Functional Programming, COMP0020 (A6U, A7P)
Sample paper, April 2022

Suitable for Cohorts: 2021/22, 2020/21, 2019/20

Answer ALL THREE questions. Do not plagiarise: especially, do not work with others, do not copy from others, do not copy from books or the web, and do not copy from the module presentations, slides, or other module material. Your answers must be your own work, in your own words.

Your answers will be marked for quality, including quality of expression.

Marks for each part of each question are indicated in square brackets.

Standard calculators are permitted.

1. (a) Explain what the following Miranda algebraic type represents:

   fred * ::= Empty | Node * [fred *]

   [2 marks]

   (b) Consider the following Miranda code (the ! operator provides an index into a list counting from zero, for example [1,2,3]!0 returns the value 1):

   t_graph * ::= Emptygraph | Node * [t_graph *]

   glist :: [t_graph char]
   glist = [Node 'A' [glist!2, glist!1],
             Node 'B' [glist!3],
             Node 'C' [glist!0, glist!3],
             Node 'D' []
           ]

   graph :: t_graph char
   graph = hd glist

   Give a diagram to illustrate the data structures glist and graph.   [4 marks]

   (c) The predefined Miranda function <u>member</u> is defined as:

   member :: [*] -> * -> bool
   member [] x = False
   member (x:xs) y = (x=y) $\lor$ member xs y

   Using the function <u>member</u>, give the definition for a Miranda function called printgraph that takes an argument of type <u>t_graph char</u> and returns a list of characters representing the data structure. The word "empty" should be used to signify an empty graph node. Your program must not loop forever; if a <u>Node</u> has been encountered previously, the word "seen" should be printed instead of its list of successor nodes.

   For example, the application <u>(printgraph graph)</u> should return the following result (NB a node that is shared and appears in different branches may appear multiple times in the output):

   Node A [Node C [Node A seen, Node D empty], Node B [Node D empty]]

   [21 marks]

   [Total for Question 1: 27 marks]

2. Provide Miranda code to implement the following simple game, which shuffles a collection of an arbitrary number of playing cards. Do **not** write a complex user interface. Your main function should take as arguments a number, a list of elements and a function that can be applied to two cards to determine if they are equal. Each element in the list represents a playing card, where each playing card has a "suit" (there are four possible suits) and a "number" (there are thirteen possible numbers, with the top four being called "Jack", "Queen", "King" and "Ace").

The Miranda code should produce as its output a "shuffled" version of the input list (the second argument). The function should shuffle the list of cards as many times as is indicated by the first argument. The action of shuffling should cut the deck in half and then interleave the cards, with the previous top card now being the second card in the pack. For example, if a list of four items A, B, C and D is shuffled once the result should be C, A, D, B. If that result is shuffled again the result should be D, C, B, A.

If the list contains an odd number of cards, you should detect this case and provide an appropriate solution. You should explain your solution – what it does and why.

Your function should detect the following two errors and provide appropriate error handling:

  (i) where there are more than 52 elements in the input list, and
  (ii) where there is any duplicated card in the input list.

Your answer will be marked for quality (including coherence, elegance, clarity,
understandability, commentary and brevity) as well as for correctness and completeness.

An ideal answer will not exceed 700 words of comment and code.

[30 Marks]

[Total for Question 2: 30 marks]

3. Provide Miranda code to implement the following simple game.

The game called "Minefield" presents the user with a 10x10 grid of cells that are initially blank. Five (5) of these cells contain hidden mines. The user is invited to enter (x,y) coordinates (such that both x and y are between 1 and 10 inclusive) for cells to be visited. The user is given one point for every cell that is visited that does not contain a mine. As soon as the user visits a cell that contains a mine, or visits a cell that has already been visited, the game is over and the program prints the user's score to the screen.

Your answer will be marked for quality (including coherence, elegance, clarity, understandability, commentary and brevity) as well as for correctness and completeness. This is an open-ended programming challenge within the time available, and you may include stretch objectives of your own choosing.

An ideal answer will not exceed 1,200 words of comments and code.

[43 marks]

[Total for Question 3: 43 marks]