Post-Mortem

CS135 Winter 2024, Midterm

Question 1

—— Almost no students got (b) correct. A majority missed the 3rd `cons` before 'b or missed the last `empty`

—— Many students wrote `"CS13"` instead for (f)

—— A large number of students didn't point out it was an error for (c). instead they wrote `empty`

Question 2

—— Calculates `(not ...)` first instead of a function

—— Calculates `(and ...)` `(or ...)` in one step

—— `(or)` remains instead of returning `true`; `(and true)` returns `true` directly

Question 3

—— Common Q3abc errors include the wrong types for the "product" parameters such as using a symbol.

—— Common Q3de errors include using `(anyof Sym Str)`, had right clauses but wrong order

Question 4

Q4ab: Common Errors

—— Thinking that `Player` or `Game` could be `empty`

—— Writing listof instead of `list`

Q4c: Common Errors

—— Very few students had their template consider all 4 parts of the game. Some accessed the fields partially, while many did not access the fields at all.

—— A majority of students did not include a contract.

—— The function name usually was not descriptive enough. A common error was not making clear that the template was for a list.

—— Some students said the base case was `empty`.

Q4d: Common Errors

—— Many students mistakenly use `length` function, in some cases their own implementation of the function

—— I encountered many cases where they forgot to check whether the input is a list/cons or not

—— There were several cases where they did not check if it is an empty list or if there is 1 item in the list and jumped directly to checking for list of 2 items.

Question 5

Q5a: Common Errors

—— When checking if there are enough items in `lop` to sell, many student forgot to include the case when the number of product to be purchased in `pr` is exactly the same as the number of items left in `lop` (in which you would need to still update the list).

—— Many student forgot to cons `(first lop)` to the recursive call in the else condition.

—— Some student did not construct the list correctly if there were enough items to sell (i.e., `lop` was updated incorrectly).

—— Although no marks were taken off, note that because the product names are unique in `lop`, if enough items can be sold, the updated product simply can be cons-ed onto `(rest lop)` instead of a recursive call.

Q5b: Common Errors

—— Most students had something completely incorrect, where no recursion was being done.

—— Incorrectly using the earlier defined list `"mylist"` as part of their implementation

—— Incorrectly returning `empty` in the base case, or checking to see if both lists were empty in the base case.

Question 6

—— The majority of students didn't use is-it from part one and tried to implement their own helper to do the same thing

- When implementing the helper a lot of students don't check if `loc` is `empty` before they call first on it
- Some student use `substring` or check the first 1 or 2 characters only

—— Some students use `string=?` or `symbol=?` incorrectly to check a match with a code

—— Some students had an incorrect base case on code instead of `loc`

Question 7

—— many students had the wrong base cases. A base case is needed for when n is zero and if the list is `empty`.

—— many students did not check if `(first lst)` was an integer before checking if it was even.

—— many students did not decrement n in their recursive call.

Question 8

—— many students did not use a helper function that consumes the `(first lst)` as a parameter.

—— many students tried comparing `(first lst)` and `(second lst)` in the main function, but this will lose track of the initial first element of the list when you recurse.