

Problem Solving

Algorithm

คือกระบวนการการแก้ไขปัญหาย่างเป็นขั้นเป็นตอน

Algorithm

$$10 + -1$$

บวก ลบ คูณ หาร สองเลข ให้ได้เลข 9

$$18 - 9$$

$$1 + 8$$

$$2 + 7$$

$$-9 + 18$$

$$9 * 1$$

$$10 - 9$$

$$4.5 * 2$$

$$9 * 1$$

$$10 - 1$$

Programming 101

บอกวิธีแยก input ว่าเป็นเลขคู่หรือเลขคี่ให้คอมพิวเตอร์ทำงานน้อยลงสิ



ห๊ะ พดอะไร
ฟังไม่ออก



เอา input ที่ป้อนเข้าไปหารด้วยเลข 2
ถ้ามันหารลงตัวก็เป็นเลขคู่
แต่ถ้าหารไม่ลงตัวก็เป็นเลขคี่



Programming 101

ห๊ะ พดอะไร
ฟังไม่ออก

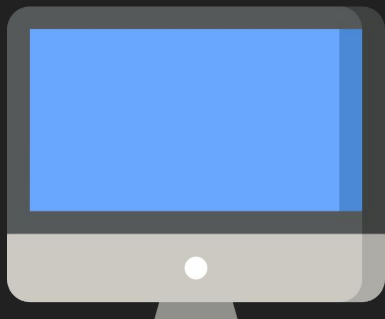


```
num = int(input(""))  
  
if (num % 2) == 0:  
    print("The number is even.")  
else:  
    print("The number is odd.")
```

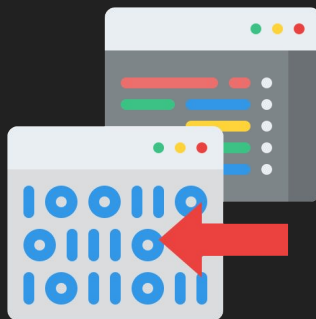


Programming 101

อ้อ เข้าใจแล้ว
รอแป๊บนะ



ไทม์นี่มันบอกว่า
"
100101110110001
100101101111010
00010110011101
"



```
num = int(input(""))  
  
if (num % 2) == 0:  
    print("The number is even.")  
else:  
    print("The number is odd.")
```



How does Java compiler works?

Source Code

Test.java



javac



Byte Code

Test.class



java



Output

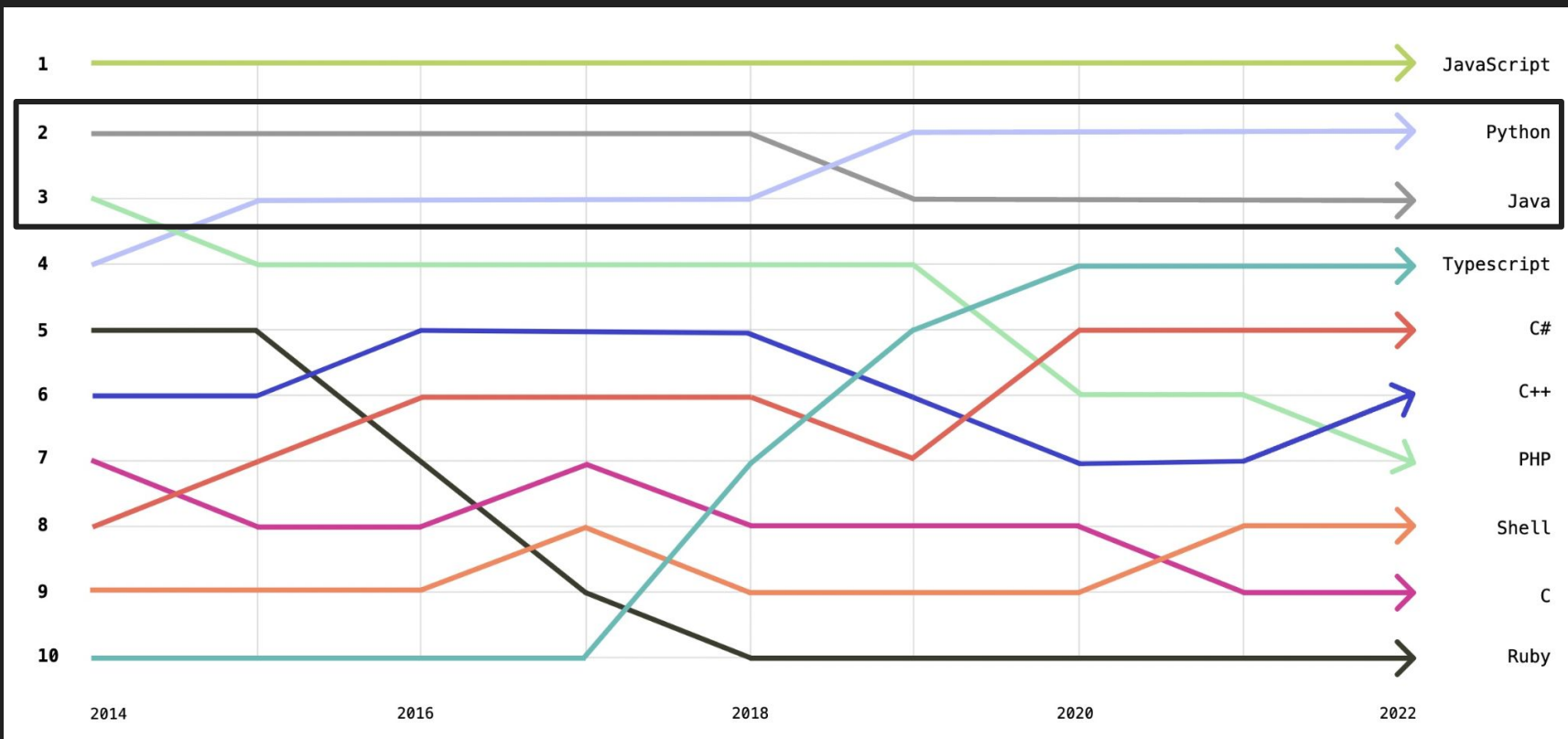
javac Test.java

java Test

Why Java

- เป็นภาษาที่ Syntax คล้ายภาษา C ทำให้มือพื้นฐานที่แน่นกว่า
- Java เป็นภาษาที่เขียนแบบ OOP
- สามารถรันได้ทุก platform, Windows, Linux, Mac
- เป็นภาษาที่นิยมมาตั้งแต่อดีตจนถึงปัจจุบัน

Why Java

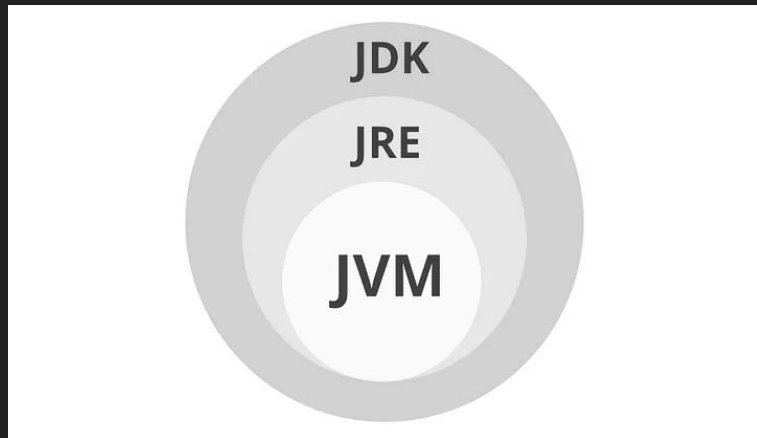


What can you build with java?

- Standalone, command-lines app
- Back-end Applications
- Game
- Mobile App
- Embed System
- etc.

How java work

- **JVM** (Java virtual machine)
เป็นส่วนที่สามารถช่วยให้คอมพิวเตอร์สามารถอ่าน ByteCode ได้ ไม่ว่าจะ platform ใหนก็ตาม
- **JRE** (Java Runtime Environment)
เป็นโปรแกรมที่ไว้รันภาษา Java
- **JDK** (Java Development Kit)
เป็นชุดเครื่องมือที่สำหรับการพัฒนา ที่รวม JRE และ JDK ไว้รวมด้วย



Basic Output

{ System.out.print() System.out.println() System.out.printf() }

System.out.print()

แสดงผลทาง Terminal

Source Code



```
1 System.out.print("Hello Camper!");
```

Output



```
1 Hello Camper!
```

System.out.println()

แสดงผลทาง Terminal แล้ว **enter** เพื่อขึ้นบรรทัดใหม่

Source Code



```
1 System.out.println("Hello Camper!");  
2 System.out.println("This is CS KMITL.");
```

Output



```
1 Hello Camper!  
2 This is CS KMITL.
```

System.out.printf()

แสดงผลผ่านทาง Terminal โดยสามารถใส่ได้หลาย Argument



The diagram shows the code `System.out.printf("Computer, %s", "Science");` with several annotations. A blue circle highlights the `%s` format specifier, with a line pointing to the label "Format Specifier" above it. A bracket under the string `"Computer, %s"` is labeled "Format String". Another bracket under the string `"Science"` is labeled "Argument List".

```
System.out.printf("Computer, %s", "Science");
```

Format Specifier

%s for string

%f for float, double

%c for char

%d for int

%b for boolean

System.out.print()

Source Code & Output



```
1 System.out.print("Hello World!");  
2 System.out.print("Welcome to CS PROF DEV 2024");
```



```
1 Hello World!Welcome to CS PROF DEV 2024
```

System.out.println()

Source Code & Output



```
1 System.out.println("Hello World!");  
2 System.out.println("Welcome to CS PROF DEV 2024");
```



```
1 Hello World!  
2 Welcome to CS PROF DEV 2024
```


Escape Sequence : to display special character that compiler does not allow w/o “\”.

\t : แทรก TAB (whitespace)

\b : แทรก backspace (ลบ)

\n : ขึ้นบรรทัดใหม่

\r : ขึ้นต้นบรรทัดเดิมใหม่

\' : แสดง single quote (')

\” : แสดง double quote (")

\\ : แสดง backslash

Data Type

{ String, char, int, float, boolean }

Data Type

Primitive

- Integers

- byte -short
- int -long

- Floating-Point

- float
- double

- Character

- char

- Boolean

- boolean

Non-primitive

- String

- Array

- List

- Set

- Stack

- ETC.

Primitive

DATA TYPES	SIZE	DEFAULT	EXPLANATION
boolean	1 bit	false	Stores true or false values
byte	1 byte/ 8bits	0	Stores whole numbers from -128 to 127
short	2 bytes/ 16bits	0	Stores whole numbers from -32,768 to 32,767
int	4 bytes/ 32bits	0	Stores whole numbers from -2,147,483,648 to 2,147,483,647
long	8 bytes/ 64bits	0L	Stores whole numbers from -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
float	4 bytes/ 32bits	0.0f	Stores fractional numbers. Sufficient for storing 6 to 7 decimal digits
double	8 bytes/ 64bits	0.0d	Stores fractional numbers. Sufficient for storing 15 decimal digits
char	2 bytes/ 16bits	'\u0000'	Stores a single character/letter or ASCII values

Non-primitive

String text, letter or digit that contain in “ ”.

Variable

{ rules of variables, reserved word, variable declaration, }

Rules of Variables

1. can contain **english** letter, **digit**, **underscore**(_), **dollar sign**(\$)
2. **must** start with **letter** (can not start with digit)
3. **can not** contain with **white space**
4. case-sensitive (uppercase & lowercase)
5. reserved word

Reserved Word

abstract	assert	boolean	break	byte
case	catch	char	class	const
continue	default	do	double	else
enum	extends	final	finally	float
for	goto	if	implements	import
instanceof	int	interface	long	native
new	non-sealed	package	private	protected
public	return	short	strictfp	super
super	switch	synchronized	this	throw
throws	transient	try	void	volatile
while				

Variable Declaration

```
data_type variable_name;
```

```
String myString;
```

```
int myInteger;
```

```
char myCharacter;
```

```
boolean myBoolean;
```

```
float myFloat;
```

Variable Initialization

var1 / var2,
123 + 456

1, "Hello", 25

↑ ↑

data_type variable_name = expression or literal;

String myString = "CS PROF DEV";

int myInteger = 2024;

char myCharacter = 'A';

boolean myBoolean = true;

float myFloat = 6.75;

Variable Initialization (2)

Byte : `byte` myByte = 99;

Short : `short` myShort = 999;

Int : `int` myInt = 9999;

Long : `long` myLong = 99999L;

Float : `float` myFloat = 123.456f;

Double : `double` myDouble = 789.012d;

Variable Initialization (3)

Char : `char myChar = 'A';` or `char myChar = 99;`

ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	\$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[END OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

What would happen
if a variable wasn't declared?



```
1 int myInput;  
2 myInput = 2+2;
```



```
1 myInput = 2+2;
```





```
1 public class varDeclaration {  
2     public static void main(String[] args) {  
3         myInput = 2+2;  
4     }  
5 }
```


Error :

Exception in thread "main" java.lang.Error: Unresolved compilation problem:
myInput cannot be resolved to a variable


at varDeclaration.main(varDeclaration.java:3)

Type Casting : to assign a value of one data type to others (Only Primitive)

Widening Casting



```
1  int myInt = 2024;  
2  double myDouble = myInt;  
3  
4  System.out.println(myInt);  
5  System.out.println(myDouble);
```




```
1  myInt = 2024  
2  myDouble = 2024.0
```

smaller type to larger type :

byte -> short -> char -> int -> long -> float -> double

Narrowing Casting



```
1  double myDouble = 123.456d;  
2  
3  System.out.println("myInt = " + (int)myDouble);
```



```
1  myInt = 123
```

larger type to smaller type :

double -> float -> long -> int -> char -> short -> byte

Operator

{ Arithmetic, Assignment, Comparison, Logical }

Arithmetic Operators

+ : เครื่องหมายบวก

* : เครื่องหมายคูณ

% : เครื่องหมายหารเอาเศษ

- : เครื่องหมายลบ

/ : เครื่องหมายหาร

Modulo(%)

หารเพื่อเอาเศษที่ได้จากการหาร
เช่น $5/2 = 2$ เศษ 1 ดังนั้น $5\%2 = 1$

วิธีการคิดเศษ :

เศษ = ตัวตั้ง - (ตัวหาร*ผลหาร)



```
1  int x = 10;  
2  
3  System.out.println("x % 2 = " + x%2);  
4  System.out.println("x % 3 = " + x%3);
```



```
1  x % 2 = 0  
2  x % 3 = 1
```

Assignment Operators

=	:	ให้ค่า		*=	:	คูณค่า	:	$x = x * _$		
+=	:	เพิ่มค่า	:	$x = x + _$		/=	:	หารค่า	:	$x = x / _$
-=	:	ลดค่า	:	$x = x - _$		%=	:	หารเอาเศษด้วย	:	$x = x \% _$



```
1  int x = 10;  
2  
3  System.out.println("x = "+x);  
4  x+=2;  
5  System.out.println("x+=2 = "+x);  
6  x-=1;  
7  System.out.println("x-=1 = "+x);  
8  x*=4;  
9  System.out.println("x*=4 = "+x);  
10 x/=2;  
11 System.out.println("x/=2 = "+x);  
12 x%=8;  
13 System.out.println("x%=8 = "+x);
```



```
1  x = 10  
2  x+=2 = 12  
3  x-=1 = 11  
4  x*=4 = 44  
5  x/=2 = 22  
6  x%=8 = 6
```

Increment & Decrement

++	:	เพิ่มค่าตัวแปรขึ้น 1	:	x+=1	:	x = x+1
--	:	ลดค่าตัวแปรลง 1	:	x-=1	:	x = x-1



```
1  int x = 5;
2
3  x--;
4  x++;
5  x--;
6  x++;
7  x--;
8  x--;
9  x--;
10 x++;
11 System.out.println("Now, x is "+x);
```



1 Now, x is 3

ข้อควรระวังในการใช้ Assignment Operators และ Increment/Decrement

ทั้ง Assignment Operators และ Increment Operator ทำหน้าที่เพิ่มค่าเหมือนกับ Arithmetic Operators และค่านั้นจะคงอยู่เป็นค่าปัจจุบันจนกว่าจะมีการดำเนินการกับตัวแปรนั้น ๆ อีกครั้ง

Increment & Decrement ไม่มีผลทันทีเมื่อใช้ใน print



```
1  int x = 5;  
2  
3  x--;  
4  x++;  
5  x--;  
6  x++;  
7  x--;  
8  x--;  
9  x--;  
10 x++;  
11 System.out.println("Now, x is "+(x++));  
12 System.out.println("Then, x is "+x);
```



```
1  Now, x is 3  
2  Then, x is 4
```

Quick Lab#1

find the value of final
"X"
(line 14)



```
1  int x = 24;
2
3  x--;
4  x++;
5  x+=10;
6  x*=0;
7  x-- ;
8  x+=100;
9  System.out.println("line 11 is "+x++);
10 System.out.println("line 12 is "+x--);
11 x%=11;
12 System.out.println("line 14 is "+x);
13 System.out.println("line 15 is "+ (x+=24));
14 System.out.println("line 16 is "+x);
```

Quick Lab#1

Answer :
24



```
1  line 11 is 99
2  line 12 is 100
3  line 14 is 0
4  line 15 is 24
5  line 16 is 24
```

Comparison Operators

`==` : (มีค่า)เท่ากัน

`!=` : ไม่เท่ากัน

`>` : มากกว่า

`<` : น้อยกว่า

`>=` : มากกว่าหรือเท่ากับ

`<=` : น้อยกว่าหรือเท่ากับ



```
1 System.out.println("x == y is "+(x==y));
2 System.out.println("x != y is "+(x!=y));
3 System.out.println("x > y is "+(x>y));
4 System.out.println("x < y is "+(x<y));
5 System.out.println("x >= y is "+(x>=y));
6 System.out.println("x <= y is "+(x<=y));
```



```
1 x == y is false
2 x != y is true
3 x > y is false
4 x < y is true
5 x >= y is false
6 x <= y is true
```

Logical Operators

&& : และ

|| : หรือ

! : นิเสธ (ไม่/ตรงข้าม)



```
1 int x = 10;  
2 int y = 23;  
3  
4 System.out.println("x > y && x >= y is "+((x>y)&&(x>=y)));  
5 System.out.println("x == y || x !=y is "+((x==y)||(x!=y)));  
6 System.out.println("!=(x > y && x >= y) is "+(!(x>y)&&(x>=y)));
```



```
1 x > y && x >= y is false  
2 x == y || x !=y is true  
3 !=(x > y && x >= y) is true
```


Why do we need to select
the Data Type appropriately?

What is the **mistake** of this source code?

Source Code



```
1 byte myByte = 127;  
2  
3 System.out.println(myByte+=3);
```

Output



```
1 -126
```

Overflow


Value of the variable is out of range.

byte	1 byte/ 8bits	0	Stores whole numbers from -128 to 127
short	2 bytes/ 16bits	0	Stores whole numbers from -32,768 to 32,767



```
1 byte myByte = 127;  
2  
3 System.out.println(myByte+=3);
```

overflow



1 -126

A red arrow points from the value -126 to the word 'overflow' above it.

LAB#1

1. ให้เขียนโปรแกรม **ประกาศตัวแปร** และ **กำหนดตัวเลข** 3,000,000,000 **พิมพ์ค่า**ตัวเลขออกมาผ่านทางหน้าจอ
2. ให้**ประกาศตัวแปรชนิดข้อมูล char** 6 ตัว และ ให้**กำหนดค่าเป็นตัวเลข** และ**พิมพ์ค่า**ผ่านทางหน้าจอ เป็นคำว่า "ComSci"
3. **ประกาศตัวแปร**มาสองตัว โดยตัวแรกเป็น **char** โดยตัวแรกมีค่าเป็น 'X' และตัวที่สองมีชนิดข้อมูลเป็น **int** โดยที่ให้นำตัวแปรแรก บวก ตัวแปรที่สอง ให้ได้ผลลัพธ์ออกมาเป็น 90
4. ให้**ประกาศตัวแปรที่กำหนดค่าเป็น 9! และ 18!** โดยห้ามใช้ **loop** และ**พิมพ์ผ่าน** **ทางหน้าจอ**

Input

Let's try some another class

Let's try some another class

That is the **Scanner**

What is The Scanner?

Scanner คือ Class ที่มีอยู่ใน Java

What does Scanner Class can do?

Scanner Class สามารถรับค่าผ่านทาง keyboard

Let's use Scanner

First import Scanner into your file

```
import java.util.Scanner;
```

Create a Object Scanner

```
Scanner sc = new Scanner(System.in);
```

Create a Object Scanner

Sample

```
1 import java.util.Scanner;
2
3 class readInput {
4     public static void main(String[] ComSci) {
5         Scanner sc = new Scanner(System.in);
6     }
7 }
```

Use methods of the Scanner class

```
1 import java.util.Scanner;
2
3 class readInput {
4     public static void main(String[] ComSci) {
5         Scanner sc = new Scanner(System.in);
6         byte Byte = sc.nextByte();
7         short Short = sc.nextShort();
8         int Int = sc.nextInt();
9         long Long = sc.nextLong();
10        float Float = sc.nextFloat();
11        double Double = sc.nextDouble();
12        boolean Bool = sc.nextBoolean();
13        String Line = sc.nextLine();
14    }
15 }
```

Each methods of the Scanner can do

Method	Description
<code>nextBoolean()</code>	Reads a <code>boolean</code> value from the user
<code>nextByte()</code>	Reads a <code>byte</code> value from the user
<code>nextDouble()</code>	Reads a <code>double</code> value from the user
<code>nextFloat()</code>	Reads a <code>float</code> value from the user
<code>nextInt()</code>	Reads a <code>int</code> value from the user
<code>nextLine()</code>	Reads a <code>String</code> value from the user
<code>nextLong()</code>	Reads a <code>long</code> value from the user
<code>nextShort()</code>	Reads a <code>short</code> value from the user

Actually has more methods of Scanner

Just said The Basic

Condition

ถ้า ฉันยังมีชีวิต ฉันเป็นเด็ก ComSci KMITL

Why Condition

เราจะกรองเงื่อนไขที่เราไม่ยากได้อย่างไร?
ดังนั้นเราจึงต้องมี Condition

About Condition

It's Check Conditions

- 1. if statement**
- 2. switch statement**

If Statement

if...
if...elseif...
else if... else...

if Statement

```
if (condition) {  
    Statement;  
    Statement;  
    Statement;  
    ...  
}
```

Description

ถ้าใน () เป็นจริง จะทำใน {...} ต่อ

Ex.1 If Statement

```
int a = 10;  
int b = 11;
```

```
if (a < b) {  
    System.out.println("3 less than 5");  
}
```

Output

3 less than 5

if - else Statement

```
if (condition) {  
    Statement;  
    ...  
} else {  
    Statement;  
    ...  
}
```

Description

ถ้าใน () เป็นจริง จะทำใน {...} ต่อ
แต่ถ้าไม่เป็นจริงใดๆ จะทำใน {...}

Ex.2 If Statement

```
int a = 11;  
int b = 10;
```

```
if (a > b) {  
    System.out.println("a greater than b");  
} else {  
    System.out.println("a less than b");  
}
```

Output

a greater than b

Ex.3 If Statement

```
int a = 5;  
int b = 11;  
  
if (a > b) {  
    System.out.println("a greater than b");  
} else {  
    System.out.println("a less than b");  
}
```

Output

a less than b

if - else if - else Statement

```
if (condition) {  
    Statement;  
    ...  
} else if (condition) {  
    Statement;  
    ...  
} else {  
    Statement;  
    ...  
}
```

Description

ถ้าใน () เป็นจริง จะทำใน {...} ต่อ

แต่ถ้าไม่เป็นจริง ให้เช็คใน () ถ้าเป็นจริง จะทำใน {...} ต่อ

แต่ถ้าไม่เป็นจริงใดๆ จะทำใน {...}

Ex.4 If Statement

```
int a = 3;  
int b = 3;  
  
if (a == b) {  
    System.out.println("a equal b");  
} else if (a > b) {  
    System.out.println("a greater than b");  
} else {  
    System.out.println("a less than b");  
}
```

Output

a equal b

Ex.5 If Statement

```
int a = 9;  
int b = 3;  
  
if (a == b) {  
    System.out.println("a equal b");  
} else if (a > b) {  
    System.out.println("a greater than b");  
} else {  
    System.out.println("a less than b");  
}
```

Output

a greater than b

Ex.6 If Statement

```
int a = 6;  
int b = 9;  
  
if (a == b) {  
    System.out.println("a equal b");  
} else if (a > b) {  
    System.out.println("a greater than b");  
} else {  
    System.out.println("a less than b");  
}
```

Output

a less than b

Nested condition

```
if (condition) {  
    Statement;  
    ...  
    if (condition) {  
        Statement;  
        ...  
    }  
}
```

Description

ถ้า () เป็นจริง ให้ทำใน
{
 ถ้า () เป็นจริง ให้ทำใน { ... } ต่อ
}

Ex.8 Nested condition

```
int a = 9;
```

```
int b = 9;
```

```
if (a == b) {  
    System.out.println("a equal b");  
    if (a % 2 == 0) {  
        System.out.println("a is even");  
    } else {  
        System.out.println("a is odd");  
    }  
}
```

Output

a equal b

Ex.9 Nested condition

```
int a = 9;  
int b = 9;  
  
if (a == b) {  
    System.out.println("a equal b");  
    if (a % 2 == 0) {  
        System.out.println("a is even");  
    } else {  
        System.out.println("a is odd");  
    }  
}
```

Output

a equal b
a is odd

Ex.10 Nested condition

```
int a = 4;  
int b = 4;  
  
if (a == b) {  
    System.out.println("a equal b");  
    if (a % 2 == 0) {  
        System.out.println("a is even");  
    } else {  
        System.out.println("a is odd");  
    }  
}
```

Output

a equal b
a is even

Statement and Expression

ต่างกันไ้?

Statement and Expression

Statement

คือการกระทำของโปรแกรมเช่น

- `system.out.println("Hello");`
- `if (condition){...}`
- `int x = 5;`
- `int sum = 2 + 3;`
- `scan.nextInt();`

Expression

คือการกระทำที่เกี่ยวข้องกับการคำนวณค่า หรือ การกำหนดค่าเท่านั้น เช่น

- `2 + 3 * 4`
- `(2 == 3)`
- `(1 == 1 || 2 > 3);`

Ternary Operator

```
if (condition) {  
    Statement;  
    ...  
} else {  
    Statement;  
    ...  
}
```

```
////////////////////////////////////  
condition ? expression1 : expression2;
```

Description

ในกรณีที่เราทำ if-else

condition คือ เงื่อนไข

เป็นจริง ให้ทำ

expression1

เป็นเท็จ ให้ทำ

expression2

Ternary Operator

```
if (condition) {  
    Statement;  
}  
else if {  
    Statement;  
}  
else {  
    Statement;  
}
```

Description

ในกรณีที่เรากำ if - else if - else

ถ้า condition1 เป็นจริง expression1 ทำงาน

ถ้า condition1 เป็นเท็จ จะเข้า condition2
condition2 เป็นจริง expression2 ทำงาน
condition2 เป็นจริง expression3 ทำงาน

////////////////////////////////////

condition1 ? expression1 : condition2 ? expression2 : expression3 ;

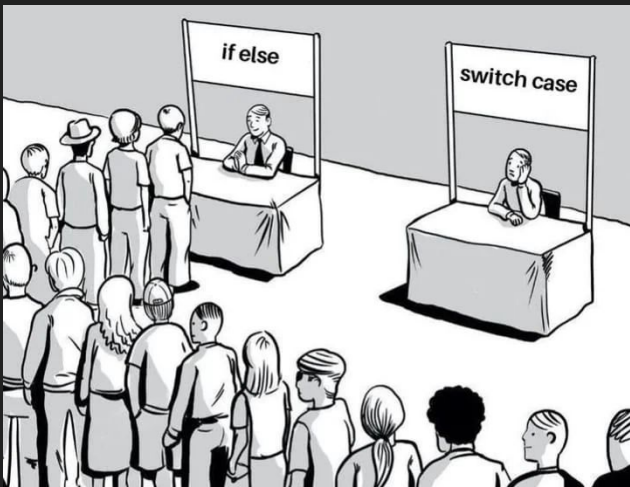


switch
case



if
else

Switch Statement



ternary
operator



if / else



switch case

Switch Statement

```
switch (expression) {  
    case x:  
        statement;  
        ...  
        break;  
    default:  
        statement;  
        ...  
        break;  
}
```

Description

expression หรือ ตัวที่เราต้องการจะทำให้การเช็ค

x คือค่าที่เราต้องการเช็คด้านใน **expression**

ว่าตรงกันหรือไม่

break; เมื่อทำทุกอย่างเสร็จแล้วให้ทำการออกจาก **switch case** นี้

Ex.11 Switch Statement

```
int day = 2;  
switch (day) {  
    case 1:  
        System.out.println("monday");  
        break;  
    case 2:  
        System.out.println("tuesday");  
        break;  
    default:  
        System.out.println("วันที่เธอรักเรา");  
        break;  
}
```

Output

tuesday

Ex.12 Switch Statement

```
int day = 2;  
switch (day) {  
    case 1:  
        System.out.println("monday");  
        break;  
    case 2:  
        System.out.println("tuesday");  
    default:  
        System.out.println("วันที่เรอรักเรา");  
        break;  
}
```

Output

tuesday
วันที่เรอรักเรา

Ex.13 Switch Statement

```
int day = 3;  
switch (day) {  
    case 1:  
        System.out.println("monday");  
        break;  
    case 2:  
        System.out.println("tuesday");  
        break;  
    default:  
        System.out.println("วันที่เรอรักเรา");  
        break;  
}
```

Output

วันที่เรอรักเรา

Ex.14 If else and Switch

If else

```
1 int score = 85; // เปลี่ยนค่าคะแนนตามต้องการ
2 char grade;
3
4 if (score >= 90) {
5     grade = 'A';
6 } else if (score >= 80) {
7     grade = 'B';
8 } else if (score >= 70) {
9     grade = 'C';
10 } else if (score >= 60) {
11     grade = 'D';
12 } else {
13     grade = 'F';
14 }
15
16 System.out.println("Your grade is: " + grade);
```

Switch

```
1 int score = 85;
2 char grade;
3
4 switch (score / 10) {
5     case 10:
6     case 9:
7         grade = 'A';
8         break;
9     case 8:
10        grade = 'B';
11        break;
12    case 7:
13        grade = 'C';
14        break;
15    case 6:
16        grade = 'D';
17        break;
18    default:
19        grade = 'F';
20        break;
21 }
22
23 System.out.println("Your grade is: " + grade);
```

Ex.15 If else and Switch

If else

```
1 int year = 2024;
2 boolean isLeapYear;
3
4 if (year % 4 == 0) {
5     if (year % 100 == 0) {
6         if (year % 400 == 0) {
7             isLeapYear = true;
8         } else {
9             isLeapYear = false;
10        }
11    } else {
12        isLeapYear = true;
13    }
14 } else {
15     isLeapYear = false;
16 }
17
18 if (isLeapYear) {
19     System.out.println(year + " is a leap year.");
20 } else {
21     System.out.println(year + " is not a leap year.");
22 }
```

Switch

```
1 int year = 2024;
2 boolean isLeapYear = false;
3
4 switch (year % 4) {
5     case 0:
6         switch (year % 100) {
7             case 0:
8                 switch (year % 400) {
9                     case 0:
10                        isLeapYear = true;
11                        break;
12                    default:
13                        isLeapYear = false;
14                        break;
15                }
16                break;
17            default:
18                isLeapYear = true;
19                break;
20        }
21        break;
22    default:
23        isLeapYear = false;
24        break;
25 }
26
27 if (isLeapYear) {
28     System.out.println(year + " is a leap year.");
29 } else {
30     System.out.println(year + " is not a leap year.");
31 }
```

If else and Switch

If else and Switch

การใช้ if else และ switch ไม่ได้มีเงื่อนไขตายตัวว่าควรใช้อันไหน

If else and Switch

การใช้ if else และ switch ไม่ได้มีเงื่อนไขตายตัวว่าควรใช้อันไหน

ตัวไหนทำให้ code อ่านได้ง่ายกว่า และ เหมาะสมต่อการใช้งาน (ไม่ทำให้ condition ซ้ำซ้อนเกินความจำเป็น) = สามารถใช้ได้



Iteration

เข้ามาเพื่อต่อรอง

What is “Iteration”

การวนซ้ำเรื่อยๆ จนกว่าเราจะให้หยุด

Why Iteration

```
System.out.println("ComSci");  
System.out.println("ComSci");  
System.out.println("ComSci");  
System.out.println("ComSci");  
System.out.println("ComSci");
```

Output

```
ComSci  
ComSci  
ComSci  
ComSci  
ComSci
```

Description

ถ้าหากเราต้องการพิมพ์คำว่า

"ComSci" 5 ครั้ง

ดูเยอะใช่ไหม

Do some Iteration

```
for (int i = 0; i < 5; i++) {  
    System.out.println("ComSci");  
}
```

Output

```
ComSci  
ComSci  
ComSci  
ComSci  
ComSci
```

Description

ยกตัวอย่างการลองใช้ **Iteration**

เพื่อพิมพ์คำว่า "ComSci" 5 ครั้ง

ดูสั้นกว่า และ สบายกว่า

How many Iteration

For loop
While loop
Do while loop

For loop

```
for (initialize; condition; expression) {  
    Statement;  
    ...  
}
```

Description

Initialize ค่าเริ่มต้นของ loop นี้

Condition หากเงื่อนไขเป็นจริงจะทำ
{...} ต่อ

expression ทำการเพิ่ม หรือ ลดค่า
ของ **initialize** เพื่อให้ทำตามจำนวน
ครั้งที่กำหนดไว้ และไม่ให้เป็น
infinity loop

Ex.16 For loop

```
for (int i = 0; i < 10; i++) {  
    System.out.println(i);  
}
```

Output

0
1
2
3
4
5
6
7
8
9

Ex.16.1 For loop

```
for (int i = 0; i < 10; i++) {  
    System.out.println(i);  
}
```

Description

เมื่อเริ่มใช้คำสั่ง for loop สิ่งที่จะเกิดขึ้นสิ่งแรกคือ กำหนดค่าเริ่มต้นให้

Ex.16.2 For loop

```
for (int i = 0; i < 10; i++) {  
    System.out.println(i);  
}
```

Description

ทำการเช็คเงื่อนไข ว่า $i < 10$ ไหม
หากเป็นจริง ทำ Statement ใน {...}
ต่อ

Output

1

Ex.16.3 For loop

```
for (int i = 0; i < 10; i++) {  
    System.out.println(i);  
}
```

Description

เมื่อทำ Statement ใน {...} เสร็จแล้ว
ให้ทำการบวก i เพิ่มขึ้น 1

Output

1

While loop

```
while (condition) {  
    Statement;  
    ...  
}
```

Description

เมื่อทำ Statement ใน {...} เสร็จแล้ว
ให้ทำการบวก i เพิ่มขึ้น 1

Ex.17 While loop

```
int i = 0;  
while (i < 5) {  
    System.out.println(i);  
    i++;  
}
```

Output

0

Description

ทำการเช็คเงื่อนไข ($i < 5$) ถ้าเป็นจริง
ให้ทำการแสดง ตัวเลข i

Ex.17.1 While loop

```
int i = 0;  
  
while (i < 5) {  
    System.out.println(i);  
    i++;  
}
```

Output

0

Description

i เพิ่มค่าขึ้น 1

Ex.17.2 While loop

```
int i = 0;

while (i < 5) {
    System.out.println(i);
    i++;
}
```

Output

```
0
1
2
3
4
```

Description

เมื่อทำจนจบ ผลลัพธ์ที่ได้คือ 5

Do While loop

```
do {  
    Statement;  
    ...  
}  
while (condition);
```

Description

ให้ทำ Statement {...} ก่อน

แล้วเช็คเงื่อนไขใน (...) หากเป็นจริง
กลับไปทำ Statement {...}

Ex.18 Do While loop

```
int i = 0;

do {
    System.out.println(i);
    i++;
}
while (i < 5);
```

Output

0
1
2
3
4

Ex.19 Do While loop

```
int i = 0;
```

```
do {  
    System.out.println(i);  
    i++;  
}  
while (i < 5);
```

Output

```
0  
1  
2  
3  
4
```

```
int i = 0;
```

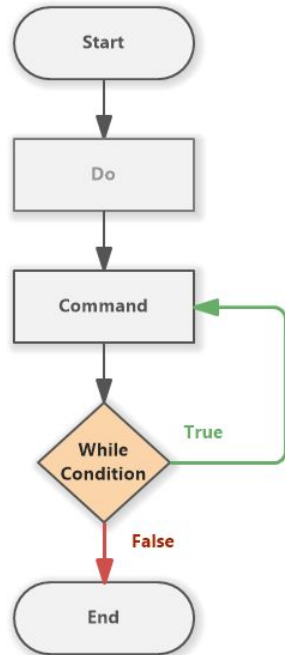
```
while (i < 5) {  
    System.out.println(i);  
    i++;  
}
```

Output

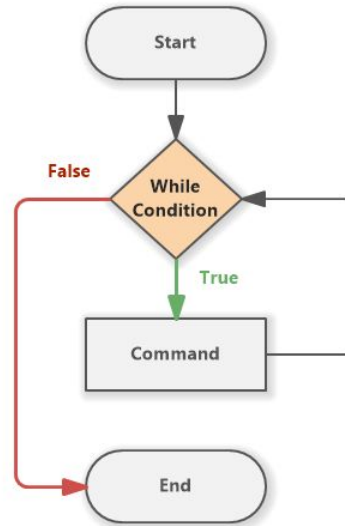
```
0  
1  
2  
3  
4
```


Ex.19 Do While loop

DO-WHILE



WHILE



Did you know
Block Statement

Block Statement

```
for (condition) {  
    Statement;  
    ...  
}
```

Description

คือสิ่งที่ถูกจัดกลุ่มกัน โดยจะถูกกำหนดด้วย { เพื่อเริ่มต้น Block Statement และ จะจบเมื่อเจอ } เพื่อบอกว่าจบ Block Statement นั้นๆ

Ex.20 Block Statement

```
for (int i = 0; i < 3; i++) {  
    System.out.println(i);  
    System.out.println("Block");  
}
```

Output

```
0  
Block  
1  
Block  
2  
Block
```

Block Statement

```
for (condition)  
    Statement;
```

Description

หากไม่มี `{ }` จะทำเพียงแค่
Statement เดียวเท่านั้น

Ex.21 Block Statement

```
for (int i = 0; i < 3; i++)  
    System.out.println(i);  
    System.out.println("Block");
```

Output

```
0  
1  
2  
Block
```

Block Statement with Declare variable

Ex.22 Block Statement

```
if (true) {  
    int number = 2;  
}
```

```
System.out.println(number);
```

Output

error: cannot find symbol

Description

หากประกาศตัวแปรใน `{ }` หากมีการเรียกใช้ตัวแปรนั้นนอก Block Statement ระบบจะหาตัวแปรไม่เจอ

Why?

Why?

อะไรก็ตามที่ถูกประกาศภายใต้ `{ }` จะไม่สามารถเรียกใช้ภายนอกของ `{ }` ได้

Ex.23 Block Statement

```
public static void main(String[] args){  
    int number = 2;  
    if (true) {  
        System.out.println(number);  
    }  
}
```

Output

2

Description

แต่หากเราประกาศภายนอก `{ }` แต่เราเรียกใช้ภายใน `{ }` จะไม่เกิด error ใดๆ สามารถเรียกใช้ได้ปกติ

เพราะการเรียกใช้อยู่ภายใต้ `{ }` ของ main เหมือนกัน

Ex.24 Block Statement

```
public static void main(String[] args) {  
    int number = 2;  
    if (true) {  
        number += 10;  
    }  
    System.out.println(number);  
}
```

Output

12

Description

เช่นเดียวกับการทำ Expression ด้านใน `{ }` แต่ก็ยังสามารถเรียกใช้ภายนอก `{ }` ได้

เพราะว่า เราได้ทำการ **declare** **number** ภายนอก `{ }` ไว้ก่อนแล้ว จึงสามารถทำ **expression** ภายใน `{ }` และ เรียกใช้ **number** ได้ โดยไม่เกิด error หรือ ค่าผิดพลาดอะไร

Iteration is easy, isn't it

Iteration is easy, isn't it

Condition have Nested Condition

Iteration is easy, isn't it

Condition have Nested Condition
Why Iteration doesn't have nested Iteration?

Nested For Loop

```
for (initialize; condition; expression) {  
    for (initialize; condition; expression) {  
        Statement;  
        ...  
    }  
}
```

Description

ถ้า **condition** เป็นจริง ให้ทำใน
{
 ถ้า **condition** เป็นจริง ให้ทำใน
 {...} ต่อ
}

Nested While

```
while (condition) {  
    while (condition) {  
        Statement;  
        ...  
    }  
}
```

Description

ถ้า **condition** เป็นจริง ให้ทำใน
{
 ถ้า **condition** เป็นจริง ให้ทำใน
 {...} ต่อ
}

Nested For loop - While

```
for (initialize; condition; expression) {  
    while (condition) {  
        Statement;  
        ...  
    }  
}
```

Description

ถ้า **condition** เป็นจริง ให้ทำใน
{
 ถ้า **condition** เป็นจริง ให้ทำใน
 {...} ต่อ
}

Nested While - For loop

```
while (condition) {  
    for (initialize; condition; expression) {  
        Statement;  
        ...  
    }  
}
```

Description

ถ้า **condition** เป็นจริง ให้ทำใน
{
 ถ้า **condition** เป็นจริง ให้ทำใน
 {...} ต่อ
}

Apply

For loop - If

```
for (int i = 1; i <= 100; i++) {  
    if (i % 2 == 0) {  
        System.out.println(i);  
    }  
}
```

Output

2
4
6
...
100

Description

เขียนโปรแกรมหาเลขคู่ระหว่าง 1 - 100

While - If

```
int num = 6;  
int i = 1;  
while (i < num) {  
    if (num % i == 0)  
        System.out.println(i);  
    i++;  
}
```

Output

```
1  
2  
3
```

Description

โปรแกรมหาจำนวนสมบูรณ์ โดยการประยุกต์การใช้ while loop และ if

Nested Iteration

```
for (int i = 1; i <= 12; i++) {  
    for (int j = 1; j <= 12; j++) {  
        System.out.printf("%d x %d = %d\n", i, j, (i * j));  
    }  
}
```

Output

```
1 x 1 = 1  
1 x 2 = 2  
...  
12 x 11 = 132  
12 x 12 = 144
```

Description

โปรแกรมทำตารางแม่สูตรคูณอย่างง่าย

Do while - Scanner

```
Scanner sc = new Scanner(System.in);
int sum = 0;
int number;
do {
    number = sc.nextInt();
    sum += number;
} while (number != 0);

System.out.print(sum);
sc.close();
```

Description

โปรแกรมหาผลรวมของเลขที่พิมพ์เข้ามา

Output

10
12
0
22

What is a lucky number?