UNIVERSITY OF CAMBRIDGE

COMPUTER SCIENCE TRIPOS Part IA

NATURAL SCIENCES TRIPOS Part IA (Paper CS/1)

Monday 1 June 2020 1.30 to 4.30

COMPUTER SCIENCE Paper 1

*Answer **one** question from each of Sections A, B and C, and **two** questions from Section D.*

*Submit the answers in five **separate** bundles, each with its own cover sheet. On each cover sheet, write the numbers of **all** attempted questions, and circle the number of the question attached.*

You may not start to read the questions printed on the subsequent pages of this question paper until instructed that you may do so by the Invigilator

STATIONERY REQUIREMENTS	SPECIAL REQUIREMENTS
Script paper	*Approved calculator permitted*
Blue cover sheets	
Tags	

SECTION A

1 Foundations of Computer Science

You need to write OCaml code to help a local park ranger count the different types of trees present in a region of Cambridgeshire woodland.

(*a*) Define an OCaml type `tree` that can distinguish between an *oak*, *birch* or *maple* tree, and also any other species with an arbitrary `string` name.

[1 mark]

(*b*) Define two OCaml values with the following signatures:

(*i*) `val describe : tree -> string`
that accepts a `tree` parameter and returns a human-readable `string`.

(*ii*) `val identify : string -> tree`
that accepts a lowercase `string` parameter and returns a `tree`.

Explain briefly how the OCaml compiler can statically check if you have handled all the input possibilities for the input parameters to `describe` and `identify`.

[4 marks]

(*c*) Define a type `stree` that only distinguishes between three species *oak*, *birch* or *maple* and no others. Implement functions for the following signatures with similar functionality to the earlier `identify` function:

```
val identify_exn : string -> stree
val identify_opt : string -> stree option
```

Briefly discuss the tradeoffs between your two approaches. [5 marks]

(*d*) You now need to implement a simple simulator before starting real surveys. Trees will occur in the following fixed infinite sequence: *oak*, *birch*, *oak*, *maple*, *maple*, and then repeat from the beginning.

(*i*) Define a function `val spotter : unit -> stree` that will return the sequence of trees when called multiple times. [5 marks]

(*ii*) Define a purely functional alternative `spotter` that calculates the next `stree` in sequence, using only the input arguments to the function to calculate the return value. Write down an example application of this function with the input arguments and the expected output result.*(Hint: you may need to pass in the complete sequence as one of the arguments.)*
[5 marks]

2 Foundations of Computer Science

In this exercise, we will develop a game engine to play a simplified version of the game of *Mastermind*.

In *simplified Mastermind*, player A selects a list of n colours among 3 possible colours: `Red`, `Green` and `Blue` (e.g., [`Red`; `Red`; `Green`; `Blue`] if $n = 4$). Player B has to guess player A's list of colours by proposing lists of colours in sequence until she finds the list proposed by player A. Every time player B proposes a list of colours, she gets feedback in the form of a number x. (x is the number of colours that are in the correct position). For example, if player A's list is [`Red`; `Red`; `Green`; `Blue`] and player B guessed [`Red`; `Green`; `Green`; `Red`], then $x = 2$ (the first `Red` and second `Green` are at the right positions). Note that $x \leq n$.

(*a*) Define a type `colour` to represent a colour. [2 marks]

(*b*) Given two colour lists, write a function `feedback` that returns x. The first argument `a` is the list of player A, and the second argument `b` is a list of player B. Raise a `SizeMismatch` exception if the lengths of both lists do not match. You may need to introduce a helper function. [4 marks]

(*c*) Using currying, define a `test` function that takes a list proposed by player B and returns x. This function should assume that player A's list is [`Blue`; `Green`; `Red`]. [2 marks]

(*d*) What is the type of `test` in Part (*c*)? [2 marks]

(*e*) Write a function `generate_lists` that generates all possible colour lists of a given length n. The function takes a single argument n. You may use the concatenation operation `@` and `List.map` function. Tip: `generate 2` should output $3^2 = 9$ lists. [6 marks]

(*f*) Given a colour list of player B and a feedback x, write a function `valid_lists` that takes two arguments b and x and returns all possible lists that player A could have chosen (such that they match the feedback given to player B). You may use `generate_lists`, `feedback`, `List.length` and/or `List.filter`. [4 marks]

SECTION B

3 Object-Oriented Programming

You have been asked to implement a user-interface component `ButtonCanvas` which is both a button that can be clicked by the user and a canvas which the user can draw on.

Your existing code contains a `Button` class and a `Canvas` class. Both of these extend `GuiComponent`.

(*a*) `ButtonCanvas` could be built using *multiple inheritance*. What does this mean in this context? [2 marks]

(*b*) Give two reasons why this might be desirable. [2 marks]

(*c*) Give two complexities that arise in this case. [2 marks]

(*d*) Java interfaces originally contained only *abstract methods* and *static final* fields. How did this restriction avoid the complexities of extending multiple classes? [3 marks]

(*e*) Draw a UML diagram for building `ButtonCanvas` using abstract interfaces rather than multiple inheritance. Explain how your design attempts to preserve the desirable properties arising from multiple inheritance.

You do not need to recall the exact UML specification, instead you may provide a key explaining your notation. [9 marks]

(*f*) Recent versions of Java added default methods to interfaces. What is the impact of this with respect to multiple inheritance? [2 marks]

4 Object-Oriented Programming

(*a*) Describe the differences between *primitive types* and objects in Java. Consider:

(*i*) the values they contain [1 mark]

(*ii*) where they are stored in memory [1 mark]

(*iii*) how they interact with Java *references* [1 mark]

(*b*) What are *auto-boxing* and *auto-unboxing*? Give an example of how they might cause an exception to be thrown. [4 marks]

(*c*) Consider the following code in which any arbitrary Java type (primitive or object) could be substituted for `T`.

```
void f(T t) { /* ... */ }

T t1 = /* ... */
f(t1);
```

For which substitutions of `T` can we guarantee that the value in `t1` is unchanged after the invocation of `f(t1)`? Justify your answer. [3 marks]

(*d*) Explain how Java's implementation of generics precludes substituting `T` with a primitive type. [2 marks]

(*e*) You are asked to redesign the standard library to incorporate an *immutable* list. Explain the relative merits of:

(*i*) `MutableList` being a subtype of `ImmutableList` [2 marks]

(*ii*) `ImmutableList` being a subtype of `MutableList` [2 marks]

(*iii*) `ImmutableList` and `MutableList` having no common supertype [2 marks]

(*iv*) `ImmutableList` and `MutableList` both subtyping `CommonList` [2 marks]

SECTION C

5 Introduction to Probability

(*a*) You are looking forward to the European Cup 2020. The tournament is going to last for 30 days. Each day during the tournament, you want to invite all of your 100 classmates to your house. But people might be busy on any given day, so you expect each classmate to come with probability 0.03 on each day, and they show up independently of one another. Let X denote the number of classmates that show up on any given day.
Note: In parts (*ii*), (*iii*) and (*iv*) you do **not** have to compute explicit numerical values.

(*i*) Give the exact and approximate distributions of X along with parameters. Explain why the approximation is valid. [3 marks]

(*ii*) What is the approximate probability that between 2 and 4 classmates, inclusive, show up on any given day? [3 marks]

(*iii*) What is the exact probability that at least 2 classmates show up on any given day? [3 marks]

(*iv*) What is the probability that there will be more than 27 days where at least 2 classmates show up? [3 marks]

(*b*) Suppose that each classmate is asked to arrive at 8pm, but the actual arrival time differs in minutes by a continuous uniform distribution $[-\theta, +\theta]$, where θ is an unknown parameter. Your data set $Y_1, Y_2, \ldots, Y_k$ is a realisation of independent random samples from that distribution, for some integer $k \geq 1$.

(*i*) Show that
$$T = \frac{3}{k} \cdot (Y_1^2 + Y_2^2 + \cdots + Y_k^2)$$
is an unbiased estimator for θ^2. [3 marks]

(*ii*) Is $\sqrt{T}$ also an unbiased estimator for θ? [1 mark]

(*iii*) Justify your answer in (*b*)(*ii*). [4 marks]

6 Introduction to Probability

(*a*) Give the probability mass function of each of the three distributions:

(*i*) Poisson distribution, [2 marks]

(*ii*) Bernoulli distribution, [2 marks]

(*iii*) Binomial distribution. [2 marks]

(*b*) The football association has asked you to analyse the England team football matches from previous big tournaments. For each of the three situations below, choose a suitable distribution and compute its expectation and variance.
Note: In Part (*b*)(*ii*) you do **not** have to compute explicit numerical values.

(*i*) You analyse 2000 penalty kicks from the last 10 years of big tournaments. It turns out that 1200 of those 2000 penalty kicks were goals. A penalty kick is chosen at random. Let X be a success if a goal is scored. [2 marks]

(*ii*) Consider again the setting from (*b*)(*i*). If you pick 50 penalty kicks without replacement, let Y be the number of missed goals out of that sample. [2 marks]

(*iii*) Taking into account all games from the last 10 years of big tournaments, the England football team scored an average of 1 goal every 30 minutes. Let Z be the number of scored goals during a match of 90 minutes. [2 marks]

(*c*) Consider the following table displaying the joint distribution of two random variables X and Y.

	x			
y	-1	0	$+1$	$\mathbf{P}[Y = y]$
-1	??	??	0	$\frac{1}{4}$
0	$\frac{1}{4}$	??	??	$\frac{1}{2}$
$+1$	??	$\frac{1}{4}$	??	$\frac{1}{4}$
$\mathbf{P}[X = x]$	$\frac{1}{4}$	$\frac{1}{2}$	??	

(*i*) Complete the table above. [2 marks]

(*ii*) Compute $\mathbf{E}[X]$ and $\mathbf{E}[Y]$. [2 marks]

(*iii*) Define independence of two discrete random variables. Are X and Y given above independent? Justify your answer. [2 marks]

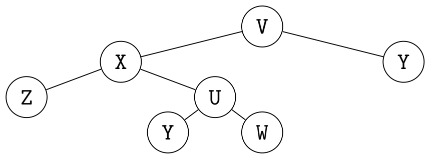
(*iv*) Define the covariance of two random variables. What is the covariance between X and Y given above? Justify your answer. [2 marks]

SECTION D

7 Algorithms

Consider binary trees whose nodes have three fields: key (a single character between `U` and `Z`), left subtree, right subtree, with the two subtrees either both empty or both non-empty. Assuming suitable constructors, indicate an empty tree as `T()` and a non-empty tree as `T(key, leftSubtree, rightSubtree)`.

(*a*) Define unambiguous pre-order, in-order and post-order representations for such a tree `t`, called $\texttt{r}_{pre}$`(t)`, $\texttt{r}_{in}$`(t)`, $\texttt{r}_{post}$`(t)`, consisting of strings over the {`U`, `V`, `W`, `X`, `Y`, `Z`, `(`, `)`} alphabet, with balanced brackets, with three characters (two brackets and a letter) for each node, starting and finishing with a bracket unless `t` is empty. Formally describe your three representations for a generic tree `t`, then produce the corresponding strings for the following tree. [6 marks]



(*b*) In this obfuscated pseudocode, the input `v0` is a syntactically correct $\texttt{r}_{post}$`(t)`. Clearly explain (*i*) the purpose of the code; (*ii*) how it works; and (*iii*) how one should invoke it. Substitute meaningful explanatory identifiers for those $\texttt{T}_x$, $\texttt{v}_y$ and $\texttt{m}_z$. Identifiers `v4` and `v6` are worth more marks than the others. [7 marks]

```
0   class T1:
1       # data members of objects of type T1
2       v2: T14 of T3
3       v4: T12
4       v6: T13
5
6       def m5(v0):
7           if v0 is "":
8               return T3()
9           else:
10              for v9 in v0:
11                  m10(v9)
12              return v2.pop()
13
14      def m10(v11):
15          if v11 is one of {"U", "V", "W", "X", "Y", "Z"}:
16              v6 = v4 is ")"
17          else if v11 is ")":
18              if v6:
19                  v8 = v2.pop(); v7 = v2.pop()
20              else:
21                  v8 = null; v7 = null
22              v2.push(T3(v4, v7, v8))
23          v4 = v11
```

(*c*) Explain in detail (*i*) how line 16 works, and (*ii*) why `v4` will never be uninitialised when line 16 is executed. [4 marks]

(*d*) Write clear pseudocode that takes a Tree `t` and returns $\texttt{r}_{in}$`(t)`. [3 marks]

8 Algorithms

Reminder:

$$f(n) \in \Theta(g(n))$$

$$\iff$$

$$\exists\ n_0, c_1, c_2 \in \mathbb{R}_{>0} \text{ such that } \forall n > n_0 : 0 < c_1 g(n) \leq f(n) \leq c_2 g(n).$$

(*a*) For the bubblesort algorithm, state its best-case Θ complexity and describe, for any given input of arbitrary size n, one permutation that would trigger this best-case behaviour. Then give the corresponding permutation of `[0, 1, 2, 3, 4, 5, 6, 7, 7, 7]`. Then repeat the above for the worst case: state the Θ complexity, saying when it is achieved, and exhibit as a concrete example a permutation of the 10 numbers given. [4 marks]

(*b*) Repeat Part (*a*) for the heapsort algorithm. [4 marks]

(*c*) Repeat Part (*a*) for the basic quicksort algorithm, where the pivot is simply chosen as the last element in the range. [4 marks]

(*d*) Write clear and efficient pseudocode to eliminate all duplicates from a linked list of n elements, without changing the order of the remaining elements. Then derive and justify its Θ complexity. [4 marks]

(*e*) [*Hint:* The Ω notation, like the Θ notation, is typically used to describe the asymptotic behaviour of a worst-case cost function $f(n)$. When we say, by extension, that a certain task has a complexity bound of $\Omega(g(n))$, we mean that this bound applies to the worst-case cost function of every possible algorithm that could solve that task.]

Give a formula for $f(n) \in \Omega(g(n))$, in a format similar to that of the Reminder above, and briefly explain it. Then derive, with a clear justification, a tight Ω complexity bound for the task of eliminating all duplicates from a linked list of n elements. [4 marks]

9 Algorithms

We are given a directed graph $g = (V, E)$. A vertex $v \in V$ is said to be an *origin* if for any other vertex $w \in V$ there is a directed path from v to w.

(a) Consider the `dfs_recurse(`g`,`s`)` algorithm as described in lecture notes. Show carefully that, once it terminates, if it has visited a vertex v then it has also visited all vertices reachable from v. [4 marks]

(b) Suppose g has an origin. Give an algorithm that returns an origin, and which has $O(V + E)$ running time. [*Hint:* Consider `dfs_recurse_all(`g`)`. What happens after it visits an origin?] [5 marks]

(c) Suppose g has an origin. Prove that the vertex returned by your algorithm in part (b) is indeed an origin. [6 marks]

(d) Give an algorithm that returns *all* origins, and which has $O(V + E)$ running time. If the graph has no origins, your algorithm should return an empty set. Explain briefly why your algorithm is correct. [5 marks]

10 Algorithms

This question concerns unbalanced binary search trees. In some scenarios, most searches are for a small interval in the key space. For example, if we are searching a social media feed for posts keyed by timestamp, perhaps 90% of searches might be for the newest 5% of posts.

(*a*) Explain why, in a balanced binary search tree with N items, the worst-case cost of searching for an item is $\Theta(\log N)$. [1 mark]

(*b*) Define *amortized cost*. [2 marks]

(*c*) Consider an unbalanced binary search tree with N items, where the root holds a dummy key, the left subtree holds $(1-\alpha)N$ items, the right subtree holds αN items, and where each subtree is balanced. Call this an α-`BALANCED` tree.

Suppose there are m searches, $(1-\beta)m$ for items on the left and βm for items on the right. Find the amortized worst-case cost of a search. For $\alpha = 5\%$ and $\beta = 90\%$, would you prefer α-`BALANCED` or a fully balanced tree? [5 marks]

(*d*) Describe the *weighted union* heuristic and the *path compression* heuristic, for the Disjoint Set data structure. Explain what is meant by a *rotation* of a binary search tree. [6 marks]

(*e*) We speculate that for real-world search frequencies it might be useful to have an binary search tree that is unbalanced not just at the root but also at other levels. We would also like the data structure to adjust its shape automatically, as searches occur.

By adapting the two heuristics above, suggest a suitable data structure. [*Note:* You do not need to give detailed pseudocode, but you should explain how you are adapting the heuristics.] [6 marks]

END OF PAPER