Post-Mortem
CS135 Fall 2023, Midterm

Question 1

Part a:
- Many students had an extra (list ...).

Part b:
- Some students forgot to cons the last empty.

Part e:
- Many students missed the outermost first.

Part f:
- Many students missed the quotes around the string (no marks were deducted this time).

Question 2

Part a:
- Many students substituted both g’s with 10. Only the first g should have been replaced.

Part b:
- Some students did not spot that symbol= would throw an error.

Question 3

Part b:
- Many students wrote true instead of false in the first blank, or had a string not of length 7 in the second blank.

Part d:
- Many students did not give a description of the error produced by the function, or described the error incorrectly.
Question 4

Parts a, b:
- Many students defined x and y coordinates as \texttt{Int} instead of \texttt{Num}.
- Many students forgot to write the contract for the template.
- Some students considered only the \texttt{Point} and did not check any of the elements inside the \texttt{Point}.
- Many students neglected to separate cases for accessing x and y coordinates.

Part d:
- Many students returned \texttt{true} instead of the \texttt{Point}.
- Many students did not check that the second element also had to be a list of 2 values, and if these values were numbers.
- Some students forgot to check that the numbers were in-range.

Question 5

Part a:
- Many students used \texttt{(rest (rest ...))} and forgot to extract the third element using \texttt{first}.
- Some students forgot an edge case test for values of 32 or 100.
- Many students forgot to check whether the consumed arguments for \texttt{symbol=?} are actually symbols.

Part b:
- Some students did not use \texttt{grade-value} written in part (a) to retrieve the grade, leading to improper handling of 'DNW cases.
- Some students forgot to filter out the students by their student number.
- Many students did not handle 'NA case properly (either missing a check for an empty list of CR or a zero count).
- Some students found the average by dividing by some constant (2 or 5) rather than the count of matching records, or nested the division within the recursion instead of using a wrapper.

Question 6

- Many students used \texttt{(first (rest (first uwcal)))} instead of \texttt{(first (second (first uwcal)))} or \texttt{(first (first (rest (first uwcal))))} to access the department name.
- Many students did not access the first element of the \texttt{UWCAL} association list before applying \texttt{first}, \texttt{rest} and/or \texttt{second} correctly.
- Many students missed the second parameter in the recursive application.
- Some students used \texttt{append} on strings.
- Many students used \texttt{symbol=?} and \texttt{=} instead of \texttt{string=?} when comparing strings.
Question 7

Part a:
- Many students were missing the requirements section.
- Many students assumed that the consumed value was a string or a symbol instead of a list of characters.
- Some students had incorrect base cases.
- Some students attempted a solution that works when the second list is longer and swapped the order of their base cases, i.e. they first checked if the first list was empty instead of the second list.
- Many students forgot that you could compare characters with \texttt{char=?}.
- Some students mixed up prefix and substring.

Part b:
- Many students did not check if the consumed value is a string.
- Many students (whose solutions for part (a) did not work if the second list is longer) neglected to check the length of the consumed string is greater than or equal to the length of “gum-gum” (7) before calling \texttt{has-prefix}.
- Many students did not convert the consumed string and GG to a list of characters before passing them to \texttt{has-prefix} as arguments.
- Many students forgot that \texttt{gum-gum} should only consume one parameter.
- Some students used 6 for the length of GG instead of 7.
- Many students incorrectly assumed that \texttt{gum-gum} consumes a list (it should consume Any).

Question 8
- Some students swapped the order of the base cases (i.e. first checking if the list is empty, then checking if n is equal to 0).
- Some students did not check when n is equal to 0, causing the recursion to continue when n is less than 0.
- Some students used the disallowed built-in length function or wrote their own length function, which does not satisfy the question’s requirement of using recursion on both n and the consumed list in lockstep.