Swift "Hello, World!" Program

```
// Swift "Hello, World!" Program

print("Hello, World!")
```

Output

```
Hello, World!
```

Swift Comment

```
// Swift "Hello, World!" Program
```

We use two slashes // to write comments in Swift.

Print Statement

```
print("Hello, World!")
```

The code above displays the text Hello, World! on your screen.

Multiline Comment

In Swift, any text between /* and */ is a multiline comment. For example,

```
/* create a variable
to store salary of employees
*/

var salary = 10000
print(salary)
```

In the above example, we have used /*...*/ to write the comment that extends for multiple lines.

Declare Variables in Swift

In Swift, we use the var keyword to declare variables. For example,

```
var siteName:String
var id: Int
```

Here,

- siteName is a variable of type String. Meaning, it can only store textual values.
- id is a variable of Int type. Meaning, it can only store integer values.

Swift Data Types

Data Types	Example	Description
Character	"s","a"	a 16-bit Unicode character
String	"hello world!"	represents textual data
Int	3, -23	an integer number
Float	2.4, 3.14, -23.21	represents 32-bit floating-point number
Double	2.422342412414	represents 64-bit floating-point number
Bool	true and false	Any of two values: true or false

Swift Character

Character is a data type that represents a single-character string ("a", "@", "5", etc).
We use the Character keyword to create character-type variables in Swift. For example,

```
var letter: Character
```

Here, the letter variable can only store single-character data.

Swift String

In Swift, a string is used to store textual data ("Hey There!", "Swift is awesome.", etc).
We use the String keyword to create string-type variables. For example,

```
let name: String
```

Here, the name variable can only store textual data.

Swift Output

In Swift, we can simply use the print() function to print output. For example,

```
print("Swift is powerful")
// Output: Swift is powerful
```

Swift Basic Input

In Swift, we cannot directly take input from the Xcode playground.
However, we can create a Command line Tool in Xcode and use the readLine() function to take input from users.

For example,

```
print("Enter your favorite programming language:")
let name = readLine()

print("Your favorite programming language is \(name!).")
```

Output

```
Enter your favorite programming language:
Swift
Your favorite programming language is Swift.
```

Swift if..else if Statement

```
// check whether a number is positive, negative, or 0.

let number = 0

if (number > 0) {
    print("Number is positive.")
}

else if (number < 0) {
    print("Number is negative")
}

else {
```

```
    print("Number is 0.")
}

print("This statement is always executed")
```

Output

```
Number is 0.
```

Simple program using Switch Statement

```
// program to find the day of the week

let dayOfWeek = 4

switch dayOfWeek {
  case 1:
    print("Sunday")

  case 2:
    print("Monday")

  case 3:
    print("Tuesday")

  case 4:
    print("Wednesday")

  case 5:
    print("Thursday")

  case 6:
    print("Friday")

  case 7:
    print("Saturday")

  default:
    print("Invalid day")
}
```

Output

```
Wednesday
```

Swift for-in Loop

In Swift, the for-in loop is used to run a block of code for a certain number of times. It is used to iterate over any sequences such as an array, range, string, etc.

The syntax of the for-in loop is:

```
for val in sequence{
  // statements
}
```

Here, val accesses each item of sequence on each iteration. Loop continues until we reach the last item in the sequence.

Swift while Loop

Swift while loop is used to run a specific code until a certain condition is met.

The syntax of while loop is:

```
while (condition){
  // body of loop
}
```

repeat...while Loop

The repeat...while loop is similar to while loop with one key difference. The body of repeat...while loop is executed once before the test expression is checked.

The syntax of repeat..while loop is:

```
repeat {
  // body of loop
} while (condition
```

Syntax of guard Statement

The syntax of the guard statement is:

```
guard expression else {
  // statements
  // control statement: return, break, continue or throw.
}
```

Here, expression returns either true or false. If the expression evaluates to

- true - statements inside the code block of guard are not executed
- false - statements inside the code block of guard are executed

Create a Swift Array

Here's how we create an array in Swift.

```
// an array of integer type
var numbers : [Int] = [2, 4, 6, 8]

print("Array: \(numbers)")
```

Output

```
Array: [2, 4, 6, 8]
```

Swift Function Declaration

The syntax to declare a function is:

```
func functionName(parameters)-> returnType {
  // function body
}
```

Here,

- func - keyword used to declare a function
- functionName - any name given to the function
- parameters - any value passed to function
- returnType - specifies the type of value returned by the function

Let's see an example,

```
func greet() {
print("Hello World!")
}
```