UNIVERSITY OF LONDON

GOLDSMITHS COLLEGE

Department of Computing

B. Sc. Examination 2020

IS51021B/C
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.

Calculators may be used in this examination; however, calculators which display graphics, text or algebraic equations are not allowed.

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. All questions apart from 1(d) are multiple choice. Give only one answer (or you will get zero marks).

(a) Which of the following is the JavaScript data type of the value `"Hello!"`? There is only one correct answer. [2]

i. Boolean
ii. String
iii. Number
iv. Undefined

(b) Which of the following commands prints a number to the console? There is only one correct answer. [2]

i. `return number;`
ii. `console.log("5" + 5);`
iii. `console.log(5 + 5);`
iv. `return 5 + 5;`

(c) Which of the following statements describes what an algorithm is? There is only one correct answer. [2]

i. An algorithm is only a method for checking if computer programs are correct.
ii. An algorithm is a method of step-by-step basic instructions which, if followed, solve a problem.
iii. An algorithm is only a method for finding if a number is an integer.
iv. An algorithm is a set of problems that can be solved by a computer.

(d) Consider the following piece of JavaScript:

```
function thing(n) {
  var x = 0;
  for (var i = 1; i < n; i++) {
    if (n / i == 1) {
      x++;
    }
  }
}
```

Which of the following is returned by the function call `thing(2)`? There is only one correct answer. [2]

i. `1`
ii. `undefined`
iii. `0`

(e) Consider the following piece of JavaScript:

```
 1  var x = 0;
 2  x = x + 3;
 3
 4  function biggerThan(n) {
 5    if (n > 2) {
 6      n++;
 7    }
 8    return n;
 9  }
10
11  x = biggerThan(x);
12
13  var thing = {
14                propOne: x,
15                propTwo: biggerThan(x),
16                meth: function() {
17                  return this.propOne + this.propTwo;
18                }
19  };
```

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 returned by `biggerThan(0)`? [2]

iv. What is the value of `x` at the end of line 11? [2]

v. What is the value of `thing.propTwo` at the end of this code? [2]

vi. What is the value returned by `thing.meth()` at the end of this code? [2]

(f) Consider the following piece of JavaScript code:

```
var array = [];
array.push(0);
array.push(3);
console.log(array[1]);
```

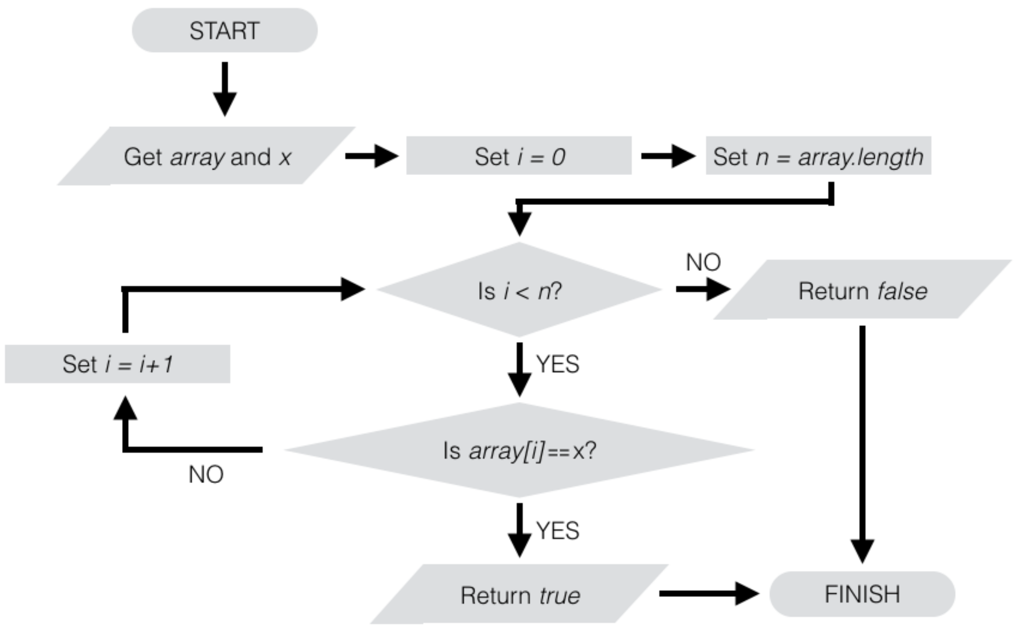
Which of the following is printed to the console by this code? There is only one correct answer. [2]

i. `0`

ii. `1`

iii. `3`

(g) Consider the following flowchart:



Which of the following describes the kind of algorithm this is? There is only one correct answer. [1]

i. A Search Algorithm

ii. A Sorting Algorithm

(h) Consider the following piece of incomplete JavaScript:

```
function partF(array, x) {
  var n = array.length;
  for (MISSING) {
    if (array[i] == x) {
      return true;
    }
  }
  return false;
}
```

This JavaScript code should implement the flowchart from question 1(g). Which of the following should go in the place of `MISSING`? [2]

i. `var i = 0; i == n; i++`

ii. `var i = 0; i < n; i++`

iii. `var i = 0; i < n - 1; i++`

(i) From where is an element removed from a queue? Choose only one option. [3]

i. tail

ii. top

iii. index

iv. head

(j) Which of the following describes the operation that adds a new element with the value o to a queue? Choose only one option. [3]

i. pop![o]

ii. enqueue![o]

iii. dequeue![o]

iv. push![o]

(k) Consider the following piece of JavaScript code:

```
function factorial(n) {
  if (n==0) {
    return 1;
  }
  return n * factorial(n);
}
```

This code is supposed to return the factorial of an integer `n`. For which of the following reasons will it not do this correctly? There is only one correct answer. [3]

i. The base case is incorrect

ii. The function will not return anything for `n` equal to 0

iii. The function does not return anything for all arguments `n`

iv. There will be infinite recursion for `n` not equal to 0

(l) Consider the following piece of incomplete JavaScript for a function `sum(n)`, which takes the number `n` as an argument, and should return the sum of all integers from `0` to `n`.

```
function sum(n) {
  if (n == 0) {
    return 0;
  }
  return MISSING;
}
```

Which of the following is the correct expression to replace `MISSING`? [4]

i. `sum(n-1)`

ii. `sum(n)`

iii. `n + sum(n-1)`

iv. `n + sum(n)`

(m) You have a database of n elements, where n will tend to get very large. There is an algorithm (called algorithm **A**) that takes at most $n^4 + 200n$ time-steps to complete a task, and another algorithm (called algorithm **B**) that takes at most $2^n + n^{100} + n^2$ steps to complete the same task.

Which of the following is the worst-case time complexity of algorithm **A**? Choose only one option. [2]

i. O(n^3)

ii. O(n^4)

iii. O(n)

Which of the following is the worst-case time complexity of algorithm **B**? Choose only one option. [2]

i. O(n^2)

ii. O(n^{100})

iii. O(2^n)

Part B

Attempt two out of the three following questions

Question 2 This question is about sorting and worst-case time complexity.

(a) From the point of view of searching an array, briefly explain why it is better if the array is already sorted. [3]

(b) Consider the following array of integers:

```
var arr = [1, 5, 7, 2, 4]
```

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]

(c) What is the worst-case time complexity of the Bubble Sort algorithm for an array of length n? [3]

(d) Give an example of a worst-case input array for the Bubble Sort algorithm of length 4 storing integers. Do not explain why it is a worst-case input (assume sorting by ascending order). [4]

(e) Another sorting algorithm is Quicksort.

i. What is the worst-case time complexity of Quicksort? [3]

ii. Briefly explain why Quicksort is preferred over Bubble Sort by making direct reference to average-case time complexity. HINT - the average-case and worst-case time complexity of Bubble Sort is the same. [2]

(f) You work for a website called 'NastyVegetables.com' that is making a list of the most popular movies from 2019 based on public votes. Two votes were taken at different times in 2020 and you need to combine the results.

You are given two unsorted arrays called `firstVote` and `secondVote`, both of length `n` with each element storing a two-element array of the form:

```
["Average Movie 2", 167],
```

where the name of the movie is stored in the first element and the number of votes it got is stored in the second element. Each movie appears only once in each of the two arrays.

You are given the following piece of JavaScript code:

```javascript
function makeOneList(firstVote, secondVote) {
  var n = firstVote.length;
  var finalList = [];
  var votes = [];
  for (var i = 0; i < n; i++) {
    votes = [];
    for (var j = 0; j < n; j++) {
      if (secondVote[j][0] == firstVote[i][0]) {
        votes.push(firstVote[i]);
        votes.push(firstVote[i][1] + secondVote[j][1]);
      }
    }
    finalList.push(votes);
  }
  return finalList;
}
```

This code will produce an array `finalList` storing the total number of votes obtained by the `n` movies. It does this by comparing elements of `firstVote` with `secondVote` to see if they have the same movie title, pushes that title to an array called `votes` with the total number of votes it received.

Briefly give an argument why the worst-case time complexity of the algorithm implemented by the function `makeOneList` is $O(\mathtt{n}^2)$ for two arrays both of length `n`. [4]

(g) Now imagine the situation where the arrays `firstVote` and `secondVote` are first sorted in alphabetical order according to the name of the movie. In this case, we have that `secondVote[i][0]` is equal to `firstVote[i][0]` for all `i`.

Consider the following incomplete piece of JavaScript:

```javascript
function sortedOneList(firstVote, secondVote) {
  var n = firstVote.length;
  var finalList = [];
  var votes = [];
  for (var i = 0; i < n; i++) {
    votes = [];
    MISSING;
    finalList.push(votes);
  }
  return finalList;
}
```

Write down the code that should go in the place of `MISSING` so that the worst-case time complexity of implementing `sortedOneList` is $O(\mathtt{n})$. There can be multiple lines of code. [4]

Question 3 This question involves Binary Search and determining if a number has an integer square root.

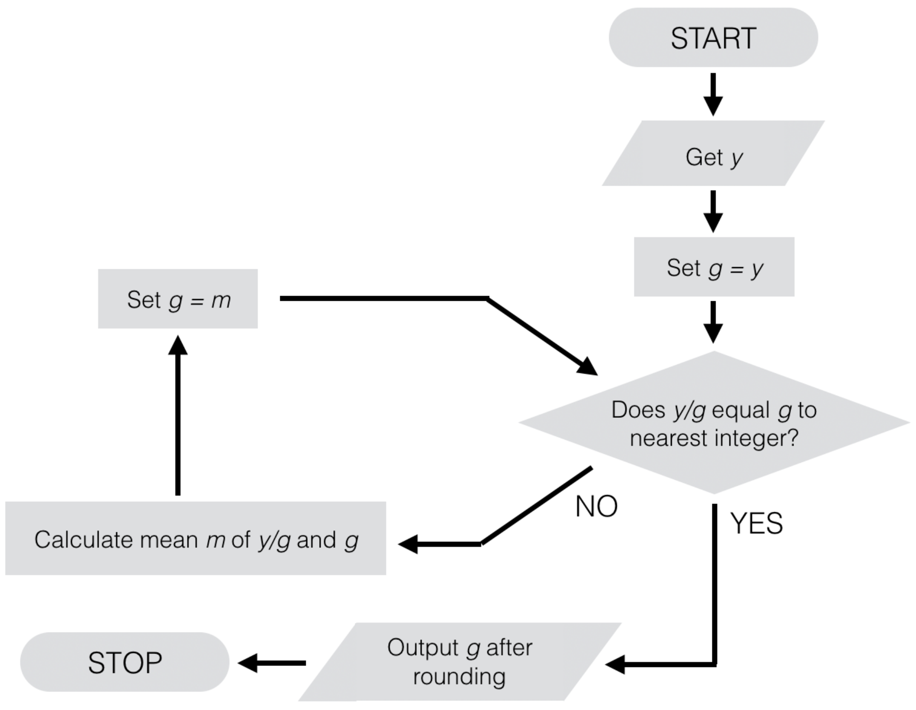
(a) Consider the following array of integers, which are the squares of the integers from 0 to 5:

```
var s = [0, 1, 4, 9, 16, 25];
```

You are tasked with algorithmically searching the array `s` to see if it has an element with the value 5. Directly run through the Binary Search algorithm by hand on `s`. Show explicitly each step taken in the algorithm. [7]

(b) What is the worst-case time complexity in n of the Binary Search algorithm for a sorted array of length n? [3]

(c) In the following we will use Heron's method to calculate the square root of an integer y rounded to the nearest integer. The following flowchart describes this method:


START
Get *y*
Set *g* = *y*
Does *y/g* equal *g* to nearest integer?
NO
YES
Calculate mean *m* of *y/g* and *g*
Set *g* = *m*
Output *g* after rounding
STOP

Here the mean m of two numbers y/g and g is equal to $\frac{1}{2}(g + y/g)$. Rounding is done to the nearest integer. Also, "Does y/g equal g to nearest integer?" is the same as checking if the rounded versions of y/g and g are equal.

Write a function in JavaScript called `heron(y)` that takes a number `y` as an argument and returns the output from implementing Heron's method. The function should implement Heron's method as shown in the above flowchart. HINT - `Math.round(x)` will round the number `x` to the nearest integer. [8]

(d) Consider the following problem:

Given that $x^2 = n$, where n is an integer, is x an integer?

One solution method, called **Method A**, for solving this problem is to calculate the square root of n rounded to the nearest integer, and then check that this gives n when squared. If it does give n, then x is an integer, otherwise it is not.

In a standard implementation of this solution method, it can be shown that the number T of time-steps required is not bigger than

$$T \leq \log_2(\log_2(1 + 2\sqrt{n})) + \log_2(\sqrt{n}) + 50.$$

Using this information, what is the worst-case time complexity of implementing **Method A**? [4]

(e) Another solution method, called **Method B**, to solve the problem in part(d) of this question is implemented in the following JavaScript code:

```
function isSquareRootInt(n) {
  var left = 0;
  var right = n;

  while (left <= right) {
    var mid = Math.floor((left + right) / 2);
    if (mid * mid == n) {
      return true;
    } else if (mid * mid < n) {
      left = m + 1;
    } else {
      right = m - 1;
    }
  }

  return false;
}
```

Give an argument why the worst-case time complexity (in a standard implementation) of **Method B** is $O(\log n)$. HINT - the following identity should be useful: $\log(n + 1) \leq \log(2n) = \log 2 + \log n$. [5]

(f) Of the two solution methods, **Method A** and **Method B**, presented above, from the point-of-view of worst-case time complexity, which should you use? Very briefly explain your reasoning. [3]

Question 4 This question is about checking if an array is a palindrome. An array is a palindrome if it is exactly the same after the order of the elements is reversed (the array is flipped). For example, `[0, 1, 2, 1, 0]`, `[1]` and `[]` are all palindromes.

(a) Given an array, one way to check that it is a palindrome is to create a new array that is the flipped version of the original. Consider the following array:

```
var array = [3,2,4,5,2,3];
```

i. What is the flipped version of this array? [2]

ii. Is `array` a palindrome? [1]

(b) Consider the following incomplete JavaScript code:

```
function flippedArray(array) {
  var n = array.length;
  var flip = [];
  for (var i = n - 1; MISSING; i--) {
    flip.push(array[i]);
  }
  return flip;
}
```

This function `flippedArray` should return an array called `flip`, which is the flipped version of the argument `array`. Which of the following should replace `MISSING` to correctly complete this function? [2]

i. `i == 0`

ii. `i >= 0`

iii. `i < 0`

iv. `i > 0`

(c) To determine algorithmically if an array is a palindrome, once the flipped array is obtained, the two arrays are then compared to see if they are equal. Consider the following incomplete JavaScript code:

```
function areEqual(array1, array2) {
  if (array1.length != array2.length) {
    return false;
  }
  for (var i = 0; i < MISSING1; i++) {
    if (array1[i] !== array2[i]) {
      return MISSING2;
    }
  }
  return true;
}
```

This function should return `true` if the argument arrays `array1` and `array2` are exactly equal, and return `false` otherwise. What should go in the place of both `MISSING1` and `MISSING2` to correctly complete this function? [4]

(d) Write a function in JavaScript called `isPalindrome(array)` that takes an array called `array` as an argument, and returns `true` if `array` is a palindrome, and `false` otherwise. You can assume the functions `flippedArray` and `areEqual` are already defined and correctly completed, and you should use them. [4]

(e) Another way to determine if an array is a palindrome is by using a stack. In the first step, all of the values in the array are pushed to the stack from left to right.

Consider the array in question 4(a). Start at the leftmost element and going along to the right, push every value to the stack. Draw a picture of what the stack looks like one all values are pushed to it. Indicate which element is the top of the stack. [4]

(f) Consider the following piece of JavaScript:

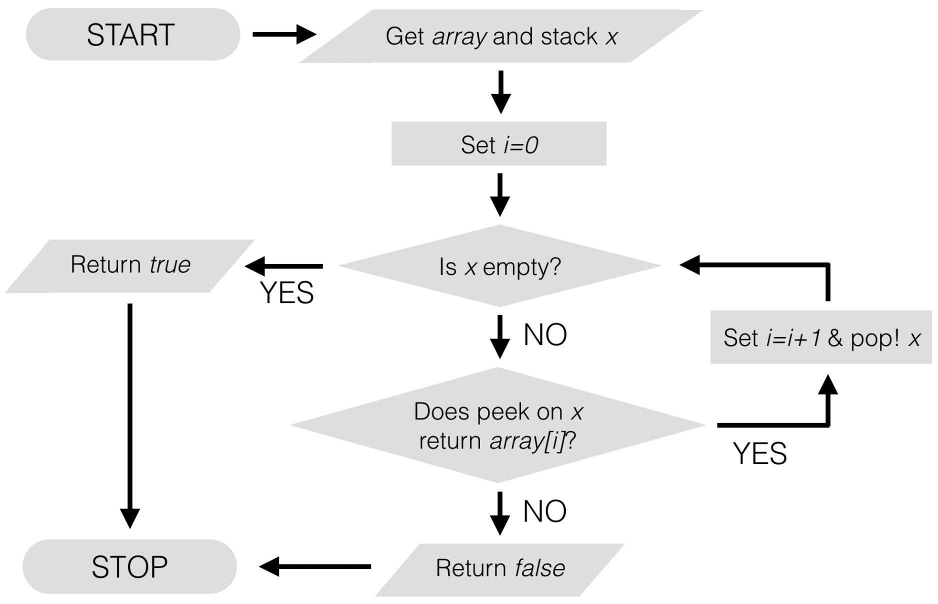
```javascript
function Stack() {
  this.arr = [];
  this.pop = function() {
    if (this.arr.length == 0) {
            return "Underflow";
    }
    return this.arr.pop();
  };
  this.push = function(el) {
    return this.arr.push(el);
  };
  this.peek = function() {
    return this.arr[this.arr.length - 1];
  };
  this.isEmpty = function() {
    return (this.arr.length == 0);
  };
}

function arrayStack(array) {
  var f = new Stack();
  var n = array.length;
  for (var i = 0; i < n; i++) {
    f.push(array[i]);
  }
  return f;
}
```

This code contains the constructor for a stack and a function that pushes values stored in an array to a stack.

i. What is returned by `arrayStack([1,2,3]).peek()`? [2]

(g) Consider the following flowchart:



Write a function in JavaScript called `stackPalindrome(array, x)` that takes an array called `array` and stack called `x` as arguments, and returns a Boolean. The function should implement the algorithm described in the flowchart. You can assume that the constructor for the stack is defined and refer to it, and to obtain full marks, use the methods associated with the stack object. [8]

(h) Write a function in JavaScript called `isPalStack(array)` that takes an array called `array` as an argument, and returns `true` if `array` is a palindrome, and `false` otherwise. This function should call only `stackPalindrome(array, x)` and `arrayStack(array)`, and no other functions. [3]