Programming Languages
27 June 2019

Rules

Time at disposal: 4 h.

Read carefully the following questions. For every questions, provide an answer in English. Please remember to justify why the results hold by writing the logical flow that lead you to the answers.

To pass the exam a passing score should be reach in both the Haskell exercises (i.e., exercises 6,7, and 8) and on the remaining questions.

At the end of the exam, remember to submit the assignment both in the PeerGrade system and on Blackboard. The submitted PDF should be named with your special DM552 anonymous ID number received by email (i.e., if the ID is 000DM522 the file should be named 000DM522.pdf). Remember to write the anonymous ID number on the submitted text too and avoid reporting your personal name and surname (the elaborate should be anonymous).

Exercises

1. The following program is written using a pseudo-code. Assume that dynamic scoping and call by value are used. Expressions are evaluated from left to right (the `x++` returns the value of the variable `x` and then increments it). What does the following program write? Motivate the answer.

```
{ int x = 2;
  void foo(value int y){
    x = x + y;
  }
  { int x = 5;
    { int x = 7;
    }
    foo(x++);
    write(x);
  }
  write(x);
}
```

Will the result differ if static scoping is used? Motivate your answer.

One of the techniques to implement dynamic scoping is by using association lists. Show graphically the status of the A-list before executing the instruction $x = x + y$ inside the foo procedure.

2. Given $PE_{L_1}^{L_2}$ a partial evaluator of L_2 written in L_1, $I_{L_1}^{L_2}$ an interpreter of L_2 written in L_1, $[\![P]\!]$ the function computed by the program P. Consider the expression $[\![PE_{L_0}^{L_1}]\!](I_{L_1}^{L_2}, I_{L_2}^{L_3})$. Are the partial evaluators, compilers, or interpreters applied to the correct languages? If so, what does the evaluation produce?

3. What does call by name and call by value-result mean?

   What does the following code write, assuming a language with static scoping and call by name.

```
{
  int x = 7;
  int w = 1;

  void fie(name int y,z){
    int x = 1;
    z = y + z + x;
  }

  fie(x+w,w);
  write(w);
}
```

   Instead of call by name can we use call by value-result in the previous code for both the parameters? If so, what will the program write?

   Motivate the answer.

4. Describe the *stop and copy* technique: what is this technique used for and what does it do? What is the difference between stop and copy compared to mark and compact? What are the pro and cons of *stop and copy* compared to mark and compact?

5. What do we mean with variable shadowing and method overriding. Give an example (pseudocode or Java at your discretion) of a variable shadowing and method overriding showing how to invoke a method that has been overridden and how to access a shadowed variable.

   What does it mean that a method is virtual? Can you provide an example of pseudocode that defines a method `foo` and changes its behaviour if `foo` is defined as a virtual method or not (e.g., write a program that prints "Virtual" if `foo` is virtual, otherwise it prints "No virtual").

6. A lot of techniques for the compression of images are based on a tree data structure called "Quad Tree". Assume that the image is square and the size of the square is a power of 2. If the image is homogeneous (same color) it is encoded, regardless of its dimension, as a leaf containing its color. If the image is not homogeneous, then it is encoded as a node whose child encode i) the upper left square, ii) the upper right square, iii) the bottom left square, and iv) the bottom right square using the data type

```
data QT a = C a | Q (QT a ) (QT a ) (QT a ) (QT a )
```

Write a function in Haskell named `centerSymmetric` that given a QuadTree checks if the image is center symmetric (i.e., if it is equivalent to itself when rotated by 180°).

Write the type signature of all the functions that you define.
Motivate your choices briefly describing your code.

7. Write a function `beauty` that taken a predicate on the colors and a Quad Tree checks if the external border of the image has pixels having colors that satisfy the given predicate.

   As an example

```
let z = C 0; u = C 1; q1 = Q u u u z
    q3 = Q u z u u
in beauty (==1) (Q q1 u q3 u)
```

   will produce `True`.

   Write the type signature of all the functions that you define.
   Motivate your choices briefly describing your code.

8. What is the keyword `Monad` in Haskell? Assuming you want to create a monad M, what operations can/must you define for M?

   Provide an example in Haskell that defines an instance of a Monad? Is there any law that you should be aware of?

   The list type is an instance of the monad where

```
return x = [x]
xs >>= f = concat (map f xs)
```

   where concat is the function that concatenates lists that can be defined as follows.

```
concat [] = []
concat (x:xs) = x ++ (concat xs)
```

   The join function is defined as

```
join :: (Monad m) => m (m a) -> m a
join xs = do
  x <- xs
  x
```

   What does join `[[1,2,3],[4,5]]` do?

   Unfold the do syntactic sugar notation and explain step by step the transformations that Haskell does to reach the final result.