Computer Science Tripos Part IA and IB

2019-2020
Exam Question
Cover sheet

Student BGN2023A**…..……**

Paper1**……**

Question number1**……**

How did you answer this question?

Timed	☐	Open Book	☑
Untimed	☑	Closed Book	☐

Questions

List all the questions you have answered for this paper here. 1,4,6,8,9

Computer Science Tripos Honour Code

1. We take it as a principle that maintaining the integrity and fairness of examinations should be regarded as a collaboration between students and the Department.
2. The students undertake that they will not help others in examinations and will not receive any help from others (students or non-students).
3. Students will actively contribute to ensuring that all students adhere to the code.
4. Students will keep to the conditions of the assessment and will accurately report those conditions when asked.
5. The Department will not make any attempt at remote invigilation of online examinations.

I **undertake to respect the Computer Science Tripos honour code**

Tick the box to confirm ☑

1a)

```
type tree =
    | Oak
    | Birch
    | Maple
    | Species of string
;;
```

b) (i)

```
let describe t =
    match t with
    | Oak -> "oak"
    | Birch -> "birch"
    | Maple -> "maple"
    | Species (name) -> name
;;
```

(ii)

```
let identify s =
    match s with
    | "oak" -> Oak
    | "birch" -> Birch
    | "maple" -> Maple
    | _ -> Species (s)
;;
```

For the `describe` function, the compiler needs to check that all inputs of type `tree` and handled. Oak, birch, and maple can just be checked one by one. `Species` must be handled for all string values, but this can be checked easily since the pattern in the function matches all strings.

For the `identify` function, the compiler needs to check that all string inputs are handled. This can be easily checked since the wildcard pattern is used which will match any string.

c)

```
type stree =
    | Oak
    | Birch
    | Maple
;;

exception UnknownTree
let identify_exn s =
    match s with
    | "oak" -> Oak
    | "birch" -> Birch
    | "maple" -> Maple
    | _ -> raise UnknownTree
```

```
;;

let identify_opt s =
    match s with
    | "oak" -> Some (Oak)
    | "birch" -> Some (Birch)
    | "maple" -> Some (Maple)
    | _ -> None
;;
```

Throwing an exception is a good idea if the program does not expect a new type of tree ever, i.e. the program is somehow broken if a new tree type is passed to the identifier.

If the program is expecting new types of trees, but doesn't want to do anything with them, then the `option` version is more suitable.

d) (i)

```
let spotter =
    let seq = [|Oak; Birch; Oak; Maple; Maple|]
    and index = ref 4
    in let get () =
        index := (!index + 1) mod 5;
        Array.get seq !index
    in
    get
;;
```

(ii)

```
let spotter =
    let seq = [Oak; Birch; Oak; Maple; Maple]
    in let get completeSeq =
        let rec helpGet input matchSeq =
            match (input, matchSeq) with
            | (x::xs, y::ys) -> helpGet xs ys
            | ([], y::ys) -> y
            | (xs, []) -> helpGet xs seq
        in
        helpGet completeSeq seq
    in
    get
;;
```

```
let s = spotter;;

s [Oak; Birch; Oak; Maple; Maple; Oak];;
(* Output: Birch *)
```