USEFUL DATA TYPES

STRING

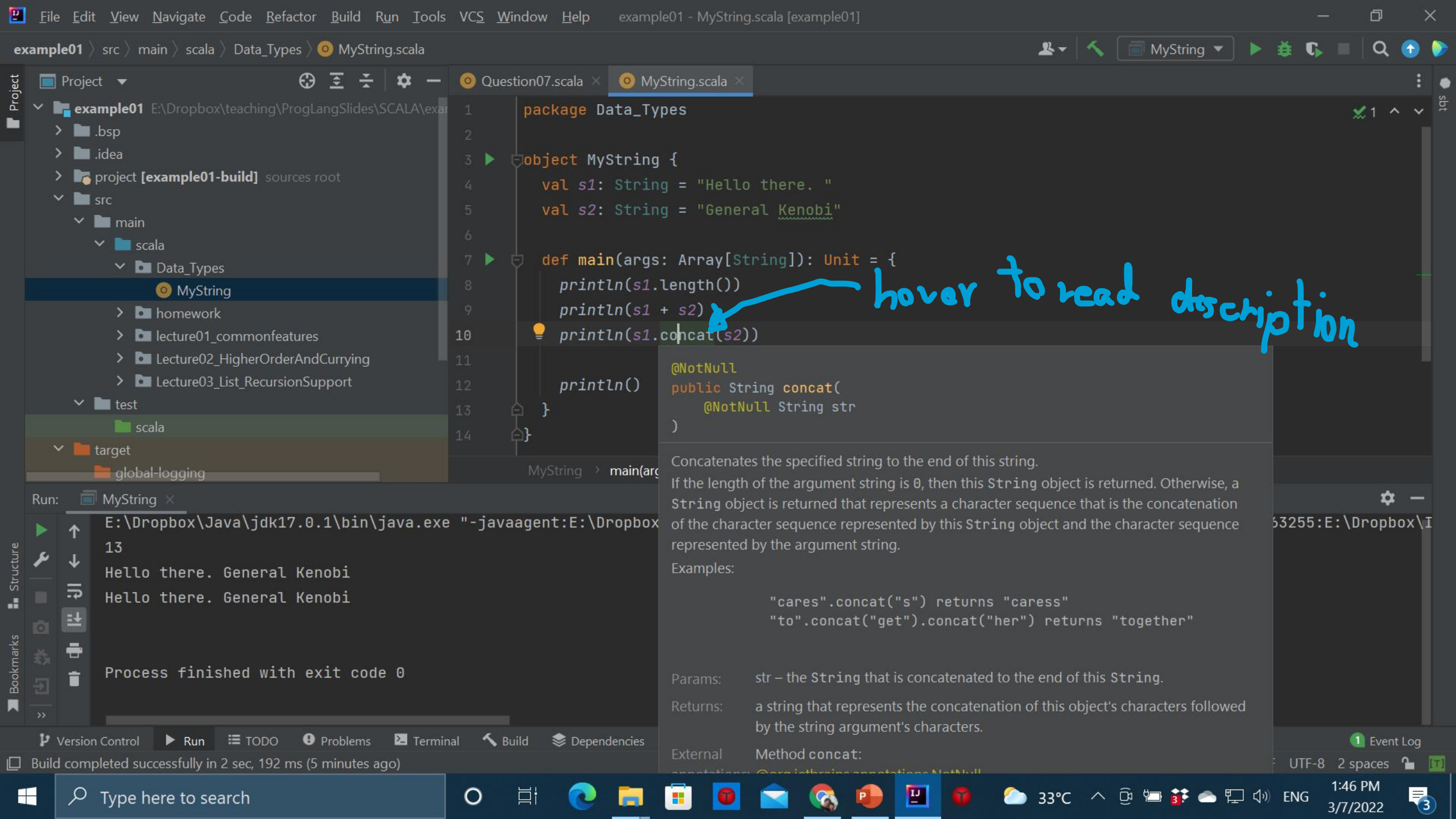
- A sequence of characters.
- It's actually from java.lang.String. It's a Java class.
- A string is immutable.

```
object MyString {
  val s1: String = "Hello there. "
  val s2: String = "General Kenobi"
  val n1 = 66;
  val n2 = 98.45

  def main(args: Array[String]): Unit = {
    println(s1.length())
①   println(s1 + s2)
②   println(s1.concat(s2))

    printf("%s: Order (%d) ,has been %f percent completed.", s1, n1,n2)
    val result = printf("%s: Order (%d) ,has been %f percent completed.", s1, n1,n2)
    println(result)
    println("%s: Order (%d) ,has been %f percent completed.".format(s1, n1,n2))


  }
}
```

result คือ "ฟังก์ชัน"

ปริ้นสิ่งที่ฟังก์ชันทำ

```
13
①Hello there. General Kenobi
②Hello there. General Kenobi
Hello there. : Order (66) ,has been 98.450000 percent completed.Hello there. : Order (66) ,has been 98.450000 percent completed.()
Hello there. : Order (66) ,has been 98.450000 percent completed.
```


File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
example01 - MyString.scala [example01]
example01 src main scala Data_Types MyString.scala
MyString
Project
example01 E:\Dropbox\teaching\ProgLangSlides\SCALA\exa
.bsp
.idea
project [example01-build] sources root
src
main
scala
Data_Types
MyString
homework
lecture01_commonfeatures
Lecture02_HigherOrderAndCurrying
Lecture03_List_RecursionSupport
test
scala
target
global-logging
Question07.scala
MyString.scala
1 package Data_Types
2
3 object MyString {
4 val s1: String = "Hello there. "
5 val s2: String = "General Kenobi"
6
7 def main(args: Array[String]): Unit = {
8 println(s1.length())
9 println(s1 + s2)
10 println(s1.concat(s2))
11
12 println()
13 }
14 }
hover to read description
@NotNull
public String concat(
@NotNull String str
)
Concatenates the specified string to the end of this string.
If the length of the argument string is 0, then this String object is returned. Otherwise, a String object is returned that represents a character sequence that is the concatenation of the character sequence represented by this String object and the character sequence represented by the argument string.
Examples:
"cares".concat("s") returns "caress"
"to".concat("get").concat("her") returns "together"
Params: str – the String that is concatenated to the end of this String.
Returns: a string that represents the concatenation of this object's characters followed by the string argument's characters.
External Method concat:
MyString main(arg
Run: MyString
E:\Dropbox\Java\jdk17.0.1\bin\java.exe "-javaagent:E:\Dropbox
13
Hello there. General Kenobi
Hello there. General Kenobi
Process finished with exit code 0
Version Control Run TODO Problems Terminal Build Dependencies
Event Log
Build completed successfully in 2 sec, 192 ms (5 minutes ago)
UTF-8 2 spaces
Type here to search
33°C
ENG
1:46 PM
3/7/2022

ARRAY

- Store fixed size sequential data (must have the same type)
- Default value for a slot depends on its data type.

```
object MyArray {
  val a: Array[Int] = new Array[Int](10)
  var b = Array(1,2,3,4)  //initializer list

  def main(args: Array[String]): Unit = {
    println(a) //will print address

    for(i <- 0 ≤ .to( ≤ a.length-1)) { // print default values
      print(a(i) + ", ")
    }
    println("------------------------------")
    for(i <- 0 ≤ .until( < a.length)) { // using "to" -> a.length-1
      a(i) = i
    }
    for(i <- 0 ≤ .until( < a.length)) {
      print(a(i) + ", ")
    }
    println("------------------------------")
    for(x <- a){  //for each
      print(x + ", ")
    }


    }
}
```

0 ถึง a.length - 1

assign ค่าใหม่

```
0, 0, 0, 0, 0, 0, 0, 0, 0, 0, ----------
0, 1, 2, 3, 4, 5, 6, 7, 8, 9, ----------
0, 1, 2, 3, 4, 5, 6, 7, 8, 9,
```

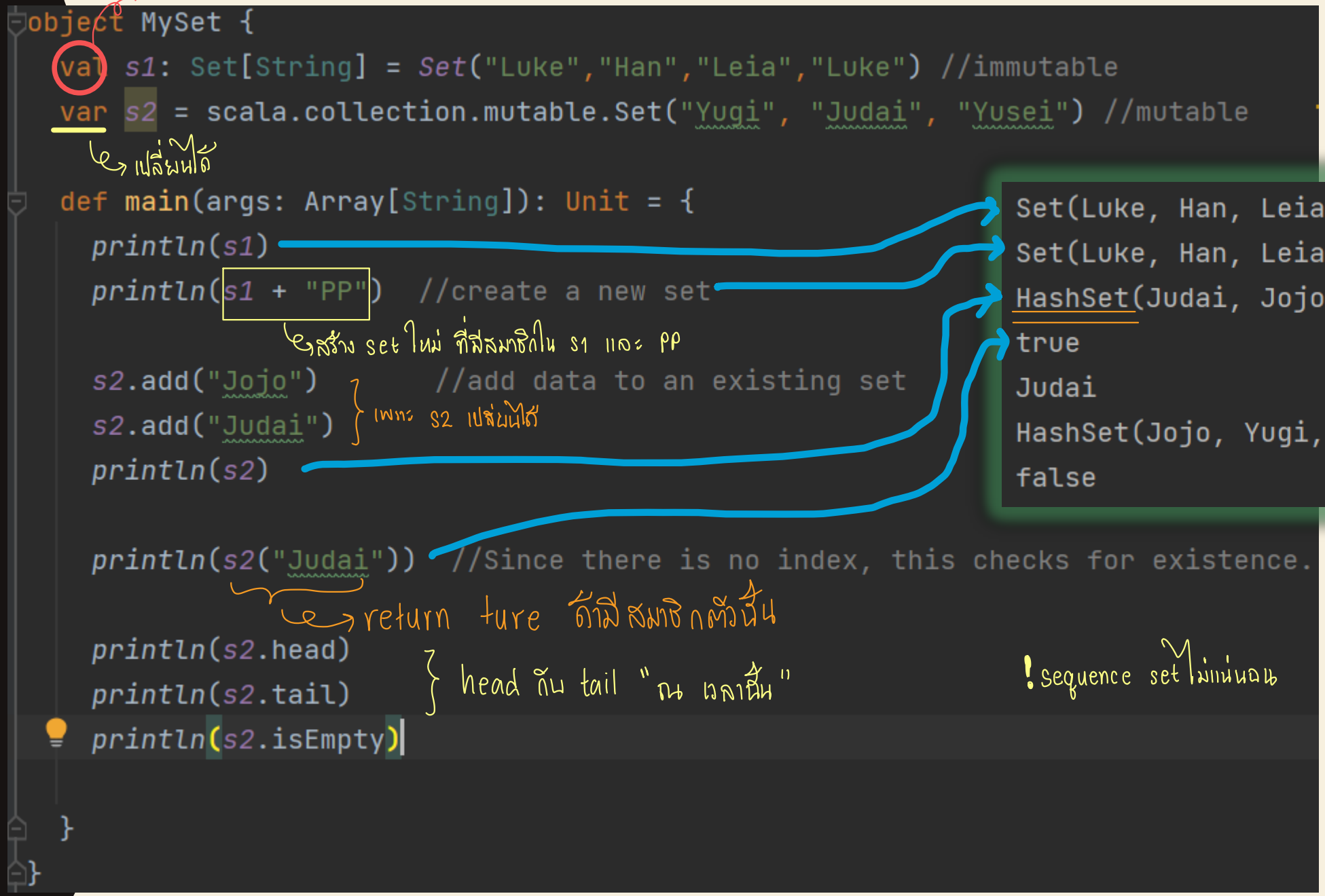
ARRAY MAY NEED "IMPORT"

```
import Array._
object MyArray02 {

  val ar1 = Array("Luke","Han","Leia")
  val ar2 = Array("Yugi", "Judai", "Yusei")

  def main(args: Array[String]): Unit = {
    val c = concat(ar1,ar2)
    for(x <- c){  //for each
      print(x + ", ")
    }
  }
}
```

```
Luke, Han, Leia, Yugi, Judai, Yusei,
```

SET

- Collection of non-duplicated data.
- They have to have the same data type.
- By default, set is immutable.
- Set is not ordered.
  - So its member does not have index.

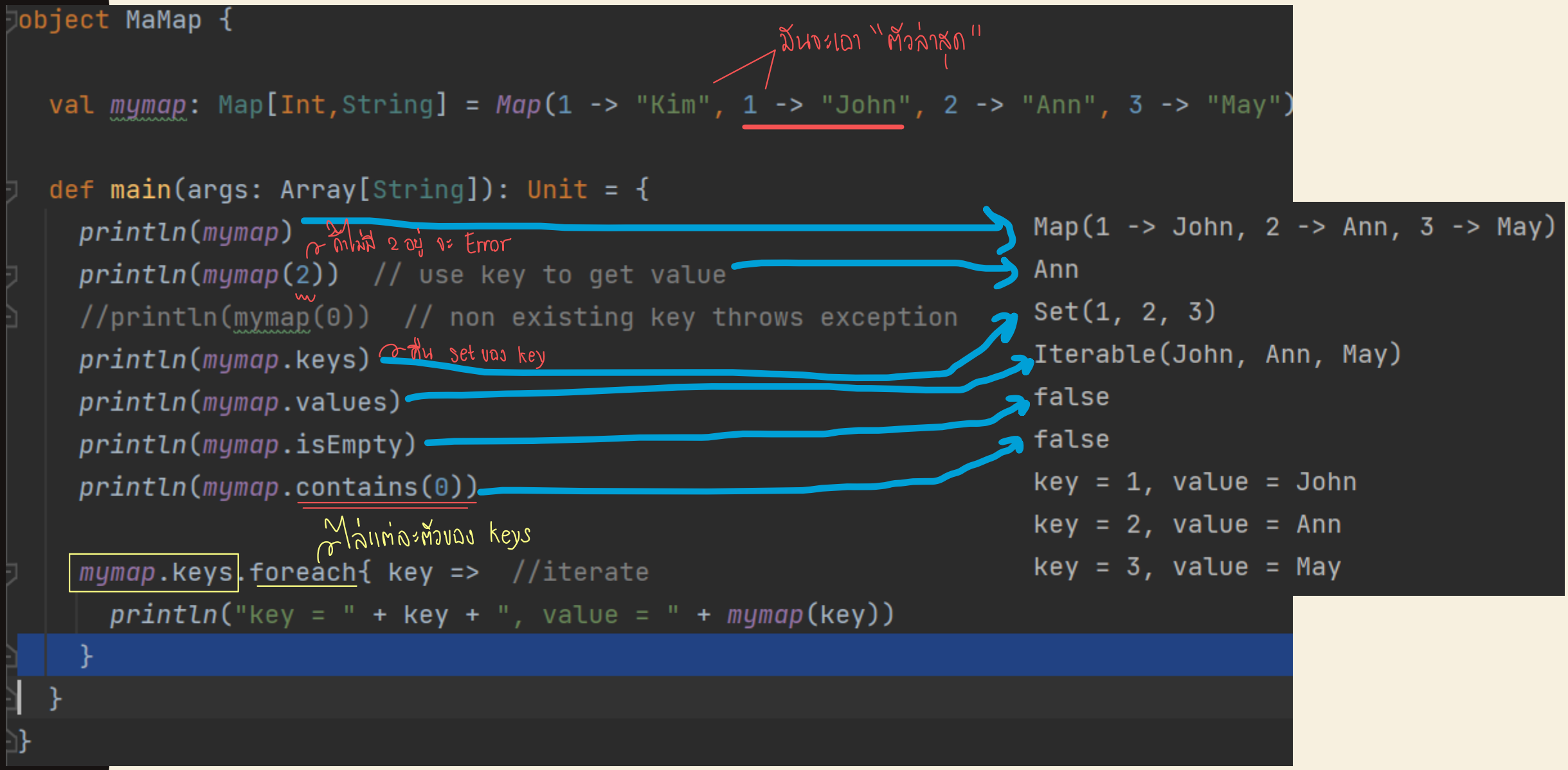

เปลี่ยนไม่ได้
object MySet {
val s1: Set[String] = Set("Luke","Han","Leia","Luke") //immutable
var s2 = scala.collection.mutable.Set("Yugi", "Judai", "Yusei") //mutable
เปลี่ยนได้
def main(args: Array[String]): Unit = {
println(s1)
println(s1 + "PP") //create a new set
สร้าง set ใหม่ ที่มีสมาชิกใน s1 และ PP
s2.add("Jojo") //add data to an existing set
s2.add("Judai")
เพราะ s2 เปลี่ยนได้
println(s2)
println(s2("Judai")) //Since there is no index, this checks for existence.
return ture ถ้ามีสมาชิกตัวนี้
println(s2.head)
println(s2.tail)
head กับ tail "ณ เวลานี้"
println(s2.isEmpty)
}
}
Set(Luke, Han, Leia)
Set(Luke, Han, Leia, PP)
HashSet(Judai, Jojo, Yugi, Yusei)
true
Judai
HashSet(Jojo, Yugi, Yusei)
false
! Sequence set ไม่แน่นอน

```scala
object MySet02 {
  val s1: Set[String] = Set("Luke","Han","Leia","Luke") //immutable
  var s2 = scala.collection.mutable.Set("Vader", "Luke", "Chewy","Han") //mutable

  def main(args: Array[String]): Unit = {
    println(s1 ++ s2) //union into new set ===> s1.++(s2)
    println(s1.&(s2)) //intersect into new set ===> s1.intersect(s2)
    println(s1.max)   // max value
    println(s1.diff(s2)) //difference into new set
    println("---------------")
    s2.foreach(println) //for loop of a set
    println("---------------")
    for(x <- s2){       //normal foreach
      println(x)
    }
  }
}
```

```
HashSet(Luke, Chewy, Vader, Han, Leia)
Set(Luke, Han)
Luke
Set(Leia)
---------------
Chewy
Han
Luke
Vader
---------------
Chewy
Han
Luke
Vader
```

MAP

- A collection of (key, value) pairs.
- A key is unique.
- you can choose between mutable/immutable map.

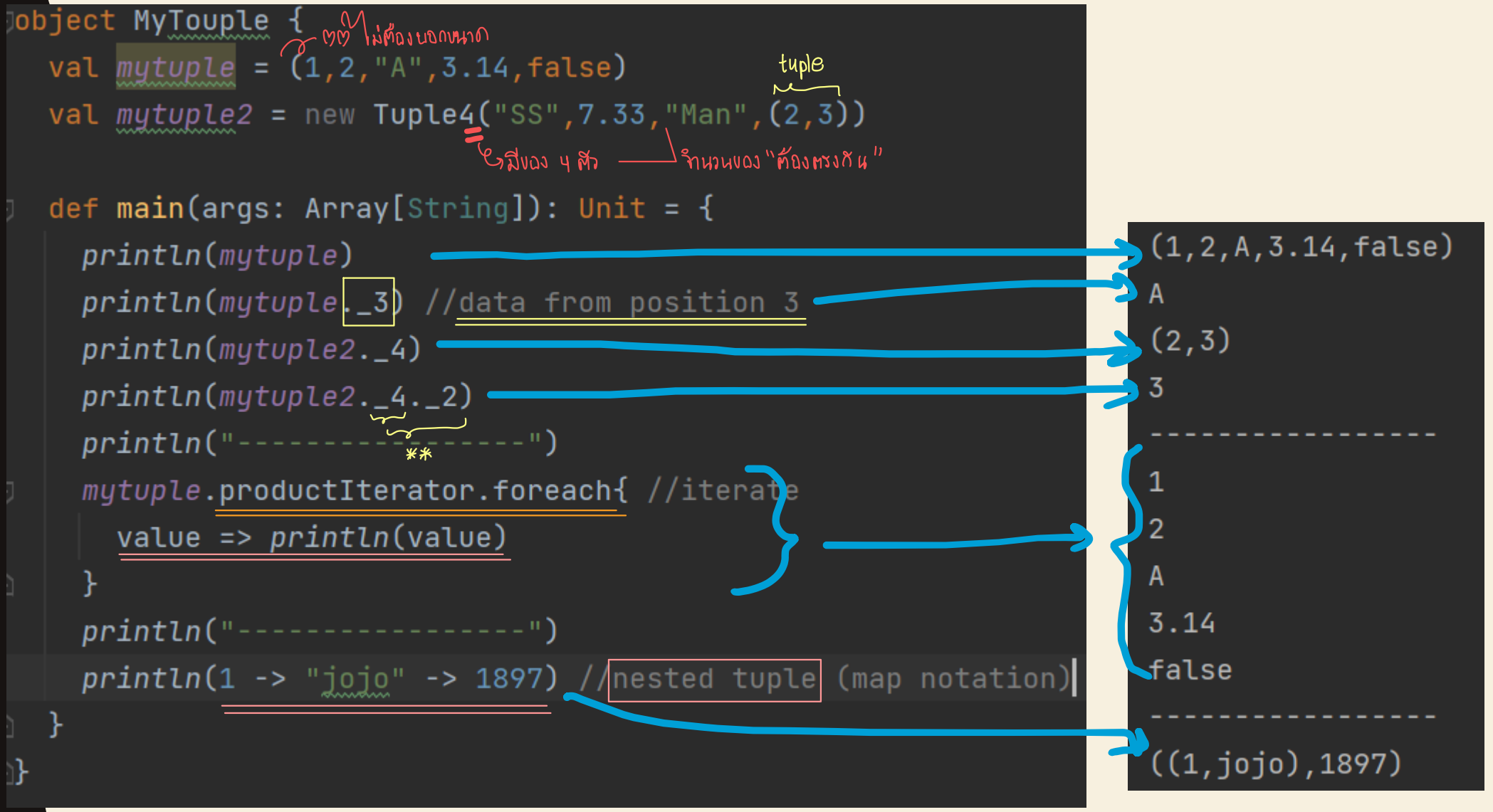

object MaMap {
val mymap: Map[Int,String] = Map(1 -> "Kim", 1 -> "John", 2 -> "Ann", 3 -> "May")
มันจะเอา "ตัวล่าสุด"
def main(args: Array[String]): Unit = {
println(mymap)
println(mymap(2)) // use key to get value
//println(mymap(0)) // non existing key throws exception
println(mymap.keys)
println(mymap.values)
println(mymap.isEmpty)
println(mymap.contains(0))
mymap.keys.foreach{ key => //iterate
println("key = " + key + ", value = " + mymap(key))
}
}
}
Map(1 -> John, 2 -> Ann, 3 -> May)
Ann
Set(1, 2, 3)
Iterable(John, Ann, May)
false
false
key = 1, value = John
key = 2, value = Ann
key = 3, value = May

```scala
object MyMap02 {
  val m1: Map[Int,String] = Map(1 -> "John", 2 -> "Ann", 3 -> "May")
  val m2 = Map(2 -> "Kim", 4 -> "Lee", 1 -> "Ann", 5 -> "Penguin")


  def main(args: Array[String]): Unit = {
    println(m1 ++ m2) // concat
    println(m1.head)
    println(m1.tail)
    println(m1.size)

  }
}
```

```
HashMap(5 -> Penguin, 1 -> Ann, 2 -> Kim, 3 -> May, 4 -> Le
(1,John)
Map(2 -> Ann, 3 -> May)
3
```

TUPLE

- Collection of values.
- Can contain different data type.
- Tuple is immutable!
- Each touple can only contain upto 22 data.
- Position in a tuple starts from 1. → เริ่มที่เลข 1 (ไม่ใช่! 0 นะ)
- Data in a map is actually a tuple.

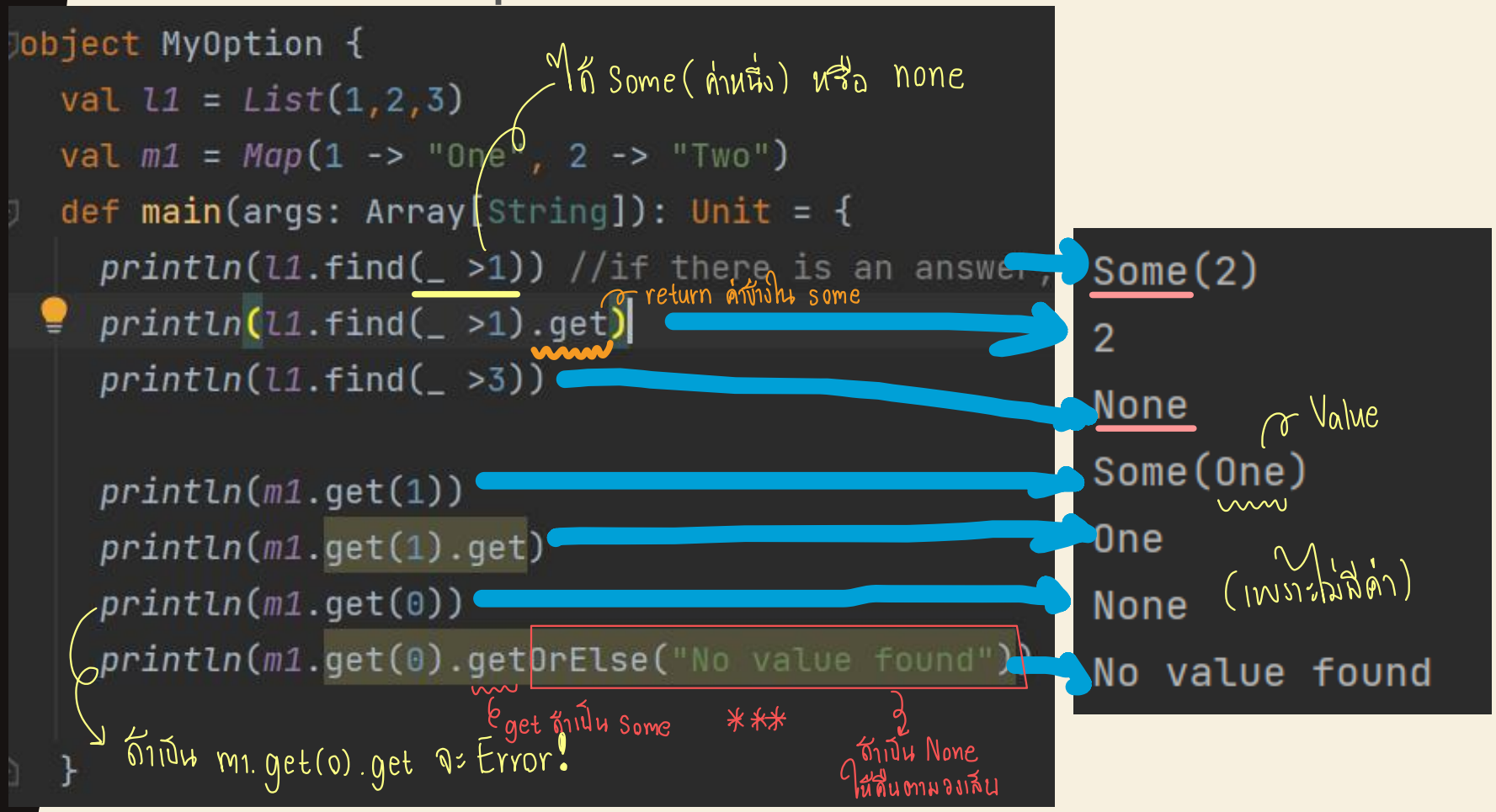

object MyTouple {
val mytuple = (1,2,"A",3.14,false)
val mytuple2 = new Tuple4("SS",7.33,"Man",(2,3))
def main(args: Array[String]): Unit = {
println(mytuple)
println(mytuple._3) //data from position 3
println(mytuple2._4)
println(mytuple2._4._2)
println("-----------------")
mytuple.productIterator.foreach{ //iterate
value => println(value)
}
println("-----------------")
println(1 -> "jojo" -> 1897) //nested tuple (map notation)
}
}
(1,2,A,3.14,false)
A
(2,3)
3

1
2
A
3.14
false

((1,jojo),1897)

OPTION TYPE

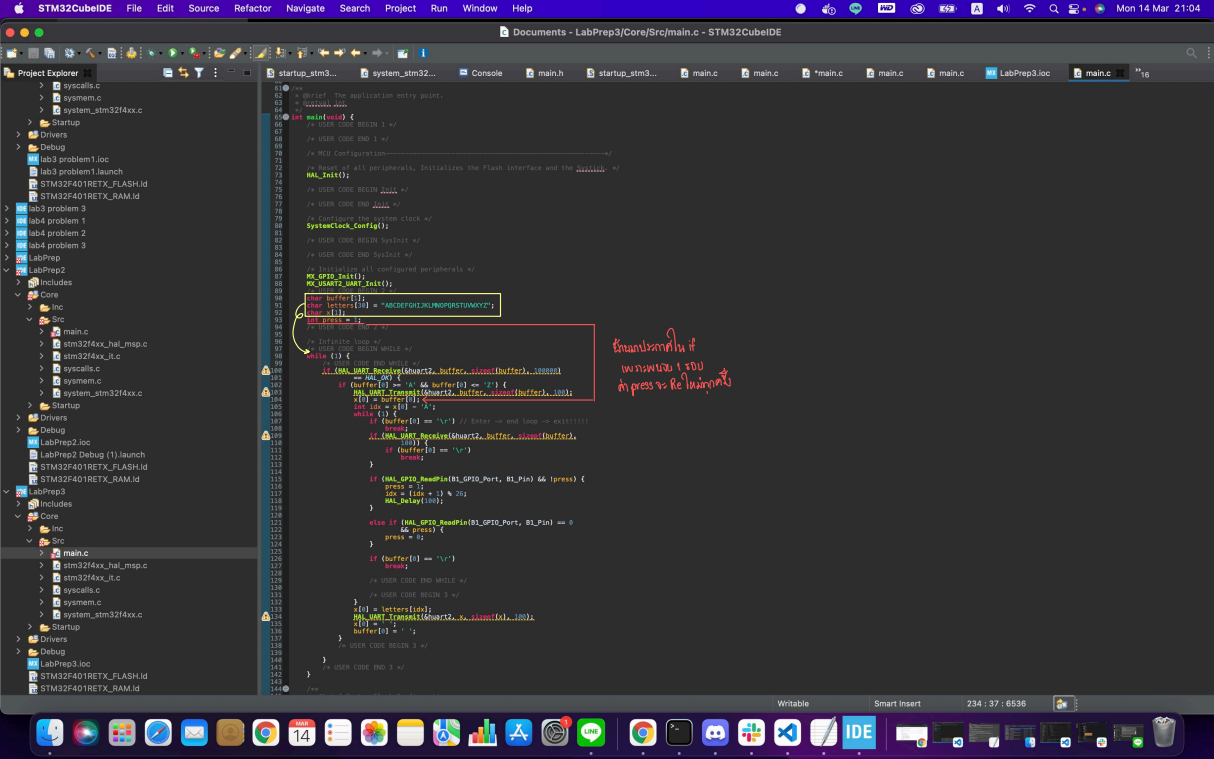
- Normally used as a return type
  - For example: return an answer or None



```scala
object MyOption2 {
  val l1 = List(1,2,3)
  val opt1: Option[Int] = None
  val opt2: Option[Int] = Some(2)

  def findPos(v:Int, l:List[Int]): Option[Int] ={
    return findPos(v,l, count = 0)
  }

  def findPos(v:Int, l:List[Int], count: Int):Option[Int] ={
    if(l.isEmpty) return None
    if(v == l.head) return Some(count)
    else {
      return findPos(v,l.tail,count+1)
    }
  }

  def main(args: Array[String]): Unit = {
    println(opt1.isEmpty)
    println(opt1.getOrElse("NO"))
    println(findPos(2,l1))
    println(findPos(4,l1))
  }
```

```
true
NO
Some(1)
None
```

index


Documents - LabPrep3/Core/Src/main.c - STM32CubeIDE
Project Explorer
main.c
ถ้าหมดประโยคที่ใน ที่
เพราะพอมัน 1 รอบ
ค่า press จะ ที่ ไม่หยุดเลยนี้