Girija S P

230701092

Object Oriented Programming Using Java

Week 5

1)

Create a class Mobile with constructor and a method basicMobile().

Create a subclass CameraMobile which extends Mobile class , with constructor and a method newFeature().

Create a subclass AndroidMobile which extends CameraMobile, with constructor and a method androidMobile().

display the details of the Android Mobile class by creating the instance. .

```
class Mobile{

}
class CameraMobile  extends Mobile {

}
class AndroidMobile extends CameraMobile {

}
```

expected output:

```
Basic Mobile is Manufactured
Camera Mobile is Manufactured
Android Mobile is Manufactured
Camera Mobile with 5MG px
Touch Screen Mobile is Manufactured
```

For example:

Result
Basic Mobile is Manufactured Camera Mobile is Manufactured Android Mobile is Manufactured Camera Mobile with 5MG px Touch Screen Mobile is Manufactured

```java
class Mobile{

    public Mobile(){

        System.out.println("Basic Mobile is Manufactured");

    }

}

class CameraMobile extends Mobile{

    public CameraMobile(){

        System.out.println("Camera Mobile is Manufactured");

    }

    public void newFeature(){

        System.out.println("Camera Mobile with 5MG px");
```

```java
        }
    }

class AndroidMobile extends CameraMobile{

    public AndroidMobile(){

        System.out.println("Android Mobile is Manufactured");

    }

    void androidMobile(){

        System.out.println("Touch Screen Mobile is Manufactured");

    }

}

class prog{

    public static void main(String[] args){

        AndroidMobile o=new AndroidMobile();

        o.newFeature();

        o.androidMobile();

    }

}
```

	Expected	Got	
✓	Basic Mobile is Manufactured Camera Mobile is Manufactured Android Mobile is Manufactured Camera Mobile with 5MG px Touch Screen Mobile is Manufactured	Basic Mobile is Manufactured Camera Mobile is Manufactured Android Mobile is Manufactured Camera Mobile with 5MG px Touch Screen Mobile is Manufactured	✓

Passed all tests! ✓

2)

Create a class known as "BankAccount" with methods called deposit() and withdraw().

Create a subclass called SavingsAccount that overrides the withdraw() method to prevent withdrawals if the account balance falls below one hundred.

For example:

Result
Create a Bank Account object (A/c No. BA1234) with initial balance of $500: Deposit $1000 into account BA1234: New balance after depositing $1000: $1500.0 Withdraw $600 from account BA1234: New balance after withdrawing $600: $900.0 Create a SavingsAccount object (A/c No. SA1000) with initial balance of $300: Try to withdraw $250 from SA1000! Minimum balance of $100 required! Balance after trying to withdraw $250: $300.0

Answer: (penalty regime: 0 %)

```java
class BankAccount {
```

```java
    private String accountNumber;
    private double balance;

    public BankAccount(String accountNumber, double balance){
        this.accountNumber=accountNumber;
        this.balance=balance;
    }

    // Method to deposit an amount into the account
    public void deposit(double amount) {
        // Increase the balance by the deposit amount
        balance+=amount;

    }

    public void withdraw(double amount) {
        if (balance >= amount) {
            balance -= amount;
        } else {
            System.out.println("Insufficient balance");
        }
    }

    // Method to get the current balance
    public double getBalance() {
        // Return the current balance
        return balance;

    }
}

class SavingsAccount extends BankAccount {
    // Constructor to initialize account number and balance
    public SavingsAccount(String accountNumber, double balance) {
```

```java
    // Call the parent class constructor

    super(accountNumber,balance);


  }


  // Override the withdraw method from the parent class

  @Override

  public void withdraw(double amount) {

    // Check if the withdrawal would cause the balance to drop below $100

    if (getBalance() - amount < 100) {

      // Print a message if the minimum balance requirement is not met

      System.out.println("Minimum balance of $100 required!");

    } else {

      // Call the parent class withdraw method

      super.withdraw(amount);

    }

  }

}


class prog {


  public static void main(String[] args) {

    // Print message to indicate creation of a BankAccount object

    System.out.println("Create a Bank Account object (A/c No. BA1234) with initial balance of $500:");

    // Create a BankAccount object (A/c No. "BA1234") with initial balance of $500

    BankAccount BA1234 = new BankAccount("BA1234", 500);

    // Print message to indicate deposit action

    System.out.println("Deposit $1000 into account BA1234:");

    // Deposit $1000 into account BA1234

    BA1234.deposit(1000);


    System.out.println("New balance after depositing $1000: $"+ BA1234.getBalance());


    // Print the new balance after deposit
```

```java
// Print message to indicate withdrawal action
System.out.println("Withdraw $600 from account BA1234:");
// Withdraw $600 from account BA1234
BA1234.withdraw(600);

// Print the new balance after withdrawal
System.out.println("New balance after withdrawing $600: $" + BA1234.getBalance());

// Print message to indicate creation of another SavingsAccount object
System.out.println("Create a SavingsAccount object (A/c No. SA1000) with initial balance of $300:");
// Create a SavingsAccount object (A/c No. "SA1000") with initial balance of $300
SavingsAccount SA1000 = new SavingsAccount("SA1000", 300);

// Print message to indicate withdrawal action
System.out.println("Try to withdraw $250 from SA1000!");
// Withdraw $250 from SA1000 (balance falls below $100)
SA1000.withdraw(250);
// Print the balance after attempting to withdraw $250
System.out.println("Balance after trying to withdraw $250: $" + SA1000.getBalance());
    }
}
```

	Expected	Got	
✓	Create a Bank Account object (A/c No. BA1234) with initial balance of $500: Deposit $1000 into account BA1234: New balance after depositing $1000: $1500.0 Withdraw $600 from account BA1234: New balance after withdrawing $600: $900.0 Create a SavingsAccount object (A/c No. SA1000) with initial balance of $300: Try to withdraw $250 from SA1000! Minimum balance of $100 required! Balance after trying to withdraw $250: $300.0	Create a Bank Account object (A/c No. BA1234) with initial balance of $500: Deposit $1000 into account BA1234: New balance after depositing $1000: $1500.0 Withdraw $600 from account BA1234: New balance after withdrawing $600: $900.0 Create a SavingsAccount object (A/c No. SA1000) with initial balance of $300: Try to withdraw $250 from SA1000! Minimum balance of $100 required! Balance after trying to withdraw $250: $300.0	✓

Passed all tests! ✓

3)

class College

{

protected   String collegeName;

public College(String collegeNameP) {

   // initialize the instance variables

   collegeName= collegeNameP;

   }

public void admitted() {

   System.out.println("A student admitted in "+collegeName);

}

}

class Student extends College{

String studentName;

String depart;

public Student(String collegeNameP, String studentNameP,String departP) {

  // initialize the instance variables

   super(collegeNameP);

  studentName=studentNameP;

  depart=departP;

}

```java
public String toString(){

    // return the details of the student

    return "CollegeName : "+collegeName+"\nStudentName : "+studentName+"\nDepartment : "+depart ;

}
}
class prog {

public static void main (String[] args) {

    Student  s1 = new Student("REC","Venkatesh","CSE");


    s1.admitted();                    // invoke the admitted() method

    System.out.println(s1.toString());

}
}
```

	Expected	Got	
✓	A student admitted in REC CollegeName : REC StudentName : Venkatesh Department : CSE	A student admitted in REC CollegeName : REC StudentName : Venkatesh Department : CSE	✓

Passed all tests! ✓