COL100 – Minor exam 1 solutions

Exam held on Feb. 3, 2018

1 Exam begins here

1. **(1 mark)** Suppose a directory `personal` has read and write access for users, but no access for groups and others, write the *permission string* that would be displayed on typing the command `ls -l personal`.

   drw_______

2. **(1 mark)** Suppose the directory structure is organised as in Figure 1 and your current working directory is as shown. Write a *single* `cd` command to go from your current directory to the directory indicated using a *relative path*.

   cd ../../../Calculator.app/Contents/MacOS

3. **(.5+1+1+1.5 marks)** For each expression below, write the value that is returned and its type. If there is an error, state what error is returned (in plain English).

   (a) `"1" ^ string_of_int 1`

   `"11", string`

   (b) `(1 < 2) = false`

   `false, boolean`

   (c) `1 + 2 / 8`

   1, int

   (d) `string_of_int (float_of_int 2) ^ "2"`

   Error. It identifies that `float_of_int` returns a `float`, but it expects an `int`

4. **(1 mark)** What is the value of `x` in the following code snippet?

```
let y = 5 in
let z = y+4 in
let x = z*5 in
let z = x+1 in
let x = y*(z+1) in x
```

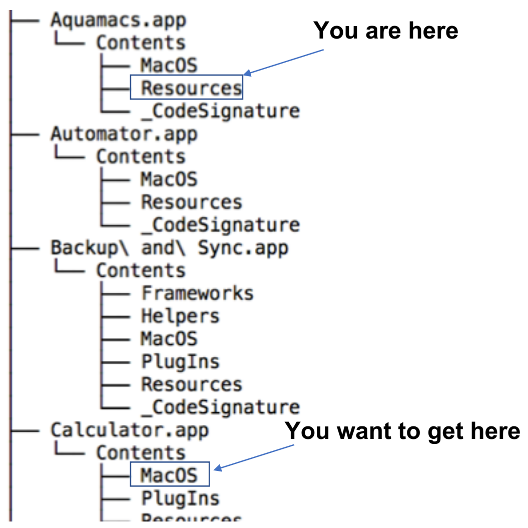
   235

5. **((1+2)+(2+2)+(2+2) marks)** For each function below (all named `f`), (i) Write the signature of the function (your notation should conform to that returned by the OCaml top level. That is, for function `let add a b = a+b`, write `int -> int -> int`), and, (ii) write a *one sentence* description of what the function does (for example, for function `add`, write "adds integers a and b").

   (a)
```
let rec f a b =
if b = 0 then a else f b (a mod b)
```

   `int -> int -> int`

   finds the GCD of integers a and b



```
├── Aquamacs.app
│   └── Contents
│       ├── MacOS
│       ├── Resources
│       └── _CodeSignature
├── Automator.app
│   └── Contents
│       ├── MacOS
│       ├── Resources
│       └── _CodeSignature
├── Backup\ and\ Sync.app
│   └── Contents
│       ├── Frameworks
│       ├── Helpers
│       ├── MacOS
│       ├── PlugIns
│       ├── Resources
│       └── _CodeSignature
├── Calculator.app
│   └── Contents
│       ├── MacOS
│       ├── PlugIns
```

Figure 1: Question 2: Directory structure

(b)
```
let rec f x n l =
match l with
| [] -> [x]
| h :: t -> if n = 0 then x :: l
else h :: f x (n-1) t
```

```
'a -> int -> 'a list -> 'a list
```

inserts element `x` at position `n` in list `l`, unless `n` > list length, in which case it inserts at the end of the list

(c)
```
let f g h x = let y = g x in h y
```

```
('a -> 'b) -> ('b -> 'c) -> 'a -> 'c
```

computes `h(g(x))`

6. **(2 marks)** Write a *single, recursive* function `altrnt:'a list -> 'a list` in OCaml, that takes in a list `lst` and returns a list containing every *alternate* element of `lst` starting from the first element (that is, the first element should be returned in the list). (*Note*: No partial marks will be awarded for this question.)

```
let rec altrnt lst =
match lst with
| [] -> []
| x :: [] -> [x]
| x :: y :: rest -> x :: altrnt rest
```

7. Define $h(n), n \geq 0$ as follows:
$h(0) = 0$
$h(n) = n - h(h(h(n-1))), n > 0$

Answer the following questions:

(a) **(2 marks)** What is $h(3)$?

2

(b) **(2 marks)** When translated into a recursive function in OCaml, which takes in n and returns the n^{th} element in the sequence, `h: int -> int` and that follows the recursion above exactly, how many times is the function `h` called in `h 3`.

22

8. We learnt about the Fibonacci sequence and how it is recursively defined. Here is a variant on that sequence. The idea is as follows: Let $Q(1) = Q(2) = 1$. For computing $Q(n)$, $n > 2$, we would like to look at the two elements before it – that is, elements at the $(n-1)^{th}$ position and the $(n-2)^{th}$ position. Let the elements at these positions be x and y, then we calculate the $Q(n)$ as the sum of the elements at $(n-x)^{th}$ position and $(n-y)^{th}$ position. Example: Suppose we want to find $Q(5)$, then we look at elements at the 4^{th} and 3^{rd} positions. Let those elements be 3 and 2. We then add the numbers at the $(5-3)^{th}$ and $(5-2)^{th}$ position (that is, the second and third position) to get $Q(5)$.

   Answer the following questions:

   (a) **(3 marks)** Write a *recursive* mathematical formulation of the above.

   $Q(1) = Q(2) = 1$
   $Q(n) = Q(n - Q(n-1)) + Q(n - Q(n-2))$, if $n > 2$
   $undefined, otherwise$

   (b) **(2 marks)** Write the first 10 elements in the sequence.

   1,1,2,3,3,4,5,5,6,6