



Kotlin 2022

Bài 1

1

Khái quát Kotlin

2

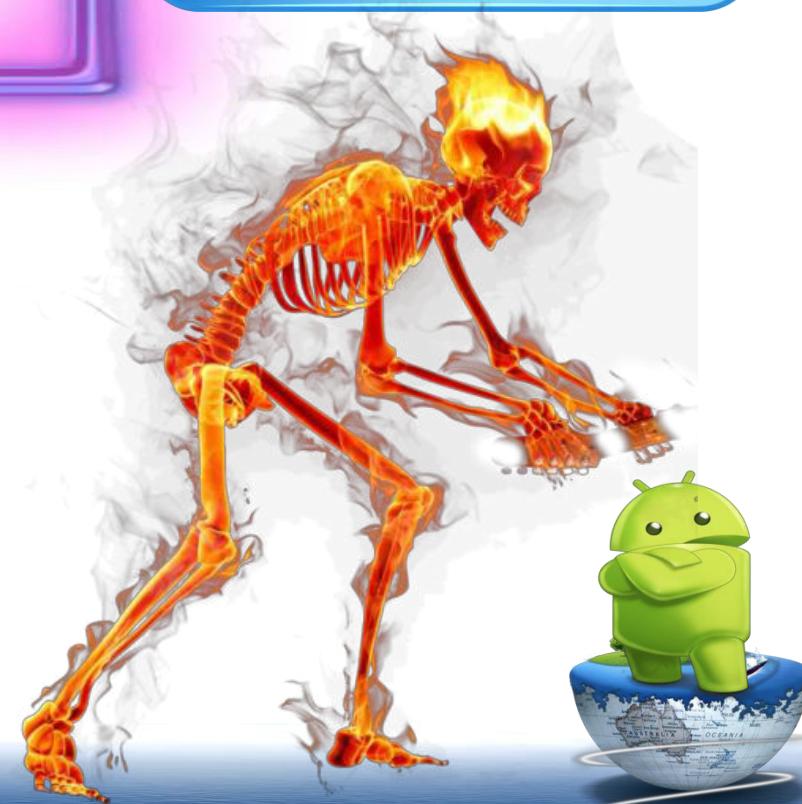
Setup JDK

3

Setup IntelliJ IDEA

4

Một số lệnh cơ bản



1

Khái quát Kotlin

❑ 1. Kotlin :

- ✓ *Kotlin là ngôn ngữ lập trình 100% tương thích với Java và Android.*
- ✓ *Kotlin được JetBrains công bố lần đầu tiên vào tháng 06/2011*
- ✓ *Vào ngày 17/05/2017, Google đã tuyên bố ưu tiên hỗ trợ hàng đầu cho Kotlin trên Android*

❑ Ưu điểm :

- ✓ *Code ngắn gọn, cấu trúc đơn giản, trực quan, rất dễ tiếp cận cũng như học hỏi đối với những người mới.*
- ✓ *Kotlin cho phép lập trình viên dùng thư viện và toàn bộ các nền tảng của Java*
- ✓ *Miễn phí: Kotlin là một dự án mã nguồn mở, miễn phí sử dụng.*



2

Setup JDK

- ✓ **JDK : (key tìm kiếm download jdk)**

<http://www.oracle.com/technetwork/java/javase/downloads/index.html>

[Java 18](#)[Java 17](#)**Java SE Development Kit 18.0.1.1 downloads**

Thank you for downloading this release of the Java™ Platform, Standard Edition Development Kit (JDK™). The JDK is a development environment for building applications and components using the Java programming language.

The JDK includes tools for developing and testing programs written in the Java programming language and running on the Java platform.

[Linux](#) [macOS](#) [Windows](#)

Product/file description	File size	Download
x64 Compressed Archive	172.8 MB	https://download.oracle.com/java/18/latest/jdk-18_windows-x64_bin.zip (sha256)
x64 Installer	153.38 MB	https://download.oracle.com/java/18/latest/jdk-18_windows-x64_bin.exe (sha256)
x64 MSI Installer	152.26 MB	https://download.oracle.com/java/18/latest/jdk-18_windows-x64_bin.msi (sha256)

```
C:\Users\kiensevt>java -version
java version "1.8.0_141"
Java(TM) SE Runtime Environment (build 1.8.0_141-b15)
Java HotSpot(TM) 64-Bit Server VM (build 25.141-b15, mixed mode)
```

Kiểm tra bằng cmd

3

Setup IntelliJ IDEA

- ✓ **IntelliJ IDEA Community Edition 2022.1.2 : (key tìm kiếm download intellij)**
<http://www.jetbrains.com/idea/download/index.html>

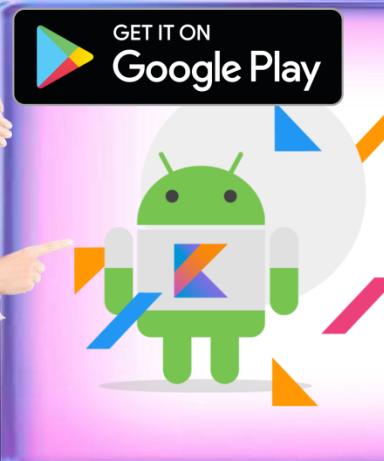
Download IntelliJ IDEA

Windows macOS Linux

Ultimate For web and enterprise development Download .exe ▾ Free 30-day trial available	Community For JVM and Android development Download .exe ▾ Free, built on open source
--	---

- ✓ **Link down file cài đặt JDK và IntelliJ khớp version với khóa học :**
<http://caidat.tuhoc.cc>





Kotlin 2022

Bài 2



1

Ghi chú

2

Xuất dữ liệu

3

1 số lý tự đặc biệt

Comments

//

Single line comment

/*

Multi-line comment



1

Ghi chú

□1. Why ?

- ✓ Ghi chú để cho các lập trình viên khác đọc Code dễ dàng hơn, hoặc chính bản thân chúng ta đọc để Debug

□2 . Các loại ghi chú cơ bản :

- ✓ Ghi chú 1 dòng :

```
//1. ghi chú trên 1 dòng
// phím tắt Ctrl + /
```



- ✓ Ghi chú nhiều dòng :

```
/*
Xin thông báo
Đây là ghi chú
trên nhiều dòng
phím tắt Shift+Ctrl+/
*/
```

- ✓ Ghi chú KDOC (ghi chú hàm)



2

Xuất dữ liệu

❑3 . Xuất dữ liệu trên từng dòng: `println`

Con trỏ nhảy xuống dưới dòng sau khi kết thúc

```
println("dong 1")  
println("dong 2")
```



```
dong 1  
dong 2
```

❑4 . Xuất dữ liệu trên cùng 1 dòng: `print`

Con trỏ ở cuối chuỗi sau khi xuất xong

```
print("dong 3")  
print("dong 4")  
print("dong 5")
```



```
dong 3dong 4dong 5
```



3

1 số ký tự đặc biệt

□5. Xuất ký tự đặc biệt

\t : Thụt vào 1 tab

```
//\t : thụt vào 1 tab
println()
println("\tTest thu /t")
```



```
dong 3dong 4dong 5
Test thu /t
```



\n : Enter xuống dòng

```
//\n : xuống dòng
print("hoc lap trinh tai, \n tuhoc.cc ")
```



```
hoc lap trinh tai,
tuhoc.cc
```

\": Dấu nháy kép

```
//\" : trích dẫn
println("Co nguoi noi \"cho di la con mai\"")
```



```
Co nguoi noi "cho di la con mai"
```

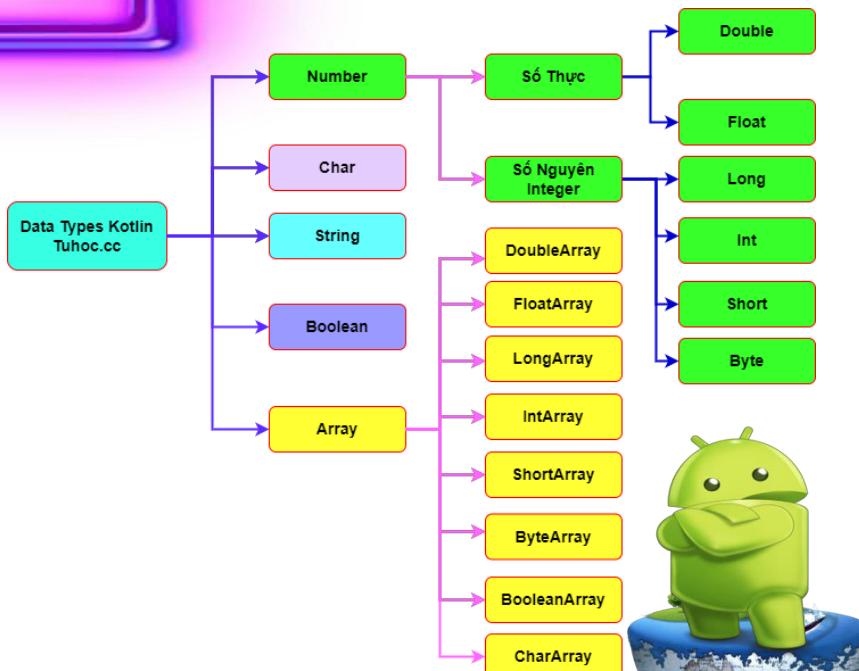




Kotlin 2022

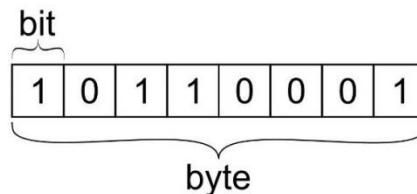
Bài 3

Biến , Hằng và các kiểu dữ liệu



1

Các kiểu dữ liệu Kotlin



Data Types Kotlin
Tuhoc.cc



Data Type	Size (bits)	Data Range
Byte	8 bit	-128 to 127
Short	16 bit	-32768 to 32767
Int	32 bit	-2,147,483,648 to 2,147,483,647
Long	64 bit	-9,223,372,036,854,775,808 to +9,223,372,036,854,775,807
Float	32 bit	1.40129846432481707e-45 to 3.40282346638528860e+38
Double	64 bit	4.94065645841246544e-324 to 1.79769313486231570e+308



2

Biến trong Kotlin

- ❑ Biến : Là tên gọi của 1 vùng nhớ, để lưu trữ dữ liệu
- ❑ Kotlin có 2 kiểu khai báo biến

var cho phép thay đổi giá trị của biến trong chương trình

val không cho phép thay đổi giá trị của biến trong (**Hàng số**)

Khai báo biến/ khởi tạo biến : **var tên_bien : kiểu [= giá_trị]**

```
//khởi tạo biến
var a:Int =113
var b:Short=8
var c[Byte]=1
```

```
//khai báo biến
var d:Int
var e:Short
var g[Byte]
```



```
//đặt $ trong phần xuất chuỗi để xuất giá trị của 1 biến
println("a = $a")
println("b = $b")
println("c = $c")
```



```
a = 113
b = 8
c = 1
```



2

Biến trong Kotlin

 Quy tắc đặt tên biến :

- ✓ *Tên biến chỉ chứa chữ cái (a-z, A-Z), chữ số (0-9), và dấu gạch chân _;*
- ✓ *Tên biến không được bắt đầu bằng chữ số ;*
- ✓ *Tên biến phân biệt hoa / thường.*
- ✓ *Tuân theo quy ước đặt tên biến theo kiểu camelCase : Lạc đà (chữ cái đầu từ thứ 2 trở đi viết Hoa , còn lại viết thường)*

```
val Pi = 3.14
val nhietDoSoi = 100
val nhietDoDong = 0
```

- ✓ *Kotlin cho phép (nhưng không khuyến khích) sử dụng các chữ cái Unicode trong định danh, nghĩa là bạn có thể đặt tên biến bằng tiếng Việt có dấu nếu muốn*

```
var biến:Int
var giáThuốcTây:Int = 60000
```



3

Chi tiết kiểu dữ liệu Kotlin

□1 . Kiểu số thực : Double , Float

Double có 2 cách khai báo

```
var y:Double=9.5 // khai báo tường minh
var z = 9.5      // khai báo tắt
//kiểm tra kiểu loại của biến
println("kiểu loại của y: ${z::class.java.typeName}")
println("kiểu loại của z: ${z::class.java.typeName}")
```



kiểu loại của y: double
kiểu loại của z: double

Float cần khai báo thêm chữ *f* hoặc *F*

```
var f:Float=8.5f // cần khai báo thêm chữ f hoặc F
println("giá trị của f là: $f")
```



giá trị của f là: 8.5



3

Chi tiết kiểu dữ liệu Kotlin

□2 . Kiểu số nguyên : Long, Int, Short, Byte

- ✓ Long thêm *L* phía sau (không dùng *l* thường)

```
var soA:Long =999L // khai báo tường minh
var soB:Long =99l // (l thường báo lỗi )
var soC =10000L //khai báo tắt, không có L sẽ thành kiểu Int
println("kiểu loại của soA: ${soA::class.java.typeName}")
println("kiểu loại của soC: ${soC::class.java.typeName}")
```



kiểu loại của soA: long
kiểu loại của soC: long

- ✓ Int có 2 cách khai báo

```
//kiểu Int
var soD =99 // khai báo tắt
var soE:Int =99 // khai báo tường minh
println("kiểu loại của soD: ${soD::class.java.typeName}")
println("kiểu loại của soE: ${soE::class.java.typeName}")
```



kiểu loại của soD: int
kiểu loại của soE: int

- ✓ Short, Byte

```
var soF:Short =32767 // khai báo tường minh
var soG:Short =32768 // vượt giới hạn của short, báo lỗi
var soH[Byte] =127 //
```



3

Chi tiết kiểu dữ liệu Kotlin

□3 . Kiểu ký tự: Char

✓ Dùng để lưu trữ một ký tự nằm trong nháy đơn

```
var kytu:Char = 'a'  
println(kytu::class.java.typeName)
```



char

□4 . Kiểu chuỗi : String

✓ Dùng để lưu trữ tập các ký tự nằm trong nháy kép

```
var str1 = "E tự học lập trình"  
println(str1)
```



E tự học lập trình

✓ Có thể khai báo 1 đoạn văn, thơ bằng bộ 3 nháy kép

```
var str2 = """Thân em như tẩm lụa đào,  
Phất phơ giữa chợ biết vào tay ai"""  
println(str2)
```



Thân em như tẩm lụa đào,
Phất phơ giữa chợ biết vào tay ai



3

Chi tiết kiểu dữ liệu Kotlin

□5 . Kiểu luận lý: Boolean

- ✓ Chỉ có 2 giá trị **True** hoặc **False**, dùng cho việc kiểm tra điều kiện

```
//boolean  
var check:Boolean = false
```

□6 . Kiểu mảng : Array

var tên_mảng : typeArray = typeArrayOf(giá trị 1, 2, 3, 4 ...)

Lưu ý: arrayOf chữ a viết thường

```
var mangSoThuc :FloatArray= floatArrayOf(0.5f,1.3f,9.1f,7.3f)  
var mangKyTu :CharArray= charArrayOf('a','b','c','x')  
var mangSoNghiem :IntArray= intArrayOf(1,2,3,4,5)
```

□7 . Hằng số (không thể thay đổi giá trị) : val

- ✓ **Hằng số là giá trị bất biến, không thay đổi được trong chương trình**
- ✓ **Nếu ta cố tình sửa, chương trình sẽ báo lỗi**

```
val Pi = 3.14  
val nhietDoSoi = 100  
val nhietDoDong = 0
```



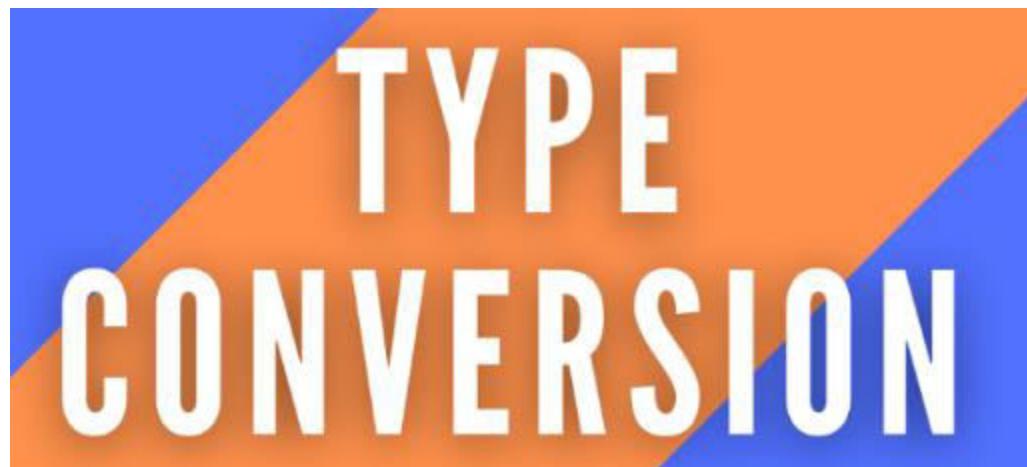


Kotlin 2022

Bài 4

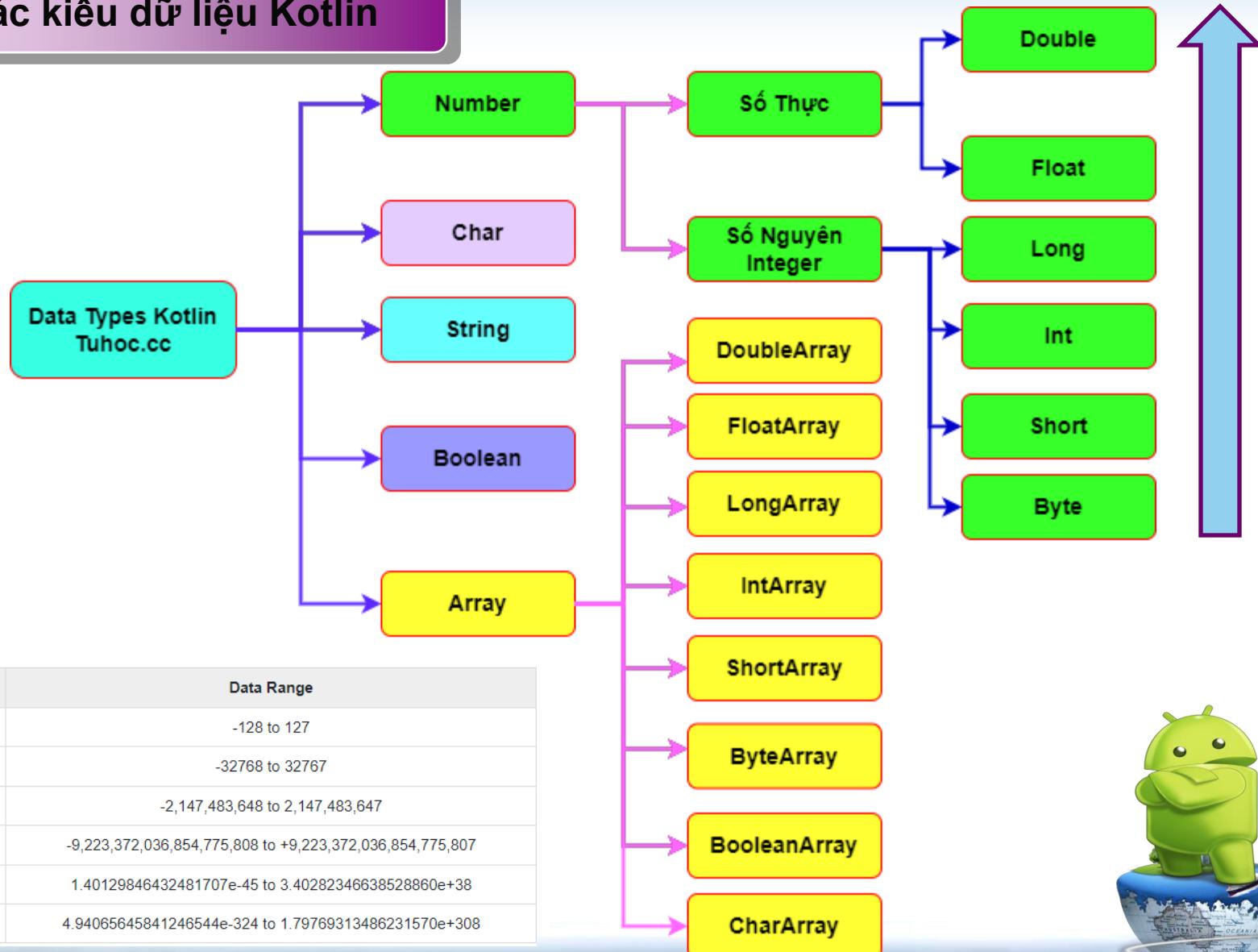


Ép kiểu dữ liệu - Kotlin Type Conversion



1

Các kiểu dữ liệu Kotlin



2

Tại sao phải ép kiểu

- * Như bài học trước, chúng ta đã biết các kiểu dữ liệu sinh ra, để phù hợp với mục đích sử dụng, tiết kiệm bộ nhớ hệ thống.
- * Trong quá trình tính toán, có thể kiểu dữ liệu trả về không còn giống với kiểu ban đầu chúng ta khai báo => Cần ép kiểu để sử dụng kiểu phù hợp

 Một số dạng chuyển đổi thường dùng

- ✓ **toByte():** chuyển đổi sang kiểu Byte
- ✓ **toShort():** chuyển đổi sang kiểu Short
- ✓ **toInt():** chuyển đổi sang kiểu Int
- ✓ **toLong():** chuyển đổi sang kiểu Long
- ✓ **toFloat():** chuyển đổi sang kiểu Float
- ✓ **toDouble():** chuyển đổi sang kiểu Double
- ✓ **toChar():** chuyển đổi sang kiểu Char



3

Các loại ép kiểu Kotlin

- ☐ Kotlin Có 2 dạng ép kiểu



Ép kiểu rộng :

ép kiểu từ dữ liệu bé > lớn

Ex: Int → Long → Float → Double

Không lo mất dữ liệu



Ép kiểu hẹp :

ép kiểu từ dữ liệu lớn > bé

Ex: double → float → long → int

Có thể mất dữ liệu



4

Ép kiểu rộng

**Ép kiểu rộng :****ép kiểu từ dữ liệu bé > lớn****Ex: Int → Long → Float → Double**

Không lo mất dữ liệu

```
var soA:Int = 20  
var soB:Long = soA //error , kotlin không cho ép, cho dù là long > int
```

Trong kotlin bắt buộc phải ép tường minh (không giống C# có thể ép tắt)

```
var soB:Long = soA.toLong()  
//check kiểu loại soB  
println(soB::class.java.typeName)
```



Long



5

Ép kiểu hẹp



Ép kiểu hẹp :

ép kiểu từ dữ liệu lớn > bé

Ex: double → float → long → int

Có thể mất dữ liệu

```
var x:Short =32767
//Nếu cố tình ép, sử dụng ép tường minh, chú ý khả năng mất dữ liệu
var y[Byte] =x.toByte()
println(y) //trả về -1
```

```
//ví dụ 2 :
var diemToan: Float =7.8f
var diemVan:Int = diemToan.toInt()
println(diemToan) // trả về 7.8
println(diemVan) // trả về 7 (mất dữ liệu)
```

```
var m:Short =120
var n[Byte] =m.toByte()
println(n) //trả về 120 do Byte vẫn đủ sức chứa
```





- 1 Các phép toán cơ bản
- 2 Bài Tập Vận Dụng
- 3 Toán tử tiền tố 1 ngôi



1

Các phép toán cơ bản

Ký hiệu (Cách 1)	Giải thích	Phương thức (Cách 2)	Cách biểu đạt	Kết quả
+	Phép cộng	a.plus(b)	$10 + 1$ $10.plus(1)$	11
-	Phép trừ	a.minus(b)	$50 - 23$ $50.minus(23)$	27
*	Phép nhân	times	$2 * 8$ $2.times(8)$	16
/	Phép chia	div	$11 / 5$ $11.div(5)$	2.2
%	Phép chia lấy dư	rem	$34 \% 10$ $34.rem(10)$	4



1

Các phép toán cơ bản

```

var a = 10 // khai báo tắt, a sẽ nhận kiểu Int
var b = 3 // khai báo tắt, a sẽ nhận kiểu Int
println(a::class.java.typeName)
println(b::class.java.typeName)
//phép cộng
println("cách 1 a+b= " +(a+b))
println("cách 2 a+b= " + (a.plus(b)))
//phép trừ
println("cách 1 a-b= " +(a-b))
println("cách 2 a-b= " + (a.minus(b)))
//phép nhân
println("cách 1 a*b= " +(a*b))
println("cách 2 a*b= " + (a.times(b)))
//phép chia thân trong
println("cách 1 a/b= " +(a/b))
println("cách 2 a/b= " + (a.div(b)))
var kq:Float = a.toFloat()/b
println("a/b =" +kq)
//chia lấy dư
println("cách 1 a/b= " +(a%b))
println("cách 2 a/b= " + (a.rem(b)))

```

Thận trọng với các phép tính có kết quả khác với kiểu dữ liệu ban đầu của các biến trong phép toán



```

cách 1 a+b= 13
cách 2 a+b= 13
cách 1 a-b= 7
cách 2 a-b= 7
cách 1 a*b= 30
cách 2 a*b= 30
cách 1 a/b= 3
cách 2 a/b= 3
a/b =3.3333333
cách 1 a/b= 1
cách 2 a/b= 1

```



2

Bài Tập Vận Dụng

Cho các biến với giá trị

i1 = 2

i2 = 5

i3 = -3

d1 = 2.0

d2 = 5.0

d3 = -0.5

Cho biết kết quả của các lệnh sau:

(a) $i1 + (i2 * i3)$

(b) $i1 * (i2 + i3)$

(c) $i1 / (i2 + i3)$

(e) $i1 / i2 + i3$

(g) $3 + 4 + 5 / 3$

(i) $(3 + 4 + 5) / 3$

(k) $d1 + (d2 * d3)$

(l) $d1 + d2 * d3$

(m) $d1 / d2 - d3$

(n) $d1 / (d2 - d3)$

(o) $d1 + d2 + d3 / 3$

(p) $(d1 + d2 + d3) / 3$

(q) $d1 + d2 + (d3 / 3)$

(r) $3 * (d1 + d2) * (d1 - d3)$



3

Toán tử tiền tố 1 ngôi

Toán tử	Giải thích	Phương thức	Cách biểu đạt	Kết quả
+	Số dương	a.unaryPlus()	var x:Float = 9.2f var z=x.unaryPlus()	9.2
-	Số âm	a.unaryMinus()	var x:Float = 9.2f var y=x.unaryMinus()	-9.2

```

var x:Float = 9.2f
//+ Trừ một ngôi (đảo dấu)
var y=x.unaryMinus() // thêm dấu -
println("y=" +y) // y = -9.2
//+ Cộng một ngôi
var z=x.unaryPlus() // thêm dấu +
println("z=" +z) // z = 9.2
  
```



y=-9.2
z=9.2





Kotlin

Full Course

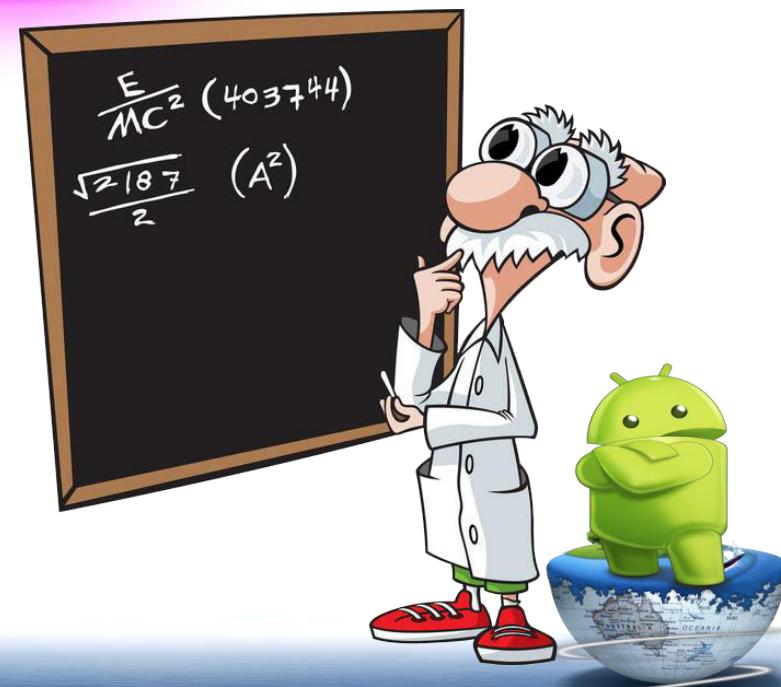
Bài 6

1 Toán tử gán = += -= *= /=

2 Các phép so sánh

Phép tính	Giải thích
=	Gán bằng
+=	Gán cộng
-=	Gán trừ
*=	Gán nhân
/=	Gán chia
%=	Gán mod

Ký hiệu	Giải thích
==	Bằng nhau
!=	Khác nhau
>	Lớn hơn
<	Nhỏ hơn
>=	Lớn hơn hoặc bằng
<=	Nhỏ hơn hoặc bằng



1

Toán tử gán = += -= *= /=

Phép tính	Giải thích	Công thức	Ý nghĩa
=	Gán bằng	$x = 1$	$x = 1$
+=	Gán cộng	$x += y$	$x = x + y$
-=	Gán trừ	$x -= y$	$x = x - y$
*=	Gán nhân	$x *= y$	$x = x * y$
/=	Gán chia	$x /= y$	$x = x / y$
%=	Gán chia lấy dư	$x \%= y$	$x = x \% y$



1

Các phép toán cơ bản

```
//gán cộng
var x=8 // var x:Int =8
println(x::class.java.typeName)
x+=5 // x=x+5
println(x)
```

int
13

```
//gán trừ
var x=8 // var x:Int =8
println(x::class.java.typeName)
x-=5 // x=x-5
println(x)
```

int
3

```
//gán nhân
var x=8 // var x:Int =8
println(x::class.java.typeName)
x*=5 // x=x*5
println(x)
```

int
40

```
//gán chia
var x=8 // var x:Int =8
println(x::class.java.typeName)
x/=5 // x=x/5
println(x)
```

int
1

```
//gán chia lấy dư
var x=8 // var x:Int =8
println(x::class.java.typeName)
x%=5 // x=x%5
println(x)
```

int
3

2

Các phép so sánh

Ký hiệu	Giải thích	Phương thức	Cách biểu đạt	Kết quả
<code>==</code>	Bằng nhau	<code>a.equals(b)</code>	<code>1 == 1</code>	True
<code>!=</code>	Khác nhau	<code>!a.equals(b)</code>	<code>1 != 1</code>	False
<code>></code>	Lớn hơn	<code>a.compareTo(b) > 0</code>	<code>1 > 2</code>	False
<code><</code>	Nhỏ hơn	<code>a.compareTo(b) < 0</code>	<code>2 < 1</code>	False
<code>>=</code>	Lớn hơn hoặc bằng	<code>a.compareTo(b) >= 0</code>	<code>2 >= 1</code>	True
<code><=</code>	Nhỏ hơn hoặc bằng	<code>a.compareTo(b) <= 0</code>	<code>2 <= 2</code>	True

```
//so sánh
var soA=1
var soB=3
println(soA==soB) // trả về false
println(soA!=soB) // trả về true
println(soA>soB) // trả về false
```

```
println(soA.equals(soB)) // trả về false
println(!soA.equals(soB)) // trả về true
println(soA.compareTo(soB)>0) // trả về false
println(soA.compareTo(soB)<0) // trả về true
println(soA.compareTo(soB)>=0) // trả về false
println(soA.compareTo(soB)<=0) // trả về true
println(soA.compareTo(soB)) // trả về -1
```



3

Bài tập vận dụng

 Bài tập Kotlin 01:

Khai báo biến $a = 6$,

thực hiện các phép tính sau đây và in kết quả ra màn hình.

A) $a+=3$?

B) $a-=5$?

C) $a*=2$?

D) $a\%=5$?

E) cho $b=2$, rút gọn biểu thức $a=a-(b+7)$





Kotlin

Full Course

Bài 7

1

Toán tử logic && ||

2

Toán tử tiền tố, hậu tố (prefix, postfix)

Ký hiệu	Giải thích
==	Toán tử logic AND (và) giữa 2 giá trị , trả về True khi cả 2 đều đúng (And nhiều giá trị tương tự)
	Toán tử logic OR (hoặc) giữa 2 giá trị , trả về False khi cả 2 đều sai (Or nhiều giá trị tương tự)
!	Phép phủ định

Ký hiệu	Giải thích
++	Tăng giá trị lên 1
--	Giảm giá trị đi 1



1

Toán tử logic **&&** **||**

Ký hiệu	Giải thích	Phương thức	Cách biểu đạt	Kết quả
&&	Toán tử logic AND (và) giữa 2 giá trị , trả về True khi cả 2 đều đúng (And nhiều giá trị tương tự)	a.and(b)	x = True, y = True x= False , y = True x= True , y = False	True False False
 	Toán tử logic OR (hoặc) giữa 2 giá trị , trả về Fasle khi cả 2 đều sai (Or nhiều giá trị tương tự)	a.or(b)	x = True, y = Fasle x = Fasle, y = True x = Fasle, y = False	True True False
!	Phép phủ định (Đảo ngược giá trị)	a.not()	X=True , !x X = Fasle , !x	Fasle True

```
var i=7
//kiểm tra i>0 và i<10 không
println(i>0 && i<10) //trả về true
//kiểm tra i<0 hoặc i<10 ?
println(i>0 || i<10) //trả về true
//phủ định
println(!(i>0 || i<10)) //trả về false
```

```
var i=7
//kiểm tra i>0 và i<10 không
println((i>0).and( other: i<10)) //trả về true
//kiểm tra i<0 hoặc i<10 ?
println((i>0).or( other: i<10)) //trả về true
//phủ định
println((i>0 || i<10).not()) //trả về false
```



2

Toán tử tiền tố, hậu tố (prefix, postfix)

Ký hiệu	Giải thích	Phương thức	Cách biểu đạt	Kết quả
++	Tăng giá trị lên 1	a.inc()	a=1, a++	a= 2
--	Giảm giá trị đi 1	a.dec()	a= 1, a--	a= 0

```

var x =99
var y =10
var z =77
var t =88
x++
y--
++z
--t
println(x) // trả về 100
println(y) // trả về 9
println(z) // trả về 78
println(t) // trả về 87
    
```

```

var a =8
var b:Int=a.inc()
println(a) // trả về 8
println(b) // trả về 9
    
```

Chú ý : Hàm inc() và dec() không thay đổi giá trị nội tại của biến, mà phải lưu sang 1 biến khác

Phép tính đơn lẻ, viết ++, -- trước hay sau đều được



2

Toán tử tiền tố, hậu tố (prefix, postfix)

Trường hợp biểu thức phức tạp thì tuân theo quy tắc sau:

Quy tắc viết dấu ++, --

a++, a-- (viết phía sau biến) => Postfix

++a, --a (viết trước biến) => Prefix

Ưu tiên tính toán Postfix, Prefix

Step 1 . Prefix

Step 2. Các phép toán còn lại

Step 3. Gán giá trị cho biến ở bên trái dấu bằng

Step 4. Tính postfix

```
var x=1
var y=2
var z = x++ - ++y +1
println(z) // trả về -1
println(x) // trả về 2
```

var x = 1

var y = 2

var z = x++ - ++y +1

Step 1: ++y => y = 3

Step 2: x=1, y = 3 => 1-3+1 =-1

Step 3: z= -1

Step 4: x++ => x = 2

Int z = 2 - 3 +1 =0

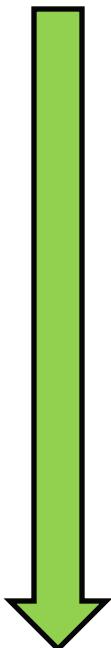
Phép tính sai



3

Ưu tiên toán tử

Kotlin có thứ tự ưu tiên toán tử trong biểu thức phức tạp, tuy nhiên tốt nhất là hãy tự đặt biểu thức trong ngoặc tròn () để thể hiện ưu tiên:



Ký hiệu	Giải thích
* / %	Phép nhân, chia, lấy phần dư
+ -	Toán tử Cộng, Trừ
<= < > >=	Các toán tử so sánh
== !=	Các toán tử so sánh
= %= /= -= += *=	Các toán tử gán
&&	Các toán tử logic





1

Ép kiểu dữ liệu cho String

2

Nhập liệu từ bàn phím

3

Bài tập kotlin 02,03



1

Ép kiểu dữ liệu cho String

Một số phương thức thông dụng để ép kiểu dữ liệu cho String :

Phương thức	Giải thích
<code>toBoolean()</code>	<i>Chuyển chuỗi về Boolean</i>
<code>toByte()</code>	<i>Chuyển chuỗi về Byte</i>
<code>toShort()</code>	<i>Chuyển chuỗi về Short</i>
<code>toInt()</code>	<i>Chuyển chuỗi về Int</i>
<code>toLong()</code>	<i>Chuyển chuỗi về Long</i>
<code>toFloat()</code>	<i>Chuyển chuỗi về Float</i>
<code>toDouble()</code>	<i>Chuyển chuỗi về Double</i>

```
var a1:Boolean="true".toBoolean()
var a2:Byte="2".toByte()
var a3:Short="20".toShort()
var a4:Int="200".toInt()
var a5:Long="2000".toLong()
var a6:Float="200.5".toFloat()
var a7:Double="2000.33".toDouble()
```



2

Nhập liệu từ bàn phím

Để nhập liệu từ bàn phím, Kotlin sử dụng hàm `readLine()`:

- Trả về **1 chuỗi dữ liệu** được nhập vào từ bàn phím

- Trả về **null** nếu không có dữ liệu

→ Từ chuỗi này, ta có thể ép kiểu chuỗi đó sang kiểu dữ liệu mong muốn

```
//nhập chuỗi từ bàn phím
println("mời cụ nhập vào tên:")
var ten:String? = readLine()
println("tên của cụ là: $ten")
```

```
//nhập số từ bàn phím
var soA:Int
println("mời nhập vào số a: ")
var s:String?= readLine()
if (s !=null)
{
    soA=s.toInt()
    println(soA)
}
```



3

Bài tập vận dụng

 Bài tập Kotlin 02: Tính chu vi, diện tích hình tròn

Viết chương trình nhập vào từ bàn phím bán kính r của đường tròn , in ra kết quả
a. Chu vi = ?
b. Diện tích = ?

Gợi ý :

$$\text{chu vi} = 2 * \pi * r$$

$$\text{diện tích} = \pi * r * r$$

```
var r:Double=0.0
println("nhập vào bán kính r: ")
var s:String? = readLine()
if (s!=null)
{
    r=s.toDouble() //ép kiểu
    println("chu vi hình tròn là: " + 2* PI*r)
    println("S hình tròn là: " + PI*r*r)
}
```



3

Bài tập vận dụng

 Bài tập Kotlin 03: Tính chu vi, diện tích hình chữ nhật

1. *Viết chương trình nhập vào 2 số thực dương a, b từ bàn phím
a, b là chiều dài và chiều rộng của hình chữ nhật.*
2. *In ra màn hình chu vi và diện tích của hình chữ nhật đó.*

Gợi ý :

*diện tích $s = a * b$,*

*chu vi $p = (a+b) * 2$*





Kotlin Full Course Bài 9

- 1 Câu lệnh if - else
- 2 Câu lệnh if – else if - else
- 3 Bài tập Kotlin 04 - 09



```
if(stomach!=null){  
    keepCoding();  
}  
else{  
    orderPizza();  
}
```

1

Lệnh if - else

Cú pháp:

if (Điều_Kiện)

<Khối lệnh khi Điều_Kiện đúng>

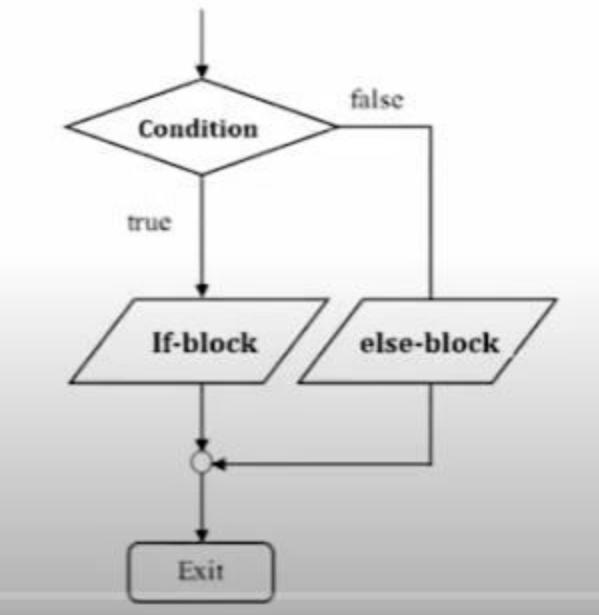
[else]

<Khối lệnh khi Điều_Kiện sai>]

Chú ý :

1. Khối else có thể không bắt buộc phải có

```
//nhập vào điểm tốt nghiệp, nếu >7.0 thì báo đã
var dtn:Float?
println("Mời thím nhập vào điểm tốt nghiệp:")
//gán dữ liệu và ép kiểu float cho chuỗi nhập vào
dtn = readLine()?.toFloat()
if (dtn!=null) //kiểm tra có dữ liệu dc truyền
{
    if (dtn >= 7.0)
        println("Bạn đã rõ")
    else
        println("Bạn đã tách ")
}
```



Chú ý :

*2. Khối lệnh nếu có từ 2 lệnh
thì phải để trong {}*



1

Lệnh if - else

❑ *if - else hoạt động như một biểu thức trả về kết quả :*

```
var a:Int =15
var b:Int =9
var max:Int
if (a>b)
    max =a
else
    max =b
println("Số lớn nhất là: $max")
```



```
var a:Int =15
var b:Int =9
var max:Int
max = if (a>b) a else b
println("Số lớn nhất là: $max")
```

Ngắn gọn



Lệnh if – else if - else

❑ Cú pháp:

```

if (Điều_Kiện_1)
<Khối lệnh 1>
else if (Điều_Kiện_2)
<Khối lệnh 2.1>
.....
else
<Khối lệnh else>
```

❑ *Chú ý :*

Có thể có nhiều Khối else if bên trong

```

/*
nhập vào điểm tb, in ra xếp loại học sinh
- Giỏi: nếu dtb <=10 và dtb>=8
- Khá: nếu 8 > dtb >= 6.5
- TB: nếu 6.5 > dtb >= 5
- Yếu: nếu dtb < 5
*/
var dtb:Float? // khai báo biến dtb
println("Mời thím nhập vào điểm trung bình: ")
dtb= readLine()?.toFloat()
if (dtb !=null)
{
    if (dtb>=8 && dtb<=10)
        println("Thím giỏi vãi nồi")
    else if ( dtb<8 && dtb >=6.5)
        println("Thím học khá")
    else if ( dtb<6.5 && dtb >=5)
        println("Thím học trung bình")
    else if (dtb<5)
        println("Thím xếp bét lớp")
    else
        println("Điểm TB tào lao")
}
```

```

// bài 3: Nhập vào 1 số, kiểm tra chẵn lẻ
var a:Int?
println("Mời cụ nhập vào 1 số nguyên: ")
// ép kiểu nguyên cho dữ liệu nhập vào
a= readLine()?.toInt()
//kiểm tra null
if (a !=null)
{
    if (a%2 ==0)
        println("số $a là số chẵn ")
    else
        println("số $a là số lẻ ")
}
```

3

Bài tập vận dụng

 Bài tập Kotlin 04: Tìm x, y khi biết tổng và hiệu của chúng

case test : Tong = 14 ,hieu = 4 => x=9, y = 5

case 2 : Tong = 8 hieu = 5 => x=6.5, y = 1.5

Gợi ý : $x + y = 14$

$x - y = 4$

Nhập vào tổng 2 số:

14

Nhập vào hiệu 2 số:

4

Giá trị x cần tìm là: 9

Giá trị y cần tìm là: 5

Nhập vào tổng 2 số:

8

Nhập vào hiệu 2 số:

5

Giá trị x cần tìm là: 6.5

Giá trị y cần tìm là: 1.5



3

Bài tập vận dụng

- Bài tập Kotlin 05: Viết chương trình nhập vào chiều cao, cân nặng, tính BMI và xuất ra thông báo

BMI<15: Thân hình quá gầy

BMI>=15 and BMI<16: Thân hình gầy

BMI>=16 and BMI<18.5: Thân hình hơi gầy

BMI>=18.5 and BMI<25:Thân hình bình thường

BMI>=25 and BMI < 30:Thân hình hơi béo

BMI >=30 and BMI<35:Thân hình béo

BMI >=35:Thân hình quá béo

Gợi ý cách tính : $BMI = canNang / (chieuCao ^ 2)$

```
// hàm mũ  
var i:Float =5.4f  
i.pow( n: 2)
```

```
Nhập vào chiều cao (m):  
1.67  
Nhập vào cân nặng (kg):  
68  
BMI của bạn =24.382374231741  
Thân hình bình thường
```



3

Bài tập vận dụng

□ **Bài tập Kotlin 06:** **Viết chương trình nhập vào 1 năm dương lịch, kiểm tra năm đó có phải năm nhuận hay không .**

□ **Gợi ý :** Năm nhuận là năm

(chia hết cho 4, và không chia hết cho 100) hoặc (chia hết cho 400)
⇒ ***((nam %4 ==0) && (nam %100 !=0)) || (nam %400 ==0)***

□ **Case test :**

Năm nhuận : 2004, 2008, 2012, 2016, 2020, 2024

Năm không nhuận : 1900, 2005



3

Bài tập vận dụng

Bài tập Kotlin07: Viết chương trình cho người dùng nhập vào 1 tháng bất kỳ từ 1 – 12 => Cho biết tháng đó có bao nhiêu ngày ?

Gợi ý :

- _ Tháng 1,3,5,7,8,10,12 có 31 ngày
- _ Tháng 4,6,9,11 có 30 ngày
- _ Nếu tháng 2 thì yêu cầu nhập thêm năm:
 - + nếu năm nhuận thì tháng 2 có 29 ngày
 - + năm không nhuận thì tháng 2 có 28 ngày



3

Bài tập vận dụng

- Bài tập Kotlin 08: Viết chương trình giải phương trình bậc 2 :**
 $ax^2 + bx + c = 0$

Phương trình bậc 2

$$\begin{aligned} & ax^2 + bx + c = 0 \\ & \Delta = b^2 - 4ac \end{aligned}$$

Bước 1: Tính $\Delta = b^2 - 4ac$

Bước 2: So sánh Δ với 0

- $\Delta < 0 \Rightarrow$ phương trình (1) vô nghiệm
- $\Delta = 0 \Rightarrow$ phương trình (1) có nghiệm kép $x_1 = x_2 = -\frac{b}{2a}$
- $\Delta > 0 \Rightarrow$ phương trình (1) có 2 nghiệm phân biệt, ta dùng công thức nghiệm sau:

$$x_1 = \frac{-b + \sqrt{\Delta}}{2a} \text{ và } x_2 = \frac{-b - \sqrt{\Delta}}{2a}$$



3

Bài tập vận dụng

Bài tập Kotlin 09: Viết chương trình nhập vào tháng trong năm, cho biết tháng đó thuộc quý mấy

Gợi ý :

1 năm có 4 quý, mỗi quý 3 tháng :

- + Quý 1 : tháng 1,2,3
- + Quý 2 : tháng 4,5,6
- + Quý 3 : tháng 7,8,9
- + Quý 4 : tháng 10,11,12





Kotlin Full Course Bài 9.1

Giải bài tập kotlin 04

❑ Bài tập Kotlin 04: Tìm x, y khi biết tổng và hiệu của chúng

case test : Tong = 14 ,hieu = 4 => x=9, y = 5

case 2 : Tong = 8 hieu = 5 => x=6.5, y = 1.5

Gợi ý : $x + y = 14$

$$x - y = 4$$

$$\Rightarrow 2x = \text{tong} + \text{hieu} \Rightarrow x = (\text{tong} + \text{hieu})/2$$

$$\Rightarrow y = \text{tong} - x$$



Ex

Bài tập Kotlin 04

- Bài tập Kotlin 04: Tìm x, y khi biết tổng và hiệu của chúng

case test : Tong = 14 ,hieu = 4 => x=9, y = 5

case 2 : Tong = 8 hieu = 5 => x=6.5, y = 1.5

Gợi ý : $x + y = \text{tong}$

$x - y = \text{hieu}$

$\Rightarrow 2x = \text{tong} + \text{hieu} \Rightarrow x = (\text{tong} + \text{hieu}) / 2$

$\Rightarrow y = \text{tong} - x$

```
Nhập vào tổng 2 số:
```

```
14
```

```
Nhập vào hiệu 2 số:
```

```
4
```

```
Giá trị x cần tìm là: 9
```

```
Giá trị y cần tìm là: 5
```

```
Nhập vào tổng 2 số:
```

```
8
```

```
Nhập vào hiệu 2 số:
```

```
5
```

```
Giá trị x cần tìm là: 6.5
```

```
Giá trị y cần tìm là: 1.5
```



Ex

Bài tập Kotlin 04

```
var tong:Float =0f
var hieu:Float =0f
var s:String?
println("Mời nhập vào tổng 2 số: ")
s= readLine()
if (s !=null)
    tong = s.toFloat()

println("Mời nhập vào hiệu 2 số: ")
s= readLine()
if (s !=null)
    hieu = s.toFloat()

//xuất thử tổng
println("tổng 2 số là : $tong")
println("hiệu 2 số là : $hieu")
//tính x, y
println("x= " +((tong+hieu)/2))
println("y= " +(tong -(tong+hieu)/2) )
```





Kotlin Full Course Bài 9.2



Giải bài tập kotlin 05

- **Bài tập Kotlin 05:** Viết chương trình nhập vào chiều cao, cân nặng, tính BMI và xuất ra thông báo

BMI<15: Thân hình quá gầy

BMI>=15 and BMI<16: Thân hình gầy

BMI>=16 and BMI<18.5: Thân hình hơi gầy

BMI>=18.5 and BMI<25:Thân hình bình thường

BMI>=25 and BMI < 30:Thân hình hơi béo

BMI >=30 and BMI<35:Thân hình béo

BMI >=35:Thân hình quá béo

Gợi ý cách tính : $BMI = canNang / (chieuCao ^ 2)$



Ex

Bài tập Kotlin 05

- **Bài tập Kotlin 05:** Viết chương trình nhập vào chiều cao, cân nặng, tính BMI và xuất ra thông báo

BMI<15: Thân hình quá gầy

BMI>=15 and BMI<16: Thân hình gầy

BMI>=16 and BMI<18.5: Thân hình hơi gầy

BMI>=18.5 and BMI<25: Thân hình bình thường

BMI>=25 and BMI < 30: Thân hình hơi béo

BMI >=30 and BMI<35: Thân hình béo

BMI >=35: Thân hình quá béo

Gợi ý cách tính : $BMI = \text{canNang} / (\text{chieuCao} ^ 2)$

```
// hàm mũ  
var _i:Float = 5.4f  
_i.pow( n: 2 )
```

```
Nhập vào chiều cao (m):  
1.67  
Nhập vào cân nặng (kg):  
68  
BMI của bạn = 24.382374231741  
Thân hình bình thường
```



Ex

Bài tập Kotlin 05

```

var chieuCao:Float =0f
var canNang:Float =0f
var BMI:Float =0f
var s:String?
println("Mời nhập chiều cao(m): ")
s= readLine()
if (s !=null)
    chieuCao = s.toFloat()

println("Mời nhập cân nặng (kg) : ")
s= readLine()
if (s !=null)
    canNang = s.toFloat()

println("Chiều cao bạn là: $chieuCao")
println("Cân nặng bạn là: $canNang")

```

```

println("Mời nhập chiều cao(m): ")
var chieuCao:Float? = readLine()?.toFloat()
println("Mời nhập cân nặng (kg) : ")
var canNang:Float? = readLine()?.toFloat()
var BMI:Float =1f
if(chieuCao != null && canNang != null)
{
    println("Chiều cao bạn là: $chieuCao")
    println("Cân nặng bạn là: $canNang")
}

```

```

//tính BMI
BMI = canNang/(chieuCao.pow( n: 2))
println("BMI của bạn = $BMI")
if (BMI < 15)
    println("Thân hình quá gầy")
else if(BMI>=15 && BMI <16)
    println("Thân hình gầy")
else if(BMI>=16 && BMI <18.5)
    println("Thân hình hơi gầy")
else if(BMI>=18.5 && BMI <25)
    println("Thân hình b thường")
else if(BMI>=25 && BMI <30)
    println("Thân hình hơi béo")
else
    println("Thân hình quá béo")

```





Kotlin
Full Course
Bài 9.3

Giải bài tập kotlin 06

- Bài tập Kotlin 06:** **Viết chương trình nhập vào 1 năm dương lịch, kiểm tra năm đó có phải năm nhuận hay không .**
- Gợi ý :** Năm nhuận là năm
(chia hết cho 4, và không chia hết cho 100) hoặc (chia hết cho 400)
 $\Rightarrow ((\text{nam} \% 4 == 0) \&\& (\text{nam} \% 100 != 0)) \mid\mid (\text{nam} \% 400 == 0)$
- Case test :**

Năm nhuận : 2004, 2008, 2012, 2016, 2020, 2024

Năm không nhuận : 1900, 2005



Ex

Bài tập Kotlin 06

□ **Bài tập Kotlin 06:** **Viết chương trình nhập vào 1 năm dương lịch, kiểm tra năm đó có phải năm nhuận hay không .**

□ **Gợi ý :** Năm nhuận là năm

(chia hết cho 4, và không chia hết cho 100) hoặc (chia hết cho 400)

$\Rightarrow ((\text{nam} \% 4 == 0) \&\& (\text{nam} \% 100 != 0)) \mid\mid (\text{nam} \% 400 == 0)$

□ **Case test :**

Năm nhuận : 2004, 2008, 2012, 2016, 2020, 2024

Năm không nhuận : 1900, 2005



Ex

Bài tập Kotlin 06

```
var nam:Int =1
var s:String?
println("Mời nhập vào 1 năm: ")
s = readLine()
if (s !=null)
    nam = s.toInt()
if (((nam %4 ==0) && (nam %100 !=0)) || (nam %400 ==0))
    println("năm $nam là năm nhuận")
else
    println("năm $nam không phải là năm nhuận")
```





Kotlin Full Course Bài 9.4

Giải bài tập kotlin 07

- ❑ Bài tập Kotlin07: Viết chương trình cho người dùng nhập vào 1 tháng bất kỳ từ 1 – 12 => Cho biết tháng đó có bao nhiêu ngày ?

❑ Gợi ý :

- _ Tháng 1,3,5,7,8,10,12 có 31 ngày
- _ Tháng 4,6,9,11 có 30 ngày
- _ Nếu tháng 2 thì yêu cầu nhập thêm năm:
 - + nếu năm nhuận thì tháng 2 có 29 ngày
 - + năm không nhuận thì tháng 2 có 28 ngày



Ex

Bài tập Kotlin 07

Bài tập Kotlin07: Viết chương trình cho người dùng nhập vào 1 tháng bất kỳ từ 1 – 12 => Cho biết tháng đó có bao nhiêu ngày ?

Gợi ý :

- _ Tháng 1,3,5,7,8,10,12 có 31 ngày
- _ Tháng 4,6,9,11 có 30 ngày
- _ Nếu tháng 2 thì yêu cầu nhập thêm năm:
 - + nếu năm nhuận thì tháng 2 có 29 ngày
 - + năm không nhuận thì tháng 2 có 28 ngày



Ex

Bài tập Kotlin 07

```
var thang:Int=1
var nam:Int =1
var s:String?
println("Mời nhập vào tháng: ")
s= readLine()
if (s !=null)
    thang =s.toInt()
if (thang==1 || thang ==3 || thang ==5 || thang==7 || thang ==8 || thang ==10 || thang==12)
    println("tháng $thang có 31 ngày")
else if (thang==4 || thang ==6 || thang ==9 || thang==11)
    println("tháng $thang có 30 ngày")
else if(thang ==2)
{
    println("Mời nhập thêm năm: ")
    s= readLine()
    if (s !=null)
        nam = s.toInt()
    // check dk năm nhuận
    if (((nam %4 ==0) && (nam %100 !=0)) || (nam %400 ==0))
        println("tháng $thang có 29 ngày")
    else
        println("tháng $thang có 28 ngày")
}
else
    println("Tháng nhập tào lao")
```





Kotlin Full Course Bài 9.5

Giải bài tập kotlin 08

Bài tập Kotlin 08: Viết chương trình giải phương trình bậc 2 :

$$ax^2 + bx + c = 0$$



Ex

Bài tập Kotlin 08

- **Bài tập Kotlin 08: Viết chương trình giải phương trình bậc 2 :**
 $ax^2 + bx + c = 0$

Phương trình bậc 2

$$\begin{aligned} & ax^2 + bx + c = 0 \\ & \Delta = b^2 - 4ac \end{aligned}$$

Bước 1: Tính $\Delta = b^2 - 4ac$

Bước 2: So sánh Δ với 0

- $\Delta < 0 \Rightarrow$ phương trình (1) vô nghiệm
- $\Delta = 0 \Rightarrow$ phương trình (1) có nghiệm kép $x_1 = x_2 = -\frac{b}{2a}$
- $\Delta > 0 \Rightarrow$ phương trình (1) có 2 nghiệm phân biệt, ta dùng công thức nghiệm sau:

$$x_1 = \frac{-b + \sqrt{\Delta}}{2a} \text{ và } x_2 = \frac{-b - \sqrt{\Delta}}{2a}$$



Ex

Bài tập Kotlin 08

```
// chú ý viết cho trường hợp a=0
println("Mời nhập vào hệ số a: ")
var a:Double? = readLine()?.toDouble()
println("Mời nhập vào hệ số b: ")
var b:Double? = readLine()?.toDouble()
println("Mời nhập vào hệ số c: ")
var c:Double? = readLine()?.toDouble()
if (a!=null && b !=null && c!=null)
{
    var delta = (b*b)-(4*a*c)
    if (delta<0)
        println("ptvn")
    else if (delta==0.0)
        println("pt có nghiệm kép x = " + -(b/(2*a)))
    else
    {
        println("Phương trình có 2 nghiệm phân biệt")
        println("x1 = "+ (-b+ sqrt(delta))/(2*a))
        println("x2 = "+ (-b- sqrt(delta))/(2*a))
    }
}
```

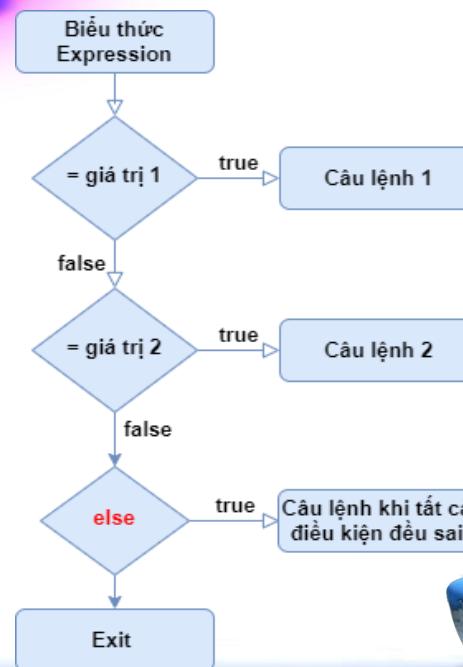




Kotlin Full Course Bài 10

when kotlin

WHEN?

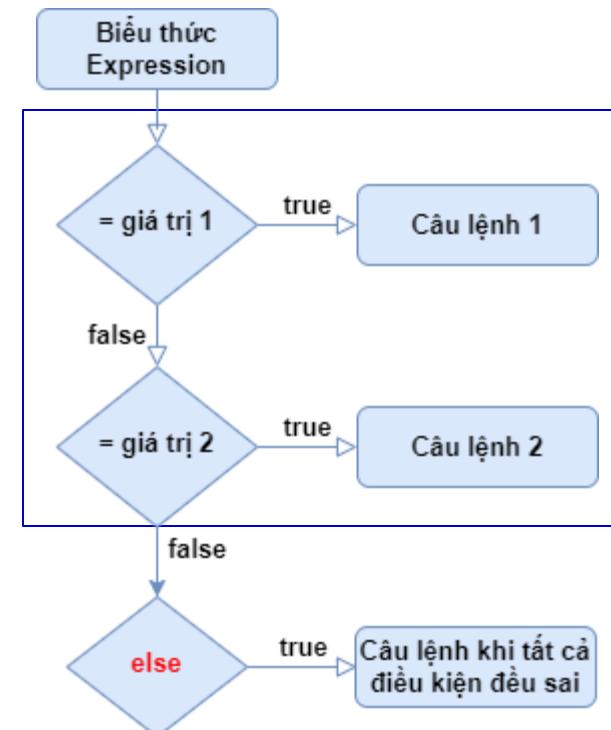
A thick, stylized yellow arrow pointing to the right, positioned below the word "WHEN?".

1

when cơ bản

❑ Cú pháp:

```
when ( expression )
{
    <giá trị 1> -> <câu lệnh 1>
    <giá trị 2> -> <câu lệnh 2>
    .....
    else -> <câu lệnh nếu 0 thỏa các điều kiện trên >
}
```



❑ Ex 1 :

```
var i:Int=1
when(i)
{
    1-> println("Thím bấm số 1")
    2-> println("Thím bấm số 2")
    3-> println("Thím bấm số 3")
    else -> println("Thím bấm linh tinh")
}
```



1

when cơ bản

- ❑ Ex 2 : nhập vào 1 số kiểm tra chẵn lẻ dùng when

```
var i:Int=0
println("Mời thím nhập vào 1 số nguyên")
var s:String? = readLine()
if (s!=null)
    i=s.toInt()
when (i%2)
{
    0-> println("$i là số chẵn ")
    else -> println("$i là số lẻ ")
}
```



```
Mời thím nhập vào 1 số nguyên
3
3 là số lẻ
```

```
Mời thím nhập vào 1 số nguyên
6
6 là số chẵn
```



2

when 1 nhóm điều kiện

- ❑ Ex 3 : nhập vào 1 tháng, kiểm tra tháng đó thuộc quý mấy

Gợi ý : Tháng 1,2,3 -> Thuộc quý 1

Tháng 4,5,6 -> Thuộc quý 2

Tháng 7,8,9 -> Thuộc quý 3

Tháng 10,11,12 -> Thuộc quý 4

```
var thang:Int=0
println("Mời thím nhập vào 1 tháng")
var s:String? = readLine()
if (s!=null)
    thang=s.toInt()
when(thang)
{
    1,2,3 -> println("Tháng $thang thuộc quý 1")
    4,5,6 -> println("Tháng $thang thuộc quý 2")
    7,8,9 -> println("Tháng $thang thuộc quý 3")
    10,11,12 -> println("Tháng $thang thuộc quý 4")
    else -> println("Tháng tào lao")
}
```



Mời thím nhập vào 1 tháng

12

Tháng 12 thuộc quý 4

Mời thím nhập vào 1 tháng

15

Tháng tào lao



3

when 1 vùng dữ liệu

 when cho phép kiểm tra theo vùng dữ liệu

```
var thuNhap:Int=0
println("Mời thím nhập vào thu nhập (triệu)/ 1 tháng")
var s:String? = readLine()
if (s!=null)
    thuNhap=s.toInt()
when(thuNhap)
{
    in 1 .. ≤ 3 -> println("nghèo kiếp xác")
    in 4 .. ≤ 7 -> println("nghèo vừa vừa")
    in 8 .. ≤ 19 -> println("thu nhập khá")
    in 20 .. ≤ 50 -> println("thu nhập cao")
    in 51 .. ≤ 1000 -> println("đại gia")
    else -> println("Bốc phét ít thôi")
}
```



```
Mời thím nhập vào thu nhập (triệu)/ 1 tháng
3
nghèo kiếp xác
```

```
Mời thím nhập vào thu nhập (triệu)/ 1 tháng
15
thu nhập khá
```

```
Mời thím nhập vào thu nhập (triệu)/ 1 tháng
2000
Bốc phét ít thôi
```



4

Biểu thức trả về when + when (if –else)

- dùng when như 1 biểu thức trả về kết quả*

```
var a:Int =101
var check = when (a)
{
    in 0 .. ≤ 100 -> false
    else -> true
}
println(check)
```

true

```
var a:Int =99
var check = when (a)
{
    in 0 .. ≤ 100 -> false
    else -> true
}
println(check)
```

false

- dùng when như 1 biểu thức if –else*

```
var a:Int =99
when // không có expression
{
    a%2 ==0 -> println("$a là số chẵn")
    a%2 !=0 -> println("$a là số lẻ")
}
```



99 là số lẻ





Kotlin

Full Course

Bài 11

1

for step range

2

for close range

3

for half – open range

4

for downTo

5

for in collection

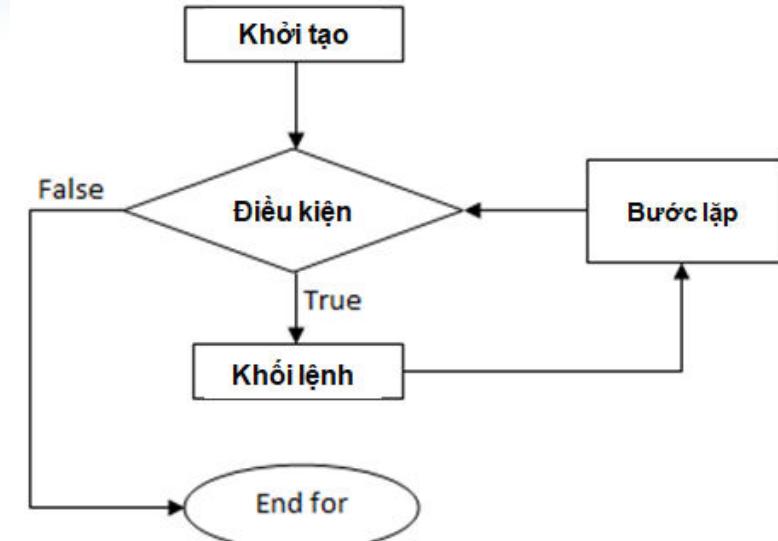


1

for step range

□ Cú pháp:

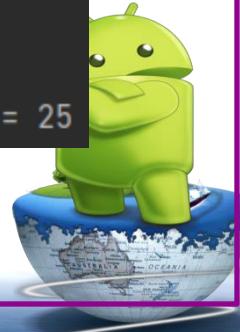
```
for ( i in a .. b step c )
{
    <lệnh xử lý với biến i>
}
```



```
//tính tổng các số lẻ
var tong:Int =0
for (i in 1 .. 9 step 2)
{
    println(i)
    tong+=i //tong = tong +i
}
println("Tổng các số lẻ từ 1 đến 9 = " +tong)
```



1
3
5
7
9
Tổng các số lẻ từ 1 đến 9 = 25



2

for close range

❑ Cú pháp:

```
for ( i in a .. b)
{
    <lệnh xử lý với biến i>
}
```

```
for (i in 1 .. 9)
{
    println(i)
}
```



1
2
3
4
5
6
7
8
9

```
//xuất bảng cửu chương 2
for (i in 1 .. 9)
{
    println("$i *2 = " + i*2)
}
```



1 *2 = 2
2 *2 = 4
3 *2 = 6
4 *2 = 8
5 *2 = 10
6 *2 = 12
7 *2 = 14
8 *2 = 16
9 *2 = 18

```
//ví dụ tính giải thừa
var gt:Int =1
var n:Int =4
for (i in 1 .. n)
{
    gt*=i
}
println("$n! = " +gt)
```



4! = 24





3

for half – open range

□ Cú pháp:

```
for ( i in a until b )
{
    <lệnh xử lý với biến i>
}
```

```
for (i in 1 ..< 5)
{
    println(i)
}
```

1
2
3
4



```
var tong:Int =0
for (i in 1 ..< 5)
{
    println(i)
    tong+=i
}
println("tổng từ 1- sát 5 là: " +tong)
```



1
2
3
4
tổng từ 1- sát 5 là: 10



4

for downTo

□ Cú pháp:

```
for ( i in a downTo b )
{
    <lệnh xử lý với biến i>
}
```

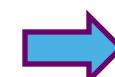
```
for (i in 5 ≥ downTo ≥ 1)
{
    println(i)
}
```



```
5
4
3
2
1
```

```
for ( i in a downTo b step c )
{
    <lệnh xử lý với biến i>
}
```

```
for (i in 5 ≥ downTo ≥ 2 step 2)
{
    println(i)
}
```



```
5
3
```



5

for in collection

□ Cú pháp:

```
for ( item in collection)
{
    <lệnh xử lý với biến item>
}
```

```
var dsTen = arrayOf("anh1.jpg", "anh2.jpg", "anh3.jpg")
for (item in dsTen)
{
    println(item)
}
```



```
anh1.jpg
anh2.jpg
anh3.jpg
```

Sang đến phần mảng sẽ học
phần này kỹ hơn

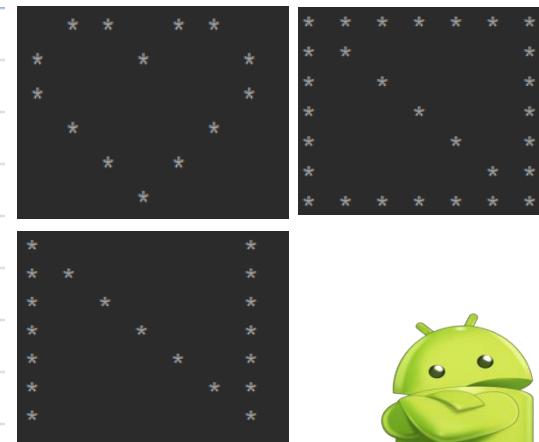




Kotlin Full Course Bài 12

- 1 for lồng nhau
- 2 Vận dụng vẽ chữ N, ❤

		j						
		0	1	2	3	4	5	6
i	0	00	01	02	03	04	05	06
	1	10	11	12	13	14	15	16
	2	20	21	22	23	24	25	26
	3	30	31	32	33	34	35	36
	4	40	41	42	43	44	45	46
	5	50	51	52	53	54	55	56
	6	60	61	62	63	64	65	66



1

for lồng nhau

□ Ví dụ 1: Xuất bảng cửu chương :

```
for (i in 1 .. 9)
{
    for (a in 2 .. 9)
    {
        //println("i = $i")
        //println("a = $a")
        println("$i*$a= ${i*a}")
    }
    println("-----")
}
```



8*2=	16
8*3=	24
8*4=	32
8*5=	40
8*6=	48
8*7=	56
8*8=	64
8*9=	72

9*2=	18
9*3=	27
9*4=	36
9*5=	45
9*6=	54
9*7=	63
9*8=	72
9*9=	81

□ Ví dụ 2: Xuất all tọa độ :

```
var h=6
for (i in 1 .. h)
{
    for (j in 1 .. h)
    {
        print("$i$j \t")
    }
    println()
}
```



11	12	13	14	15	16
21	22	23	24	25	26
31	32	33	34	35	36
41	42	43	44	45	46
51	52	53	54	55	56
61	62	63	64	65	66

	j						
	1	2	3	4	5	6	7
i	1	11	12	13	14	15	16
	2	21	22	23	24	25	26
	3	31	32	33	34	35	36
	4	41	42	43	44	45	46
	5	51	52	53	54	55	56
	6	61	62	63	64	65	66
	7	71	72	73	74	75	76

2

Vận dụng vẽ chữ N, ❤

	j						
	1	2	3	4	5	6	7
i	11	12	13	14	15	16	17
1	21	22	23	24	25	26	27
2	31	32	33	34	35	36	37
3	41	42	43	44	45	46	47
4	51	52	53	54	55	56	57
5	61	62	63	64	65	66	67
6	71	72	73	74	75	76	77
7							

```
var h=7
for (i in 1 .. ≤ h)
{
    for (j in 0 .. ≤ h)
    {
        if(j==1 || j==h || i==j)
            print("$i$j")
            print("\t")
    }
    println()
}
```

11	17
21	22
31	33
41	44
51	55
61	66
71	77

	j						
	1	2	3	4	5	6	7
i	11	12	13	14	15	16	17
1	21	22	23	24	25	26	27
2	31	32	33	34	35	36	37
3	41	42	43	44	45	46	47
4	51	52	53	54	55	56	57
5	61	62	63	64	65	66	67
6	71	72	73	74	75	76	77
7							

```
var h=7
for (i in 1 .. ≤ h)
{
    for (j in 1 .. ≤ h)
    {
        if ((i==1 && (j==2 || j==3 || j==4)) ||
            (i==2 && (j==1 || j==3 || j==5)) ||
            (i==3 && (j==1 || j==2 || j==4 || j==6)) ||
            (i==4 && (j==1 || j==2 || j==3 || j==5 || j==7)))
            print("$i$j")
            print("\t")
    }
    println()
}
```

12	13	15	16
21		24	27
31			37
	42		46
	53	55	
		64	





Kotlin Full Course

Bài 13

- 1 while kotlin
- 2 do while kotlin
- 3 while (true)



1

while kotlin

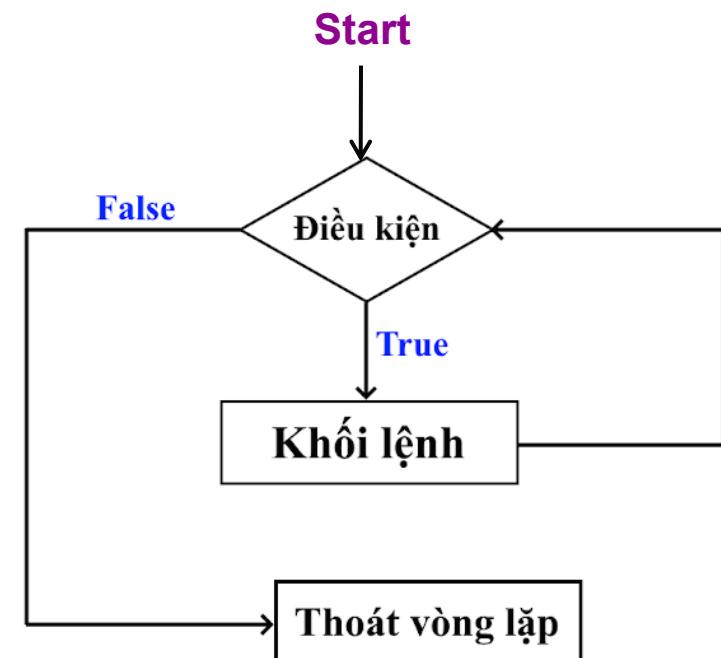
□ Cú pháp :

```
while (Điều_Kiện_Lặp)
{
    < Khối lệnh lặp>
}
```

```
var i= 0;
while (i<=5)
{
    println("i= $i")
    i++ // i = i+1
}
println("thoát while i= $i")
```



```
i= 0
i= 1
i= 2
i= 3
i= 4
i= 5
```



1

while kotlin

Ví dụ thực hành: *Viết chương trình nhập vào số n từ bàn phím.
n phải nằm trong khoảng từ 1 đến 99
Nhập sai bắt nhập lại*

```
var n=0;
while (n<1 || n >99)
{
    println("Mời nhập vào 1 số nguyên từ 1-99: ")
    var s:String? = readLine()
    if (s !=null)
        n=s.toInt()
}
println("Bạn nhập hợp lệ, n= $n")
```



```
Mời nhập vào 1 số nguyên từ 1-99:  
-5  
Mời nhập vào 1 số nguyên từ 1-99:  
100  
Mời nhập vào 1 số nguyên từ 1-99:  
70  
Bạn nhập hợp lệ, n= 70
```



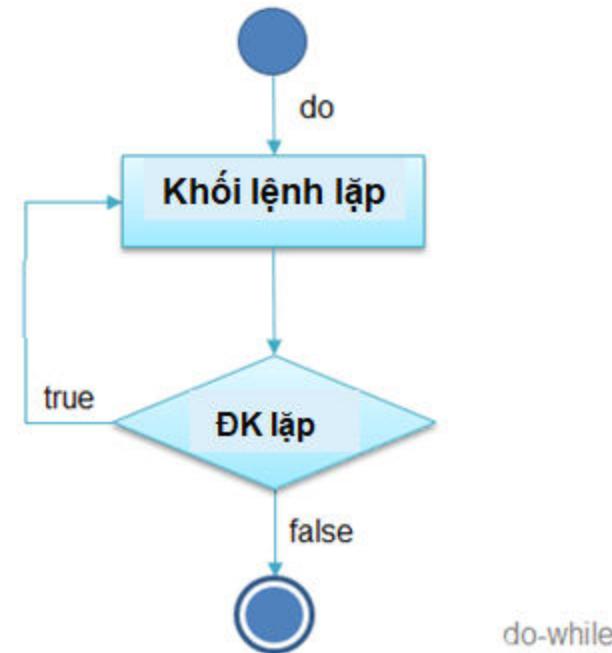
2

do while kotlin

□ Cú pháp:

```
do
{
    <khối lệnh lặp lại>
}
while (<Điều kiện lặp>);
```

```
//tính tổng các số từ 1 đến 5
var i = 1
var tong=0
do
{
    tong+=i // tong = tong+i
    i++
}while (i<=5)
println("tổng từ 1-5 là: " +tong)
```



tổng từ 1-5 là: 15

□ **Chú ý:** Vòng lặp này khác while ở chỗ, nó thi hành khối lệnh trước, sau đó mới kiểm tra điều kiện, kiểm tra thấy điều kiện vẫn true thì lặp tiếp.



3

while (true)

Vòng lặp này chạy vĩnh cửu, đến khi gặp điều kiện thỏa mãn
=> dùng lệnh **break** thoát lặp

```
while (true)
{
    println("Bạn có muốn thoát chương trình không? (y để thoát)")
    var s:String? = readLine()
    if (s=="y" || s=="Y")
        break
}
println("Đã đăng xuất khỏi trái đất")
```



```
Bạn có muốn thoát chương trình không? (y để thoát)
n
Bạn có muốn thoát chương trình không? (y để thoát)
y
Đã đăng xuất khỏi trái đất
```

```
var n=0
while(true)
{
    n++
    println("n=" +n)
    if (n==5)
        break // nếu n=5 thì thoát vòng lặp
}
```



3

while (true)

```
var n=0
while (true)
{
    n++
    println("n= $n")
    if (n==100)
        break
}
println("Đã thoát c trinh, n= $n")
```





Kotlin Full Course Bài 14

1

continue , break

2

Bài tập Kotlin 10 - 16



1

continue , break

□ Cách dùng :

- **break** thường được dùng để thoát khỏi 1 vòng lặp
- **continue** dùng để bỏ qua 1 giá trị trong vòng lặp

```
//tính tổng từ 1 - 5 (nhưng bỏ qua số 3)
var tong:Int =0
for (i in 1 .. 5)
{
    if (i==3)
        continue // nếu i=3 thì bỏ qua
    else
    {
        println("i= " +i)
        tong +=i
    }
}
println("tổng = $tong")
```

```
i= 1
i= 2
i= 4
i= 5
tổng = 12
```

```
var n:Int =0
while (n<100)
{
    n++ // tăng n lên 1
    println("n trong vòng lặp = $n")
    if (n==4)
        break
}
println("n= $n")
```

```
n trong vòng lặp = 1
n trong vòng lặp = 2
n trong vòng lặp = 3
n trong vòng lặp = 4
n= 4
```



2

Bài tập Kotlin 10 - 16

Bài tập Kotlin 10:

Viết chương trình nhập vào số nguyên n, in ra kết quả n!

- ✓ Dùng vòng lặp for
- ✓ Dùng vòng lặp while

```
mời nhập vào số nguyên n:  
4  
kết quả 4!= 24
```

Bài tập Kotlin 11:

Viết chương trình nhập nhập số a từ bàn phím,

- ✓ Nếu a chẵn thì tính tổng các số chẵn từ 0 đến a
- ✓ Nếu a lẻ thì in ra dòng chữ “tôi o tính tổng số lẻ, bye bye ” và thoát chương trình



2

Bài tập Kotlin 10 - 16

Bài tập Kotlin 12:

Viết chương trình tính tổng các số lẻ từ 1 đến n, n nhập từ bàn phím

- ✓ Nhập $n = 7$, Bỏ qua không cộng tổng với số 3 => in ra kết quả
(gợi ý đáp án : $1+5+7 =13$)
- ✓ Thử break khi vòng lặp chạy đến giá trị $n=3$

Bài tập Kotlin 13:

Viết chương trình :

- ✓ Tìm những số chia hết cho 3 từ 10 đến 50

Bài tập Kotlin 14:

Viết chương trình :

- ✓ Tính tổng $S = 1! + 2! + 3! + \dots + 10!$



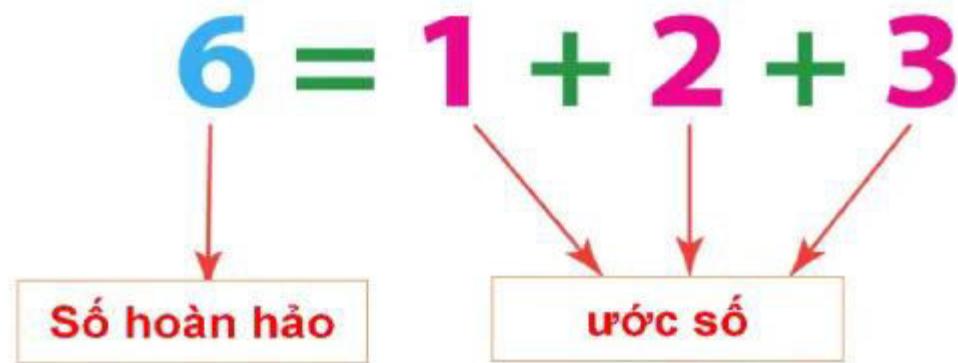
2

Bài tập Kotlin 10 - 16

 Bài tập Kotlin 15:

Số hoàn thiện (hay còn gọi là số hoàn chỉnh, số hoàn hảo hoặc số hoàn thành) là một số nguyên dương mà tổng các ước nguyên dương chính thức của nó (số nguyên dương bị nó chia hết ngoại trừ nó) bằng chính nó.

✓ Tìm tất cả những số hoàn thiện trong phạm vi từ 1-1000



2

Bài tập Kotlin 10 - 16

 Bài tập Kotlin 16:

- Viết chương trình nhập vào số nguyên $a > 0$ từ bàn phím.
- ✓ Cho biết đó có phải số ng tố
 - (số ng tố là số > 1 , và chỉ chia hết cho 1 và chính nó)
- ✓ Kết thúc chương trình hỏi người dùng: Bạn có muốn tiếp tục sử dụng phần mềm không? Nếu chọn không thì thoát c trinh





Kotlin Full Course Bài 14.1

Giải bài tập Kotlin 10

❑ Bài tập Kotlin 10:

Viết chương trình nhập vào số nguyên n, in ra kết quả n!

- ✓ Dùng vòng lặp for
- ✓ Dùng vòng lặp while

```
mời nhập vào số nguyên n:  
4  
kết quả 4!= 24
```



Ex

Bài tập Kotlin 10

□ Viết chương trình nhập vào số nguyên n, in ra kết quả n!

- ✓ Dùng vòng lặp for
- ✓ Dùng vòng lặp while

```
//c1 dùng for
var gt =1
println("Mời nhập vào n: ")
var n:Int? = readLine()?.toInt()
if (n!=null)
{
    for (i in 1 .. n)
    {
        gt*=i
    }
}
println("$n! = $gt")
```

```
mời nhập vào số nguyên n:
4
kết quả 4!= 24
```

```
// cách 2 dùng while
var gt =1
var i=1
println("Mời nhập vào n: ")
var n:Int? = readLine()?.toInt()
if (n!=null)
{
    while (i<= n)
    {
        gt*=i
        i++
    }
}
println("$n! = $gt")
```





Kotlin Full Course Bài 14.2

Giải bài tập Kotlin 11

Bài tập Kotlin 11:

Viết chương trình nhập nhập số a từ bàn phím,

- ✓ Nếu a chẵn thì tính tổng các số chẵn từ 0 đến a
- ✓ Nếu a lẻ thì in ra dòng chữ “tôi o tính tổng số lẻ, bye bye ” và thoát chương trình



Ex

Bài tập Kotlin 11

□ Viết chương trình nhập nhập số a từ bàn phím,

- ✓ Nếu a chẵn thì tính tổng các số chẵn từ 0 đến a
- ✓ Nếu a lẻ thì in ra dòng chữ “Tôi o tính tổng số lẻ, bye bye ” và thoát chương trình

```
var tong=0
println("Mời nhập vào số a: ")
var a:Int? = readLine()?.toInt()
if (a!=null)
{
    if (a%2==0)
    {
        //tính tổng các số chẵn từ 0 - a
        for (i in 0 .. a step 2)
        {
            println(i)
            tong+=i
        }
        println("Tổng các số chẵn từ 0 -> $a = $tong")
    }
    else
        println("Tôi không tính tổng vì $a là số lẻ, byebyy ")
}
```

Mời nhập vào số a:

7

Tôi không tính tổng vì 7 là số lẻ, bybyy

Mời nhập vào số a:

6

0

2

4

6

Tổng các số chẵn từ 0 -> 6 = 12





Kotlin

Full Course

Bài 14.3

Giải bài tập Kotlin 12

Bài tập Kotlin 12:

Viết chương trình tính tổng các số lẻ từ 1 đến n, n nhập từ bàn phím

- ✓ Nhập $n = 7$, Bỏ qua không cộng tổng với số 3 => in ra kết quả
(gợi ý đáp án : $1+5+7 = 13$)
- ✓ Thủ break khi vòng lặp chạy đến giá trị $n=3$



Ex

Bài tập Kotlin 12

- Viết chương trình tính tổng các số lẻ từ 1 đến n, n nhập từ bàn phím
- ✓ Nhập n = 7, Bỏ qua không cộng tổng với số 3 => in ra kết quả
(gợi ý đáp án : 1+5+7 =13)
 - ✓ Thử break khi vòng lặp chạy đến giá trị n=3

```
var tong=0
println("Mời nhập vào số nguyên n: ")
var n:Int? = readLine()?.toInt()
if (n !=null)
{
    for (i in 1 .. n)
        if (i%2!=0)
        {
            tong+=i
        }
    println("Tổng các số lẻ từ 1 -> $n = $tong")
}
```

Mời nhập vào số nguyên n:

7

Tổng các số lẻ từ 1 -> 7 = 16

```
var tong=0
println("Mời nhập vào số nguyên n: ")
var n:Int? = readLine()?.toInt()
if (n !=null)
{
    for (i in 1 .. n)
        if (i%2!=0)
        {
            if (i==3)
                continue
            else
                tong+=i
        }
    println("Tổng các số lẻ từ 1 -> $n = $tong")
}
```

Mời nhập vào số nguyên n:

7

Tổng các số lẻ từ 1 -> 7 = 13



Ex

Bài tập Kotlin 12

- Viết chương trình tính tổng các số lẻ từ 1 đến n, n nhập từ bàn phím
- ✓ Nhập n = 7, Bỏ qua không cộng tổng với số 3 => in ra kết quả
(gợi ý đáp án : 1+5+7 =13)
 - ✓ Thử break khi vòng lặp chạy đến giá trị n=3

```
var tong=0
println("Mời nhập vào số nguyên n: ")
var n:Int? = readLine()?.toInt()
if (n !=null)
{
    for (i in 1 .. n)
        if (i%2!=0)
        {
            if (i==3)
                break
            else
                tong+=i
        }
    println("Tổng các số lẻ từ 1 -> $n = $tong")
}
```

```
Mời nhập vào số nguyên n:
7
Tổng các số lẻ từ 1 -> 7 = 1
```





Kotlin Full Course Bài 14.4

Giải bài tập Kotlin 13,14

Bài tập Kotlin 13:

Viết chương trình :

- ✓ Tìm những số chia hết cho 3 từ 10 đến 50

Bài tập Kotlin 14:

Viết chương trình :

- ✓ Tính tổng $S = 1! + 2! + 3! + \dots + 10!$



Ex

Bài tập Kotlin 13

- Viết chương trình :**
- ✓ Tìm những số chia hết cho 3 từ 10 đến 50

```
fun main(args: Array<String>) {  
    for (i in 10 .. 50)  
        if (i%3==0)  
            print("$i ")  
}
```

```
12 15 18 21 24 27 30 33 36 39 42 45 48
```



Ex

Bài tập Kotlin 14

- **Viết chương trình :**
✓ **Tính tổng $S = 1! + 2! + 3! + \dots + 10!$**

```
var tong=0
var gt=1
for (i in 1 .. 10)
{
    gt*=i // gt = gt*i
    //1. i=1, gt = 1
    //2. i=2, gt= 1 => gt = 2
    //3. i=3, gt= 2 => gt = 6
    //4. i=4, gt= 6 => gt = 24
    tong+=gt
}
println("S= $tong")
```

S= 4037913

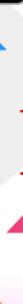




Kotlin

Full Course

Bài 14.5



Giải bài tập Kotlin 15

Số hoàn thiện (hay còn gọi là số hoàn chỉnh, số hoàn hảo hoặc số hoàn thành) là một số nguyên dương mà tổng các ước nguyên dương chính thức của nó (số nguyên dương bị nó chia hết ngoại trừ nó) bằng chính nó.

✓ Tìm tất cả những số hoàn thiện trong phạm vi từ 1-1000

$$6 = 1 + 2 + 3$$

Số hoàn hảo

Ước số





Ex

Bài tập Kotlin 15

- ❑ Số hoàn thiện (hay còn gọi là số hoàn chỉnh, số hoàn hảo hoặc số hoàn thành) là một số nguyên dương mà tổng các ước nguyên dương chính thức của nó (số nguyên dương bị nó chia hết ngoại trừ nó) bằng chính nó.
- ✓ **Tìm tất cả những số hoàn thiện trong phạm vi từ 1-1000**

```
//kiểm tra số hoàn hảo
println("Mời nhập vào 1 số nguyên n: ")
var n:Int? = readLine()?.toInt()
if (n!=null)
{
    var tong = 0
    for (i in 1 .. until < n) // ước o bao gồm chính nó
    {
        if (n%i ==0)
        {
            println(i)
            tong+=i
        }
    }
    if (n==tong)
        println("$n là số hoàn hảo")
    else
        println("$n không phải hoàn hảo")
}
```

```
//kiểm tra số hoàn hảo
for (n in 1 .. 1000)
{
    var tong = 0
    for (i in 1 .. until < n) // ước o bao gồm chính nó
    {
        if (n%i ==0)
        {
            //println(i)
            tong+=i
        }
    }
    if (n==tong)
        print("$n ")
}
```





Kotlin Full Course Bài 14.6

Giải bài tập Kotlin 16

Bài tập Kotlin 16:

- Viết chương trình nhập vào số nguyên $a > 0$ từ bàn phím.
- ✓ Cho biết đó có phải số ng tố
 - (số ng tố là số > 1 , và chỉ chia hết cho 1 và chính nó)
 - ✓ Kết thúc chương trình hỏi người dùng: Bạn có muốn tiếp tục sử dụng phần mềm không? Nếu chọn không thì thoát cctrinh



Ex

Bài tập Kotlin 16

```

while (true)
{
    println("Mời nhập vào số nguyên n: ")
    var n:Int? = readLine()?.toInt()
    if (n !=null)
    {
        // cho for chạy từ 1 đến n
        // (demUoc=2 => là số n tố)
        var demUoc = 0
        for (i in 1 .. n)
            if (n%i==0)
                demUoc++
        if (demUoc==2)
            println("$n là số nguyên tố")
        else
            println("$n không phải số nguyên tố")
    }
    println("Bạn có tiếp tục, bấm n để thoát: ")
    var s:String? = readLine()
    if (s!=null)
    {
        if (s=="n" || s=="N")
            break
    }
}

```

Mời nhập vào số nguyên n:
 7
 7 là số nguyên tố
 Bạn có tiếp tục, bấm n để thoát:
 0
 Mời nhập vào số nguyên n:
 4
 4 không phải số nguyên tố
 Bạn có tiếp tục, bấm n để thoát:
 n





Kotlin Full Course Bài 15



- 1 Hàm trong kotlin
- 2 Ví dụ vận dụng
- 3 Bài tập Kotlin 17

Functions in Kotlin



1

Hàm trong Kotlin

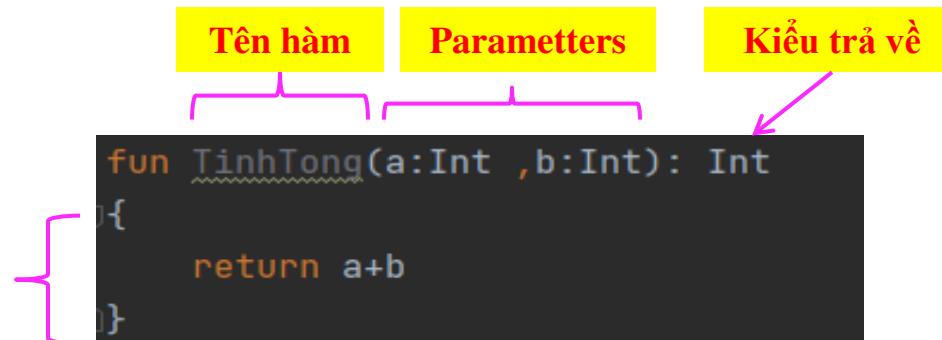
- 1. Khái quát về hàm :
 - Khi muốn thực thi một đoạn code nào nó nhiều lần, thay vì phải copy đi copy lại đoạn code đó, dẫn đến chương trình bị trùng lặp code rất
=> Khi đó ta sử dụng hàm
 - Hàm là 1 khối lệnh thực hiện 1 công việc hoàn chỉnh (module)
Hàm còn được gọi là chương trình con, phương thức, hành vi
- Hàm có 2 loại:
 - Trả về giá trị (**có return**)
 - **Không trả về giá trị**: gọi là hàm thủ tục (**procedure**)
- Công dụng :
 1. Chia nhỏ phân việc của dự án
 2. Tái sử dụng: khi cần chỉ cần gọi lại chương trình con mà o cần phải viết lại



1

Hàm trong Kotlin

□ 2. Khai báo hàm (có return):



```

fun TinhTong(a:Int ,b:Int): Int
{
    return (a+b)
}

fun main(args: Array<String>) {
    var tong:Int = TinhTong( a: 7, b: 5)
    println(tong)
}
  
```

□ 3. Hàm thủ tục, (không có return)

```

fun XinChao(s:String)
{
    if(s== "nam")
        println("Xin chào, mình là boy")
    else if ( s=="nu")
        println("Xin chào, mình là girl")
}
  
```

```

Xinchao( s: "nu")
Xinchao( s: "nam")
  
```

```

Xin chào, mình là girl
Xin chào, mình là boy
  
```

□ 4. Đặt tên hàm:

- *Giống quy tắc đặt tên biến*
- *Nên ghi tương minh (VD: BinhPhuong() , TinhGiaiThua())*



2

Ví dụ vận dụng

 Viết chương trình giải ptb2 dùng hàm :

1. Tìm nghiệm của ptb2 với : $a = 1, b = 2, c = 3$
2. Tìm nghiệm của ptb2 với : $a = 1, b = 2, c = 1$
3. Tìm nghiệm của ptb2 với : $a = 1, b = 2, c = -3$

Giải phương trình bậc 2**Bước 1:** Tính $\Delta = b^2 - 4ac$ **Bước 2:** So sánh Δ với 0

- $\Delta < 0 \Rightarrow$ phương trình (1) vô nghiệm
- $\Delta = 0 \Rightarrow$ phương trình (1) có nghiệm kép $x_1 = x_2 = -\frac{b}{2a}$
- $\Delta > 0 \Rightarrow$ phương trình (1) có 2 nghiệm phân biệt, ta dùng công thức nghiệm sau:

$$x_1 = \frac{-b + \sqrt{\Delta}}{2a} \text{ và } x_2 = \frac{-b - \sqrt{\Delta}}{2a}$$

Mẹo nhẩm nghiệm phương trình bậc 2 nhanh:

- Nếu $a+b+c=0$ thì $x_1 = 1, x_2 = c/a$
- Nếu $a-b+c=0$ thì $x_1 = -1, x_2 = -c/a$



2

Ví dụ vận dụng

```

fun giaoPTB2(a:Double, b:Double, c:Double): String
{
    if (a==0.0) // bx+c= 0
    {
        if (b==0.0 && c==0.0)
            return "Pt vô số nghiệm"
        else if (b==0.0 && c!=0.0)
            return "pt vô nghiệm"
        else
            return "Pt có nghiệm x = ${-c/b}"
    }
    else // giải pt bậc 2
    {
        var delta = (b*b)-(4*a*c)
        if(delta<0)
            return "pt vô nghiệm"
        else if (delta ==0.0)
        {
            var x = -b/(2*a)
            return "pt có nghiệm kép x1 = x2 = $x"
        }
        else
        {
            var x1 = (-b+ sqrt(delta))/(2*a)
            var x2 = (-b- sqrt(delta))/(2*a)
            return "pt có 2 nghiệm phân biệt, x1= $x1 , x2 = $x2"
        }
    }
}

```

```

var kq1 = giaoPTB2( a: 1.0, b: 2.0, c: 3.0)
var kq2 = giaoPTB2( a: 1.0, b: 2.0, c: 1.0)
var kq3 = giaoPTB2( a: 1.0, b: 2.0, c: -3.0)
println(kq1)
println(kq2)
println(kq3)

```

```

fun main(args: Array<String>) {
    var a:Double?
    var b:Double?
    var c:Double?
    println("Nhập hệ số a:")
    a= readLine()?.toDouble()
    println("Nhập hệ số b:")
    b= readLine()?.toDouble()
    println("Nhập hệ số c:")
    c= readLine()?.toDouble()
    if(a==null || b==null || c==null) return
    var kq=giaoPTB2(a,b,c)
    println(kq)
}

```

3

Bài tập Kotlin 17

□ Viết chương trình nhập vào từ bàn phím 2 số a, b

Nhập 1 ký tự +, -, *, /

(Làm theo 2 cách, C1: Dùng function, C2: Dùng when)

1. Nếu nhập + : Tính $a+b$ và in ra kết quả

2. Nếu nhập - : Tính $a-b$ và in ra kết quả

3. Nếu nhập * : Tính $a*b$ và in ra kết quả

4. Nếu nhập / : Tính a/b và in ra kết quả (chú ý ktra mău số khác 0)

```
mời nhập vào số a  
6  
mời nhập vào số b  
5  
mời nhập vào phép tính(+,-,*,/):  
+  
kết quả = 11.0
```

```
mời nhập vào số a  
5  
mời nhập vào số b  
2  
mời nhập vào phép tính(+,-,*,/):  
*  
kết quả = 10.0
```

```
mời nhập vào số a  
9  
mời nhập vào số b  
0  
mời nhập vào phép tính(+,-,*,/):  
/  
0 chia được mà ơi|
```





Kotlin Full Course Bài 15.1



Giải bài tập Kotlin 17



- **Viết chương trình nhập vào từ bàn phím 2 số a, b
Nhập 1 ký tự +, -, *, /**

(*Làm theo 2 cách, C1: Dùng function , C2: Dùng when*)

1. Nếu nhập + : Tính $a+b$ và in ra kết quả
2. Nếu nhập - : Tính $a-b$ và in ra kết quả
3. Nếu nhập * : Tính $a*b$ và in ra kết quả
4. Nếu nhập / : Tính a/b và in ra kết quả (chú ý ktra mâu số khác 0)



Ex

Bài tập Kotlin 17

Viết chương trình nhập vào từ bàn phím 2 số a, b

Nhập 1 ký tự +, -, *, /

(Làm theo 2 cách, C1: Dùng function, C2: Dùng when)

1. Nếu nhập + : Tính $a+b$ và in ra kết quả

2. Nếu nhập - : Tính $a-b$ và in ra kết quả

3. Nếu nhập * : Tính $a*b$ và in ra kết quả

4. Nếu nhập / : Tính a/b và in ra kết quả (chú ý ktra mău số khác 0)

```
mời nhập vào số a
6
mời nhập vào số b
5
mời nhập vào phép tính(+,-,*,/):
+
kết quả = 11.0
```

```
mời nhập vào số a
5
mời nhập vào số b
2
mời nhập vào phép tính(+,-,*,/):
*
kết quả = 10.0
```

```
mời nhập vào số a
9
mời nhập vào số b
0
mời nhập vào phép tính(+,-,*,/):
/
0 chia được mà ơi|
```



Ex

Bài tập Kotlin 17 Cách 1

```
fun Tong(a:Double,b:Double){  
    println("$a+$b = ${a+b}")  
}  
  
fun Hieu(a:Double,b:Double){  
    println("$a-$b = ${a-b}")  
}  
  
fun Tich(a:Double,b:Double){  
    println("$a*$b = ${a*b}")  
}  
  
fun Thuong(a:Double,b:Double){  
    if (b==0.0)  
        println("Không chia được cho 0 mà ơi ")  
    else  
        println("$a/$b = ${a/b}")  
}
```

```
fun main(args: Array<String>){  
    println("Mời nhập vào số a: ")  
    var a:Double? = readLine()?.toDouble()  
    println("Mời nhập vào số b: ")  
    var b:Double? = readLine()?.toDouble()  
    println("Mời nhập vào phép tính (+ - * /): ")  
    var c:String? = readLine()  
    if (a!= null && b!=null && c!=null)  
    {  
        if (c=="+")  
            Tong(a,b)  
        else if (c=="-")  
            Hieu(a,b)  
        else if (c=="*")  
            Tich(a,b)  
        else if (c=="/")  
            Thuong(a,b)  
        else  
            println("Em chưa học đến phép tính này")  
    }  
}
```



Ex

Bài tập Kotlin 17 Cách 2

when (expression)

```
{
    <giá trị 1> -> <câu lệnh 1>
    <giá trị 2> -> <câu lệnh 2>
    .....
    else -> <câu lệnh nếu 0 thỏa các điều kiện trên >
}
```

```

var a:Double?
var b:Double?
var c:String?
var dapAn:String

println("mời nhập vào số a")
a = readLine()?.toDouble()
println("mời nhập vào số b")
b = readLine()?.toDouble()
println("mời nhập vào phép tính(+,-,*,/): ")
c = readLine()

if (a== null || b==null || c== null) return

when (c)
{
    "+"-> println("$a+$b = ${a+b}")
    "-"> println("$a-$b = ${a-b}")
    "*"> println("$a*$b = ${a*b}")
    "/"->
        if (b==0.0)
            println("Không chia được cho 0 mà ơi")
        else
            println("$a/$b = ${a/b}")
    else->
        println("Máy tính chỉ tính được + - * / ")
}

```



Kotlin Full Course

Bài 16



- 1 try - catch - finally
- 2 throw statement



1

try - catch - finally

 1. Khái quát ngoại lệ :

- ✓ Ngoại lệ là những lỗi có thể phát sinh lúc thực thi. Chẳng hạn như đọc một file không tồn tại. Hoặc kết nối với database không có sẵn...
- ✓ Có rất nhiều lỗi mà đôi khi lập trình viên không thể dự đoán hết được
- ✓ Khi chạy chương trình ta có thể gặp lỗi, xử lý ngoại lệ giúp :

1. Xuất thông báo lỗi

2. Tiếp tục chạy chương trình mà không làm gián đoạn, treo phần mềm

```
var a=10  
var b=0  
var c=a/b  
println(c)  
println("Đoạn code phía sau")
```

```
Exception in thread "main" java.lang.ArithmetricException Create breakpoint : / by zero  
at Bai16_bietleKt.main(bai16-bietle.kt:21)
```



1

try - catch - finally

```
fun main(args: Array<String>) {
    var a=10
    var b=0
    try{
        var c=a/b
        println(c)
    }
    catch (e:Exception)
    {
        e.printStackTrace()
    }
    finally {
        println("đoạn code muốn thực thi, kể cả khi có lỗi")
    }
    println("Đoạn code phía sau")
}
```

```
fun main(args: Array<String>) {
    var a=10
    var b=5
    try{
        var c=a/b
        println(c)
    }
    catch (e:Exception)
    {
        e.printStackTrace()
    }
    finally {
        println("đoạn code muốn thực thi, kể cả khi có lỗi")
    }
    println("Đoạn code phía sau")
}
```

```
java.lang.ArithmetricException Create breakpoint : / by zero
  at Bai16_bietleKt.main(bai16-bietle.kt:22)
đoạn code muốn thực thi, kể cả khi có lỗi
Đoạn code phía sau
```

2
 đoạn code muốn thực thi, kể cả khi có lỗi
 Đoạn code phía sau



2

throw statement

```
fun Thuong(a:Int, b:Int):Int
{
    if (b==0)
        throw Exception("Mẫu = 0 sao chia")
    return a/b
}
fun main(args: Array<String>) {
    Thuong( a: 5, b: 0)
}
```

Exception in thread "main" java.lang.Exception
at Bai16_bietleKt.Thuong([bai16-bietle.kt:4](#))
at Bai16_bietleKt.main([bai16-bietle.kt:8](#))

```
fun Thuong(a:Int, b:Int):Int
{
    if (b==0)
        throw Exception("Mẫu = 0 sao chia")
    return a/b
}
fun main(args: Array<String>) {
    try
    {
        Thuong( a: 5, b: 0)
    }
    catch (e:Exception)
    {
        println(e.message)
    }
    println("Đoạn code phía sau")
}
```

Mẫu = 0 sao chia
Đoạn code phía sau





GET IT ON
Google Play

Kotlin
Full Course
Bài 17

A smartphone screen displays a "GET IT ON Google Play" button. Below it is a green Android robot icon with a blue Kotlin logo on its chest. The text "Kotlin Full Course" is displayed in large red letters, followed by "Bài 17" in red underlined text.

- 1 Datetime
- 2 Ví dụ vận dụng



1

Datetime

1. Khởi tạo :

```
//trả về ngày tháng hiện tại  
var cal = Calendar.getInstance()  
println(cal)
```

```
minimalDaysInFirstWeek=1,ERA=1,YEAR=2022,MONTH=5,WEEK_OF_YEAR=27,WEEK_OF_MONTH=5,DAY_OF_MONTH=29,DAY_OF_YEAR=180
```

2. Lấy thông tin ngày, tháng, năm ... :

```
//get năm, tháng, ngày ...  
var nam = cal.get(Calendar.YEAR)  
var thang = cal.get(Calendar.MONTH)  
var ngay = cal.get(Calendar.DAY_OF_MONTH)  
println("năm hiện tại = $nam")  
//Chú ý tháng luôn chạy từ 0->11  
println("tháng hiện tại = $thang")  
println("ngày hiện tại = $ngay")
```

```
năm hiện tại = 2022  
tháng hiện tại = 5  
ngày hiện tại = 29
```



1

Datetime

3. Set ngày tháng theo ý muốn :

```
//.3 Đặt ngày tháng năm  
cal.set(Calendar.YEAR, 1990)  
cal.set(Calendar.MONTH, 6)  
cal.set(Calendar.DAY_OF_MONTH, 20)  
//kiểm tra năm sau khi thay đổi  
var ngaySet = cal.get(Calendar.DAY_OF_MONTH)  
var thangSet = cal.get(Calendar.MONTH)  
var namSet = cal.get(Calendar.YEAR)  
println("ngày tháng năm set = $ngaySet/$thangSet/$namSet")
```

```
ngày tháng năm set = 20/6/1990
```



1

Datetime

4. Xuất theo định dạng Ngày/ Tháng / Năm

```
var date = cal.time  
// khai báo biến để định dạng  
var dinhDang= SimpleDateFormat( pattern: "dd/MM/yyyy")  
println(dinhDang.format(date))
```

20/07/1990

5. Xuất theo định dạng Ngày/ Tháng / Năm, giờ phút giây

```
//6. xuất ngày tháng năm, giờ phút giây: HH lớn để xuất định dạng 24h  
var dinhDang2=SimpleDateFormat( pattern: "dd/MM/yyyy hh:mm:ss a")  
println(dinhDang2.format(date))
```

20/07/1990 08:09:48 PM



2

Ví dụ vận dụng

6. Ví dụ vận dụng:

*Viết chương trình nhập vào ngày, tháng, năm sinh.
Cho biết số tuổi và in ra màn hình*

```
var dinhDangDate = SimpleDateFormat( pattern: "dd/MM/yyyy")
println("Mời nhập vào ngày tháng năm sinh (dd/MM/yyyy): ")
var s = readLine()
if (s==null)return
// ép kiểu
var dateInput = dinhDangDate.parse(s)
//khởi tạo biến date time
var timeNamSinh = Calendar.getInstance()
//set timeNamSinh = thời gian nhập vào
timeNamSinh.time=dateInput
//get năm sinh, tháng sinh, ngày sinh
var namSinh = timeNamSinh.get(Calendar.YEAR)
var thangSinh = timeNamSinh.get(Calendar.MONTH)
var ngaySinh = timeNamSinh.get(Calendar.DAY_OF_MONTH)
println("Bạn sinh ngày $ngaySinh/$thangSinh/$namSinh ")

//khởi tạo biến date time
var timeHienTai = Calendar.getInstance()
//lấy năm hiện tại
var namHienTai = timeHienTai.get(Calendar.YEAR)
//Tính tuổi
var tuoi = namHienTai - namSinh
println("Bạn sinh năm $namSinh, hiện bạn đã $tuoi tuổi")
```

21/11/1990

Bạn sinh ngày 21/10/1990

Bạn sinh năm 1990, hiện bạn đã 32 tuổi





Kotlin Full Course Bài 18

1

Random (Số Ngẫu Nhiên)

2

Ví dụ vận dụng



1

Random (Số Ngẫu Nhiên)

1. Random số nguyên :

```
//khởi tạo random
var rd = Random
//random số nguyên
var rdNguyen = rd.nextInt( until: 10)
// chạy từ 0 đến 9
println("Số ngẫu nhiên : $rdNguyen")
```

Số ngẫu nhiên : 2

Random số nguyên trong đoạn [a..b) từ a đến sát b

```
//random số nguyên [a,b)
var rd2 = rd.nextInt( from: -1, until: 1)
println(rd2)
```



1

Random (Số Ngẫu Nhiên)

2. Random số thực :

- ✓ Các số thực chạy từ [0..1) : chạy từ 0 đến sát 1

```
//rd số thực  
var rd3 = rd.nextDouble()  
println(rd3)
```

0.2009583462694734

- ✓ Random số thực > 1 :

- **Cách 1:** Có thể ghép random nguyên và phần thập phân để tạo

```
//random số thực >1  
var rd3 = rd.nextInt( from: -10, until: 31)  
var rd4 = rd.nextDouble()  
var rd5 = rd3+rd4  
println(rd5)
```

-2.5963720161866704

- **Cách 2:** Nhân với 10 100 1000...

```
//cách 2 * với 10 100 1000  
var rd6 = rd.nextDouble()*100  
println(rd6)
```

52.633498100025975



2

Ví dụ vận dụng

3. Ví dụ vận dụng:

Viết Game đoán số, máy tính tạo ra 1 số ngẫu nhiên từ 0 đến 99

Người chơi sẽ được đoán 7 lần

Nếu đoán sai, thì báo sai, và thông báo số dự đoán nhỏ hơn hay lớn hơn số máy

Nếu đoán đúng thì báo :Chúc mừng

=> Kết thúc game hỏi người chơi có muốn chơi tiếp?

58

Mời bạn đoán số từ 1-99:

3

Số bạn đoán nhỏ hơn máy đoán

Mời bạn đoán số từ 1-99:

58

Bạn đã đoán đúng, số của bạn = số máy = 58

Bạn có muốn chơi tiếp không (y/n)



2

Ví dụ vận dụng

```
fun play()
{
    var rd = Random
    var soMay = rd.nextInt(until: 100)
    //hack
    println(soMay)
    //đếm lượt chơi
    var dem=1
    while (true)
    {
        //cho người dùng nhập vào số dự đoán
        println("Mời bạn đoán số từ 1-99: ")
        var nguoiDoan:Int? = readLine()?.toInt()
        if (nguoiDoan==null) return
        if (dem==7)
        {
            println("Bạn đã chơi hết 7 lần, bạn đã thua")
            break
        }
        else
        {
            dem++
            if (nguoiDoan==soMay)
            {
                println("Bạn đã đoán đúng, số của bạn = số máy = $soMay")
                break
            }
            else if (nguoiDoan < soMay)
                println("Số bạn đoán nhỏ hơn máy đoán")
            else
                println("Số bạn đoán lớn hơn máy đoán")
        }
    }
}
```

```
fun main(args: Array<String>)
{
    while (true)
    {
        play()
        //hỏi chơi tiếp
        println("Bạn có muốn chơi tiếp không (y/n) ")
        var s:String? = readLine()
        if (s!=null)
        {
            if (s=="n" || s=="N")
            {
                println("Cảm ơn bạn đã chơi game, hẹn gặp lại")
                break
            }
        }
    }
}
```





Kotlin Full Course Bài 19



1

Kiểu ký tự Char

Char

A	B	C	...	a	b
65	66	67	...	97	98

Value

ASCII table

2

Char method



1

Kiểu ký tự Char

 1. Cú pháp:

Cách 1: var <tên biến>:Char ='<ký tự>'

```
var kyTu:Char = 'a'
```

Cách 2: var <tên biến>:Char = Char(mã ASCII)

```
var kyTu2:Char =Char( code: 64)
println(kyTu2) // @
```

*Chú ý khai báo char phải để trong ngoặc đơn, và chỉ đc khai báo 1 ký tự
=> nếu dùng nháy kép sẽ báo lỗi*

```
var kyTu3:Char ="a"
Type mismatch.
Required: Char
Found:   String
Change type of 'kyTu3' t
```



1

Kiểu ký tự Char

ASCII Table

Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char	Dec	Hex	Oct	Char
0	0	0		32	20	40	[space]	64	40	100	@	96	60	140	'
1	1	1		33	21	41	!	65	41	101	A	97	61	141	a
2	2	2		34	22	42	"	66	42	102	B	98	62	142	b
3	3	3		35	23	43	#	67	43	103	C	99	63	143	c
4	4	4		36	24	44	\$	68	44	104	D	100	64	144	d
5	5	5		37	25	45	%	69	45	105	E	101	65	145	e
6	6	6		38	26	46	&	70	46	106	F	102	66	146	f
7	7	7		39	27	47	'	71	47	107	G	103	67	147	g
8	8	10		40	28	50	(72	48	110	H	104	68	150	h
9	9	11		41	29	51)	73	49	111	I	105	69	151	i
10	A	12		42	2A	52	*	74	4A	112	J	106	6A	152	j
11	B	13		43	2B	53	+	75	4B	113	K	107	6B	153	k
12	C	14		44	2C	54	,	76	4C	114	L	108	6C	154	l
13	D	15		45	2D	55	-	77	4D	115	M	109	6D	155	m
14	E	16		46	2E	56	.	78	4E	116	N	110	6E	156	n
15	F	17		47	2F	57	/	79	4F	117	O	111	6F	157	o
16	10	20		48	30	60	0	80	50	120	P	112	70	160	p
17	11	21		49	31	61	1	81	51	121	Q	113	71	161	q
18	12	22		50	32	62	2	82	52	122	R	114	72	162	r
19	13	23		51	33	63	3	83	53	123	S	115	73	163	s
20	14	24		52	34	64	4	84	54	124	T	116	74	164	t
21	15	25		53	35	65	5	85	55	125	U	117	75	165	u
22	16	26		54	36	66	6	86	56	126	V	118	76	166	v
23	17	27		55	37	67	7	87	57	127	W	119	77	167	w
24	18	30		56	38	70	8	88	58	130	X	120	78	170	x
25	19	31		57	39	71	9	89	59	131	Y	121	79	171	y
26	1A	32		58	3A	72	:	90	5A	132	Z	122	7A	172	z
27	1B	33		59	3B	73	;	91	5B	133	[123	7B	173	{
28	1C	34		60	3C	74	<	92	5C	134	\	124	7C	174	
29	1D	35		61	3D	75	=	93	5D	135]	125	7D	175	}
30	1E	36		62	3E	76	>	94	5E	136	^	126	7E	176	~
31	1F	37		63	3F	77	?	95	5F	137	_	127	7F	177	



1

Kiểu ký tự Char

□ 2. Ép kiểu char cho dữ liệu nhập vào :

```
println("Mời nhập vào 1 ký tự: ")
var s:String? = readLine()
if (s==null) return
var kt:Char = s.first() //cách 1
var kt2:Char = s[0]     //cách 2
var kt3:Char = s.single()//cách 3
println(kt)
println(kt2)
println(kt3)
println(kt::class.java.typeName)
println(kt2::class.java.typeName)
println(kt3::class.java.typeName)
```



```
Mời nhập vào 1 ký tự:
r
r
r
r
char
char
char
```

Chú ý string cũng chỉ đc là 1 ký tự,

- ✓ ví dụ 1 string có 1 ký tự là b
- ✓ ví dụ 2 string có 1 ký tự là c



1

Kiểu ký tự Char

□ 3. compareTo: So sánh 2 ký tự => trả về 1 số nguyên:

```
//1. ký tự giống nhau: trả về 0  
println('a'.compareTo('a'))  
//2. ký tự 1 < 2 => trả về âm  
println('a'.compareTo('b'))  
//3. ký tự 1 > 2 trả về dương  
println('b'.compareTo('a'))  
println('m'.compareTo('a'))
```



0
-1
1
1

✓ Tóm lại : (So sánh dựa vào mã ASCII)

- ❖ 2 ký tự trùng khớp nhau thì Compare =0
- ❖ Ký tự 1 < ký tự 2 thì kết quả âm
- ❖ Ký tự 1 > Ký tự 2 thì kết quả dương

□ 4. Equals : So sánh 2 ký tự => trả về True, False

```
//so sánh bằng nhau  
println('a'.equals('b')) // false  
println('a'.equals('a')) // true
```



2

Char method

STT	Cú pháp	Nội dung	Ví dụ
1	<code>Char.isDigit()</code>	True nếu <code>ch</code> truyền vào là <code>chữ số</code> (0,1,2....)	<code>'1'.isDigit()</code>
2	<code>Char.isLetter()</code>	True nếu <code>ch</code> truyền vào là <code>chữ cái</code> (a,b,c,A..)	<code>'a'.isLetter()</code>
3	<code>Char.IsWhiteSpace()</code>	True nếu <code>ch</code> truyền vào là <code>space</code>	<code>' '.isWhitespace()</code>
4	<code>Char.isLowerCase()</code>	True nếu <code>ch</code> truyền vào là ký tự thường	<code>'a'.isLowerCase()</code>
5	<code>Char.isUpperCase()</code>	True nếu <code>ch</code> truyền vào là ký tự viết hoa	<code>'A'.isUpperCase()</code>

```

println('1'.isDigit()) //true
println('a'.isLetter()) //true
println(' '.isWhitespace()) //true
println('a'.isLowerCase()) //true
println('A'.isUpperCase()) //true
  
```





GET IT ON
Google Play



Kotlin

Full Course

Bài 20.2



1

String Kotlin (kiểu chuỗi)

2

Ví dụ vận dụng

3

Bài tập String Kotlin 18 - 22



1

String Kotlin (kiểu chuỗi)

□ 1. Khái niệm :

- ✓ Trong kotlin String là 1 lớp quản lý dữ liệu văn bản
- ✓ Chuỗi là tập hợp các ký tự ex: “Abc dắt dê đi ...học “

□ 2. Khai báo chuỗi :

```
var <điều kiện> :String =“ <xâu ký tự>”
```

```
var <điều kiện> =“ <xâu ký tự> ” // khai báo tắt
```

```
var s1:String = "Abc dắt dê đi học"  
var s2= "Bành thị nòi"  
println(s1)  
println(s2)  
println(s1::class.java.typeName)  
println(s2::class.java.typeName)
```



```
Abc dắt dê đi học  
Bành thị nòi  
java.lang.String  
java.lang.String
```



1

String Kotlin (kiểu chuỗi)

□ 3. Bảng 1 số ký tự đặc biệt:

STT	Ký tự	Ý nghĩa
1	\"	Dấu nháy kép
2	\`	Dấu chéo
3	\n	Dòng mới
4	\t	Tab ngang

```

var s3:String = "D:\\galailaptrinh\\\\Maria"
var s4:String = "C:\\Program Files (x86)"
var s5:String = "Có người nói rằng: \"Abc ...\""
var s6:String = "dòng 1 \n dòng 2"
var s7:String = "một \t hai"
    
```



```

D:\galailaptrinh\Maria
C:\Program Files (x86)
Có người nói rằng: "Abc ..."
dòng 1
dòng 2
một      hai
    
```



1

String Kotlin (kiểu chuỗi)

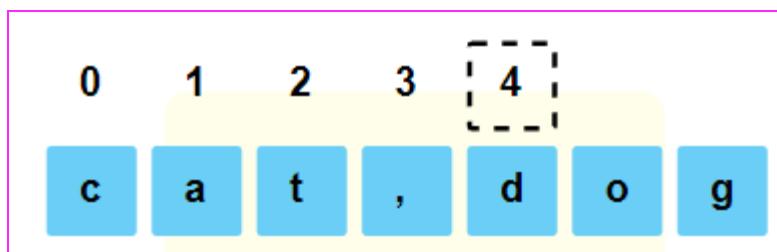
- 4 . Length: Trả về độ dài của chuỗi (số ký tự chuỗi, bao gồm cả space)

```
var s8:String ="Em học tại tuhoc.cc"  
println("Độ dài s8= "+s8.length)
```



Độ dài s8= 19

- 5 . Xuất ký tự theo vị trí index



Chú ý index text bắt đầu từ 0

```
var s8:String ="Em học tại tuhoc.cc"  
println("Độ dài s8= "+s8.length)  
//xuất ký tự thứ i theo index  
println("ký tự có index 1 ="+s8[1])
```



ký tự có index 1 =



1

String Kotlin (kiểu chuỗi)

□ 6 . indexOf

Kiểm tra vị trí xuất hiện đầu tiên của ký tự hoặc chuỗi, trả về -1 nếu không tìm thấy

```
var s9:String = "abcdeaf"  
println(s9.indexOf("a"))  
println(s9.indexOf("g"))
```



```
0  
-1
```

□ 7 . lastIndexOf

Giống indexOf nhưng trả về vị trí index xuất hiện cuối cùng trả về -1 nếu không tìm thấy

```
var s9:String = "abcdeaf"  
println(s9.lastIndexOf("a"))  
println(s9.lastIndexOf("g"))
```



```
5  
-1
```



1

String Kotlin (kiểu chuỗi)

□ 8. Contains : Kiểm tra chuỗi con

```

var s10=".mp3"
var s11="tuhoc.mp3"
//kiểm tra s11 có chứa s10 không?
var check:Boolean =s11.contains(s10)
if (check) // viết tương tự: check==true
    println("Có .mp3 trong chuỗi")
else
    println("không tìm thấy .mp3 trong chuỗi")

```



Có .mp3 trong chuỗi

□ 9 . Substring (trích lọc chuỗi con từ chuỗi ban đầu)

```

var s12= "Em học lập trình tại tuhoc.cc"
// lấy từ index 2 đến hết
var s13=s12.substring( startIndex: 2)
//lấy từ index 2 đến sát 10
var s14 =s12.substring(2,10)
println(s12)
println(s13)
println(s14)

```



Em học lập trình tại tuhoc.cc
học lập trình tại tuhoc.cc
học lập



1

String Kotlin (kiểu chuỗi)

□ 10 . Replace("str old", "str new", "ignoreCase= true/false")

Thay thế toàn bộ chuỗi old bằng chuỗi new

ignoreCase:

Bỏ qua phân biệt hoa thường nếu không truyền, kotlin tự hiểu = false

```
//replace
var s15 = "Học học nữa học mãi"
//lưu ý: replace không làm thay đổi nội tại biến s15
var s16 = s15.replace( oldValue: "học", newValue: "ngù")
println(s15)
println(s16)
```



Học học nữa học mãi
Học ngù nữa ngù mãi

```
//replace không phân biệt hoa thường
var s17 = s15.replace( oldValue: "học", newValue: "ngù", ignoreCase: true)
println(s15)
println(s17)
```



Học học nữa học mãi
ngù ngù nữa ngù mãi



1

String Kotlin (kiểu chuỗi)

□11 . ReplaceFirst ("str old", "str new", "ignoreCase= true/false")

Thay thế chuỗi old đầu tiên tìm thấy bằng chuỗi new

ignoreCase:

Bỏ qua phân biệt hoa thường nếu không truyền, kotlin tự hiểu = false

```
var s18 = s15.replaceFirst( oldValue: "học", newValue: "Thay", ignoreCase: true)
println(s15)
println(s18)
```

```
Học học nữa học mãi
Thay học nữa học mãi
```





Kotlin Full Course Bài 20.7

Giải bài tập kotlin 22

□ Kotlin 22 :

Viết chương trình tách số và chữ từ chuỗi nhập vào thành 2 chuỗi :

* ví dụ nhập vào : abc123 sẽ tách và in ra thành 2 chuỗi abc và 123



Giải bài tập kotlin 18



1

String Kotlin (kiểu chuỗi)

□ 1. Khái niệm :

- ✓ Trong kotlin String là 1 lớp quản lý dữ liệu văn bản
- ✓ Chuỗi là tập hợp các ký tự ex: “Abc dắt dê đi ... học “

□ 2. Khai báo chuỗi :

```
var <điều kiện> :String = " <xâu ký tự>"
```

```
var <điều kiện> = " <xâu ký tự>" // khai báo tắt
```

```
var s1:String = "Abc dắt dê đi học"  
var s2= "Bành thị nòi"  
println(s1)  
println(s2)  
println(s1::class.java.typeName)  
println(s2::class.java.typeName)
```



```
Abc dắt dê đi học  
Bành thị nòi  
java.lang.String  
java.lang.String
```



1

String Kotlin (kiểu chuỗi)

□ 3. Bảng 1 số ký tự đặc biệt:

STT	Ký tự	Ý nghĩa
1	\"	Dấu nháy kép
2	\`	Dấu chéo
3	\n	Dòng mới
4	\t	Tab ngang

```

var s3:String = "D:\\galailaptrinh\\\\Maria"
var s4:String = "C:\\Program Files (x86)"
var s5:String = "Có người nói rằng: \"Abc ...\""
var s6:String = "dòng 1 \n dòng 2"
var s7:String = "một \t hai"
    
```



```

D:\galailaptrinh\Maria
C:\Program Files (x86)
Có người nói rằng: "Abc ..."
dòng 1
dòng 2
một      hai
    
```



1

String Kotlin (kiểu chuỗi)

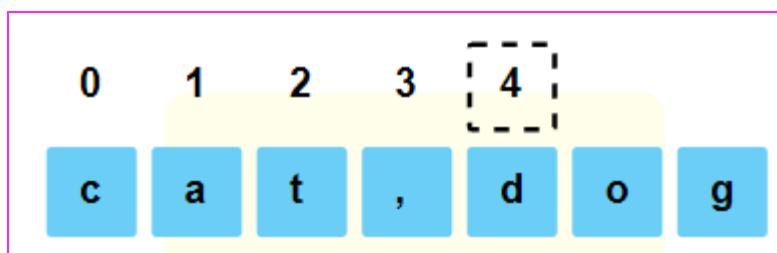
- 4 . Length: Trả về độ dài của chuỗi (số ký tự chuỗi, bao gồm cả space)

```
var s8:String ="Em học tại tuhoc.cc"  
println("Độ dài s8= "+s8.length)
```



Độ dài s8= 19

- 5 . Xuất ký tự theo vị trí index



Chú ý index text bắt đầu từ 0

```
var s8:String ="Em học tại tuhoc.cc"  
println("Độ dài s8= "+s8.length)  
//xuất ký tự thứ i theo index  
println("ký tự có index 1 ="+s8[1])
```



ký tự có index 1 =



1

String Kotlin (kiểu chuỗi)

□ 6 . indexOf

Kiểm tra vị trí xuất hiện đầu tiên của ký tự hoặc chuỗi, trả về -1 nếu không tìm thấy

```
var s9:String = "abcdeaf"  
println(s9.indexOf("a"))  
println(s9.indexOf("g"))
```



```
0  
-1
```

□ 7 . lastIndexOf

Giống indexOf nhưng trả về vị trí index xuất hiện cuối cùng trả về -1 nếu không tìm thấy

```
var s9:String = "abcdeaf"  
println(s9.lastIndexOf("a"))  
println(s9.lastIndexOf("g"))
```



```
5  
-1
```



1

String Kotlin (kiểu chuỗi)

□ 8. Contains : Kiểm tra chuỗi con

```

var s10=".mp3"
var s11="tuhoc.mp3"
//kiểm tra s11 có chứa s10 không?
var check:Boolean =s11.contains(s10)
if (check) // viết tương tự: check==true
    println("Có .mp3 trong chuỗi")
else
    println("không tìm thấy .mp3 trong chuỗi")

```



Có .mp3 trong chuỗi

□ 9 . Substring (trích lọc chuỗi con từ chuỗi ban đầu)

```

var s12= "Em học lập trình tại tuhoc.cc"
// lấy từ index 2 đến hết
var s13=s12.substring( startIndex: 2)
//lấy từ index 2 đến sát 10
var s14 =s12.substring(2,10)
println(s12)
println(s13)
println(s14)

```



Em học lập trình tại tuhoc.cc
học lập trình tại tuhoc.cc
học lập



1

String Kotlin (kiểu chuỗi)

□ 10 . Replace("str old", "str new", "ignoreCase= true/false")

Thay thế toàn bộ chuỗi old bằng chuỗi new

ignoreCase:

Bỏ qua phân biệt hoa thường nếu không truyền, kotlin tự hiểu = false

```
//replace
var s15 = "Học học nữa học mãi"
//lưu ý: replace không làm thay đổi nội tại biến s15
var s16 = s15.replace( oldValue: "học", newValue: "ngù")
println(s15)
println(s16)
```



Học học nữa học mãi
Học ngù nữa ngù mãi

```
//replace không phân biệt hoa thường
var s17 = s15.replace( oldValue: "học", newValue: "ngù", ignoreCase: true)
println(s15)
println(s17)
```



Học học nữa học mãi
ngù ngù nữa ngù mãi



1

String Kotlin (kiểu chuỗi)

□11 . ReplaceFirst ("str old", "str new", "ignoreCase= true/false")

Thay thế chuỗi old đầu tiên tìm thấy bằng chuỗi new

ignoreCase:

Bỏ qua phân biệt hoa thường nếu không truyền, kotlin tự hiểu = false

```
var s18 = s15.replaceFirst( oldValue: "học", newValue: "Thay", ignoreCase: true)
println(s15)
println(s18)
```

```
Học học nữa học mãi
Thay học nữa học mãi
```





Kotlin Full Course Bài 21

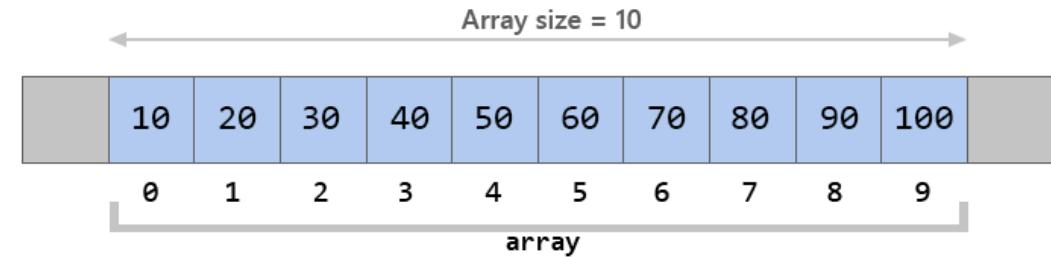


1

Mảng 1 chiều Kotlin

2

Bài tập kotlin 23



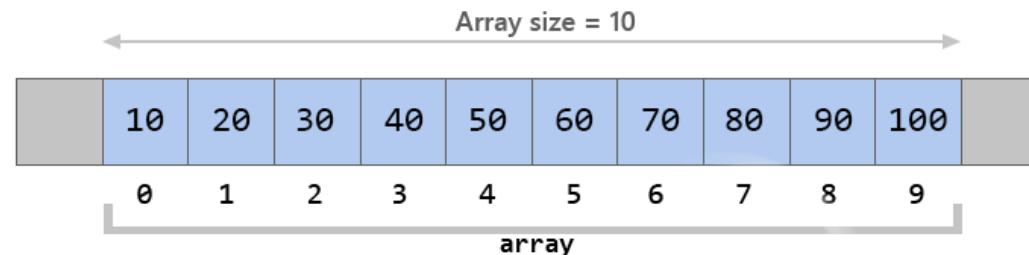
1

Mảng 1 chiều Kotlin

- 1. Khái niệm : Mảng là tập hợp các đối tượng có cùng kiểu dữ liệu.

Chúng có kích thước cố định không thể thay đổi

Mảng có index bắt đầu từ 0



- 2 . Tại sao phải dùng mảng:

Ví dụ : Chúng ta có khoảng 50 điểm của học sinh cần lưu, nếu không dùng mảng thì chúng ta phải khai báo 50 biến float

=> Gom nhóm các đối tượng có chung tính chất lại với nhau giúp code gọn gàng hơn.



1

Mảng 1 chiều Kotlin

□3. Khai báo mảng :

Tên mảng

Kiểu dữ liệu

Số phần tử của mảng

```
var M:IntArray = IntArray(size: 5) //mảng 5 pt là các số 0
var M2:FloatArray = FloatArray(size: 4) //mảng 4 pt là các số 0
println(M::class.java.typeName) // int[]
```

```
//đọc mảng
for (i in M)
    print("$i\t")
```



```
0 0 0 0 0
```

□4. Khởi tạo và gán giá trị cho mảng:

```
//khai báo
var M3:IntArray = intArrayOf(0,2,-5,1,3)
//khai báo tắt (kotlin tự nội suy)
var M4 = intArrayOf(1,2,3,4)
var M5 = arrayOf("tuhoc","kotlin","python")
println(M4::class.java.typeName) // int[]
```



1

Mảng 1 chiều Kotlin

□ 5. Truy xuất, thay đổi phần tử của mảng qua vị trí index :

```
//truy xuất ptu qua vị trí index
var M6= intArrayOf(9,8,7,6,5)
println(M6[0]) //9
println(M6[1]) //8
println(M6[2]) //7
```

```
//thay đổi giá trị của mảng qua index
M6[0]=1
M6[1]=1
M6[2]=3
println(M6[0]) //1
println(M6[1]) //1
println(M6[2]) //3
```

□6. Khởi tạo mảng ngẫu nhiên :

```
var rd = Random
var M7:IntArray = IntArray(size: 7)
println(M7.indices) // trả về đoạn 0..6
//đuýet từng phần tử mảng
for (i in M7.indices)
{
    //println(i)
    M7[i] = rd.nextInt(until: 100)
}
```

```
//xuất mảng cách 1
for (j in M7)
    print("$j\t")
```

```
//xuất mảng cách 2
for (j in M7.indices)
    print("${M7[j]}\t")
```

95	74	98	29	34	49	27
95	74	98	29	34	49	27



1

Mảng 1 chiều Kotlin

□7. Size : trả về số phần tử của mảng

```
println("Số phần tử của mảng 7 là:" + M7.size)
```



Số phần tử của mảng 7 là:7

□8. Phép gán mảng :

**** gán mảng mới = mảng đã tồn tại** (tham chiếu cùng vùng nhớ)

chú ý : khi thay đổi giá trị của 1 mảng, thì mảng còn lại cũng thay đổi , // do mảng là kiểu tham chiếu, nó sẽ tham chiếu đến 1 vùng nhớ.

```
var M9 = arrayOf(1,2,3,4,5)
var M10=M9
M9[0]=99
println(M9[0]) // 99
println(M10[0]) //99
```

□9. Clone (tạo ra mảng mới trên vùng nhớ mới)

```
var M11 = arrayOf(1,2,3,4,5)
var M12=M11.clone()
M11[0] =99 // thay đổi phần tử index 0 = 99
println(M11[0]) // 99
println(M12[0]) //1
```



1

Mảng 1 chiều Kotlin

□ 10 . **Array.reverse()** : Đảo ngược mảng

```
var M13 = arrayOf(1,5,8,9)
M13.reverse()
println("Mảng 13 sau khi đảo: ")
for (i in M13)
    print("$i\t")
```



Mảng 13 sau khi đảo:
9 8 5 1

□ 11. **Array.Sort()** (Sắp xếp mảng tăng dần)

```
//12.sắp xếp
var M14 = arrayOf(1,6,13,9)
M14.sort()
println("Mảng 14 sau sxep: ")
for (i in M14)
    print("$i\t")
println()
```



Mảng 14 sau sxep:
1 6 9 13



1

Mảng 1 chiều Kotlin

□ 12 . arr.filter() : Trích lọc mảng theo điều kiện=> trả về 1 ArrayList

```
var M15 = arrayOf(1, 3, 5, 2, 8, 10)
var ds = M15.filter { x->x!=null }
println(ds) // [1, 3, 5, 2, 8, 10]
println(ds::class.java.typeName) // ArrayList
// lọc các số chẵn
var ds2 = M15.filter { y->y%2==0 }
println(ds2) // [2, 8, 10]
```

□ 13. Tìm max, min

```
//tìm max min
val arr2: Array<Int> = arrayOf(6, 3, 2, 5, 10)
arr2.sort();
println("Minimum: ${arr2.first()}")           // Minimum: 2
println("Minimum: ${arr2[0]})")           // Minimum: 2
println("Maximum: ${arr2.last()}")           // Maximum: 10
println("Maximum: ${arr2[arr2.size-1]}")           // Maximum: 10
```



2

Bài tập kotlin 23

- ✓ 1. *Viết chương trình tạo 1 mảng 1 chiều gồm các phần tử là số nguyên, có n phần tử ngẫu nhiên, n do người dùng nhập từ bàn phím*
- ✓ 2. *Xuất các giá trị trong mảng*
- ✓ 3. *Đảo ngược mảng, và xuất mảng sau khi đảo ngược*
- ✓ 4. *Sắp xếp mảng tăng dần*
- ✓ 5. *Tính tổng các phần tử trong mảng*
- ✓ 6. *Cho người dùng nhập 1 số bất kỳ, kiểm tra số đó có tồn tại trong mảng hay không, nếu có xuất ra vị trí index của số đó trong mảng*





Kotlin

Full Course

Bài 21.2



Giải Bài tập kotlin 23

- ✓ 1. Viết chương trình tạo 1 mảng 1 chiều gồm các phần tử là số nguyên, có n phần tử ngẫu nhiên, n do người dùng nhập từ bàn phím
- ✓ 2. Xuất các giá trị trong mảng
- ✓ 3. Đảo ngược mảng, và xuất mảng sau khi đảo ngược
- ✓ 4. Sắp xếp mảng tăng dần
- ✓ 5. Tính tổng các phần tử trong mảng
- ✓ 6. Cho người dùng nhập 1 số bất kỳ, kiểm tra số đó có tồn tại trong mảng hay không, nếu có xuất ra vị trí index của số đó trong mảng



Giải Bài tập kotlin 23

```
//1. tạo mảng n số nguyên
println("Mời nhập số phần tử của mảng: ")
var s:String?= readLine()
if (s==null) return
//ép kiểu, gán vào biến n
var n=s.toInt()
//tạo mảng ngẫu nhiên
var rd = Random
var M:IntArray = IntArray(n)
//duyệt từng phần tử, và gán giá trị ngẫu nhiên
for (i in M.indices)
    M[i] =rd.nextInt(until: 101)

//2. Xuất giá trị của mảng
println("Mảng vừa tạo là: ")
for (i in M.indices)
    print("${M[i]}\t")
println()
```

```
//3. Đảo ngược mảng
M.reverse()
println("Mảng vừa đảo ngược là: ")
for (i in M.indices)
    print("${M[i]}\t")
println()

//4. Sắp xếp tăng dần
M.sort()
println("Mảng sau sx tăng dần là: ")
for (i in M.indices)
    print("${M[i]}\t")
println()

//5. tính tổng các ptu trong mảng
println("tổng các phần tử: " +M.sum())
```



Giải Bài tập kotlin 23

```
//6. Cho ng dùng nhập 1 số a, kiểm tra a có trong mảng?  
//in ra vị trí index  
println("Mời cụ nhập vào 1 số a: ")  
s= readLine()  
if (s==null) return  
//ép kiểu gán vào biến a  
var a:Int = s.toInt()  
if (M.contains(a))  
{  
    println("Có $a trong mảng M")  
    //in ra vị trí index  
    println("Vị trí index số a trong mảng")  
    for (i in M.indices)  
        if (M[i]==a)  
            print("$i\t")  
}  
else  
    println("Không có $a trong mảng M")
```



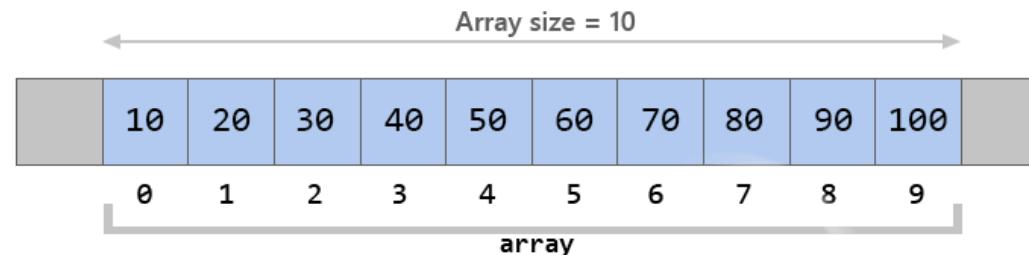
1

Mảng 1 chiều Kotlin

- 1. Khái niệm : Mảng là tập hợp các đối tượng có cùng kiểu dữ liệu.

Chúng có kích thước cố định không thể thay đổi

Mảng có index bắt đầu từ 0



- 2 . Tại sao phải dùng mảng:

Ví dụ : Chúng ta có khoảng 50 điểm của học sinh cần lưu, nếu không dùng mảng thì chúng ta phải khai báo 50 biến float

=> Gom nhóm các đối tượng có chung tính chất lại với nhau giúp code gọn gàng hơn.



1

Mảng 1 chiều Kotlin

□3. Khai báo mảng :

Tên mảng

Kiểu dữ liệu

Số phần tử của mảng

```
var M:IntArray = IntArray(size: 5) //mảng 5 pt là các số 0
var M2:FloatArray = FloatArray(size: 4) //mảng 4 pt là các số 0
println(M::class.java.typeName) // int[]
```

```
//đọc mảng
for (i in M)
    print("$i\t")
```



```
0 0 0 0 0
```

□4. Khởi tạo và gán giá trị cho mảng:

```
//khai báo
var M3:IntArray = intArrayOf(0,2,-5,1,3)
//khai báo tắt (kotlin tự nội suy)
var M4 = intArrayOf(1,2,3,4)
var M5 = arrayOf("tuhoc","kotlin","python")
println(M4::class.java.typeName) // int[]
```



1

Mảng 1 chiều Kotlin

□ 5. Truy xuất, thay đổi phần tử của mảng qua vị trí index :

```
//truy xuất ptu qua vị trí index
var M6= intArrayOf(9,8,7,6,5)
println(M6[0]) //9
println(M6[1]) //8
println(M6[2]) //7
```

```
//thay đổi giá trị của mảng qua index
M6[0]=1
M6[1]=1
M6[2]=3
println(M6[0]) //1
println(M6[1]) //1
println(M6[2]) //3
```

□6. Khởi tạo mảng ngẫu nhiên :

```
var rd = Random
var M7:IntArray = IntArray(size: 7)
println(M7.indices) // trả về đoạn 0..6
//đuýet từng phần tử mảng
for (i in M7.indices)
{
    //println(i)
    M7[i] = rd.nextInt(until: 100)
}
```

```
//xuất mảng cách 1
for (j in M7)
    print("$j\t")
```

```
//xuất mảng cách 2
for (j in M7.indices)
    print("${M7[j]}\t")
```

95	74	98	29	34	49	27
95	74	98	29	34	49	27



1

Mảng 1 chiều Kotlin

□7. Size : trả về số phần tử của mảng

```
println("Số phần tử của mảng 7 là:" + M7.size)
```



Số phần tử của mảng 7 là:7

□8. Phép gán mảng :

**** gán mảng mới = mảng đã tồn tại** (tham chiếu cùng vùng nhớ)

chú ý : khi thay đổi giá trị của 1 mảng, thì mảng còn lại cũng thay đổi , // do mảng là kiểu tham chiếu, nó sẽ tham chiếu đến 1 vùng nhớ.

```
var M9 = arrayOf(1,2,3,4,5)
var M10=M9
M9[0]=99
println(M9[0]) // 99
println(M10[0]) //99
```

□9. Clone (tạo ra mảng mới trên vùng nhớ mới)

```
var M11 = arrayOf(1,2,3,4,5)
var M12=M11.clone()
M11[0] =99 // thay đổi phần tử index 0 = 99
println(M11[0]) // 99
println(M12[0]) //1
```



1

Mảng 1 chiều Kotlin

□ 10 . **Array.reverse()** : Đảo ngược mảng

```
var M13 = arrayOf(1,5,8,9)
M13.reverse()
println("Mảng 13 sau khi đảo: ")
for (i in M13)
    print("$i\t")
```



Mảng 13 sau khi đảo:
9 8 5 1

□ 11. **Array.Sort()** (Sắp xếp mảng tăng dần)

```
//12.sắp xếp
var M14 = arrayOf(1,6,13,9)
M14.sort()
println("Mảng 14 sau sxep: ")
for (i in M14)
    print("$i\t")
println()
```



Mảng 14 sau sxep:
1 6 9 13



1

Mảng 1 chiều Kotlin

□ 12 . arr.filter() : Trích lọc mảng theo điều kiện=> trả về 1 ArrayList

```
var M15 = arrayOf(1, 3, 5, 2, 8, 10)
var ds = M15.filter { x->x!=null }
println(ds) // [1, 3, 5, 2, 8, 10]
println(ds::class.java.typeName) // ArrayList
// lọc các số chẵn
var ds2 = M15.filter { y->y%2==0 }
println(ds2) // [2, 8, 10]
```

□ 13. Tìm max, min

```
//tìm max min
val arr2: Array<Int> = arrayOf(6, 3, 2, 5, 10)
arr2.sort();
println("Minimum: ${arr2.first()}")           // Minimum: 2
println("Minimum: ${arr2[0]})")           // Minimum: 2
println("Maximum: ${arr2.last()}")           // Maximum: 10
println("Maximum: ${arr2[arr2.size-1]}")           // Maximum: 10
```



2

Bài tập kotlin 23

- ✓ 1. *Viết chương trình tạo 1 mảng 1 chiều gồm các phần tử là số nguyên, có n phần tử ngẫu nhiên, n do người dùng nhập từ bàn phím*
- ✓ 2. *Xuất các giá trị trong mảng*
- ✓ 3. *Đảo ngược mảng, và xuất mảng sau khi đảo ngược*
- ✓ 4. *Sắp xếp mảng tăng dần*
- ✓ 5. *Tính tổng các phần tử trong mảng*
- ✓ 6. *Cho người dùng nhập 1 số bất kỳ, kiểm tra số đó có tồn tại trong mảng hay không, nếu có xuất ra vị trí index của số đó trong mảng*





Kotlin

Full Course

Bài 22

Mảng 2 chiều Kotlin



	Cột 0	Cột 1	Cột 2	Cột 3
Dòng 0	M[0][0] 7	M[0][1] 2	M[0][2] 9	M[0][3] 0
Dòng 1	M[1][0] 9	M[1][1] 5	M[1][2] 4	M[1][3] 1
Dòng 2	M[2][0] 8	M[2][1] 0	M[2][2] 3	M[2][3] 6
Tên mảng			Chỉ số cột	Chỉ số dòng



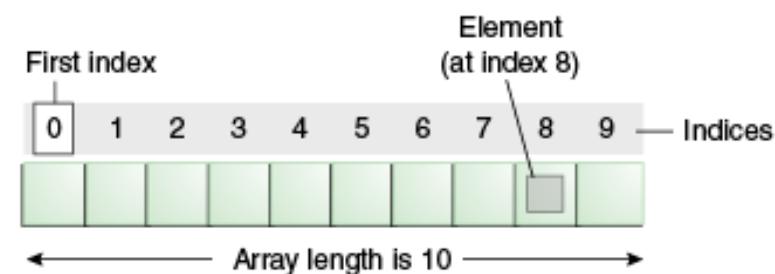
1

Mảng 2 chiều Kotlin

□ 1. Khái niệm : *Mảng 2 chiều hay ma trận là tập hợp nhiều mảng 1 chiều cùng kích thước*

	Cột 0	Cột 1	Cột 2	Cột 3
Dòng 0	M[0][0] 7	M[0][1] 2	M[0][2] 9	M[0][3] 0
Dòng 1	M[1][0] 9	M[1][1] 5	M[1][2] 4	M[1][3] 1
Dòng 2	M[2][0] 8	M[2][1] 0	M[2][2] 3	M[2][3] 6
Tên mảng				

Chỉ số cột
Chỉ số dòng



1

Mảng 2 chiều Kotlin

□2. Khai báo mảng :



1

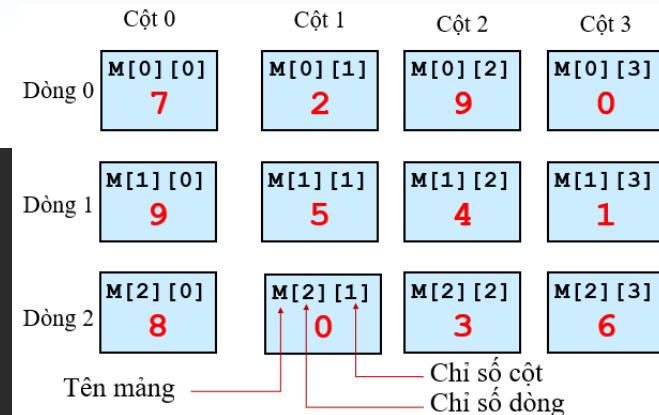
Mảng 2 chiều Kotlin

□3. Khởi tạo mảng ngẫu nhiên :

```

var M3:Array<IntArray> = Array( size: 3, { IntArray( size: 4 ) })
var rd = Random
println(M3.indices) // chạy chỉ số hàng 0..2
for (i in M3.indices)
{
    for (j in M3[i].indices)
    {
        M3[i][j]=rd.nextInt(until: 51)
    }
}

```



```

//xuất phần tử thứ i, j
println(M3[0][0])
println(M3[0][1])
println(M3[0][2])

```

□4. Xuất mảng :

```

for (i in M3.indices)
{
    for (j in M3[i].indices)
        print("$i$j \t")
        //print("${M3[i][j]}\t")
    println()
}

```

00	01	02	03
10	11	12	13
20	21	22	23

```

for (i in M3.indices)
{
    for (j in M3[i].indices)
        //print("$i$j \t")
        print("${M3[i][j]}\t")
    println()
}

```

8	30	25	26
3	33	47	20
44	38	4	17





Kotlin Full Course Bài 23



1 List Kotlin

2 Các phương thức

3 Bài tập Kotlin 24- 28



1

List Kotlin

1. Lời nói đầu:

- * *Do hạn chế của mảng : không thể thêm , xóa phần tử*
- * *Kotlin cung cấp list, chúng ta có thể thêm, xóa, và thay đổi kích cỡ của list !*

□ 2. MutableList và List :

✓ *MutableList* Là collection có thể thay đổi kích thước dữ liệu: Có thể thêm, sửa, xóa...

✓ *List* là collection chỉ có nhiệm vụ *readOnly*, dùng để hiển thị thông tin.
Ưu điểm : *List* tối ưu bộ nhớ hơn so với *MutableList*. Do đó nếu chỉ để hiển thị thông tin thì nên dùng *List*.



1

List Kotlin

3. Khai báo list :

```
//1. Khai báo MutableList có thể thay đổi giá trị  
var ds1:MutableList<Int> = mutableListOf()  
println(ds1::class.java.typeName)
```



java.util.ArrayList

```
//2. Khai báo List (chi đọc, không thêm, sửa dc...)  
var ds2>List<Int> = listOf()  
println(ds2::class.java.typeName)
```



java.util.Arrays\$ArrayList

□4.Khởi tạo danh sách có sẵn một số phần tử:

```
//1. khởi tạo list có thể thêm, sửa xóa được  
var ds3:MutableList<Int> = mutableListOf(1,2,3,4,5,6)  
println(ds3) // [1, 2, 3, 4, 5, 6]  
  
//2. khởi tạo list read only  
var ds4>List<Int> = listOf(2,4,6,8)  
println(ds4) // [2, 4, 6, 8]
```



1

List Kotlin

5. Duyệt list :

```
var ds5:MutableList<Int> = mutableListOf(1,5,8,7)
println(ds5.indices) //0..3
for(i in ds5.indices) // = i in 0..3
    //print(i) => trả về index 0,1,2,3
    print("${ds5[i]}\t")
```



1 5 8 7

```
var ds5:MutableList<Int> = mutableListOf(1,5,8,7)
println(ds5.indices) //0..3
for(i in ds5.indices) // = i in 0..3
{
    //in ra phần tử chia hết cho 2 trong list
    if (ds5[i] %2==0)
        println("số chia hết cho 2 là:" + ds5[i])
}
```



số chia hết cho 2 là:8



2

Các phương thức

STT	<i>phương thức</i>	<i>Nội dung</i>
1	<i>size</i>	<i>Trả về số phần tử của list</i>
2	<i>add()</i>	<i>Thêm một phần tử</i>
3	<i>addAll()</i>	<i>Thêm nhiều phần tử</i>
4	<i>removeAt()</i>	<i>Xóa theo vị trí</i>
5	<i>remove()</i>	<i>Xóa theo đối tượng</i>
6	<i>clear()</i>	<i>Xóa toàn bộ danh sách</i>
7	<i>sort()</i>	<i>Sắp xếp tăng dần</i>
8	<i>sortDescending()</i>	<i>Sắp xếp giảm dần</i>
9	<i>filter { }</i>	<i>Lọc dữ liệu</i>
10	<i>contains()</i>	<i>Kiểm tra Collection có chứa phần tử nào đó hay không</i>

□ 6 . Size (trả về số phần tử)

```
//khởi tạo MutableList  
var ds6:MutableList<Int> = mutableListOf(1,1,4)  
println("danh sách ban đầu: "+ds6)  
  
//1. ktra size (Số phần tử )  
println("Số pt của ds6 là: " +ds6.size)
```



danh sách ban đầu: [1, 1, 4]
Số pt của ds6 là: 3

□ 7 . add (thêm 1 phần tử)

```
//2. add() thêm phần tử  
ds6.add(2) // thêm phần tử vào cuối  
println(ds6) // [1, 1, 4, 2]  
ds6.add(index: 0, element: 3) // thêm phần tử vào vị trí index chỉ định  
println(ds6) // [3, 1, 1, 4, 2]
```



□ 8 . addAll (thêm nhiều phần tử)

```
ds6.addAll(mutableListOf(9,9,9)) //thêm vào cuối  
println("ds sau thêm nhiều: "+ds6)  
//8.2 thêm từ vị trí index  
ds6.addAll(index: 0, mutableListOf(88,88))  
println("ds sau thêm " +ds6)
```



```
ds sau thêm nhiều: [3, 1, 1, 4, 2, 9, 9, 9]  
ds sau thêm [88, 88, 3, 1, 1, 4, 2, 9, 9, 9]
```

□ 9 . removeAt (xóa tại vị trí index chỉ định)

```
ds6.removeAt(index: 2)  
println("ds sau xóa là" +ds6)
```



```
ds sau thêm [88, 88, 3, 1, 1, 4, 2, 9, 9, 9]  
ds sau xóa là[88, 88, 1, 1, 4, 2, 9, 9, 9]
```

□ 10 . remove (xóa phần tử đầu tiên trùng khớp)

```
//remove: xóa pt đầu tiên tìm thấy  
ds6.remove(element: 1)  
println("ds sau xóa số 1" +ds6)
```



```
ds sau xóa số 1[88, 88, 1, 4, 2, 9, 9, 9]
```



□ 11 . sort (sắp xếp tăng dần)

```
//sort (sắp xếp tăng dần)
ds6.sort()
println("ds sau sx tăng" +ds6)
```



```
ds sau sx tăng[1, 2, 4, 9, 9, 9, 88, 88]
```

□ 12 . sortDescending() (sắp xếp giảm dần)

```
ds6.sortDescending()
println("ds sau sx giảm" +ds6)
```



```
ds sau sx giảm[88, 88, 9, 9, 9, 4, 2, 1]
```

□ 13 . filter (lọc phần tử theo điều kiện)

```
//lọc các phần tử <10
var ds7 = ds6.filter { x->x>10 }
println("ds sau lọc >10: "+ds7)
```



```
ds sau lọc >10: [88, 88]
```



❑ 14 . contains()

Kiểm tra Collection có chứa phần tử nào đó hay không

Nếu có trả về true

Không trả về false

```
//contains  
println(ds6.contains(88))
```



true

❑ 15 . clear (xóa trắng list => trả về list có 0 phần tử)

```
ds6.clear() //xóa toàn bộ  
println(ds6)  
println(ds6.size)
```



[]
0

❑ 16 . List readonly Không thêm, sửa, xóa pt được

```
//list readonly sẽ không thêm, sửa hay xóa được  
var ds8:List<Int> = listOf(1,1,5)  
ds8.add
```

Unresolved reference: add ...
Rename reference Alt+Shift+Enter More actions... Alt+Enter



3

Bài tập Kotlin 24- 28

❑ Kotlin 24 :

*Viết chương trình tạo ra 1 list có n phần tử, n nhập từ bàn phím
Các phần tử là số ngẫu nhiên từ (1,100)*

❑ Kotlin 25 :

Viết chương trình nhập vào 1 danh sách list sau đó:

#1. tạo ra 1 list mới bình phương các phần tử

#2.Xác định bao nhiêu phần tử lớn hơn 50

❑ Kotlin 26 :

Viết chương trình trả lời kết quả các phép tính

*quest = {"2 + 5 + 7 =", "5 * 10 =", "sqrt(16) =", "12%2 ="}*

```
2 + 5 + 7 = 5
wrong, the answer is 14
5 * 10 = 50
correct
sqrt(16) = 4
correct
12%2 = |
```



3

Bài tập Kotlin 24- 28

❑ Kotlin 27 :

Viết chương trình nhập vào 1 list

#1 in ra có bao nhiêu số nhỏ hơn 5,

#2 và in ra vị trí index các số đó

❑ Kotlin 28 :

Viết chương trình in số lớn thứ 2 và số nhỏ thứ 2 trong list

2: in ra vị trí index số đó

ví dụ list

lst={1,2,3,4,5}

số lớn thứ 2: 4 , vị trí index trong list là 3

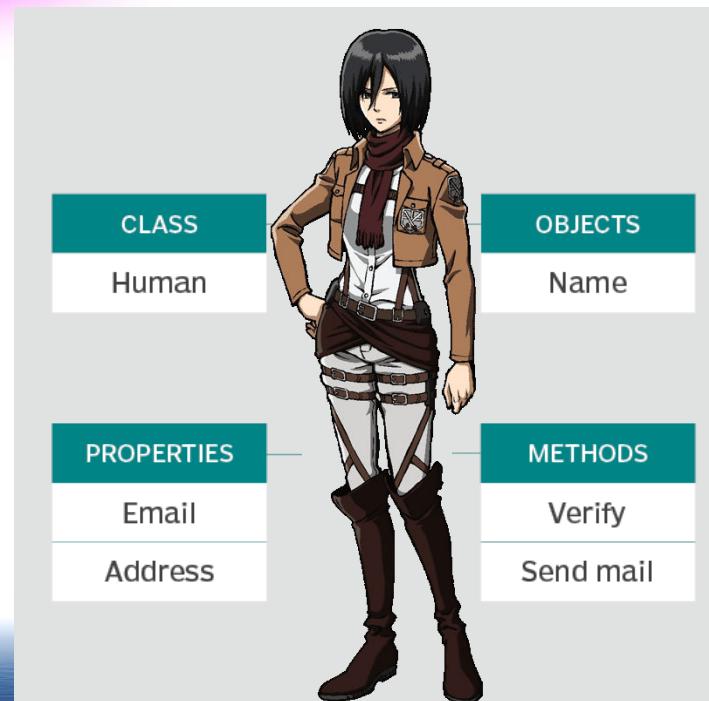
số nhỏ thứ 2 trong list là 2, vị trí index trong list là 1





Hướng đối tượng Kotlin Part 1

- 1 Khái quát OOP
 - 2 Khai báo lớp
 - 3 Constructor



1

Khái quát OOP Kotlin

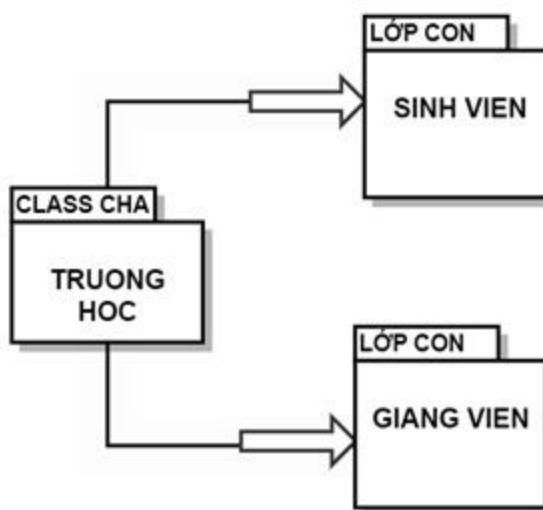
Object-Oriented Programming

1. Khái quát lớp và đối tượng :

- ✓ Đối tượng (object) trong lập trình hướng đối tượng giống như 1 đối tượng cụ thể trong thế giới thực

Mỗi đối tượng có thuộc tính và hành vi riêng

- + Thuộc tính : Đặc điểm của đối tượng
- + Phương thức: Hành vi của đối tượng
- + 1 con chó tên Lucy : là 1 đối tượng cụ thể
- ✓ Các đối tượng có các phương thức, thuộc tính giống nhau được gom thành 1 lớp để dễ quản lý



Thuộc tính (attribute)			
TÊN	CCCD	NĂM SINH	QUÊ QUÁN
HỌC	Chơi Game	Tính DTB	ĐKý Học Phần
Phương Thức (Method)			
DẠY	XEM TIKTOK	CHẨM ĐIỂM	SỬA ĐIỂM



2

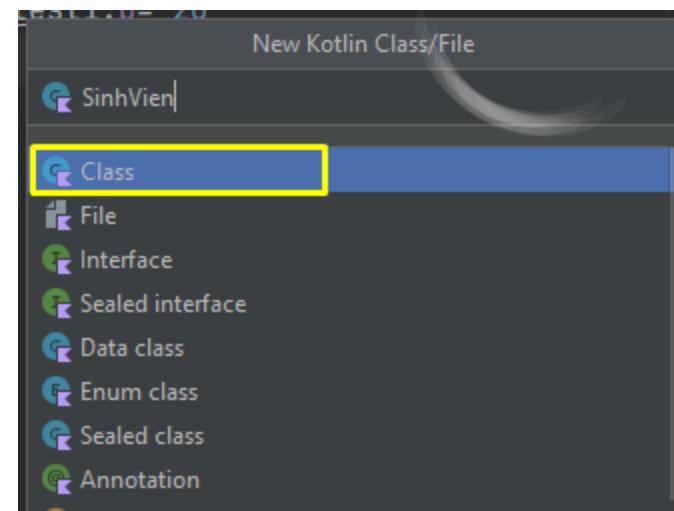
Khai báo lớp

□2 . Tạo class :

Tên lớp nên là 1 danh từ hoặc 1 cụm DT, nên viết hoa ký tự đầu tiên (Car, Bird, Buom, SinhVien)

Cấu trúc chung class

- *Tên Lớp*
- *Các constructors*
- *Các thuộc tính (biến lớp)*
- *Các getter/setter*
- *Các phương thức*



```
class SinhVien {  
}
```



3

constructors

3. Constructors (hàm tạo):

- ✓ Là hàm dùng để tự động khởi tạo giá trị cho đối tượng, khi đối tượng được sinh ra.
- ✓ Kotlin có 2 loại Constructors : **primary constructor** và **secondary constructors**

3.1 primary constructor (Hàm tạo chính) :

Hàm tạo chính là một phần của tiêu đề lớp, nó **đặt sau tên lớp**

```
class SinhVien constructor(ma:Int,ten:String){  
}
```

- **primary constructor chỉ có 1**
- **Nếu muốn thực hiện các logic code thì khai báo trong init {}**

```
class SinhVien constructor(ma:Int,ten:String){  
    init {  
        println("Bạn đang ở Primary constructor")  
        println(" $ma - $ten")  
    }  
}
```



3

constructors

3.1 primary constructor (Hàm tạo chính) :

❑ Tạo 1 đối tượng mới : var tenDoiTuong = TenLop()

```
//1. Tạo 1 đối tượng với Primary constructor  
var sv1 = SinhVien( ma: 1, ten: "Bành Thị Nòi")  
var sv2 = SinhVien( ma: 2, ten: "Sung Thị Sướng")
```



```
Bạn đang ở Primary constructor  
1 - Bành Thị Nòi  
Bạn đang ở Primary constructor  
2 - Sung Thị Sướng
```



3

constructors

3.2 secondary constructors (Hàm tạo phụ) :

- ✓ **secondary constructors** được đặt trong thân class
- ✓ Có thể có nhiều secondary constructors cùng nằm trong 1 classs
- ✓ constructors có thể có hoặc không có các Parameter (các đối số)

```
class SanPham {  
    constructor()  
    {  
        println("Đây là secondary constructor không đối số")  
    }  
    constructor(ma:Int,ten:String,donGia:Int)  
    {  
        println("Đây là secondary constructor 3 đối số")  
        println("$ma - $ten - $donGia")  
    }  
}
```

□ Tạo 1 đối tượng mới : var tenDoiTuong = TenLop()

```
var sp1 = SanPham()  
var sp2 = SanPham( ma: 1, ten: "Tivi", donGia: 15000)
```

```
Đây là secondary constructor không đối số  
Đây là secondary constructor 3 đối số  
1 - Tivi - 15000
```



3

constructors

□4. Biến lớp :

1. *Quy tắc khai báo* giống khai báo biến thông thường, quy tắc camel (*maSV*, *tenSV*, *tenBien*, *canhHuyen*,)

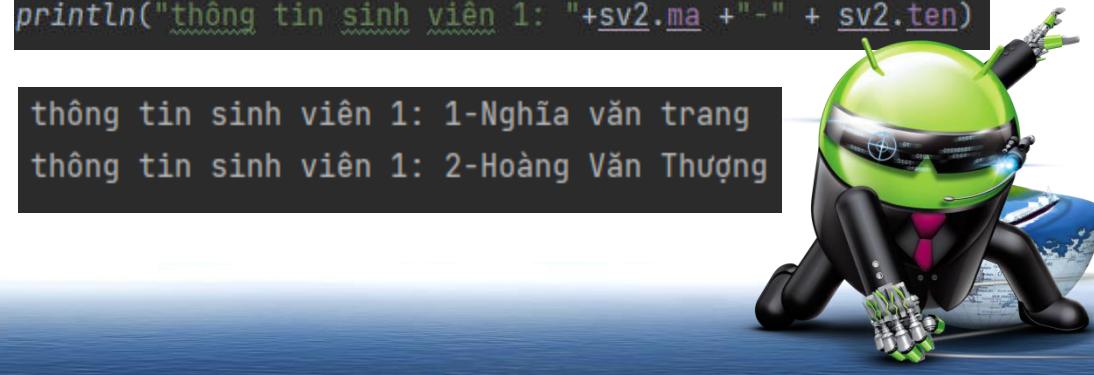
2. *Mức độ truy xuất* : mức bảo vệ đi từ thấp đến cao (1 -> 3)

- ✓ 1. public – hoặc không ghi gì : Truy xuất dc mọi nơi
- ✓ 2. protected: Chỉ truy xuất ở trong class hoặc class kế thừa
- ✓ 3. private : Truy xuất trong class

```
class LopSV2 {
    var ma:Int =0
    var ten:String ="" | Khởi tạo Biến lớp
    //khai báo constructor không đối số
    constructor()
    {
        ma=0
        ten="No name"
    }
    //constructor 2 đối số
    constructor(m:Int,t:String)
    {
        ma=m
        ten=t
    }
}
```

```
//1. Tạo 1 đối tượng mới không truyền giá trị
var sv1 = LopSV2() //khởi tạo đối tượng sv1
sv1.ten="Nghĩa văn trang" // thay đổi tên của sv1
sv1.ma=1 // thay đổi mã sv1
println("thông tin sinh viên 1: "+sv1.ma + "-" + sv1.ten)
//2. Tạo 1 đối tượng mới có truyền vào giá trị ban đầu
var sv2 = LopSV2( m: 2, t: "Hoàng Văn Thượng")
println("thông tin sinh viên 1: "+sv2.ma + "-" + sv2.ten)
```

```
thông tin sinh viên 1: 1-Nghĩa văn trang
thông tin sinh viên 1: 2-Hoàng Văn Thượng
```



3

constructors

□5. Review cấu trúc class với *secondary constructors* :

Tên lớp

```

class LopSV2 {
    var ma:Int =0
    var ten:String =""
    //khai báo constructor không đối số
    constructor()
    {
        ma=0
        ten="No name"
    }
    //constructor 2 đối số
    constructor(m:Int, t:String)
    {
        ma=m
        ten=t
    }
}

```

Khởi tạo Biến lớp

Không có đối số

Giá trị gán mặc định

2 đối số

Giá trị = giá trị truyền vào

```

//1. Tạo 1 đối tượng mới không truyền giá trị
var sv1 = LopSV2() //khởi tạo đối tượng sv1
sv1.ten="Nghĩa văn trang" // thay đổi tên của sv1
sv1.ma=1 // thay đổi mã sv1
println("thông tin sinh viên 1: "+sv1.ma + "-" + sv1.ten)
//2. Tạo 1 đối tượng mới có truyền vào giá trị ban đầu
var sv2 = LopSV2( m: 2, t: "Hoàng Văn Thương")
println("thông tin sinh viên 1: "+sv2.ma + "-" + sv2.ten)

```

thông tin sinh viên 1: 1-Nghĩa văn trang
 thông tin sinh viên 1: 2-Hoàng Văn Thương



3

constructors

□6. Test các mức độ truy xuất :

Mức độ truy xuất : mức bảo vệ đi từ thấp đến cao (1 -> 3)

- ✓ 1. public – hoặc không ghi gì : Truy xuất dc mọi nơi
- ✓ 2. protected: Chỉ truy xuất ở trong class hoặc class kế thừa
- ✓ 3. private : Truy xuất trong nội tại class, không cho truy xuất từ class con

```
class LopSV2 {
    public var ma:Int =0
    var ten:String =""
}
```



```
//1. Tạo 1 đối tượng mới không truyền giá trị
var s
sv1.ten="Nghĩa văn trang" // thay đổi tên của sv1
sv1.ma=1 // thay đổi mã sv1
```

Truy xuất vào ma từ bên ngoài OK

```
class LopSV2 {
    protected var ma:Int =0
    var ten:String =""
}
```



```
sv1.ma=1 // thay đổi mã sv1
print Cannot access 'ma': it is protected in 'LopSV2'
```

Không truy xuất được ma từ bên ngoài

```
class LopSV2 {
    private var ma:Int =0
    var ten:String =""
}
```



```
sv1.ma=1 // thay đổi mã sv1
print Cannot access 'ma': it is private in 'LopSV2'
```

Không truy xuất được ma từ bên ngoài





Kotlin Full Course Bài 24.2

An illustration of a green Android robot. It has a white rectangular body with a blue "K" logo on it. It also has a white head with a blue "K" logo on its forehead. The robot is surrounded by colorful geometric shapes like triangles and squares.



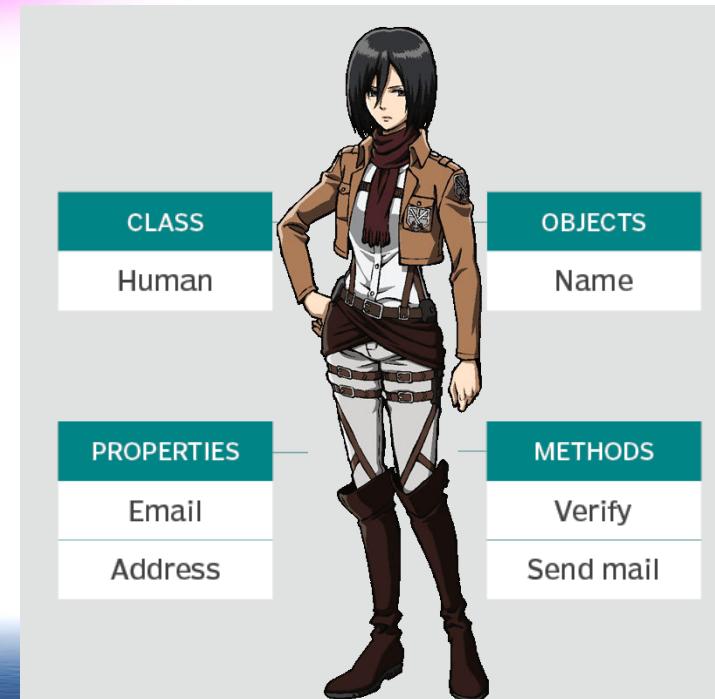
Hướng đối tượng Kotlin Part 2

4

get – set

5

Method (Phương thức)



4

get – set

7. get-set : Xem sửa dữ liệu khi biến đặt private:*Mức độ truy xuất : mức bảo vệ đi từ thấp đến cao (1 -> 3)*

- ✓ 1. public – hoặc không ghi gì : Truy xuất dc mọi nơi
- ✓ 2. protecteted: Chỉ truy xuất ở trong class hoặc class kế thừa
- ✓ 3. private : Truy xuất trong nội tại class, không cho truy xuất từ class con

Trong nhiều trường hợp, chúng ta chỉ muốn 1 nhóm người truy cập được vào, thay đổi các giá trị của biến private => Sử dụng getter và setter

Ví dụ : Tại class LopSV2, ta khai báo thêm diemToan private, chỉ giáo viên dạy toán mới dc sửa điểm.

```
class LopSV2 {
    var ma:Int = 0
    var ten:String =""
    //khai báo điểm toán private
    private var diemToan:Float=0f
```

```
//thử truy cập điểm toán từ bên ngoài
sv2.diemToan
}
Cannot access 'diemToan': it is private in 'LopSV2'
```

Báo lỗi truy xuất

Get, set sẽ giúp xem, sửa dữ liệu từ bên ngoài

Các biến để get, set người ta còn gọi là các **properties**



4

get – set

Khai báo properties:

```

class LopSV2 {
    var ma:Int =0
    var ten:String =""
    //khai báo điểm toán private
    private var diemToan:Float=0f
    //khai báo properties
    var DiemToan:Float
        get() {return diemToan} //xem thông tin biến diemToan
        set(value) {diemToan=value} // set diemToan = value truyền vào

    //khai báo constructor không đối số
    constructor()
    {
        ma=0
        ten="No name"
    }
    //constructor 2 đối số
    constructor(m:Int,t:String)
    {
        ma=m
        ten=t
    }
}

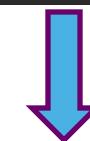
```

Tên Properties viết hoa chữ cái đầu

```

//truy cập sửa điểm toán qua get, set
sv2.diemToan // báo lỗi khi truy cập private diemToan
println("điểm toán sv2 trước sửa: " +sv2.DiemToan)
sv2.DiemToan= 9f
println("điểm toán sv2 sau sửa: " +sv2.DiemToan)

```



điểm toán sv2 trước sửa: 0.0
điểm toán sv2 sau sửa: 9.0



5

Method (Phương thức)

8 . Bản chất phương thức trong lập trình hướng đối tượng là các hàm bên trong lớp (hay nói cách khác nó là các khối lệnh thực hiện 1 công việc hoàn chỉnh)

- Ví dụ :
1. Tính điểm Trung bình môn của Sinh viên
 2. Tính lương cho giảng viên
 3. Xuất thông tin sinh viên ...

9 . Có 2 loại phương thức có return và không có return :

```
//khai báo thêm điểm văn
private var diemVan:Float=0
//khai báo properties
var DiemVan:Float
    get() {return diemVan} //xem thông tin biến
    set(value) {diemVan=value} // set diemToan = value truyền vào
```



5

Method (Phương thức)

```
//1. hàm tính điểm trung bình có return
fun DTB(diemToan:Float,diemVan:Float):Float
{
    return (diemToan+diemVan)/2
}

//2. hàm xuất Thông tin sinh viên: 0 có return
fun XuatThongTin() //
{
    println("tên sinh viên là: $ten")
    println("mã sinh viên là: $ma")
}

//3. hàm đặc biệt toString
override fun toString(): String {
    return "mã sinh viên : $ma, tên sinh viên : $ten"
}
```

```
//thêm điểm văn, tính DTB
sv2.DiemVan=8f
//1. gọi hàm DTB
var kq = sv2.DTB(sv2.DiemToan,sv2.DiemVan)
println("kq = $kq ")
//2. gọi hàm xuất thông tin
sv2.XuatThongTin()

//3. test toString
println(sv2)
```



```
kq = 8.5
tên sinh viên là: Hoàng Văn Thượng
mã sinh viên là: 2
mã sinh viên : 2, tên sinh viên : Hoàng Văn Thượng
```



5

Method (Phương thức)

10 . Có 2 dạng phương thức triển khai trong Class:

* Một lớp có nhiều phương thức, có những phương thức public ra ngoài(public, hay service method)

* Còn những phương thức chỉ sử dụng trong lớp (private, gọi là support method)

dạng 1. Support Method dùng để hỗ trợ bên trong phương thức. Không truy xuất được từ bên ngoài => Dùng từ khóa **private**

dạng 2. Service Method Truy xuất được từ bên ngoài => Dùng từ khóa **public, hoặc không ghi gì**.

```
//support method  
//chỉ sử dụng nội bộ trong class  
private fun CheckTuoi():Boolean  
{  
    var tuoi=2022-namSinh  
    return tuoi>=18  
}
```

```
//service method : Có thể gọi từ c trình chính  
fun KiemTra()  
{  
    if (CheckTuoi())  
        println("Mời bạn đặt vé.. ")  
    else  
        println("Không đủ tuổi xem phim này, vui lòng chọn phim khác")
```



5

Method (Phương thức)

*10. Có 2 dạng phương thức triển khai trong Class:
Service method và Support method*

```

class Tuoi {
    //khai báo biến namSinh private
    private var namSinh:Int=0
    //khai báo properties
    var NamSinh:Int
    get() {return namSinh} //xem tuổi
    set(value) {namSinh=value} //set tuổi = value truyền vào

    //khai báo constructor
    constructor(y:Int)
    {
        namSinh=y
    }
    //support method
    //chỉ sử dụng nội bộ trong class
    private fun CheckTuoi():Boolean
    {
        var tuoi=2022-namSinh
        return tuoi>=18
    }
    //service method : Có thể gọi từ chương trình chính
    fun KiemTra()
    {
        if (CheckTuoi())
            println("Mời bạn đặt vé.. ")
        else
            println("Không đủ tuổi xem phim này, vui lòng chọn phim khác")
    }
}

```

```

fun main(args: Array<String>) {
    var kh1 = Tuoi( y: 2005)
    println("năm sinh kh1 là ${kh1.NamSinh}")
    //check support method
    kh1.CheckTuoi() // báo lỗi
    //check service method
    kh1.KiemTra() // Cannot access 'CheckTuoi': it is private in 'Tuoi'
}

```

```

fun main(args: Array<String>) {
    var kh1 = Tuoi( y: 2005)
    println("năm sinh kh1 là ${kh1.NamSinh}")
    //check support method
    //kh1.CheckTuoi() // báo lỗi
    //check service method
    kh1.KiemTra()
}

```

năm sinh kh1 là 2005

Không đủ tuổi xem phim này, vui lòng chọn phim khác



GET IT ON
Google Play



Kotlin Full Course Bài 24.3



Hướng đối tượng Kotlin Part 3

6

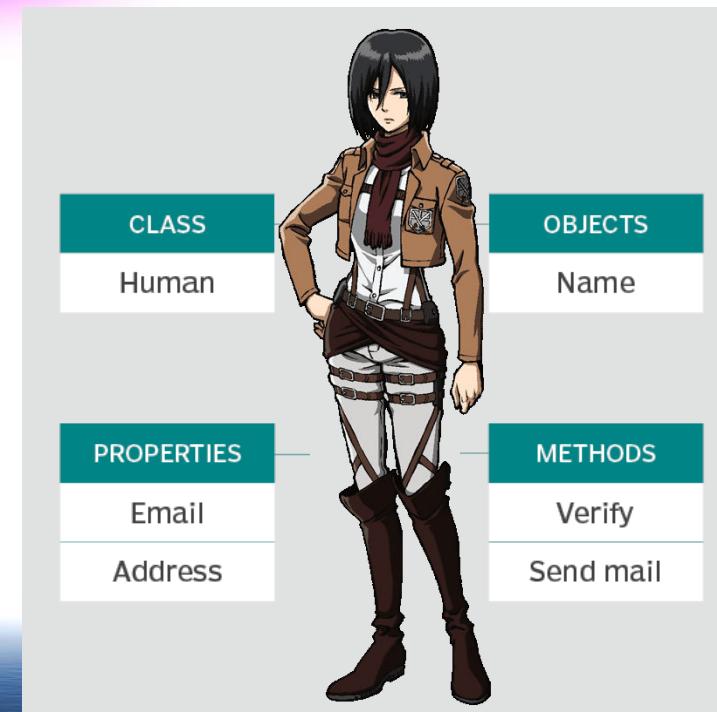
Tham chiếu this

7

Overloading Method

8

Parameter List Method



6

Tham chiếu this

11 . Phân biệt Instance variable và Local variable

```
class ThamChieuThis {
    //Instance variable
    // ( biến có thể truy xuất tại mọi nơi trong class)
    var diemVan:Float = 0f
    var diemToan:Float = 0f
    fun TestCucBo(diemVan:Float,diemToan:Float)
    {
        println("Tổng điểm gọi biến cục bộ= "+(diemVan+diemToan))//ưu tiên biến cục bộ
        println("Tổng điểm gọi Instance variable= " +(this.diemToan+this.diemVan))
            //dùng this để gọi Instance variable
    }
}
```

Instance variable

Local variable

Khi biến Local và biến Instance trùng tên.

Trong hàm muốn sử dụng biến Instance phải sử dụng từ khóa this

```
fun main(args: Array<String>)
{
    //khai báo 1 đối tượng mới
    var sv1 = ThamChieuThis()
    sv1.diemToan=9f
    sv1.diemVan=8f

    println("Điểm toán sv1 = " +sv1.diemToan)
    println("Điểm văn sv1 = " +sv1.diemVan)
    //gọi hàm Test cục bộ
    sv1.TestCucBo( diemVan: 7f, diemToan: 7f)
}
```

```
Điểm toán sv1 = 9.0
Điểm văn sv1 = 8.0
Tổng điểm gọi biến cục bộ= 14.0
Tổng điểm gọi Instance variable= 17.0
```



7

Overloading Method

□12. Overloading Method :

1. *Signature gọi là khác nhau nếu chúng khác nhau về*

1. *Số lượng các đối số*
2. *Kiểu dữ liệu các đối số*
3. *Thứ tự các đối số*

2. *Overloading Method : Trong cùng class có nhiều phương thức cùng tên nhưng khác nhau về Signature*

3. *Constructor : là trường hợp đặc biệt của Overloading Method*



7

Overloading Method

□12. Overloading Method :

```

class Overloading {
    private var maSP:Int =0
    private var tenSP:String =""
    private var giaSP:Double=0.0
    //constructor 3 đối số
    constructor(maSP:Int,tenSP:String,giaSP:Double)
    {
        Signature 3 đối số
        this.maSP=maSP
        this.tenSP=tenSP
        this.giaSP=giaSP
    }
    //constructor 2 đối số (sp tăng kèm mặc định giá 0đ)
    constructor(maSP:Int,tenSP:String)
    {
        Signature 2 đối số
        this.maSP=maSP
        this.tenSP=tenSP
    }
    fun ChietKhau():Double
    {
        Signature 0 đối số
        return this.giaSP*0.95 //giảm 5%
    }
    //nếu khách hàng mua đúng ngày sinh, giảm 10%
    fun ChietKhau(sinhNhat:Boolean):Double
    {
        Signature 1 đối số
        return this.giaSP*0.9 //giảm 10%
    }
}

```

```

//thêm sản phẩm mới
var sp1 =Overloading( maSP: 1, tenSP: "Tivi 45 inch SS")
var sp2 = Overloading( maSP: 2, tenSP: "Galaxy note 20", giaSP: 1200.0)
//tính giá SP
var gia1 = sp2.ChietKhau()
var gia2 = sp2.ChietKhau( sinhNhat: true)
println("Giá áp dụng chiết khấu mặc định cửa hàng: " +gia1)
println("Giá áp dụng sinh nhật khách hàng: " +gia2)

```



Giá áp dụng chiết khấu mặc định cửa hàng: 1140.0
 Giá áp dụng sinh nhật khách hàng: 1080.0



8

Parameter List Method

□13. Parameter List Method : (là 1 trường hợp của Overloading Method)

Trong trường hợp ta không thể nắm được số lượng đối số vào, hoặc số lượng đối số quá lớn . Kotlin cung cấp thêm lựa chọn **Parameter List**

```
class HocParameter {  
    //vararg ( Parameter list)  
    fun TinhTong(vararg dsSo: Int):Int  
{  
    var tong =0  
    for (so in dsSo)  
        tong+=so  
    return tong  
}
```

```
//khởi tạo đối tượng tổng tổng  
var dt1 = HocParameter() Signature  
var kq = dt1.TinhTong( ...dsSo: 1,2,3,4,5,6,8,7,9)  
println(kq)  
var kq2 = dt1.TinhTong( ...dsSo: 1,2,3) Signature  
println(kq2)
```

```
45  
6
```





GET IT ON
Google Play



Kotlin Full Course Bài 24.4



Hướng đối tượng Kotlin Part 4

9

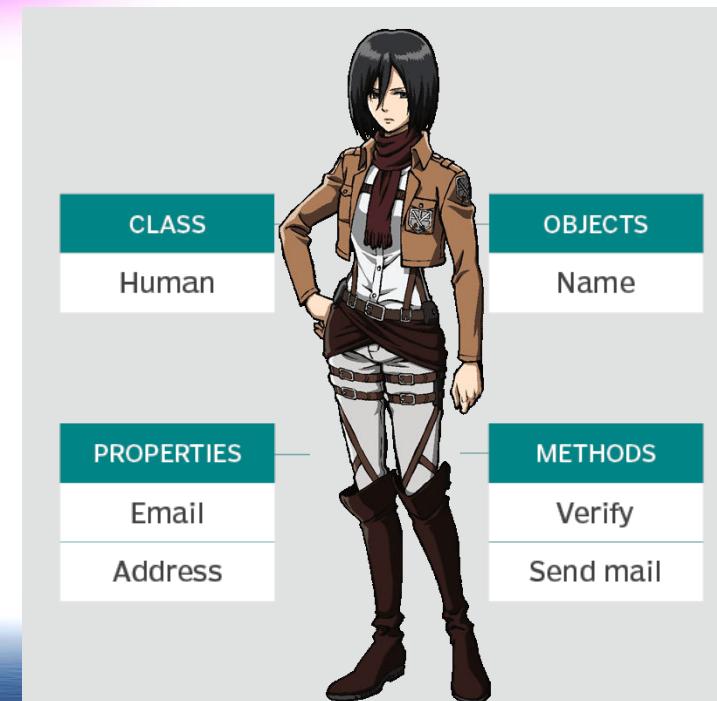
Kế thừa – inheritance

10

Kế thừa từ Interface

11

Overriding method

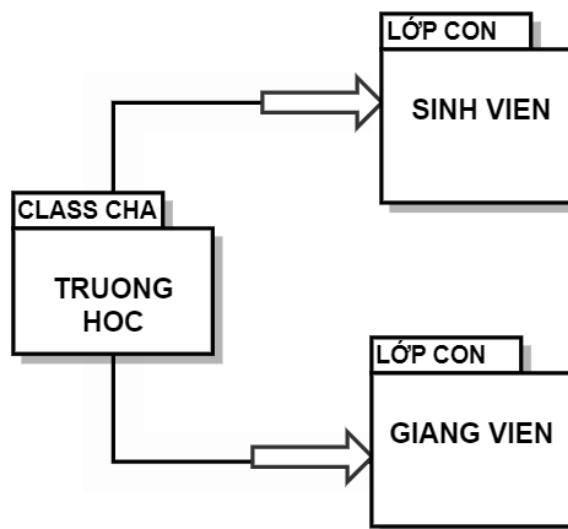


9

Kế thừa – inheritance

14. Khái quát kế thừa:

- ✓ **Kế thừa :** Tạo ra các lớp con, để tái sử dụng lại những thành phần của lớp cha đã có
- ✓ **Ưu điểm :** Giúp code ngắn gọn, không cần phải viết lại những code mà lớp cha đã có => thuận tiện trong quản lý, dễ sửa đổi theo từng khối



Thuộc tính (attribute)			
TÊN	CCCD	NĂM SINH	QUÊ QUÁN
HỌC	Chơi Game	Tính DTB	ĐKý Học Phần

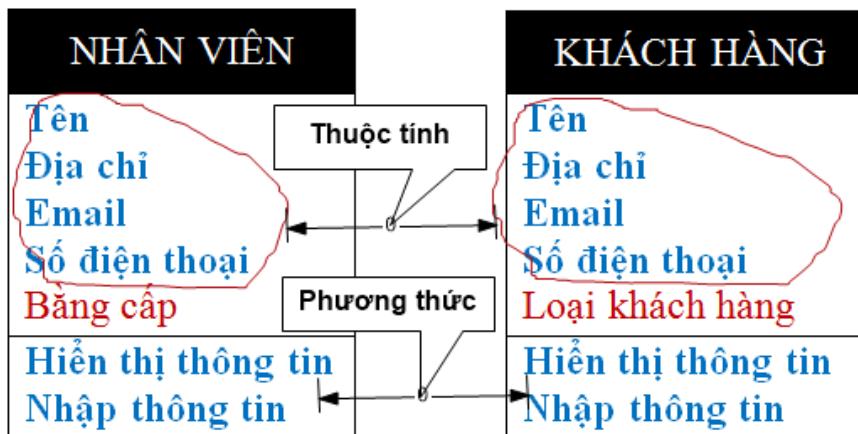
Phương Thức (Method)			
DẠY	XEM TIKTOK	CHẨM ĐIỂM	SỬA ĐIỂM



9

Kế thừa – inheritance

Ví dụ :



NGƯỜI
Tên
Địa chỉ
Email
Số điện thoại
Hiển thị thông tin
Nhập thông tin

NHÂN VIÊN
Băng cấp
Hiển thị thông tin
Nhập thông tin

KHÁCH HÀNG
Loại khách hàng
Hiển thị thông tin
Nhập thông tin



9

Kế thừa – inheritance

- 15. Cú pháp :
- Lớp cha sử dụng open để cho phép kế thừa :

```
open abstract class NhanVien {  
    protected var ten:String=""  
    protected var cccd:String=""  
    protected var que:String=""  
    val luongCoBan:Double=1200.0 //(1200$)  
    //hàm trùu tượng, đn tinh lương, kiểu trả về Double  
    public abstract fun TinhLuong():Double  
    //khai báo constructor 3 đối số  
    constructor(ten:String,cccd:String,que:String)  
    {  
        this.ten=ten  
        this.que=que  
        this.cccd=cccd  
    }  
    constructor(ten:String,cccd:String)  
    {  
        this.ten=ten  
        this.cccd=cccd  
    }  
}
```

Nếu có hàm abstract (Hàm trùu tượng trong thân class Cha).
Thì cần thêm keyword abstract sau open



9

Kế thừa – inheritance

□ **class con:** <Tên lớp con> : <Lớp Cha>

Ex1: **class NhanVien: CongTy**

class NhanVienHanhChinh: NhanVien

class NhanVienDiCa: NhanVien

```

class NhanVienHanhChinh:NhanVien {
    constructor(ten:String,cccd:String,que:String):super(ten, cccd, que)
    constructor(ten:String,cccd:String):super(ten,cccd)
    //khi lớp cha có hàm abstract phải định nghĩa lại ở class con
    override fun TinhLuong(): Double {
        return luongCoBan
    }
}

```

Sử dụng keyword super để gọi các constructor từ class cha

```

class NhanVienDiCa:NhanVien {
    constructor(ten:String,cccd:String,que:String):super(ten, cccd, que)
    constructor(ten:String,cccd:String):super(ten,cccd)
    //khi lớp cha có hàm abstract phải định nghĩa lại ở class con
    override fun TinhLuong(): Double {
        return luongCoBan*1.05 //đi ca cao hơn hành chính 5%
    }
}

```



9

Kế thừa – inheritance

 Test kế thừa trên chương trình chính :

```
fun main(args: Array<String>) {  
    var ca1 = NhanVienDiCa( ten: "linh", cccd: "123456", que: "Đà Nẵng")  
    var hc1 = NhanVienHanhChinh( ten: "Trang", cccd: "123457", que: "Long An")  
    println("Lương nhân viên ca1 = " +ca1.TinhLuong())  
    println("Lương nhân viên hc1 = " +hc1.TinhLuong())  
}
```



```
Lương nhân viên ca1 = 1260.0  
Lương nhân viên hc1 = 1200.0
```

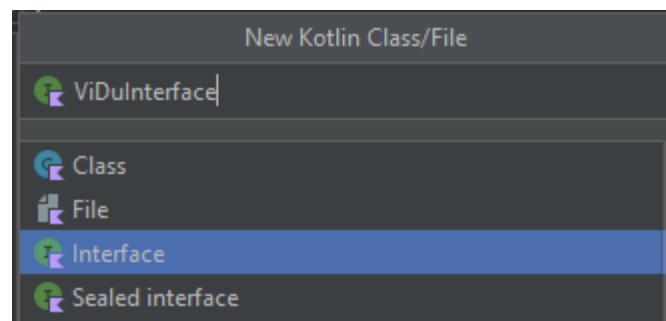


10

Kế thừa từ Interface

❑ 16. *Interface* : có thể hiểu là 1 tập quy luật, ép các class con của nó phải có cấu trúc mà nó đã quy định

❑ Ví dụ : Bắt buộc thông tin các class con phải có : *ten:String,tuoi:Int,cccd:String*



```
interface ViDuInterface {  
    fun ThongTin(ten:String, tuoi:Int, cccd:String)  
}
```



10

Kế thừa từ Interface

☐ *Tạo class con thừa hưởng từ Interface :*

```
class TestInterface:ViDuInterface {  
}  
Class 'TestInterface' is not abstract and does not implement abstract member  
public abstract fun ThongTin(ten: String, tuoi: Int, cccd: String): Unit defined in ViDuInterface  
Make 'TestInterface' 'abstract' Alt+Shift+Enter
```

Đè Alt + Enter để tái định nghĩa



```
class TestInterface:ViDuInterface {  
    override fun ThongTin(ten: String, tuoi: Int, cccd: String) {  
        TODO(reason: "Not yet implemented")  
    }  
}
```



```
class TestInterface:ViDuInterface {  
    override fun ThongTin(ten: String, tuoi: Int, cccd: String) {  
        println("Đã định nghĩa lại hàm Thông tin")  
        println("Tên= " +ten)  
        println("tuoi= " +tuoi)  
        println("cccd= " +cccd)  
    }  
}
```



11

Overriding method

□ 17. Tạo class con thừa hưởng từ Interface :

- ✓ Trong một tập các lớp có mối quan hệ huyết thống có các phương thức giống signature y xì (nội dung phương thức khác nhau)

```
class NhanVienHanhChinh: NhanVien
```

```
class NhanVienDiCa: NhanVien
```

```
class NhanVienHanhChinh: NhanVien {  
    constructor(ten:String, cccd:String, que:String):super(ten, cccd, que)  
    constructor(ten:String, cccd:String):super(ten, cccd)  
    //khi lớp cha có hàm abstract phải định nghĩa lại ở class con  
    override fun TinhLuong(): Double {  
        return luongCoBan  
    }  
}
```

```
class NhanVienDiCa: NhanVien {  
    constructor(ten:String, cccd:String, que:String):super(ten, cccd, que)  
    constructor(ten:String, cccd:String):super(ten, cccd)  
    //khi lớp cha có hàm abstract phải định nghĩa lại ở class con  
    override fun TinhLuong(): Double {  
        return luongCoBan*1.05 //đi ca cao hơn hành chính 5%  
    }  
}
```



11

Overriding method

```
fun main(args: Array<String>) {  
    var ca1 = NhanVienDiCa( ten: "Linh", cccd: "123456", que: "Đà Nẵng")  
    var hc1 = NhanVienHanhChinh( ten: "Trang", cccd: "123457", que: "Long An")  
    println("Lương nhân viên ca1 = " +ca1.TinhLuong())  
    println("Lương nhân viên hc1 = " +hc1.TinhLuong())  
}
```



```
Lương nhân viên ca1 = 1260.0  
Lương nhân viên hc1 = 1200.0
```

- ✓ Cùng 1 hàm TinhLuong nhưng công thức tính (nội dung hàm) khác nhau
- ✓ => Dễ nhớ, dễ quản lý





GET IT ON
Google Play



Kotlin Full Course Bài 24.5



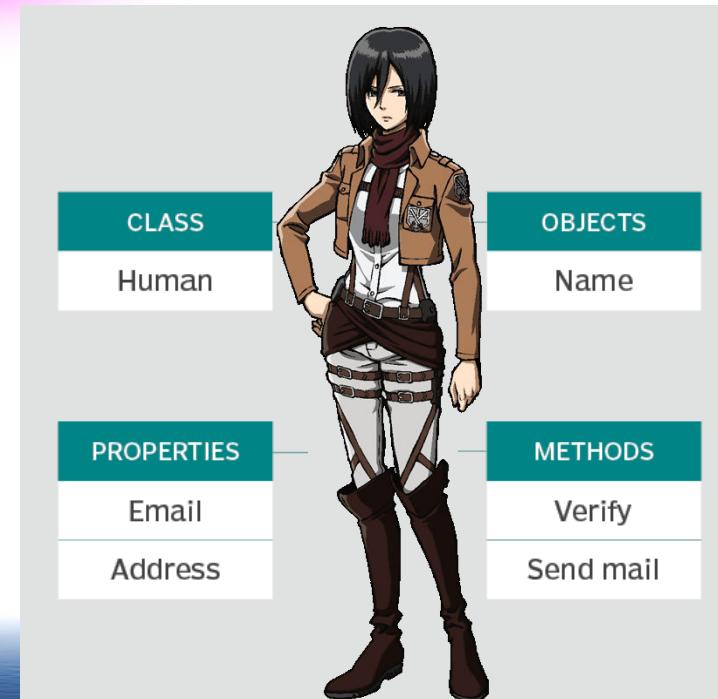
Hướng đối tượng Kotlin Part 5

12

Tính Đa Hình

13

Data Classes



12

Tính Đa Hình

✓ **Đa hình là tại cùng thời điểm đối tượng sẽ có các hình thái khác nhau trong những hoàn cảnh khác nhau.**

❑ **18.1 Đa hình từ kế thừa lớp :** (Ex: Phó Phòng → thăng cấp Trưởng Phòng)

```
open abstract class NhanSu {
    //khai báo hàm trừu tượng TinhLuong
    public abstract fun TinhLuong(ngaycong:Int):Int
}
```

```
class TruongPhong:NhanSu() {
    override fun TinhLuong(ngaycong:Int):Int
    {
        return 50*ngaycong
        //50$ 1 ngày
    }
}
```

```
class PhoPhong:NhanSu() {
    override fun TinhLuong(ngaycong:Int):Int
    {
        return 40*ngaycong
        //40$ 1 ngày
    }
}
```

```
fun main(args: Array<String>) {
    //khai báo đối tượng nv1, kiểu dữ liệu NhanSu
    //nv1 là đối tượng nằm trong class PhoPhong()
    var nv1:NhanSu =PhoPhong() ✅
    println("Lương nv1 khi làm phó phòng: " +nv1.TinhLuong( ngaycong: 23))

    //Nhân viên 1 được thăng cấp lên trưởng phòng
    nv1 =TruongPhong() ✅
    println("Lương nv1 khi làm trưởng phòng: " +nv1.TinhLuong( ngaycong: 23))
```

Tính đa hình

```
Lương nv1 khi làm phó phòng: 920
Lương nv1 khi làm trưởng phòng: 1150
```



12

Tính Đa Hình

□ 18.2 Đa hình từ kế thừa Interface

```
interface TinhToan {
    public abstract fun Tinh(a:Int,b:Int)
}
```

```
class TinhCong:TinhToan {
    override fun Tinh(a: Int, b: Int) {
        println("tổng $a+$b =" + (a+b))
    }
}
```

```
class TinhHieu:TinhToan {
    override fun Tinh(a: Int, b: Int) {
        println("hiệu $a-$b= " +(a-b))
    }
}
```

```
//khai báo đối tượng dt1, kiểu Tính Toán
var dt1:TinhToan = TinhCong()
//gọi phương thức Tinh
dt1.Tinh( a: 7, b: 5) // tính cộng

//chuyển sang đối tượng là Tính hiệu
dt1=TinhHieu()
dt1.Tinh( a: 7, b: 5)
```

Tính đa hình

tổng 7+5 =12
hiệu 7-5= 2



13

Data Classes

- ✓ Trong quá trình xử lý, đôi khi ta chỉ cần lưu trữ dữ liệu mà không dùng đến các phương thức. Kotlin có hỗ trợ tạo một lớp đặc biệt, lớp này gọi là Data Class.
- ✓ Các Data Class trong Kotlin sẽ tự động cung cấp:
 - ❖ `equals()` / `hashCode()`
 - ❖ `toString()`
 - ❖ `componentN()`
 - ❖ `copy()`

```
fun main(args: Array<String>) {  
    data class TestData(var ten:String, var cccd:String)  
    var data1=TestData( ten: "Hứa thị lèo", cccd: "1234567")  
    // test phương thức copy  
    var data2 = data1.copy()  
    println(data2)  
    //copy có sửa đổi dữ liệu  
    var data3 = data1.copy(cccd = "9999")  
    println(data3)  
    //componentN(): để ánh xạ tới thuộc tính của đối tượng  
    println(data3.component1())  
    println(data3.component2())  
}
```





Kotlin Full Course Bài 24.6

Hướng đối tượng Kotlin Part 6

14

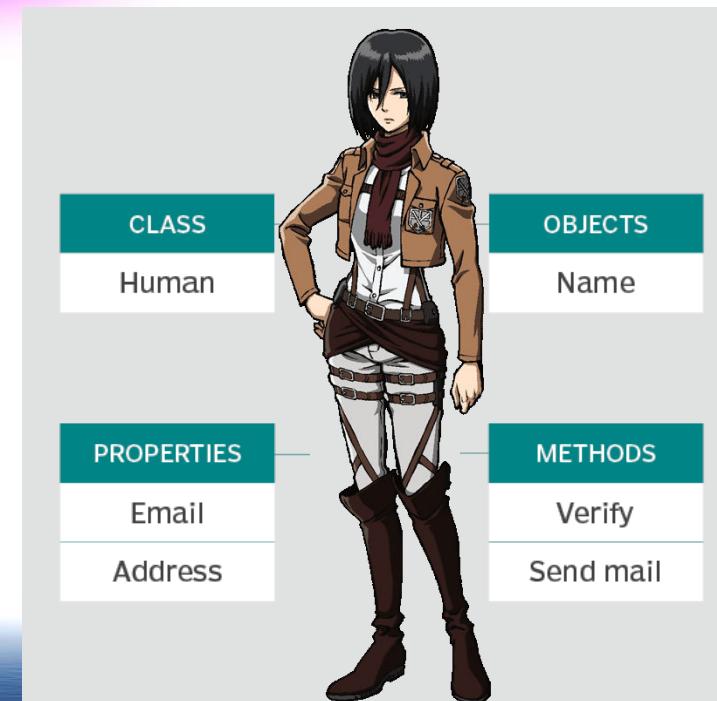
Nested Classes

15

inner Classes

16

enum Classes



14

Nested Classes

Nested Classes: là khả năng cho phép Lớp nằm trong lớp khác.

Lưu ý: các lớp Nested sẽ **không** truy suất được các thông tin ở ngoài lớp chứa nó

```
class Ong {  
    var a:Int=1  
    class Bo{  
        fun Ham()  
        {  
            //println(a)//báo lỗi 0 truy xuất được  
            println("Đây là hàm của class Bo")  
        }  
    }  
    //hàm của class Ong  
    fun Ham()  
    {  
        println(a)//truy xuất được  
    }  
}
```

```
fun main(args: Array<String>) {  
    Ong.Bo().Ham() //gọi hàm trong class Bo()  
    Ong().Ham() // gọi hàm trong class Ong  
}
```



Đây là hàm của class Bo
1



15

inner Classes

inner Classes: hoạt động giống Nested Classes, nhưng thêm keyword **inner**
Lưu ý: các lớp con truy suất được các thông tin ở ngoài lớp chứa nó

```
class Ong {  
    var a:Int=1  
    inner class Bo{  
        fun Ham()  
        {  
            println("a class Bo = "+ a)// truy xuất đc biến a  
            println("Đây là hàm của class Bo")  
        }  
    }  
    //hàm của class Ong  
    fun Ham()  
    {  
        println(a)//truy xuất được  
    }  
}
```

```
fun main(args: Array<String>) {  
    Ong().Bo().Ham() //gọi hàm trong class Bo()  
    Ong().Ham() // gọi hàm trong class Ong  
}
```



```
a class Bo = 1  
Đây là hàm của class Bo  
1
```



16

enum Classes

❑ **enum Classes:** *kiểu dữ liệu liệt kê ,dùng để lưu trữ tập các giá trị là hằng số không đổi
Các giá trị phân cách nhau bởi dấu phẩy*

❑ *Cách khai báo:*

```
enum class BangPhai {  
    NgaMi,  
    Conlon,  
    ThieuLam,  
    CaiBang}
```

❑ *Kiểm tra thứ tự giá trị enum :*

```
fun main(args: Array<String>) {  
    //thứ tự các giá trị  
    println(BangPhai.NgaMi.ordinal) // 0  
    println(BangPhai.Conlon.ordinal) // 1  
}
```



16

enum Classes

□ Thuộc tính mặc định **name** và **ToString()**:

```
// thuộc tính mặc định name  
println(BangPhai.NgaMi.name)      //NgaMi  
//Trong enum cũng có ToString() mặc định nên  
println(BangPhai.NgaMi)            //NgaMi
```

□ **values()**

```
//Values(): Trả về mảng danh sách các pt của enum  
var ds = BangPhai.values()  
//in ds mảng  
ds.forEach { println(it) }
```



NgaMi
Conlon
ThieuLam
CaiBang



16

enum Classes

□ Sử dụng với câu lệnh **when**

```
//khởi tạo đối tượng thuộc BangPhai
var a:BangPhai=BangPhai.NgaMi
when(a)
{
    <ul style="list-style-type: none; padding-left: 0;">
        <li>💡 Add else branch >
        <li>💡 Add remaining branches >
        <li>💡 Add remaining branches with * import >
        <li>📝 Inspection 'Control flow with empty body' options >
        <li>📝 Add remaining branches >
        <li>📝 Inline 'when' argument >
    </ul>
    Press Ctrl+Q to open preview
}
```

Alt + Enter

```
//khởi tạo đối tượng thuộc BangPhai
var a:BangPhai=BangPhai.NgaMi
when(a)
{
    BangPhai.NgaMi -> println("Bạn thuộc phái NgaMi")
    BangPhai.Conlon -> println("Bạn thuộc phái Conlon")
    BangPhai.ThieuLam -> println("Bạn thuộc phái ThieuLam")
    BangPhai.CaiBang -> println("Bạn thuộc phái CaiBang")
}
```

Bạn thuộc phái CaiBang



16

enum Classes

□ Thêm các properties (thuộc tính) cho enum

```
enum class BangPhai(val gioiTinh:String) {
    NgaMi(gioiTinh: "Nữ"),           //ordinal =0   name="NgaMi"
    Conlon(gioiTinh: "Nam"),         //ordinal =1   name="Conlon"
    ThieuLam(gioiTinh: "Nam"),       //ordinal =2   name="ThieuLam"
    CaiBang(gioiTinh: "Nam")         //ordinal =3   name="CaiBang"
}
```

```
//thêm các properties(thuộc tính) vào enum
println(BangPhai.NgaMi.gioiTinh)
println(BangPhai.Conlon.gioiTinh)
```

Nữ
Nam

```
enum class BangPhai(val gioiTinh:String, val sucManh:Int) {
    NgaMi(gioiTinh: "Nữ", sucManh: 1200),
    Conlon(gioiTinh: "Nam", sucManh: 1300),
    ThieuLam(gioiTinh: "Nam", sucManh: 1400),
    CaiBang(gioiTinh: "Nam", sucManh: 1500)
}
```

```
println(BangPhai.NgaMi.sucManh)
println(BangPhai.Conlon.sucManh)
```

1200
1300





GET IT ON
Google Play



Kotlin Full Course

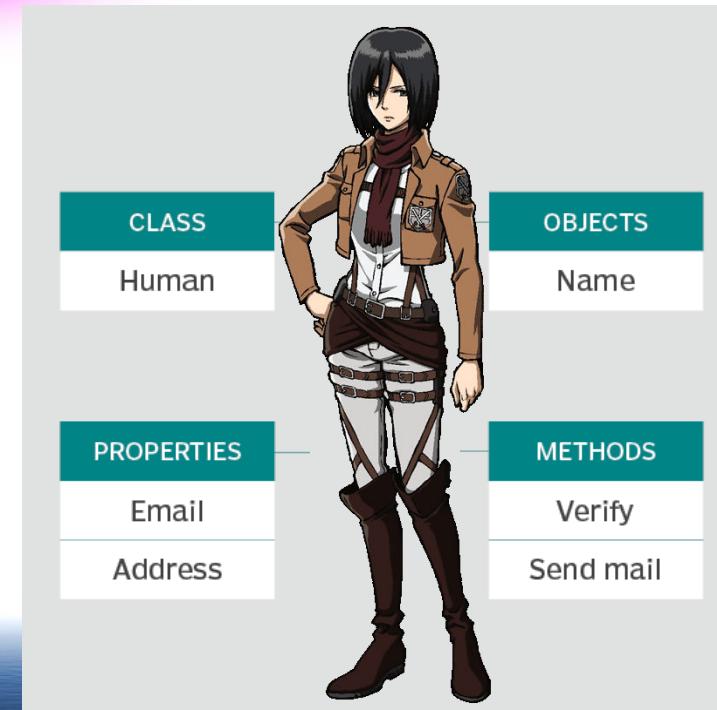
Bài 24.7



Hướng đối tượng Kotlin Part 7

17

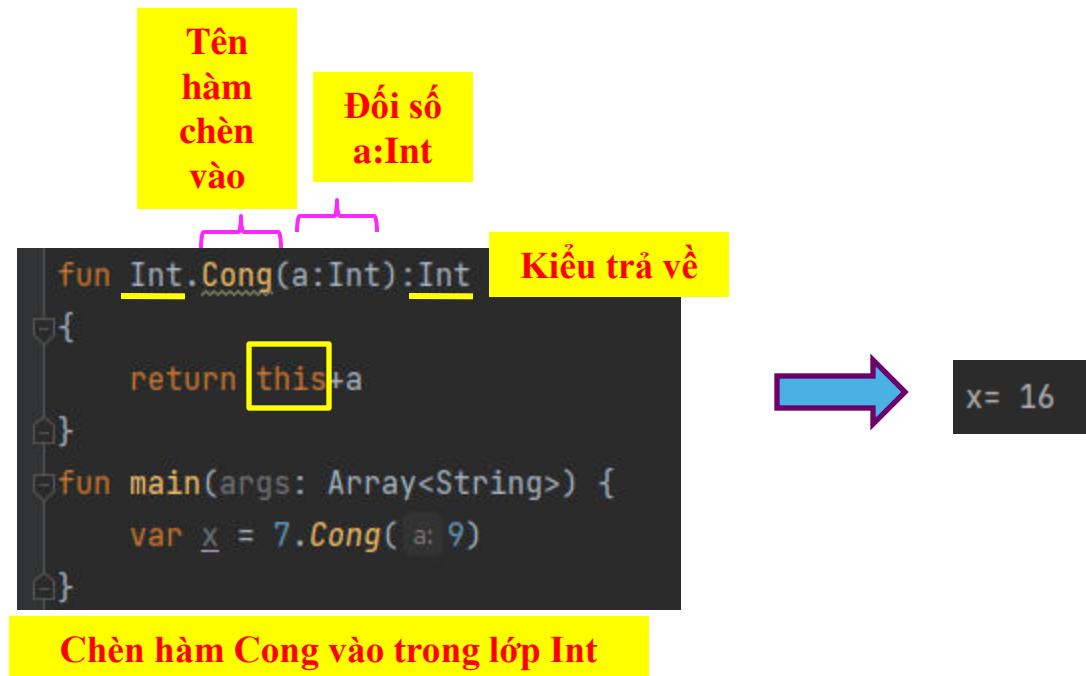
Extensions Method



17

Extensions Method

- ❑ **Extensions Method** cho phép ta chèn thêm phương thức vào các Lớp có sẵn mà không cần phải chỉnh sửa Lớp, không cần kế thừa....



17

Extensions Method

- Ví dụ 2: Cài hàm **Kiểm Tra 1 số có phải là nguyên tố hay không vào Class Int**
Số nguyên tố là số chỉ có 2 ước, là 1 và chính nó. Ví dụ : 3,5,7

```
//hàm kiểm tra số nguyên tố
fun Int.CheckNguyenTo():Boolean
{
    var dem=0
    for (i in 1 .. this)
    {
        if (this%i==0)
            dem++
    }
    return dem==2
}
```

```
fun main(args: Array<String>) {
    //check số nguyên tố
    var kq= 3.CheckNguyenTo()
    println(kq)
}
```

true

Sử dụng debug để hiểu cách hoạt động ☺



17

Extensions Method

□ Ví dụ 3: Cài hàm **Tính tuổi** vào class *Sinhvien* sẵn có

```
import java.util.*  
  
class SV {  
    private var namSinh:Date?  
    public var NamSinh:Date?  
    get() {return namSinh}  
    set(value) {namSinh=value}  
    constructor(namSinh: Date?)  
    {  
        this.namSinh=namSinh  
    }  
}
```

```
//hàm tính tuổi :  
fun SV.Tuoi():Int{  
    //khởi tạo đối tượng ngày tháng (bài 17)  
    var cal = Calendar.getInstance()  
    var namHienTai = cal.get(Calendar.YEAR)  
    //set năm sinh gán vào biến time  
    cal.time = this.NamSinh  
    var yearNamSinh = cal.get(Calendar.YEAR)  
    return namHienTai-yearNamSinh+1  
}
```

Tự học ☺

```
//test tính tuổi  
var ns = Calendar.getInstance()  
//set năm sinh  
ns.set(Calendar.YEAR,1995)  
var Trang = SV(ns.time)  
var tuoiTrang = Trang.Tuoi()  
println(tuoiTrang)
```

