CST0
COMPUTER SCIENCE TRIPOS Part IA

Tuesday 4 June 2024 09:00 to 12:00

COMPUTER SCIENCE Paper 1

*Answer **one** question from each of Sections A, B, C, D, and E.*

*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
Script paper
Blue cover sheets
Tags

SPECIAL REQUIREMENTS
Approved calculator permitted

SECTION A

1 Foundations of Computer Science

The chief examiner of a Tripos course would like to do some statistical analysis of the results of an exam. There are 150 students who have taken an exam consisting of 10 questions, of which the student has answered 6. If a student has not attempted a question, this is indicated by a zero in the list. The data is provided as follows:

```
type marks = int list
let results : marks list = [
  [ 30; 25; 20; 0; 0; 18; 30; 0; 0; 8 ];
  [ 27; 0; 18; 9; 0; 30; 28; 0; 0; 17 ];
  [ 10; 18; 0; 7; 0; 29; 25; 0; 1; 0 ];
  (*... 147 more rows ...*) ]
```

The mean $\bar{x}$ and standard deviation s of a `marks` value can be calculated as follows.

$$\bar{x} = \frac{1}{n}\sum_{i=0}^{n-1} x_i \qquad\qquad s = \sqrt{\frac{\sum_{i=0}^{n-1}(x_i - \bar{x})^2}{n-1}}$$

(a) Define functions `fold` that applies a function in order over a list to return a value, `map` that applies a function to each element of a list, and `filter` that removes items not satisfying a given predicate from a list.

```
val fold: ('a -> 'b -> 'a) -> 'a -> 'b list -> 'a
val map: ('a -> 'b) -> 'a list -> 'b list
val filter: ('a -> bool) -> 'a list -> 'a list
```

[6 marks]

(b) The examiner evaluates performance by taking a `marks` value, filtering out the zeros, and then calculating mean and standard deviation. Write a function that does this and returns an appropriate type that you have defined. Remember that the result is not defined for some `marks` values. You may assume that two predefined functions `val sqrt : float -> float` and `val float_of_int : int -> float` are available in your answer. [8 marks]

(c) The examiner wants to assess the exam itself by looking at the mean and standard deviation for each question. Define an auxiliary function `nth` that returns the `n`th item of a list, and write two functions to calculate this value.

```
val nth : int -> 'a list -> 'a option
val qmean : int -> marks list -> float
val qstd : int -> marks list -> float
```

[6 marks]

2 Foundations of Computer Science

(*a*) A prime number is a natural number greater than 1 that has no positive divisors other than 1 and itself. We wish to implement a primality test in OCaml that checks if a positive input integer is prime. A simple primality test is via trial division: given an positive input number n, check if it is divisible by any prime number between 2 and $\sqrt{n}$. For any divisor $p \geq \sqrt{n}$, there must be another divisor $n/p \leq \sqrt{n}$, and a prime divisor q of n/p, and therefore looking for prime divisors where $p \leq \sqrt{n}$ is sufficient.

Define a function `is_prime` which accepts a positive input integer and returns a boolean to indicate if it is prime or not. To simplify your code,you can avoid calculating square roots by checking for prime divisors where $p^2 \leq n$. You can assume the existence of a `(mod)` operator which returns the integer remainder of two integers. For example, `3 mod 2` will return `1`. The type definitions are:

```
val (mod): int -> int -> int
val is_prime : int -> bool
```

[8 marks]

(*b*) In functional programming, *fold* functions are higher order functions that process data structure elements in order and build a return value.

(*i*) `fold_range a b f acc` is a specialised integer fold that applies $f(n)$ where $b \leq n \leq a$, with initial value `acc`. For example, `fold_range 1 3 (+) 10` would return `16` (from `10+1+2+3`). Define this function with the type:

```
val fold_range: int -> int -> (int -> 'a -> 'a) -> 'a -> 'a
```

[4 marks]

(*ii*) `fold f acc l` is a more generic function that applies `f` over a list `l` with initial value `acc`. Define this function with the type:

```
val fold: ('a -> 'b -> 'a) -> 'a -> 'b list -> 'a
```

[2 marks]

(*iii*) Explain the time and space complexity of your implementation of `fold` and briefly discuss whether it is tail recursive or not. [3 marks]

(*c*) `all_primes a b` is a function that returns all of the valid prime numbers between `a` and `b` inclusively. Define this function and its associated type, using the previous definitions of `fold_range` and `is_prime`. [3 marks]

SECTION B

3 Object-Oriented Programming

(*a*) Explain the effect of the access modifiers `public`, `protected`, `private`, or default (no modifier) in Java. [4 marks]

(*b*) Consider the statement "public state of a Java class should be final and static"

(*i*) Explain why this is considered good programming practice. [4 marks]

(*ii*) Give one example of a member variable that should *not* be declared `public`, even if `final` and `static`. [3 marks]

(*c*) Consider a language identical to Java except that all class member variables are `private`. Class methods can still be `public`, `protected`, `private`, or default. The rationale for this change is that access can be provided via methods and so this simplifies the language. Compare and contrast this approach with the Java approach. [4 marks]

(*d*) The Python programming language does not have explicit private access modifiers in its classes. Instead all variables and methods are public and a *convention* is used whereby names prefixed with an underscore are to be considered hidden despite there being no enforcement of this by the compiler. Compare this to Java's explicit access modifier approach. [5 marks]

4 Object-Oriented Programming

A university manages its students using a program that has a class `Student` with subclasses `FirstYear`, `SecondYear`, and `ThirdYear` for year-specific state and behaviour. The program has a `List<Student>` that contains all `Student`s.

(*a*) Should `Student` be a class, an abstract class or an interface? Explain your answer. [2 marks]

(*b*) Write a `Comparator` that can be used to sort the `List<Student>` by year group and then by name, both ascending, and show how it would be used. You should assume the existence of a `String getName()` method within `Student`. [4 marks]

(*c*) At the end of each year, the third year students graduate and must be removed. This is done by passing the list to the following method:

```
void removeThirdYears(List<Student> students) {
    for (Student student : students) {
      try {
        ThirdYear thirdyear =(ThirdYear) student;
        students.remove(thirdyear);
      }
      catch(ClassCastException cce) { }
    }
}
```

(*i*) What will happen when the call to `remove()` is made? Explain why and fix the code. [4 marks]

(*ii*) Comment on this use of exceptions and propose an alternative that does not rely on them. [3 marks]

Full marks required correct use of an iterator or equivalent. Copying the list did not earn the mark for the solution.

(*d*) Also at the end of the academic year, the first and second year students move up a year.

(*i*) Explain why this class design makes this problematic. [3 marks]

(*ii*) Propose an alternative design and explain in detail how it addresses the problems you identified in (*d*)(*i*). [4 marks]

SECTION C

5 Introduction to Probability

General comment: For any of the answers below, unless stated otherwise, you do not need to give specific numerical values.

A company produces a bundle of 20 USB sticks. Each USB stick is broken (i.e., malfunctioning) with probability 1/500, independently.

(*a*) Let Z be the number of broken USB sticks within one bundle. What is the distribution of Z? Also state the expectation and variance. [3 marks]

(*b*) Let p be the probability that there are at least two broken USB sticks in a bundle. Determine p. [2 marks]

(*c*) Now let p be the probability that there are exactly five broken USB sticks in a bundle. Describe a suitable method for approximating the value of p and state the result. [3 marks]

(*d*) Consider a bundle of 20 sticks that has exactly 2 broken USB sticks. If someone takes out 3 different USB sticks chosen randomly, what is the probability that exactly one is broken? [3 marks]

(*e*) Suppose a retailer purchases a bundle and inspects each of the 20 USB sticks. If there are at least two broken USB sticks, the retailer asks the company for a new bundle which is delivered on the next day, and the process continues. What is the distribution of the number of days until the retailer has obtained a bundle with at most one broken USB stick? Also state the expectation of that distribution. [3 marks]

Consider now two producers A and B, each selling the same bundle but for different prices. For producer A, the price is $X \sim 2 + \mathsf{Uni}(1, 2)$, and for producer B, the price is $Y \sim 3 + \mathsf{Uni}(0, 1.5)$ (here $\mathsf{Uni}(a, b)$ refers to the uniform continuous random variable with range $[a, b]$).

(*f*) What are $\mathbf{E}[X]$ and $\mathbf{E}[Y]$? [2 marks]

(*g*) Assume that X and Y are independent random variables. What is $\mathbf{P}[X \leq Y]$? (For full marks, complete your computation to obtain a specific numerical value.) [4 marks]

6 Introduction to Probability

The weight of a female elephant is $F \sim N(\mu_F, \sigma_F^2)$ with $\mu_F = 10$, $\sigma_F^2 = 2$, and the weight of a male elephant is $M \sim N(\mu_M, \sigma_M^2)$ with $\mu_M = 15$, $\sigma_M^2 = 4$.

(*a*) Use Chebhyshev's inequality to bound the probability $\mathbf{P}[\, F \leq 5\,]$. [4 marks]

(*b*) Compute $\mathbf{P}[\, M \geq 19\,]$.
[*Hint:* You do not need to give a numerical answer; it suffices if you express your answer in terms of $\Phi(x)$ for a suitable value of x.] [3 marks]

(*c*) Consider the weight difference $D := M - F$. What is the distribution of D? Also state the expectation and variance of D. [4 marks]

(*d*) Suppose that the height of a female elephant has an unknown distribution with finite mean μ and finite variance σ^2. You are given a sample of heights $(2, 3, 5, 2)$. How would you estimate μ and σ^2? [3 marks]

Now let Z be a continuous random variable with the following cumulative distribution function (CDF):

$$F_Z(x) = \begin{cases} 0 & \text{if } x < 0, \\ \frac{1}{2}x^2 & \text{if } 0 \leq x < 1, \\ \frac{1}{2} - \frac{x^2-4x+3}{2} & \text{if } 1 \leq x \leq 2, \\ 1 & \text{if } x > 2. \end{cases}$$

(*e*) Determine the probability density function (PDF) f_Z. [3 marks]

(*f*) Using your answer in (*e*), compute the expectation of $\mathbf{E}[\, Z\,]$. [3 marks]

SECTION D

7 Algorithms 1

Given a sequence `s` of items from a domain U with a total order $<$, we derive a *subsequence* of `s` by dropping zero or more items from `s`. A *descending* sequence is one in which every item except the first is strictly smaller than its predecessor.

[*Note:* In this question, we use 0-indexing and Python slice notation to indicate parts of a sequence, and so should you in your answer. For example, if `s` is `[A, B, C, D]`, then `s[0]` is `A`, `s[1:3]` is `[B, C]`, `s[:3]` is `[A, B, C]`, `s[3:]` is `D`, `s[4:]` is `[]` and `s[-1]` is `D`.]

We seek an efficient algorithm that, given a sequence `s` of length n, returns a descending subsequence `t` of maximal length.

(a) Discuss the following statement and prove that it is incorrect. [2 marks]

> This problem exhibits *optimal substructure*, meaning that the optimal solution may be expressed in terms of optimal solutions to subproblems, and may therefore be solved by dynamic programming. If we split sequence `s` into two parts, any optimal solution (a maximally long descending subsequence of `s`) must consist of an optimal solution to the first part concatenated with an optimal solution to the second part.

(b) Prove that the following statement is also incorrect. [3 marks]

> Any optimal solution for `s[:k]` must start with an optimal solution for `s[:k-1]`, possibly with `s[k-1]` appended to it if this combination still is a descending sequence.

(c) Give a correct recursive description and formula for the optimal solution in terms of optimal solutions to subproblems, with clear justification. The recursive description and formula must solve the given problem directly, as opposed to reducing it to another problem. [*Note:* The optimal solution is a subsequence, not the length of a subsequence.] [5 marks]

(d) Clearly explain how to solve the problem through the alternative approach of reducing it to one of the well-known dynamic programming problems studied in the lecture course, giving a correct recursive description and formula for the optimal solution to the well-known problem. [5 marks]

(e) Give clear pseudocode to solve the problem using the approach in Part (d), using recursive memoized top-down dynamic programming. [5 marks]

8 Algorithms 1

(*a*) An object has a method `gulp(x)`, which accepts a 64-bit floating point number. On its n-th invocation since the creation of the object, this method returns the median of the n numbers fed to the object up to that point. Can this method be implemented with worst-case asymptotic cost of $O(1)$ in both time and space (with respect to n)? Provide either clear pseudocode or a clear explanation of impossibility. [6 marks]

[*Note:* The median of $m > 0$ values is defined as follows. Imagine the m values have been sorted, and are indexed from 0 to $m - 1$. If m is odd, the median is the value at position $\frac{m-1}{2}$. Otherwise, it is the average of the two values closest to the midpoint, i.e. those at positions $\frac{m}{2} - 1$ and $\frac{m}{2}$.]

(*b*) Given an unsorted singly-linked list of n integers, each of which painted in one of the seven colours of the rainbow, we seek an algorithm of optimal worst-case asymptotic complexity for rearranging the list so that all the integers of the same colour are adjacent (forming a contiguous *region* of consecutive list elements with the same colour), and the coloured regions appear one after the other in the same order as in the rainbow, and the integers within each region retain the same relative order as they did in the original list.

(*i*) Give a clear, concise and complete description of your strategy. [6 marks]

(*ii*) Give clear pseudocode implementing your strategy from Part (*b*)(*i*). Details of pointer manipulation are required for full marks. [6 marks]

[*Note:* Please use the following notation: `l.head` = pointer to head node of list `l`; `n.colour` = colour of node `n`; `n.value` = integer value of node `n`; `n.next` = pointer to the successor of node `n` in the corresponding list; and so forth, defining and using new fields as appropriate.]

(*iii*) Derive the worst-case asymptotic complexity of your approach and prove it is optimal. [2 marks]

SECTION E

9 Algorithms 2

We are given a directed graph with edge costs; let $c(u \to v) > 0$ be the cost of edge $u \to v$. We are also given a start vertex s and an end vertex t, and we assume that t is reachable from s. An edge is said to be a *bottleneck* if increasing its cost results in an increase in the distance from s to t, and it is said to be an *opportunity* if decreasing its cost results in a decrease in that distance.

(a) For each of the following claims, either prove it or provide a counterexample:

(i) The graph must have an opportunity;

(ii) The graph must have a bottleneck;

(iii) All bottlenecks are opportunities.

[6 marks]

(b) For each edge $u \to v$, define the relaxed cost to be $c'(u \to v) = c(u \to v) + d_u - d_v$, where d_u is the distance from s to u, and d_v the distance from s to v. Prove that all opportunities have relaxed cost equal to zero. [4 marks]

(c) Give an algorithm for computing all opportunities in a graph g. Your algorithm should have $O(E + V \log V)$ running time, where E is the number of edges in g and V the number of vertices. Prove that your algorithm is correct, and explain why it has the desired running time. [10 marks]

10 Algorithms 2

Consider a hash table implemented with open addressing. This question is concerned with how its underlying storage array is to be managed in order to efficiently support a dynamic collection of items—in particular, when it is to be resized and its contents rehashed.

Let the underlying storage array have capacity c, and let it consist of n slots containing values, and d slots with a `DELETED` marker, plus empty slots. For performance reasons we wish to maintain $n + d \leq \gamma c$, where $\gamma = 0.75$ is the maximum allowed load factor. The `set` and `delete` operations then take $O(1)$ time to update the array contents; they might also trigger a rehash which takes $O(n + d)$.

(*a*) Define *potential function*. Given a potential function, explain how one can use it to obtain amortized costs. [3 marks]

(*b*) When should the rehashing be triggered, and what should the new size be? Give a policy for which `set` and `delete` have amortized cost $O(1)$. Using a potential function, or otherwise, justify your answer. [7 marks]

For memory efficiency reasons we don't want to store too many `DELETED` markers: we wish to maintain $c \leq (1 + \delta)n$, where $\delta = 3$ is the maximum allowed memory overhead.

(*c*) Modify your procedure (if necessary) to support this memory constraint in addition to the load factor constraint. Justify why `set` and `delete` have amortized cost $O(1)$. [*Hint:* Suppose the table is rehashed, and then there is a sequence of deletions, triggering a second rehash. What's the smallest number of deletions needed to trigger the second rehash?] [10 marks]

END OF PAPER