Name: Aditya Ranjan Pradhan

Emp ID: 219861

Assignments Solution for OOP's and Java 8 features:

1Q) What are the main Types of OOP's ?

Object-Oriented Programming (OOP): is a way of designing and writing software where you group related tasks and data together into "objects

There are 4 Types of OOPS:

1. Encapsulation

2. Inheritance

3. Polymorphism

4. Abstraction

2Q. What you learn in Inheritance and what is the types of Inheritance ?

Inheritance: When the properties & methods of base class are passed into derived class

(Or one class acquiring the properties of other class.)

Why do we need inheritance: code reusability, polymorphism, abstraction.

Types of Inheritance:

a) Single Inheritance: In single inheritance, a class inherits from one base (parent) class.

b) Multilevel Inheritance: In multilevel inheritance, a class is derived from another class, which is also derived from another class.

c) Hierarchical Inheritance: In hierarchical inheritance, multiple classes are inherited from a single base class.

d) Multiple Inheritance (Java does not support directly) ---> We can do it by using interface.

Reason: To reduce the time complexity and simplify the language.

e) Hybrid Inheritance: It is the combination of single and multiple inheritance.

3Q. What you learn in Encapsulation, give real time examples?

Encapsulation->It is defined as wrapping up of data under a single unit is encapsulation.

It also implements data hiding.

Why do we need encapsulation: To avoid data directly exposed to the outside world.

Example:

```
class BankAccount {
```

```
private String accNo;
private String accHolderName;
private double bal;
public BankAccount(String accNo, String accHolderName, double initialBalance) {
   this.accNo = accNo;
   this.accHolderName = accHolderName;
   this.bal = initialBalance;
}
public void deposit(double amount) {
   if (amount > 0) {
      bal += amount;
      System.out.println("Deposited: " + amount);
   } else {
      System.out.println("Invalid deposit amount");
   }
}
public void withdraw(double amount) {
   if (amount > 0 && amount <= bal) {
      bal -= amount;
      System.out.println("Withdrew: " + amount);
   } else {
      System.out.println("Invalid withdrawal amount or insufficient balance");
   }
}
public String getAccountNumber() {
   return accNo;
}
public void setAccountNumber(String accNo) {
   this.accNo = accNo;
}
public String getAccountHolderName() {
   return accHolderName;
```

```
    }
    public void setAccountHolderName(String accountHolderName) {
        this.accHolderName = accountHolderName;
    }
     public double getBalance() {
        return bal;
    }
    public void setBalance(double balance) {
        this.bal = balance;
    }
}
public class Banking {
    public static void main(String[] args) {
        BankAccount account = new BankAccount("12345678", "Aditya Ranjan Pradhan", 1000.0);
        account.deposit(500.0);
        account.withdraw(200.0);
        System.out.println("Current Balance: " + account.getBalance());
        account.setAccountHolderName("Aditya Ranjan Pradhan");
        System.out.println("Updated Account Holder: " + account.getAccountHolderName());
    }
}
```

4Q. What will be the output of this program?

```
class TestMain {
    void add(int a, int b) {
        System.out.println(a + b);
    }
    void add(double a, double b) {
        System.out.println(a + b);
    }
    void add(double a) {
        System.out.println(a);
    }
```

```
    void add(int a, int b, int c) {
        System.out.println(a + b + c);
    }
}
public class ExampleMain {
    public static void main(String[] args) {
        TestMain testmain = new TestMain();
        testmain.add(10.5);
        testmain.add(10.5, 11.5);
        testmain.add(2, 4);
        testmain.add(5, 10, 15);
    }
}
```

Output:

```
C:\Users\adityap16\Desktop\Training Day-01\Day-01\B_Assignment> cmd /C ""C:\Program F
iles\jdk-17.0.1\bin\java.exe" -XX:+ShowCodeDetailsInExceptionMessages -cp C:\Users\ad
ityap16\AppData\Roaming\Code\User\workspaceStorage\5bc94a61c087b883c204e00144078028\r
edhat.java\jdt_ws\B_Assignment_1de7f0ce\bin Q5Output "
10.5
22.0
6
30
```

5Q. What will be the Output of the code ?

```
public class A {
void m1() {
System.out.println("class - A- m1 () method"); } }
public class B extends A {
@Override
void m1() {
System.out.println("class - B- m1 () method"); }
void m7() {
System.out.println("class- B- m7() method");
```

```
} }

public class TestMain {

public static void main(String[] args) {

B b= new B();

b.m1();

b.m7();

} }
```

OUTPUT:

```
C:\Users\adityap16\Desktop\Training Day-01\Day-01\B_Assignment> cmd /C ""C:\Program Files\jdk-17.0.1\bin\java.exe" -XX:+Show
CodeDetailsInExceptionMessages -cp C:\Users\adityap16\AppData\Roaming\Code\User\workspaceStorage\5bc94a61c087b883c204e001440
78028\redhat.java\jdt_ws\B_Assignment_1de7f0ce\bin Q6Output "
class - B- m1 () method
class- B- m7() method
```

6Q. Class for Employee with Multiple Inheritance (practice it on SpringBoot application):

Create a class Person with basic information like name and age.

Create another class Job with attributes like position and salary.

Create a class Employee that inherits from both Person and Job and displays all the information.

Person.java

```
package com.question7.question7;
public class Person {
   private String name;
   private int age;
   public Person(String name, int age) {
      this.name = name;
      this.age = age;
   }

   public String getName() {
      return name;
   }
   public void setName(String name) {
      this.name = name;
   }
   public int getAge() {
      return age;
   }
   public void setAge(int age) {
      this.age = age;
   }
}
```

Job.java

```
package com.question7.question7;

public interface Job {
   String getPosition();
   double getSalary();
}
```

Employee.java	Main.java

package com.question7.question7;
public class Employee extends Person implements Job {
  private String position;
  private double salary;
  public Employee(String name, int age, String position, double salary) {
  super(name, age);
  this.position = position;
  this.salary = salary;
  }
  @Override
  public String getPosition() {
  return position;
  }
  @Override
  public double getSalary() {
  return salary;
  }
  public void displayInfo() {
  System.out.println("Name: " + getName());
  System.out.println("Age: " + getAge());
  System.out.println("Position: " + getPosition());
  System.out.println("Salary: " + getSalary());
  }
}
	package com.question7.question7;
	
	import org.springframework.boot.SpringApplication;
	import org.springframework.boot.autoconfigure.SpringBootApplication;
	
	@SpringBootApplication
	public class Question7Application {
	
	public static void main(String[] args) {
	SpringApplication.run(Question7Application.class, args);
	Employee employee = new Employee("Aditya Ranjan", 21, "Software Developer", 75000.0);
	employee.displayInfo();
	}
	
	}

Output:

```
ers\adityap16\Desktop\Training Day-01\Day-01\B_Assignment\question7\target\classes started by
 adityap16 in C:\Users\adityap16\Desktop\Training Day-01\Day-01\B_Assignment\question7)2024-1
0-21T17:29:51.831+05:30  INFO 9476 --- [question7] [           main] c.q.question7.Question7A
pplication       : No active profile set, falling back to 1 default profile: "default"
2024-10-21T17:29:52.383+05:30  INFO 9476 --- [question7] [           main] c.q.question7.Ques
tion7Application       : Started Question7Application in 0.902 seconds (process running for 1
.25)
Name: Aditya Ranjan
Age: 21
Position: Software Developer
Salary: 75000.0
```

7Q. Try to create a Music Player Class:

Define a class Song with attributes like title, artist, and duration.

Create a class Playlist that maintains a collection of songs.

Implement methods to add songs, remove songs, and display the current playlist.

import java.util.*;

class Song{

```
        String tile;
        String artist;
        String duration;
        Song(String title,String artist, String duration){
            this.tile=title;
            this.artist=artist;
            this.duration=duration;
        }
        void display(){
            System.out.println("Song Title: "+this.tile+" Song artist: "+this.artist+" Song Duration: "+this.duration);
        }
}
 class PlayList{
    List<Song>playList;
    PlayList(){
        playList=new ArrayList<Song>();
    }
    void addSong(Song s){
        playList.add(s);
    }
    void removeSong(){
        if(!playList.isEmpty())
        playList.remove(playList.size()-1);
    }
    void displayPlayList(){
        for(Song s: playList){
```

```
            s.display();
        }
    }
}
public class Q7SongPlayList {
    public static void main(String[] args) {
        PlayList pl=new PlayList();
        pl.addSong(new Song("abc", "Badsha", "5:00"));
        pl.addSong(new Song("xyz", "Honey singh", "4:00"));
        pl.addSong(new Song("klm", "Sundhi chauhan", "1:00"))
        pl.displayPlayList();
        pl.removeSong();
        pl.displayPlayList();
    }
}
```

Output:

```
C:\Users\adityap16\Desktop\Training Day-01\Day-01\B_Assignment> cmd /C ""C:\Program F
iles\jdk-17.0.1\bin\java.exe" -XX:+ShowCodeDetailsInExceptionMessages -cp C:\Users\ad
ityap16\AppData\Roaming\Code\User\workspaceStorage\5bc94a61c087b883c204e00144078028\r
edhat.java\jdt_ws\B_Assignment_1de7f0ce\bin Q7SongPlayList "
Song Title: abc Song artist: Badsha Song Duration: 5:00
Song Title: xyz Song artist: Honey singh Song Duration: 4:00
Song Title: klm Song artist: Sundhi chauhan Song Duration: 1:00
Song Title: abc Song artist: Badsha Song Duration: 5:00
Song Title: xyz Song artist: Honey singh Song Duration: 4:00
```

8Q. What will be the output of below program ?

```
public class ElementWithA_Character {
public static void main(String[] args) {
List<String>name=Arrays.asList("Aniruddha","Pravin","Amit","Arvind","Swaraj","Babasaheb","C
at"," Extream","Lemon");
List<String> name1=name.stream().filter(i -> i.startsWith("A")).collect(Collectors.toList());
```

System.out.println(" The names starting with character A is :- " + name1);

} }

Output:

```
C:\Users\adityap16\Desktop\Training Day-01\Day-01\B_Assignment> cmd /C ""C:\Program Files\jdk-17.0.1\bin\java.exe" -XX:+Show
CodeDetailsInExceptionMessages -cp C:\Users\adityap16\AppData\Roaming\Code\User\workspaceStorage\5bc94a61c087b883c204e001440
78028\redhat.java\jdt_ws\B_Assignment_1de7f0ce\bin Q8Output "
 The names starting with character A is :- [Aniruddha, Amit, Arvind]
```

9Q. Take the below List of Integer and use Map functionality to return integers by Square. List<Integer> square = Arrays.asList(1,2,3,4,5,6,7,8,9);

```
import java.util.Arrays;

import java.util.List;

import java.util.stream.Collectors;

public class TStreamAPI2

{

   public static void main(String[] args)

   {

      List<Integer> sq = Arrays.asList(1,2,3,4,5,6,7,8,9);

      List<Integer> sq1 = sq.stream().map(i->i*i).collect(Collectors.toList());

      System.out.println("The square is: "+sq1);

   }

}
```

Output:

```
C:\Users\adityap16\Desktop\Training Day-01\Day-01\B_Assignment> cmd /C ""C:\Program F
iles\jdk-17.0.1\bin\java.exe" -XX:+ShowCodeDetailsInExceptionMessages -cp C:\Users\ad
ityap16\AppData\Roaming\Code\User\workspaceStorage\5bc94a61c087b883c204e00144078028\r
edhat.java\jdt_ws\B_Assignment_1de7f0ce\bin Q9Square "
The square is: [1, 4, 9, 16, 25, 36, 49, 64, 81]
```

10Q. What will be the output of below program ?

public class NoGreaterThan {

public static void main(String[] args) {

List<Integer> itr= Arrays.asList(10,20,30,40,50,60,70,80,90,100,1000);

List<Integer> itr2 =itr.stream().filter(i -> i>30).collect(Collectors.toList());

System.out.println(" This are the numbers greater than 30 :- " + itr2);

} }

Output:

```
C:\Users\adityap16\Desktop\Training Day-01\Day-01\B_Assignment> cmd /C ""C:\Program F
iles\jdk-17.0.1\bin\java.exe" -XX:+ShowCodeDetailsInExceptionMessages -cp C:\Users\ad
ityap16\AppData\Roaming\Code\User\workspaceStorage\5bc94a61c087b883c204e00144078028\r
edhat.java\jdt_ws\B_Assignment_1de7f0ce\bin Q10Output "
 This are the numbers greater than 30 :- [40, 50, 60, 70, 80, 90, 100, 1000]
```

Q. What will be the output of below program and add the substract method (int substract (int x,int y);) into interface and write business logic into 'TestMain' class and print the output.

package com.lambda;

public interface Addition {

int add(int x,int y);

}

----

package com.lambda;

public class TestMain {

public static void main(String[] args) {

Addition addition=(int a,int b) ->(a+b);

System.out.println(addition.add(20, 30));

}

}

Solution:

interface Addition {

    int add(int x, int y);

}

```
interface Subtraction {
    int sub(int x, int y);
}


public class Q11Quest {
    public static void main(String[] args) {
        Addition add = (int a, int b) -> (a + b);
        System.out.println(add.add(20, 30));
        Subtraction sub = (int a, int b) -> (a - b);
        System.out.println(sub.sub(10, 5));
    }
}
```

Output: