ОБ’ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ

Лабораторна робота №1

Основні поняття ООП. Класи та об’єкти. Функції доступу. Вказівник this

Мета: ознайомитись з основними поняттями ООП. Вивчити поняття клас, об’єкт, сеттер, геттер та навчитись їх програмно реалізовувати мовою С++.

Теоретична частина

Клас - це конструкція в ООП, яка представляє собою шаблон або опис структури та поведінки об'єктів. Клас є абстракцією, яка групує дані (властивості) та функції (методи), що пов'язані між собою логічно. Він визначає набір атрибутів (члени класу), які представляють стан об'єктів даного класу, а також методи, які визначають операції, які можуть бути виконані над цими об'єктами.

Класи визначаються за допомогою ключового слова class та мають ім'я, за допомогою якого їх можна ідентифікувати. Клас може мати конструктори, які використовуються для ініціалізації об'єктів класу, а також може містити статичні поля та методи, які використовуються для спільного використання даних між усіма об'єктами даного класу.

Об'єкт - це конкретний екземпляр класу. Він створюється на основі класу шляхом процесу, який називається інстанціюванням або створенням об'єкта. Кожен об'єкт має свій власний набір значень властивостей, які відображають його стан, та може виконувати методи, що визначені в класі.

Об'єкти зберігають свій внутрішній стан у вигляді значень властивостей, які можуть бути унікальними для кожного об'єкта. Методи об'єктів представляють собою операції або функції, які можуть бути виконані над цими об'єктами. Об'єкти взаємодіють один з одним шляхом виклику методів та обміну даними.

Основна ідея ООП полягає в організації програми як набору взаємодіючих об'єктів, які співпрацюють для вирішення певних задач. Класи визначають загальну структуру та поведінку, а об'єкти є конкретними представниками цих класів, здатними зберігати стан та виконувати дії, визначені класом.

Наприклад:

```
#include <iostream>
using namespace std;

// Клас "Автомобіль"
class Car {
```

```
public:
    string brand;
    string model;
    int year;

    void startEngine() {
        cout << "Двигун автомобіля запущено!" << endl;
    }

    void stopEngine() {
        cout << "Двигун автомобіля зупинено!" << endl;
    }
};

int main() {
    // Створення об'єкта класу "Автомобіль"
    Car myCar;

    // Налаштування властивостей об'єкта
    myCar.brand = "BMW";
    myCar.model = "X5";
    myCar.year = 2022;

    // Виклик методів об'єкта
    cout << "Марка автомобіля: " << myCar.brand << endl;
    cout << "Модель автомобіля: " << myCar.model << endl;
    cout << "Рік випуску автомобіля: " << myCar.year << endl;

    myCar.startEngine(); // Запуск двигуна
    myCar.stopEngine(); // Зупинка двигуна

    return 0;
}
```

У цьому прикладі ми оголосили клас "Car", який представляє автомобіль. Він має властивості brand, model та year, а також методи startEngine() та stopEngine(), які відповідають за запуск та зупинку двигуна.

У функції main() ми створюємо об'єкт myCar класу Car. Потім ми налаштовуємо властивості об'єкта, використовуючи оператор крапки (.), і викликаємо методи об'єкта.

В результаті на консоль будуть виведені властивості автомобіля (марка, модель, рік випуску), а також повідомлення про запуск та зупинку двигуна.

Функції доступу (сеттери та геттери) є методами, використовуваними в об'єктно-орієнтованому програмуванні (ООП) для отримання (читання) та встановлення (запису) значень приватних членів класу. Ці методи надають контроль доступу до даних класу та допомагають забезпечити принцип

інкапсуляції, що означає, що дані класу знаходяться внутрішньо і мають обмежений доступ ззовні.

Геттери (getter methods) - це методи, які використовуються для отримання (читання) значень приватних членів класу. Вони зазвичай повертають значення цих членів і мають повернутий тип, що відповідає типу члена класу. Геттери дозволяють зовнішньому коду отримувати доступ до значень приватних даних класу без безпосереднього доступу до них. Вони можуть бути корисними для отримання значень для подальшого використання або виведення на екран.

Наприклад, у класі "Person" може бути приватний член name, і для отримання цього значення може бути створений геттер:

```
class Person {
private:
    string name;

public:
    // Геттер для отримання значення name
    string getName() {
        return name;
    }
};
```

Сеттери (setter methods) - це методи, які використовуються для встановлення (запису) значень приватних членів класу. Вони приймають параметр з новим значенням і встановлюють це значення для відповідного приватного члена класу. Сеттери дозволяють контролювати доступ до приватних даних класу і забезпечувати перевірку або обробку значень перед їх збереженням.

Продовжуючи приклад з класом "Person", може бути створений сеттер для встановлення нового значення для name:

```
class Person {
private:
    string name;

public:
    // Геттер для отримання значення name
    string getName() {
        return name;
    }

    // Сеттер для встановлення значення name
    void setName(string newName) {
        name = newName;
```

```
    }
};
```

За допомогою сеттера зовнішній код може встановити нове значення для name, і внутрішній код класу може здійснити додаткову перевірку або обробку цього значення перед збереженням.

Використання геттерів та сеттерів дозволяє забезпечити кращий контроль над доступом до приватних даних класу, зберігаючи їх недоступними безпосередньо ззовні класу і забезпечуючи правильність операцій з цими даними.

Ключове слово this в об'єктно-орієнтованому програмуванні (ООП) вказує на поточний об'єкт, з яким пов'язаний виклик методу або доступ до членів класу. Воно представляє вказівник на сам об'єкт, на якому викликається метод або з якого доступ до членів класу.

Ключове слово this може використовуватись в контексті класу для посилання на його власні члени (змінні та методи). Це особливо корисно, коли ім'я параметра методу або локальної змінної конфліктує з іменем члена класу. Використання this дозволяє уникнути неоднозначностей і вказати на конкретний член класу.

Наприклад, розглянемо клас "Person" з методом "setName", який встановлює значення приватного члена "name":

```
class Person {
private:
    string name;

public:
    void setName(string name) {
        this->name = name;
    }
};
```

У цьому прикладі параметр методу "setName" має таке ж ім'я, як і приватний член "name". Використання this->name дозволяє чітко вказати, що ми звертаємось саме до члена класу "name", а не до параметра методу.

Крім того, ключове слово this може бути корисним при передачі посилання на поточний об'єкт як аргумент у виклику методу або при створенні ланцюжка методів.

Загалом, this використовується для доступу до членів класу зсередини самого класу і допомагає уникнути конфліктів ім'ен, які можуть виникати між параметрами методів та членами класу.

Завдання

Варіант 1

Завдання 1

1. Напишіть клас "Book" для представлення книги. Клас повинен мати наступні властивості та функціональність:

Приватні поля класу:

- title (назва книги)
- author (автор книги)
- year (рік видання книги)

Публічні методи класу:

- Метод setTitle(), який дозволяє задати назву книги.
- Метод getTitle(), який повертає назву книги.
- Метод setAuthor(), який дозволяє задати автора книги.
- Метод getAuthor(), який повертає автора книги.
- Метод setYear(), який дозволяє задати рік видання книги.
- Метод getYear(), який повертає рік видання книги.

2. Створіть об'єкт класу "Book".

3. Задайте значення полів об'єкта за допомогою відповідних методів.

4. Виведіть інформацію про книгу на екран, використовуючи методи доступу до полів.

5. Реалізувати програму за допомогою роздільної компіляції.

У вашому рішенні можуть бути додаткові методи та поля, якщо ви вважаєте їх необхідними.

Завдання 2

Реалізувати вище наведену задачу за допомогою структурного програмування. У висновку описати різницю цих методів.

Варіант 2

Завдання 1

1. Напишіть клас "BankAccount" для керування банківським рахунком. Клас повинен мати наступні властивості та функціональність:

Приватні поля класу:

- balance (баланс рахунку, типу double)
- accountNumber (номер рахунку, типу string)
- ownerName (ім'я власника рахунку, типу string)

Публічні методи класу:

- Метод setBalance(), який дозволяє задати початковий баланс рахунку.
- Метод getBalance(), який повертає поточний баланс рахунку.
- Метод setAccountNumber(), який дозволяє задати номер рахунку.
- Метод getAccountNumber(), який повертає номер рахунку.
- Метод setOwnerName(), який дозволяє задати ім'я власника рахунку.
- Метод getOwnerName(), який повертає ім'я власника рахунку.
- Метод deposit(), який дозволяє здійснити поповнення рахунку на певну суму.
- Метод withdraw(), який дозволяє зняти гроші з рахунку на певну суму.

2. Створіть об'єкт класу "BankAccount".

3. Задайте значення полів об'єкта, включаючи баланс рахунку, номер рахунку та ім'я власника.

4. Використовуйте методи для отримання та зміни значень полів об'єкта.

5. Викличте методи для поповнення та зняття грошей з рахунку.

6. Виведіть інформацію про рахунок на екран.

7. Реалізувати програму за допомогою роздільної компіляції.

У вашому рішенні можуть бути додаткові методи та поля, якщо ви вважаєте їх необхідними.

Завдання 2

Реалізувати вище наведену задачу за допомогою структурного програмування. У висновку описати різницю цих методів.

Варіант 3

Завдання 1

1. Створіть клас "Person" (людина), який має наступні властивості та функціональність:

Приватні поля класу:

- name (ім'я людини)
- age (вік людини)
- address (адреса людини)

Публічні методи класу:

- Метод setName(), який дозволяє задати ім'я людини.
- Метод getName(), який повертає ім'я людини.
- Метод setAge(), який дозволяє задати вік людини.
- Метод getAge(), який повертає вік людини.
- Метод setAddress(), який дозволяє задати адресу людини.
- Метод getAddress(), який повертає адресу людини.

2. Створіть об'єкт класу "Person".

3. Задайте значення полів об'єкта за допомогою відповідних методів.

4. Виведіть інформацію про людину на екран, використовуючи методи для отримання значень полів.

5. Реалізувати програму за допомогою роздільної компіляції.

У вашому рішенні можуть бути додаткові методи та поля, якщо ви вважаєте їх необхідними.

Завдання 2

Реалізувати вище наведену задачу за допомогою структурного програмування. У висновку описати різницю цих методів.

Варіант 4

Завдання 1

1. Створіть клас "Employee" (співробітник), який має наступні властивості та функціональність:

Приватні поля класу:

- name (ім'я співробітника)
- id (ідентифікатор співробітника)
- salary (заробітна плата співробітника)

Публічні методи класу:

- Метод setName(), який дозволяє задати ім'я співробітника.
- Метод getName(), який повертає ім'я співробітника.
- Метод setId(), який дозволяє задати ідентифікатор співробітника.
- Метод getId(), який повертає ідентифікатор співробітника.
- Метод setSalary(), який дозволяє задати заробітну плату співробітника.
- Метод getSalary(), який повертає заробітну плату співробітника.

2. Створіть об'єкт класу "Employee".

3. Задайте значення полів об'єкта за допомогою відповідних методів.

4. Виведіть інформацію про співробітника на екран, використовуючи методи для отримання значень полів.

5. Реалізувати програму за допомогою роздільної компіляції.

У вашому рішенні можуть бути додаткові методи та поля, якщо ви вважаєте їх необхідними.

Завдання 2

Реалізувати вище наведену задачу за допомогою структурного програмування. У висновку описати різницю цих методів.

Варіант 5

Завдання 1

1. Створіть клас "Student" (студент), який має наступні властивості та функціональність:

Приватні поля класу:

- name (ім'я студента)
- age (вік студента)
- major (спеціальність студента)

Публічні методи класу:

- Метод setName(), який дозволяє задати ім'я студента.
- Метод getName(), який повертає ім'я студента.
- Метод setAge(), який дозволяє задати вік студента.
- Метод getAge(), який повертає вік студента.
- Метод setMajor(), який дозволяє задати спеціальність студента.
- Метод getMajor(), який повертає спеціальність студента.

2. Створіть об'єкт класу "Student".

3. Задайте значення полів об'єкта за допомогою відповідних методів.

4. Виведіть інформацію про студента на екран, використовуючи методи для отримання значень полів.

5. Реалізувати програму за допомогою роздільної компіляції.

У вашому рішенні можуть бути додаткові методи та поля, якщо ви вважаєте їх необхідними.

Завдання 2

Реалізувати вище наведену задачу за допомогою структурного програмування. У висновку описати різницю цих методів.

Варіант 6

Завдання 1

1. Створіть клас "Animal" (тварина), який має наступні властивості та функціональність:

Приватні поля класу:

- name (ім'я тварини)
- species (вид тварини)
- age (вік тварини)

Публічні методи класу:

- Метод setName(), який дозволяє задати ім'я тварини.
- Метод getName(), який повертає ім'я тварини.

- Метод setSpecies(), який дозволяє задати вид тварини.
- Метод getSpecies(), який повертає вид тварини.
- Метод setAge(), який дозволяє задати вік тварини.
- Метод getAge(), який повертає вік тварини.

2. Створіть об'єкт класу "Animal".

3. Використайте метод setName() для задання імені тварини.

4. Використайте метод getName() для отримання імені тварини та виведіть його на екран.

5. Використайте метод setSpecies() для задання виду тварини.

6. Використайте метод getSpecies() для отримання виду тварини та виведіть його на екран.

7. Використайте метод setAge() для задання віку тварини. Використайте метод getAge() для отримання віку тварини та виведіть його на екран.

8. Реалізувати програму за допомогою роздільної компіляції.

У вашому рішенні можуть бути додаткові методи та поля, якщо ви вважаєте їх необхідними.

Завдання 2

Реалізувати вище наведену задачу за допомогою структурного програмування. У висновку описати різницю цих методів.

Варіант 7

Завдання 1

1. Створіть клас "Movie" (фільм), який має наступні властивості та функціональність:

Приватні поля класу:

- title (назва фільму)
- director (режисер фільму)
- year (рік випуску фільму)
- duration (тривалість фільму в хвилинах)

Публічні методи класу:

- Метод setTitle(), який дозволяє задати назву фільму.
- Метод getTitle(), який повертає назву фільму.
- Метод setDirector(), який дозволяє задати режисера фільму.
- Метод getDirector(), який повертає режисера фільму.
- Метод setYear(), який дозволяє задати рік випуску фільму.
- Метод getYear(), який повертає рік випуску фільму.
- Метод setDuration(), який дозволяє задати тривалість фільму.

- Метод getDuration(), який повертає тривалість фільму.

2. Створіть об'єкт класу "Movie".

3. Використайте метод setTitle() для задання назви фільму.

4. Використайте метод getTitle() для отримання назви фільму та виведіть її на екран.

5. Використайте метод setDirector() для задання режисера фільму.

6. Використайте метод getDirector() для отримання режисера фільму та виведіть його на екран.

7. Використайте метод setYear() для задання року випуску фільму.

8. Використайте метод getYear() для отримання року випуску фільму та виведіть його на екран.

9. Використайте метод setDuration() для задання тривалості фільму.

10. Використайте метод getDuration() для отримання тривалості фільму та виведіть її на екран.

11. Реалізувати програму за допомогою роздільної компіляції.

У вашому рішенні можуть бути додаткові методи та поля, якщо ви вважаєте їх необхідними.

Завдання 2

Реалізувати вище наведену задачу за допомогою структурного програмування. У висновку описати різницю цих методів.

Варіант 8

Завдання 1

1. Створіть клас "TravelDestination" (туристичний напрямок), який має наступні властивості та функціональність:

Приватні поля класу:

- name (назва туристичного напрямку)
- country (країна, в якій знаходиться напрямок)
- rating (рейтинг популярності напрямку)

Публічні методи класу:

- Метод setName(), який дозволяє задати назву туристичного напрямку.
- Метод getName(), який повертає назву туристичного напрямку.
- Метод setCountry(), який дозволяє задати країну, в якій знаходиться напрямок.
- Метод getCountry(), який повертає країну, в якій знаходиться напрямок.
- Метод setRating(), який дозволяє задати рейтинг популярності напрямку.
- Метод getRating(), який повертає рейтинг популярності напрямку.

2. Створіть об'єкт класу "TravelDestination".

3. Задайте значення полів об'єкту за допомогою відповідних методів.

4. Виведіть інформацію про туристичний напрямок на екран, використовуючи методи getName(), getCountry() та getRating().

5. Змініть значення рейтингу популярності напрямку за допомогою методу setRating().

6. Виведіть оновлену інформацію про туристичний напрямок на екран, використовуючи знову методи getName(), getCountry() та getRating().

7. Реалізувати програму за допомогою роздільної компіляції.

У вашому рішенні можуть бути додаткові методи та поля, якщо ви вважаєте їх необхідними.

Завдання 2

Реалізувати вище наведену задачу за допомогою структурного програмування. У висновку описати різницю цих методів.

Варіант 9

Завдання 1

1. Створіть клас "Country" (країна), який має наступні властивості та функціональність:

Приватні поля класу:

- name (назва країни)
- capital (столиця країни)
- population (населення країни)

Публічні методи класу:

- Метод setName(), який дозволяє задати назву країни.
- Метод getName(), який повертає назву країни.
- Метод setCapital(), який дозволяє задати столицю країни.
- Метод getCapital(), який повертає столицю країни.
- Метод setPopulation(), який дозволяє задати населення країни.
- Метод getPopulation(), який повертає населення країни.

2. Створіть об'єкт класу "Country".

3. Використайте метод setName() для задання назви країни.

4. Використайте метод getName() для отримання назви країни та виведіть її на екран.

5. Використайте метод setCapital() для задання столиці країни.

6. Використайте метод getCapital() для отримання столиці країни та виведіть її на екран.

7. Використайте метод setPopulation() для задання населення країни.

8. Використайте метод getPopulation() для отримання населення країни та виведіть його на екран.

9. Реалізувати програму за допомогою роздільної компіляції.

У вашому рішенні можуть бути додаткові методи та поля, якщо ви вважаєте їх необхідними.

Завдання 2

Реалізувати вище наведену задачу за допомогою структурного програмування. У висновку описати різницю цих методів.

Варіант 10

Завдання 1

1. Створіть клас "Company" (компанія), який має наступні властивості та функціональність:

Приватні поля класу:

- name (назва компанії)
- revenue (прибуток компанії)
- expenses (витрати компанії)

Публічні методи класу:

- Метод setName(), який дозволяє задати назву компанії.
- Метод getName(), який повертає назву компанії.
- Метод setRevenue(), який дозволяє задати прибуток компанії.
- Метод getRevenue(), який повертає прибуток компанії.
- Метод setExpenses(), який дозволяє задати витрати компанії.
- Метод getExpenses(), який повертає витрати компанії.
- Метод calculateProfit(), який обчислює прибуток компанії (прибуток = прибуток - витрати).
- Метод getProfit(), який повертає прибуток компанії.

2. Створіть об'єкт класу "Company".

3. Використайте метод setName() для задання назви компанії.

4. Використайте метод getName() для отримання назви компанії та виведіть її на екран.

5. Використайте метод setRevenue() для задання прибутку компанії.

6. Використайте метод getRevenue() для отримання прибутку компанії та виведіть його на екран.

7. Використайте метод setExpenses() для задання витрат компанії.

8. Використайте метод getExpenses() для отримання витрат компанії та виведіть їх на екран.

9. Використайте метод calculateProfit() для обчислення прибутку компанії та виведіть його на екран.

10. Використайте метод getProfit() для отримання прибутку компанії та виведіть його на екран.

11. Реалізувати програму за допомогою роздільної компіляції.

У вашому рішенні можуть бути додаткові методи та поля, якщо ви вважаєте їх необхідними.

Завдання 2

Реалізувати вище наведену задачу за допомогою структурного програмування. У висновку описати різницю цих методів.