Complex Data Types

- Arrays in Swift work the same way you would expect them to in other programming languages.
- Sets are collections of objects that work similarly to how arrays do, save for two key differences:
  - objects are not ordered
  - no duplicate objects allowed, duplicate etries to be ignored.
- I can see sets being more useful for defining the parameters of a dataset as opposed to storing actual data.
- Tuples allow you to store multiple values attached to a single variable, similar to an object. Values are ordered, and you may also attach identifiers to the values.
  - `var name = (first: "Taylor", last: "Swift")`
  - `name.0` // Taylor
  - `name.first` // also Taylor
- The data types of each var are set after the tuple has been created.
- Dictionaries are collections of values,

Similar to arrays, but values may be accessed via any identifier necessary.

- let ages = [
  "harrison" = 21
  "liam" = 20
  "jess" = 20
  ]

- These identifiers are called keys, and may be used to retrieve data.
- You can set a default value to be returned when a dictionary address does not exist:
  - dictionary[default: "value"]
- You can set an empty complex data type by setting it equal to the preferred data type in brackets followed by empty parenthesis.
  - Var teams = [String: String]()
  - teams["Paul"] = "Red"
  - Var scores = Dictionary<String, Int>()
  - var results = Array<Int>()
- Enumeration can be used to define a variable with a select number of unique states:
  - enum number {
    case positive

case zero

case negative

}