Lab 10 Javascript 3 - Random Number and While Lo

General Information

What you should do

• You should first review Lecture 4 (Part B) up to slide 16 and be familiar with the concepts on Javascript as well as the related code examples so that you can refer to the specific techniques from the corresponding slides when completing the tasks in this lab.
• You should try to come up with the code yourself as much as possible. Do not be afraid of making mistakes, since debugging (finding out where your code goes wrong and fixing it) is part of the learning process.
• We do not give out model programs to the exercises. There can be multiple ways to write the code that solves the same problem. It is important that you build up the program logic yourself instead of merely looking at some code that you do not understand. At any time if you are lost or if you have any questions, feel free to ask the instructor, tutor, or teaching assistant and we will be very happy to help you.

Self-Discovery

• Most lab tasks are designed to be relatively simple such that you can take the time to think about the related underlying concepts. Besides, we also encourage you to discover things on your own which may not be specified in the tasks.

Task 1.1 Create a New HTML File from Template in Komodo Edit

Create a new HTML file from Komodo Edit in the same way as described in Lab 08 Task 1.1:

1. Open Komodo Edit and select “New File from Template” (or press the shortcut keys Ctrl+Shift+N)
2. A dialog box will pop up on the right. Select “HTML 5” from the list

Remember to save the file after you make changes before opening it on a browser to view it.

Note that you should NOT copy and paste any code from this document to your code editor. Type up the code instead, otherwise it may not work properly, especially with symbols such as quotation marks “ ”.

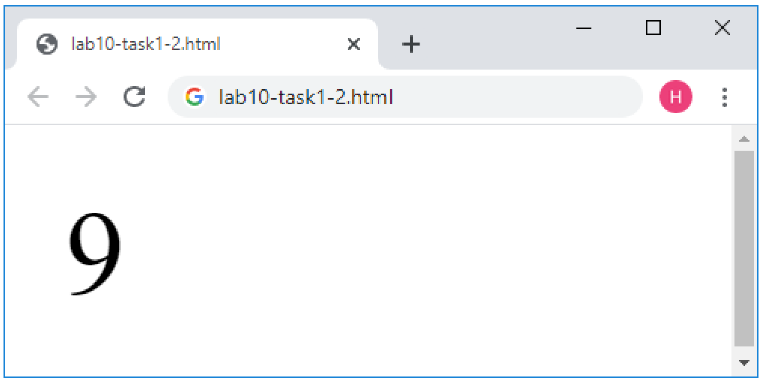
Task 1.2 Generate Random Numbers

From the lecture, you have learnt that the function `Math.random()` generates a random floating-point numbers in the range of 0 (inclusive) to 1 (exclusive). To generate a random integer, you scale the above function by multiplying it with an integer, apply the floor function `Math.floor()` and then add an offset to it. The following program shows an example of generating a random integer in the range of 1 to 10:

```
<!DOCTYPE html>

<html>
<head>
    <title></title>
    <script>
           function init() {
                var d;
                d = Math.floor(Math.random()*10)+1;
                document.getElementById("result").innerHTML = d;
           }
    </script>
</head>
<body onload="init();">
    <div id="result"></div>
</body>
</html>
```

The following shows a sample webpage from the above program:



```
9
```

You can click the refresh button and see another random integer shows up on the webpage.

Exercise 1.2: Modify the program so that it generates different ranges of random numbers for each of the following cases:

a) Generate a random number x such that $x \in \{-1, 1\}$, i.e., the value of x is either -1 or 1
b) Generate a random number x such that $x \in \{0, 1, 2\}$ but instead of equal probabilities, the probabilities of getting 0,1,2 are 0.25,0.5,0.25 respectively
c) Generate a random number y such that $y \in \{1.5, 2.0, 2.5, 3.0, 3.5, 4.0\}$
d) Generate a random floating-point number x such that $-3<x\leq-1$ or $1\leq x<3$, i.e., $x \in]-3,-1] \cup [1,3[$
e) Generate a random floating-point number x such that $0<x\leq2$ or $4\leq x<6$

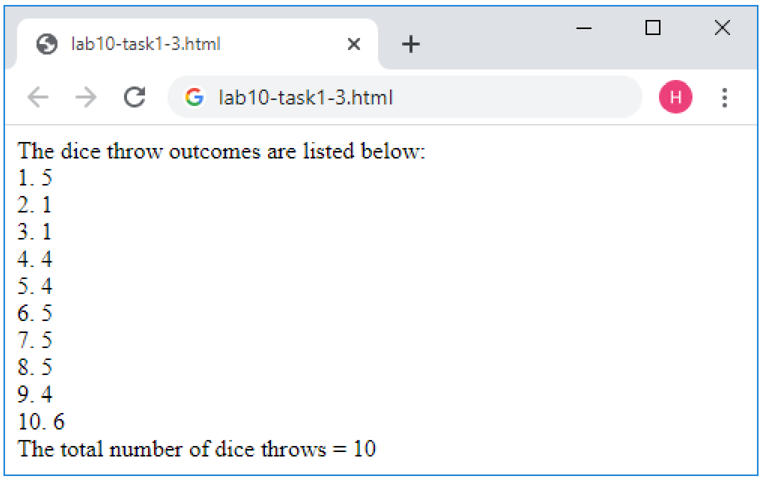
Try to come up with 2 ways of solving the above problems: 1) without using conditionals; 2) by using conditionals.

Task 1.3 Simulate Dice Throw

In this task, we would like to simulate a dice throw with the following requirements for the program:

1. When the webpage is opened on the browser, the program will generate a random integer in the range of 1 to 6 to simulate a dice throw. The outcome of the dice throw is shown on the webpage.
2. If the outcome of the dice throw is not 6, then step 1) will be repeated.
3. If the outcome of the dice throw is 6, then there is no need to throw the dice any more. The webpage will show how many times the dice has been thrown.

A sample webpage is shown in the following screenshot:



```
The dice throw outcomes are listed below:
1. 5
2. 1
3. 1
4. 4
5. 4
6. 5
7. 5
8. 5
9. 4
10. 6
The total number of dice throws = 10
```

Now try to compose the program by yourself. If you have trouble writing it, then refer to the following hints:

- Use the `onload` event handler to call a function so that the function will be called after the webpage is opened on the browser
- Refer to Task 1.2 about how to generate a random integer in the range of 1 to 6
- Use the `while` loop with the appropriate continuation condition
- Use a variable to count how many iterations carried out by the loop
- Create a `div` element in the HTML and assign an ID to it. Then set `document.getElementById([ID]).innerHTML` to the corresponding message listing the random numbers generated at each loop iteration and the total number of dice throw at the end (You need to gradually build up this message by appending the proper string in the while loop)

We will now show you 2 sample HTML and Javascript codes implemented according to the above requirements for your reference:

Implementation 1:

```html
<!DOCTYPE html>

<html>
<head>
    <title></title>
    <script>
        function init() {
            var count, d, s;
            d = Math.floor(Math.random()*6)+1;
            count = 1;
            s = document.getElementById("result").innerHTML + count + ". " +d+"<br/>";
            while (d!=6) {
                d = Math.floor(Math.random()*6)+1;
                count++;
                s += count + ". " +d+"<br/>";
            }
            s += "The total number of dice throws = "+count;
            document.getElementById("result").innerHTML = s;
        }
    </script>
</head>
<body onload="init();">
    <div id="result">The dice throw outcomes are listed below:<br/></div>
</body>
</html>
```

Implementation 2:

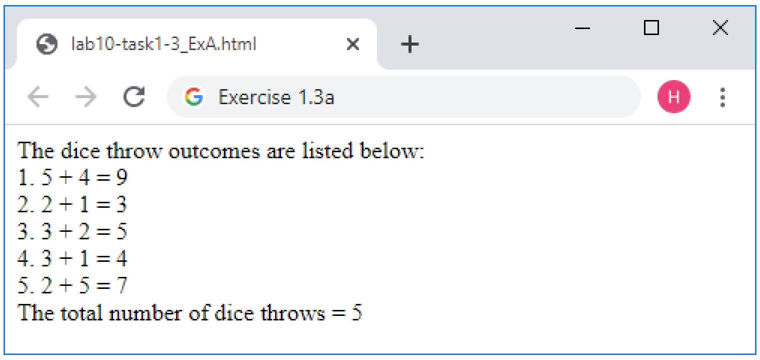
```html
<!DOCTYPE html>

<html>
<head>
    <title></title>
    <script>
        function init() {
            var count, d, s, isDone;
            count = 0;
            s = document.getElementById("result").innerHTML;
            isDone = false;
            while (!isDone) {
                d = Math.floor(Math.random()*6)+1;
                count++;
                s += count + ". " +d+"<br/>";
                isDone = d==6;
            }
            s += "The total number of dice throws = "+count;
            document.getElementById("result").innerHTML = s;
        }
    </script>
</head>
<body onload="init();">
    <div id="result">The dice throw outcomes are listed below:<br/></div>
</body>
</html>
```

You can see that one difference between these 2 implementations is that the dice throw is simulated once before the while loop in Implementation 1 but no dice throw is simulated before the while loop in Implementation 2. For implementation 2, the while loop is executed at least once since `isDone` is initialized to be false and the continuation of the while loop is `!isDone`. The variable `count` is initialized as 1 and 0 for Implementation 1 and Implementation 2 respectively.

Exercise 1.3: Modify your program so that it satisfies the following specifications. You should focus on one specification at a time, test the program each time after you finish with one specification to make sure that it is working correctly before moving to work on the next specification:

a) Instead of just 1 dice, each time 2 dice are thrown instead. The two dice outcomes as well as their sum are displayed. The two dice will be thrown again if their sum is not equal to 7. The total number of dice throws is shown at the end of the webpage. A sample webpage is shown in the following screenshot:



```
The dice throw outcomes are listed below:
1. 5 + 4 = 9
2. 2 + 1 = 3
3. 3 + 2 = 5
4. 3 + 1 = 4
5. 2 + 5 = 7
The total number of dice throws = 5
```

b) Each time 3 dice are thrown. The 3 dice outcomes will be displayed. Write separate programs to let the dice throwing stop according to each of the following conditions:
   i) The outcomes of 3 dice throws are the same
   ii) The outcomes of 3 dice throws are all different
   iii) At least the outcomes of 2 dice throws are the same
   iv) At least the outcomes of 2 dice throws are different

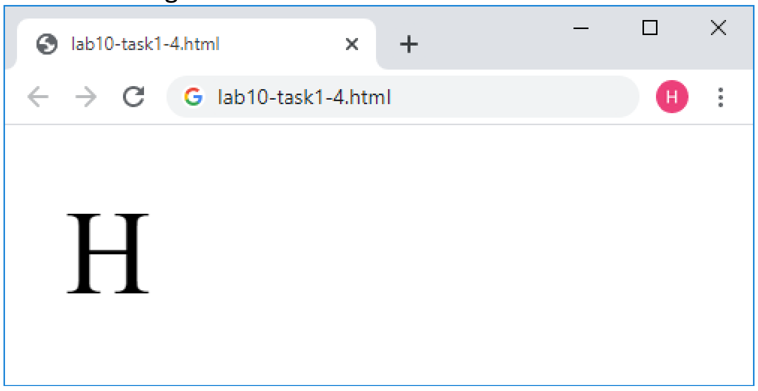
Task 1.4 Generate Random Characters

Download the file lab10-task1-4.html from the Canvas course page. You will see the following code if you open the file on Komodo Edit:

```
<!DOCTYPE html>

<html>
<head>
    <title></title>
    <script>
        function randomUppercaseCharacter() {
            var d, c;
            d = Math.floor(Math.random()*26);
            c = String.fromCharCode(65+d);
            return c;
        }
        function init() {
            var s;
            s = randomUppercaseCharacter();
            document.getElementById("result").innerHTML = s;
        }
    </script>
</head>
<body onload="init();">
    <div id="result"><br/></div>
</body>
</html>
```

When you open this html file on a browser, you will see a random character displayed on the webpage as shown in the following screenshot:



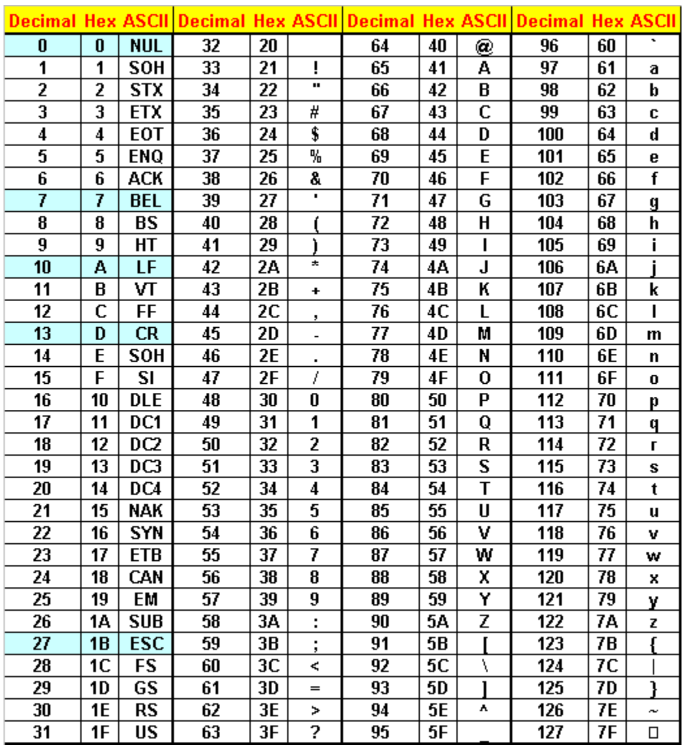
Here are some explanations about the code:

Line 9: `d = Math.floor(Math.random()*26);`

`d` is assigned the value of a random integer in the range of 0 to 25.

Line 10: `c = String.fromCharCode(65+d);`

`c` is assigned the value of a character corresponding to the Unicode number `65+d`. Since the value `d` is in the range of 0 to 25, the value of `65+d` is in the range of 65 to 90. Besides, the first 128 Unicodes represent the same characters as the ASCII codes. From Lecture 2, you have seen the following ASCII table:



Decimal	Hex	ASCII	Decimal	Hex	ASCII	Decimal	Hex	ASCII	Decimal	Hex	ASCII
0	0	NUL	32	20		64	40	@	96	60	`
1	1	SOH	33	21	!	65	41	A	97	61	a
2	2	STX	34	22	"	66	42	B	98	62	b
3	3	ETX	35	23	#	67	43	C	99	63	c
4	4	EOT	36	24	$	68	44	D	100	64	d
5	5	ENQ	37	25	%	69	45	E	101	65	e
6	6	ACK	38	26	&	70	46	F	102	66	f
7	7	BEL	39	27	'	71	47	G	103	67	g
8	8	BS	40	28	(	72	48	H	104	68	h
9	9	HT	41	29	)	73	49	I	105	69	i
10	A	LF	42	2A	*	74	4A	J	106	6A	j
11	B	VT	43	2B	+	75	4B	K	107	6B	k
12	C	FF	44	2C	,	76	4C	L	108	6C	l
13	D	CR	45	2D	-	77	4D	M	109	6D	m
14	E	SOH	46	2E	.	78	4E	N	110	6E	n
15	F	SI	47	2F	/	79	4F	O	111	6F	o
16	10	DLE	48	30	0	80	50	P	112	70	p
17	11	DC1	49	31	1	81	51	Q	113	71	q
18	12	DC2	50	32	2	82	52	R	114	72	r
19	13	DC3	51	33	3	83	53	S	115	73	s
20	14	DC4	52	34	4	84	54	T	116	74	t
21	15	NAK	53	35	5	85	55	U	117	75	u
22	16	SYN	54	36	6	86	56	V	118	76	v
23	17	ETB	55	37	7	87	57	W	119	77	w
24	18	CAN	56	38	8	88	58	X	120	78	x
25	19	EM	57	39	9	89	59	Y	121	79	y
26	1A	SUB	58	3A	:	90	5A	Z	122	7A	z
27	1B	ESC	59	3B	;	91	5B	[	123	7B	{
28	1C	FS	60	3C	<	92	5C	\	124	7C	|
29	1D	GS	61	3D	=	93	5D	]	125	7D	}
30	1E	RS	62	3E	>	94	5E	^	126	7E	~
31	1F	US	63	3F	?	95	5F	_	127	7F	□

You can see from the above table that when the ASCII codes are 65 to 90, the corresponding characters are the uppercase alphabets A to Z. As a result, line 10 is assigning `c` the value of an uppercase character.

Line 11: `return c;`

The value of `c` is returned for this function call.

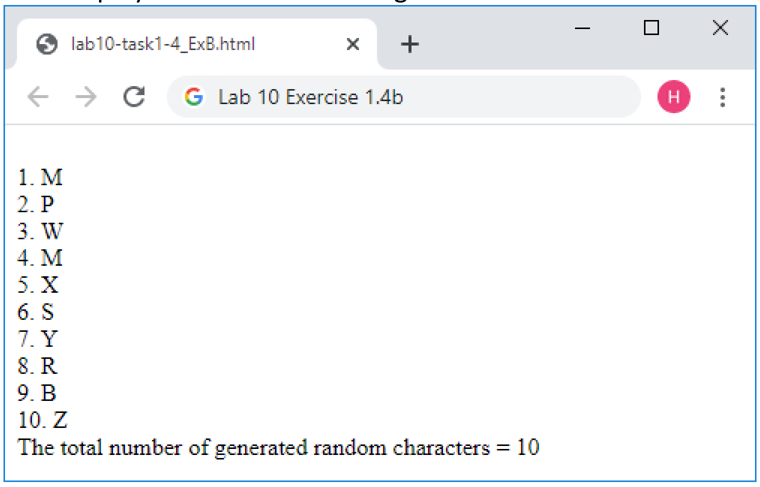
Line 15: `s = randomUppercaseCharacter();`

The function `randomUppercaseCharacter()` is called. This function would return a random uppercase alphabet, and its value will be assigned to `s`.

Exercise 1.4: Modify the program so that it satisfies the following specifications. You should focus on one specification at a time, test the program each time after you finish with one specification to make sure that it is working correctly before moving to work on the next specification:

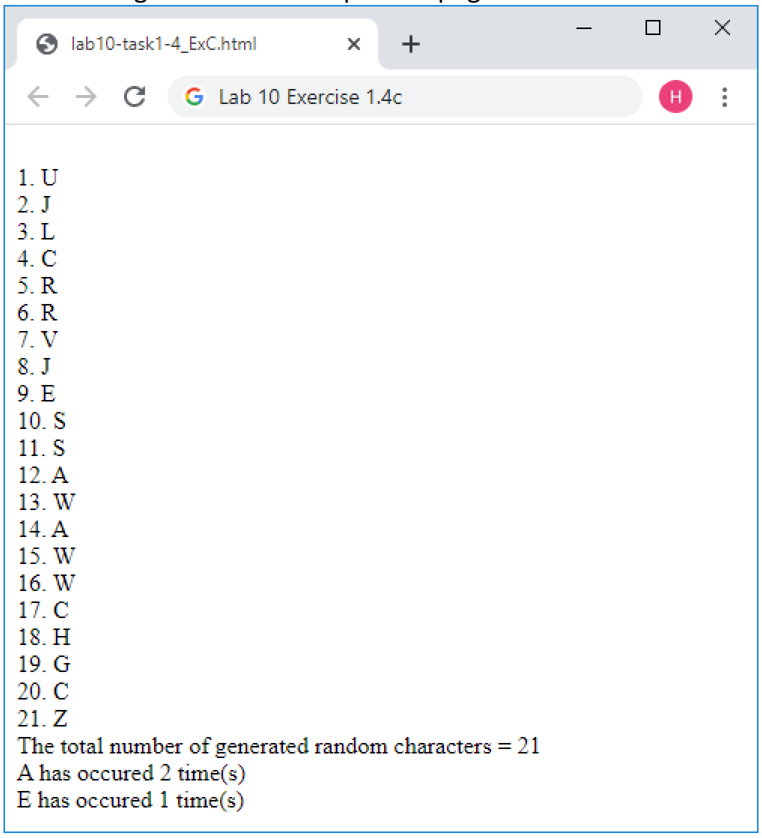
a) Instead of generating an uppercase alphabet, generate a random lowercase alphabet instead.

b) Keep generating and displaying the random uppercase alphabets until the character ‘Z’ is displayed. Then display the total number of generated random characters. A sample webpage is shown below:



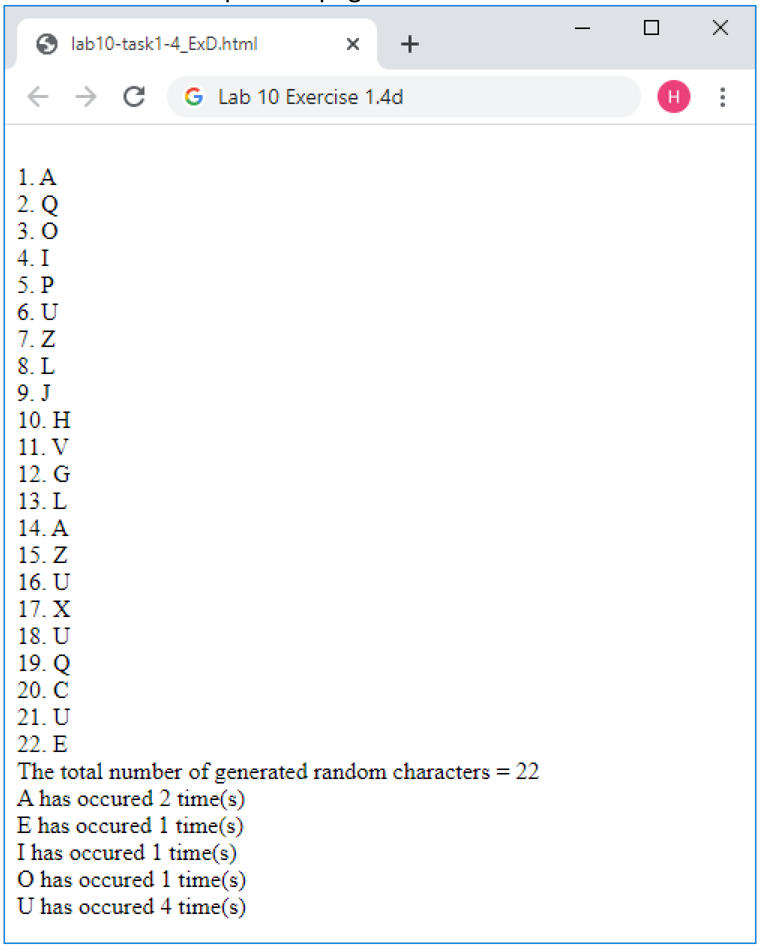
```
1. M
2. P
3. W
4. M
5. X
6. S
7. Y
8. R
9. B
10. Z
The total number of generated random characters = 10
```

c) Keep generating and displaying the random uppercase alphabets until the character ‘Z’ is displayed. Count how many times the letter ‘A’ and how many times the letter ‘E’ have been generated. Then display the total number of generated random characters as well as the number of times ‘A’ and ‘E’ have been generated. A sample webpage is shown below:



```
1. U
2. J
3. L
4. C
5. R
6. R
7. V
8. J
9. E
10. S
11. S
12. A
13. W
14. A
15. W
16. W
17. C
18. H
19. G
20. C
21. Z
The total number of generated random characters = 21
A has occured 2 time(s)
E has occured 1 time(s)
```

d) Keep generating and displaying the random uppercase alphabets until each vowel ('A', 'E', 'I', 'O', 'U') has been generated at least once. Count how many times each vowel has been generated. Then display the total number of generated random characters as well as the number of occurrence for each vowel. A sample webpage is shown below:



```
1. A
2. Q
3. O
4. I
5. P
6. U
7. Z
8. L
9. J
10. H
11. V
12. G
13. L
14. A
15. Z
16. U
17. X
18. U
19. Q
20. C
21. U
22. E
The total number of generated random characters = 22
A has occured 2 time(s)
E has occured 1 time(s)
I has occured 1 time(s)
O has occured 1 time(s)
U has occured 4 time(s)
```

Task 2 Review Previous Labs

You can review the previous labs by reading and following the lab instructions, and check that you are able to complete the lab tasks and exercises by yourself. If you have any questions, you can ask the lab tutor or TA during the lab session.

Task 3 Complete the assessment from the Canvas course page

You should complete the Lab 10 Assessment from the Canvas course page before the posted deadline.

Task 4 Challenge your classmates

You can first reflect on what you have learnt in this lab, and then come up with problems to challenge your classmates. You can post your problem on the Canvas course page, under this Discussion page. One should be able to solve your problem by using what he/she learns in Lab 10. You will not get extra marks by posting a challenging problem or solving a challenging problem posted by another student, but you will earn your fame so that you can impress the course leader, the lab tutors, and your classmates.