```
# ---------- COMPARISON OPERATOR ----------

== # complete equality, including type
!= # complete inequality, including type
< # comparison operator
> # comparison operator
<= # comparison operator
>= # comparison operator

# ---------- LOGICAL OPERATOR ----------

&& # and
|| # or
! # not
```

Data structures

```
# ---------- DATA STRUCTURE ----------

# ---------- ARRAY ----------
    # initialized with [] square brackets
    # lists of dynamic size that can store values of different types
    # empty arrays should specify their data type

anArray = [1,2,3,4,5,6]
anotherArray = [1, "hello", 'x']
anIntArray = Array(Int32).new

# ARRAY METHODS

anArray[1] # returns 2, all data structures are zero-indexed
anArray[-1] # returns 6, negative indexing returns values from the back similar to Python
anArray[2,3] # returns [3,4,5], a start index and size returns a subarray starting from the specified index and counts to the specified size
anArray[1..3] # returns [2,3,4], a range returns a subarray comprising elements from the parent array of the specified indexes within the range
anArray << 7 # returns [1,2,3,4,5,6,7], appends a value to the array
anArray.pop # returns 7, pop operates the same as in Python and removes and returns the last element of the array
anArray.shift # returns 1, shift operates the same as in Python and removes and returns the first element of the array
anArray.includes? 3 # returns true, .includes? checks for the existence of a value within an array and returns a boolean value

# ---------- HASH ----------
    # equivalent to an associative array in PHP, a dictionary in Python or an object in Javascript
    # stores key-value pairs of any type, the key-value pairs can be of different types even within the same hash
```