Assignment Questions 3

Q1.Write a simple Banking System program by using OOPs concept where you can get account Holder name balance etc?

ans:-

```
import java.util.Scanner;
class BankDetails {
   private String accno;
   private String name;
   private String acc_type;
   private long balance;
   Scanner sc = new Scanner(System.in);
   //method to open new account
   public void openAccount() {
      System.out.print("Enter Account No: ");
      accno = sc.next();
      System.out.print("Enter Account type: ");
      acc_type = sc.next();
      System.out.print("Enter Name: ");
      name = sc.next();
      System.out.print("Enter Balance: ");
      balance = sc.nextLong();
   }
   //method to display account details
   public void showAccount() {
```

```
        System.out.println("Name of account holder: " + name);
        System.out.println("Account no.: " + accno);
        System.out.println("Account type: " + acc_type);
        System.out.println("Balance: " + balance);
    }
    //method to deposit money
    public void deposit() {
        long amt;
        System.out.println("Enter the amount you want to deposit: ");
        amt = sc.nextLong();
        balance = balance + amt;
    }
    //method to withdraw money
    public void withdrawal() {
        long amt;
        System.out.println("Enter the amount you want to withdraw: ");
        amt = sc.nextLong();
        if (balance >= amt) {
            balance = balance - amt;
            System.out.println("Balance after withdrawal: " + balance);
        } else {
            System.out.println("Your balance is less than " + amt + "\tTransaction
failed...!!" );
        }
    }
```

```
    //method to search an account number
    public boolean search(String ac_no) {
        if (accno.equals(ac_no)) {
            showAccount();
            return (true);
        }
        return (false);
    }
}
public class BankingApp {
    public static void main(String arg[]) {
        Scanner sc = new Scanner(System.in);
        //create initial accounts
        System.out.print("How many number of customers do you want to input? ");
        int n = sc.nextInt();
        BankDetails C[] = new BankDetails[n];
        for (int i = 0; i < C.length; i++) {
            C[i] = new BankDetails();
            C[i].openAccount();
        }
        // loop runs until number 5 is not pressed to exit
        int ch;
        do {
            System.out.println("\n ***Banking System Application***");
```

```
System.out.println("1. Display all account details \n 2. Search by Account number\n 3. Deposit the amount \n 4. Withdraw the amount \n 5.Exit ");
System.out.println("Enter your choice: ");
ch = sc.nextInt();
   switch (ch) {
      case 1:
         for (int i = 0; i < C.length; i++) {
            C[i].showAccount();
         }
         break;
      case 2:
         System.out.print("Enter account no. you want to search: ");
         String ac_no = sc.next();
         boolean found = false;
         for (int i = 0; i < C.length; i++) {
            found = C[i].search(ac_no);
            if (found) {
               break;
            }
         }
         if (!found) {
            System.out.println("Search failed! Account doesn't exist..!!");
         }
         break;
      case 3:
```

```
    System.out.print("Enter Account no. : ");
    ac_no = sc.next();
    found = false;
    for (int i = 0; i < C.length; i++) {
      found = C[i].search(ac_no);
      if (found) {
        C[i].deposit();
        break;
      }
    }
    if (!found) {
      System.out.println("Search failed! Account doesn't exist..!!");
    }
    break;
  case 4:
    System.out.print("Enter Account No : ");
    ac_no = sc.next();
    found = false;
    for (int i = 0; i < C.length; i++) {
      found = C[i].search(ac_no);
      if (found) {
        C[i].withdrawal();
        break;
      }
```

```
                }
                if (!found) {
                   System.out.println("Search failed! Account doesn't exist..!!");
                }
                break;
             case 5:
                System.out.println("See you soon...");
                break;
          }
       }
       while (ch != 5);
    }
 }
```

Q2. Write a Program where you inherit method from parent class and show method Overridden Concept?

ans:-

```
class Vehicle{
  //defining a method
  void run(){System.out.println("Vehicle is running");}
}
//Creating a child class
class Bike2 extends Vehicle{
```

```
  //defining the same method as in the parent class
  void run(){System.out.println("Bike is running safely");}

  public static void main(String args[]){
  Bike2 obj = new Bike2();//creating object
  obj.run();//calling method
  }
}
```

Q3.Write a program to show run time polymorphism in java?

ans:-

```
class Animal{
void eat(){System.out.println("eating...");}
}
class Dog extends Animal{
void eat(){System.out.println("eating bread...");}
}
class Cat extends Animal{
void eat(){System.out.println("eating rat...");}
}
class Lion extends Animal{
void eat(){System.out.println("eating meat...");}
}
```

```
class TestPolymorphism3{
public static void main(String[] args){
Animal a;
a=new Dog();
a.eat();
a=new Cat();
a.eat();
a=new Lion();
a.eat();
}}
```

Q4.Write a program to show Compile time polymorphism in java?

ans:-

```
public class MethodOverloading {

	// 1 parameter
	void show(int num1)
	{
		System.out.println("number 1 : " + num1);
	}

	// 2 parameter
	void show(int num1, int num2)
```

```
    {
        System.out.println("number 1 : " + num1
                                + " number 2 : " + num2);
    }

    public static void main(String[] args)
    {
        MethodOverloading obj = new MethodOverloading();

        // 1st show function
        obj.show(3);

        // 2nd show function
        obj.show(4, 5);
    }
}
```

Q5. Achieve loose coupling in java by using OOPs concept?

ans:-

```
// parent or base class
class A
{
void foo()
```

```
{
  System.out.println("Inside the foo method of base class.");
}
}

// child or derived class
class B extends A
{
void foo()
{
  System.out.println("Inside the foo method of derived class.");
}
}

public class CouplingExample
{
// main method
public static void main(String argvs[])
{

// creating an object of class B
B obj = new B();
obj.foo();
}
```

}

Q6. What is the benefit of encapsulation in java?

ans:- Encapsulation prevents access to data members and data methods by any external classes. The encapsulation process improves the security of the encapsulated data.

Q7.Is java a t 100% Object oriented Programming language? If no why ?

ans:- No! Java is not a "PURE" Object Oriented Language , because it uses primitive data types such as (int,float,char...etc). The developers of java could have made these primitive data types as objects(like String... etc), but the primitive data types such as int float... are more faster than objects! So, in order to achieve faster execution of java program they kept Primitive data type as it is!!

So Java is said to be 98% Object oriented language but not PURE Object Oriented language!!

Q8.What are the advantages of abstraction in java?

ans:-

Advantages of Abstraction

It reduces the complexity of viewing things. Avoids code duplication and increases reusability. Helps to increase the security of an application or program as only essential details are provided to the user. It improves the maintainability of the application.

Q9.What is an abstraction explained with an Example?

ans:- As Java is an OOP language, abstraction can be seen as one of the important features and building blocks of the Java language. In Java, abstraction is implemented using an abstract class and interface.

```
//abstract class
abstract class Car{
   abstract void accelerate();
}
//concrete class
class Suzuki extends Car{
   void accelerate(){
      System.out.println("Suzuki::accelerate");

   }
}
class Main{
   public static void main(String args[]){
      Car obj = new Suzuki();    //Car object =>contents of Suzuki
      obj.accelerate();         //call the method
   }
}
```

Q10.What is the final class in Java?

ans:- Final Class in Java uses the final keyword for its declaration. The final keyword in java is used to restrict the user, similarly, the final class means that the class cannot be extended. We can only create a final class if it is complete in nature, which means it cannot be an abstract class. All wrapper classes in Java are final classes, such as String, Integer, etc. Final class cannot be inherited by any subclass,

If we try to inherit a final class, then the compiler throws an error during compilation.