CHOO TIAN LONG 24068668

WIX1002 Fundamentals of Programming
Tutorial 8 Class

1. Write statements for each of the following
   a. Define a class Student.

```
public class Student{}
```

   b. Declare the instance variable that used to store contact number.

```
private String contactNumber;
```

   c. Create the constructor that initializes the contact number to null.

```
  public Student(){
    this.contactNumber = null;
  }
```

   d. Create another constructor that assign the parameter value to the contact number.

```
    public Student(String c){
      this.contactNumber = c;
    }
```

   e. Create an accessor and mutator method for the contact number.

```
    public String getContactNumber(){
      return contactNumber;
    }
    public void setContactNumber(String c){
      this.contactNumber = c;
    }
```

   f. Create a method that used to display the contact number.

```
    public void displayContactNumber(){
      System.out.println("The contact number is " + this.contactNumber);
    }
```

CHOO TIAN LONG 24068668

g. Create an object of the class Student.

```
Student s = new Student();
```

h. Change the contact number using the mutator method.

```
s.setContactNumber("012-3456789");
```

i. Create an object of the class Animal.

```
Animal a = new Animal();
```

j. Create an object of the class Animal that used to represent a cat.

```
Animal cat = new Animal("cat");
```

k. Create an object of the class Number with the value 20 and 40.

```
Number n = new Number(20,40);
```

2. Write statements for each of the following
   a. Define a class Digit.

```
public class Digit{}
```

b. Declare the instance variable that used to store a number.

```
private int number;
```

c. Create a constructor that assign the parameter value to the number.

```
    public Digit(int n){
      this.number = n;
    }
```

d. Create a digitMultiplication method that returns the multiplication of the number. If the number is 1345, the method will return 60.

```
    public int digitMultiplication(){
        int recursion = this.number;
        int result = 1;
        while(recursion > 0){
            result *= (recursion % 10);
            recursion /= 10;
        }
        return result;
    }
```

e. Create a method that used to display the digit multiplication of the number.

```
    public void displayDigitMultiplication(){
        System.out.printf("The digit multiplication of %d is %d\n", this.number,
digitMultiplication());
    }
```

f. Create a tester class that displays the digit multiplication of 4567.

```
    public class tester{
        public static void main(String[] args) {
            Digit d = new Digit(4567);
            d.displayDigitMultiplication();
        }
    }
```

CHOO TIAN LONG 24068668

3. Create a class that used to represent the 2 dimension coordinate system. The class consists of constructors, instance variables, accessor and mutator method and an output method that display the x-coordinate and y-coordinate.

```
public class coordinate {
    // instance variables
    private int x;
    private int y;

    // Without arguments constructor
    public coordinate(){
        this.x = 0;
        this.y = 0;
    }

    // With arguments constructor
    public coordinate(int x, int y){
        this.x = x;
        this.y = y;
    }

    // accessor 'Get'
    public int getX(){
        return x;
    }

    public int getY(){
        return y;
    }

    // mutator 'Set'
    public void setX(int x){
        this.x = x;
    }
```

```
        public void setY(int y){
            this.y = y;
        }

        public void displayCoordinate(){
            System.out.printf("Coordinate: (%d,%d)\n", this.x, this.y);
        }
}
```

4. Create a class Payment that accept different type of payment methods such as cash payment, cheque payment and credit card payment. For cash payment, the class accepts the amount in cash; for cheque payment, the class accepts the amount and the cheque number; for credit card payment, the class accepts the amount, card holder name, cardType, expiration date and validation code. Use the same method name for the payment.

```
public class Payment{
    private double amount;
    private String chequeNumber, cardHolderName, cardType, expirationDate,
      validationCode;

    // Method overloading
    // cash
    public void payment(double a){
        this.amount = a;

        System.out.println("Total pay by cash is " + this.amount);
    }

    // cheque
    public void payment(double a, String cn){
        this.amount = a;
        this.chequeNumber = cn;

        System.out.println("Total pay by cheque is " + this.amount);
        System.out.println("Cheque number: " + this.chequeNumber);
    }

    // credit card
    public void payment(double a, String chn, String ct, String ed, String vc){
        this.amount = a;
        this.cardHolderName = chn;
        this.cardType = ct;
        this.expirationDate = ed;
        this.validationCode = vc;
```

```
                System.out.println("Total pay by credit card is " + this.amount);
                System.out.println("Credit Card number: " + this.chequeNumber);
                System.out.println("Card Type: " + this.cardType);
                System.out.println("Validation Code: " + this.validationCode);
            }
}
```

CHOO TIAN LONG 24068668

5. Create a class Connection. The Connection class keeps track of the number of connections to the server. Whenever an object is created, a connection is established. The class has a disconnect method and a display method that display the number of connections to the server.

```
    public class Connection{
        private static int number_connections = 0;


        public Connection(){
            number_connections++;
        }


        public void disconnect(){
            number_connections--;
        }


        public void displayConnection(){
            System.out.println("The number of connections to the server: " +
number_connections);
        }
    }
```