II / Complex data types

---

1 / Arrays

Arrays are collections of values that are stored as a single value

Exemple:

let john = "John Lennon"
let paul = "Paul McCartney"
let george = "George Harrison"
let ringo = "Ringo Starr"

let beatles = [john, paul, george, ringo]

Total score: 6/6 checked

---

2 / Sets

Sets are collections of values just like arrays, except:

- Items aren’t stored in any order; they are stored in what is effectively a random order.
- No item can appear twice in a set; all items must be unique.

You need it for times when you want to check whether a word appears in a dictionary
Fixed collection of related values where each item has a precise position or name

Exemple:

let colors = Set(["red", "green", "blue"])

let set = Set(["aardvark", "astronaut", "azalea"])

NB : insert a duplicate item into a set, the duplicates get ignored

Total score: 12/12 checked

---

3 / Tuples

To store several values together in a single value or ether need a specific, fixed collection of related values where each item has a precise position or name, you should use a tuple:

1. You can’t add or remove items from a tuple; they are fixed in size.
2. You can’t change the type of items in a tuple; they always have the same types they were created with.
3. You can access items in a tuple using numerical positions or by naming them, but Swift won’t let you read numbers or names that don’t exist.

Exemples:

var name = (first: "Taylor", last: "Swift")

var website = (name: "Apple", url: "www.apple.com")

let address = (house: 555, street: "Taylor Swift Avenue", city: "Nashville")

Access items using numerical positions starting from 0

Ex :

name.0

Access items using their names

Ex :

name.first

Total score: 6/6 checked

———————————————————————————————————————

4 / a) Dictionaries

Collections of values just like arrays but the différent is that you can access them using anything

Rather than trying to remember that array index 7 means a user's country, we could just write user["country"]

Exemple : **[String: Double]** or **[String: String]** or **[String: Bool]** or **[Int: String]**

```
let heights = [
    "Taylor Swift": 1.78,
    "Ed Sheeran": 1.73
]
```

These identifiers are called *keys* **use them to read data back out of the dictionary**

Exemple :

heights["Taylor Swift"]

Total score: 6/6 checked

———————————————————————

b) Dictionaries default values

try to read a value from a dictionary using a key that doesn't exist, Swift will send you back nil

To avoid that fix this by giving the dictionary a default value of “Unknown”

Exemples :

```
let favoriteIceCream = [
    "Paul": "Chocolate",
    "Sophie": "Vanilla"
]

=> favoriteIceCream["Charlotte", default: "Unknown"]


let historyResult = results["history", default: 0]
```

Total score: 12/12 checked

———

5 / Creating empty collections

Arrays, sets, and dictionaries are called *collections*

1. create an empty dictionary with strings :

```
var teams = [String: String]()

teams["Paul"] = "Red"
```

2. create an empty array to store integers :

```
var results = [Int]()
```

3. create an empty set :

```
var words = Set<String>()
var numbers = Set<Int>()
```

4. Create arrays and dictionaries with similar syntax

    var scores = Dictionary<String, Int>()
    var results = Array<Int>()

Total score: 6/6 checked