CS135F16 Adventures of Bob the Knight

XCHONG (Last edit: N/A)

Final: Thursday December 15th, 12:30-15:00

Prologue

Once upon a time, there was a kingdom that faced the greatest economical crisis ever recorded in history; the Great Student Loan Depression. In order to save the kingdom, the queen sent out all the knights and heroes to find a solution to the crisis. The following text details the adventures of one such brave warrior, seeking answers from a mythical document known as The Degree. Hidden somewhere beyond the land of Finals, it is rumoured that whoever finds this document will be granted infinite wisdom, or at least a decent career. Join Bob the Knight as he goes forth into the land of Finals, and help him save the kingdom! :D

Module 1: Arithmetic Functions

Goals: Understand basic Racket syntax, forming expressions, error messages. Be comfortable with the terms: "function, parameter, application, argument, constant, expression". Be able to define and use simple arithmetic functions.

Bob the Knight is getting ready to travel to the land of Finals. Over there, the currency is exclusively in gil, so Bob must change his dollars to gil first. However, the rate of exchange varies depending on the weight of the king of Finals. Here is the rate, given the king's weight in pounds (w):

$$\$1 : X\text{gil}, \quad X = \max(A, B)$$

$$A = \text{quotient of } C/D$$

$$B = \text{quotient of } D/C$$

$$C = \left\lfloor (\sqrt{w + e^{1.605}})^{\pi - e} + \frac{|w^e - 1.605^\pi|}{\max(w, 160.5)} \right\rfloor$$

$$D = \left\lceil \min(\frac{\pi}{2}, \frac{160.5}{w/2}) \times \frac{e \times \pi}{1 + (\text{remainder of } \frac{\text{ceiling}[w]}{\text{floor}[\min(\frac{w}{2}, e^\pi)]})} \right\rceil$$

Note: $\lfloor \quad \rfloor$ are floor brackets, $\lceil \quad \rceil$ are ceiling brackets.

Bob has $100. The king's weight is reported at 160 pounds today, but is predicted to be at around 161 pounds tomorrow (the king is known to stress-eat around this time of year). On which day would it be better for Bob to exchange his money, and how much gil would he receive?

Write the function dollars-to-gil which consumes a positive real number of dollars and a positive real number representing weight, and produces the amount of gil exchanged for based on the exchange rate formula given above. Assume that the king's weight is never less than 20 pounds.

Module 2: Booleans, Conditionals, String Functions and In/Equalities

Goals: Understand what the design recipe is, and its components. Understand how to write meaningful examples and tests. Understand how to use Boolean data, and conditional expressions. Know how to handle string data types. Notice opportunities to use helper functions where appropriate.

After crossing the border, Bob the Knight is faced with a difficult decision: should he take the main road that goes across the plains, the smaller road that goes through the forest, or the dangerous shortcut across the swamp? Or maybe just sleep on it? In order to make his decision, he considers a few things. Does he have enough money to rent a horse? Is he hungry? Is it going to be raining? Is it the weekend? Somehow, Bob comes to a decision based on these factors, and this is his thought process:

```
(define (travel-plans money? hungry? raining? weekend?)
  (cond [hungry?
         (cond [raining?
                (cond [weekend? 'sleep]
                      [else (cond [money? 'plains]
                                  [else 'forest])])]
                [else 'swamp])]
         [raining?
         (cond [money?
                (cond [weekend?
                       (cond [(not raining?) 'forest]
                             [else 'sleep])]
                       [else 'plains])]
                [else (cond [(not weekend?)
                             (cond [raining?
                                    (cond [weekend? 'sleep]
                                          [else 'forest])]
                                    [else 'forest])]
                             [else (cond [(not raining?) 'forest]
                                         [else 'sleep])])])]
         [else (cond [(not money?)
                      (cond [(not weekend?)
                             (cond [raining?
                                    (cond [weekend? 'swamp]
                                          [else 'sleep])]
                                    [else (cond [hungry? 'plains]
                                                [else 'forest])])]
                             [else (cond [raining? 'swamp]
                                         [else 'forest])])]
                      [else (cond [money?
                                   (cond [weekend? 'forest]
                                         [else (cond [weekend? 'sleep]
                                                     [(not hungry?) 'plains]
                                                     [else 'swamp])])]
                                   [else 'swamp])])])))
```

Unfortunately, Bob's thought process is so confusing that even he cannot understand what he wants to do. Help him by simplifying the above code into a function with a single conditional containing no more than four cases. If Bob is not hungry, and Bob has enough money to rent a horse, and if he is planning to travel on a sunny weekend, then will he take the swamp, forest, plains, or sleep at the inn for another night?

Rewrite the function travel-plans which consumes four boolean values in the same order as above, and produces a symbol representing Bob's decision. Use cond only once, and do not use more than four cases. Use of and/or is allowed.

Bob decides to go through the forest to get to the capital. Before leaving the inn, the innkeeper passes him an old map, but all the names of the roads are different from the current names. Help Bob decipher the names on the map using the following instructions that the innkeeper gave:

If the name is five letters or shorter, just take off the first letter.
If the name is 6 to 9 letters long, append "COLUM" to the front of the last three letters, removing all the letters before the last three.
If the name is more than 9 letters long, and the first letter is before M, then append the first five letters onto the last four.
If the name is more than 9 letters long, and the first letter is after L, then append the last four letters onto the first five.

If the path that Bob wants to take on the old map is HERB, MOUNTAINWEST, VICTORBIA, and BEECHDENLAWN, then what are the new names of those roads?

Write the function new-name that consumes a string and produces a new string according to the above rules.

Module 3: Tracing

Goals: Understand the substitution-based semantic model of Racket, and be prepared for future extensions. Be able to trace the series of simplifying transformations of a Racket program.

As Bob walks through the forest, looking for the right paths to turn on, he starts getting very bored. Thankfully, the newspaper he brought with him has a bunch of fun puzzles to solve. And since it's the weekend, there's even more puzzles in the weekend section!

Module 4: Structures, Mixed Data

Goals: Be able to write code to define a structure, and to use the functions that are defined when you do so. Understand the data definitions we have used, and be able to write your own. Be able to write the template associated with a structure definition, and to expand it into the body of a particular function that consumes that type of structure. Understand the use of type predicates and be able to write code that handles mixed data. Understand (anyof . . .) notation in contracts.

As Bob is finishing up the last puzzle, he hears loud cursing coming from up ahead. In front, there is a merchant with all of her wares scattered on the ground, her small cart toppled over in a ditch. Being a helpful knight, Bob offers to help her reorganize her merchandise. The merchant sells juice, soda and tea, and keeps each type sorted into crates.

```
(define-struct juicecrate (apple orange lemon))
(define-struct sodacrate (rootbeer gingerale vanillacola)
(define-struct teacrate (green black herbal bubble)

;; A JuiceCrate is a (make-juicecrate Nat Nat Nat)
```

Write the data definitions for SodaCrate and TeaCrate in the same way as JuiceCrate, and write a template for each structure. Then write a template for a function that consumes any of these three crate types. Remember that a template includes a contract.

The merchant, Alice, is glad to accept Bob's help. Help Bob count her inventory, as she passes each crate for him to count.

Write the function count-crate that consumes one of JuiceCrate, SodaCrate or TeaCrate, and produces the sum of the drinks in that crate. If the sum is odd, add one to the final sum.

Module 5: Lists, lists, lists...

Understand the data definitions for lists, how the template mirrors the definition, and be able to use the template to write recursive functions consuming this type of data. Understand box-and-pointer visualization of lists and (listof . . .) notation in contracts. Understand strings, their relationship to characters and how to convert a string into a list of characters (and vice-versa). Be comfortable with lists of structures, including understanding the recursive definitions of such data types, and you should be able to derive and use a template based on such a definition.

Alice's horse ran away when she crashed her cart, so Bob hooks up her cart to his, and agrees to bring her to the capital in exchange for a small crate of bubble tea. They end up talking and Bob tells her about his quest to claim the Degree. Being a practical merchant, she asks him if he's well stocked up on the equipment he will need for his journey. Bob shows her the list of things he brought...

(cons (cons 'pants (cons 'boots empty)) (cons (cons 'underwear (cons 'shirts empty)) (cons 'bucket (cons 'sword (cons (cons 'potion (cons (cons 'hi-potion (cons 'lotion empty)) empty)) (cons 'water (cons 'waterloo-approved-calculator empty)))))))

Bob also brought some stranger things:

'(list (+ 2 3) '(list (cons 1 (cons 2 empty))) "words" () pi pie 'pie "pie")

Convert the above lists into (list ...) form. Check your answers in Racket.

Alice wants to help Bob organize his stuff, because he's clearly not a very organized person. Bob has his stuff organized into boxes, but he needs a more methodical way to sort through them.

;; A Box-of-Stuff is one of:
;; * empty
;; * (listof Sym Box-of-Stuff)

Alice points out that his template could be made more specific, since a box of stuff would never be a box with nothing in it.

Help Bob rewrite his list data definition as a non-empty list data definition, and write a template for it as well.

Bob and Alice are almost through the forest. But just as they reach the end, they hit a fork in the road. The sign is written in a foreign language that Bob has never seen before, and it isn't on his map. Alice thinks she knows how to translate it though, with the help of her dictionary which has a list of translations.

(define-struct translation (key value)
;; A Translation is a (make-translation Char (listof Char))

Write the data definition and template for a (listof Char). Then write the template for a Translation. Finally, write the template for a Dictionary, which is a non-empty (listof Translation).

Alice's dictionary of translations is interesting because it translates each letter into a group of letters. However, the group of letters in her dictionary are backwards. Meaning, if the key #\A has the value (list #\D #\A #\S) and the key #\B has the value (list #\S #\S #\E #\N), then the word "AB" would be translated as "SADNESS".

Write the function translate, which consumes a string representing the word being translated and a Dictionary, and produces the translation of the word by the rules given above.