COL100M Major

Aman Godara

TOTAL POINTS

30 / 40

QUESTION 1

5 pts

1.1 1 / 1

✓ **+ 1 pts** **Correct**

- 0.5 pts Incorrect / missing type

+ 0 pts Incorrect

1.2 1 / 1

✓ **+ 1 pts** **Correct**

- 0.5 pts Incorrect / missing type

+ 0 pts Incorrect

1.3 1 / 1

✓ **+ 1 pts** **Correct**

- 0.5 pts Incorrect / missing type

- 0.5 pts Incorrect / Missing output

+ 0 pts Incorrect

1.4 1 / 1

✓ **+ 1 pts** **Correct**

+ 0.5 pts Reason of the error not mentioned

+ 0 pts Incorrect

1.5 1 / 1

✓ **+ 1 pts** **Correct**

+ 0 pts Incorrect

QUESTION 2

5 pts

2.1 1 / 1

✓ **+ 1 pts** **Correct**

+ 0.5 pts Only one of Input type, output type correct

+ 0 pts Incorrect

2.2 2 / 2

✓ **+ 2 pts** **Correct**

+ 0.5 pts Says type 'a list / float list/list

+ 1 pts Only one of input type, output type correct

+ 0 pts Incorrect

2.3 2 / 2

✓ **+ 2 pts** **Correct**

+ 1 pts Only one of input type, output type correct

+ 0 pts Incorrect

QUESTION 3

3 1 / 2

+ 2 pts Correct

✓ **+ 1 pts** **Only one of g1 and g2 correctly identified**

+ 0 pts Incorrect

no explanantion for g2.

QUESTION 4

3 pts

4.1 2 / 2

✓ **+ 2 pts** **Correct**

+ 0 pts Incorrect

4.2 1 / 1

✓ **+ 1 pts** **Correct**

+ 0 pts Incorrect

QUESTION 5

5 3 / 3

✓ **+ 1 pts** **Inorder Correct**

✓ **+ 1 pts** **Preorder Correct**

✓ **+ 1 pts** **Postorder Correct**

+ 0 pts Incorrect

QUESTION 6

6 1 / 1

✓ **+ 1 pts Correct**

+ 0.5 pts one of them is correct

+ 0 pts Incorrect / not attempted

QUESTION 7

7 1 / 1

✓ **+ 1 pts Correct**

+ 0 pts Incorrect / not attempted

+ 0.5 pts one of them is correct

QUESTION 8

8 3 / 3

✓ **+ 3 pts Correct**

+ 0.5 pts Root level Correct

+ 1 pts Second level Correct

+ 1.5 pts Leaf level Correct

+ 0 pts Incorrect / not attempted

QUESTION 9

4 pts

9.1 1 / 1

✓ **+ 1 pts Correct**

+ 0 pts Incorrect / not attempted

9.2 0 / 3

+ 3 pts Correct

+ 1 pts Correct till first eval expression

✓ **+ 0 pts Incorrect / not attempted**

refer to correct solution

QUESTION 10

10 3 / 3

✓ **+ 0.5 pts Base case Correct**

✓ **+ 1 pts Inductive hypothesis correct**

✓ **+ 1.5 pts Inductive step correct**

+ 0 pts Incorrect / not attempted

+ 1 pts Inductive step partially correct

- 0.5 pts Did not use induction hypothesis for proof

- 0.5 pts Imprecise

QUESTION 11

11 1.5 / 3

✓ **+ 1 pts Order Correct**

+ 1 pts Proof f1 < f2 correct

+ 1 pts Proof f2 < f3 correct

+ 1 pts proved f1 < f3

+ 0 pts Incorrect / not attempted

✓ **+ 0.5 pts Proof without explicitly simplifying; just stating c and n0**

QUESTION 12

7 pts

12.1 2 / 2

✓ **+ 2 pts Correct**

+ 1.5 pts Mentions sorted, but sorted list incorrect

+ 0 pts Incorrect / not attempted

12.2 0.5 / 5

+ 5 pts Correct

+ 2 pts Recognised O(n) complexity of the recursive calls in the 'match' expression. A clear argument should be made as to why the complexity is O(n).

+ 2 pts Recognised O(n^2) complexity of the recursive calls in the outer expression.

+ 0.5 pts Recognised O(n) complexity of the recursive calls in the outer expression and derived O(n^2).

+ 1 pts Put the above together, to derive O(n^3)

+ 0.5 pts Written a correct recurrence

+ 0 pts Incorrect or not attempted

- 1 pts Unclear explanation, but correct idea.

- 1 pts Final run time complexity not shown

+ 0.5 pts Identified that worst case occurs when the list is sorted in decreasing order.

+ 1 pts Mentions time complexity as O(n^3) (when list comparision is O(1)) or O(n^4) (when list comparision is O(n)) with some explanation about different cases of calls to the function f.

+ 1 pts Run time O(n^3) or O(n^4) correct with sketchy proof.

+ 0.5 pts Run time O(n^2) with sketchy proof.

+ 0.5 Point adjustment

NO! "f l(n)" does not call "f l(n-1)" n times, instead, it only calls it atmost 3 times. This is not easy to observe. Please have a look at the sample solution.

COL100 – Major Exam

May 4, 2018

NAME:

2017 TT 10876 Aman Godara

ID:

2017 TT 10876

1 Instructions

1. Write your answers *only in the space provided.* We are using software which automatically detects the answer region. If you write in the margins or in the wrong space, *your answer will not be graded.*
2. This exam requires you to write mathematical proofs and derivations. As repeatedly stressed in class, please write *precise* and *complete* answers. Anything short of this will result in loss of marks.
3. No calculators, phones, notes, or other resources are allowed. This is a closed book exam.
4. Time allocated for the exam: *2 hrs.*

2 Exam begins here

1. Write the output and its type for the following OCaml expressions. (*Note:* If a specific value is returned, write the value. Otherwise, state what the expression computes. If it outputs an error, then state that and briefly explain why.)

   (a) **(1 mark)** `x + y`

   x → int , y → int
   output → int , expression adds x and y

   (b) **(1 mark)** `"s" ^ "r"`

   s → 'a , r → 'a
   output → string , expression contatenates two strings
   output → "sr"

   (c) **(1 mark)** `true || false`

   true → bool
   false → bool
   output → bool
   output → true

(d) **(1 mark)** `1 :: 2`

error
as 2 is not a list, expression cannot add 1 to 1 2

(c) **(1 mark)** Write the type of x.

```
type tp = {a: int; b: string}
let x = {a = 1; b = "x"}
```

x belongs to type tp

2. Write the function signature for the following functions.

(a) **(1 mark)** `let f () = print_string "hello world"`

unit → unit

(b) **(2 marks)** `let f x y = (x+y) :: []`

int → int → int list

(c) **(2 marks)** `let f x y z = (x+y,z^" in the world")`

int → int → string → int * string

3. **(2 marks)** What do the following two functions do (explain briefly in English)? *Note:* g1 and g2 are *mutually recursive functions*, therefore, they have both been defined simultaneously using the and keyword. abs x returns the absolute value of x.

```
let rec g1 x =
   if x = 0 then true
   else g2 (abs(x)-1)
and g2 x =
   if x = 0 then false
   else g1(abs(x)-1)
```

g1 tells whether the given input is even or not, for even it gives true else false
g1(0) → true g(-2), g(2) → true

4. The following function takes in an array as input. Answer the following questions.

```
let f ar =
   let i = ref 0 in
   let j = ref (Array.length ar - 1) in
   while !i < !j do
     let tmp = ar.(!i) in
     ar.(!i) <- ar.(!j);
     ar.(!j) <- tmp;
     incr i;
     decr j
   done
```

(a) **(2 marks)** What is the output of the function, if the input is `[|4;2;6;7;1;5|]`?

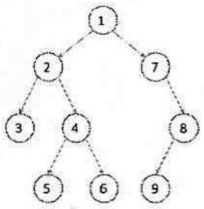
[|5; 1; 7; 6; 2; 4|]

(b) **(1 mark)** For the input array given in (a), how many times is the condition `!i < !j` checked in the while loop?

total = 3 + 1 = 4

3 → while loop executes, 1 → detected that now !i ✗ !j

5. **(3 marks)** Given the following binary tree, write the sequence of integers visited in a inorder, preorder and postorder traversal of the tree. *Note:* Label the traverals correctly.



inorder → 3 2 5 4 6 1 7 9 8

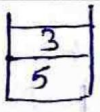
preorder → 1 2 3 4 5 6 7 8 9

postorder → 3 5 6 4 2 9 8 7 1

(7) → has no left subtree

(8) → has no right subtree

6. **(1 mark)** Given an empty stack `S` of integers, show what the stack contains after the following operations (assume that all operations are being applied to `S`): `push (5)`, `push(4)`, `push(1)`, `pop()`, `top()`, `pop()`, `push(3)`, `top()`. Clearly mark the bottom and top of the stack.



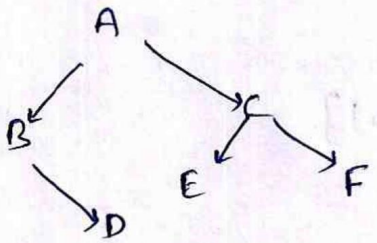
3 → top

5 → bottom

7. **(1 marks)** Given an empty queue `Q` of integers, show what the queue contains after the following operations (assume that all operations are being applied to `Q`: `insert(1)`, `delete()`, `insert(2)`, `insert(6)`, `delete()`, `insert(5)`. Clearly mark the front and back of the queue.

6 → front of queue
5 → back of queue

8. **(3 marks)** Given that a binary tree has the following traversals, reconstruct the tree. Preorder: `A,B,D,C,E,F`, Inorder: `B,D,A,E,C,F`, Postorder: `D,B,E,F,C,A`



9. Recall the evaluation of postfix expressions from your lab assignment using stacks. Given the following postfix expression: `1 3 2 4 + * -`, answer the following questions:

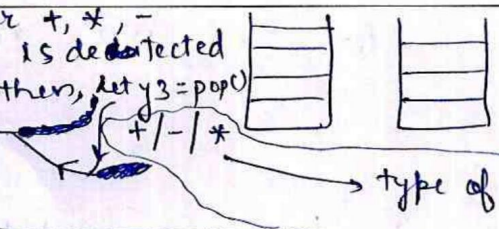
   (a) **(1 mark)** Write the infix expression corresponding to this postfix expression.

   1 - 3 * 2 + 4 (1 - (3 * (2+4)))

   (b) **(3 marks)** Write the *sequence of operations* that occur when you use a stack to implement a program to evaluate the postfix expression. *Note*: Your answer should contain a sequence of the following form: `push(X)`, `Y = pop ()`, `Z = eval (expr)`, etc., where `X` is an element, and `expr` is a mathematical expression such as `Y * Y` that is evaluated by the function `eval`.

① let y1 = pop()
let y2 = pop()
push(eval(y1 y3 y2))

whenever +, *, - is detected
then, let y3 = pop()



first we will go from left to right whenever +/-/* is detected we do this process.

10. **(3 marks)** Show by induction that

$$\sum_{k=1}^{n} \frac{1}{k(k+1)} = \frac{n}{n+1}$$

basic step:

$f(n) = \sum_{k=1}^{n} \frac{1}{k(k+1)}$

$f(1) = \sum_{1}^{1} \frac{1}{k(k+1)}$

$= \frac{1}{(1)(2)} = \frac{1}{2}$

$\frac{n}{n+1}$ at $n = 1$ is equal to $= \frac{1}{1+1} = \frac{1}{2}$

Inductive step:

Assume $f(p)$ gives correct output $= \sum_{k=1}^{p} \frac{1}{k(k+1)}$

$= \frac{p}{p+1}$

($p \geq 1$)

To prove: $f(p+1)$ gives correct output

$f_{p+1}(p+1) = \sum_{k=1}^{p+1} \frac{1}{k(k+1)} = \sum_{k=1}^{p} \frac{1}{k(k+1)} + \frac{1}{(p+1)(p+2)}$

$= \frac{p}{p+1} + \frac{1}{(p+1)(p+2)}$

$= \frac{1}{(p+1)} \left(\frac{p(p+2)+1}{(p+2)} \right) = \frac{p+1}{p+2}$

$\frac{n}{n+1}$ at $p+1$ is equal to $\frac{p+1}{p+2}$

$\therefore f(p+1)$ gives correct answer

by Principle of mathematical induction:

$f(n) = \frac{n}{n+1}$

11. **(3 marks)** Arrange the following functions in increasing orders of growth and explain why the order is correct:

$$f1 = \log_2 n, f2 = 2^{\sqrt{\log_2 n}}, f3 = n^{\frac{1}{3}}$$

(*Note:* You essentially need to determine whether $f(n) = O(g(n))$ for the given functions and give the required constants.)

f3 > f2 > f1 (f3 will grow fastest f1 slowest)
as $\log_2 n$ is increasing let it be p
$f_1 = p$ $f_2 = 2^{\sqrt{p}}$ $f_3 = (2)^{(p/3)}$
and we know 2^n is increasing function and growth of $2^n > n$ and growth of $\frac{n}{3} > \sqrt{n}$ thus we concluded answer.
growth of $2^{\sqrt{n}} > n$ and $2^{n/3} > n$
$2^{\log_2 n} = n$ and we know ~~log n~~ growth of $\log_2 n < \sqrt{n}$ thus $2^{\sqrt{n}} > n$.

12. Consider the following OCaml code and answer the questions below.

```
let rec f l =
  let r =
  match l with
  | x :: y :: t ->
     if x > y then
        y :: f (x :: t)
     else x :: f (y :: t)
  | _ -> l
  in
  if l = r then l
  else f r
```

(a) **(2 marks)** What is the output of `f [4;2;6;7;1;5]`?

[1; 2; 4; 5; 6; 7]

(b) **(5 marks)** Derive the runtime complexity of the implementation. (*Hint:* You may use the recursion tree method, but that alone will not give you the complete solution.)

f takes in input a list of size (n)
i will denote as such
→ f l(n)

~~f l(n) level 0~~ ~~total no. of calls = $\sum_{i=0}^{n-1} 1$ = n~~

↓

~~f l(n-1) level 1~~ f l(1) does not call back f(2)

↓

~~f l(n-2)~~ f l(2) calls f l(1) and f l(2)

⋮

f l(n) calls f(n-1) n times + (n-1) times more

~~f l(1) level (n-1)~~

f l(2) calls f, 2(1) + (2-1) times
= 3 times

f l(3) calls f, 3(3) + (2) times
= 11 times

if we neglect (n-1) more times call we will get that f l(n) calls f l(n-1), n times.

runtime complexity is (n)(n-1)(n-2) ..... (1)

, so complexity is $\Theta(n^n)$