Assignment: 3
Due: Thursday, October 4th, 9:00 pm
Language level: Beginning Student
Files to submit: chess.rkt, complexmath.rkt, quaternion.rkt
Warmup exercises: 6.3.2, 6.4.1, 7.1.2
Practice exercises: 6.3.3, 6.4.2, 6.4.3, 6.5.2, 7.1.3, 7.5.1, 7.5.2, 7.5.3

• Make sure you read the OFFICIAL A03 post on Piazza for the answers to frequently asked questions.
• This assignment covers concepts up to the end of Module 04. Unless otherwise specified, you may only use Racket language features we have covered up to that point.
• Structure definitions, both the structure and field names, the names of functions we tell you to write, and symbols and strings we specify must match the descriptions in the assignment questions exactly. Any discrepancies in your solutions may lead to a severe loss of correctness marks. Basic tests results will catch many, but not necessarily all of these types of errors.
• Policies from Assignment 2 carry forward.

Here are the assignment questions you need to submit.

1. In this question you will perform step-by-step evaluations of Racket programs, as you did in assignment one. Please review the instructions on stepping in A01.

To begin, visit this web page:

https://www.student.cs.uwaterloo.ca/~cs135/stepping

Note: the use of https is important; that is, the system will not work if you omit the s. This link is also in the table of contents on the course web page.

When you are ready, complete the six stepping problems in the “Assignment questions" category, using the semantics given in class for Beginning Student.

2. We learned about the Posn type in the lecture. For this question we will define some arithmetic operators that work over Posns.

   a) Given two Posns \((x_1,y_1)\) and \((x_2,y_2)\). Then we define the product between these two Posns in the following way:

\[
(x_1,y_1) \cdot (x_2,y_2) = (x_1 \cdot x_2 - y_1 \cdot y_2, x_1 \cdot y_2 + x_2 \cdot y_1).
\]

Write a function posn-mult, which consumes two Posns and produces a Posn that is the result of the multiplication as defined above.
b) Given two \( \text{Posns} (x_1, y_1) \) and \((x_2, y_2)\). Then we define the division between these two \text{Posns} in the following way:

\[
(x_1, y_1)/(x_2, y_2) = ((x_1 \cdot x_2 + y_1 \cdot y_2)/(x_2^2 + y_2^2), (y_1 \cdot x_2 - x_1 \cdot y_2)/(x_2^2 + y_2^2)).
\]

Write a function \text{posn-div}, which consumes two \text{Posns} and produces the \text{Posn} that is the result of dividing the first parameter by the second, according to the above formula. Add appropriate requirements in your contract to ensure that any valid arguments consumed by this function will not result in division by zero.

c) Finally we will define the reciprocal of a \text{Posn} to be:

\[
\frac{1}{(x,y)} = \left( \frac{x}{x^2 + y^2}, -\frac{y}{x^2 + y^2} \right)
\]

Write a function \text{posn-reciprocal}, which consumes one \text{Posn} and produces the \text{Posn} that is the reciprocal of it, as given by the above formula. You must require that the provided \text{Posn} does not result in division by zero.

Please put your functions into a file called \text{complexmath.rkt}.

3. Chess is a popular two-player board game that dates back to the 6th-century. Chess is played on an \( 8 \times 8 \) grid using a variety of different pieces. Each type of piece in Chess has specific rules about how it can move. In order to discuss where pieces are located on a Chessboard and how they can attack we will consider a Chessboard’s rows and columns to be labelled from \( 1-8 \) and \( a-h \) respectively, as below.

![Chessboard diagram]

In this assignment we are concerned about only four types of pieces: bishops, rooks, queens, and knights. Please familiarize yourself with the rules of movement for these pieces here https://docs.kde.org/trunk5/en/extragear-games/knights/piece-movement.html.
Note: for this question we are only concerned about movement of pieces on an otherwise empty Chess board. That means you do not have to concern yourself with pieces getting blocked by other pieces.

This question uses the following structure to represent a position on a Chess board:

```
(define-struct square (row column))
;; A Square is a (make-square Nat Sym)
;; requires: 1 <= row <= 8
;;          column: (anyof 'a 'b 'c 'd 'e 'f 'g 'h)
```

Your code must use this same structure for representing a position on a Chess board.

(a) Provide a structure definition and data definition for `piece` with the following field names:
   i. pos, a valid `square` structure.
   ii. unit, a symbol where `unit` is (anyof ‘knight ’rook ’bishop ’queen).

(b) Write a template function `my-piece-fn` that consumes a `piece` and produces Any. Note that for this template function you only need to provide a contract and a body.

(c) Write a predicate function `valid-move?` that consumes two parameters, first a `piece` structure and second a `square` structure. The predicate shall determine if the location on the board (denoted by the second parameter) is a valid move for the piece (denoted by the first parameter) to make. Making no move is also a valid move.

   **Hint 1:** Being unable to calculate with the symbols used for columns is a significant problem. What can you do about that?

   **Hint 2:** This problem can be solved by comparing against specific positions on the board, however such solutions will receive few, if any, marks.

(d) Write a function `knight-next-move` that consumes a `piece` and produces a `square` to represent the next movement that `piece` make, based on a specific policy. This function requires that for the `piece` passed in the `unit` is a ‘knight. The function must follow the following policy for knight movement:

   - Our knights will always move closer to square H1 if possible.
   - If a knight may move to any squares that are closer to H1 than the square it currently occupies, then its move is to the square which is closest to H1.
   - If there are two squares that a knight may move to that are equidistant to H1 then it chooses the square with the lower row number (e.g. in the following diagram, the knight would choose to move to F2 over G3).
   - If there are no squares that are available to the knight that are closer to H1 than it’s current position, then its next move is not to move.
4. **BONUS QUESTION (5%)** Sir William Rowan Hamilton introduced in the 19th century the notion of quaternions. These are of the form \( x_0 + x_1i + x_2j + x_3k \), where \( x_0, x_1, x_2, x_3 \) are numbers and \( i, j, k \) are symbols. For two numbers \( a, b \) (both not equal to zero), the multiplication rules of \( i, j \) and \( k \) can be described by the following multiplication table.

\[
\begin{array}{ccc}
  i & j & k \\
  i & a & k & aj \\
  j & -k & b & -bi \\
  k & -aj & bi & -ab \\
\end{array}
\]

**Remark:** As we do not have commutativity here, the order matters. You read this table as “row times column”, not the other way around. For example, in this table you can deduce that the product \( i \cdot k \) will be equal to \( aj \).

The symbols \( i, j \) and \( k \) commute with any number, i.e. for every number \( c \) we have \( ci = ic \), \( cj = jc \) and \( ck = kc \).

**Example:** Let us assume the classical quaternions given with \( a = -1, b = -1 \). Then we have

\[
(2 + 3i + 0j + 0k) \cdot (5 + 6i + 0j + 0k) = -8 + 27i + 0j + 0k.
\]

(You might recognize a certain relation to the multiplication we defined for posn structures above while trying to understand this example.)

You are provided the following structure for an element \( x_0 + x_1i + x_2j + x_3k \) in the quaternions.

```
(define-struct quaternion (cc ic jc kc))
;; A Quaternion is a (make-quaternion Num Num Num Num).
```
You can copy this definition into your code (but do not do “Copy&Paste”, as mentioned in section 1.2 of the Style guide).

The quaternion structure, as you can see, contains 4 fields

- **cc** – representing the constant coefficient \( x_0 \).
- **ic** – representing the coefficient of \( i \), i.e. \( x_1 \).
- **jc** – representing the coefficient of \( j \), i.e. \( x_2 \).
- **kc** – representing the coefficient of \( k \), i.e. \( x_3 \).

Write a function `quat-mult`, which consumes two numbers \( a, b \) and two elements \( q_1 \) and \( q_2 \) in the quaternions. This function shall produce the result of the multiplication \( q_1 \cdot q_2 \).

As usual, this is an all-or-nothing type bonus question. No partial marks are awarded.

Please put your solutions for this bonus question into a file called `quaternion.rkt`