Лабораторна робота №3

Тема. **Спадкування. Інтерфейси.**

Мета роботи: Ознайомитися з механізмом успадкування класів **Java**.

Тривалість роботи: 4 години.

Теоретичні відомості.

Спадкування (**inheritance**) - це відношення між класами, при якому клас використовує структуру або поведінку іншого (одиночне спадкування) або інших (множинне спадкування) класів. Спадкування вводить ієрархію "загальне / приватне", в якій підклас успадковує від одного або декількох більш загальних суперкласів. Підкласи зазвичай доповнюють або скасовують успадковану структуру і поведінка.

Синтаксис оголошення класу з успадкуванням:

```
<modifier>* class <class_name>
[extends parent_class |
implements parent_interfaces ]
 {
    <attribute_declaration>*
    <constructor_declaration>*
    <method_declaration>*
}
```

Інкапсуляція (encapsulation) - це приховування реалізації класу і відділення його внутрішнього уявлення від зовнішнього (інтерфейсу). При використанні об'єктно-орієнтованого підходу не прийнято використовувати прямий доступ до властивостей якого-небудь класу з методів інших класів. Для доступу до властивостей класу прийнято використовувати спеціальні методи цього класу для отримання і зміни його властивостей.

Відкриті члени класу складають зовнішній інтерфейс об'єкта. Це та функціональність, яка доступна іншим класам. закритими зазвичай оголошуються всі властивості класу, а так само допоміжні методи, які є деталями реалізації і від яких не повинні залежати інші частини системи.

Завдяки приховування реалізації за зовнішнім інтерфейсом класу можна міняти внутрішню логіку окремого класу, не змінюючи код інших компонентів системи.

Поліморфізм (**polymorphism**) - положення теорії типів, згідно якому імена (наприклад, змінних) можуть позначати об'єкти різних (Але мають загального батька) класів. Отже, будь-який об'єкт, позначається поліморфним ім'ям, може по-своєму реагувати на якийсь загальний набір операцій.

Рекомендації з проектування класів

- Завжди зберігайте дані в змінних, оголошених як **private**
- Завжди ініціалізуйте дані
- Не використовуйте в класі занадто багато простих типів

- Не для всіх полів треба створювати методи доступу і модифікації
- Використовуйте стандартну форму визначення класу
- Розбивайте на частини занадто великі класи
- Вибирайте для класів і методів осмислені імена

Приклад. Створити клас суперклас (базовий клас) Працівник(**`Employee`**) з даними про працівника, та підкласи (похідні класи) Менеджер(**`Manager`**) та Директор (**`Director`**) з додатковими полями. Розробити програму, яка вводить інформацію про співробітників та записує у масив та друкує введений масив.

```
import java.util.Scanner;
import java.io.IOException;
public class Lab3 {

    public static void main(String[] args) throws IOException {
        System.out.println("Привіт !");
        Scanner in = new Scanner(System.in);
        System.out.print("Введіть ім'я: ");
        String name = in.nextLine();
        System.out.print("Введіть розмір зарплати : ");
        int zarp = in.nextInt();
        System.out.print("Введіть рік прийому на роботу : ");
        int r = in.nextInt();
        System.out.print("Введіть місяць прийому на роботу : ");
        int m = in.nextInt();
        System.out.print("Введіть день прийому на роботу : ");
        int d = in.nextInt();
        Employee.num_r =0;
        Employee obj= new Employee(name,zarp,r,m,d);
        System.out.println("Ваше имя: " + obj.getName() + "  зарплата " +
obj.getSalary() );
        Employee obj1= new Employee("Степаненко С.");
        System.out.println("Ваше имя: " + obj1.getName() + "  зарплата " +
obj1.getSalary() );
        Employee obj2= new Employee();
        System.out.println("Ваше имя: " + obj2.getName() + "  зарплата " +
obj2.getSalary() );
        System.out.println(obj2.toString() );
        Employee Mg = new Manager(" Степан ");
        System.out.println("Ваше имя: " + Mg.getName() + "  зарплата = " +
Mg.getSalary() );
        Employee DirC = new Director(" Вася ");
        System.out.println("Ваше имя: " + DirC.getName() + "  зарплата = " +
DirC.getSalary() );

        if (obj.equals(obj1))  System.out.println(" ... Так "); else
System.out.println(" ... Ні ");
        in.close();
        System.out.println(" Кількість введених працівників " + Employee.num_r
);
    }
}

import java.util.Date;
import java.util.GregorianCalendar;

public class Employee  {
    /* перерахування полів класу*/
    private String name; // ім'я
    private double salary; // розмір заробітної плати
```

```
    private Date hiredate; // дата прийняття на роботу
    public static int num_r;   // номер співробітника
    // конструктор класу, завдання якого - задання  значень полів класу
    {
        name = "Noname N.";
        salary = 1005.01;
        hiredate = (new GregorianCalendar()).getTime();

    }
    public Employee (String n, double s, int year, int month, int day) {
        name = n;
        salary = s;
        hiredate = (new GregorianCalendar (year, month-1, day)). getTime ();
        num_r++;  //
    }
    public Employee(String in_name){
        name=in_name;
        salary=1000;
        hiredate=(new GregorianCalendar(2017,12,31)).getTime();
        num_r++;
    }
    public Employee(){
                num_r++;
    }

    // методи класу
    public String getName() {// повертає ім'я співробітника
        return name;
    }
    public double getSalary() {/* повертає розмір заробітної плати
співробітника*/
        return salary;
    }
    public Date getDate() {// повертає дату прийому на роботу
        return hiredate;}

    @Override
    public String toString() {
        String s;
        s = name + salary + hiredate.toString();
        return s;
    }

    @Override
    public boolean equals(Object obj) {
        boolean b = false;
        if ( obj instanceof Employee )
        {
            Employee obj1 = (Employee) obj;
            if (    name==obj1.getName() &&
                    salary==obj1.getSalary()  &&
                    hiredate == obj1.getDate() ) b = true;

        }
        return b;
    }
}
public class Manager extends Employee {


    public Manager(String n, double s, int year, int month, int day) {
        super(n, s, year, month, day);
    }
```

```
    public Manager(String in_name) {
        super(in_name);
    }

    public Manager() {
    }

    @Override
    public String getName() {
        return super.getName();
    }

    @Override
    public double getSalary() {
        return super.getSalary()+500;
    }


    @Override
    public String toString() {
        return "Manager{} " + super.toString();
    }
}

import java.util.Date;

public class Director extends  Employee {
    public Director(String n, double s, int year, int month, int day) {
        super(n, s, year, month, day);
    }

    public Director(String in_name) {
        super(in_name);
    }

    public Director() {
    }

    @Override
    public String toString() {
        return "Director{} " + super.toString();
    }

    @Override
    public Date getDate() {
        return super.getDate();
    }

    @Override
    public double getSalary() {
        return super.getSalary() + 2000;
    }
}
```

Завдання до лабораторної роботи

1. Зайти в свій обліковий запис на github.com. Зайти в github classroom.
2. Клонувати репозиторій: https://classroom.github.com/a/mugvYBVB **(211 група)** на робочий комп'ютер.
   (коледж https://classroom.github.com/a/YKUOhXNy)
3. Розв'язати задачі згідно варіанту.
4. В процесі написання функцій розв'язання задач лабораторної роботи періодично здійснювати синхронізацію з репозиторієм на github.com, з поясненням виконаної роботи (**git** add ., **git** commit -m"коментар", **git** push).
5. Оформити звіт про виконання лабораторної роботи.
6. Звіт відправити в для оцінювання в https://moodle.chnu.edu.ua або Google Classroom.

Завдання 1. Побудувати ієрархію класів відповідно до варіанта завдання. Згідно завдання вибрати суперклас (базовий клас) та підкласи (похідні класи). В класах задати поля, які характерні для кожного класу. Для всіх класів розробити метод Show(), який виводить дані про об'єкт класу. Розробити програму, яка вводить інформацію про об'єкти заданих сутностей згідно варіанту в масив типу суперкласу та друкує введений масив (з використанням методу Show()) .

1. Студент, викладач, персона, завідувач кафедри.
2. Службовець, персона, робітник, інженер.
3. Робітник, кадри, інженер, адміністрація.
4. Деталь, механізм, виріб, вузол.
5. Організація, страхова компанія, нафтогазова компанія, завод.
6. Журнал, книга, друковане видання, підручник.
7. Тест, іспит, випускний іспит, випробування.
8. Місце, область, місто, мегаполіс.
9. Іграшка, продукт, товар, молочний продукт.
10. Квитанція, накладна, документ, рахунок.
11. Автомобіль, поїзд, транспортний засіб, експрес.
12. Двигун, двигун внутрішнього згоряння, дизель, реактивний двигун.
13. Республіка, монархія, королівство, держава.
14. Ссавець, парнокопитне, птах, тварина.
15. Корабель, пароплав, вітрильник, корвет.

Завдання 2. Реалізувати абстрактний базовий клас з вказаними абстрактними методами. Створити підкласи(похідні класи) суперкласу(базового класу), в яких

здійснити реалізацію всіх абстрактних методів. Самостійно визначити, які поля необхідні і які з них визначити в базовому класі, а які – в похідних.

В похідних класах мають бути перевантажені методи toString та equal. Створити масив об'єктів . Проілюструвати роботу всіх методів підкласів(похідних класів).

1. Створити абстрактний базовий клас Figure з абстрактними методами обчислення площі і периметру. Створити похідні класи: Rectangle (прямокутник), Circle (коло), Trapezium (трапеція) зі своїми функціями для обчислення площі і периметра.
2. Створити абстрактний базовий клас Body (тіло) з абстрактними функціями обчислення площі поверхні і об'єму. Створити похідні класи Parallelepiped (паралелепіпед) і Ball (куля) зі своїми функціями площі поверхні і об'єму.
3. Створити абстрактний базовий клас Currency (валюта) для роботи з грошовими сумами. Визначити абстрактні функції переводу суми в гривні і виводу на екран. Створити похідні класи Dollar (доллар) і Euro (євро) зі своїми функціями переводу в гривні і друку.
4. Створити абстрактний базовий клас Root (корінь) з абстрактними функціями обчислення коренів і друку результату. Визначити похідні класи Linear (лінійне рівняння) і Square (квадратне рівняння) з власними функціями обчислення коренів і друку.
5. Створити абстрактний базовий клас Function (функція) з абстрактними методами обчислення значення функції в заданій точці x і виводу результату на екран. Визначити похідні класи Ellipse (еліпс) і Hyperbola (гіпербола) з власними функціями обчислення значення і друку.
6. Створити абстрактний базовий клас Pair з абстрактними арифметичними операціями додавання, віднімання та множення. Створити похідні класи Complex (комплексне число) і Rational (раціональний дріб) з власною реалізацією арифметичних операцій.
7. Створити абстрактний базовий клас Triad з абстрактними функціями збільшення на 1 і друку значення. Створити похідні класи Date (поточна дата), Time (поточний час) з власною реалізацією вищенаведених функцій.
8. Створити абстрактний базовий клас Pair з абстрактними арифметичними операціями додавання, віднімання, множення і ділення на ціле число. Створити похідні класи Money (гроші) та Fraction (дріб).
9. Створити абстрактний базовий клас Integer (ціле) з абстрактними арифметичними операціями додавання, віднімання та множення. Створити похідні класи Heximal (шіснадцяткове число) Binary (двійкове число) і перевантажити для них вищевказані функції. Число має бути представлене масивом своїх цифр.

10. Створити абстрактний базовий клас Series (прогресія) з абстрактними функціями обчислення вказаного члена прогресії та її суми. Визначити похідні класи Linear (арифметична прогресія) та Exponential (геометрична прогресія) з власною реалізацією вищенаведених функцій.
11. Створити абстрактний базовий клас Container з абстрактними методами sort() і обробці елементів контейнера norma(). Створити похідні класи Bubble і Choice. Для Bubble метод sort має реалізовувати бульбашкове сортування, а foreach обчислювати квадратний корінь з суми елементів. Для Choice метод sort має реалізовувати сортування методом вибору максимального елемента, а norma середнє арифметичне всіх елементів.
12. Створити абстрактний базовий клас Norm з абстрактними функціями обчислення норми вектора. Визначити похідні класи Vector2D (вектор на площині) і Vector3D (вектор у просторі) з власною реалізацією вищенаведених функцій. Для Vector2D модуль обчислювати як корінь з суми квадратів, для Vector3D як максимальне з компонентів вектора. Норму для похідних класів обчислювати як квадрат модуля.

13. Створити абстрактний базовий клас Number з абстрактними методами – арифметичними операціями додавання, віднімання, множення і ділення. Створити похідні класи Integer (ціле число), Real (дійсне число).
14. Створити абстрактний базовий клас Pair з абстрактними арифметичними операціями додавання, віднімання та множення і ділення на ціле число. Створити похідні класи Money (гроші) та Complex (комплексне число) з власною реалізацією вищевказаних функцій.
15. Створити абстрактний базовий клас Figure3D з абстрактними методами обчислення площі поверхні та об’єму. Створити похідні класи: Cube (куб), Sphere (сфера), Cone (конус) , Cylinder (циліндр), TriangularPyramid (трикутна піраміда) зі своїми функціями для обчислення площі поверхні та об’єму.

Завдання 3. Виконати попереднє завдання (Завдання 2) замінивши абстрактний клас інтерфейсом.