UNIVERSITY OF LONDON

GOLDSMITHS COLLEGE

Department of Computing

B. Sc. Examination 2019

IS51021B
Problem Solving for Computer Science

Duration: 2 hours 15 minutes

Date and time:

This paper is in two parts: part A and part B. You should answer ALL questions from part A and TWO questions from part B. Part A carries 40 marks, and each question from part B carries 30 marks. The marks for each part of a question are indicated at the end of the part in [.] *brackets.*

There are 100 marks available on this paper.

No calculators should be used.

THIS PAPER MUST NOT BE REMOVED FROM THE EXAMINATION ROOM

Part A

Attempt all parts of this question

Question 1 You should attempt an answer for all of these questions.

(a) Which of the following commands prints the value `5` to the console? [2]

i. `console.log(5 * 3);`

ii. `return 2 + 3;`

iii. `var x = 5;`

iv. `console.log(10 / 2);`

(b) Write a function in JavaScript called `isFive(n)` that takes a number `n` as an input parameter, and prints the string `"yes"` to the console if `n` is `5`, and prints the string `"no"` otherwise. [4]

(c) Consider the following piece of JavaScript:

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

i. What is the value of `x` at the end of line 1? [1]

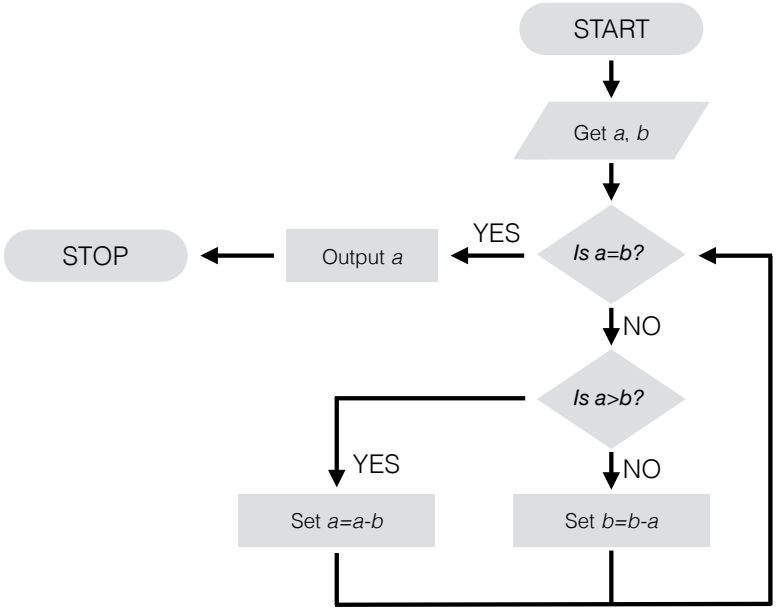
ii. What is the value of `x` at the end of line 2? [1]

iii. What is the value of `change(0)`? [1]

iv. What is the value of `x` at the end of line 11? [1]

v. What is the value of `x` at the end of the code? [1]

(d) Consider the following flowchart of the Euclidean algorithm:



Write a function in JavaScript that implements the algorithm in this flowchart. The function should take two input parameters and return a single number. [5]

(e) Consider the following piece of JavaScript:

```
1 function newFunction(n) {
2   if (n % 2 == 0) {
3     return false;
4   } else {
5     return true;
6   }
7 }
8 console.log(newFunction(3));
```

i. What is printed in the console after executing this code? [1]

ii. Write a new function called `secondFunction(x,y)` that will take two input parameters `x` and `y`, and returns `true` if `x + y` is perfectly divisible by 2, or `false` otherwise. You may assume that `newFunction(n)` is already defined and thus you may call it. [3]

(f) Consider the following piece of JavaScript:

```
1 var x = [1,9,8,4];
2 var a = x[0];
3 for (var i = 1; i < 4; i++) {
4   if (x[i] <= a){
5   a = x[i];
6   }
7 }
```

i. What is the value of `a` after this code is executed? [1]

ii. Is `a` the largest or smallest element value in the array `x`? Give a brief explanation why this is the case. [2]

iii. If `a` is to retain this property for an array `x` of arbitrary length with integers as elements, how would you amend line 3 of the code above? [2]

(g) Consider the following piece of JavaScript, which implements the function of summing together all integers from `1` to `n` for a given positive integer `n`.

```
1 function addTogether(n) {
2   var a = 0;
3   for (var i = 1; i <= n; i++){
4     a = a+i;
5   }
6   return a;
7 }
```

Write a new function that is a recursive implementation of this function in JavaScript, without using any form of iteration. It should take the same inputs as `addTogether(n)`, and produce the same output. [5]

(h) An extensible, sequential collection of elements where only two elements can be addressed: the head and the tail. Which of the following data structures satisfies this description?

i. A vector

ii. A queue

iii. A stack [2]

Describe how elements are added and removed from this data structure. [3]

(i) You are trying to perform a task on a large database of N elements, where N will tend to get very large. One colleague tells you that they have an algorithm (called algorithm **A**) that takes at most $6N^2 + N$ steps, another colleague says they found an algorithm (called algorithm **B**) that takes at most $18N \log_2(N) + 2N$ steps.

i. Give the worst-case run-times of algorithm **A** and algorithm **B** in terms of 'Big O' notation. [2]

ii. Which algorithm are you going to use? Explain your reasoning. [3]

Part B

Attempt two out of the three following questions

Question 2 This question is about the Fibonacci numbers: $0, 1, 1, 2, 3, 5, 8, 13, ...$, where each new number is obtained by adding the previous two in the sequence. The nth Fibonacci number is denoted $\mathbf{F}_n$, and thus $\mathbf{F}_n = \mathbf{F}_{n-1} + \mathbf{F}_{n-2}$. By convention the first two Fibonacci numbers are $\mathbf{F}_0 = 0$ and $\mathbf{F}_1 = 1$.

(a) Consider the following piece of JavaScript:

```
1  function fibonacci(n) {
2    var a = 0;
3    var b = 1;
4    if (n != 0) {
5      for (var i = 1; i <= n; i++) {
6        b = a + b;
7        a = b - a;
8        }
9    }
10   return a;
11 }
```

Complete the following table for the first five values of `n`. In the second column the output of the function `fibonacci(n)` should be entered, and in the third column enter the final value of `b` achieved within the body of the function.

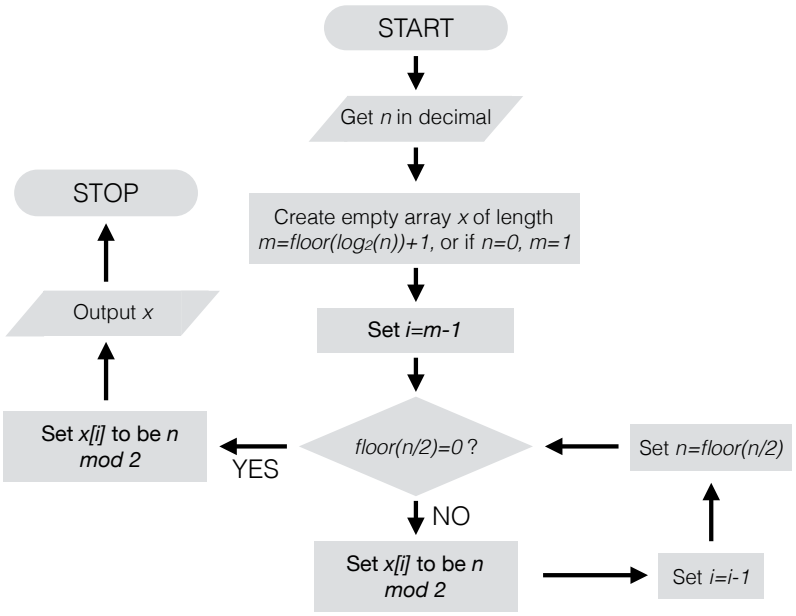
n	fibonacci(n)	b
0		
1		
2		
3		
4		

[6]

(b) Give a brief explanation why the number of operations in a standard implementation of `fibonacci(n)` is in the 'Big O' class $O(n)$ for the input n. [4]

(c) In a piece of JavaScript code, write a new recursive function called `recFibonacci(n)`, which takes the same input parameters and returns the same output as `fibonacci(n)` in part (a) of this question (assume `n` is a non-negative integer). This new function should not use any form of iteration. [6]

(d) Consider the following flowchart. This algorithm will produce an array of length $floor(\log_2(n)) + 1$ that gives the binary representation of the number n. In this flowchart, $floor(y)$ gives the largest integer smaller or equal to y.



The following piece of incomplete JavaScript code implements the algorithm in this flowchart. What should go in the place of `MISSING1`, `MISSING2`, and `MISSING3` to complete this code? [6]

```
1  function myFunction(n) {
2    if (MISSING1) {
3      var m = 1;
4    } else {
5      var m = Math.floor(Math.log2(n))+1;
6    }
7    var x = new Array(m);
8    var a = n;
9    var i = MISSING2;
10   while (Math.floor(a / 2) != 0) {
11     x[i] = a % 2;
12     i = i - 1;
13     a = MISSING3;
14   }
15   x[i] = a % 2;
16   return x;
17 }
```

(e) Recall that the function `fibonacci(n)` in part (a) of this question calculates the Fibonacci number $\mathbf{F}_n$. An equivalent way of defining the Fibonacci number $\mathbf{F}_n$ for $n > 2$ is as the number of binary strings (or bit-strings) of length `n-2` without consecutive `1`'s in the string. An example of a binary string without consecutive `1`'s is `1010`. For $0 \leq n \leq 2$, we just note the regular values of $\mathbf{F}_n$. The following function `notConsecutive(array)` takes an array as input, and will return a Boolean: it will return `true` if the array has elements with no consecutive `1`'s, and `false` otherwise.

```
1  function notConsecutive(array) {
2    var con = 0;
3    for (var k = 0; k < array.length - 1; k++) {
4      if ((array[k] == 1) && (array[k + 1] == 1)) {
5        con++;
6      }
7    }
8    if (con == 0) {
9      return true;
10   }
11   return false;
12 }
```

A new algorithm, called Algorithm **S** for calculating $\mathbf{F}_n$ is the following: for $n > 2$, one by one, generate all binary strings of length `n-2` and check if there are any consecutive `1`'s, and use this information to count the number of strings without consecutive `1`'s; for $0 \leq n \leq 2$, the algorithm will just output the Fibonacci numbers for those values of n.

Write a function in JavaScript called `algorithmS(n)` that implements the Algorithm **S**. Your function takes as input `n`, the integer `n` for $\mathbf{F}_n$, and returns an output that is equal to $\mathbf{F}_n$. HINT: you may assume that functions `myFunction(n)` and `notConsecutive(array)` are already defined (and completed) and you may call them within your function, if you need them. Also, note that the binary string of all-ones of length n corresponds to the number $2^n - 1$. [8]

Question 3 This question is about searching and sorting elements of arrays.

(a) Consider the following array of integers:

$$\text{A} = [3,7,4,3,9,2,11,1]$$

i. You are tasked with algorithmically searching the array A to see if it has an element with the value 1. Directly run through a `Binary Search` algorithm by hand on A. Show explicitly each step taken in the algorithm. [7]

ii. Does this implementation of the `Binary Search` algorithm find the value 1? Give the reason, in brief, for your answer. [2]

iii. What is the worst-case time complexity in `n` of the Binary Search algorithm for an array of length `n`? [3]

(b) Consider the following array of integers:

$$\text{B} = [4,8,2,5,9]$$

i. Implement the standard `Bubble Sort` algorithm on the array B, explicitly showing how the array changes in the algorithm. [7]

ii. What is the worst-case time complexity in `n` of the Bubble Sort algorithm for an array of length `n`? [3]

(c) Consider the following piece of JavaScript:

```
1  function goldSearch(arr, el) {
2    var left = 0;
3    var right = arr.length - 1;
4    while (left<=right) {
5      var fir = left + Math.floor((right - left) / 3);
6      var sec = left + Math.floor(2 * (right - left) / 3);
7      if ((arr[fir] == el) || (arr[sec] == el)) {
8        if (arr[fir] == el) {
9          return fir;
10       } else {
11         return sec;
12       }
13     } else if (el < arr[fir]) {
14       right = fir - 1;
15     } else if (arr[sec] < el) {
16       left = sec + 1;
17     } else {
18       left = fir + 1;
19       right = sec - 1;
20     }
21   }
22   return false;
23 }
```

One of your professors has proposed a new searching algorithm called the `GoldSearch` algorithm, which works for sorted arrays of numbers where the numbers go from lowest first to highest last. The JavaScript code above is the implementation of this algorithm.

i. In addition to the function `goldSearch(arr, el)`, we would like a function that tells us simply whether there is an element with the value `el`, or not. In JavaScript, write a function called `isThere(arr, el)`, which takes an array `arr` and value `el` as input parameters and prints `"yes"` or `"no"` to the console if there is an element with the value `el` or not, respectively. You may assume that `goldSearch(arr, el)` is already defined and so you can call it within your code. [5]

ii. Explain to your professor briefly why this algorithm will not be much better than the `Binary Search` algorithm. [3]

Question 4 Your bank will securely send you your new four-digit PIN in encrypted form. The bank previously told you how they generate the encrypted number from the PIN. This is how they do it: from left to right, take the PIN and convert each digit to the **factorial** (the definition of factorial is at the end of the question) of that number if it is greater than 0, and convert to 0 if the digit is 0.

Example: if the PIN is 1234, then the resulting encrypted (and longer) number is 12624, since 1 is converted to 1, 2 to 2, 3 to 6, and 4 to 24.

(a) i. What is the encrypted set of digits for the four-digit PIN 2424? [3]

ii. Write down a four-digit PIN whose encrypted digits are exactly the same as the original four-digit PIN. [3]

(b) Consider the following piece of incomplete JavaScript that does the task of converting the PIN, which is initiated as an array `pin` of four elements, into a string of digits corresponding to the encrypted digits. Recall that the method `toString()` will return a string, e.g. `n.toString()` converts a number `n` into a string and thus returns that string.

```
1  function factorial(n) {
2    var a = 1;
3    for (var i = 1; i <= n; i++) {
4      a = MISSING1;
5    }
6    return a;
7  }
8
9  function encrypt(pin) {
10   var out = "";
11   var b = 0;
12   if (pin.length != 4) {
13     return false;
14   }
15   for (var j = 0; j <= 3; j++) {
16     if (MISSING2) {
17       b = 0;
18     } else if (pin[j] > 9) {
19       return false;
20     } else {
21       b = factorial(pin[j]);
22     }
23     MISSING3;
24   }
25   return out;
26 }
```

i. What expression should go in the place of `MISSING1`? [2]

ii. What expression should go in the place of `MISSING2`? [2]

iii. What expression should go in the place of `MISSING3`? [2]

(c) The key component of this encryption scheme is the use of a factorial function. Instead of the function in part (b) with only one variable `a`, we can use a data structure to store multiple values of variables: a stack. Consider the following piece of incomplete JavaScript:

```
1  function Stack() {
2    this.arr = [];
3
4    this.pop = function() {
5      if (this.arr.length == 0) {
6              return "Underflow";
7      }
8      return this.arr.pop();
9    };
10
11   this.push = function(el) {
12     return this.arr.push(el);
13   };
14
15   this.peek = function() {
16     return this.arr[this.arr.length - 1];
17   };
18 }
19
20 function factorialStack(n) {
21   var f = new Stack();
22   f.push(1);
23   if (n > 1) {
24     for (var i = 2; i <= n; i++) {
25       f.push(i * f.peek());
26     }
27   }
28   var g = new Stack();
29   MISSING
30 }
```

The code involves a function `Stack()` that constructs a stack as an object when called, then a function `factorialStack(n)`. When completed, this second function should return a stack `g`, which is `f` but **with the order of the elements reversed**. That is, the top (bottom) element of `f` is the bottom (top) element of `g`.

i. Replace `MISSING` with code using the methods in the object created by the `Stack` constructor to create the correct `g`. There should be multiple lines of missing code. [5]

ii. The array associated with the stack `f` in part (c) for `factorialStack(9)` will be `[1, 2, 6, 24, 120, 720, 5040, 40320, 362880]`. What will be returned by `factorialStack(9).arr` once completed? [2]

iii. This stack `g` does not include 0! since this will not be useful for our decryption scheme. Outside of the function definition `factorialStack(n)`, what line of code should you add to push the encrypted digit for 0 to our desired stack? [3]

(d) An algorithm to find the original PIN from the encrypted digits is the following: one by one, generate all four-digit numbers, encrypt them, and then look at the resulting encrypted digits to see if they match the digits given by the bank.

In JavaScript, write a function called `findPIN(enc)` that carries out this algorithm. The function should take as input the encrypted digits `enc` as a string, and return the original four-digit PIN as a string. It should return `"Invalid input!"` if the input could not have resulted from a four-digit PIN. You may assume that the `Stack` constructor and the completed version of `factorialStack(n)` are defined, and you may call them. You may also assume that the solution to question 4c.(iii) has already been correctly implemented. [8]

Definition: The factorial of an integer n is written $n!$ and is defined as $n! = n \times (n-1) \times (n-2) \times ... \times 2 \times 1$, and $0! = 1$.