B0-1027,

B0-1527

b0-902

B0-1494

B0-1451

***http://online.swiftplayground.  
https://www.onlinegdb.com/  
https://www.jdoodle.com/***

https://www.geeksforgeeks.org/swift-integer-floating-point-numbers/

***print("Integer Type Min Max")***

***print("UInt8 \(UInt8.min) \(UInt8.max)")***

***print("UInt16 \(UInt16.min) \(UInt16.max)")***

***print("UInt32 \(UInt32.min) \(UInt32.max)")***

***print("UInt64 \(UInt64.min) \(UInt64.max)")***

***print("Int8 \(Int8.min) \(Int8.max)")***

***print("Int16 \(Int16.min) \(Int16.max)")***

***print("Int32 \(Int32.min) \(Int32.max)")***

***print("Int64 \(Int64.min) \(Int64.max)")***

OUTPUT

***Integer Type Min Max***

***UInt8 0 255***

***UInt16 0 65535***

***UInt32 0 4294967295***

***UInt64 0 18446744073709551615***

***Int8 -128 127***

***Int16 -32768 32767***

***Int32 -2147483648 2147483647***

***Int64 -9223372036854775808 9223372036854775807***

**let pi = 3 + 0.141592654**

**print(type(of: pi))**

\_\_\_\_\_\_\_\_\_\_\_OUTPUT\_\_\_\_\_\_\_\_\_\_\_

Double

Excercise 9 :

Uint can contains 0 to 255 then -17 is out of range

Excercise 10:

overflow because Int16 contains values ***-32768 to 32767***

Excercise 11:

let pi = 3.141592654 pi has a double type

and approximatePi is a integer, so they don’t have the same type and we cannot assign

convert a Double type to Integer type

let approximativePi = Int(pi)

Excercise 12 :

// this is comment one line

/\* this is the comment on

multi lines

\*/

Excercise 13 :

/\* this is the comment on multi lines

/\* this is the nested comment

line

\*/

Others comments is here

\*/

Excercise 14 :

/\* this is the comment on multi lines, this is the nested comment ine, Others comments is here \*/

================================ASSIGNMENT 2----------------------------

Excercise 15:

**let teamName : String = "Larionov"**

**let teamNumber: Int = 8**

**var team = (teamName, teamNumber)**

Excercise 16:

**let name:String = team.0**

**let number: Int = team.1**

Is nil NULL in Swift?

***In Swift: nil is not a pointer, it's the absence of a value of a certain type. NULL and nil are equal to each other, but nil is an object value while NULL is a generic pointer value ((void\*)0, to be specific). [NSNull null] is an object that's meant to stand in for nil in situations where nil isn't allowed***

***In Swift: nil is not a pointer, it's the absence of a value of a certain type. NULL and nil are equal to each other, but nil is an object value while NULL is a generic pointer value ((void\*)0, to be specific). [NSNull null] is an object that's meant to stand in for nil in situations where nil isn't allowed***

***As a programmer, sometimes you will need to define “nothingness”…*** 🧐

***Data stores can have a value, or simply be nothing, this nothing comes in different flavors (nil, Nil, Null, NSNull) which all came to be called “null”.***

Exercise 17 :

let value: Int? = nil

print(value)

let otherValue: Int? = 6

Excercise 18:

**let value: Int? = 17**

**let banana: Int = value**

**it doesn’t wok because Int? Is optional type which is different to Int type**

Excercise 19:

**let value: Int? = nil**

**let banana: Int = value!**

**Error message :**

**Fatal error: Unexpectedly found nil while unwrapping an Optional value**

**because, not possible to unwrap a nil.( only possible for a value not nil)**

Excercise 20:

The better way to assign value to the banana constant is to declare banana such as a optional constant

Let banana: Int? = value

**Excercise 21 :**

**var** val1 = 15

**var** val2 = 15

**var** sum = val1 + val2

**func** sumInt(v1 : Int, v2 : Int)->Int{

**if** val1 == val2 {

**return** sum\*3

}

**else**{

**return** sum

}

}

**var** res = sumInt(v1:val1, v2:val2)

print("Result=",res)

**Exercise 22:**

**var** Arr:[Int] = [1,3,7,5]

**if** ((Arr[0] == 5) || (Arr[Arr.count-1] == 5)) {

print("First or Last Element is equal 5")

}

**else** {

print("First or Last Element is not equal 5")

}

Exercise 23 :

**var** Arr = [1, 3, 7, 5]

**var** Arr2:[Int] = Arr.reversed()

print("Arr2=",Arr2)

Exercise 24:

**var** RotArr=[1,2,3]

print(RotArr)

**var** first = RotArr[0]

**for** i **in** 0..<RotArr.count-1{

RotArr[i] = RotArr[i+1]

}

RotArr[RotArr.count-1] = first

print(RotArr)

Exercise 25:

**var** ArrSum = [5,10,15,20]

//var someInts = [Int]()

**var** sum:Int = 0

**for** i **in** ArrSum{

sum += i

}

print(sum)

**Exercise 26:**

**var** n: Int = 52

**let** nb: Int = 51

**if** n > nb {

print("Double of Absolute difference =",abs(n-nb)\*2)

}

**else**{

print("Absolute difference =",abs(n-nb))

}

**Ercercise 27:**

I don’t know how to input the value by console

**Ercercise 28:**

I don’t know how to input the value by console

**Ercercise 29:**

**var** intA = 30

**var** intB = 1

**var** rangNb = 1...30

print(rangNb.contains(intA) && rangNb.contains(intB))

**Exercise 30:**

**There are two way :**

**First way:**

**var** MyString = "abcde"

**var** subString = MyString.prefix(4)

**var** firstChar = MyString.prefix(1)

**var** lastChar = MyString.suffix(1)

print(lastChar + subString.suffix(3) + firstChar)

**Second way :**

**var** MyString = "abcde"

**let** start = MyString.index(MyString.startIndex, offsetBy: 1)//

**let** end = MyString.index(MyString.startIndex, offsetBy: 3)

**var** NewString = String(MyString[start...end])

print(NewString)

print(MyString.suffix(1) + NewString + MyString.prefix(1))