ใบงานการทดลองที่ 1

เรื่อง ภาษา Dart

1. จุดประสงค์

1 ใช้เครื่องมือสำหรับพัฒนาโปรแกรมภาษา Dart ได้

2 เขียนประกาศตัวแปร เงื่อนไขและลูปในภาษา Dart ได้

2. ทฤษฎี

บทที่ 1.2 ตัวแปร

1.2.1 ประเภทตัวแปรพื้นฐานในภาษา Dart

ในภาษา Dart มีประเภทตัวแปรพื้นฐานหลายประเภทที่ใช้ในการประกาศตัวแปรและเก็บข้อมูลต่าง ๆ สามารถดูได้ดังตารางที่ 1.2.1.1

ประเภท	ความหมาย
int	ใช้เก็บจำนวนเต็ม เช่น 1, -5, 1000
double	ใช้เก็บจำนวนทศนิยม เช่น 3.14, -0.5, 100.0
String	ใช้เก็บข้อความ เช่น "Hello, Dart", 'OpenAI', "12345"
bool	ใช้เก็บค่าความจริง (true หรือ false) เช่น ‘true’, ‘false’
List	ใช้เก็บชุดข้อมูลที่เรียงต่อกัน เช่น [1, 2, 3], ["apple", "banana", "cherry"]
Map	ใช้เก็บข้อมูลแบบ key-value pairs เช่น {"name": "John", "age": 30, "city": "New York"}
dynamic	สามารถเก็บค่าข้อมูลประเภทใดก็ได้ (เปลี่ยนแปลงประเภทได้เรื่อย ๆ)
var	สามารถเก็บค่าข้อมูลประเภทใดก็ได้ (ไม่สามารถเปลี่ยนแปลงประเภทได้หลังจากรับประเภทตัวแปรแรก)

ตารางที่ 1.2.1.1 ตารางแสดงประเภทตัวแปรและความหมาย

1.2.2 รูปแบบการประกาศตัวแปร

ในการเขียนโปรแกรมคอมพิวเตอร์นั้น สิ่งที่นักเขียนโปรแกรมจะต้องรู้คือ รูปแบบการ ประกาศตัว เพื่อนำมาใช้ในการทำงาน ของโปรแกรมที่ได้เขียนขึ้นมา ในภาษาDartนั้น มีรูปแบบการประกาศตัวแปรดังต่อไปนี้

type variable_name; หรืออีกรูปแบบหนึ่ง type variable_name = value;

อธิบาย

type	คือ ชนิดของตัวแปรที่จะประกาศ ซึ่งควรจะมีการกำหนดชนิดของตัวแปรให้มีความเหมาะสมกับงานที่จะนำไปใช้
variable_name	คือ ชื่อของตัวแปรที่ต้องการสร้าง
value	คือ ค่าของตัวแปรนั้น ๆ ตามที่กำหนดให้

1.2.3 การประกาศและใช้งานตัวแปร

ตัวเลข (numbers)

ตัวเลขในภาษา Dart แบ่งออกเป็น 2 ประเภทหลักๆ คือ

จำนวนเต็ม (int)

- เป็นตัวเลขที่ไม่มีทศนิยม
- มีขอบเขตตั้งแต่ -2,147,483,648 ถึง 2,147,483,647

ตัวเลขทศนิยม (double)

- เป็นตัวเลขที่มีทศนิยม
- มีขอบเขตกว้างกว่าจำนวนเต็ม

```
int age = 25;
double height = 1.70;
num weight = 70.5;
```

ข้อความ (strings)

ข้อความในภาษา Dart เรียกว่า String สามารถสร้างข้อความได้โดยการใช้เครื่องหมายอัญประกาศคู่ หรือเครื่องหมายอัญประกาศสามคู่

ตัวอย่างการใช้ประเภทข้อมูลข้อความ

```
String name = "John Doe";
String message = """ Hello, world! """;
```

ตรรกะ (booleans)

ตรรกะในภาษา Dart เรียกว่า bool มีค่าได้เพียง 2 ค่า คือ true หรือ false

ตัวอย่างการใช้ประเภทข้อมูลตรรกะ

```
bool isStudent = false;
```

รายการ (lists)

รายการในภาษา Dart เรียกว่า List เป็นโครงสร้างข้อมูลแบบลำดับที่สามารถเก็บค่าข้อมูลได้หลายค่า

ตัวอย่างการใช้ประเภทข้อมูลรายการ

```
List<int> numbers = [1, 2, 3, 4, 5];
List<String> names = ["John", "Doe", "Smith"];
```

ตัวแปรแบบไดนามิก (dynamic)

ตัวแปรแบบไดนามิกในภาษา Dart เรียกว่า dynamic สามารถเก็บค่าข้อมูลประเภทใดก็ได้ สามารถเปลี่ยนชนิดข้อมูลของตัวแปรแบบ dynamic ได้ตลอดเวลา

ตัวอย่างการใช้ประเภทข้อมูลตัวแปรแบบไดนามิก

```
dynamic value = 1;
dynamic value = "Hello, world!";
dynamic value = true;
```

ตัวแปรแบบ var

การประกาศตัวแปรแบบ var ในภาษา Dart เป็นการประกาศตัวแปรโดยไม่จำเป็นต้องกำหนดชนิดข้อมูลให้กับตัวแปร ตัวแปรแบบ var จะกำหนดชนิดข้อมูลให้กับตัวแปรโดยอัตโนมัติตามค่าที่ให้กับตัวแปรตอนประกาศ ไม่สามารถเปลี่ยนชนิดข้อมูลของตัวแปรแบบ var หลังจากที่กำหนดค่าให้กับตัวแปรแล้ว

ตัวอย่างการใช้ประเภทข้อมูลตัวแปรแบบ var

```
var name = "Teerasej";
var age = 25;
var isMale = true;
```

ข้อดีและข้อเสียในการประกาศตัวแปรแบบ var และ dynamic

การประกาศตัวแปรแบบ var

- ข้อดี: เขียนโค้ดได้สั้นและกระชับ
- ข้อเสีย: อาจทำให้โค้ดไม่ชัดเจนและเข้าใจยาก

การประกาศตัวแปรแบบ dynamic

- ข้อดี: ยืดหยุ่นและสามารถปรับเปลี่ยนโค้ดได้สะดวก
- ข้อเสีย: อาจทำให้โค้ดไม่ปลอดภัยและก่อให้เกิดช่องโหว่ด้านความปลอดภัย

การเลือกการประกาศตัวแปรแบบ var หรือ dynamic

ควรเลือกการประกาศตัวแปรแบบ var หรือ dynamic ให้เหมาะสมกับการใช้งาน โดยทั่วไปแล้ว ควรใช้การประกาศตัวแปรแบบ var หากมั่นใจว่าตัวแปรจะมีชนิดข้อมูลคงที่ตลอดอายุการใช้งาน และใช้การประกาศตัวแปรแบบ dynamic หากต้องการความยืดหยุ่นในการเก็บค่าของตัวแปร

บทที่ 1.3 เครื่องหมาย

เครื่องหมายในภาษา Dart มีความสำคัญมากในการเขียนโปรแกรม เนื่องจากใช้ในการดำเนินการทางคณิตศาสตร์ เปรียบเทียบ และตรรกะ ซึ่งจะช่วยให้เราสามารถเขียนโปรแกรมได้อย่างถูกต้องและมีประสิทธิภาพ เครื่องหมายในภาษา Dart มี 3 ประเภทหลักๆ ดังนี้

1.3.1 เครื่องหมายเชิงคณิตศาสตร์

เครื่องหมายเชิงคณิตศาสตร์ในภาษา Dart ใช้ในการดำเนินการทางคณิตศาสตร์ เช่น การบวก การลบ การคูณ การหาร และการหารส่วนเหลือ โดยสามารถดูสัญลักษณ์เครื่องหมายและความหมายได้ดังตารางที่ 1.3.1.1

เครื่องหมาย	ความหมาย
+	เครื่องหมายบวกใช้ในการบวกค่าตัวแปร
-	เครื่องหมายลบใช้ในการลบค่าตัวแปร
*	เครื่องหมายคูณใช้ในการคูณค่าตัวแปร
/	เครื่องหมายหารใช้ในการหารค่าตัวแปร
%	เครื่องหมายโมดูลัสใช้ในการหารค่าตัวแปรและคืนเศษ

ตารางที่ 1.3.1.1 ตารางแสดงเครื่องหมายเชิงคณิตศาสตร์และความหมาย

1.3.2 เครื่องหมายเชิงเปรียบเทียบ

เครื่องหมายเปรียบเทียบในภาษา Dart ใช้ในการเปรียบเทียบค่าสองค่า เช่น เท่ากับ ไม่เท่ากับ มากกว่า น้อยกว่า มากกว่าหรือเท่ากับ น้อยกว่าหรือเท่ากับ โดยสามารถดูสัญลักษณ์เครื่องหมายและความหมายได้ดังตารางที่ 1.3.2.1

เครื่องหมาย	ความหมาย
==	เครื่องหมายเท่ากับเท่ากับใช้ในการเปรียบเทียบว่าสองค่าเท่ากันหรือไม่
!=	เครื่องหมายไม่เท่ากับใช้ในการเปรียบเทียบว่าสองค่าไม่เท่ากัน
>	เครื่องหมายมากกว่าใช้ในการเปรียบเทียบค่าทางตัวเลขและตรรกะว่าค่าด้านซ้ายมากกว่าค่าด้านขวา
<	เครื่องหมายน้อยกว่าใช้ในการเปรียบเทียบค่าทางตัวเลขและตรรกะว่าค่าด้านซ้ายน้อยกว่าค่าด้านขวา
>=	เครื่องหมายมากกว่าหรือเท่ากับใช้ในการเปรียบเทียบค่าทางตัวเลขและตรรกะว่าค่าด้านซ้ายมากกว่าหรือเท่ากับค่าด้านขวา
<=	เครื่องหมายน้อยกว่าหรือเท่ากับใช้ในการเปรียบเทียบค่าทางตัวเลขและตรรกะว่าค่าด้านซ้ายน้อยกว่าหรือเท่ากับค่าด้านขวา

ตารางที่ 1.3.2.1 ตารางแสดงเครื่องหมายเปรียบเทียบและความหมาย

1.3.3 เครื่องหมายเชิงตรรกะ

เครื่องหมายตรรกะในภาษา Dart ใช้ในการดำเนินการทางตรรกะ เช่น AND OR NOT โดยสามารถดูสัญลักษณ์เครื่องหมายและความหมายได้ดังตารางที่ 1.3.3.1

เครื่องหมาย	ความหมาย
&&	เครื่องหมายและใช้ในการรวมเงื่อนไขตรรกะและควบคุมการสร้างเงื่อนไขที่ต้องเป็นจริงทั้งสองข้าง
||	เครื่องหมายหรือใช้ในการรวมเงื่อนไขตรรกะและควบคุมการสร้างเงื่อนไขที่ต้องเป็นจริงอย่างน้อยหนึ่งข้าง
!	เครื่องหมายตรรกะ NOT ใช้ในการเปลี่ยนค่าตรรกะจากจริงเป็นเท็จหรือจากเท็จเป็นจริง

ตารางที่ 1.3.3.1 ตารางแสดงเครื่องหมายตรรกะและความหมาย

บทที่ 1.4 เงื่อนไขและลูป

1.4.1 คำสั่งเงื่อนไข

โครงสร้างเงื่อนไขในภาษา Dart ใช้ในการกำหนดว่าโปรแกรมจะดำเนินการต่ออย่างไรขึ้นอยู่กับเงื่อนไขที่กำหนดไว้ โครงสร้างเงื่อนไขในภาษา Dart มี 2 ประเภทหลักๆ ดังนี้

- If - else
- Switch – case

คำสั่ง If - else

คำสั่ง if-else เป็นคำสั่งที่มีเงื่อนไขเดียว ทำงานเฉพาะเมื่อเงื่อนไขที่กำหนดไว้เป็นจริง หากเงื่อนไขเป็นเท็จ คำสั่งในบล็อก else จะถูกเรียกทำงาน

โครงสร้าง if-else มีรูปแบบดังนี้

```
if (เงื่อนไข) {
  // ทำคำสั่งหากเงื่อนไขเป็นจริง
} else {
  // ทำคำสั่งหากเงื่อนไขเป็นเท็จ
}
```

ตัวอย่างการใช้งาน

```
void main() {
  var a = 10;
  var b = 20;

  if (a > b) {
    print("a มากกว่า b");
  } else {
    print("a ไม่มากกว่า b");
  }
}
```

ผลลัพธ์

```
a ไม่มากกว่า b
```

ในตัวอย่างนี้ เราจะกำหนดตัวแปร a เป็นค่า 10 และตัวแปร b เป็นค่า 20 จากนั้นเราจะใช้โครงสร้างเงื่อนไข if-else เพื่อตัดสินใจว่าจะทำคำสั่งใดต่อไป โดยเงื่อนไขคือ a > b ถ้าเงื่อนไขเป็นจริงก็จะพิมพ์ข้อความว่า "a มากกว่า b" แต่ถ้าเงื่อนไขเป็นเท็จก็จะพิมพ์ข้อความว่า "a ไม่มากกว่า b"

คำสั่ง switch-case

คำสั่ง switch-case เป็นคำสั่งที่มีเงื่อนไขหลายเงื่อนไข ทำงานเฉพาะเมื่อเงื่อนไขที่กำหนดไว้ตรงกับเงื่อนไขใน case หากเงื่อนไขใน case ใดตรงกัน คำสั่งในบล็อก case นั้นจะถูกเรียกทำงาน

โครงสร้างเงื่อนไข switch มีรูปแบบดังนี้

```
switch (ตัวแปร) {
  case ค่าที่ 1:
    // คำสั่งที่จะทำถ้าค่าของตัวแปรตรงกับค่าที่ 1
    break;
  case ค่าที่ 2:
    // คำสั่งที่จะทำถ้าค่าของตัวแปรตรงกับค่าที่ 2
    break;
  ...
  default:
    // คำสั่งที่จะทำถ้าค่าของตัวแปรไม่ตรงกับค่าใดๆ ที่กำหนดไว้ใน
case
    break;
}
```

ตัวอย่างการใช้งาน

```
void main() {
  var number = "1";
  switch (number) {
    case "1":
      print("เป็นเลข 1");
      break;
    case "2":
      print("เป็นเลข 2");
      break;
    default:
      print("เลขไม่ตรงกับเลขอะไรเลย");
      break;
  }
}
```

```
ผลลัพธ์
เป็นเลข 1
```

ในตัวอย่างนี้ เราจะกำหนดตัวแปร number เป็นค่า "1" จากนั้นเราจะใช้โครงสร้างเงื่อนไข switch เพื่อตัดสินใจว่าจะทำคำสั่งใดต่อไป โดยค่าของตัวแปร number จะตรงกับค่าที่กำหนดไว้ใน case นั้นๆ ถ้าค่าของตัวแปร number ตรงกับค่าที่กำหนดไว้ก็จะทำคำสั่งที่อยู่ใน case นั้น แต่ถ้าค่าของตัวแปร number ไม่ตรงกับค่าที่กำหนดไว้ ก็จะทำคำสั่งที่อยู่ใน default คำสั่ง break ใช้ในโครงสร้างเงื่อนไข switch เพื่อหยุดการวนซ้ำของโครงสร้างเงื่อนไข

1.4.2 คำสั่งทำซ้ำ

คำสั่งทำซ้ำเป็นคำสั่งควบคุมที่ทำงานซ้ำๆ คำสั่งในโครงสร้างวนซ้ำจะถูกเรียกทำงานซ้ำๆ จนกว่าจะถึงเงื่อนไขที่กำหนดไว้

โครงสร้างวนซ้ำในภาษา Dart มี 3 รูปแบบ คือ

- คำสั่ง while
- คำสั่ง for
- คำสั่ง do-while

คำสั่ง while

โครงสร้างคำสั่ง while ในภาษา Dart ใช้ในการวนซ้ำคำสั่งต่างๆ ไปเรื่อยๆ ตราบที่เงื่อนไขที่กำหนดไว้เป็นจริง โครงสร้างคำสั่ง while มีรูปแบบดังนี้

```
while (เงื่อนไข) {
  // คำสั่งที่จะทำซ้ำๆ
}
```

ตัวอย่างการใช้งาน

```
void main() {
  var i = 1;

  while (i <= 5 ) {
    print(i);
    i++;
  }
}
```

```
ผลลัพธ์
1
2
3
4
5
```

ในตัวอย่างนี้ เราจะวนซ้ำคำสั่ง print() ไปเรื่อยๆ ตราบที่ค่าของตัวแปร i น้อยกว่าหรือเท่ากับ 5 เมื่อค่าของตัวแปร i มากกว่า 5 เงื่อนไขจะเป็นเท็จ และโครงสร้างคำสั่ง while จะหยุดการวนซ้ำ

คำสั่ง for

โครงสร้างคำสั่ง for ในภาษา Dart ใช้ในการวนซ้ำคำสั่งต่างๆ ไปเรื่อย ๆ ตามจำนวนรอบที่กำหนด โครงสร้างคำสั่ง for มีรูปแบบดังนี้

```
for (ตัวแปรเริ่มต้น; เงื่อนไข; คำสั่งเปลี่ยนแปลงตัวแปร) {
  // คำสั่งที่จะทำซ้ำๆ
}
```

ตัวอย่างการใช้งาน

```
void main() {
  for (var i = 1; i <= 5; i++) {
    print(i);
  }
}
```

```
ผลลัพธ์
1
2
3
4
5
```

ในตัวอย่างนี้ เราจะวนซ้ำคำสั่ง print() ไปเรื่อย ๆ จำนวน 5 รอบ โดยเริ่มต้นค่าของตัวแปร i เป็น 1 จากนั้นจะเพิ่มค่าของตัวแปร i ทีละ 1 ในแต่ละรอบ

คำสั่ง do – while

โครงสร้างคำสั่ง do-while ในภาษา Dart ใช้ในการวนซ้ำคำสั่งต่างๆ ไปเรื่อย ๆ อย่างน้อย 1 รอบ ตราบที่เงื่อนไขที่กำหนดไว้เป็นจริง โครงสร้างคำสั่ง do-while มีรูปแบบดังนี้

```
do {
  // คำสั่งที่จะทำซ้ำๆ
} while (เงื่อนไข);
```

ตัวอย่างการใช้งาน

```
void main() {
  var i = 1;

  do {
    print(i);
    i++;
  } while (i <= 5);
}
```

```
ผลลัพธ์
1
2
3
4
5
```

ในตัวอย่างนี้ เราจะวนซ้ำคำสั่ง print() ไปเรื่อย ๆ อย่างน้อย 1 รอบ โดยเริ่มต้นค่าของตัวแปร i เป็น 1 จากนั้นจะเพิ่มค่าของตัวแปร i ทีละ 1 ในแต่ละรอบ

3. เครื่องมือและอุปกรณ์การทดลอง

3.1 คอมพิวเตอร์ 1 เครื่อง

3.2 ใบงานที่ 1 เรื่อง คำสั่งเลือกเงื่อนไข

4. ลำดับขั้นการทดลอง

4.1 นักเรียนศึกษาเนื้อหา เรื่อง คำสั่งเลือกเงื่อนไข

4.2 ให้นักเรียนตอบคำถาม ลงในใบงานที่ 1

4.3 ส่งงานครูหลังจากเสร็จเรียบร้อย

คำสั่ง ให้นักศึกษาเขียนคำตอบตามที่โจทย์กำหนดให้ถูกต้อง (สามารถแนบรูปโค้ดและผลลัพธ์คำตอบของโปรแกรมได้)

1. เขียนโปรแกรมคำนวณพื้นที่สามเหลี่ยมมุมฉาก

```
void main() {
  double base = 50.5;
  double high = 50.5;
  print("1/2 x $base x $high = ${1/2*base*high}");
}
```

```
1/2 x 50.5 x 50.5 = 1275.125
```

2. หากมีการประกาศตัวแปร **var n1=10** และ **dynamic n2=5** ตามในกล่องข้อความ แต่มีการเปลี่ยนแปลงค่าตัวแปรที่ประกาศในภายหลัง เมื่อมีการ print ค่า n1 และ n2 นักศึกษาคิดว่าจะสามารถ print ออกมาได้หรือไม่ เพราะเหตุใด และผลลัพธ์เป็นอย่างไร

```
ประกาศตัวแปร
var n1 = 10;
dynamic n2 = 5;

เปลี่ยนค่าใหม่
n1 = 11.5;
n2 = 15.4;
print('$n1');
print('$n2');
```

สามารถ print ค่าออกมาได้แค่ค่าเดียวคือค่าของ n2 จะได้ 15.4 เพราะ n1 มีการกำหนดค่าเป็นแบบ var ซึ่งค่าที่ var กำหนดเป็นค่าจำนวนเต็ม(int) n1 จึงไม่สามารถเป็นค่าได้เพราะ n1 = 11.5 ซึ่งแตกต่างจาก n2 ที่ประกาศเป็น dynamic ซึ่งมีความยืดหยุ่นกว่า ผลลัพธ์คือเกิด Error ขึ้นที่ n1 = 11.5 เพราะ type ไม่ตรงกับ var n1 = 10

3. หากมีการประกาศตัวแปร **var n1 และ dynamic n2** พร้อมทั้งกำหนดค่าให้ในบรรทัดต่อมาเป็น **n1=10** และ **n2=5** ตามในกล่องข้อความ แต่มีการเปลี่ยนแปลงค่าตัวแปรที่ประกาศในภายหลัง เมื่อมีการ print ค่า n1 และ n2 นักศึกษาคิดว่าจะสามารถ print ออกมาได้หรือไม่ เพราะเหตุใด

```
ประกาศตัวแปร
var n1;
dynamic n2;
n1 = 10;
n2 = 5;

เปลี่ยนค่าใหม่
n1 = 11.5;
n2 = 15.4;
print('$n1');
print('$n2');
```

เมื่อทำการ print ค่า n1 และ n2 จะสามารถเป็นค่าออกมาได้คือ 11.5 และ 15.4 เพราะ var ประกาศแค่ชื่อของตัวแปรแต่ไม่ได้กำหนดค่า ซึ่งจะเป็นการกำหนด type ให้กับ var ด้วยและเมื่อเราทำการเปลี่ยนค่าของ n1 จะเป็นการเปลี่ยนค่าของ n1=10 ที่ประกาศก่อนหน้าให้เป็นอันใหม่ n1 = 11.5 ไม่ได้เปลี่ยนแปลง var n1 ส่วน dynamic n2 ก็สามารถเปลี่ยนค่าได้อยู่แล้ว จึงสามารถเปลี่ยนค่าได้

4. เขียนโปรแกรมตรวจสอบค่าตัวเลขจำนวนเต็ม (number) โดยหาว่าเป็นจำนวนเต็มบวก (Positive Integer), จำนวนเต็มศูนย์ (Zero Integer) หรือ จำนวนเต็มลบ (Negative Integer) โดยใช้ if else

```
void main() {
   int num = 10;

    if(num < 0){
        print("$num is negative integer");

    }else if(num > 0){
        print("$num is positive integer");

    }else{
        print("$num is zero integer");

    }
}
```

```
10 is positive integer
```

5. เขียนโปรแกรม สูตรคูณแม่ M โดยใช้ for

```
void main() {
  for (int i = 1; i <= 12; i++) {
    print("$i x 2 = ${i * 2}");
  }
}
```

```
1 x 2 = 2
2 x 2 = 4
3 x 2 = 6
4 x 2 = 8
5 x 2 = 10
6 x 2 = 12
7 x 2 = 14
8 x 2 = 16
9 x 2 = 18
10 x 2 = 20
11 x 2 = 22
12 x 2 = 24
```

6. เขียนโปรแกรม สูตรคูณแม่ M โดยใช้ while

```
void main() {
  int i = 1;
  while (i <= 12) {
    print("$i x 3 = ${i * 3}");
    i++;
  }
}
```

```
1 x 3 = 3
2 x 3 = 6
3 x 3 = 9
4 x 3 = 12
5 x 3 = 15
6 x 3 = 18
7 x 3 = 21
8 x 3 = 24
9 x 3 = 27
10 x 3 = 30
11 x 3 = 33
12 x 3 = 36
```

5. สรุปผลการทดลอง

จากการทำลอง พบว่าการใช้ประเภทตัวแปรพื้นฐานในภาษา Dart มีความคล้ายครึงกับการใช้ของภาพ C เป็นอย่างมากแต่อาจจะแตกต่างกันที่ บางตัวแปรที่สามารถเปลี่ยนค่าได้หรือกำหนดค่าเองให้ได้อัตโนมัติ เช่น var และ dynamic และให้แสดงให้เห็นถึงความแตกต่างกันของ ตัวแปรพื้นฐาน 2 ตัวนี้ว่า var เขียนโค้ดได้สั้นและกระชับแต่จะไม่สามารถเปลี่ยนแปลงค่าได้ถ้ากำหนด typeให้กับ var แล้วแต่ dynamic มีความยืดหยุ่นและสามารถปรับเปลี่ยนโค้ดได้สะดวกมากกว่า และการให้คำสั่งประเภทเงื่อนไขต่างๆ ที่ต้องเลือกใช้ให้เหมาะกับสถานะการ เช่น if-else ใช้ในการเลือกเงื่อนไขเดียวคือจริง-เท็จ for เหมาะสำหรับการใช้เมื่อเรารู้จุดเริ่มต้นและสิ้นสุด while ใช้ในการที่เราไม่รู้จุดสิ้นสุด

6. คำถามหลังการทดลอง

6.1 var และ dynamic ต่างกันอย่างไร

var ใช้ต่อเมื่อเราไม่ต้องการเปลี่ยนประเภทของตัวแปรแล้วเพราะ var เปลี่ยนค่าได้แต่เปลี่ยนประเภทไม่ได้
dynamic สามารถเปลี่ยนค่าได้และเปลี่ยนประเภทได้

6.2 คำสั่ง if-else กับ switch-case ต่างกันอย่างไร

if-else เป็นคำสั่งที่มีเงื่อนไขเดียว ทำงานเฉพาะเมื่อเงื่อนไขที่กำหนดไว้เป็นจริง
switch-case เป็นคำสั่งที่มีเงื่อนไขหลายเงื่อนไข ทำงานเฉพาะเมื่อเงื่อนไขที่กำหนดไว้ตรงกับเงื่อนไขในcase

6.3 while มีโครงสร้างการทำงานอย่างไร และหากเงื่อนไขไม่เป็นจริง loop จะยังทำงานอยู่หรือไม่

โครงสร้าง while ใช้ในการวนซ้ำคำสั่งต่างๆ ไปเรื่อยๆ ตราบที่เงื่อนไขที่กำหนดไว้เป็นจริง

```
while (เงื่อนไข) {
    // คำสั่งที่จะทำซ้ำ
}
```

และเมื่อเงื่อนไขไม่เป็นจริง จะไม่ทำงานต่อ