. Створіть структуру ProgrammingLanguage, що містить інформацію про мову програмування. Кожна мова програмування описується назвою (перерахований тип), розробником (перерахований тип), роком створення (ціле число) та поточною версією (дробове число).
8. Створіть структуру Food, яка містить дані про їжу. Кожна страва описується назвою (перерахований тип), кухнею (перерахований тип), ціною (дробове число) та кількістю калорій (ціле число).
9. Створіть структуру PhoneInfo для представлення інформації про телефон. Кожен телефон має бренд (перерахований тип), модель (перерахований тип), рік випуску (ціле число) та розмір екрану (дробове число).
10. Створіть структуру Car, яка представляє інформацію про автомобіль. Кожен автомобіль має марку (перерахований тип), модель (перерахований тип),* рік випуску (ціле число) та об'єм двигуна в літрах (дробове число).

Контрольні запитання

1. Що таке алгоритм?
2. Які є способи вираження алгоритмів?
3. Які є властивості алгоритмів?
4. Як оцінюється складність алгоритмів?
5. Що таке фундаментальні типи C++?
6. Що таке структура даних?
7. Яке є призначення структури даних?
8. Назвіть деякі з можливих переваг використання ефективних структур даних.
9. Що таке перечислений тип?