นางสาวธัญเรศ โอบอ้อม รหัส 63543206018-7

ใบงานการทดลองที่ 3

เรื่อง อาเรย์สตริง และฟังก์ชัน ในภาษาจาวา

1. จุดประสงค์ทั่วไป

1.1. รู้และเข้าใจการโปรแกรมเชิงวัตถุร่วมกับอาเรย์และสตริง

1.2. รู้และเข้าใจการโปรแกรมเชิงวัตถุร่วมกับฟังก์ชัน

2. เครื่องมือและอุปกรณ์

เครื่องคอมพิวเตอร์1 เครื่อง ที่ติดตั้งโปรแกรม Eclipse

3. ทฤษฎีการทดลอง

3.1. โครงสร้างข้อมูลแบบ “อาเรย์” มีลักษณะเป็นอย่างไร ? มีองค์ประกอบอะไรบ้าง ? อธิบายพร้อมยก ตัวอย่างประกอบ

ลักษณะของอาร์เรย์ คือตารางที่แบ่งเป็นช่องๆ แต่ละช่องสามารถเก็บข้อมูลได้ 1 หน่วย อาจเป็นตัวเลขหนึ่งตัวหรือตัวอักษรหนึ่งตัว หรือหลายๆ ตัว ช่องทุกช่องของอาร์เรย์ต้องเก็บข้อมูลแบบเดียวกันนั้นคือ เป็นตัวอักษรล้วนหรือเป็นตัวเลขล้วนและ ขนาดของอาร์เรย์แต่ละช่องต้องเท่ากันหมด อาร์เรย์เป็นโครงสร้างที่เกือบกล่าวได้ว่าเป็นที่คุ้นเคยมากที่สุดและเข้าใจง่ายที่สุด ทั้งนี้เนื่องจากโครงสร้างของอาร์เรย์ตรงกับความเป็นจริงตามธรรมชาติของข่าวสารข้อมูลหลายประการ เช่น คะแนนของนักเรียนในชั้นเรียน บัญชีรายชื่อพนักงานของบริษัทฯ นอกจากนี้โปรแกรมที่เขียนคำนวณงานเมตริกซ์ หรือพีชคณิตเชิงเส้น ก็ต้องใช้อาร์เรย์เป็นที่เก็บตัวเลข นอกจากนี้อาร์เรย์ยังเป็นโครงสร้างพื้นฐานของโครงสร้างที่สำคัญอื่น ๆ อีก เช่น stack , queue , tree เป็นต้น

3.2. การเข้าถึงแต่ละ Element ของอาเรย์สามารถทำได้อย่างไร ? อธิบายพร้อมยกตัวอย่างประกอบ

เราจะทำการดึงข้อมูลหรือ element ในตัวแปรอาเรย์ออกมาใช้งานกัน โดยวิธีที่ง่ายที่สุดคือการกำหนดตัวเลขดัชนี (index) ให้กับข้อมูลอาเรย์นั้นๆ เพื่อเป็นการกำกับว่าต้องการดึงข้อมูลขนิดไหนออกมา เช่น

Flower[0]; //ข้อมูล Rose

Flower[1]; //ข้อมูล Violet

Flower[2]; //ข้อมูล Tulip

3.3. คำสั่ง length เกี่ยวข้องกับอาเรย์อย่างไร ? อธิบายพร้อมยกตัวอย่างประกอบ

เราสามารถหาขนาดของอาเรย์ได้ด้วยการอ่านค่าจาก Property length โดย Property นั้นจะส่งค่ากลับเป็นจำนวนสมาชิกทั้งหมดในอาเรย์ นี่เป็นตัวอย่าง

```
let fruits  = ["Apple", "Banana", "Orange"];
console.log(fruits);
console.log("Length of fruits: ", fruits.length);
```

```
[ 'Apple', 'Banana', 'Orange' ]
Length of fruits:  3
```

3.4. จงยกตัวอย่างประกอบในการวนรอบเพื่อแสดงค่าภายในตัวแปรอาเรย์ตั้งแต่ค่าแรกจนถึงค่าสุดท้าย

```
let numbers = [10, 20, 30, 40, 50];
for (let index in numbers) {
    console.log(`numbers[${index}]: ${numbers[index]}`);
}
```

```
numbers[0]: 10
numbers[1]: 20
numbers[2]: 30
numbers[3]: 40
numbers[4]: 50
```

3.5. จงยกตัวอย่างการใช้งานคำสั่ง for each เพื่อแสดงค่าภายในตัวแปรอาเรย์

```
let continents = [
    "Asia", "Africa", "North America", "South America",
    "Antarctica", "Europe", "Australia"
];

continents.forEach(function(value, index) {
    console.log(value, index);
});
```

```
Asia 0
Africa 1
North America 2
South America 3
Antarctica 4
Europe 5
Australia 6
```

3.6. เหตุใดจึงต้องมีคำสั่ง import java.util.Arrays ; ในส่วนต้นของไฟล์?

ต้องทำการ import ก่อน คลาสถึงจะใช้ได้

3.7. คำสั่ง Arrays.copyof(____ , ____) ; มีหน้าที่ทำอะไร ?

ซึ่งการใช้เมธอดนี้เราสามารถปรับให้ขนาดของอาเรย์ใหม่เล็กกว่าหรือใหญ่กว่าเดิมได้ โดยข้อมูลที่ถูกสำเนามาจะถูกเก็บตามดัชนีเดิม สังเกตุว่าเมื่อเราแก้ไขค่าในตัวแปรใหม่ ค่าในตัวแปรเดิมจะไม่เปลี่ยน แสดงว่าอาเรย์ที่สำเนามาเป็นคนละอันกับอาเรย์เดิมจริงๆ

3.8. จงยกตัวอย่างการประกาศ String และกำหนดค่าคำว่า “Hello World” ในภาษาจาวา

```
// Hello World Program
public class HelloWorld {

    public static void main (String[] args) {
        System.out.println("Hello World!");
    }

}
```

3.9. จงอธิบายและยกตัวอย่างประกอบการใช้งานคำสั่ง toUpperCase() ในภาษาจาวา

เป็นรูปแบบ property และ method เกี่ยวกับข้อความ (String) โดย toUpperCase() จะเป็นการแปลงข้อความ String ให้อยู่ในรูปแบบของ ตัวอักษรพิมพ์ใหญ่

```
package com.java.myapp;

public class MyClass {

      public static void main(String[] args) {

            String a = "Welcome to ThaiCreate.Com";

            System.out.println(a.toUpperCase());

      }

}
```

3.10. จงอธิบายและยกตัวอย่างประกอบการใช้งานคำสั่ง toLowerCase() ในภาษาจาวา

เป็นรูปแบบ property และ method เกี่ยวกับข้อความ (String) โดย toLowerCase() จะเป็นการแปลงข้อความ String ให้อยู่ในรูปแบบของ ตัวอักษรพิมพ์เล็ก

```
package com.java.myapp;

public class MyClass {

	public static void main(String[] args) {

		String a = "Welcome to ThaiCreate.Com";

		System.out.println(a.toLowerCase());

	}

}
```

3.11. จงอธิบายและยกตัวอย่างประกอบการใช้งานคำสั่ง indexOf() ในภาษาจาวา

เป็นรูปแบบ property และ method เกี่ยวกับข้อความ (String) โดย indexOf() จะเป็นการหาตำแหน่งของข้อความที่ต้องการค้นหา

```
package com.java.myapp;

public class MyClass {

	public static void main(String[] args) {

		String a = new String("Welcome to ThaiCreate.Com Version 2013");
		String b = new String("ThaiCreate.Com");

		System.out.println(a.indexOf("Version"));
		System.out.println(a.indexOf(b));

	}

}
```

3.12. จงอธิบายความแตกต่างระหว่างการเชื่อม String แบบปกติและแบบใช้คำสั่ง concat( )

Concat() ใช้ในการเชื่อมต่อข้อความ ใช้งานเหมือนการใช้เครื่องหมาย “+” สะดวกมากกว่าการเพิ่มแบบปกติหลายๆบรรทัด

3.13. หากต้องการแสดงสัญลักษณ์พิเศษภายในตัวแปร String ควรทำอย่างไร ?

ใช้ Unicode ได้ (กรณีที่แป้นพิมพ์ไม่มีปุ่มให้กด)

3.14. จงอธิบายและยกตัวอย่างประกอบการสร้างฟังก์ชันในภาษาจาวา

กลุ่มของชุดคำสั่งที่ถูกรวมเข้าด้วยกันสำหรับการทำงานบางอย่าง ฟังก์ชันสามารถรับพารามิเตอร์เพื่อนำมาทำงานและส่งค่ากลับได้ นอกจากนี้การสร้างฟังก์ชันทำให้เราสามารถเรียกใช้มันซ้ำๆ ซึ่งเป็นการนำโค้ดกลับมาใช้ใหม่ได้ (Reusable) ซึ่งนี่เป็นแนวคิดพื้นฐานของการใช้งานฟังก์ชันในการเขียนโปรแกรม

```
function name(param1, param2, ...) {
    // Statements
    return value; // Optional
}
```

3.15. อธิบายข้อแตกต่างระหว่าง Pass by value และ Pass by reference

Pass by Value คือ การส่งค่า (value) เป็น argument ของฟังก์ชัน ดังนั้นค่าที่ทำในฟังก์ชันจึงไม่ส่งผลต่อตัวแปรนอกฟังก์ชัน

Pass by Reference คือ การส่งตัวแปร (variable) เป็น argument ของฟังก์ชัน ดังนั้นตัวแปรที่มีการดำเนินการใด ๆ ในฟังก์ชันจะส่งผลให้ตัวแปรนอกฟังก์ชันมีการเปลี่ยนแปลงด้วย

3.16. ความแตกต่างระหว่างการประกาศฟังก์ชันแบบ void กับแบบ int, double, float, string คืออะไร ?

type เป็นประเภทของฟังก์ชันสำหรับการส่งค่ากลับ ประเภทของฟังก์ชันนั้นจะเป็นเหมือนประเภทของตัวแปร เช่น Integer, Floating, Double หรือแบบอ็อบเจ็ค ประเภทแบบ void หมายความว่าฟังก์ชันไม่มีค่าที่ต้องส่งกลับ

3.17. โครงสร้างข้อมูลแบบ Stack แตกต่างกับ Array อย่างไร ?

เป็นโครงสร้างข้อมูลแบบเชิงเส้น ที่มีการใส่ข้อมูลเข้า และนำข้อมูลออกเพียงด้านเดียว ดังนั้น ข้อมูลที่เข้าไปอยู่ใน stack ก่อนจะออกจาก stack หลังข้อมูลที่เข้าไปใน stack ทีหลัง นั่นคือ การ "เข้าทีหลังแต่ออกก่อน" (Last In First Out : LIFO) ต่างจาก array โดยที่ array จะทำหน้าที่เก็บค่าเฉยๆโดยจะเรียงข้อมูลเอาไว้เป็นช่องๆ สามารถดึงข้อมูลออกใช้ได้อย่างอิสระ

3.18. อธิบายพร้อมยกตัวอย่างประกอบกระบวนการทำงานของคำสั่ง Push ในโครงสร้างข้อมูลแบบ Stack

การทำงานจะตรงข้ามกับ Push จะดึงเอาข้อมูลที่อยู่บนสุดออกมาก่อน แต่ก่อนที่จะดึงจะมีการตรวจสอบว่ากองซ้อนว่างหรือไม่ ถ้าว่างจะไม่สามารถนำข้อมูลออกได้ แสดงว่ากองซ้อนว่าง (Stack Empty)ถ้าไม่ว่างจะนำเอาข้อมูลออกแล้วเลื่อนตัวชี้ไปยังตำแหน่งถัดลงไป

3.19. อธิบายพร้อมยกตัวอย่างประกอบกระบวนการทำงานของคำสั่ง Pop ในโครงสร้างข้อมูลแบบ Stack

กระทำที่ส่วนบนของสแตก (Pop) ซึ่งต้องมีการตรวจสอบก่อนว่าสแตกเต็มหรือไม่ เป็นการดำเนินการที่นำข้อมูลเข้าไปเก็บไว้ด้านบนสุดของกองซ้อน (Top of the Stack) เรื่อย ๆ จนกว่ากองซ้อนไม่สามารถนำข้อมูลเข้าไปเก็บได้จะเรียกว่า กองซ้อนเต็ม (Stack Full)

3.20. อธิบายพร้อมยกตัวอย่างประกอบกระบวนการทำงานของคำสั่ง isEmpty ในโครงสร้างข้อมูลแบบ Stack

EmptyStack : เป็นการดำเนินการเพื่อตรวจสอบว่าสแตกนั้นมีข้อมูลหรือไม่ กรณีใช้โครงสร้างข้อมูลอาร์เรย์ ต้องสั่งให้ตรวจสอบว่าที่ Top ของสแตกนั้นมีค่า = 0 หรือไม่

EmptyStack := Stack.top := 0; {ถ้าเป็นจริงจะส่งค่า True ออกมาให้โปรแกรมหลัก}

กรณีใช้โครงสร้างข้อมูลลิงค์ลิสต์ ให้ตรวจสอบว่าตัวชี้สแตกไม่ชี้ไปที่ไหน

EmptyStack := Stack.Nil; {ถ้าเป็นจริงคืนค่า True ถ้าไม่จริง คืนค่า False}

3.21. อธิบายพร้อมยกตัวอย่างประกอบความหมายของคำว่า Stack overflow

ปัญหาที่เกิดจากหน่วยความจำระหว่างเรียกเมท็อดไม่พอ จนเกิดปัญหา Stack overflow หรือปัญหาสแตกล้นขึ้น (อธิบายสั้น ๆ คือ การจองใช้หน่วยความจำจนเต็ม) ปัญหานี้มาจากความผิดพลาดของผู้เขียนโปรแกรมเอง ที่ไม่มีการออกแบบโปรแกรมและตรวจสอบให้ดี

```
def deepnesting(depth):
  r = {}
  for i in range(depth):
    n = {}
    n[i] = r
    r = n
  return r
```

4. ลำดับขั้นการปฏิบัติการ

4.1. จงแก้โจทย์ปัญหาดังต่อไปนี้

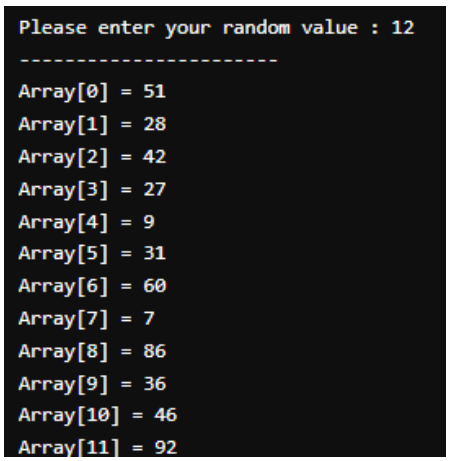
4.1.1. จงเขียนโปรแกรมเพื่อสุ่มค่าเข้าไปในอาเรย์ 1 มิติตามจำนวนค่าที่รับจากผู้ใช้ โดยค่าที่ถูกสุ่มจะ ต้องเป็นตัวเลขจำนวนเต็มที่อยู่ระหว่าง 0 ถึง 99 เท่านั้น

Test case 1	Test case 2
Please enter your random value : 8 ---------- Axray{0] = 94 Avray(t] = 32 Arrayl2] = 46 Aray(3} = 18 Arrayl4] = 27 Arrayl5] = 5 Array{6] = 31 Arrayl7] = 17	Please enter your random value :12 ---------- Arrayl0) = 56 Array(t) = 27 Array(2) = 13 Array(3) = 15 Avrayla] = 65 Avrayl5) = 29 Avrayl6] = 11 Avray(7] = 92 Avrayl6] = 95 Avrayl9] = 47 Avray(10] = 58 Array(11) = 62

4.1.2. ผังงานแสดงกระบวนการทำงานและโค้ดโปรแกรม(ที่ตรงตามผังงาน)

ผังงาน	โค้ดโปรแกรม
	

```java
package LABOOP;

import java.util.Scanner;
import java.util.Random;

public class LAB3_1 {

    public static void main(String[] args) {

        System.out.print("Please enter your random value : ");

            int n;

            Scanner scanIn = new Scanner(System.in);
            n = scanIn.nextInt();
            scanIn.close();

        System.out.println(" ");
        System.out.println("----------------------");

        int[] ar = new int[n];
        Random rand = new Random();

        for(int i = 0; i <= n ; i++ ) {
            ar[i]= rand.nextInt(100);
            System.out.println("Array[" + i + "] = "+ ar[i]);
        }
    }
}
```



```
Please enter your random value : 12
----------------------
Array[0] = 51
Array[1] = 28
Array[2] = 42
Array[3] = 27
Array[4] = 9
Array[5] = 31
Array[6] = 60
Array[7] = 7
Array[8] = 86
Array[9] = 36
Array[10] = 46
Array[11] = 92
```

4.2. จงแก้โจทย์ปัญหาดังต่อไปนี้

4.2.1. จงเขียนฟังก์ชันการจัดการโครงสร้างข้อมูลแบบ Stack พร้อมจำลองการทำงานโดยการเรียกใช้ คำสั่งพื้นฐานดังต่อไปนี้

คำสั่ง Push(String Value) ; เพื่อนำข้อมูลเข้าไปเก็บไว้ใน Stack

คำสั่ง Pop( ) ; เพื่อนำข้อมูลบนสุดออกจาก Stack

คำสั่ง isEmpty( ) ; เพื่อตรวจสอบข้อมูลใน Stack ว่ามีอยู่หรือไม่

คำสั่ง Top( ) ; เพื่อตรวจสอบข้อมูลที่อยู่ชั้นบนสุด

คำสั่ง CheckStack( ) ; เพื่อตรวจสอบค่าภายใน Stack ทั้งหมด

คำสั่ง SetStackSize(int size) ; เพื่อกำหนดขนาดเริ่มต้นของ Stack

Test case
SetStackSize(3)
isEmpty
----| Yes
Top
----| NULL
Push : Hello
CheckStack
----| STACK : Hello
Push : Hi
CheckStack
----| STACK : Hi, Hello
Push : Test
CheckStack
----| STACK : Test, Hi, Hello
Top
----| Top = Test
Pop
CheckStack
----| STACK : Hi, Hello
Isempty
----| No
Push : OK
CheckStack
----| STACK : OK, Hi, Hello
Push : RMUTL
----| Stack Overflow
CheckStack
----| STACK : OK, Hi, Hello

4.2.2. ผังงานแสดงกระบวนการทำงานและโค้ดโปรแกรม(ที่ตรงตามผังงาน)

ผังงาน	โค้ดโปรแกรม
	

```java
package LABOOP;

import java.util.*;
import java.util.Scanner;

public class LAB3_2 {

    public static void main(String[] args) {

        int n;
        int i = 0;
        String text;

        System.out.print("Input Stack Size : ");
           Scanner sc = new Scanner(System.in);
           n = sc.nextInt();

        Stack<String> stack = new Stack<String>();
        stack.setSize(n);
        stack.clear();

        do{
              System.out.println("-------------------------------- ");
              System.out.println("Stack Fn ");
              System.out.println("-------------------------------- ");
              System.out.println("1 : Push ");
              System.out.println("2 : Pop ");
              System.out.println("3 : isEmpty ");
              System.out.println("4 : Top ");
              System.out.println("5 : CheckStack ");
              System.out.println("10 : END");
              System.out.println("-------------------------------- ");

              System.out.print("Input ");
              i = sc.nextInt();
              System.out.println("");
              System.out.println("-------------------------------- ");

        switch(i) {

        case 1:
            System.out.print("Push : ");
            Scanner sct = new Scanner(System.in);
            text = sct.nextLine();
            if(stack.size() == n) {
```

```
    text = scr.nextLine();
    if(stack.size() == n) {
        System.out.println("----| STACK OVERFLOW!!!!!!");
    }else {
          stack.push(text);
    }
    break;

case 2:
     if(stack.size() == 0) {
          System.out.println("----| STACK IS EMPTY");
       }else {
          System.out.println("Pop");
          stack.pop();
       }//end if
    break;

case 3:
    if(stack.isEmpty() == true) {

        System.out.println("----| Yes");

    }else {

        System.out.println("----| No");
    }//end if
    break;

case 4:
    if(stack.size() == 0) {
        System.out.println("----| NULL");
    }else {
        System.out.println("----| Top : "+stack.peek());
    }//end if
    break;

case 5:
    System.out.println("----| STACK : "+stack);
    break;

case 10:
    i = 10;
    break;
}// end switch
```

5. สรุปผลการปฏิบัติการ

ได้เรียนรู้เกี่ยวกับพวกการใช้ method , array , stack ต่างๆ เพื่อนำไปใช้ในการฝึกฝนทำโจทย์

6. คำถามท้ายการทดลอง

6.1. ข้อควรระวังในการใช้งาน Array ในภาษาจาวาคืออะไร ?

ข้อมูลที่จะเก็บในกล่องนี้ได้ ต้องเป็นข้อมูลชนิดเดียวกัน และที่สำคัญคือต้องแจ้งไว้ล่วงหน้าด้วยว่าเราจะใช้กี่กล่อง

6.2. ข้อควรระวังในการใช้งาน String ในภาษาจาวาคืออะไร ?

1. ข้อความจะต้องอยู่ภายใน Double Quote (“”)

2. หลีกเลี่ยงการต่อ String โดยใช้การ + โดยเฉพาะเมื่อต่อ String จำนวนมาก เพราะ Performance ไม่ดี

3. หากเขียนโปรแกรมที่เป็น Thread safe ให้ใช้ StringBuilder เนื่องจากมี Performance ที่ดีกว่า (การต่อ String โดยทั่วไปจะเข้าข่ายข้อนี้มากที่สุด)

4. หากต้องการต่อ String แบบ Thread safe เนื่องจากโปรแกรมที่เราเขียนไม่มีการป้องกัน Thread safe ก็สามารถใช้ StringBuffer (การต่อ String ทั่วไปไม่ควรตกข้อนี้)

6.3. ฟังก์ชันในภาษาจาวาไม่สามารถใช้งานแบบ Pass by reference ในภาษาซีได้คุณมีแนวทางการ แก้ไขปัญหานี้ได้อย่างไร ?

การเปรียบเทียบค่าระหว่าง String ซึ่งเป็น Object หนึ่งของ Java สามารถทำได้โดยใช้ (==) ถึงแม้ว่า ถ้าใช้กับ Object จะเป็นการเทียบ reference ก็ตาม หรือใช้ .equals()

6.4. โครงสร้างข้อมูลแบบ Stack แตกต่างกับโครงสร้างข้อมูลแบบ Array อย่างไร ?

เป็นโครงสร้างข้อมูลแบบเชิงเส้น ที่มีการใส่ข้อมูลเข้า และนำข้อมูลออกเพียงด้านเดียว ดังนั้น ข้อมูลที่เข้าไปอยู่ใน stack ก่อนจะออกจาก stack หลังข้อมูลที่เข้าไปใน stack ทีหลัง นั่นคือ การ "เข้าทีหลังแต่ออกก่อน" (Last In First Out : LIFO) ต่างจาก array โดยที่ array จะทำหน้าที่เก็บค่าเฉยๆโดยจะเรียงข้อมูลเอาไว้เป็นช่องๆ สามารถดึงข้อมูลออกใช้ได้อย่างอิสระ