สัญลักษณ์	ความหมาย
+	เป็นการกำหนด access mode เป็น public
-	เป็นการกำหนด access mode เป็น private
#	เป็นการกำหนด access mode เป็น protected
Class	บ่งบอกว่าเป็นคลาสปกติ
AbstractClass*	***ชื่อคลาสเป็นตัวเอียงหนา*** บ่งบอกว่าเป็นคลาสไม่สมบูรณ์
<<Interface>>	บ่งบอกว่าเป็นอินเตอร์เฟส
	บ่งบอกการสืบทอด extends
	บ่งบอกการ implements
	บ่งบอกการเป็นส่วนหนึ่ง composite
methodname() *	*ตัวเอียง* บ่งบอกว่าเป็นเมธอดไม่สมบูรณ์
attribute หรือ **methodname()**	**ตัวหนา** บ่งบอกความเป็น final
attribute หรือ methodname()	ตัวขีดเส้นใต้ บ่งบอกความเป็น static

I. Net-Beans

- (สำคัญ) วิธีการสร้าง Project Java ใน NetBeans
  - Categories --> Java with Ant
  - Projects --> Java Application
  - ตอนรันโค้ดอย่าลืมรันใน psvm!!!

II. การแปลงชนิดข้อมูล

1. เครื่องหมาย "="

```
int a = 10;
double b =
```

2. ตัวดำเนินการ Operator

```
double a = 3.34;
System.out.println("Line 1: " + a)
```

3. วิธีการ Casting

```
double a = 3.14;
int b = (int) a;
```

4. เมธอดจากคลาส Wrapper

```
String text1 = "12345";
Int num = Integer.parseInt(str);
```

int --> String	String.valueof(i);
String --> int	Integer.parseInt("200");
double --> String	String.valueOf(num2);
String --> double	Double.parseDouble("3.14");

--- Object-Oriented Programming ---

III. Encapsulation (การห่อหุ้ม)

- Access Modifier

Public (+)	ทุกคลาส
Private (-)	ภายในคลาสเท่านั้น
Protected (#)	คลาสแม่-ลูก
Default	คลาสใน package เดียวกัน

IV. Inheritance (การสืบทอด) คลาสแม่มี ลูกก็มี

```
public class EVBus extends Bus{}
```

V. Polymorphism (การมีหลายรูปแบบ)

1. **Overloaded** — เป็นการกำหนดให้เมธอด ที่มีชื่อเหมือนกันภายในคลาสเดียวกัน โดยจะมีการทำงาน, อินพุธ, ผลลัพธ์ที่แตกต่างกัน
2. **Overridden** — เมธอดในคลาสลูกมีชื่อเหมือนกับของคลาสแม่ เพื่อให้คลาสลูกเพิ่มเติมการทำงานเฉพาะของตัวเอง
3. **Instanceof** — คือตัวดำเนินการที่ใช้กับอ็อบเจกต์และคลาส เพื่อตรวจสอบว่าเป็นอ็อบเจกต์ของคลาสนั้นหรือไม่ ซึ่งจะให้ผลลัพธเป็นข้อมูลชนิด boolean โดยถ้าผลลัพธ์เป็น true แสดงว่าอ็อบเจกต์เป็นของคลาสนั้นหรือเป็นของคลาสที่คลาสนั้นสืบทอดมา

VI. Constructor — คือ เมธอดที่มีชื่อเหมือนกับคลาส ซึ่งใช้กำหนดค่าเริ่มต้นให้กับแอททริบิวต์หรือการทำงานเบื้องต้น

VII. This / Super

this.*methods()*	ใช้เพื่อเรียกใช้งานแอททริบิวต์หรือเมธอด ภายในคลาส
super.*methods()*	ใช้เพื่อเรียกใช้งานแอททริบิวต์หรือเมธอด ของคลาสแม่
this(...)	เมธอดที่เรียกใช้งานเพื่อเรียกคอนสตรัคเตอร์ ภายในคลาส
super(...)	เมธอดที่เรียกใช้งานเพื่อเรียกคอนสตรัคเตอร์ของคลาสแม่

VIII. Keyword "Final"

Class	คลาสอื่นไม่สามารถสืบทอดคลาสนี้ได้
Method	คลาสอื่นไม่สามารถ overridden ได้
Variable	ค่าคงที่, สามารถกำหนดค่าได้แค่**ครั้งเดียวเท่านั้น**
Attribute	

IX. Abstract Class

คลาสแบบ abstract คือ **คลาสที่ไม่สมบูรณ์** คลาสที่มีเมธอดที่ยังไม่สามารถระบุการกระทำ (เมธอด) ให้ชัดเจนได้

- Abstract Method

```
[modifier] abstract <className>{}
[modifier] abstract <return_type> <methodName>();
```

X. Interface

```
[modifier] interface InterfaceName(){
    [abstract method()];
}
```

มีลักษณะคล้ายกับ abstract class แต่จะประกอบด้วยเมธอดที่ยังไม่สมบูรณ์เท่านั้น กำหนดขึ้นมาเพื่อให้คลาสอื่นนำไปใช้งาน โดยใช้คีย์เวิร์ด implements โดยมีรูปแบบดังนี้

แอททริบิวต์ของอินเตอร์เฟสจะกำหนดให้เป็น *public static final* และเมธอดของอินเตอร์เฟสจะกำหนดให้เป็น *public abstract*

XI. Arrays & Collections

ตัวแปรอาร์เรย์ คือ ตัวแปรที่เป็นชนิด ***ข้อมูลแบบอ้างอิงที่ใช้เก็บข้อมูลชนิดเดียวกันได้หลายค่า*** การประกาศ:

```
String[] s_arr;   or String s_arrp[];
Student[] stds;   or Student stds[];
```

การสร้าง Array :

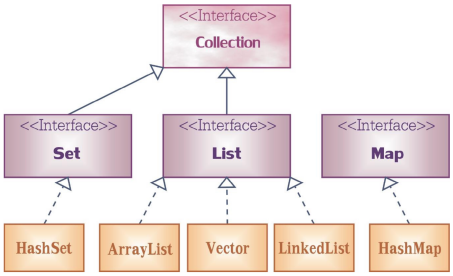
```
"variableName = new datatype[size];"
s_arr = new String[69];
```

การกำหนดค่า :การกำหนดค่าเริ่มต้นกับข้อมูลใน Array:

```
"variableName[index] = value;"
s_arr[15] = "Quentin Tarantino"
```

Collection สามารถนำมาใช้เพื่อเก็บข้อมูลได้หลายตัว

```
"datatype []name = {v1, v2, v3, … ,vn};"
String []a = {"Quentin", "Tarantino"};
```



Generic ใช้กำหนดชนิดของข้อมูลที่ถูกเก็บภายใน Collection นั้นๆ

```
LinkedList<String> myList = new
    LinkedList<String>();
```

XII. Exception

มีการจัดการได้ 2 วิธีคือ try-catch และ throws

1. try-catch

```
try { statement }
catch (ExceptionType name) {statement}
finally { statement }
```

finally จะถูกทำเสมอยกเว้น System.exit(0);

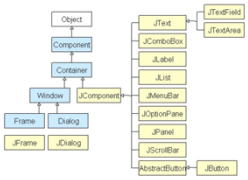
Union ของ try-catch

```
catch (IOException |
NumberFormatException ex ) {statement}
```

2. Throws — เป็นการส่งออปเจค์ประเภท Exception มี 3 แบบคือ Throwing, Rethrowing และ Wrapping

```
public void openFile(String s) throws
    FileNotFoundException, UnknownHostException {
```

XIII. GUI



1.) JFrame — ชั้นนอกสุด

```
JFrame frame = new JFrame("Tilte GUI 02");
JLabel label = new JLabel("Faculty of Information Technology");
frame.add(label);
frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
frame.pack();
frame.setVisible(true);
```

setDefaultCloseOperation : DO_NOTHING, HIDE, DISPOSE, EXIT

2.) JPanel

```
JPanel panel = new JPanel();          panel.add(button1);
panel.setLayout(new BorderLayout());
panel.add(button, BorderLayout.SOUTH);
```

3.) Layout Manager

4.) Component

JButton: A clickable button that can perform an action when clicked.

JLabel: A non-editable text field that can display a single line of text or an image.

JTextField: A text field that allows the user to input and edit text.

JTextArea: A multi-line text field that allows the user to input and edit text.

JCheckBox: A check box that can be selected or deselected.

JRadioButton: A radio button that can be selected among a group of options.

JComboBox: A drop-down list of options that the user can choose from.

JList: A list of items that the user can select from.

JTable: A table that displays data in rows and columns.

JScrollPane: A component that allows the user to scroll through a larger component, such as a text area or table.

JMenuBar: A menu bar that can contain menus, which can contain menu items.

JMenu: A menu that can contain menu items or sub-menus.

JMenuItem: A menu item that performs an action when selected.

JProgressBar: A bar that displays the progress of a task.

JDialog: A dialog box that displays a message or prompts the user for input.

JOptionPane: A dialog box that displays messages and/or prompts the user for input.

XIV. MDI