Revision Exercises (not assessed)

Reading week: week of 15th February 2021

Note: In this revision exercises sheet, we have exercises based on the first half of the module. You can use this as preparation for the final online test. The marks associated with each question are indicative of how many marks such a question could be worth in an online test.

Question (a): Which of the following lines of JavaScript prints a string to the console?

1. `console.log(100 / 2);`
2. `console.log((100 / 2) === 50);`
3. `console.log("" + 100 / 2);`
4. `console.log([100,2]);`

[2 marks]

Question (b): Write a function in JavaScript that takes a number `n` as an input parameter (which is assumed to be an integer) and returns `true` if the number is perfectly divisible by three, i.e. leaves no remainders when divided by three, or returns `false` if it is not perfectly divisible by three.

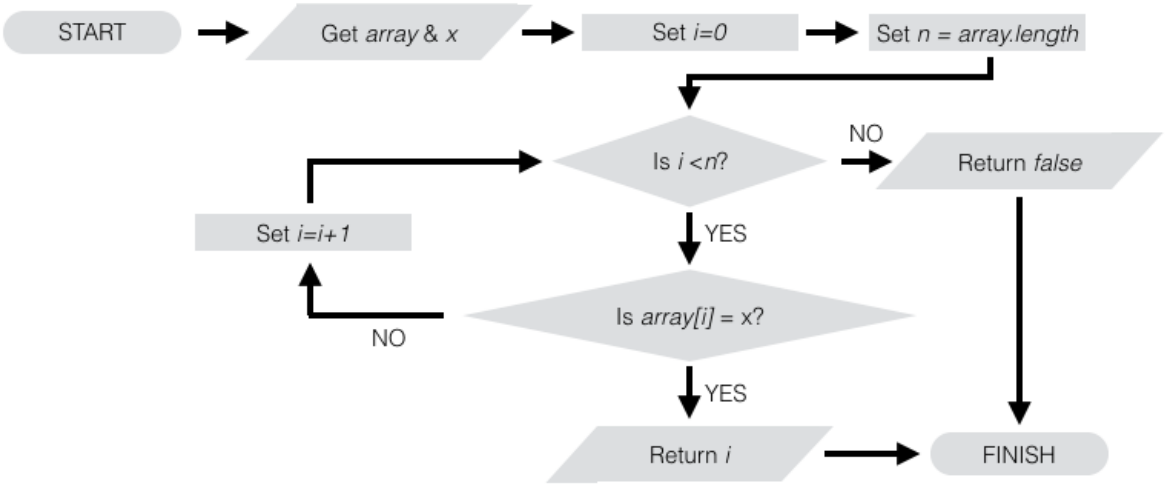
[4 marks]

Question (c): Consider the following piece of JavaScript:

```
1  var a = 2;
2  a = a ** 3;
3  function smallerThan(n) {
4    if (n < 10) {
5      return n;
6    } else {
7      return 10;
8    }
9  }
10 a = smallerThan(a);
11 a++;
12 a = smallerThan(a);
```

1. What is the value of `a` at the end of line 2? **[1 mark]**
2. What is returned by `smallerThan(9+3)`? **[1 mark]**
3. What is the value of `a` at the end of line 10? **[1 mark]**
4. What is the value of `a` at the end of line 11? **[1 mark]**
5. What is the value of `a` at the end of line 12? **[1 mark]**

Question (d): Consider the following flowchart:



Write a JavaScript function that implements the algorithm in this flowchart. You may name the function whatever you wish but it should take an array and a number as input parameters and return either a number or a Boolean.

[5 marks]

Question (e): In your coursework you were assigned tasks from 1 to 10. These tasks need to be completed in a specific order for the work to be completed correctly, with the first task being completed first, the second task being completed second and so on. The text of each task in your coursework can be stored as a string in memory, and then this string can be stored in an element of an abstract data structure: tasks will be stored if they are not yet completed, and removed when they are completed. Of the two following abstract data structures, which is the best way of storing the strings to make it easier to schedule your coursework tasks?

- A stack
- A queue

[2 marks]

Give a justification for your choice of abstract data structure; in doing so explain how data is added and removed from the abstract data structure in such a way that is suited to completing your coursework.

[3 marks]

Question (f): Consider the following array of integers:

```
var arr = [10,3,3,1,5,6];
```

You are tasked with algorithmically sorting the array `arr` so that the smallest value is in the first element and largest value is in the last element (ascending order). By hand, directly run through the Bubble Sort algorithm on `arr`. Show explicitly each step taken in the algorithm and what happens to the array.

[7 marks]

Question (g): Consider the following piece of JavaScript:

```
function Queue() {

  this.arr = [];

  this.head = function() {
    return this.arr[0];
  };

  this.dequeue = function() {
    if (this.arr.length == 0) {
      return "Queue underflow!";
    } else {
      return this.arr.shift();
    }
  };

  this.enqueue = function(o) {
    this.arr.push(o);
  };
}

queue = new Queue();
queue.enqueue(10);
queue.enqueue(5);
console.log(queue.arr);
```

1. What is printed in the console by this piece of code? **[2 marks]**
2. What is returned by the call `queue.head()`? **[2 marks]**
3. Write a method called `isEmpty` for this constructor that returns `true` if the object created by the constructor is empty, and `false` if it is not. **[3 marks]**
4. Write a function called `lengthQueue`, which takes a queue as argument, and returns the number of elements in the queue and only uses the methods of the queue, and leaves the queue as it was before the function is called. **[6 marks]**

Question (h): Consider the following piece of incomplete JavaScript:

```
function logBase2(n) {
  return (Math.log(n) / Math.log(2));
}

function logArray(input) {
  var array = [];
  for (var i = 0; i < input.length; i++) {
    MISSING
  }
  return array;
}
```

Here we have two functions that both take a number `n` as an input parameter (which is assumed to be an integer). The function `logBase2(n)` will compute the logarithm of a number `n` in base 2.

The function `logArray(input)` is supposed to return an array called `array` given an array `input` as an input parameter, where every element `array[i]` is equal to `logBase2(input[i])`. Answer the following:

1. What should go in the place of `MISSING`? **[2 marks]**
2. Write a JavaScript function called `expArray(input)` that takes an array called `input` as an input parameter and returns an array called `array` where every element `array[i]` is equal to 2 to the power of `input[i]`. **[5 marks]**
3. What is returned by `logArray([1,2,4,8])` once it is correctly completed? **[2 marks]**
4. What is returned by `expArray([0,1,2,3])` if `expArray(input)` is correctly implemented? **[2 marks]**
5. What is returned by `logArray(expArray([16,18,30,42]))` if `expArray(input)` and `logArray(input)` are both correctly implemented? **[2 marks]**

Consider the following piece of JavaScript, which will use the function `expArray(input)`:

```
function powers(n) {
  var array = [];
  for (var i = n; i >= 0; i--) {
    array[i] = n-i;
  }
  return expArray(array);
}
```

Given any integer in binary, we can convert it to decimal by multiplying each of the bitvalues by a relevant power of 2. For example, the bitstring 1001 can be turned into a decimal number by calculating

$$1 \times 2^3 + 0 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$

which is equal to 9. In the following part of the question we wish to convert a bitstring stored in an array, e.g. 1001 will be $[1, 0, 0, 1]$, into a number (float).

6. Write a function called `decConvert(array)`, which takes an array as an input parameter and returns a number: the array will represent a bitstring, and the number return will be the decimal number corresponding to that bitstring.

You may assume that `powers(n)` is already defined, and thus you may call it in your function (assuming also that `expArray(array)` is already defined).

[6 marks]