Faculty of Science and Engineering

Computer Science

CS-205 Declarative Programming

January 2024

Time Allowed: 2 hours

Do NOT turn over your question paper until instructed to do so.

Exam Paper Information

NB. The content of this exam paper has been thoroughly checked, however if you think you have spotted an error, please raise your hand and report to an invigilator. Unless it has a significant impact on the paper the error will NOT be corrected in the exam venue. Errors not corrected in the exam should be noted in your answer booklet and you should continue to answer the examination paper to the best of your ability as it is written. It will be noted by the module coordinator and the Board of Examiners when it considers your results.

Special Instruction(s)

Please start a new page for each subquestion 1a, 1b, and so on. Do not change the order, and clearly indicate when a question is continued elsewhere.

Specific Items

Dictionaries - Candidates may only refer to the English and Welsh language dictionaries available

Calculators - Candidates may NOT use a calculator

Open Book - This is NOT an Open Book Exam

Question 1: Functional Programming [30 marks]

(a) For each of the following Haskell expressions, write out what result would be computed by `ghci` (you do not need to spell out the intermediate evaluation steps).

(i) `(+ 3) 2`

(ii) `"Hello " ++ (reverse "Hi")`

(iii) `[ x / 2 | x <- [4,6,9]]`

(iv) `filter (< 5) [1..100]`

(v) `map (\ x -> [x, x]) [1,5]`

[5 marks]

(b) The final grade for the CS-205 module is weighted as follows: the exam is worth 70% of the total grade, the coursework is worth 15% and the lab classes are worth 15%. All grades, including the final grade, are rounded to the nearest integer and are out of a total of 100.

Write a function, with its type signature, that takes as inputs a grade for each component and outputs the final grade for the module.

You may use the function `fromIntegral` to convert an integer into a floating point value and `round` to round a floating point value into an integer value.

[5 marks]

(c) Explain briefly (in no more than two sentences) what it means for a function to be

(i) polymorphic

(ii) higher-order

Then give an example of a function which is both polymorphic and higher-order.

[5 marks]

(d) For all of the following Haskell definitions, indicate if they typecheck, and if they do, write down their (most general) types.

(i) `myPair = ("abc", "de")`

(ii) `tryCycle (x : xs) = xs ++ x`

(iii) `add11 = 1 + True`

(iv) `f a b = a ++ (reverse b)`

[4 marks]

(e) Let the following Haskell program be given

```
repeatAndCount :: Int -> IO ()
repeatAndCount n = do
                   {
                     s <- getLine;
                     if s == ":q" then
                       putStrLn ("Total size of inputs " ++ show n)
                     else
                       do {
                         putStrLn s;
                         repeatAndCount (n + length s)
                       }
                   }
```

(i) What is the type of the function `putStrLn`?

(ii) Explain informally how this code works when we call `repeatAndCount 0`. Under which condition will this program terminate?

(iii) Give an example of a sequence of user inputs that will result in exactly one recursive call being performed. What are the corresponding outputs to the console?

[6 marks]

(f) Consider the following data type for binary trees

```
data BTree a = Leaf a | BNode (BTree a) a (BTree a)
```

Such a binary tree is called a *heap* if the value at any node is greater than all of the values held in all subtrees below that node. Write a Haskell function

```
isHeap :: Ord a => BTree a -> Bool
```

which takes as input such a binary tree and checks if it is a heap. Hint: it might be useful to define an auxiliary function

```
treeMax :: Ord a => Btree a -> a
```

such that `treeMax t` is equal to the maximal value of labels in `t`.

[5 marks]

Question 2: Programming in Prolog **[20 marks]**

(a) Indicate for the following pairs of terms whether they are unifiable or not. If a pair is not unifiable, explain why this is the case. If a pair is unifiable, give a unifier.

(i) `child(X,mia)` and `child(harry,Y)`.

(ii) `child(X,bob)` and `mia`.

(iii) `g(X,f(X),Z)` and `g(f(Z),Y,a)`.

(iv) `child(Tom,bob)` and `child(harry,Tom)`.

(v) `[X,Y]` and `[a,b,c,d]`.

[5 marks]

(b) Consider the following knowledge base including the members of the royal family.

```
parent(elizabeth,elizabethII).
parent(georgeVI,elizabethII).
parent(elizabethII,charles).
parent(elizabethII,andrew).
parent(elizabethII,anne).
parent(elizabethII,edward).
parent(philip,charles).
parent(philip,andrew).
parent(philip,anne).
parent(philip,edward).
parent(diana,william).
parent(diana,harry).
parent(charles,william).
parent(charles,harry).
parent(kate,georgejun).
parent(kate,charlotte).
parent(kate,louis).
parent(william,georgejun).
parent(william,charlotte).
parent(william,louis).
parent(meghan,archie).
parent(meghan,lilibet).
parent(harry,archie).
parent(harry,lilibet).
```

The following two predicates define the lists of all female and male members of the Royal Family respectively:

```
the_royal_females([elizabeth,elizabethII,anne,diana,
       kate,charlotte,meghan,lilibet]).

the_royal_males([georgeVI,philip,charles,andrew,edward,
      william,harry,georgejun,louis,archie]).
```

Define the following predicates

(i) `royal_female/1` and `royal_member/1` holding for members of the royal females, royal family respectively. (You may use the member predicate.)

(ii) `mother/2` and `grandmother/2`.

(iii) `ancestor/2`. (The ancestors are the parents, grandparents, parents of grandparents, and so on...)

(iv) `sibling/2`. (One joint parent sufficies to define the sibling relationship.)

Translate the following questions into Prolog queries:

(v) Who has a child (one or more)?

(vi) Who has a sibling who is a grandparent?

(vii) Who is an ancestor of `louis`? (Also give the answers Prolog will provide).

[11 marks]

(c) Consider the following Prolog program:

```
mystery([],1).

mystery([X|Xs],P) :-
        mystery(Xs,Q),
        P is X*X*Q.
```

(i) What does the query `?-mystery([1,2,3,2],R)` yield?

(ii) Explain what we understand by tail recursion.

(iii) Is this predicate tail recursive? [If yes, explain why, if no, explain how you can turn it into a tail recursive predicate.]

[4 marks]

End of Paper