CS261 ASSIGNMENT 7

NAME:

ARCHIT AGRAWAL

ROLL NO. :

202051213

SECTION:

A

1. **Write a java program to illustrate use of final keyword with inheritance.**

***Code***

import java.util.\*;

class Library{

    //declared final so that it can't be changed anywhere

    protected final String name = "IIIT Vadodara Library";

    private int numOfBooks = 5000;

    public int getNumOfBooks(){

        return this.numOfBooks;

    }

    public void setNumOfBooks(int n){

        this.numOfBooks = n;

    }

    public final void getLibraryDetails(){

        System.out.println("Library Name : "+this.name);

    }

}

class Book extends Library{

    //instance variables

    private String title;

    private String author;

    private String genre;

    private String issuedTo;

    private int bookShelfNo;

    //constructor

    public Book(String title, String author, String genre, int n){

        this.title = title;

        this.author = author;

        this.genre = genre;

        this.bookShelfNo = n;

    }

    //getter-setter

    public String getIssuedTo(){

        return this.issuedTo;

    }

    public void setIssuedTo(String name){

        this.issuedTo = name;

    }

    public int getBookShelf(){

        return this.bookShelfNo;

    }

    public void setBookShelf(int n){

        this.bookShelfNo = n;

    }

    //method to display details

    public void printDetails(){

        System.out.println("Title     : " +this.title);

        System.out.println("Author    : " +this.author);

        System.out.println("Genre     : " +this.genre);

        System.out.println("Book Shelf: " +this.bookShelfNo);

        System.out.println("Issued To : " +this.issuedTo);

    }

}

//driver class

public class Driver{

    public static void main(String[] args){

        //creating book object

        Book book1 = new Book("Three Men in a Boat", "Jerome K. Jerome", "Travel/Comedy", 5);

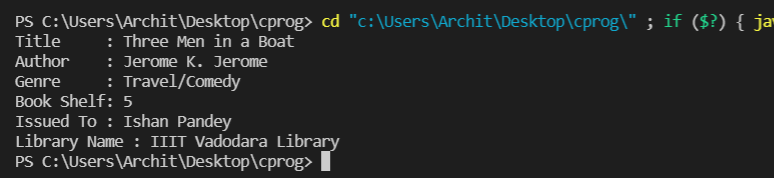
        book1.setIssuedTo("Ishan Pandey");

        book1.printDetails();

        book1.getLibraryDetails();

    }

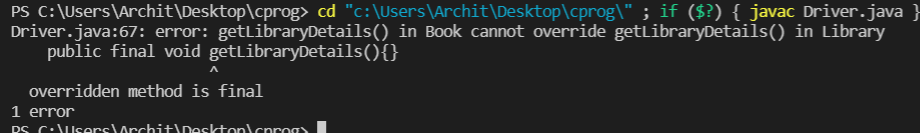
}

***Output***

Let us try overriding the getLibraryDetails() method in the class Book by adding the following method in the class Book.

//overriding a final method

public final void getLibraryDetails(){}

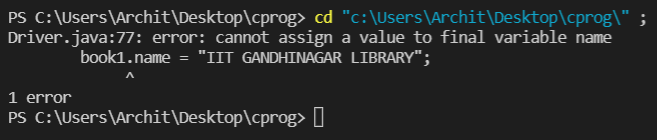
The following output is obtained.

The method cannot be overridden because it is declared final in the superclass.

Now, let us try to change the library name in the Driver class.

//changing a final variable

book1.name = "IIT GANDHINAGAR LIBRARY";

We receive the following output.

1. **Write a program in JAVA code to implement private and protected access modifiers in the same code.**

***Code***

import java.util.\*;

class Person{

    protected String name;

    protected int age;

    public Person(String name, int age){

        this.name = name;

        this.age = age;

    }

    public String getName(){

        return this.name;

    }

    public int getAge(){

        return this.age;

    }

    public void setName(String name){

        this.name = name;

    }

    public void setAge(int n){

        this.age = n;

    }

}

class Employee extends Person{

    private String company;

    private String position;

    private float salary;

    public Employee(String name, int age, String company, String position, float salary){

        super(name, age);

        this.company = company;

        this.position = position;

        this.salary = salary;

    }

    public String getCompany(){

        return this.company;

    }

    public void setCompany(String company){

        this.company = company;

    }

    public String getPosition(){

        return this.position;

    }

    public void setPosition(String position){

        this.position = position;

    }

    public float getSalary(){

        return this.salary;

    }

    public void setSalary(float salary){

        this.salary = salary;

    }

    public void showDetails(){

        System.out.println("Name     : "+this.name);

        System.out.println("Age      : "+this.age);

        System.out.println("Company  : "+this.company);

        System.out.println("Position : "+this.position);

        System.out.println("Salary   : "+this.salary);

    }

}

public class Main{

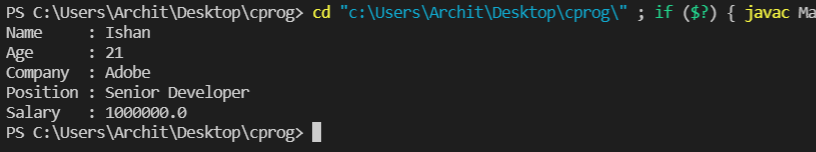
    public static void main(String[] args){

        Employee emp1 = new Employee("Ishan", 21, "Adobe", "Senior Developer", 1000000.0f);

        emp1.showDetails();

    }

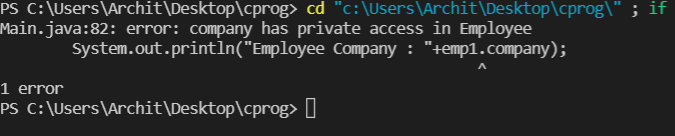
}

***Output***

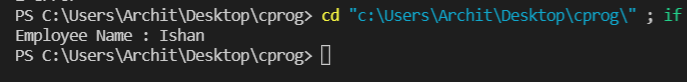
Let us try to access the emp1’s company directly from class Main using the following code in class Main

System.out.println("Employee Company : "+emp1.company);

We receive the following output.



Now, let us try to access the name of person, which has access modifier protected in the Person class, from the main class.



Since, the protected access specifier specifies the access in the same package and to sub-classes in different package, we got the name of the employee, as the Main class is in the same package.