+0/1/60+

Final Exam (Part 1) in Program Design and Data Structures (1DL201)

Teachers: Tjark Weber and Johannes Borgström

2019-01-14 / 14:00–19:00

Instructions

Read and follow these instructions carefully to increase your chance of getting good marks.

- This is a closed book exam. You may use a standard English dictionary. Otherwise, **no notes, calculators, mobile phones, or other electronic devices are allowed.** Cheating will not be tolerated.
- This is a multiple-choice exam. There are **twenty** questions. Each question has exactly **one** correct answer.
- Note your responses on the question sheets, and transfer them to the answer sheet only when you are ready to hand in.
- Read the instructions on the answer sheet before you start.
- You may keep these question sheets. **Only hand in the answer sheet.**
- Tjark Weber will come to the exam hall around 16:00 to answer questions.

Good luck!

Questions

Please choose a single answer for each question. Read the questions carefully, and watch out for negations (**not**, **except**, etc.).

```
{- func f xs
   PRE: ?PRE?
   RETURNS: ?RETURNS?
 -}

-- VARIANT: ?VARIANT?

func :: ?TYPE?
func f [] = []
func f (x:xs) = x : map f (func f xs)
```

Question 1: What is the value of `func (+1) [1,2,3]` ?

A `[1,2,3]`
B `[]`
C `[2,3,4]`
D `[1,3,5]`
E `[(+2),(+3),(+4)]`

Question 2: What is a type (`?TYPE?`) of `func`?

A `(String -> Int) -> [String] -> [Int]`
B `(a -> b) -> (a,a) -> (a,b)`
C `a -> b -> c`
D `(a -> b) -> [a] -> [b]`
E `(a -> a) -> [a] -> [a]`

Question 3: What is the most appropriate precondition (`?PRE?`) for `func f xs`?

A `f` is a function
B `xs` is a list
C `xs` is non-empty
D `True`
E `f` is associative

Question 4: What is an appropriate description of the return value (`?RETURNS?`) for `func f xs`?

A a list of type `[a]`
B `x : map f (func f xs)`
C `[`x_0`, f `x_1`, f (f `x_2`), ..., f`^{n} x_n`]`, where `xs` is `[`x_0`, ..., `x_n`]`
D the head of `xs`, concatenated with the result of mapping `f` over the recursive result obtained by applying `func f` to the tail of `xs`
E None of the above.

Question 5: Which of the following is a variant (`?VARIANT?`) for the function `func f xs`?

A $\Theta(\texttt{f}(n))$
B `f`
C `length xs`
D `xs`
E `length (x:xs)`

Question 6: Consider the following expression:

```
(True || (1 `div` 0 > 42)) && (False || (1 `div` 0 < 42))
```

Evaluating this expression will result in ...

A a type error.
B `True`
C `False`
D `Infinity`
E a run-time error.

Question 7: Consider the declaration

```
f x y =  let x = case y of x -> x in x
```

Which of the following is equivalent to the declaration above?

A `f x y =  let u = case y of v -> v in x`
B `f x y =  let u = case y of v -> x in u`
C `f x y =  let u = case y of v -> v in u`
D `f x y =  let u = case y of v -> v in v`
E None of the above.

Question 8: Recall the tower of Hanoi problem: Given three rods of which two are empty and one holds a sorted stack of n disks, move the entire stack to another rod, obeying these rules:

- Only one disk may be moved at a time.
- Each move consists of taking the upper disk from one of the stacks and placing it on top of another stack.
- No disk may be placed on top of a smaller disk.

The solution to this problem discussed at the lecture is an example of a particular form of recursion. Which?

A Simple recursion
B Complete recursion
C Multiple recursion
D Mutual recursion
E None of the above

Question 9: Consider the function

```
odds :: [a] -> [a]
odds []  = []
odds [a] = [a]
odds (a:b:l) = a : odds l
```

Which of the following is the best description of the return value (RETURNS) for `odds l`?

A the odd numbers of `l` in same order as they appear in `l`
B odd-positioned elements of `l` in the same order as they appear in `l`
C a sublist of `l`
D a list with the same type as `l`
E None of the above.

Question 10: Which of the following is a variant for `odds l`?

A `0 <= length l`
B `(length l) ‘div‘ 2`
C `l`
D the number of odd elements of `l`
E None of the above.

Question 11: Which of the following is a type of `map map` ?

A `[a] -> [b]`

B `[a -> b] -> [a -> b]`

C `[a -> b] -> [[a] -> [b]]`

D `(a -> b) -> [a] -> [b]`

E None of the above.

Question 12: What is the result of running the following comprehension?

```
[x+2*y | x <- [1..6], y <- [2,3,5], 0 == x `rem` y ]
```

A `[6,9,8,15,10,12]`

B `[6,9,15,12,20,25,30]`

C `[8,15,12,35,16,24]`

D An error

E None of the above.

Question 13: Recall the quicksort algorithm. Suppose the algorithm is applied to the input list `[8,12,6,3,19,4,6,2]` and the value `8` is chosen as the pivot. What are the arguments to the immediate recursive calls of the quicksort algorithm?

A `[8,6,3,4,6,2]` and `[12,19]`

B `[6,3,4,6,2]` and `[8,12,19]`

C `[2,3,4,6,6]` and `[12,19]`

D `[6,3,4,6,2]` and `[12,19]`

E None of the above.

Question 14: Which of the following is **false**?

A $\Theta(n^2) \subseteq \Omega(n^3)$

B $\Theta(n^2) = \Theta(2n^2)$

C $\Theta(n^2) = \Omega(n^2) \cap O(n^2)$

D $\Theta(n^2) \subseteq O(n^3)$

E $\Theta(n^2) = \Theta(n^2 + 2)$

Question 15: Consider the following function

```
foo :: Int -> Int
foo 0 = 0
foo 1 = 0
foo 2 = 1
foo n = foo (n-1) + foo (n-2) * foo (n-3)
```

Which of the following recurrences describes the run-time cost of `foo` n for $n \geq 3$? (The base cases are $T(n) = \Theta(1)$ for $n \leq 2$.)

A $T(n) = T(n-1) + T(n-2) + T(n-3) + \Theta(1)$

B $T(n) = T(n-1) + T(n-2) \cdot T(n-3) + \Theta(1)$

C $T(n) = \Theta(n-1) + \Theta(n-2) + \Theta(n-3) + \Theta(1)$

D $T(n) = T(n-1) + (n-2) \cdot T(n-3) + \Theta(1)$

E $T(n) = 3 \cdot T(n-2) + \Theta(1)$

Question 16: What is the closed form of the following recurrence?

$$T(0) = 1$$
$$T(n+1) = 3 \cdot T(n) + 2$$

A $T(n) = 3^n$

B $T(n) = \frac{8}{3}n^3 - 4n^2 + \frac{16}{3}n + 1$

C $T(n) = 3 \cdot T(n-1) + 2$

D $T(n) = 2 \cdot 3^n - 1$

E $T(n) = 1, 5, 17, 53, \ldots$

Question 17: Suppose `f` is a function of type `(a,b) -> (b,a)`. Which of the following expressions is **not** type correct?

A `f ("a","b")`

B `f ('a','b')`

C `f ('b','a')`

D `f . f`

E `f [('a','b')]`

Question 18: Which of the following expressions, if it evaluates successfully, will always (i.e., for any choice of `f`, `b`, and `xs`) return a list that is exactly as long as `xs`?

A `foldl f b xs`

B `foldr f b xs`

C `map f xs`

D `filter f xs`

E None of the above.

Question 19: Which of the following expressions does **not** evaluate to the same value as the other four?

A `foldl (+) (-6) [1,2,3,4]`

B `foldr max 0 [1,2,3,4]`

C `foldl (-) 0 [1,2,3,4]`

D `foldr (-) 6 [1,2,3,4]`

E `length [1,2,3,4]`

Question 20: Suppose you want to describe a physical design, where objects have positions and lengths. The length of an object is given by a value of type `Double`. Lengths are measured either in meters (m) or in inches (in).
Which of the following type declarations should you use to make sure that lengths measured in meters are not confused with lengths measured in inches?

A `type Length = Double`

B `type LengthMeters = Double; type LengthInches = Double`

C `data Length = Meters | Inches`

D `data Length = Length Double`

E `data Length = Meters Double | Inches Double`