

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 $(\text{base} * \text{height})/2$
- Don't forget to return the result

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

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

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. $(100 - \text{number of guesses})$ 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!



```
index.js
2 let randomNumber = Math.floor(Math.random() * 100) + 1;
3 let guesses = 0;
4 let score = 100;
5
6 while (true) {
7   let guess = prompt("Guess a number between 1 and 100:");
8   guess = parseInt(guess);
9   guesses++;
10  if (guess === randomNumber) {
11    alert("Congratulations, you guessed the number in
12    `${guesses}` tries! Your score is `${score}`.");
13    break;
14  }
15  else if (guess > randomNumber) {
16    alert("Your guess is too high.");
17    score -= 1;
18  }
19  else {
20    alert("Your guess is too low.");
21    score -= 1;
22  }
23  if (guesses === 100) {
24    alert("Sorry, you have run out of guesses. The number was "
25    + randomNumber + ".");
26    break;
27  }
28 }
```

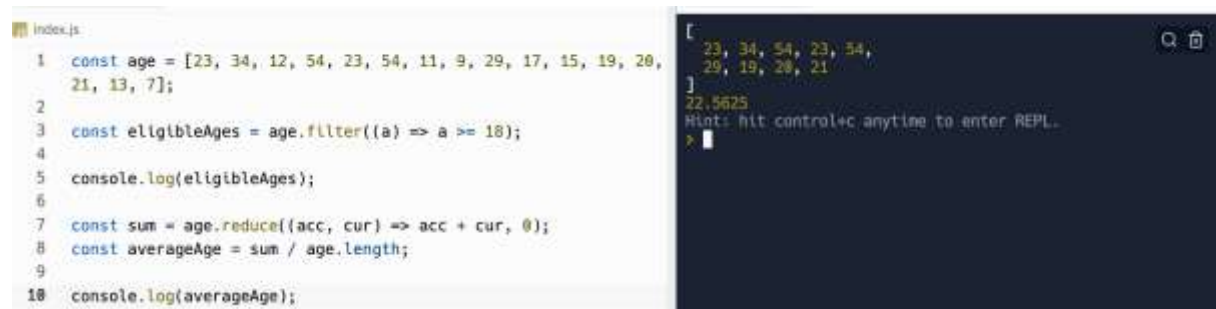
```
Guess a number between 1 and 100:> 5
Your guess is too low.
Guess a number between 1 and 100:> 50
Your guess is too low.
Guess a number between 1 and 100:> 70
Your guess is too low.
Guess a number between 1 and 100:> 90
Your guess is too low.
Guess a number between 1 and 100:> 99
Congratulations, you guessed the number in 5 tries! Your
score is 96.
Hint: hit control+c anytime to enter REPL.
> 
```

HIGHER ORDER ARRAY METHODS

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

- a. Filter the array of age who can apply for citizenships

b. Find the average age of a given array



The image shows a code editor on the left and a terminal on the right. The code editor contains the following JavaScript code:

```
1 const age = [23, 34, 12, 54, 23, 54, 11, 9, 29, 17, 15, 19, 20, 21, 13, 7];
2
3 const eligibleAges = age.filter((a) => a >= 18);
4
5 console.log(eligibleAges);
6
7 const sum = age.reduce((acc, cur) => acc + cur, 0);
8 const averageAge = sum / age.length;
9
10 console.log(averageAge);
```

The terminal on the right shows the output of the code:

```
[
  23, 34, 54, 23, 54,
  29, 19, 20, 21
]
22.5625
Hint: hit control+c anytime to enter REPL.
>
```

c. 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 }
];
```

The image shows a code editor with a file named `index.js` and a console window. The code in `index.js` defines an array of company objects and performs three filter operations. The console shows the results of these operations.

```
1 - const companies = [
2   { name: "ABC", category: "Finance", start: 1981, end: 2004 },
3   { name: "XYZ", category: "Retail", start: 1991, end: 2012 },
4   { name: "DGF", category: "Finance", start: 1976, end: 2006 },
5   { name: "LFT", category: "Retail", start: 1971, end: 1979 },
6   { name: "MND", category: "Retail", start: 1995, end: 2010 },
7   { name: "HCK", category: "Technology", start: 1987, end: 2011 },
8   { name: "BMC", category: "Technology", start: 1989, end: 2009 },
9   { name: "TIC", category: "Retail", start: 1993, end: 2005 },
10  { name: "NAC", category: "Technology", start: 1991, end: 2010 },
11  { name: "ITC", category: "Finance", start: 1998, end: 2016 }
12 ];
13 const retailCompanies = companies.filter(company =>
14   company.category === "Retail");
15 console.log(retailCompanies);
16 const eightiesCompanies = companies.filter(company =>
17   company.start >= 1980 && company.start < 1990);
18 console.log(eightiesCompanies);
19 const lastedTenYearsOrMore = companies.filter(company =>
20   (company.end - company.start) >= 10);
21 console.log(lastedTenYearsOrMore);
```

The console output shows the results of the three filter operations:

```
[
  { name: "XYZ", category: "Retail", start: 1991, end: 2012 },
  { name: "LFT", category: "Retail", start: 1971, end: 1979 },
  { name: "MND", category: "Retail", start: 1995, end: 2010 },
  { name: "TIC", category: "Retail", start: 1993, end: 2005 },
  { name: "ABC", category: "Finance", start: 1981, end: 2004 },
  { name: "HCK", category: "Technology", start: 1987, end: 2011 },
  { name: "BMC", category: "Technology", start: 1989, end: 2009 },
  { name: "DGF", category: "Finance", start: 1976, end: 2006 },
  { name: "NAC", category: "Technology", start: 1991, end: 2010 },
  { name: "ITC", category: "Finance", start: 1998, end: 2016 }
],
[
  { name: "ABC", category: "Finance", start: 1981, end: 2004 },
  { name: "XYZ", category: "Retail", start: 1991, end: 2012 },
  { name: "DGF", category: "Finance", start: 1976, end: 2006 },
  { 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 }
],
[
  { name: "XYZ", category: "Retail", start: 1991, end: 2012 },
  { name: "DGF", category: "Finance", start: 1976, end: 2006 },
  { 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 }
]
```

- Filter the retail companies
- Get the 80s companies from the array
- Get the companies that lasted for 10 or more years