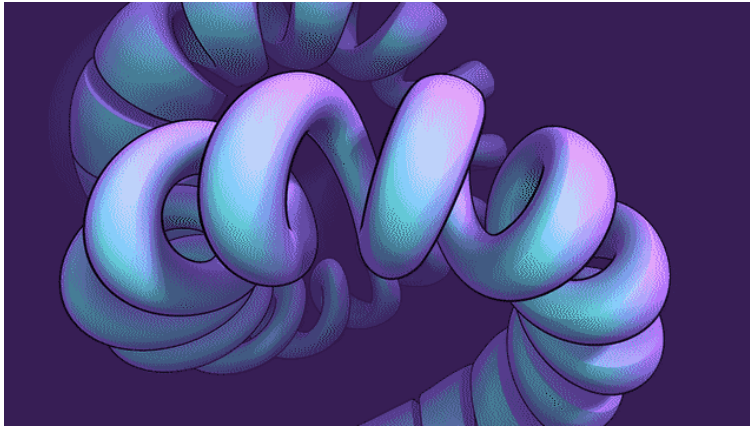




Project: While Loops

On command line, create a file called `'while_loops_project.js'` in your `ASCProjects/week1/day3` directory.

Part 1a - COUNTING DOWN

Task:

Write a `while` loop that **counts down** from 100 to 0.

Note:

A **countdown** loop would be very similar to the `while` loop that counts up (*written in Code Along*) with a slight change in logic.

Sample run:

```
ASCStudent@DESKTOP-RLIDLQU MINGW64 ~/Desktop/ASCProjects/week1/day3
$ node while_loops_project.js
100
99
98
97
96
95
94
93
92
91
90
89
88
87
```

Etc, etc. (You get it)

Part 1b - COUNTING DOWN + USER INPUT

Task:

Modify the countdown loop from [Part 1a](#) to have the loop count down from **a number inputted by the user**.

Sample run:

```
ASCStudent@DESKTOP-RLIDLQU MINGW64 ~/Desktop/ASCProjects/week1/day3
$ node while_loops_project.js 5
5
4
3
2
1
0
```

Note: **User input** in sample run: 5

Part 2 - WHILE LOOP GYMNASTICS

Question #1:

How many `Little Donald Drumpf`'s will be printed onto the console? 🤔

In other words, how many times will the `while` loop iterate?

```
let c = 0;
while (c < 5) {
  console.log("Little Donald Drumpf");
  c += 2;
}
```

Question #2:

How many `Little Donald Drumpf`'s will be printed onto the console? 🤔

```
let a = 0;
let b = 5;

while (a < b) {
  console.log("Little Donald Drumpf");
  a++;
  b++;
}
```

Question #3:

How many `Little Donald Drumpf`'s will be printed onto the console? 

```
let a = 0;
let b = 5;

while (a < b) {
  console.log("Little Donald Drumpf");
  a += 2;
  b += 1;
}
```

Question #4:

How many `Little Donald Drumpf`'s will be printed onto the console? 

```
let counter = 10;

while (counter > 0) {
  console.log("Little Donald Drumpf");
  counter -= 3;

  if (counter % 2 == 0) {
    break;
  }
}
```

Question #5:

Modify the faulty `while` loop to print **10** `Little Donald Drumpf` lines onto the console? 

```
let num = 0;

while (num > 10) {
  break;
  console.log("Little Donald Drumpf");
  num++;
}
```

Part 3a - TWO RANDOM NUMBERS

Write a program that would **continually** generate **2 random integers** between 0 and 10 (inclusive). The program would end after it generates **two of the same random integer**.

Note:

A `while` loop is **necessary** here as there is no telling how many tries the program will take to generate 2 of the same random integer.

Sample run #1:

```
ASCStudent@DESKTOP-RLIDLQU MINGW64 ~/Desktop/ASCProjects/week1/day3
$ node while_loops_project.js
1st random number: 6 2nd random number: 7
1st random number: 7 2nd random number: 6
1st random number: 0 2nd random number: 1
1st random number: 6 2nd random number: 8
1st random number: 9 2nd random number: 9
Same random numbers! End Loop
```

Sample run #2:

```
ASCStudent@DESKTOP-RLIDLQU MINGW64 ~/Desktop/ASCProjects/week1/day3
$ node while_loops_project.js
1st random number: 0 2nd random number: 5
1st random number: 5 2nd random number: 4
1st random number: 5 2nd random number: 6
1st random number: 6 2nd random number: 10
1st random number: 9 2nd random number: 6
1st random number: 0 2nd random number: 7
1st random number: 8 2nd random number: 10
1st random number: 8 2nd random number: 2
1st random number: 3 2nd random number: 6
1st random number: 6 2nd random number: 4
1st random number: 0 2nd random number: 9
1st random number: 6 2nd random number: 2
1st random number: 0 2nd random number: 6
1st random number: 4 2nd random number: 4
Same random numbers! End Loop
```

Sample run #3:

```
ASCStudent@DESKTOP-RLIDLQU MINGW64 ~/Desktop/ASCProjects/week1/day3
$ node while_loops_project.js
1st random number: 8 2nd random number: 8
Same random numbers! End Loop
```

Part 3b - TWO RANDOM NUMBERS

Modify your program from [Part 3a](#) to output the **number of tries** it took to generate 2 of the same random integer.

Sample run:

```
ASCStudent@DESKTOP-RLIDLQU MINGW64 ~/Desktop/ASCProjects/week1/day3
$ node while_loops_project.js
Try #1 - 1st random number: 5 2nd random number: 4
Try #2 - 1st random number: 7 2nd random number: 6
Try #3 - 1st random number: 0 2nd random number: 4
Try #4 - 1st random number: 8 2nd random number: 3
Try #5 - 1st random number: 0 2nd random number: 8
Try #6 - 1st random number: 4 2nd random number: 4
Same random numbers on try #6! End Loop
```

Extra Credit - BINARY NUMBERS

Write a program that **takes a positive number** from command line and outputs the **binary value** of the number and the **calculation steps** involved.

For more info, see **Method 1 - Performing Short Division by Two with Remainder:**

<https://www.wikihow.com/Convert-from-Decimal-to-Binary>

Sample runs:

User Input: 6

```
ASCStudent@DESKTOP-RLIDLQU MINGW64 ~/Desktop/ASCProjects/week1/day3
$ node while_loops_project.js 6
6 % 2 = 0 --- 0
6 / 2 = 3

3 % 2 = 1 --- 10
3 / 2 = 1

1 % 2 = 1 --- 110
1 / 2 = 0

Number in decimal: 6
Number in binary : 110
```

User Input: 14

```
ASCStudent@DESKTOP-RLIDLQU MINGW64 ~/Desktop/ASCProjects/week1/day3
$ node while_loops_project.js 14
14 % 2 = 0 --- 0
14 / 2 = 7

7 % 2 = 1 --- 10
7 / 2 = 3

3 % 2 = 1 --- 110
3 / 2 = 1

1 % 2 = 1 --- 1110
1 / 2 = 0

Number in decimal: 14
Number in binary : 1110
```

User Input: 100

```
ASCStudent@DESKTOP-RLIDLQU MINGW64 ~/Desktop/ASCProjects/week1/day3
$ node while_loops_project.js 100
100 % 2 = 0 --- 0
100 / 2 = 50

50 % 2 = 0 --- 00
50 / 2 = 25

25 % 2 = 1 --- 100
25 / 2 = 12

12 % 2 = 0 --- 0100
12 / 2 = 6

6 % 2 = 0 --- 00100
6 / 2 = 3

3 % 2 = 1 --- 100100
3 / 2 = 1

1 % 2 = 1 --- 1100100
1 / 2 = 0

Number in decimal: 100
Number in binary : 1100100
```