МІНІСТЕРСТВО ОСВІТИ І НАУКИ УКРАЇНИ
Київський національний університет будівництва і архітектури

ОБ’ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ

Методичні вказівки
до виконання лабораторних робіт
для здобувачів першого (бакалаврського) рівня вищої освіти
за спеціальністю 174 «Автоматизація, комп’ютерно-інтегровані технології
та робототехніка»

Київ 2025

УДК 681.3.06
О-13

Укладач О.В. Бондарчук, доцент

Рецензент В.Ю. Луценко, канд. техн. наук, доцент

Відповідальний за випуск А.В. Запривода, канд. техн. наук, доцент

Затверджено на засіданні кафедри автоматизації технологічних процесів, протокол №5 від 2 грудня 2024 року.

В авторській редакції.

Об’єктно-орієнтоване програмування : методичні вказівки до
О-13 виконання лабораторних робіт / уклад. О.В. Бондарчук. – Київ : КНУБА, 2025. – 32 с.

Розглянуто загальні методичні вказівки до проведення лабораторних та практичних робіт в середовищі системи проєктування AutoCAD.

Призначено для здобувачів першого (бакалаврського) рівня вищої освіти за спеціальністю 174 «Автоматизація, комп’ютерно-інтегровані технології та робототехніка» для набуття практичних навичок під час проведення лабораторних та практичних робіт в сфері об’єктно-орієнтованого програмування.

ЗМІСТ

ЗАГАЛЬНІ ПОЛОЖЕННЯ

Метою виконання лабораторних та практичних робіт є закріплення набутих під час вивчення дисципліни «Об'єктно-орієнтоване програмування» теоретичних знань студентами.

Завдання виконання лабораторних та практичних робіт полягає в закріпленні основних теоретичних питань, які стосуються головних принципів об'єктно-орієнтованого програмування в програмному середовищі Visual Studio, а також у створенні студентами програм на мові програмування С++.

Мову С++ можна віднести до класу мов програмування проміжного (або середнього) рівня. Це означає, що застосування мови С++ дозволяє створювати як високорівневі програми, так і низькорівневі бібліотеки, які забезпечують взаємодію з апаратними засобами ЕОМ. Для багатьох програмістів мова С++ є оптимальною комбінацію, оскільки, будучи мовою високого рівня, дозволяє створювати складні програми, забезпечуючи, при цьому, можливість досягнення високої продуктивності саме за рахунок контролю над використанням ресурсів та їх доступністю.

1. **Загальні методичні вказівки з проведення лабораторних та практичних робіт**

На кожне лабораторне або практичне заняття студенти повинні приходити з повністю оформленим та підготовленим до здачі викладачеві звітом попередньої роботи, а також бути готовими до проведення наступної лабораторної або практичної роботи. Одержаний результат роботи необхідно показати викладачу.

В кожній лабораторній або практичній роботі наведені приклади складання програм для задач аналогічного типу.

Звіти лабораторних та практичних робіт повинні виконуватись кожним студентом самостійно. До звіту входять: титульний аркуш, мета роботи, короткі теоретичні відомості, код програми, результати роботи програми, висновок (пояснення до програми).

Лабораторна робота №1

***Тема**: «Класи і об'єкти»*

Мета роботи: навчитися розробляти програми, які містять класи та об'єкти.

Короткі теоретичні відомості

Класом називається опис деякої структури програми, яка має набір внутрішніх змінних – властивостей та функцій, які мають доступ до властивостей – методів. Процес об'єднання змінних та методів, в результаті якого отримуємо клас, називається *інкапсуляцією*.

Проте клас – це тільки опис, аналогічний опису типу даних, і він не доступний для прямого використання у програмі. Для отримання доступу до властивостей і методів класу необхідно створити екземпляр класу, який називається об'єктом. До одного класу може належати одночасно декілька об'єктів, кожний з яких має унікальне ім'я.

Об'явлення класу має наступний вигляд:

class ім'я класу {

закриті поля і методи класу

public:

відкриті поля і методи класу

} список об'єктів;

До закритих членів класу можна звертатися тільки всередині класу. Відкриті члени класу доступні також за його межами.

Приклад розробки програми

1. ***Умова:***

Написати програму для обчислення експоненти . Необхідно створити спеціальний клас з відповідним методом. Аргумент **x** і верхня межа ряду визначаються як поля класу. Розглянути випадок, коли поле, яке визначає межу ряду, є статичним.

2. ***Код програми:***

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

class Exp {
public:
   double x;
   static int N;
   int f(int n) {
      int res = 1;
      for (int i = 1; i <= n; ++i) {
         res *= i;
      }
      return res;
   }
   double exp() {
      double sum = 0;
      double term = x;
      for (int i = 0; i <= N; i++) {
         sum += term / f(i);
         term *= x;
      }
      return sum;
   }
};
int Exp::N = 10;
int main() {
   Exp a;
   cout << "enter x = ";
```

```
    cin >> a.x;
    cout << "exp(" << a.x << ") = " << a.exp() << endl;
    return 0;
}
```

3. ***Результат роботи програми:***

```
enter x = 2
exp(2) = 14.778


...Program finished with exit code 0
Press ENTER to exit console.
```

4. ***Пояснення до програми:***

Створюється спеціальний клас Exp, який має два поля, одне з яких статичне (N). У програмі створюється метод exp(), у якому обчислюється заданий вираз. У головному коді програми з клавіатури значення поле x та створюються об'єкт a, за допомогою якого відбувається звернення до методів класу. Результат роботи методу виводиться на екран.

Лабораторна робота №2

Тема: *«Конструктори і деструктори»*

Мета роботи: проаналізувати особливості використання конструкторів та деструкторів в ООП.

Короткі теоретичні відомості

Конструктор – метод, який автоматично викликається під час створення об’єкта. Деструктор – це метод, який викликається автоматично під час вилучення об’єкта з пам’яті.

Створення і перевантаження конструктора

Конструктор в класі створюється практично так само, як і звичайні методи, але з урахуванням трьох принципових особливостей:

1) ім’я конструктора співпадає з ім’ям класу;
2) конструктор не повертає результат і для нього тип результату не вказується взагалі;

3) метод необхідно викликати у явному вигляді, а конструктор викликається автоматично і тільки під час створення об’єкта.

Конструктор може мати аргументи, а може і не мати. Конструктор можна перевантажувати.

Деструктори, як і конструктори, можна явно описувати під час створення класу.

Правила створення деструктора:

1) ім’я деструктора співпадає з ім’ям класу, але перед ім’ям деструктора вказується символ «тільда»~;
2) тип результату для деструктора не вказується;
3) у деструктора немає аргументів.

Приклад розробки програми

1. ***Умова:***

   Написати програму для обчислення ряду .

2. ***Код програми:***

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

class Sine {
private:
   double x;
   int terms;
public:
   Sine(double _x, int _terms) : x(_x), terms(_terms) {
      cout << "Конструктор викликано. Параметри: x = " << x << ", кількість членів = " << terms << endl;
   }
   long long factorial(int n) {
      long long result = 1;
      for (int i = 2; i <= n; i++) {
         result *= i;
      }
      return result;
   }
   double calculateSin() {
      double sum = 0.0;
      for (int n = 0; n < terms; ++n) {
         double numerator = pow(x, 2 * n + 1);
```

```
            long long denom = factorial(2 * n + 1);
            sum += numerator / denom;
        }
        return sum;
    }
    ~Sine() {
        cout << "Деструктор викликано. Об'єкт видалено.\n";
    }
};
int main() {
    setlocale(LC_ALL, "");
    double x;
    int terms;
    cout << "Введіть значення x: ";
    cin >> x;
    cout << "Введіть кількість членів ряду для обчислення: ";
    cin >> terms;
    Sine sinhCalculator(x, terms);
    double result = sinhCalculator.calculateSin();
    cout << "Значення sh(" << x << ") = " << result << endl;
    return 0;
}
```

3. ***Результат роботи програми:***

```
Введіть значення x:4
Введіть кількість ?ленів ряду для обчислення:5
Конструктор викл?кано. Параметри: x = 4, кількість членів = 5
Значення sh(4) = 27.1732
Деструктор викликано. Об'єкт видалено.
```

4. ***Пояснення до програми:***

В програмі створюється клас **Sine,** в якому є закриті поля класу, відкритий метод factorial(), а також метод calculateSin(), в якому обчислюється даний вираз. В класі також описані конструктор і деструктор.

Лабораторна робота №3

Тема: *«Перевизначення операторних функцій»*

Мета роботи: проаналізувати використання перевизначення зовнішніх і внутрішніх операторних функцій в ООП.

Короткі теоретичні відомості

Перевизначення операторів здійснюється шляхом створення операторних функцій. Головна відмінність операторних функцій від звичайних в тому, що формально назвою операторної функції є перевизначений оператор. Важливе значення має належність операторної функції певному класу (для роботи з об'єктами якого перевизначається відповідний оператор) або вона є зовнішньою функцією.

Зовнішня операторна функція

Загальний вигляд:

тип _результату **operator** оператор (аргументи: тип і назва)

{ код програми}

тип _результату повертається операторною функцією;

оператор перевизначається операторною функцією;

operator – ключове слово;

Це може бути, наприклад, оператор +,-,*,/.

Аргументами операторної функції є операнди.

Приклад розробки програми

1. ***Умова*:**

Написати програму з класами для реалізації комплексних чисел в алгебраїчній та тригонометричній формах. В цьому випадку комплексне число z=x+yi визначається модулем і аргументом φ (, і може бути представлене у вигляді z=r. Шляхом перевантаження операторів зробити можливим піднесення комплексного числа в алгебраїчній формі у цілочисельну степінь. Результат – комплексне число у тригонометричній формі.

2. ***Код програми:***

```
#include <iostream>
#include <cmath>
using namespace std;
class MyClass
{
public:
	double x;
	double y;
	double otv;
		MyClass(double x, double y)
	{
		this->x = x;
		this->y = y;
	}
	void calc_for_xy()
	{
		double modul_z;
		modul_z = sqrt(x * x + y * y);
		cout << "Trigonometric form: Z = " << modul_z << "*(cos(" << x /
modul_z << ") + i*(sin(" << y / modul_z << "))" << endl;
	}
	double operator^(int a)
	{
		double modul_z;
		double ch;
		modul_z = sqrt(this->x * this->x + this->y * this->y);
		ch = modul_z * (cos(this->x / modul_z) + sin(this->y / modul_z));
		return pow(ch, a);
	}

};
int main()
{
	string str;
	double x;
	double y;
	cout << " x: ";
	cin >> x;
	cout << " y: ";
	cin >> y;
	MyClass(x, y).calc_for_xy();
	MyClass obj(x, y);
	double tnp = obj ^ 2;
	cout << "Number pow 2: " << tnp << endl;
```

}

3. ***Результат роботи програми:***

```
 x:5
 y:7
Trigonometric form: Z = 8.60233*(cos(0.581238) + i*(sin(0.813733))
Number pow 2: 180.696


...Program finished with exit code 0
Press ENTER to exit console.
```

4. ***Пояснення до програми:***

У програмі створюється клас MyClass. Викликається метод calc_for_xy, у якому відбувається обчислення модуля комплексного числа. Далі відбувається перевантаження оператора operator^(int a) та піднесення комплексного числа в алгебраїчній формі у цілочисельну степінь. В головному коді програми створюються об’єкти, за допомогою яких відбувається звернення до методів класу, щоб отримати результат.

Лабораторна робота №4

***Тема**: «Чисто віртуальні функції»*

Мета роботи: проаналізувати призначення чисто віртуальних функцій в ООП.

Короткі теоретичні відомості

Чисто віртуальною функцією називається така віртуальна функція, яка не має визначення в базовому класі.

virtual тип результату ім'я функції (аргументи)=0;

Клас, який містить хоча б одну чисто віртуальну функцію, називається абстрактним.

Приклад розробки програми

1. ***Умова:***

На основі існуючого прикладу написати програму з використанням чисто віртуальних функцій (обчислити периметр геометричних фігур).

2. ***Код програми:***

```
#include <iostream>
#include <cmath>
using namespace std;
class Figure {
public:
   double a;
   double b;
   double c;
   virtual double perimeter() = 0;
};
class Square : public Figure {
public:
   Square(double side) { a = side; }
   double perimeter() {
      return 4 * a;
   }
};
class Rectangle : public Figure {
public:
   Rectangle(double sideA, double sideB) {
      a = sideA;
```

```
        b = sideB;
    }
    double perimeter() {
        return 2 * (a + b);
    }
};
class Triangle : public Figure {
public:
    Triangle(double sideA, double sideB, double sideC) {
        a = sideA;
        b = sideB;
        c = sideC;
    }
    double perimeter() {
        return a + b + c;
    }
};
int main() {
    setlocale(LC_ALL, "");
    Square square(6);
    Rectangle rectangle(5, 6);
    Triangle triangle(3, 4, 5);
    cout << "Периметр квадрата: " << square.perimeter() << endl;
    cout << "Периметр прямокутника: " << rectangle.perimeter() << endl;
    cout << "Периметр трикутника: " << triangle.perimeter() << endl;
    return 0;
}
```

3. ***Результат роботи програми:***

```
Периметр квадрата: 24
Периметр прямокутника: 22
Периметр трикутника: 12


...Program finished with exit code 0
Press ENTER to exit console.
```

4. ***Пояснення до програми:***

В програмі створюється абстрактний клас **Figure** з числовим полями a, b, c і чисто віртуальною функцією **perimeter()**для обчислення периметра фігури. В кожному похідному класі функція перевизначається.

Лабораторна робота №5

Тема: *«Узагальнені функції та класи»*

Мета роботи: проаналізувати використання шаблонів в програмі та зробити відповідні висновки.

Короткі теоретичні відомості

Узагальнена функція – це функція, в яку тип даних, з якими вона працює, передається у вигляді параметру.

Загальний вигляд:

template <class тип> тип результату ім'я функції (аргументи) { код функції }

В C++ можна створювати класи, в яких тип даних передається як формальний параметр. Такі класи називають узагальненими, або просто шаблонами.

Приклад розробки програми

1. ***Умова задачі:***

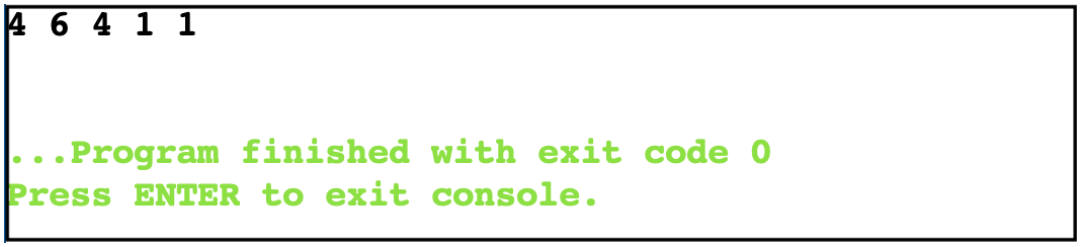
Написати програму з узагальненою функцією і аргументом – масивом. В результаті виконання функції на екран виводяться елементи масиву.

2. ***Код програми:***

```
#include <iostream>
#include <cstdlib>
#include <ctime>
using namespace std;
template <class X, int n> class MyClass {
public:
	X xarray[n];
	MyClass() {
		srand(time(0));
		for (int i = 0; i < n; i++) {
			xarray[i] = (X)(rand() % 10);
		}
	}
	void getx() {
		for (int i = 0; i < n; i++)
```

```
                cout << xarray[i] << " ";
            cout << endl;}
};
int main() {
    const int n = 5;
    MyClass <int, n>a;
    a.getx();
    return 0;
}
```

3. ***Результат роботи програми:***



4. ***Пояснення до програми:***

У даній програмі створюється узагальнена функція з аргументом – масивом template <class X, int n> class MyClass. Далі створюється поле класу – масив, викликається конструктор MyClass, у якому масив заповнюється випадковим чином. Викликається метод getx для відображення масиву. У головному коді програми вказується розмір масиву-поля класу та створюються об’єкти, за допомогою яких відбувається звернення до методів класу, щоб отримати результат.

Практична робота №1

***Тема**: «Створення даних типу «клас»»*

Мета роботи: здобути початкові практичні навички під час розробки програм в сфері ООП.

Приклад розробки програми

1. ***Умова:***

Завдання: створити клас з (стовпець 2) полями і методами:

1) конструктор за замовчуванням;
2) конструктор перевантаження з параметрами;
3) деструктор;
4) функції обробки даних (стовпці 3,4).

	Дата – день, місяць, рік	Збільшити рік на 1	Зменшити дату на 2 дні

2. ***Код програми:***

```
#include <iostream>
using namespace std;
class Date {
private:
   int day;
   int month;
   int year;
public:
   Date() {
      day = 1;
      month = 10;
      year = 2024;
   }
   Date(int d, int m, int y) {
      day = d;
      month = m;
      year = y;
   }
   ~Date() {};

   void Year() {
      year += 1;
   }
```

```
    void Days() {
        day -= 2;
        if (day < 1) {
            month -= 1;
            if (month < 1) {
                month = 12;
                year -= 1;
            }
            if (month == 2) {
                if ((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0))
                    day = 29;
                else
                    day = 28;
            }
            else if (month == 4||month == 9 ||month == 6|| month == 11) {
                day = 30;
            }
            else {
                day = 31;
            }
        }
    }
    void print() {
        cout << "Дата: " << day << "/" << month << "/" << year << endl;
    }
};
int main() {
    setlocale(LC_ALL, "");
    Date date1;
    date1.print();
    date1.Year();
    cout << "Після додавання одного року: ";
    date1.print();
    date1.Days();
    cout << "Після зменшення дати на 2 дні: ";
    date1.print();
    return 0;
     }
```

3. ***Результат роботи програми:***

```
Дата: 1/10/2024
Після додавання одного року: Дата: 1/10/2025
Після зменшення дати на 2 дні: Дата: 30/9/2025


...Program finished with exit code 0
Press ENTER to exit console.
```

4. ***Пояснення до програми:***

Створюється клас **Date**, який має три поля **day, month, year**.

У класі є конструктори з аргументами і без аргументів, а також деструктор. Створюються три методи **Year(), Days()** та **print**. У головному коді програми створюються об'єкт **date1**, за допомогою якого відбувається звернення до методів класу. Результат роботи методів виводиться на екран.

Завдання до самостійної роботи

№	Клас і його поля	Функція – метод (1) обробки даних	Функція – метод (2) обробки даних
1	Правильний дріб: чисельник, знаменник	Перетворити значення дробу на відсотки	Знайти суму цифр значення знаменника
2	Книга – назва, автор, рік видання	Обчислити, скільки років книзі	Кількість днів, які пройшли після року видання книги
3	Книга – назва, кількість сторінок, вартість	Обчислити середню вартість однієї сторінки	Збільшити вартість книги у два рази, якщо кількість сторінок більше, ніж 100
4	Робітник – прізвище, оклад, рік народження	Обчислити вік робітника	Визначити кількість календарних днів до виповнення робітнику 50 років
5	Дата – день, місяць, рік	Визначити, чи є рік високосним (кратність 4)	Збільшити дату на 5 днів
6	Дата – день, місяць, рік	Збільшити рік на 1	Зменшити дату на 2 дні
7	Робітник – прізвище, оклад, рік вступу на роботу	Обчислити стаж роботи робітника	Визначити кількість днів, що пройшли після року вступу на роботу

Практична робота №2

***Тема**: «Успадкування класів»*

Мета роботи: проаналізувати один із чотирьох принципів ООП – успадкування і здобути практичні навички під час розробки програм.

Короткі теоретичні відомості

В ООП можна створювати нові класи на основі вже наявних. Новий клас називається похідним, а наявний – базовим, або батьківським. Під час успадкування похідний клас отримує всі атрибути та методи базового і може додавати власні. Успадкування дає можливість повторно використовувати код та створювати ієрархії класів «від загального до приватного» для реалізації складних схем. Це сприяє підвищенню ефективності розробки та забезпечує більш логічну організацію коду. Не потрібно постійно переписувати однакові властивості для різних об'єктів, достатньо успадкувати їх від одного «батька».

Приклад розробки програми

1. ***Умова:***

Завдання: створити похідний клас з додатковим полем і реалізувати в ньому методи:

1) конструктор;
2) функція обробки значень полів.

	Книга – назва, автор, рік видання	Книжковий магазин: *назва, автор, рік видання,* ціна	Збільшити вартість книги на 20%, якщо книжці менше, ніж 3 роки

2. ***Код програми:***

```
#include <iostream>
#include <string>
using namespace std;
class Book {
```

```
protected:
    string title;
    string author;
    int year;
public:

    Book(string t, string a, int y) : title(t), author(a), year(y) {
    }
    void printBook() {
        cout << "Назва: " << title << ", Автор: " << author << ", Рік видання: "
<< year << endl;
    }
    int getYear() {
        return year;
    }
};
class BookStore : public Book {
private:
    double price;
public:

    BookStore(string t, string a, int y, double p) : Book(t, a, y), price(p) {

    }
    void PriceNew() {
        int Year = 2024;
        if (Year - year < 3) {
            price *= 1.2;
        }
    }
    void print() {
        printBook();
        cout << "Ціна: " << price << " грн" << endl;
    }
};
int main() {
    setlocale(LC_ALL, "");
    BookStore book("Мистецтво війни", "Сунь-цзи", 2020, 250.00);
    book.print();
    book.PriceNew();
    cout << "Після збільшення ціни (якщо не минуло 3 роки після виходу її в
світ): " << endl;
    book.print();
    return 0;
```

}

3. ***Результат роботи програми:***

```
Назва: Мистецтво війни, Автор: Сунь-Ґзи, Рік видання: 2020
Ціна: 250 грн
Після збільшення ціни (Якщо книжка молодша 3 років):
Назва: Мистецтво війни, Автор: Сунь-цзи, Рік видання: 2020
Ціна: 250 грн


...Program finished with exit code 0
Press ENTER to exit console.
```

Минуло 3 роки після виходу книги

```
Назва: Мистецтво війни, Автор: Сунь-Ґзи, Рік видання: 2022
Ціна: 250 грн
Після збільшення ціни (Якщо книжка молодша 3 років):
Назва: Мистецтво війни, Автор: Сунь-цзи, Рік видання: 2022
Ціна: 300 грн


...Program finished with exit code 0
Press ENTER to exit console.
```

Минуло меньше 3-х років після виходу книги

4. Пояснення до програми:

Створюється батазовий клас **Book**, який має три поля **title, author, year**. У класі є конструктор з аргументами, два методи **printBook(), getYear()**.

Також створюється похідний клас **BookStore,** який успадковує властивості базового класу, а також має власний конструктор і метод **print()**. У головному коді програми створюються об'єкт **book**, за допомогою якого відбувається звернення до методів класу. Результат роботи методів виводиться на екран.

Завдання до самостійної роботи

№	Базовий клас і його поля	Похідний клас і його поля (поля базового класу виділені курсивом)	Функція-метод обробки даних об'єкта похідного класу
1	Дата – день, місяць, рік	Список друзів: ПІБ,	Обчислити кількість

		телефон, *дата* народження	днів до наступного дня народження
2	Дата – день, місяць, рік	Ліки: назва, *дата випуску*, фірма	Скільки днів пройшло після виготовлення ліків
3	Товар: назва, ціна, рік випуску	Фірмовий товар: *назва*, *ціна, рік випуску*, дата надходження товару	Кількість днів після року випуску до поточного дня
4	Товар: назва, ціна, виробник	Товар: *назва, ціна, виробник,* рік випуску, знижка у відсотках	Змінити вартість товару з урахуванням знижки для товарів, які виготовлені більше 2 років тому
5	Робітник–прізвище, оклад, рік народження	Робітники фірми: *прізвище, оклад, рік народження ,* посада	Збільшити оклад робітникам на посаді програміста на 20%
6	Книга – назва, автор, рік видання	Книжковий магазин: *назва, автор, рік видання,* ціна	Збільшити вартість книги на 20%, якщо книжці менше, ніж 3 роки, від випуску
7	Чотири цілих числа:a,b,c,d	П’ять чисел: *чотири цілих числа (a,b,c,d)* і число y	Обчислити добуток квадратів різниці кожного з чотирьох чисел і числа y

Практична робота №3

Тема: *«Перевизначення методів. Віртуальні функції»*

Мета роботи: здобути практичні навички щодо використання віртуальних функцій під час перевизначення методів.

Короткі теоретичні відомості

Ознакою віртуального методу (функції) (тобто такого методу, який перевизначається у разі успадкування) є ключове слово virtual. Під час перевантаження змінюється прототип методу. Під час перевизначення прототип методу залишається незмінним.

Приклад розробки програми

1. ***Умова задачі:***

Створити клас «**Трикутник**» з полями (довжини трьох сторін a, b, c) та методами «**Периметр**» і «**Площа**». Визначити метод, який буде виводити інформацію про трикутник: довжини сторін, периметр і площа. Створити похідний клас «**Чотирикутник**» з додатковими параметрами – довжиною четвертої сторони (d) і довжинами діагоналей (e,f) і перевизначити методи: методи «**Периметр**» і «**Площа**».

2. ***Код програми***:

```
#include <iostream>
#include <string>
#include <cmath>
using namespace std;
class triangle
{
public:
	double a;
	double b;
	double c;
	double pr;
	triangle()
	{
		this->a = NULL;
		this->b = NULL;
		this->c = NULL;
	}
	triangle(double a, double b, double c)
	{
		this->a = a;
		this->b = b;
		this->c = c;
	}
	virtual void perimeter()
	{
		this->pr = this->a + this->b + this->c;
		cout << "perimeter: " << this->a + this->b + this->c << endl;
	}
	virtual  void area()
	{
		double p = this->pr / 2;
		cout << "area: " << sqrt(p * (p - this->a) * (p - this->b) * (p - this->c)) << endl;;
```

```
        }
        ~triangle()
        {

        }
};
class rectangle : private triangle
{
        triangle tr;
        double d;
        double e;
        double f;
public:
        rectangle() : triangle()
        {
                this->d = NULL;
                this->e = NULL;
                this->f = NULL;
        }
        rectangle(double a, double b, double c, double d) : triangle(a, b, c)
        {
                this->d = d;
                this->e = NULL;
                this->f = NULL;
        }
 void perimeter()
        {
                cout << "perimeter: " << this->a + this->b + this->c + this->d
<< endl;
        }
        void aera()
        {
                this->e = sqrt(pow(this->a, 2) + pow(this->b, 2));
                this->f = sqrt(pow(this->c, 2) + pow(this->d, 2));
                double tmp;
                double tmp2;
                double tmp3;
                        tmp = pow((pow(b, 2) + pow(d, 2) - pow(a, 2) - pow(c,
                2)), 2);
                tmp2 = (4 * pow(e, 2) * pow(f, 2) - tmp);
                tmp3 = sqrt(tmp2 / 16);
                cout << "aera: " << tmp3 << endl;
        }
        ~rectangle()
```

```
        {

        }
};
int main()
{
        triangle tr(8, 6, 7);
        tr.perimeter();
        tr.area();
        rectangle rc(8, 6, 8, 6);
        rc.perimeter();
        rc.aera();
        return 0;
}
```

3. ***Результат роботи програми:***

```
perimeter: 21
area: 20.3332
perimeter: 28
aera: 48


...Program finished with exit code 0
Press ENTER to exit console.
```

4. ***Пояснення до програми:***

У програмі створюється базовий клас triangle, в якому описується віртуальний метод perimeter, де обчислюється периметр трикутника.

Наступним створюється віртуальний метод area, у якому обчислюється площа трикутника. Далі створюється похідний клас rectangle, де описується метод perimeter, в якому обчислюється периметр прямокутника, і метод area, у якому обчислюється площа прямокутника. В головному коді програми створюються об'єкти, за допомогою яких відбувається звернення до методів класу, щоб отримати результат роботи програми.

Завдання до самостійної роботи

1. Завдання: створити базовий клас з вказаними полями і методами та похідний клас – з додатковим полем і функцією, яка перевизначається.

№	Базовий клас, його поля та функція «якості» (Q)	Похідний клас, його поле та функція «якості» (Qp)
1	ВНЗ: назва закладу, кількість студентів, зарахованих на 1 курс, кількість випускників **Q=число випускників/кількість зарахованих**	P: відсоток випускників, які працюють за спеціальністю **Qp=P*Q**
2	Солдат: прізвище, зріст (см), вага (кг) **Q=зріст*вага**	P: освіта (навчальна, середня, вища) **Qp:** якщо освіта вища, то **Qp=2*Q;** а якщо навчальна, то **Qp=0.5*Q,** інакше **Qp=Q**
3	Мобільний телефон: марка, ціна, об'єм пам'яті. **Q=об'єм пам'яті/ціна**	P: кількість SIM карт **Qp=P*Q**
4	Програміст: прізвище, кількість програм, які він написав, кількість мов програмування, якими він пише програми. **Q=кількість програм*кількість мов**	P: кількість програм, які працюють правильно **Qp=P*Q / (кількість всіх програм)**
5	Вистава: назва, n1 – кількість глядачів спочатку, n2 – кількість глядачів в кінці. **Q=(n2-n1)/n1**	P: рік написання п'єси **Qp= Q * (T-P+1), де T-поточний рік**
6	Алмаз: назва, вага (в каратах), якість огранювання в балах. **Q=0,4*вага+0,6*якість огранювання**	P: колір (білий, голубий, жовтий тощо) **Qp: якщо колір голубий, то Qp= Q+1; якщо колір жовтий, то Qp= Q-0,5; інакше Qp= Q**
7	Комп'ютерна мережа: назва організації, число робочих станцій, середня відстань між станціями (м). **Q=число станцій*середня відстань**	P: середня швидкість передачі даних в мережі (Мб/с) **Qp= Q*P**
8	Армія: вид військ, чисельність (тис. чоловік), озброєність (в балах). **Q=0,3*чисельність+0,7*озброєність**	P: досвід (кількість місяців, на протязі яких армія вела бойові дії) **Qp= Q * (P+1)**

Практична робота №4

Тема: *«Узагальнені функції та класи»*

Мета роботи: проаналізувати використання шаблонів в програмі та здобути практичні навички в написанні програм.

Приклад розробки програми

1. ***Умова:***

Написати програму з узагальненим класом, у якого є поле – двовимірний масив. Описати метод, за допомогою якого міняються місцями два стовпці масиву. Індекси стовпців масиву передаються аргументами методу.

2. ***Код програми:***

```
#include <iostream>
#include <string>
     using namespace std;
const int SIZE = 4;
template <class T> class myclass {
     public:
    double arr[SIZE][SIZE], arr1[SIZE];
    int n, k;
    myclass() {
            enterarr();
            show();
            method(n, k);
            cout << endl;
            show();
    }
    void enterarr() {
            for (int i = 0; i < SIZE; i++) {
                    for (int j = 0; j < SIZE; j++) {
                            arr[i][j] = rand() % 20 + 1.25;
                    }
            }
            cout << "Enter number of first column: ";
            cin >> n;
            cout << "Enter number of secondcolumn: "
            cin >> k;
```

```
    }
    void show() {
        for (int i = 0; i < 4; i++) {
            for (int j = 0; j < 4; j++) {
                cout << arr[i][j] << "\t";
            }
            cout << endl;
        }
    }
    void method(int numb1, int numb2) {
        numb1 -= 1;
        numb2 -= 1;
        for (int i = 0; i < SIZE; i++) {
            arr1[i] = arr[i][numb1];
        }
        for (int i = 0; i < SIZE; i++) {
            arr[i][numb1] = arr[i][numb2];
        }
        for (int i = 0; i < SIZE; i++) {
            arr[i][numb2] = arr1[i];
        }
    }
};
int main()
{
    srand(time(NULL));
    myclass<int> first;
    return 0;
}
```

3. ***Результат програми:***

```
Enter number of first column: 2
Enter number of second column: 3
19.25   11.25   1.25    3.25
14.25   19.25   10.25   5.25
7.25    8.25    6.25    17.25
3.25    3.25    6.25    9.25

19.25   1.25    11.25   3.25
14.25   10.25   19.25   5.25
7.25    6.25    8.25    17.25
3.25    6.25    3.25    9.25


...Program finished with exit code 0
Press ENTER to exit console.
```

4. ***Пояснення до програми:***

В даній програмі створюється шаблон класу. У шаблоні описується двовимірний масив і заповнюється випадковими числами. А також описується метод, за допомогою якого міняються місцями два стовпці масиву. У головному коді програми створюється об’єкт. Користувач з клавіатури вводить номера стовбців, які будуть мінятися місцями. Після цього на екран виводяться відповідний результат.

Завдання до самостійної роботи

1) Написати програму з узагальненою функцією і аргументом – масивом. В результаті виконання функції міняються місцями елементи, індекси яких передаються аргументами функції.

2) Написати програму з узагальненою функцією і аргументом – масивом. В результаті виконання функції на екран виводяться елементи масиву.

3) Написати програму з узагальненою функцією і аргументом – масивом. В результаті виконання функції елементи масиву розміщуються у зворотному порядку.

4) Написати програму з узагальненою функцією для підрахунку співпадаючих елементів масиву – аргументу функції.

5) Написати програму з узагальненим класом, у якого є поле – двовимірний масив. Описати метод, за допомогою якого виконується пошук елементу в масиві. В якості результату повертається загальна кількість збігів.

6) Написати програму з узагальненим класом, у якого є поле – двовимірний масив. Описати метод, за допомогою якого міняються місцями два стовпці масиву. Індекси стовпців масиву передаються аргументами методу.

СПИСОК ЛІТЕРАТУРИ

1. *Григорович В.Г.* Об’єктно-орієнтоване програмування. Частина 1 : підручник / В.Г. Григорович. – Київ : видавництво «Магнолія 2006», 2023. – 284 с.
2. *Кравець П.О.* Об’єктно-орієнтоване програмування : навчальний посібник / П.О. Кравець. – Львів : видавництво «Львівська політехніка», 2012. – 624 с.
3. *Зеленський В. С.* Об’єктно-орієнтоване програмування на С++ : навчальний посібник / О. С. Зеленський, В. С. Лисенко. – Кривий Ріг : 2023. – 215 с.
4. *Бобков В. Б.* Програмування. Об’єктно-орієнтоване програмування : навчальний посібник [Електронний ресурс] : навчальний посібник / В.Б. Бобков, Ю. Є. Грудзинський, К. В. Крилов. – Київ : КПІ ім. Ігоря Сікорського, 2023. – 77 с. – Назва з екрана.

Навчально-методичне видання

ОБ’ЄКТНО-ОРІЄНТОВАНЕ ПРОГРАМУВАННЯ

Методичні вказівки
до виконання лабораторних робіт
для здобувачів першого (бакалаврського) рівня вищої освіти
за спеціальністю 174 «Автоматизація, комп’ютерно-інтегровані технології
та робототехніка»

Укладач **Бондарчук** Ольга Вячеславівна

Випусковий редактор *Л. С. Тавлуй*
Комп’ютерне верстання *К. А. Мавроді*

Підписано до друку 05.06.2025. Формат 60 х $84_{1/16}$
Ум. друк. арк. 1,86. Обл.-вид. арк. 2,0.
Електронний документ. Вид. № 52/ІІІ-25

Видавець і виготовлювач:
Київський національний університет будівництва і архітектури

Проспект Повітряних Сил, 31, Київ, Україна, 03037

Свідоцтво про внесення до Державного реєстру суб’єктів
видавничої справи ДК № 808 від 13.02.2002

32