CSCI3180 – Principles of Programming Languages – Spring 2020

Assignment 4 — Declarative Programming

Deadline: May 6, 2020 (Wednesday) 23:59

1 Introduction

Declarative programming is a programming paradigm that emphasises the expression of "what" the problem is over "how" to solve the problem. You will gain experience of declarative programming with Prolog (Logic Programming) and ML (Functional Programming) in this assignment.

You are supposed to test your work on the machine `solar1.cse.cuhk.edu.hk` using

- SICStus 3.12.7
- Standard ML of New Jersey, Version 110.0.7

They can be invoked using the commands `sics` and `sml` respectively.

2 Logic Programming

Implement the predicates or queries of the following problems in a Prolog program file named "asg4.pl", and recall the successor notation `s()` to represent natural numbers in this task. You should clearly indicate using *comments* the corresponding question number of each problem.

1. Recall the list operations taught in lectures and tutorials.

   (a) Define `element_last(X, L)` which is true if the last element in list `L` is `X`. Example:
   ```
   ?- element_last(e, [a,b,c,d,e]).
   yes
   ```

   (b) Define `element_n(X, L, N)` which is true if the `N`-th element in list `L` is `X`. Example:
   ```
   ?- element_n(c, [a,b,c,d,e], s(s(s(0)))).
   yes
   ```

   (c) Define `remove_n(X, L1, N, L2)` which is true if the resulting list `L2` is obtained from `L1` by removing the `N`-th element, and `X` is the removed element. Example:
   ```
   ?- remove_n(X, [a,b,c,d,e], s(s(s(0))), L2).
   X = c,
   L2 = [a, b, d, e]
   ```

   (d) Based on (c), give a query to find which list will become `[c,b,d,e]` after removing its second element "`a`".

   (e) Define `insert_n(X, L1, N, L2)` which is true if the resulting list `L2` is obtained by inserting `X` to the position before the `N`-th element of list `L1`. Example:
   ```
   ?- insert_n(h, [a,b,c], s(s(0)), L2).
   L2 = [a,h,b,c]
   ```

   (f) Define `repeat_three(L1, L2)` which is true if the resulting list `L2` has each element in list `L1` repeated three times. Example:
   ```
   ?- repeat_three([a,b,c,d,e],X).
   X = [a,a,a,b,b,b,c,c,c,d,d,d,e,e,e]
   ```

   (g) Based on (f), give a query to find which list will become `[i,i,i,m,m,m,n,n,n]` after repeating each element of it for three times.

2. A *multi-way tree* is composed of a root and a sequence of sub-trees (children), which are multi-way tree themselves. The list of sub-trees of a certain node is ordered from left to right and also called a *forest*. The root of a multi-way tree is a node containing a value. Diagrammatically, a multi-way tree consists of a set of nodes and lines connecting parents

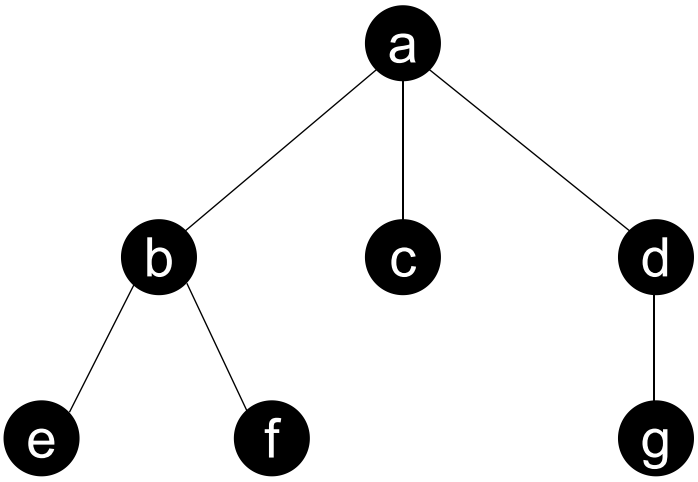

Figure 1: An example of multi-way tree

and children. The nodes are depicted by circles with values written inside. A multi-way tree is never empty.

In Prolog, we can represent a multi-way tree by the term "`mt(X, F)`", where `X` denotes the root and `F` denotes the forest of sub-trees. Please finish the following questions using the successor notation `s()` to represent natural numbers in this task. You can use the `sum/3` predicate from the lecture.

(a) Represent the multi-way tree in Figure 1 as a Prolog term, with order of the sub-trees from left to right. (Hint: represent forest as a list of multi-way tree(s)).

(b) Define the predicate `is_tree(Term)` which is true if `Term` represents a multi-way tree.

(c) Define the predicate `num_node(Tree, N)` which is true if `N` is the number of nodes of the given multi-way tree `Tree`.

(d) Define `sum_length(Tree, L)` which is true if `L` is the sum of lengths of all internal paths in `Tree`. The length of an internal path from the root node r to an internal node n is the distance from r to n. For example, the sum of lengths of all internal paths in the multi-way tree in Figure 1 is: 1 (b) + 1 (c) + 1 (d) + 2 (e) + 2 (f) + 2 (g) = 9 (i.e., s(s(s(s(s(s(s(s(s(0)))))))))).

3 Functional Programming

Implement the required functions of the following problems in an ML program file named "asg4.ml". You should clearly indicate using comments the corresponding question number of each problem.

This task involves a card game simplified from Texas Hold'em. In each round, two players will get five cards respectively. The player who has the better hand will win the game. You have to implement several ML helper functions to determine the better hand.

Please consider the following eight different hands. (We refer to the rules defined on `https://www.pagat.com/poker/rules/ranking.html`.)

- **Four of a kind**

  Four cards of the same rank - such as four queens. The fifth card, known as the kicker, can be anything. Between two fours of a kind, the one with the higher set of four cards is higher, so 3-3-3-3-A is beaten by 4-4-4-4-2. If two players have four of a kind of the same rank, the rank of the kicker decides.

- **Full House**

This combination consists of three cards of one rank and two cards of another rank - for example three sevens and two tens. When comparing full houses, the rank of the three cards determines which is higher. For example 9-9-9-4-4 beats 8-8-8-A-A. If the threes of a kind are equal, the rank of the pairs decides.

- **Flush**

  Five cards of the same suit. When comparing two flushes, the highest card determines which is higher. If the highest cards are equal then the second highest card is compared; if those are equal too, then the third highest card, and so on. For example ♠K-♠J-♠9-♠3-♠2 beats ♢K-♢J-♢7-♢6-♢5 because the nine beats the seven. If all five cards are equal, the flushes are equal.

- **Straight**

  Five cards of mixed suits in sequence. For example ♠Q-♢J-♡10-♠9-♣8. When comparing two sequences, the one with the higher ranking top card is better. Ace can count high or low in a straight, but not both at once. So A-K-Q-J-10 and 5-4-3-2-A are valid straights, but 2-A-K-Q-J is not. 5-4-3-2-A, known as a wheel, is the lowest kind of straight, the top card being the five.

- **Three of a Kind**

  Three cards of the same rank plus two unequal cards. When comparing two threes of a kind the rank of the three equal cards determines which is higher. If the sets of three are of equal rank, then the higher of the two remaining cards in each hand are compared, and if those are equal, the lower odd card is compared. So for example 5-5-5-3-2 beats K-4-4-4-5, which beats Q-9-4-4-4, which beats Q-8-4-4-4.

- **Two Pairs**

  A pair consists of two cards of equal rank. In a hand with two pairs, the two pairs are of different ranks (otherwise you would have four of a kind), and there is an odd card to make the hand up to five cards. When comparing hands with two pairs, the hand with the highest pair wins, irrespective of the rank of the other cards - so J-J-2-2-A beats 10-10-9-9-8 because the jacks beat the tens. If the higher pairs are equal, the lower pairs are compared, so that for example 8-8-6-6-3 beats 8-8-5-5-4. Finally, if both pairs are the same, the odd cards are compared, so Q-Q-5-5-4 beats Q-Q-5-5-3.

- **Pair**

  A hand with two cards of equal rank and three cards which are different from these and from one another. When comparing two such hands, the hand with the higher pair is better - so for example 6-6-4-3-2 beats 5-5-4-3-2. If the pairs are equal, compare the highest ranking odd cards from each hand; if these are equal compare the second highest odd card, and if these are equal too compare the lowest odd cards. So J-J-9-3-2 beats J-J-8-7-3 because the 9 beats the 8.

- **Nothing**

  Five cards which do not form any of the combinations listed above. The cards must all be of different ranks, not consecutive, and contain at least two different suits. When comparing two such hands, the one with the better highest card wins. If the highest cards are equal the second cards are compared; if they are equal too the third cards are compared, and so on. So A-J-9-5-3 beats A-10-9-6-4 because the jack beats the ten.

Note that cards are already sorted in *descending* order according to the rank (with A always ordered last). The order of cards of the same rank is *arbitrary*. Each card has two attributes, i.e., suit and rank. The type definition of a card in ML is shown below. The joker cards are not included in this game. To further simplify the problem, we convert all ranks to integers. i.e., A = 1, J = 11, Q = 12, and K = 13.

```
datatype suit = Clubs | Diamonds | Hearts | Spades;
```

When you implement the following functions, please use *pattern matching* as much as possible.

1. Write an ML function `check_flush`, which takes a list of five cards and returns if the hand is a flush.

2. Write an ML function `compare_flush`, which takes two *flush* card lists. The return value is a string selected from three candidates. *i.e.*, "Hand 1 wins", "Hand 2 wins" and "This is a tie".

3. Write an ML function `check_straight`, which takes a list of five cards and returns if the hand is a straight.

4. Write an ML function `compare_straight`, which takes two *straight* card lists. The return value is a string selected from three candidates. *i.e.*, "Hand 1 wins", "Hand 2 wins" and "This is a tie".

5. Write an ML function `count_patterns`, which takes a list of five cards and returns the hand type (Nothing, Pair, Two_Pairs, Three_of_a_Kind, Full_House, Four_of_a_Kind) and a list of rank-quantity pairs. Note that the list should be ordered according to the order of comparison of each type of hand.

   Here are some examples to help you understand the problem:

   - The card list [(Clubs, 13), (Clubs, 11), (Spades, 7), (Spades, 3), (Hearts, 2)] should return (Nothing, [(13, 1), (11, 1), (7, 1), (3, 1), (2, 1)]).
   - The card list [(Spades, 11), (Spades, 9), (Hearts, 8), (Diamond, 8), (Diamonds, 3)] should return (Pair, [(8, 2), (11, 1), (9, 1), (3, 1)]).
   - The card list [(Clubs, 13), (Spades, 13), (Hearts, 6), (Spades, 1), (Diamonds, 1)] should return (Two_Pairs, [(13, 2), (1, 2), (6, 1)]).
   - The card list [(Clubs, 10), (Clubs, 9), (Hearts, 9), (Spades, 9), (Spades, 3)] should return (Three_of_a_Kind, [(9, 3), (10, 1), (3, 1)]).
   - The card list [(Diamonds, 6), (Clubs, 6), (Spades, 6), (Spades, 4), (Diamonds, 4)] should return (Full_House, [(6, 3), (4, 2)]).
   - The card list [(Diamonds, 11), (Spades, 11), (Clubs, 11), (Hearts, 11), (Hearts, 10)] should return (Four_of_a_Kind, [(11, 4), (10, 1)]).

6. Write an ML function `compare_count`, which takes two card lists and returns a string selected from three candidates. *i.e.*, "Hand 1 wins", "Hand 2 wins" and "This is a tie". You only need to consider the remaining six cases except flush and straight, and assume that

   Nothing < Pair < Two_Pairs < Three_of_a_Kind < Full_House < Four_of_a_Kind.

4 Submission Guidelines

Please read the guidelines CAREFULLY. If you fail to meet the deadline because of submission problem on your side, marks will still be deducted. So please start your work early!

1. In the following, **SUPPOSE**

   your name is *Chan Tai Man*,
   your student ID is *1155234567*,
   your username is *tmchan*, and
   your email address is *tmchan@cse.cuhk.edu.hk*.

2. In your source files, insert the following header. REMEMBER to insert the header according to the comment rule of Prolog and ML.

```
/*
 * CSCI3180 Principles of Programming Languages
 *
 * --- Declaration ---
 *
 * I declare that the assignment here submitted is original except for source
 * material explicitly acknowledged.  I also acknowledge that I am aware of
 * University policy and regulations on honesty in academic work, and of the
 * disciplinary guidelines and procedures applicable to breaches of such policy
 * and regulations, as contained in the website
 * http://www.cuhk.edu.hk/policy/academichonesty/
 *
 * Assignment 4
 * Name : Chan Tai Man
 * Student ID : 1155234567
 * Email Addr : tmchan@cse.cuhk.edu.hk
 */
```

The sample file header is available at

`http://course.cse.cuhk.edu.hk/~csci3180/resource/header.txt`

3. Late submission policy: less 20% for 1 day late and less 50% for 2 days late. We shall not accept submissions more than 2 days after the deadline.

4. `Tar` your source files to `username.tar` by

```
tar cvf tmchan.tar asg4.pl asg4.ml
```

5. `Gzip` the `tarred` file to `username.tar.gz` by

```
gzip tmchan.tar
```

6. `Uuencode` the `gzipped` file and send it to the course account with the email title "HW4 *studentID yourName*" by

```
uuencode tmchan.tar.gz tmchan.tar.gz \
| mailx -s "HW4 1155234567 Chan Tai Man" csci3180@cse.cuhk.edu.hk
```

7. Please submit your assignment using your Unix accounts.

8. An acknowledgement email will be sent to you if your assignment is received. **DO NOT** delete or modify the acknowledgement email. You should contact your TAs for help if you do not receive the acknowledgement email within 5 minutes after your submission. **DO NOT** re-submit just because you do not receive the acknowledgement email.

9. You can check your submission status at

`http://course.cse.cuhk.edu.hk/~csci3180/submit/hw4.html`.

10. You can re-submit your assignment, but we will only grade the latest submission.

11. Enjoy your work :>