

Simple Values

let for constants; var for variables

Constants: value doesn't need to be known at compile time, but must assign it a value exactly once

Rule for types: constant or variable must have the same type as the value you want to assign it.

Inferring Types: Type can be inferred if value provided when creating constant/var.

Specifying types: Write after the variable, separated by colon

```
let explicitDouble: Double = 70
```

Values are never implicitly converted to another type. If conversion required, must be done explicitly

```
let lottoString = "Today's winning lotto number is "
let lottoNumber = 94
let lottoSentence = lottoString + String(lottoNumber)
```

Backslash string interpolation

```
let friendCount = 2
let friendCountString = "I have \(friendCount) number of friends."
```

Use three double quotation marks for strings that take up multiple lines.

Create arrays and dictionaries using brackets, and access their elements by writing the index or key in brackets.

```
var toys = ["lamborghini", "simba", "furby"]
toys[1] = "fugle r"
toys.append("jellycat")
var bestItem = ["summer fruit": "mango", "winter fruit": "peach"]
bestItem["summer fruit"] = "peach"
```

Arrays grow automatically.

For empty array, write `emptyArray = []`. For an empty dictionary, write `emptyDictionary = [:]`.

If assigning empty array or dict to new var, you need to specify type.

```
let arrayOfGoodCoffee: [String] = []
let dictionaryWithMachines: [String: Float] = [:]
```

Control Flow (cont)

```
let score = 10
if teamScore > 10 {
    // do something
} else {
    // do something else
}

print("Score: ", teamScore, score)
```

Optionals: either contains a value or contains nil to indicate a value is optional. A question mark after type of value marks it as optional.

```
var optionalString: String? = "Hello"
print(optionalString == nil)
// prints "false"
```

Use if and let to work with missing values. If optional value is nil, conditional is false and code in braces is skipped. Otherwise, optional value is unwrapped and assigned to constant after let, which makes the unwrapped value available inside block of code.

```
var optionalName: String? = "Big Dog"
var greeting = "Hello!"

if let name = optionalName {
    greeting = "Hello, \(name)"
} else {
    greeting = "Wow!"
}
```

Coercion operator: If optional value missing, default value used instead

```
let holidayInNov: String? = nil
let holidayInDec: String = "Bali"
let welcomeMsg = "Enjoy \(holidayInNov ?? holidayInDec)"
```

Switches: support any kind of data and variety of comparisons

After executing the code inside the switch case that matched, program continues from the switch statement. Execution doesn't continue to the next case, so you don't need to explicitly break out of the switch at the end of each case's code.

```
let perfume = "Eau Rose"
switch perfume {
case "Electric Cherry":
    print("Jasmine sambac, ambrette musk")
case "On a Date", "Born In Roma":
    print("These have a black currant note.")
case let x where x.hasSuffix("rose"):
    print("Is it a fragrance with \(x)?")
default:
    print("A delicious choice.")
}

// Prints "Is it a fragrance with rose?"
```

for-in: Use to iterate over items in a dict by providing pair of names for key-value pair.

Control Flow

Conditionals: if, switch

Loops: for-in, while, repeat - while

Parentheses around condition/loop variable are optional, braces are required

```
let croissant Scores = [1.2, 2.3, 2.2, 4]
for score in croissant Scores {
  if score > 2.5 {
    print( " Great croissant !!")
  }
}
```

In if statement, conditional must be a boolean expression. if score { ... } is an error, not implicit comparison to zero.

You can assign if conditions to an assignment

```
let interestingNumbers = [
  " country codes from my holidays": [66, 1, 86, 8],
  " cricket players": [49, 56, 26, 30],
]

var largest = 0
for (_, numbers) in interestingNumbers {
  for number in numbers {
    if number > largest {
      largest = number
    }
  }
}
```

while: condition can be at end too, to ensure loop runs at least once

```
var croissant sLeft = 2
while croissant sLeft < 100 {
  croissant sLeft *= 2
}

print( croissant sLeft)
// prints " 128 "
```

```
var cakesLeft = 2
repeat {
  cakesLeft *= 2
} while cakesLeft < 0
print( cakesLeft)
```



By nixik09

cheatography.com/nixik09/

Not published yet.

Last updated 17th December, 2025.

Page 1 of 3.

Sponsored by **CrosswordCheats.com**

Learn to solve cryptic crosswords!

<http://crosswordcheats.com>

Control Flow (cont)

Keep an index in a loop: Use ... for a range that includes both values and ...< for range that omits upper val

```
var total = 0
for i in 0...4 {
    total += i
}
print( total)
// Prints " 10"
```

Functions and Closures

functions: Use `func` to declare a function. Call a function by following its name with a list of arguments in parentheses. Use `->` to separate the param names and types from the function's return type

```
func countr yGr eet ing (gr eeting: String, country: String) -> Stri
ng {
    return " \ (g ree ting), welcome to \ (coun try )."
}
greet( gre eting: " Nam ast e", country: " Nep al")
```



By nixik09

cheatography.com/nixik09/

Not published yet.

Last updated 17th December, 2025.

Page 2 of 3.

Sponsored by **CrosswordCheats.com**

Learn to solve cryptic crosswords!

<http://crosswordcheats.com>