**Hands – On Lab Workshop 3***.*

# AREA OF TRIANGLE

Write a function that takes the base and height of a triangle and return its area.

Example:

Areaoftriangle (3, 4) 6

Areaoftriangle (7, 8) 28 Notes

* Area of triangle is (base \* height)/2
* Don’t forget to return the result

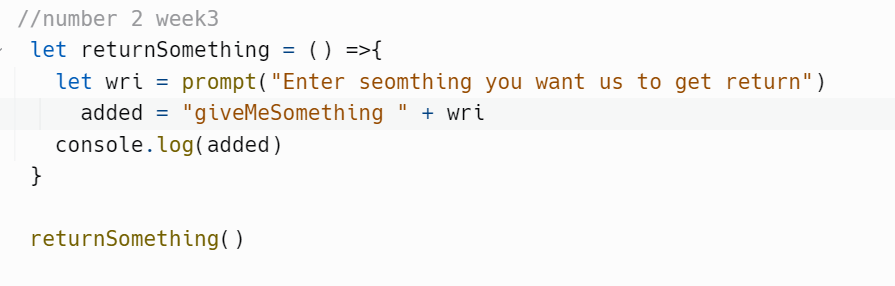


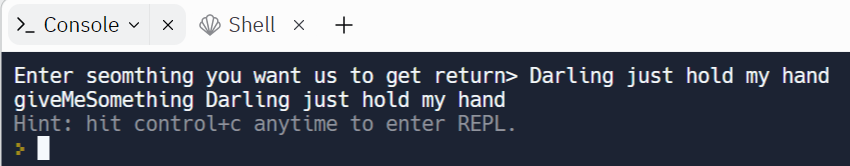
**RETURN SOMETHING TO ME!**

Write a function that returns the string "something" joined with a space " " and the given argument.

# Examples

giveMeSomething("is better than nothing") ➞ "something is better than nothing" giveMeSomething("Bob Jane") ➞ "something Bob Jane" giveMeSomething("something") ➞ "something something”



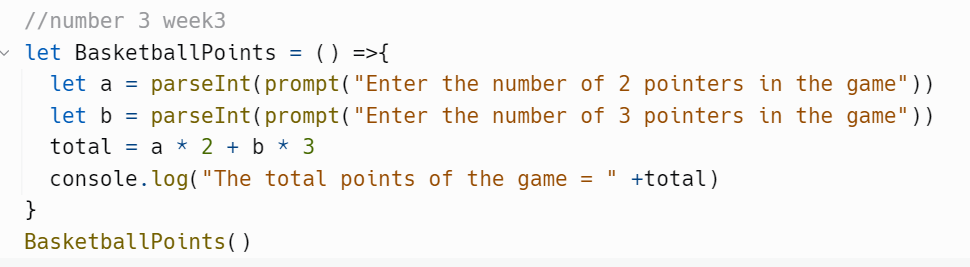


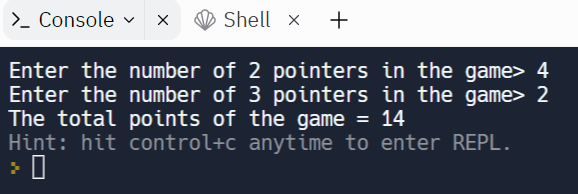
# BASKETBALL POINTS

You are counting points for a basketball game, given the amount of 2 – pointer scored and 3 – pointer scored, find the final points for the team and return the value.

Example:

points (3,5) 3\*2 + 5\*3 = 21 points (1,1) 5





**LESS THAN 100?**

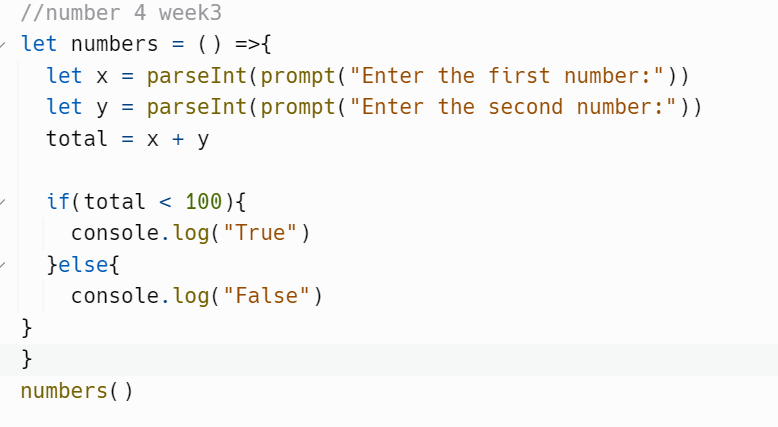
Given two numbers, return true if the sum of both numbers is less than 100.

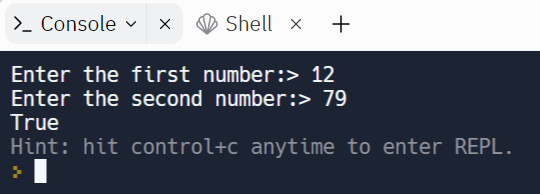
Otherwise return false.

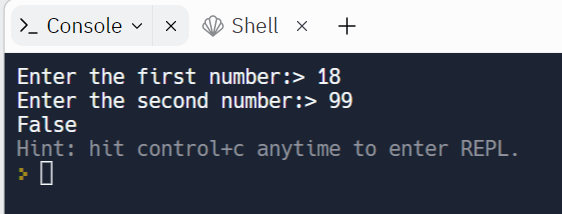
# Examples

lessThan100(22, 15) ➞ true

// 22 + 15 = 37 lessThan100(83, 34) ➞ false // 83 + 34 = 117 lessThan100(3, 77) ➞ true

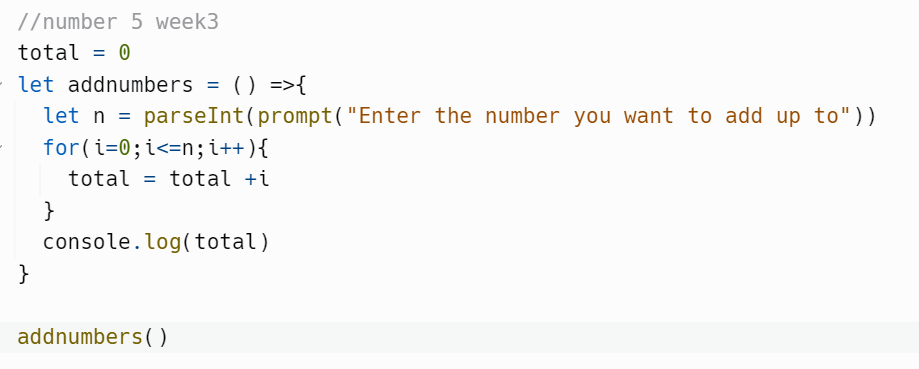


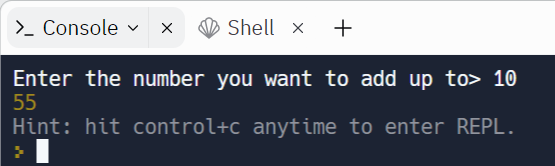




# ADD UPTO THE NUMBER FROM A SINGLE NUMBER

Create a function that takes a number as an argument. Add up all the numbers from 1 to the number you passed to the function. For example, if the input is 4 then your function should return 10 because 1+2+3+4 = 10





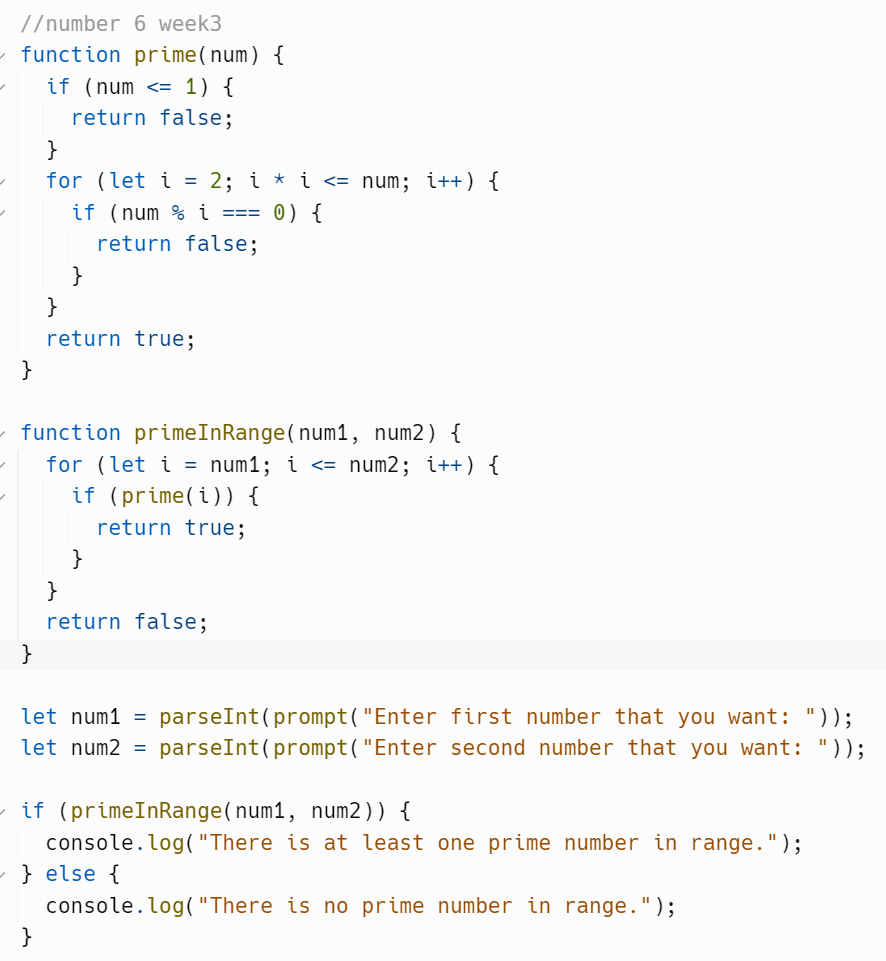
# ANY PRIME NUMBER IN RANGE

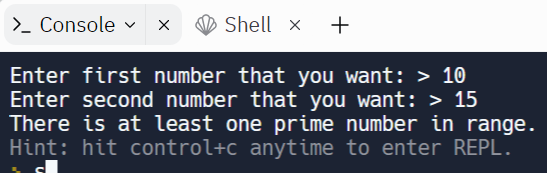
Create a function that return true if there is at least one prime number in the given range(n1 to n2) inclusive, false otherwise.

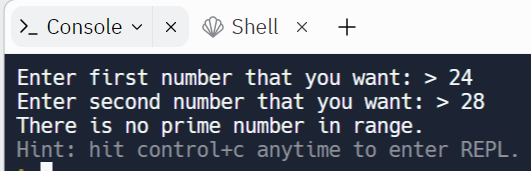
**Example:** primeInRange(10,15) true

// prime number is range : 11, 13 primeInRange(3,1) true

// prime number is range : 3, 5







# ODDISH VS. EVENISH

Create a function that determines whether a number is Oddish or Evenish. A number is Oddish if the sum of all of its digits is odd, and a number is Evenish if the sum of all of its digits is even. If a number is Oddish, return "Oddish". Otherwise, return "Evenish".

For example, oddishOrEvenish(121) should return "Evenish", since 1 + 2 + 1 =

4. oddishOrEvenish(41) should return "Oddish", since 4 + 1 = 5.

# Examples

oddishOrEvenish(43) ➞ "Oddish"

// 4 + 3 = 7 // 7 % 2 = 1

oddishOrEvenish(373) ➞ "Oddish"

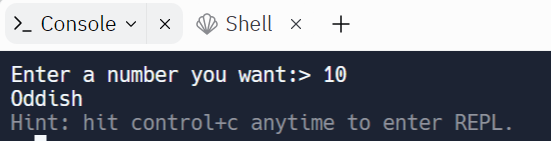
// 3 + 7 + 3 = 13 // 13 % 2 = 1

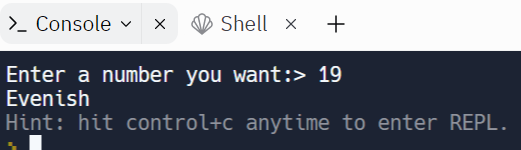
oddishOrEvenish(4433) ➞ "Evenish"

// 4 + 4 + 3 + 3 = 14

// 14 % 2 = 0







# LEFT SHIFT BY POWERS OF TWO

The left shift operation is similar to multiplication by powers oftwo.

Sample calculation using the left shift operator (<<):

10 << 3 = 10 \* 2^3 = 10 \* 8 = 80

-32 << 2 = -32 \* 2^2 = -32 \* 4 = -128

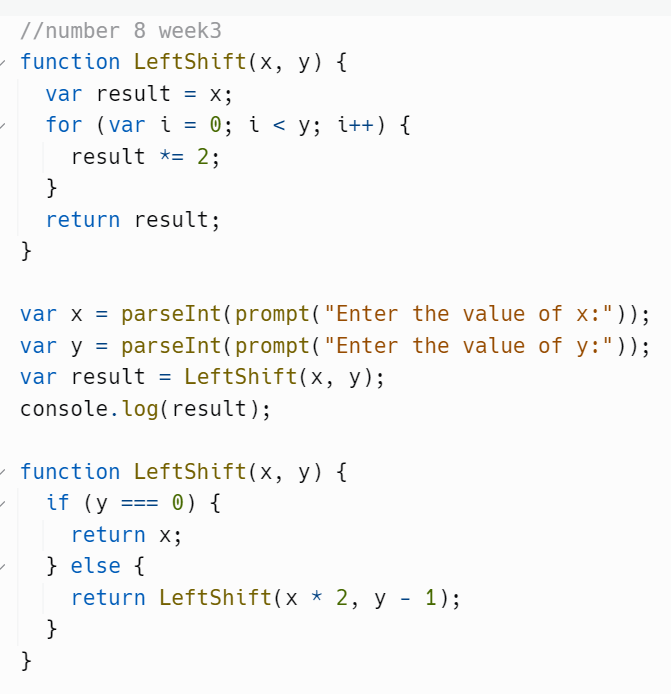
5 << 2 = 5 \* 2^2 = 5 \* 4 = 20

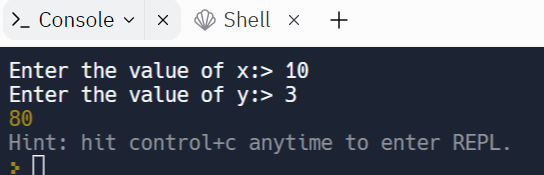
Write a function that mimics (withoutthe use of <<)the left shift operator and returns the resultfrom the two given integers.

# Examples

shiftToLeft(5, 2) ➞ 20 shiftToLeft(10, 3) ➞ 80 shiftToLeft(-32, 2) ➞ -128 shiftToLeft(-6, 5) ➞ -192 shiftToLeft(12, 4) ➞ 192 shiftToLeft(46, 6) ➞ 2944 Notes

* There will be no negative values for the second parameter y.
* This challenge is more like recreating the left shift operation,thus,the use ofthe operator directly is prohibited.
* Alternatively, you can solve this challenge via recursion.





# CONVERT A NUMBER TO BASE-2

Create a function that returns a base-2 (binary) representation of a base-10 (decimal) string number. To convertis simple: ((2) means base-2 and (10) means base-10) 010101001(2) = 1 + 8 + 32 + 128.

Going from rightto left,the value ofthe most right bitis 1, now from that every bitto the left will be x2. The values of an 8 bit binary number are (256, 128, 64, 32, 16, 8, 4, 2, 1).

# Examples

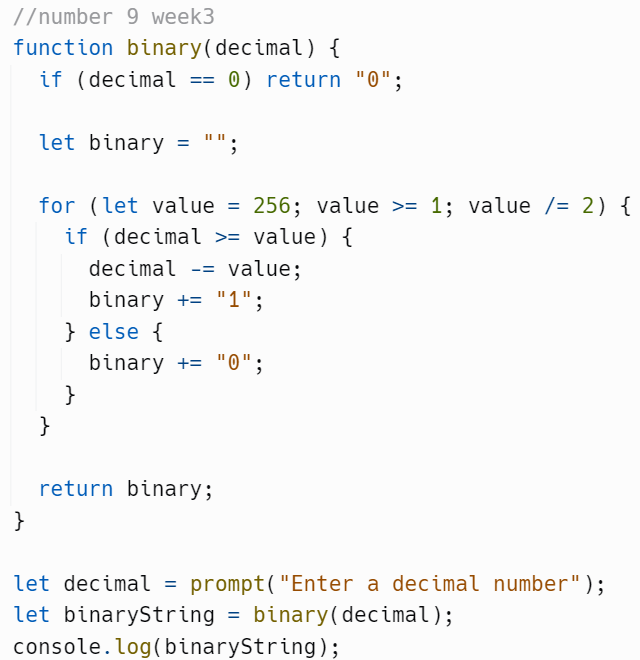
binary(1) ➞ "1"

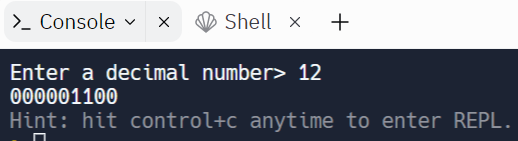
// 1\*1 = 1 binary(5) ➞ "101" // 1\*1 + 1\*4 = 5 binary(10) ➞ "1010"

// 1\*2 + 1\*8 = 10

Notes

* Numbers will always be below 1024 (notincluding 1024).
* The && operator could be useful.
* The strings will always go to the length at which the mostleft bit's value gets bigger than the number in decimal.
* If a binary conversion for 0 is attempted, return "0".





# GUESSING GAME

Generate a random number (do research) and store it in a variable. Write a program to take input from the user and tell them whether their guessed number is correct, greater or lesser than the original number. (100 – number of guesses) is the score of user. The program is expected to terminate once the number is guessed. Number should be between 1 – 100.

Example:

Random number generated by computer: 54

User input: 34

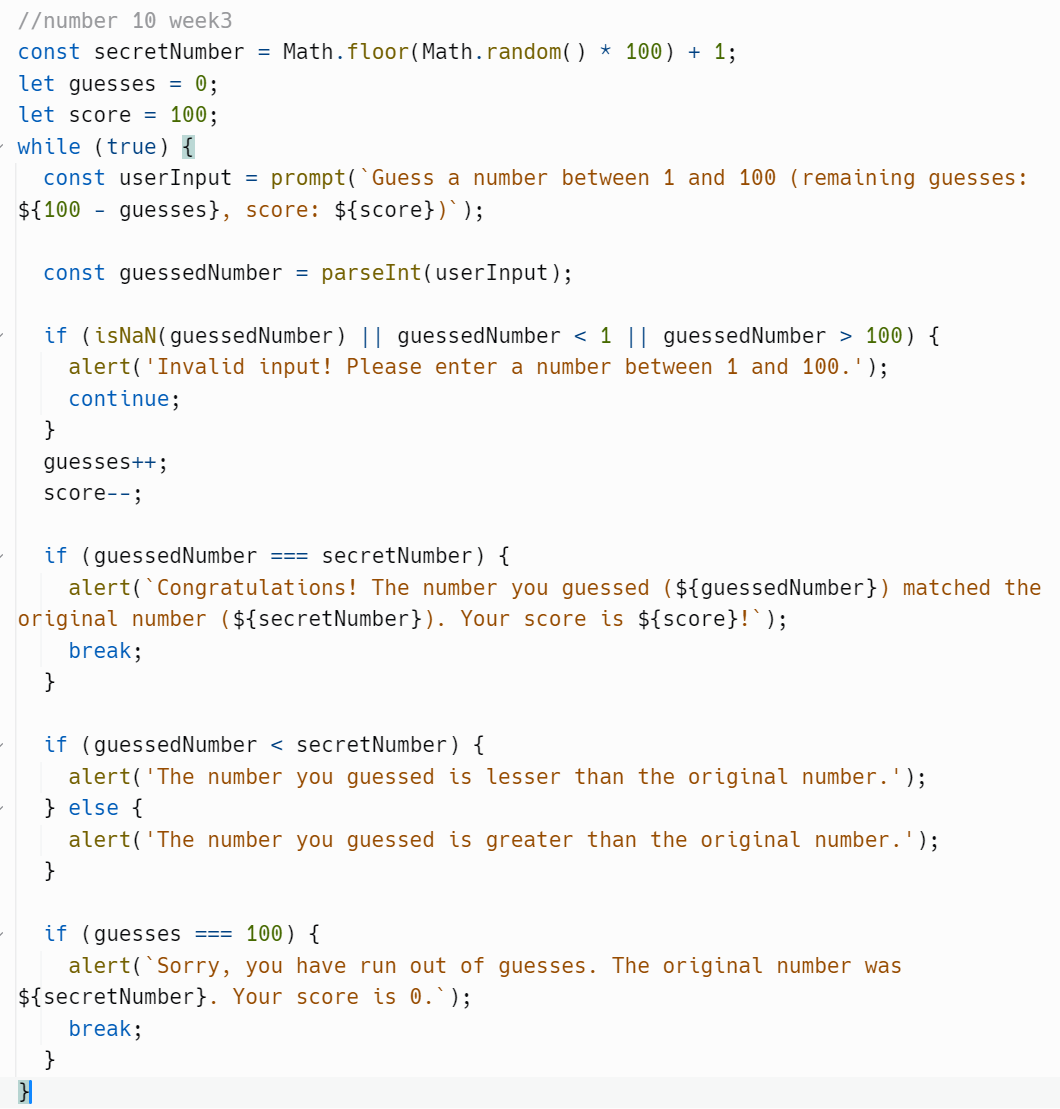
// lesser than original number

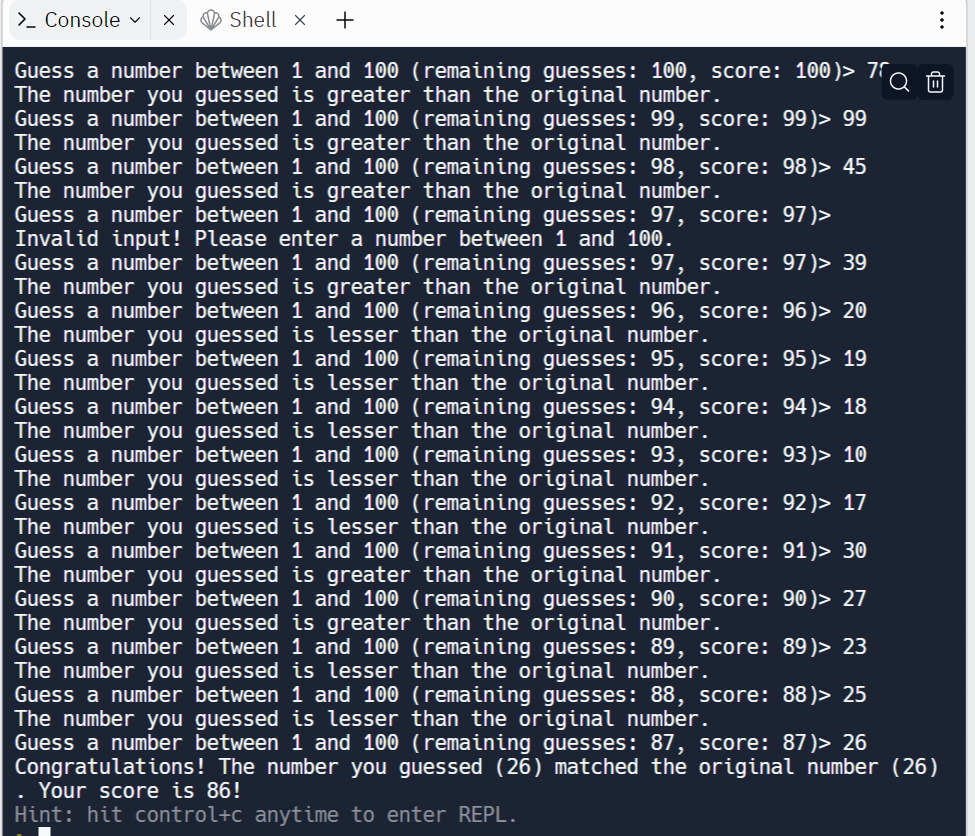
User input: 67

// greater than original number

User input: 54

// congratulations!!! The number you guessed matched the original number. Your score is 97!

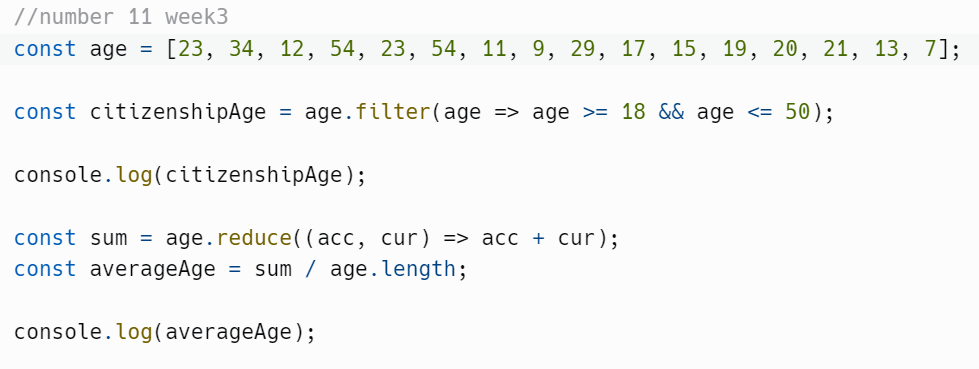


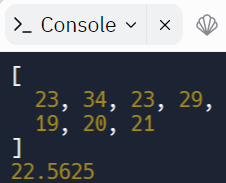


# HIGHER ORDER ARRAY METHODS

Const age = [23,34,12,54,23,54,11,9,29,17,15,19,20,21,13,7]

1. Filter the array of age who can apply for citizenships





1. Find the average age of a given array Const companies = [

{ name: "ABC", category: "Finance", start: 1981, end: 2004 },

{ name: "XYZ", category: "Retail", start: 1991, end: 20012 },

{ name: "DGF", category: "Finance", start: 1976, end: 2008 },

{ name: "LFT", category: "Retail", start: 1971, end: 1979 },

{ name: "MND", category: "Retail", start: 1995, end: 2010 },

{ name: "HCK", category: "Technology", start: 1987, end: 2011 },

{ name: "BMC", category: "Technology", start: 1989, end: 2009 },

{ name: "TIC", category: "Retail", start: 1993, end: 2005 },

{ name: "NAC", category: "Technology", start: 1991, end: 2010 },

{ name: "ITC", category: "Finance", start: 1998, end: 2016 }

];

1. Filter the retail companies
2. Get the 80s companies from the array
3. Get the companies that lasted for 10 or more years



