## **CCPS 721 Take-Home Midterm Winter 2019: Poker hands**

This take-home midterm for the course [CCPS 721 Artificial Intelligence](https://docs.google.com/document/d/138jB5apssZyEP88JctdN2FAkGt_LapAphe270IWC04o/edit?usp=sharing) in the Winter 2019 term requires you to write a bunch of predicates in Prolog that **recognize and generate all five-card poker hands of the given type**. These predicates can then be used to answer interesting combinatorial queries such as "How many full houses contain the five of diamonds?" or "Using only cards whose rank is higher than seven, how many two pair hands exist?"

Write all the predicates and rules in one file *poker.pl* and submit this file to the instructor attached to an email. The deadline for midterm submissions is **noon Sunday March 3.** Submit your *poker.pl* file in the course D2L page.

**This is an exam, and all Ryerson rules for academic integrity are in full effect the same way as they are for all Ryerson exams. Until the deadline is over and the results have been tallied, you are not allowed to discuss these problems and their solutions with anybody else either inside or outside this course, but must complete your work entirely on your own.**

This take-home midterm is worth 30% of your total course grade.

To keep this fair, the instructor will not provide any hints or help to any question. However, if you have strong and well-reasoned belief that there is an error somewhere in this midterm or the test suite, which is an actual possibility with a non-trivial nonzero probability of happening, please email the instructor about it as soon as possible. (But your belief should have some more firm chain of reasoning behind it than merely some dressing of "I don't really understand what is going on here.")

**The setup**

To define the familiar ordinary deck of 52 playing cards that are made out of 4 suits and 13 ranks, you **must** start with the following definitions for these ranks and suits. Copy-paste the following block of code to the start of your Prolog source code file *poker.pl* that you write and submit in this lab.

suits([clubs, diamonds, hearts, spades]).

ranks([deuce, three, four, five, six, seven, eight, nine, ten, jack, queen, king, ace]).

suit(X) :- suits(S), member(X, S).

rank(X) :- ranks(S), member(X, S).

card((R, S)) :- rank(R), suit(S).

Individual playing cards will be represented as Prolog terms that are two-element tuples of rank and suit, separated by a comma inside parentheses. For example (deuce, diamonds) or (queen, hearts).

|  |  |
| --- | --- |
| **Example query** | **Expected result** |
| card(C). | (52 solutions) |
| card((R, hearts)). | (13 solutions) |

**Question 1: Comparing individual cards**

To make the operations on poker hands easier and more efficient to compute, we require that whenever a poker hand is represented as a list of cards, that list is **sorted in order of decreasing rank**, to make sure that every possible poker hand has exactly one unique possible representation in our system. Ties between equal ranks are resolved by suit, in order of the descending suit rankings "spades, hearts, diamonds, clubs" order of contract bridge.

For example, any nine must come in the hand before any card that is eight or lower. The nine of hearts must come before the nine of diamonds, but after the nine of spades. The ace of spades is the highest card in this ordering and therefore the first in every hand where it appears, and the deuce of clubs is the lowest card that must therefore always be the last card in every hand where it appear. **All the predicates that are implemented in the questions after this one must accept and generate only hands that are sorted this way.**

Start by writing the predicates to compare the individual suits and ranks to each other. Write the following predicates:

/\* Succeeds if S1 is a lower suit than the suit S2. \*/

lower\_suit(S1, S2) :- ...

/\* Succeeds if R1 is a lower rank than the rank R2. \*/

lower\_rank(R1, R2) :- ...

/\* Succeeds if card (R1, S1) is lower than card (R2, S2) in the hand ordering. \*/

lower\_card((R1, S1), (R2, S2)) :- ...

|  |  |
| --- | --- |
| **Example query** | **Expected result** |
| lower\_suit(S, spades). | (three solutions: clubs, diamonds, and hearts in some order) |
| lower\_rank(R, nine), lower\_rank(six, R). | (two solutions: seven and eight in some order) |
| lower\_card(C, (seven, hearts)). | (22 solutions) |
| lower\_card((jack, spades), C). | (12 solutions) |

**Question 2: Sorting your hand**

Write the predicate sorted\_hand(H, N) that succeeds if H is a list of cards that contains exactly N different cards **sorted in descending order from highest card to lowest**, as defined by the previous predicate lower\_card/1. This predicate should fail if the hand has some other number of cards than N, or contains some term that is not a legal playing card, or has any two of its legal playing cards in the wrong order.

Note that this predicate has to work correctly for any nonnegative integer N, not just for the special case of N = 5 used in the game of poker.

|  |  |
| --- | --- |
| **Example query** | **Correct result** |
| /\* How many sorted two-card hands exist? \*/  findall(H, sorted\_hand(H, 2), \_L), length(\_L, Len). | 1326 |
| /\* How many ways can this sorted hand of three cards be completed? \*/  findall(C, sorted\_hand([(queen, spades), (four, diamonds), C], 3), \_L), length(\_L, Len). | 9 |
| /\* How many sorted five-card hands exist with nine of clubs in the middle? \*/  findall(H, sorted\_hand([\_ ,\_ ,(nine, clubs), \_, \_], 5), \_L), length(\_L, Len). | 95634 |
| /\* How many sorted five-card hands have some spade as the highest card, and then the deuce of hearts in the middle? \*/  findall(H, sorted\_hand([(\_, spades) ,\_ ,(deuce, hearts), \_, \_], 5), \_L), length(\_L, Len). | 312 |
| /\* A hand can contain only legal cards from the deck of 52. \*/  findall(\_, sorted\_hand([hello, world, 42], 3), \_L), length(\_L, Len). | 0 |

**Question 3: Four of a kind and full house**

A five-card poker hand is called a **four of a kind** if it contains four cards of the same rank and some fifth card. The five-card poker hand is called a **full house** if it contains three cards of the same rank, and a pair of two cards of some other rank. Write the predicates four\_of\_kind(H) and full\_house(H) that succeed if H is a sorted five card poker hand that is four of a kind or a full house, respectively.

|  |  |
| --- | --- |
| **Example query** | **Expected result** |
| /\* How many four of kinds exist in total? \*/  findall(H, four\_of\_kind(H), \_L), length(\_L, Len). | 624 |
| /\* How many full houses exist in total? \*/  findall(H, full\_house(H), \_L), length(\_L, Len). | 3744 |
| /\* How many full houses contain the deuce of spades? \*/  findall(H, (full\_house(H), member((deuce, spades), H)), \_L), length(\_L, Len). | 360 |

**Question 4: Straights and flushes**

A five card poker hand is called a **flush** if all its cards are of the same suit, and a **straight** if its five ranks are consecutive. As an extra twist, an ace can be treated as the highest or the lowest card for the purposes of a straight. (An ace-to-five lowest possible straight is called a **bicycle straight**, as opposed to the highest possible straight, ten-to-ace **Broadway straight**.)

To qualify as a flush, that hand cannot simultaneously be a straight, and to qualify as a straight, the hand cannot simultaneously be a flush. A hand that qualifies for both flush and straight is called a **straight flush**. (Furthermore, a straight flush that is a Broadway is called a **royal flush**, the highest possible hand that can exist in the ordinary five card poker. For simplicity, we shall consider royal flushes to be the highest possible straight flushes, instead of being a separate hand category.)

Write the predicates flush(H), straight(H) and straight\_flush(H) that succeed if H is a flush, straight or a straight flush, respectively. (Again, royal flushes are counted as straight flushes for the purposes of this question. Straight flushes are neither straights or flushes, but an entire separate hand category.)

|  |  |
| --- | --- |
| **Example query** | **Expected result** |
| /\* How many straights exist in total? \*/  findall(X, straight(X), \_L), length(\_L, Len). | 10200 |
| /\* You can't make a straight without a five or a ten. \*/  findall(X, (straight(X), not(member((five,\_),X)), not(member((ten,\_),X))),\_L),length(\_L, Len). | 0 |
| /\* How many flushes exist in total? \*/  findall(X, flush(X), \_L), length(\_L, Len). | 5108 |
| /\* How many straight flushes exist in total? \*/  findall(X, straight\_flush(X), \_L), length(\_L, Len). | 40 |
| /\* No flush is a straight. \*/  findall(X, (flush(X), straight(X)), L). | [] |

**Question 5: Three of a kind, two pair, and one pair**

A five card poker hand is called a **three of a kind** if it contains three cards of the same rank, and some two other cards that do not pair with any other card in the hand. A five card poker hand is called **two pair** if it contains two separate pairs of cards with equal rank, and a fifth card that does not pair with any one of them. A five card poker hand is called a **pair** if it contains two cards with the same rank, and some other three cards that do not pair with any other card in the hand. Write the predicates three\_of\_kind(H), two\_pair(H) and and one\_pair(H) that succeed if H is a three of a kind or a pair, respectively.

|  |  |
| --- | --- |
| **Example query** | **Expected result** |
| /\* How many three of kinds exist in total? \*/  findall(X, three\_of\_kind(X), \_L), length(\_L, Len). | 54912 |
| /\* How many two pair hands exist in total? \*/  findall(X, two\_pair(X),\_L),length(\_L, Len). | 123552 |
| /\* How many one pair hands have five of diamonds in the middle? \*/  findall(X, (X = [\_, \_, (five, diamonds), \_, \_], one\_pair(X)), \_L), length(\_L, Len). | 23328 |

**Grading your midterm submissions**

Your midterm will be graded with the aid of the following tester predicate that can be given any Query to execute and any Test to check that the results were correct. The parameter Inf unifies with the number of logical inferences performed during the execution of your method. This number is also printed on the terminal, being a nice measure of the execution speed that is independent of the computer that these tests are run on.

/\* Query: query to test

X: Term to solve in query

Inf: number of inferences needed to execute the query

Res: the variable representing the list of all solutions

Test: query that must be true for Res for test to pass

\*/

test(Query, X, Inf, Res, Test) :-

statistics(inferences, I1),

call(findall(X, Query, Res)),

statistics(inferences, I2),

Inf is I2 - I1,

call(Test) ->

(write('success '), write(Inf), nl, !) ;

(write('failure '), write(Res), nl, fail).

test(\_, \_, 0, \_, \_).

The actual test bed will consist of a series of fifteen tests cases. Each test case that you pass is worth 2 points for the midterm mark of 30 points.

Here is an example test bed that you can use to test the correctness your solutions. The actual grading will then be done with a different test\_all predicate that has the same general structure as this one, but different test case queries.

test\_all(Inf) :-

write('1. Card comparisons: '),

/\* How many cards are between eight of spades and jack of diamonds? \*/

test((lower\_card(C, (jack, diamonds)), lower\_card((eight, spades), C)),

C, Inf1, R1, length(R1, 9)),

write('2. Sorting your hand: '),

/\* How many sorted five-card poker hands have the queen of hearts and then

\* the six of diamonds specifically as the fourth card? \*/

test((H1 = [\_, \_, \_, (six, diamonds), \_], sorted\_hand(H1, 5), member((queen, hearts), H1)),

H1, Inf2, R2, length(R2, 8976)),

write('3. Four of a kind: '),

/\* How many four of kind hands contain the jack of diamonds? \*/

test((four\_of\_kind(H2), member((jack, diamonds), H2)), H2, Inf3, R3, length(R3, 60)),

write('4. Full house: '),

/\* How many full houses have a diamond as second card, and jack of spades as third? \*/

test((H3=[\_, (\_, diamonds), (jack, spades), \_, \_], full\_house(H3)), H3, Inf4, R4,

length(R4, 18)),

write('5. Flush: '),

/\* How many flushes have a ten as second card, and a six as a third card? \*/

test((H4=[\_, (ten, \_), (six, \_), \_, \_], flush(H4)), H4, Inf5, R5, length(R5, 96)),

write('6. Straight: '),

/\* How many straights start and end with a diamond? \*/

test((H5=[(\_,diamonds),\_,\_,\_,(\_, diamonds)], straight(H5)), H5, Inf6, R6, length(R6, 630)),

write('7. Straight flush: '),

/\* How many straight flushes do not contain an eight? \*/

test((straight\_flush(H6), not(member((eight, \_), H6))), H6, Inf7, R7, length(R7, 20)),

write('8. Three of a kind: '),

/\* How many three of a kind hands do not contain any spades? \*/

test((three\_of\_kind(H7), not(member((\_, spades), H7))), H7, Inf8, R8, length(R8, 7722)),

write('9. One pair: '),

/\* How many hands that have one pair have the suit pattern HSHSH? \*/

test((H8=[(\_,hearts), (\_,spades), (\_,hearts), (\_, spades), (\_, hearts)], one\_pair(H8)),

H8, Inf9, R9, length(R9, 1430)),

write('10. Two pair: '),

/\* How many sorted two pair hands have the suit pattern C\*C\*H ? \*/

test((H9 = [(\_, clubs),(\_, \_),(\_, clubs),(\_, \_),(\_, hearts)], two\_pair(H9)), H9, Inf10,

R10, length(R10, 858)),

/\* Total inferences \*/

Inf is Inf1 + Inf2 + Inf3 + Inf4 + Inf5 + Inf6 + Inf7 + Inf8 + Inf9 + Inf10.

Here is another test bed that you can use. This one is a bit heavier than the previous test bed, and can easily cause stack overrun terminations if your rules for the poker predicates are inefficient.

test\_all2(Inf) :-

write('1. Three cards: '),

/\* How many three card hands have the suit pattern HH\* ? \*/

test((H1=[(\_, hearts), (\_, hearts), \_], sorted\_hand(H1, 3)), H1, Inf1, R1, length(R1, 1300)),

write('2. No pairs: '),

/\* How many hands made of cards lower than nine don't contain any pairs? \*/

test((lower\_rank(R, nine),H11=[(R,\_)|\_],sorted\_hand(H11, 5), not(one\_pair(H11) ; two\_pair(H11) ; three\_of\_kind(H11) ; four\_of\_kind(H11) ; full\_house(H11))),

H11, Inf2, R2, length(R2, 21504)),

write('3. Sorted hand: '),

/\* No sorted hand contains its first card later again. \*/

test((sorted\_hand([H2|T],5), member(H2, T)), H2, Inf3, R3, length(R3, 0)),

write('4. Full house: '),

/\* How many full houses have seven of hearts as the middle card? \*/

test((H3=[\_, \_, (seven, hearts), \_, \_], full\_house(H3)), H3, Inf4, R4,

length(R4, 42)),

write('5. Flush: '),

/\* How many inferences are needed to find out that a flush can't start and end with two different suits ? \*/

test((HH = [(\_,spades),\_,\_,\_,(\_,clubs)], flush(HH)),HH,Inf5,R5,length(R5, 0)),

write('6. Straight: '),

/\* How many straights start with jack of spades? \*/

test((H5=[(jack,spades),\_,\_,\_,\_], straight(H5)), H5, Inf6, R6, length(R6, 255)),

write('7. Four of a kind: '),

/\* How many four of a kinds do not contain an eight? \*/

test((four\_of\_kind(H6), not(member((eight, \_), H6))), H6, Inf7, R7, length(R7, 528)),

write('8. Three of a kind: '),

/\* How many three of a kinds have some higher and some lower additional card? \*/

test( (H7 = [\_,(R,\_),(R,\_),(R,\_),\_], three\_of\_kind(H7)), H7, Inf8, R8, length(R8, 18304)),

write('9. One pair: '),

/\* No one pair is also two pair. \*/

test((H8=[(\_,diamonds),\_,\_,\_,\_],one\_pair(H8), two\_pair(H8)),

H8, Inf9, R9, length(R9, 0)),

write('10. High card: '),

/\* Using only cards lower than ten, how many hands have nothing better than a high card? \*/

test((lower\_rank(R, ten), X = [(R,\_)|\_],sorted\_hand(X, 5), not(one\_pair(X) ; two\_pair(X) ; three\_of\_kind(X) ;

four\_of\_kind(X) ; full\_house(X) ; flush(X) ; straight(X) ;

straight\_flush(X))), X, Inf10,

R10, length(R10, 53040)),

/\* Total inferences \*/

Inf is Inf1 + Inf2 + Inf3 + Inf4 + Inf5 + Inf6 + Inf7 + Inf8 + Inf9 + Inf10.