

CS camp

ONLINE DAY 1

Sponsored by



OUTLINE

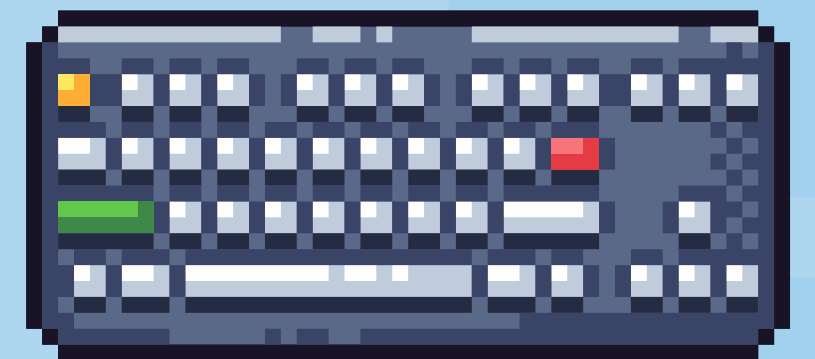
- Algorithm
- Basic Programming
- Data Type

Algorithm คืออะไร ?



Algorithm

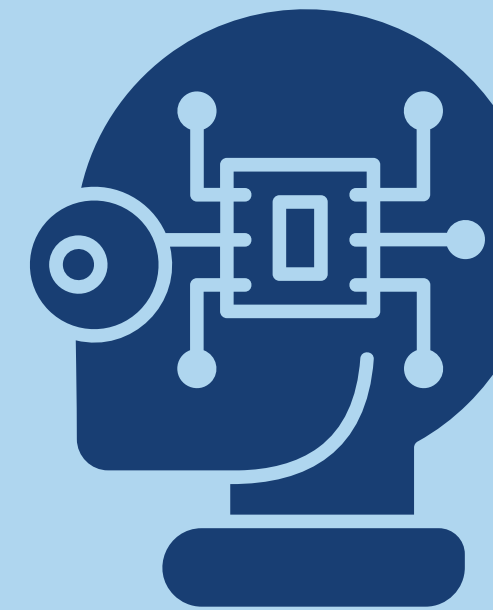
- ขั้นตอนในการแก้ปัญหาซึ่งจะช่วยให้ผู้พัฒนาโปรแกรมเห็นขั้นตอนการเขียนโปรแกรมง่ายขึ้น



Algorithm
=
Pseudocode ?

Algorithm

อธิบายการทำงานออกมา
เป็นขั้นตอนที่ระบุได้ชัดเจน



Pseudo Code

การอธิบายการทำงานด้วยรูปแบบ
ของโปรแกรมอย่างง่าย



EX. เลขไหนมากกว่ากัน

Algorithm

- 1.รับค่า A , B เป็นตัวเลข
- 2.ตรวจสอบค่า A,B
 - ถ้า $A > B$
แสดงผลเป็น $A > B$
 - ถ้า $A < B$
แสดงผลเป็น $A < B$
 - ถ้า $A = B$
แสดงผลเป็น $A = B$

Pseudo Code

```
A = INPUT AS INT
B = INPUT AS INT
IF A > B
    PRINT A > B
ELSE IF A < B
    PRINT A < B
ELSE
    PRINT A = B
```



EX. บวกเลขจนกว่าจะได้ผลเป็น -1

Algorithm

1. กำหนดค่า A เป็นตัวเลขเท่ากับ 0
2. รับค่า B เป็นตัวเลข
3. นำค่า A บวกด้วยค่า B
4. แสดงผลค่า A
5. ถ้า A ไม่เท่ากับ -1
 กลับไปข้อ 2



EX. บวกเลขจนกว่าจะได้ผลเป็น -1

Pseudo Code

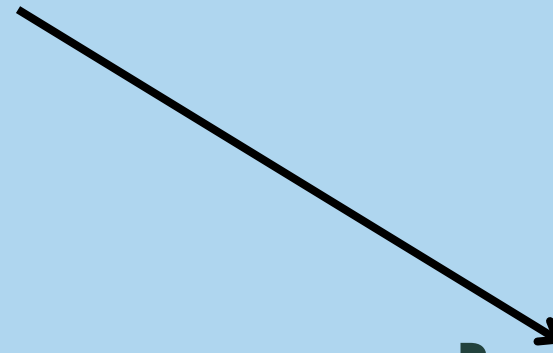
```
A = 0  
WHILE A != -1  
    B = INPUT AS INT  
    A += B  
PRINT A
```



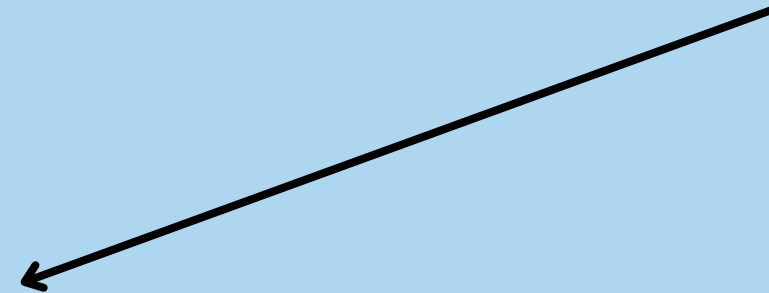
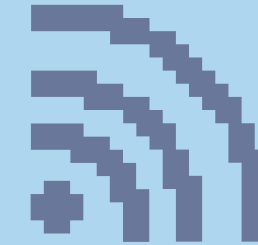
algorithm

เป็นอย่างไร?

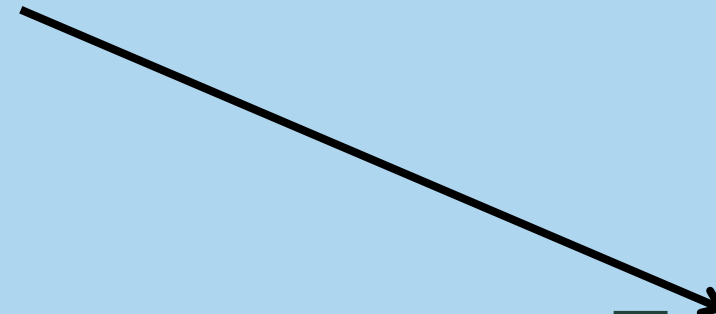
มีปัญหา



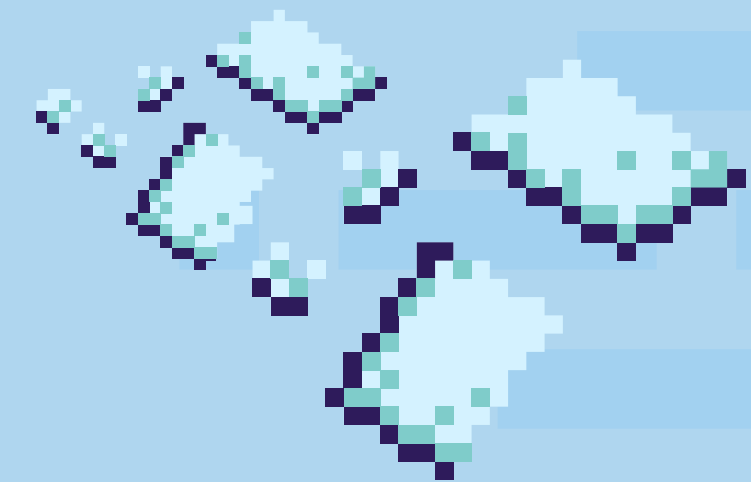
วิธีแก้ปัญหา

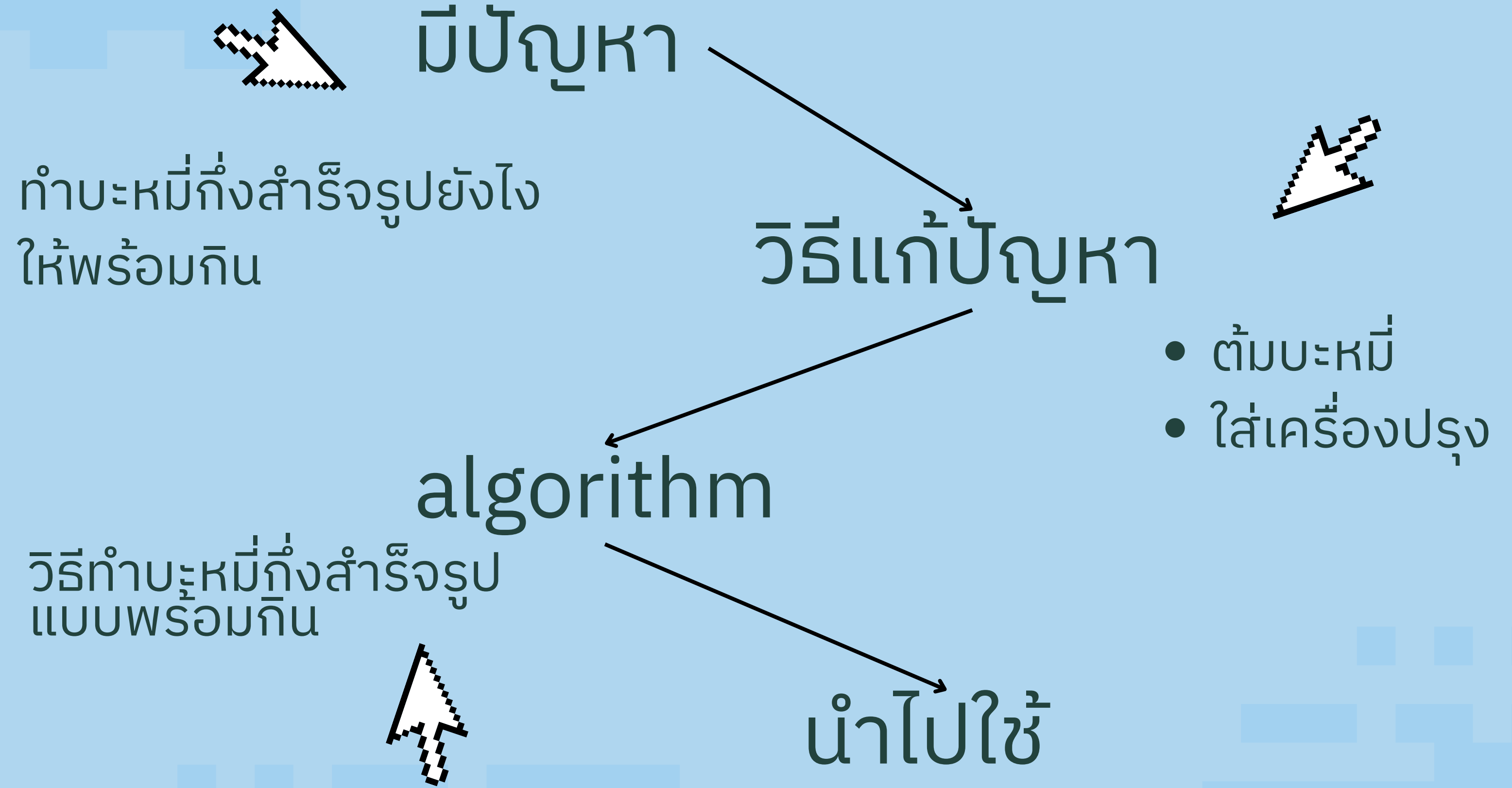


algorithm



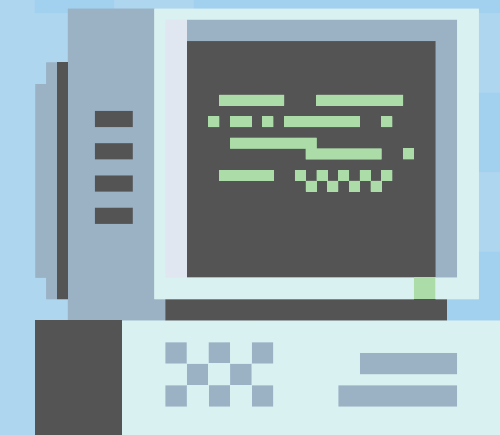
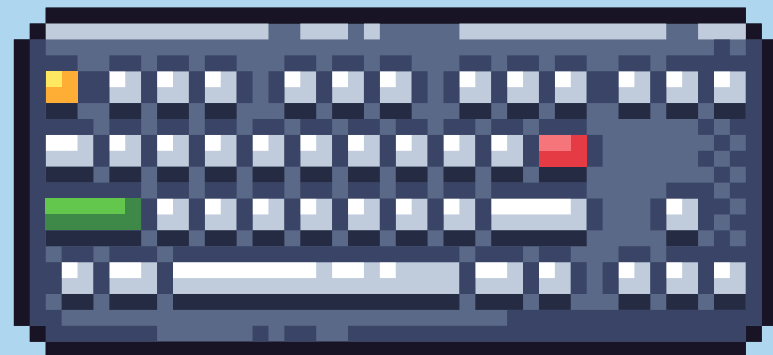
นำไปใช้





หลักการเขียนอัลกอริทึม

1. กระบวนการสำคัญควรเริ่มต้นที่จุดเดียว
2. กำหนดการทำงานเป็นขั้นเป็นตอนอย่างชัดเจน
3. การทำงานแต่ละขั้นตอนควรสั้นกระชับ
4. ผลลัพธ์ในแต่ละขั้นตอนควรต่อเนื่องกัน



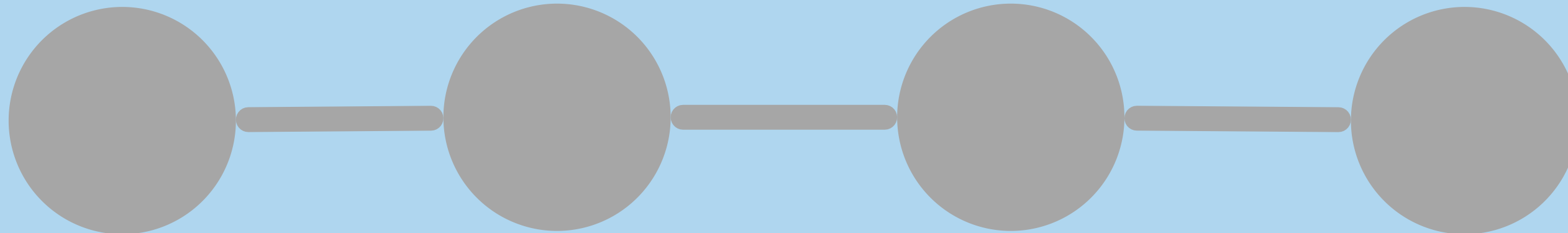
การเขียนอัลกอริทึม



1. กระบวนการสำคัญควรเริ่มต้นที่จุดเดียว

ในการมีจุดเริ่มต้นหลายที่จะทำให้กระบวนการวิธี
สับสน จนในที่สุดอาจทำให้ผลลัพธ์ที่ได้ไม่ตรงกับความต้องการ
หรืออาจทำให้ไม่สามารถทำงานได้เลย

START

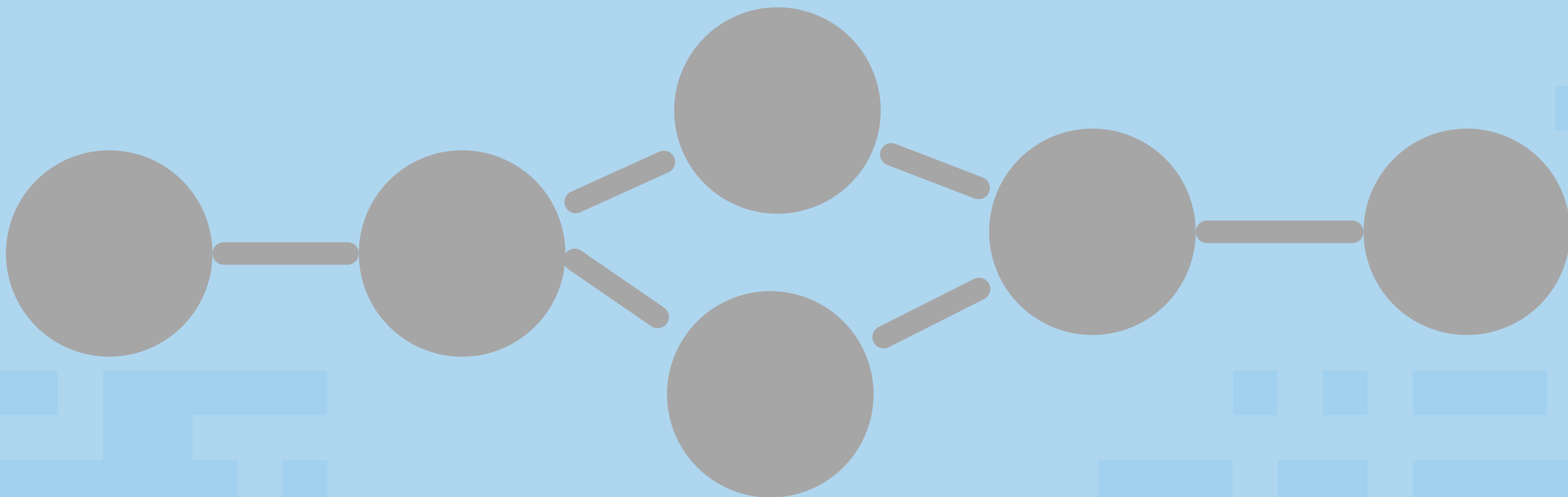
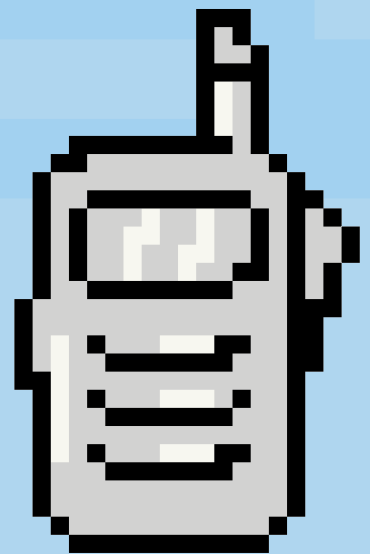


END

การเขียนอัลกอริทึม

2. กำหนดการทำงานเป็นขั้นเป็นตอนอย่างชัดเจน

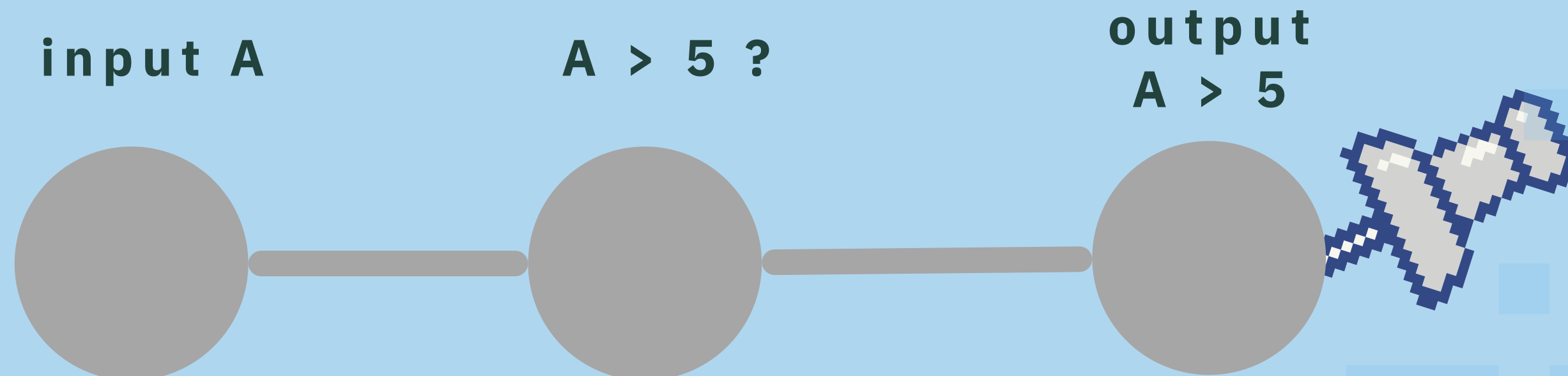
ขั้นตอนที่ชัดเจนไม่คลุมเครือ เช่น หากทำงานในขั้นตอนที่ 1 เสร็จจากนั้นก็ไปทำงานในขั้นตอนที่ 2 ต่อโดยที่ต้องกำหนดให้ชัดเจนว่าจากขั้นตอนที่ 1 ไปขั้นตอนที่ 2 นั้นมีเงื่อนไขอย่างไร



การเขียนอัลกอริทึม

3. การทำงานแต่ละขั้นตอนควรสั้นกระชับ

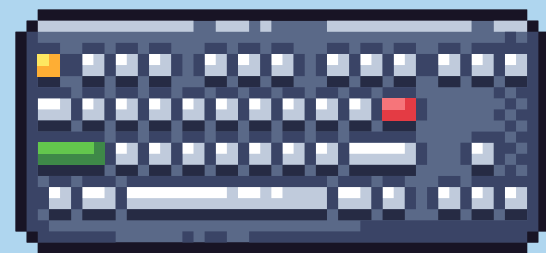
การกำหนดขั้นตอนการทำงานให้สั้นกระชับนอกจากที่จะ
ทำให้โปรแกรมทำงานเร็วขึ้นแล้วก็ยังสามารถทำให้ผู้ที่มา
ทำการพัฒนา algorithm ต่อจากเรานั้นอ่านง่ายด้วย



การเขียนอัลกอริทึม

4. ผลลัพธ์ในแต่ละขั้นตอนควรต่อเนื่องกัน

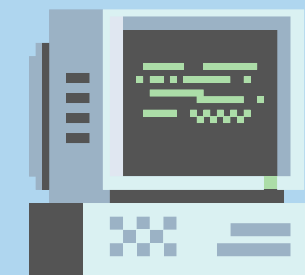
ผลลัพธ์จากขั้นตอนแรกควรเป็นข้อมูลสำหรับนำเข้า ให้กับข้อมูลในขั้นตอนต่อไป ต่อเนื่องกันไปจนกระทั่งได้ผลลัพธ์ตามที่ต้องการ



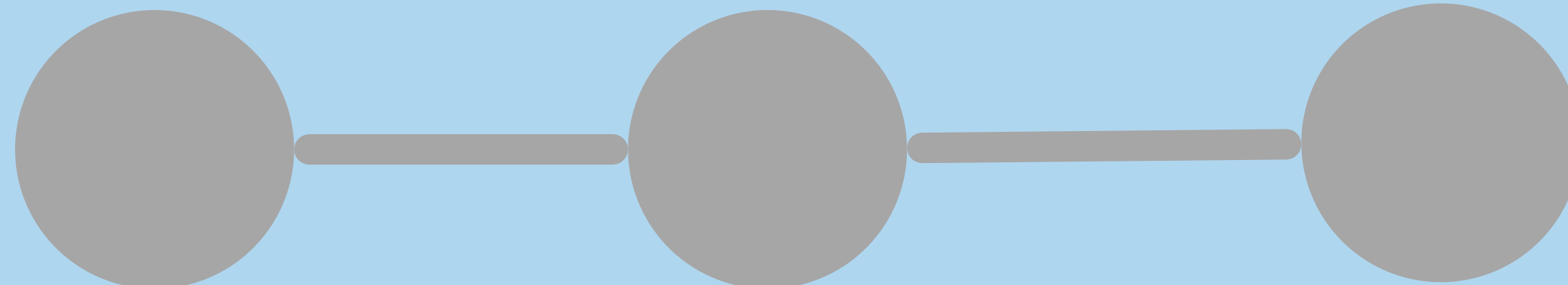
input



process



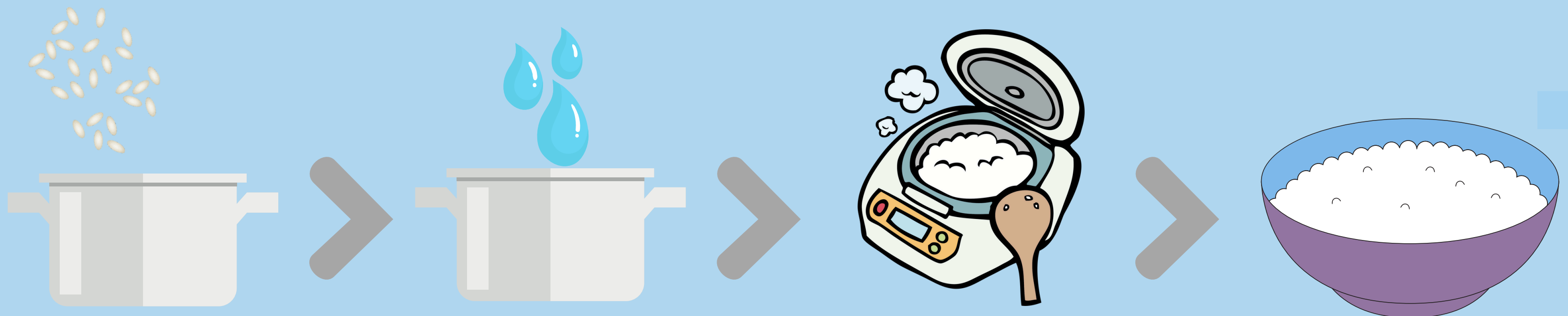
output



ตัวอย่าง ALGORITHM ในชีวิตประจำวัน

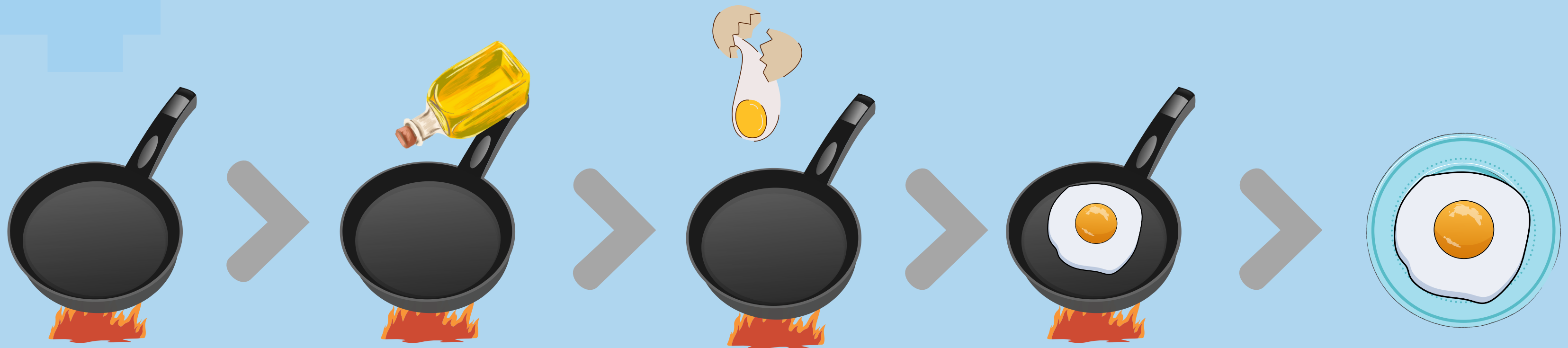
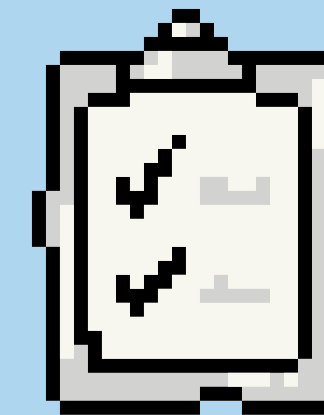


วิธีการหุงข้าว



ตัวอย่าง ALGORITHM ในชีวิตประจำวัน

วิธีในการทอดไข่ดาว

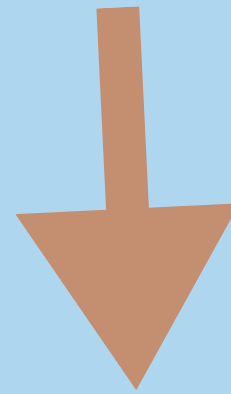


BASIC PROGRAMING

เราจะมาดูโครงสร้างคร่าวๆของภาษา java กันว่ามีอะไรบ้าง

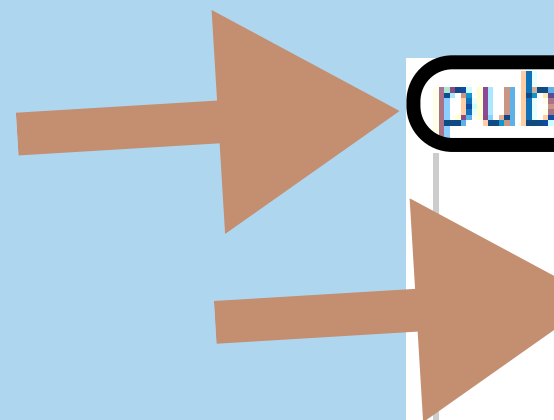
```
public class Sawaddee {  
    public static void main(String[] args) {  
        System.out.println(x:"Hell o World!");  
    }  
}
```

1. ชื่อ class



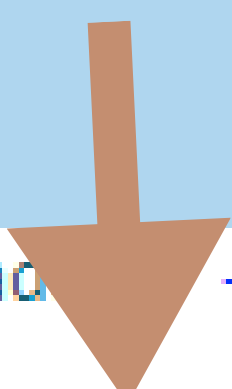
```
public class Sawaddee {  
    public static void main(String[] args) {  
        System.out.println("Hell o World!");  
    }  
}
```

2.access modifier



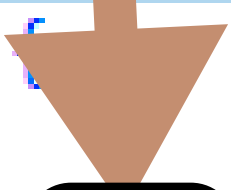
```
public class Sawaddee {  
    public static void main(String[] args) {  
        System.out.println(x:"Hell o World!");  
    }  
}
```

3.Method return type



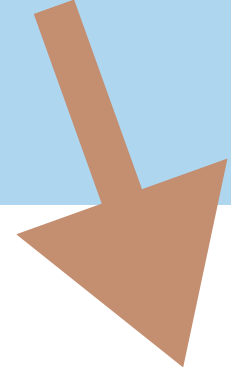
```
public class Sawad {  
    public static void main(String[] args) {  
        System.out.println(x:"Hell o World!");  
    }  
}
```


4.Method name



```
public class Sawaddee {  
    public static void main(String[] args) {  
        System.out.println(x:"Hell o World!");  
    }  
}
```

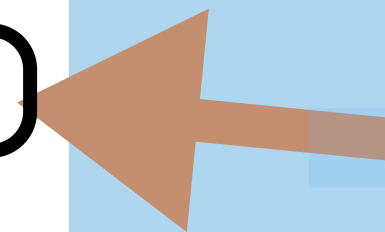
5.Parameter list



```
public class Sawaddee {  
    public static void main(String[] args) {  
        System.out.println(x:"Hell o World!");  
    }  
}
```

6.Method Body

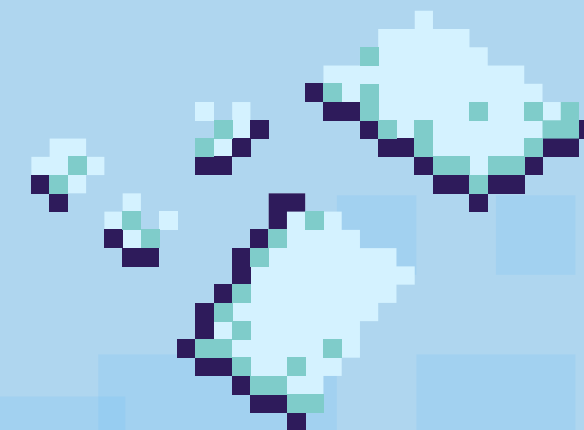
```
public class Sawaddee {  
    public static void main(String[] args) {  
        System.out.println(x:"Hell o World!");  
    }  
}
```



การตั้งชื่อ

การที่จะตั้งชื่อ ไม่ว่าจะตั้งให้กับ class, method หรือ ตัวแปรต่าง ๆ นั้นจะมีกฎในการตั้งอยู่ เช่น

- ชื่อสามารถประกอบด้วยตัวเลข ตัวอักษร และ \$ หรือ _ ก็ได้
- ชื่อห้ามขึ้นต้นด้วยตัวเลข
- ชื่อห้ามมีช่องว่าง
- ตัวอักษรเล็กและใหญ่ถือเป็นคนละตัวกัน
- ชื่อต้องไม่เป็นคำสงวน



ตัวอย่างคำสั่งของ java

| | | | |
|-----------------------|----------------------------|------------------------|---------------------------|
| <code>abstract</code> | <code>do</code> | <code>int</code> | <code>short</code> |
| <code>assert</code> | <code>double</code> | <code>interface</code> | <code>static</code> |
| <code>boolean</code> | <code>else</code> | <code>long</code> | <code>strictfp</code> |
| <code>break</code> | <code>enum</code> | <code>native</code> | <code>super</code> |
| <code>byte</code> | <code>extends</code> | <code>new</code> | <code>switch</code> |
| <code>case</code> | <code>final finally</code> | <code>null</code> | <code>synchronized</code> |
| <code>catch</code> | <code>float for of</code> | <code>package</code> | <code>this</code> |
| <code>class</code> | <code>implements</code> | <code>private</code> | <code>throw</code> |
| <code>continue</code> | <code>import</code> | <code>protected</code> | <code>throws</code> |
| <code>default</code> | <code>instanceOf</code> | <code>return</code> | <code>transient</code> |

DATA TYPE

DATA TYPE

แบ่งได้เป็น 2 กลุ่มใหญ่ๆ คือ

1. ข้อมูลพื้นฐาน (PRIMITIVE DATA TYPES)
2. ข้อมูลอ้างอิง (REFERENCE DATA TYPES)

DATA TYPE

ข้อมูลพื้นฐาน

1. จำนวนเต็ม

เลขจำนวนเต็มใดๆที่เป็นได้ทั้ง
เต็มบวก เต็มลบ เต็มศูนย์

เช่น

เต็มบวก : 1 15 75

เต็มลบ : -9 -55 -8

เต็มศูนย์ : 0

2. จำนวนจริง

เลขจำนวนเต็มใดๆที่มีทศนิยม

เช่น

99.99 58.00 1.0 1.11

-9.05 -4.98 -0.000001

DATA TYPE

ข้อมูลพื้นฐาน

3. ตัวอักษร :

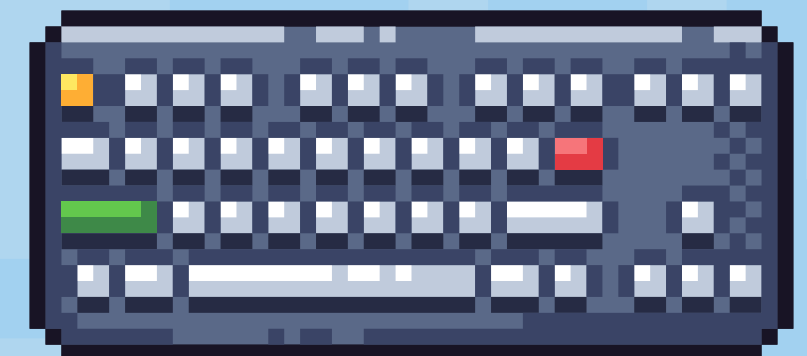
ตัวเลข ตัวอักษร หรือ สัญลักษณ์
พิเศษที่เป็นอักขระเดี่ยว โดยจะใช้
single quote(' ') ในการกำกับ

เช่น

'a' 'A' '4' '0' '%' '^'

4. ตรรกะ :

ค่าทางตรรกะ โดยจะเก็บได้ 2 ค่า
คือ true และ false



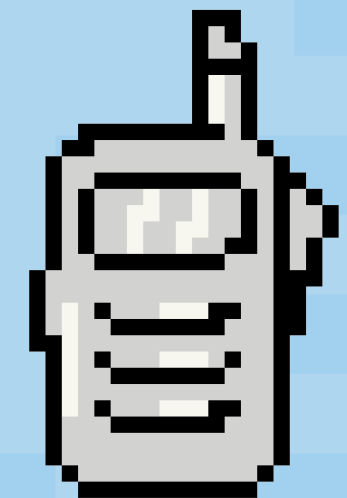
DATA TYPE

ข้อมูลอ้างอิง

class - ประเภทของ class นั้นมีอยู่มากมาย
โดยจะขอกล่าวถึงอันที่ได้ใช้บ่อยๆ เช่น String

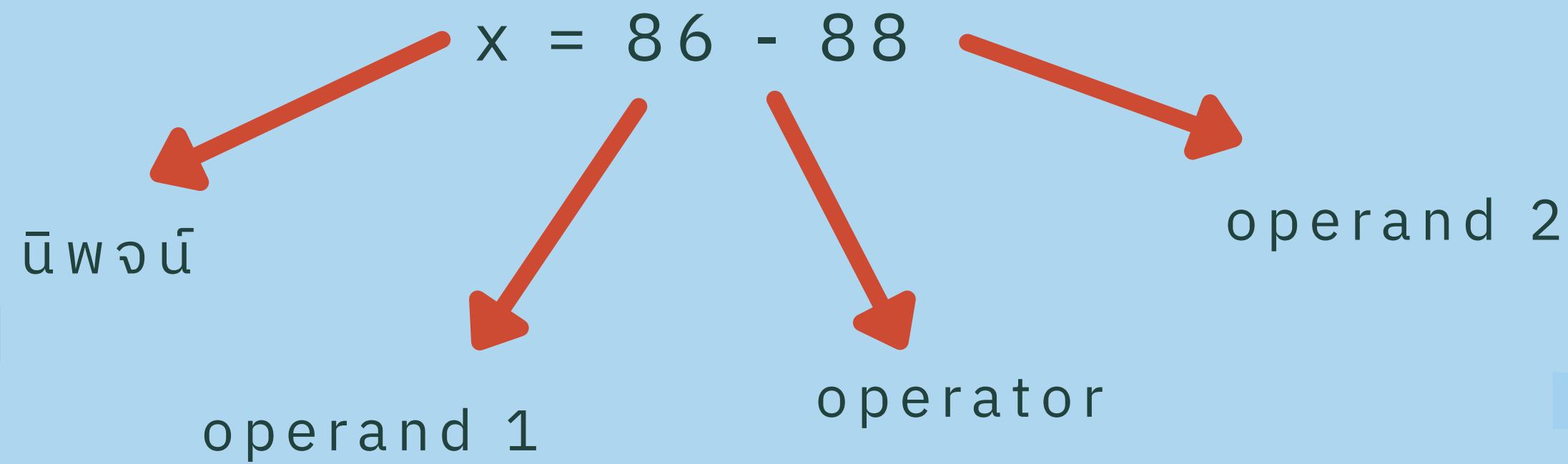
ตัวของ String นั้นจะเก็บข้อมูลเป็นตัวอักษรตัว
เดียวหรือหลายตัวก็ได้ โดยที่จะมี double
quote(“ ”) ในการกำกับ

เช่น “hell o world!” “muhahaha” “lnwza0007”



นิพจน์ (EXPRESSION)

รูปแบบการคำนวณค่าต่างๆโดยจะประกอบไปด้วย operand และ operator เป็นตัวเชื่อม เช่น



นิพจน์ (EXPRESSION)

Operator คือสัญลักษณ์ที่ใช้ทางคณิตศาสตร์ โดยจะพิจารณาลำดับการทำงานตามหลักคณิตศาสตร์

| Operator | ความหมาย |
|----------|-------------------|
| + | บวก |
| - | ลบ |
| * | คูณ |
| / | หาร |
| % | หารแบบเอาเศษเหลือ |

วิธีสร้างตัวแปร

- ตัวแปรทุกตัวต้องมีการกำหนดชื่อและกำหนดประเภทของข้อมูล
- ต้องประกาศค่าของตัวแปรก่อนที่นำมาตัวแปรนั้นมาใช้



วิธีสร้างตัวแปร

```
public static void main(String[] args) {  
    System.out.println(x:"Hell o World!");  
}
```

ประเภทข้อมูล

| | |
|---------|--------|
| int | x |
| double | y |
| char | z |
| boolean | isTrue |
| String | say |

| |
|---------------|
| = 10 + 90; |
| = 100.004854; |
| = 'A'; |
| = true; |
| = "Hello"; |

ค่าที่เก็บใน
ตัวแปร

ตัวแปรที่ยังไม่ได้ใส่ค่า
(ยังนำไปใช้ไม่ได้)

int empty;

ชื่อตัวแปร