IT22037 Name: Md Shahin Mia

TOPIC NAME : _____ DAY : _____

TIME : DATE : / /

1. Classes and Object

```
Class car {
    String color;
    int speed;
  Void drive() {
      System.out.println("Car is driving...");
    }
}

Public class Main {
Public static void main (String[] args) {
Car mycar = new car(); //object creation
    mycar.color = "Red";
    mycar.speed = 100;
    mycar.drive();
  }
}
```

2. Access Modifiers

```
Class Person {
    Private string name;
  Public void setName (string newname) {
    name = newName; }
```

GOOD LUCK

TOPIC NAME : ID: IT 22037 DAY : TIME : DATE : / /

```
Public String getName(){
    return name;
}

Public class main {
    Public static void main (String[] args) {
        Person p = new person();
        P.setName("Alice");
        System.out.println( P.getName());
```

3. Inheritance and Protected Access

```
Class Animal{
    Protected String type = "Animal".
    void display(){
        System.out.println("This is an animal.");
    }
}
Class Dog extends Animal{
    void bark(){
        system.out.println(type+" Says Woof!");
    }
}
```

GOOD LUCK

```
Public class Main {
    Public Static void main( String[] args)
    {
        Dog d = new Dog ();
        d. display ();
        d. bank ();
    }
}
```

4. Encapsulation

```
Class BankAccount {
    Private double balance;
    Public void deposit (double amount);
        if (amount > 0) balance + = amount;
        }
    Public double getBalance ();
        return balance;
    }
}

Public Class main {
    Public static void main (String[] args);
        BankAccount acc = new BankAccount ();
        acc. deposit (500);
```

```
system.out.println(acc.getBalance());
    }
}
```

5. Abstract Classes

```
abstract class Animal {
    abstract void makeSound();
    void sleep() {
        system.out.println("Sleeping...");
    }
}

class Dog extends Animal {
    void makeSound() {
        System.out.println("Bark Bark");
    }
}

Public class main {
    Public static void main(String[] args) {
        Dog d = new Dog();
        d.makeSound();
        d.sleep();
    }
}
```

6. Interface:

```
interface Animal {
    void sound();
}
Class cat implements Animal{
    Public void sound () {
        System.out.println("Meow")
        }
}
Public class main{
    Public static void main (String [] args){
        cat c = new cat ()
        c.sound ();
        }
}
```

7. Multiple Inheritence using Interface

```
Interface flyable {
    void fly ();
}
interface Swimmable {
    void swim ();
}
```

```
class Duck implements Flyable, swimmable {
  Public void fly() {
    System.out.println ("Duck is flying ... ");
  }
Public void swim() {
    System.out.println ("Duck is swimming ... ");
  }
}
Public class main {
  Public static void main (string [] args) {
      Duck d = new duck();
       d.fly();
      d.swim();
  }
}
```

8: ATM - Project (Mini project idea)

```
import Java.util.Scanner;
Public class ATM {
    Private double balance = 1000.0;
  Public void deposite (double amount) {
    balance += amount; }
```

```
else{
    System.out.println("Insufficient balance!");
    }
}

Public void checkBalance(){
    System.out.println("Current Balance: " + balance);
}

Public static void main(String[] args){
    ATM atm = new ATM();
    Scanner sc = new Scanner(System.in);
    while(true){
    System.out.println("\n1. Deposit 2. Withdraw 3. Balance
                        4. Exit");
    int choice = sc.nextInt();
    Switch(choice){
    Case 1;
        System.out.println("Enter amount:");
        atm.deposit(sc.nextDouble());
        break;
Case 2: System.out.println("Enter Amount:");
```

TOPIC NAME : IT22037 DAY : TIME : DATE : / /

```
break.
case 3:
    atmCheckBalance();
    break;
case 4: system.exit(0);
}
}
}
```

Calculator:

```
import java.util.Scanner;
Public class Calculator {
  Public static void main (string[] args) {
  Scanner se = new Scanner (System.in)
  System.out.println("Enter first number");
  double num2 = se.nextDouble();
  System.Out.print("Enter second number)
  double num2 = se.nextDouble();
System.out.println("Chose Operation (+,-,*,/);");
  Char op = se.next().charAt(0);
  double result = 0;
  Switch (OP) {
      case '+': result = num1 + num2; break
      case '-': result = num1 - num2; break
```

```
case'/':
    if (num2 != 0)
        result = num1 / n
    else
        System.out.println("Cannot divide by zero");
    break
default : System.out.println("Invalid");
    }
}
System.out.println("Result " + result);
```