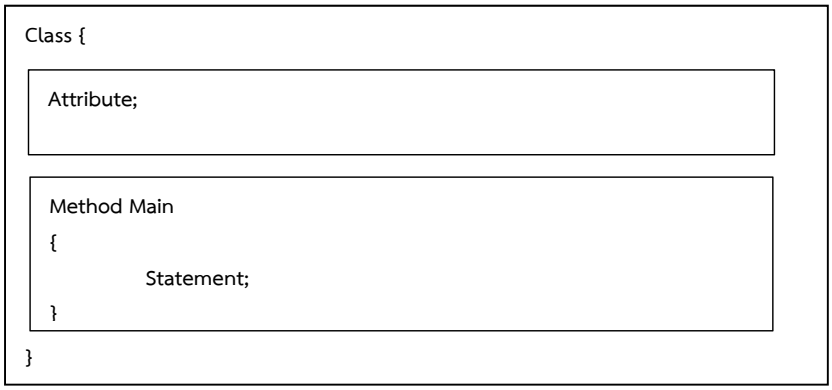
Programming Fundamentals II

Lap4:
- Method declaration
- Parameter Passing
- Passing by reference
- Classes and Object

1.1 เมธอด Method

จากแลปนิสิตได้มีโอกาสเขียน Method เองมาบ้างแล้วโดย Method ที่เขียนขึ้นมีชื่อว่า Main ซึ่งเป็น Method พิเศษ ที่เป็นจุดเริ่มต้นของโปรแกรมโดยมีโครงสร้างเป็นดังภาพด้านล่าง



สำหรับโครงสร้างของภาษา Java ที่มีหลาย Method อยู่ใน Class จะมีโครงสร้างที่มี Method ที่กำหนดขึ้นจะต้องประกาศไว้ใน class แต่อยู่ภายนอก Method อื่น ๆ และไม่จำเป็นต้องประกาศ Method Main เป็นลำดับแรกใน class

1.2 Method ชนิดไม่คืนค่าข้อมูล

Method แบบไม่คืนค่า คือ Method ที่เหมาะสำหรับใช้ทำงานใดงานหนึ่ง ซึ่งเมื่อทำงานเสร็จสิ้นแล้วไม่ต้องการผลลัพธ์ส่งกลับมาแก่โปรแกรมหลัก หรือ Method ที่เรียกใช้

Lab 4.1 ให้นิสิตสร้าง class ว่า Lab41Method จากนั้นให้นิสิตคัดลอกโปรแกรมนี้ลงใน Editor/IDE จากนั้นศึกษาการทำงานของโปรแกรม จากนั้นบันทึกผลลัพท์ที่ได้

```
public class Lab41Method
{
	public static void printClass()
	{
		for( int i = 0 ; i< 10 ; i++ )
		{
			System.out.print("=");
		}
		System.out.print(" Class : Fundamental-II \n");
	}

	public static void main(String[] args)
	{
		printClass();
		printClass();
	}
}
```

จากตัวอย่าง จงเขียน Method ใหม่เพิ่มเติมชื่อ printUniversity โดยที่เมธอด printUniversity จะพิมพ์อักษร '=' ทั้งหมด 5 อักษร จากนั้นพิมพ์คำว่า "University : Kasetsart Sriracha"

```
public static _______  ____________________
{




}
```

หากต้องการให้โปรแกรมแสดงผลตามผลลัพท์ด่านล่าง ส่วนของโปรแกรมในเมธอด Main ควรแก้ไขเป็นเช่นไร

```
========== Class : Fundamental-II
====== University: Kasetsart Sriracha
========== Class : Fundamental-II
====== University: Kasetsart Sriracha
```

เขียน code ที่อยู่ใน method main ใส่มาในกล่องข้อความด่านล่าง

```
public static void main(String[] args)
{




}
```

1.3 การส่งค่าให้ Method

เราสามารถเขียน Method ปลายทางให้มีการรับค่าจาก Method ต้นทาง เพื่อให้ Method ปลายทางนำค่านั้นๆไปใช้ในการคำนวณหรือกำหนดพฤติกรรมบางอย่างของ Method ปลายทางนั้นๆได้ ค่าที่ถูกส่งไปนี้เรียกว่า Argument ส่วน Method ปลายทางจะรับค่าเหล่านีผ่านมาทาง Parameter ซึ่ง Method แต่ละ Method สามารถมี Parameter กี่ตัวก็ได้

Lab 4.2 ให้นิสิตสร้าง class ว่า Lab42MethodV2 โดยให้สร้าง Method printClassV2 ต่อไปนี้เป็น Method ที่พัฒนามากจาก Method printClass ใน Class: Lab41Method โดย Method printClassV2 นีมี้การกำหนด Parameter 2 ตัวที่รับมาจาก Method ต้นทางคือ name เป็น Parameter ชนิด string และ length เป็น Parameter ชนิด int

```
public class Lab42MethodV2
{
	public static void printClassV2(String name, int length)
	{
		for( int i = 0 ; i< length ; i++ )
		{
			System.out.print("=");
		}
		System.out.printf(" Class : %s \n",name);
	}

	public static void main(String[] args)
	{
		printClassV2(------------(A)--------- , --------(B)--------- );

	}
}
```

โปรแกรมข้างต้นให้ผลลัพธ์เป็นเช่นไรเมื่อแทนที่พื้นที่ (A) และ (B) ด้วยข้อความต่อไปนี้

(A)	(B)	ผลลัพท์
“Objects”	15	
“Argument”	25	
45	“Attribute”	

Lab 4.3 ให้นิสิตสร้าง class ว่า Lab43MethodV3 ให้นิสิตเขียน Method ชื่อ DrawSqure ให้สมบูรณ์เพื่อให้เป็น Method ที่มี Parameter 2 ตัวประกอบด้วย c เป็น Parameter ชนิด char และ length เป็น Parameter ชนิด int โดย Method จะนำอักขระในพารามิเตอร์ c ไปวาดรูปสี่เหลี่ยมจัตุรัสที่มีความยาวด้านเท่ากับ length หน่วย ดังตัวอย่าง

```
// drawSqure('*', 5);    /* ตัวอย่าง Statement ที่ใส่ใน Method main */

  *  *  *  *  *
  *  *  *  *  *
  *  *  *  *  *
  *  *  *  *  *
  *  *  *  *  *
```

เขียนโคดใน class Lab43MethodV3 ลงในช่องคำตอบด้านล่าง

```
public class Lab43MethodV3
{
	public static _________ drawSqure(_____________________________)
	{









	}

	public static void main(String[] args)
	{
		drawSqure('*', 5);
	}
}
```

1.4 การส่งข้อมูลประเภท Array ให้ Method

การส่งพารามิเตอร์ให้ Method ในภาษา Java นอกจากเป็นประเภทข้อมูลพื้นฐาน (เช่น int, string, double, char, bool เป็นต้น) แล้ว เราสามารถส่งข้อมูลประเภท Arrays แก่ Method ได้เช่นกัน พิจารณาตัวอย่างต่อไปนี้

Lab 4.4 ให้นิสิตสร้าง class ว่า Lab44MethodV4 ให้นิสิตเขียนโปรแกรมคำนวณค่าเฉลี่ยเลขคณิต นิสิตจะได้ศึกษาการส่ง Arrays ให้กับ Method โดย Method ชื่อ calAverage เป็น Method ที่รับ Arrays เป็น Parameter แล้วใช้ Arrays นั้นหาค่าเฉลี่ยของข้อมูล

```
public class Lab44MethodV4
{
	public static double calAverage(int[] input)
	{
		double sum = 0;
		for(int i = 0 ; i < input.length ; i++ )
		{
			System.out.printf("%5d",input[i]);
			sum = sum + input[i];
		}
		System.out.println("");
		return (double)(sum/input.length);
	}

	public static void main(String[] args)
	{
		double average;
		int[] data = { 120, 98, 125, 90, 82 };
		average = calAverage(data);
		System.out.printf("Average : %f",average);
	}
}
```

จากโปรแกรมตัวอย่างด้านบน ให้นิสิตเติมคำในช่องว่างใน Method Main ที่กำหนดให้และเขียน Method ชื่อ findMax เพื่อให้เป็น Method ที่หาตัวเลขที่มากที่สุดใน Arrays ที่ Method รับมาเป็น Parameter ตัวอย่างผลลัพท์ด้านล่าง

```
Please input n : 5
Input {1} = 13
Input {2} = 98
Input {3} = 139
Input {4} = 130
Input {5} = 76
Max is 139
```

เติมโคดใน Method Main ใน class Lab44MethodV4 ลงในช่องคำตอบด้านล่าง

```
public static void main(String[] args)
{
	int n,i;
	Scanner scan = new Scanner(System.in);
	System.out.print("Please input n : ");
	n = scan.nextInt();

	_____________________________________________        // ประกาศ Arrays


	for( _______________  ; _____________________; __________ )
	{

		System.out.printf("Input {%d} = ",_____________);

		_____________________ = scan.nextInt();
	}


	System.out.printf("Max is %d",_____________________________ );
}
```

เขียนโคดใน Method findMax ใน class Lab44MethodV4 ลงในช่องคำตอบด้านล่าง

```
public static int findMax(_____________________________)
{









}
```

จาก Lab3 Arrays ที่ผ่านมานิสิตจะพบว่าการแสดง Matrix ใดๆ นั้นมี code ที่ยุ่งยาก ยิ่งหากต้องแสดงหลายๆครั้งก็จะทำให้ code ในโปรแกรมยาวและอ่านยากมากขึ้นดังนั้น เราจึงควรมี Method สำหรับแสดง Matrix เพื่อที่เวลาต้องการจะแสดง Matrix ใดๆนั้นก็เพียงแค่เรียกใช้ Method นี้

Lab 4.5 ให้นิสิตสร้าง class ว่า Lab45MethodArrays และเขียน Method ชื่อ showMatrix ซึ่งเป็น Method ที่รับ Arrays เป็น Parameter โดย Method showMatrix จะนำค่าใน Arrays นั้นแสดงออกมาในรูปแบบของ Matrix ดังตัวอย่าง

```
Matrix A:
    5    3    8
    2    6   10
    1    8   25
   12    3   30
```

เขียนโคดในลงในช่องคำตอบด้านล่างเพื่อให้ได้ผลลัพท์ที่แสดงด้านบน

```
public class Lab45MethodArrays
{
      public static void showMatrix (_____________________)
      {

            for(____________ ;____________; ____________)
            {

                  for( ____________ ;_____________; ____________)
                  {

                        System.out.printf("_______",__________);
                  }
                  System.out.println("");
            }
      }

      public static void main(String[] args)
      {
            int[][] A = {
                        { 5, 3, 8},
                        { 2, 6, 10},
                        { 1, 8, 25},
                        { 12, 3, 30}

            };
            System.out.println("Matrix A: ");
            showMatrix(A);
      }
}
```

จากแบบโปรแกรมข้างต้น ให้นิสิตเขียน Method ชื่อ showAddMatrix เพิ่มเติมซึ่งเป็น Method แสดงผลบวกของ Matrix 2 Matrix ใดๆ และเติมคำในช่องว่างของ Method Main ต่อไปนี้ให้สมบูรณ์ เพื่อให้โปรแกรมคำนวณและแสดงผลเป็นดังตัวอย่าง

```
Matrix A:
   -3    5    6
    5    0   -2
Matrix B:
    9    0   -5
   -3   -2   -1
Matrix A+B:
    6    5    1
    2   -2   -3
```

เขียนโคดใน class Lab45MethodArrays ลงในช่องคำตอบด้านล่าง

```
public class Lab45MethodArrays2
{
	public static void showMatrix (_____________________)
	{









	}

	public static void showAddMatrix (_____________________________)
	{









	}

	public static void main(String[] args)
	{
		int[][] A = {
				{ -3, 5, 6},
				{ 5, 0, -2}
		};
		int[][] B = {
				{ 9, 0, -5},
				{ -3, -2, -1}
		};

		System.out.println("Matrix A: ");

		_____________________________________

		System.out.println("Matrix B: ");

		_____________________________________

		System.out.println("Matrix A+B: ");

		_____________________________________
	}
}
```

1.5 การส่งค่าให้ Method โดยการอ้างอิง (Pass by reference)

จากโปรแกรมที่ผ่านมานิสิตได้ฝึกเขียน Method มามากมาย ซึ่งการส่ง Argument ให้ Parameter ใน Method ทั้งหมดที่ผ่านมานั้นเป็นการส่ง Argument โดยใช้ค่า (Pass by value) กล่าวคือ ค่าของ Argument จะถูกคัดลอกลงไปใน Parameter ของ Method และ Parameter เหล่านี้จะถูกใช้งานภายใน Method เสมือนเป็นตัวแปรที่ถูกประกาศภายใน Method นั้นๆนั่นเอง ดังนั้นการเปลี่ยนแปลงค่าของ Parameter จึงไม่มีผลกระทบต่อค่าภายใน Argument ใดๆทั้งสิ้น

แต่สำหรับ Method บาง Method การส่ง Argument โดยใช้ค่านั้นอาจไม่เหมาะสมจึงควรใช้การส่งค่าแบบอื่นซึ่งก็คือการส่ง Argument โดยใช้การอ้างอิง (Pass by reference) แทน ซึ่งการส่ง Argument โดยใช้การอ้างอิงนี้ Parameter ใน Method จะเปรียบเสมือนเป็นตัวเดียวกันกับ Argument นั่นหมายถึงการกำหนดค่าใหม่ให้แก่ Parameter ชนิดนีก็จะส่งผลให้ค่าของ Argument ที่สอดคล้องกัน เปลี่ยนแปลงตามไปด้วย ซึ่งเป็นประโยชน์อย่างมากกับ Method ที่ต้องการให้มีการคืนค่ามากกว่าหนึ่งค่า (โดยปกติ Method จะใช้คำสั่ง return ค่าได้เพียงค่าเดียวเท่านั้น) ในภาษา Java นี้จะเป็นการส่ง object หรือเรียกอีกอย่างว่า instances ของคลาสเหล่านี้จะถูกส่งแบบ Pass by reference ทั้งสิ้น

Lab 4.6 ให้นิสิตสร้าง class ว่า Lab46PassByRef และทดลองพิมพ์ code ตัวอย่างด้านล่าง

```
public class Lab46PassByRef
{
	public static void doubleMe1(Integer x)
	{
		x = x * 2;
		System.out.printf("In method doubleMe1: x={%d} \n",x);
	}

	public static void doubleMe2(int x)
	{
		x = x * 2;
		System.out.printf("In method doubleMe2: x={%d} \n",x);
	}

	public static void main(String[] args)
	{
		Integer a = new Integer(10);
		int b = 20;

		System.out.printf("Before methods: a={%d}, b={%d} \n",a,b);
		doubleMe1(a);
		doubleMe2(b);
		System.out.printf("After methods: a={%d}, b={%d} \n",a,b);
	}
}
```

แสดงผลลัพท์ที่นิสิตได้จากโปรแกรมข้างต้น

ผลลัพท์ที่ได้จากโปรแกรมด้านบน นิสิตอาจจะเกิดความสังสัยว่า int ที่เป็น Object กับ int ที่เป็น Datatype ก็ถูกส่งแบบ pass by value เช่นกันแล้วถ้าอยากดำเนินการ pass by reference ในภาษา java จะต้องมีการเพิ่มขั้นตอนเข้ามาดังต่อไปนี้

1. ต้องทำการสร้าง class ใหม่ที่มีการบรรจุ instance variable ไว้
2. สร้าง Constructor Method ที่สามารถกำหนดค่าเริ่มต้นให้กับ instance variable

(Class แบบเบื้องต้น กำหนดให้ ตัวแปร/Method เป็น Public เพื่อง่ายต่อความเข้าใจ)

```
public class Lab46PassByRef
{
	public static void doubleMe1(AddInt x)
	{
		x.value = x.value * 2;
		System.out.printf("In method doubleMe1: x={%d} \n",x.value);
	}
	public static void doubleMe2(int x)
	{
		x = x * 2;
		System.out.printf("In method doubleMe2: x={%d} \n",x);
	}
	public static void main(String[] args)
	{
		AddInt a = new AddInt(10);
		int b = 20;

		System.out.printf("Before methods: a={%d}, b={%d} \n",a.value,b);
		doubleMe1(a);
		doubleMe2(b);
		System.out.printf("After methods: a={%d}, b={%d} \n",a.value,b);
	}
}
class AddInt
{
	public int value;

	// Constructor Method
	public AddInt()
	{
		this(0);
	}
	public AddInt(int value)
	{
		this.value = value;
	}
}
```

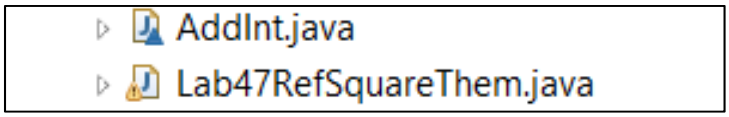
แสดงผลลัพท์ที่นิสิตได้จากโปรแกรมข้างต้น

จะเห็นว่าค่าของตัวแปร a จึงเปลี่ยนไปจากเดิมแต่ค่าของตัวแปร b มีค่าคงเดิม ซึ่งดูคร่าวๆ อาจจะนึกว่า ภาษา java เป็น pass by reference กับ object ให้นิสิตลองเพิ่ม **x = null;** ใน Method doubleMe1 ตามตัวอย่าง

```
public static void doubleMe1(AddInt x)
{
	x.value = x.value * 2;
	System.out.printf("In method doubleMe1: x={%d} \n",x.value);
	x = null;	// ตำแหน่งที่เพิ่มเติม ถ้า pass by ref เมื่อรันควรจะเกิด NullPointerException
}
```

ซึ่งถ้า pass by reference จริง Object a ของ class AddInt ใน Method Main ก็น่าจะเป็น null ด้วยตามทฤษฎี แต่เมื่อลองใส่ดู นิสิตจะพบว่า Object a ไม่ได้มีค่า null ตามที่คิดไว้ ดังนั้นสามารถนำกระบวนการโปรแกรมเช่นนี้ในการพิสูจน์ว่าภาษา java ไม่มี pass by reference แต่ ภาษา java มีกระบวนการเปลี่ยนตำแหน่งของการอ้างอิงด้วยกระบวนอัตโนมัติ แต่ถึงอย่างไรกระบวนการอ้างอิงของภาษา java ก็ยังคงมีการใช้ตำแหน่งอ้างอิงข้อมูลเช่นเดียวกับภาษา OOP ภาษาอื่นๆ เช่น c++ แต่ใน java การอ้างอิงจะต้องอาศัยความรู้ และทักษะ ในการฝึกการเขียนโปรแกรมภาษา java ต่อไป

จากตัวอย่างข้างบน การเขียน class ใน java สามารถกระทำได้โดยการเขียน class file ขึ้นมาใหม่ ในตัวอย่างด้านบนให้นิสิตนำส่วนของ class AddInt มาสร้าง class ไฟล์โดยให้อยู่ใน Folder หรือ Package เดียวกับไฟล์แลปวันนี้



Lab 4.7 ให้นิสิตสร้าง class ว่า Lab47RefSquareThem และให้นิสิตเติมคำในช่องว่างเพื่อออกแบบ Method ชื่อ squareThem โดยรับตัวแปรตัวเลขจำนวนเต็ม สองตัวและเปลี่ยนค่าในตัวแปรนั้นให้เป็นกำลังสองของค่าเริ่มต้น

ตัวอย่างผลลัพท์

```
Enter number#1 : 8
Enter number#2 : 5
Results are {64} and {25}
```

เขียนโคดใน class Lab47RefSquareThem ลงในช่องคำตอบด้านล่าง

```
import java.util.Scanner;

public class Lab47RefSquareThem
{
	public static void squareThem(_____________________________)
	{









	}

	public static void main(String[] args)
	{
		Scanner scan = new Scanner(System.in);

		_____________________________ // ประกาศตัวแปร ชื่อ num1 และสร้าง Object AddInt

		_____________________________ // ประกาศตัวแปร ชื่อ num2 และสร้าง Object AddInt

		System.out.printf("Enter number#1 : ");
		num1.value = scan.nextInt();

		System.out.printf("Enter number#2 : ");
		num2.value = scan.nextInt();

		squareThem(num1, num2);

		System.out.printf("Results are {%d} and
{%d}",num1.value,num2.value);
	}
}
```

1.6 การคืนค่าของเมธอด (Method Returning)

จากโปรแกรมที่ผ่านมาในสัปดาห์นี้ นิสิตได้รู้จัก Method แบบคืนค่าไปบ้างแล้ว โดยหากต้องการเขียน Method แบบคืนค่า ต้องระบุชนิดของข้อมูลที่ Method จะส่งค่ากลับเอาไว้ในส่วน return_type (ลองกลับไปดู Lab 4.2 -4.4 นะครับ) และภายในตัว Method เองจะต้องมีการใช้คำสั่ง return เสมอ แต่สิ่งที่นิสิตพบก็คือการใช้ Method แบบคืนค่านี้สามารถคืนค่าได้เพียงค่าเดียวเท่านั้น หากต้องการเขียน Method ที่มีการคืนค่ามากกว่าหนึ่งค่า นิสิตต้องนำการส่ง Argument โดยใช้การอ้างอิง (Pass by reference) ในลักษณะของภาษา java มาประยุกต์ใช้

Lab 4.8 ให้นิสิตสร้าง class ว่า Lab48RefReadTwoNumber และเขียนเมธอด readTwoInput สำหรับรับข้อมูลเป็นเลขจำนวนเต็ม 2 จำนวนพร้อมเติมคำในช่องว่างของเมธอด Main() ที่กำหนดให้ต่อไปนี้ให้สมบูรณ์เพื่อให้โปรแกรมแสดงผลตามตัวอย่าง

```
Enter number#1 : 10
Enter number#2 : 5
Two numbers are {10} and {5}
```

เขียนโคดใน class Lab48RefReadTwoNumber ลงในช่องคำตอบด้านล่าง

```
import java.util.Scanner;

public class Lab48RefReadTwoNumber
{
	public static void readTwoInput (_____________ , _____________)
	{









	}
	public static void main(String[] args)
	{
		_____________________________ // ประกาศตัวแปร ชื่อ num1 และสร้าง Object AddInt

		_____________________________ // ประกาศตัวแปร ชื่อ num2 และสร้าง Object AddInt

		readTwoInput(___________,___________);

		System.out.printf("Two numbers are {%d} and
{%d}",num1.value,num2.value);

	}
}
```

1.7 การส่งค่าของ Arrays กลับให้ Method ที่เรียก

ค่าที่ส่งกลับจาก Method นอกจากจะเป็นชนิดทั่วๆ ไปเช่น int, double, char, string หรืออื่นๆ แล้ว Method ยังสามารถส่งค่าชนิด Arrays กลับได้อีกด้วย ให้นิสิตศึกษาการส่งค่าชนิด Arrays กลับจากตัวอย่างด้านล่าง

ให้นิสิตสร้าง class ว่า Lab49ArraysReturn และให้ดูผลลัพท์ และทำการศึกษาการทำงานการส่งค่า Arrays

ตัวอย่างผลลัพท์

```
Numbers of data : 3
Please input your data
data[1] = 90
data[2] = 13
data[3] = 39
Your data is   [90]  [13]  [39]
```

เขียนโคดใน class Lab49ArraysReturn แล้วนำไปรันทดสอบดู

```
import java.util.Scanner;

public class Lab49ArraysReturn
{
	static Scanner scan = new Scanner(System.in);

	public static int[] readArrayData(int num)
	{
		System.out.println("Please input your data");
		int[] data = new int[num];

		for(int i = 0 ;  i<data.length ; i++)
		{
			System.out.printf("data[%d] = ", i + 1);
			data[i] = scan.nextInt();
		}

		return data;
	}

	public static void main(String[] args)
	{
		int n;

		System.out.print("Numbers of data : ");
		n = scan.nextInt();

		int[] data;
		data = readArrayData(n);

		System.out.print("Your data is ");

		for(int i = 0 ;  i<data.length ; i++)
		{
			System.out.printf("  [%d]",data[i]);
		}
	}
}
```

Lab 4.9 ให้นิสิตสร้าง class ว่า Lab49MatrixReturn ให้นิสิตเขียนโปรแกรม และเติมคำในช่องว่างให้สมบูรณ์เพื่อให้โปรแกรมด้านล่างทำการถามขนาดของ Matrix จากผู้ใช้และทำการเรียก Method ชื่อ readMatrix เพื่อสร้าง Matrix ตามขนาดที่กำหนดและอ่านข้อมูลของ Matrix มาทีละค่าจากผู้ใช้จากนั้นจึงทำการเรียก Method ชื่อ showMatrix เพื่อแสดง Matrix ออกทางหน้าจอ

ตัวอย่างผลลัพท์

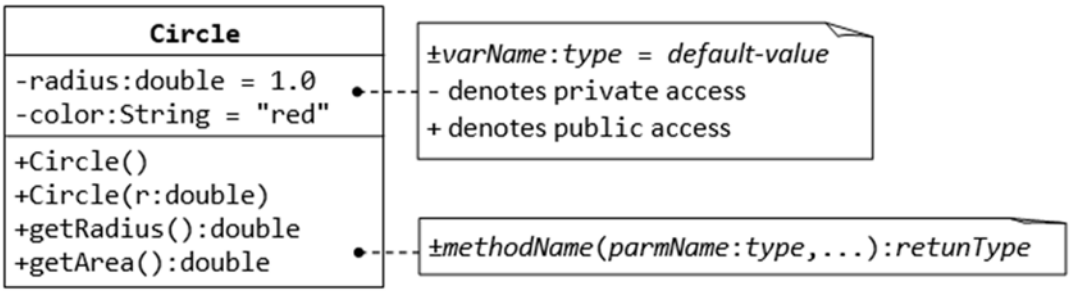
```
How many rows : 2
How many columns : 3
Enter element[1,1]: 9
Enter element[1,2]: 8
Enter element[1,3]: 7
Enter element[2,1]: 6
Enter element[2,2]: 5
Enter element[2,3]: 4
Matrix A is
    9    8    7
    6    5    4
```

เขียนโคดใน class Lab49MatrixReturn ลงในช่องคำตอบด้านล่าง

```
import java.util.Scanner;

public class Lab49MatrixReturn
{
	static Scanner scan = new Scanner(System.in);

	public static __________ readMatrix( ___________, _____________ )
	{
		int[][] matrix = new int[______][________];

		for(int i = 0 ; i < ________ ; i++ )
		{
			for(int j = 0 ; j < ________ ; j++ )
			{
				System.out.printf("Enter element[%d,%d]: ",i+1,j+1);

				_____________  = scan.nextInt();
			}
		}
		return _____________;
	}

	public static _________ showMatrix (_____________,_________,_____________)
	{
		for(int i = 0 ; i < __________; i++)
		{
			for(int j =0 ; j < _____________ ; j++)
			{
				System.out.printf("%5d",__________________);
			}
			System.out.println("");
		}
	}

	public static void main(String[] args)
	{
		int numRows, numCols;
		int[][] A;

		System.out.print("How many rows : ");
		numRows = scan.nextInt();

		System.out.print("How many columns : ");
		numCols = scan.nextInt();

		_____________________________________________

		System.out.println("Matrix A is");

		showMatrix(A,numRows,numCols);
	}
}
```

1.8 Classes and Object

ในเบื้องต้นจะให้นิสิตศึกษาตัวอย่างการทำโปรแกรมเชิงวัตถุจากแผนภาพ class diagram ด้านล่าง



จากรูป class diagram ด้านบนจะแสดงให้เห็น ส่วนประกอบหลักของ Class เชิงวัตถุ แบบเบื้องต้น ประกอบด้วย

1. ชื่อ Class : Circle
2. Instance variable 2 ตัวแปร คือ 2.1 radius 2.2 color ซึ่งกำหนด encapsulation คือ private
3. Method มีจำนวน 4 Method ประเภท public แบ่งเป็น 2 Overloaded constructors + 2 Instance method

Lab 4.10 ให้นิสิตสร้าง class ว่า Circle และทำการเขียนโคดตัวอย่างตามที่แสดงให้ดูด้านล่าง

```
public class Circle
{
	// private instance variable, not accessible from outside this class
	private double radius;
	private String color;

	// The default constructor with no argument.
	public Circle()
	{
		radius = 1.0;
		color = "red";
	}
	// 2nd constructor with given radius, but color default
	public Circle(double r)
	{
		radius = r;
		color = "red";
	}

	// A public method for retrieving the radius
	public double getRadius()
	{
		return radius;
	}
	// A public method for computing the area of circle
	public double getArea()
	{
		return radius*radius*Math.PI;
	}
}
```

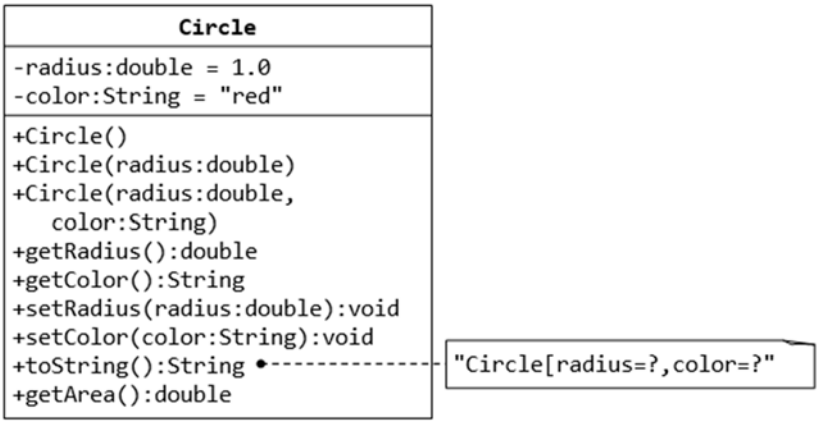
และให้นิสิตเขียน Class ชื่อว่า **CircleTest** ซึ่งมี main method เพื่อทดสอบ class Circle โดยให้มีผลลัพท์ดังนี้

```
The circle has Radius of 1.00
The circle has Area of 3.14

The circle has Radius of 2.00
The circle has Area of 12.57
```

นิสิตทดลองเขียนโปรแกรมใน method main และมาเติมในช่องว่าง

```
public class CircleTest
{
	public static void main(String[] args)
	{
		Circle c1 = _____________________;
		double radius,area;

		radius = _____________________;

		area = _____________________;

		System.out.printf("The circle has Radius of %.2f\n", radius);
		System.out.printf("The circle has Area of %.2f\n\n", area);


		Circle c2 = _____________________;

		radius = _____________________;

		area = _____________________;

		System.out.printf("The circle has Radius of %.2f\n",radius);
		System.out.printf("The circle has Area of %.2f\n",area);
	}
}
```

นิสิตจะสังเกตุว่าโค้ดที่เขียนการออกแบบ class diagram ยังไม่สมบูรณ์ โดยปกติแล้วการออกแบบ class diagram การที่จะกำหนดค่าให้กับ instance variable จะต้องทำ Method ที่เรียกว่า getter/setter เพื่อทำการเปลี่ยนค่า และแสดงค่าของ Object ปรับปรุงการออกแบบ แผนภาพการออกแบบ class diagram ได้ดังนี้



ตัวอย่างการเขียน Method setRadius เพิ่มไปยัง Class Circle ให้นิสิตไปเพิ่มในไฟล์โค้ดของตนเอง

```
public void setRadius(double radius)
{
	radius = radius;
}
```

ให้นิสิตลองเขียน Method setColor ที่จะนำไปเพิ่มใน Class Circle ลงในช่องด้านล่าง

```
public void setColor(_________________)
{


}
```

จาก Setter Method ที่ผ่านมา นิสิตจะสังเกตุว่าชื่อตัวแปรซ้ำซ้อนกัน ซึ่งปกติแล้วนำไปรันอาจจะได้ผลลัพท์ดังที่ต้องการ แต่ภาษา Java มี Keyword “this” เพื่อที่จะระบุให้ว่าตัวนี้คือตัวแปร instance variable เพื่อป้องกันความผิดพลาดให้นิสิตแก้ไขไฟล์ Circle ดังตัวอย่างข้างล่าง ()

```
public class Circle
{
	// private instance variable, not accessible from outside this class
	private double radius;
	private String color;

	// The default constructor with no argument.
	public Circle()
	{
		this(1.0,"red");
	}
	// 2nd constructor with given radius, but color default
	public Circle(double radius)
	{
		this(radius,"red");
	}
	// 3nd constructor with given radius, and color
	public Circle(double radius, String color)
	{
		this.radius = radius;
		this.color = color;
	}

	// A public method for retrieving the radius
	public double getRadius()
	{
		return this.radius;
	}
	public void setRadius(double radius)
	{
		this.radius = radius;
	}

	// A public method for retrieving the radius
	public String getColor()
	{
		return this.color;
	}
	public void setColor(String color)
	{
		this.color = color;
	}

	// Return a description of this instance Circle[radius=r,color=c]
	public String toString()
	{
	   return "Circle[radius=" + this.radius + " color=" + this.color + "]";
	}

	// A public method for computing the area of circle
	public double getArea()
	{
		return this.radius * this.radius * Math.PI;
	}
}
```

ในส่วนของ Method toString() เป็นอีก Method หนึ่งที่นิสิตอาจจะต้องมีการนำไปใช้ในการออกแบบ Class หลักการของ Method นี้คือเมื่อทำการ print แสดงค่า Object ออกทาง Console ค่าของ Object จะแสดงผลตามการออกแบบ String ใน Method toString นั่นเอง ให้นิสิตลอง เพิ่มโค้ดด้านล่าง ไว้บรรทัดสุดท้ายของ Main method แล้วรันดู

```
	System.out.println(c2.toString());
	System.out.println(c2);
```

สุดท้าย ต้องการได้ผลลัพท์ ดังตัวอย่างข้างล่าง

```
Circle[radius=72.0 color=Green]
The circle has Area of 16286.02

Circle[radius=2.0 color=Blue]
The circle has Area of 12.57
```

.ในส่วนที่ต้องแก้ไขคือ class CircleTest โดยจะต้องใช้ Method setRadius และ setColor เพื่อกำหนดค่า Instance variable ของ Object จากนั้นนำมาเติมช่องว่างให้สมบูรณ์

```
public class CircleTest
{
	public static void main(String[] args)
	{









	}
}
```

แบบทดสอบ Lab4

ข้อ1 จงเขียน Class ชื่อ **Grader** สำหรับคำนวณคะแนนในวิชาเรียน ซึ่งมี Method ดังนี้

Method ใน Class Grader

1. **Grader(String name)** เป็น constructor ซึ่งรับชื่อวิชาเป็นอาร์กิวเมนต์
2. **void addScore(double score)** บันทึกคะแนนสำหรับหนึ่งคน
3. **int countStudents()** ซึ่งส่งค่ากลับเป็นจำนวนของคะแนน (หรือจำนวนนักเรียน) ที่ได้บันทึกเข้ามา
4. **double mean()** เพื่อคำนวณค่าเฉลี่ยจากข้อมูลที่รับมาและส่งกลับเป็นผลลัพธ์
5. **int belowMean()** เพื่อหาจำนวนข้อมูลที่ต่ำกว่าค่าเฉลี่ยและส่งกลับเป็นผลลัพธ์
6. **int aboveMean()** เพื่อหาจำนวนข้อมูลที่สูงกว่า หรือเท่ากับ ค่าเฉลี่ยและส่งกลับเป็นผลลัพธ์
7. **String getCourseName()** ซึ่งจะส่งค่ากลับเป็นชื่อวิชาของ Grader ตัวนี้

แอททริบิวต์ (Instance variables) ที่ควรจะมี

1. courseName เป็น String
2. scores เป็น ArrayList

และเขียน Class ชื่อว่า **GraderTest** ซึ่งมี main method เพื่อทดสอบ class Grader โดยสร้าง Grader ขึ้นสองตัว สำหรับสองวิชา และให้แสดงผลดังนี้ (การแสดงผลและรับค่าจากผู้ใช้ทั้งหมดเป็นหน้าที่ของ GraderTest คลาส Grader จะไม่แสดงผลและรับค่าจากคีย์บอร์ดใด ๆ ทั้งสิ้น)

- บรรทัดแรกเป็นชื่อวิชา
- บรรทัดต่อมาเป็นจำนวนเต็ม n
- n บรรทัดถัดมา เป็นจำนวนจริงที่ต้องการหาค่าเฉลี่ย โดยแต่ละบรรทัดเป็นจำนวนจริง 1 ตัว
- แสดงผลค่าเฉลี่ย (ทศนิยมสองตำแหน่ง ระบุชื่อวิชาด้วย) และจำนวนนักเรียนที่ตำกว่าค่าเฉลี่ย และสูงกว่าหรือเท่ากับค่าเฉลี่ย
- ทำแบบนี้สำหรับอีกวิชาด้วย

ตัวอย่างผลลัพท์ เมื่อทำการรันโปรแกรม

```
Enter course name: Python
Enter number of students, followed by scores:
5
12
45
88.5
64
51.5

Mean for Python = 52.20
Below mean = 3
Above mean = 2

Enter course name: Java
Enter number of students, followed by scores:
2
80
75

Mean for Java = 77.50
Below mean = 1
Above mean = 1
```

*** สำหรับแบบฝึกหัด จะต้องส่งทั้ง2ไฟล์คือ 1.**Grader.java** 2.**GraderTest.java** และทำการ comment ในโปรแกรมในส่วนของ statement หลักๆของโค้ดที่นิสิต ถ้าไม่มีการ comment จะถือว่าโค้ดไม่ครบสมบูรณ์