

COMP5111 Assignment 2 Task 2

Problem

The problem I use LLMs to solve is finding faults in function scope and automatic debugging.

Prompt

There is a piece of erroneous code:

```
/**
 * Parses a string to extract a map of name/value pairs.
 *
 * @param str the string containing a sequence of name/value pairs
 * @param separators the separators used to separate the name/value pairs
 * @return a map of name/value pairs
 */
public Map<String, String> parse(final String str, char[] separators) {
    if (separators == null || separators.length == 0) {
        return new HashMap<>();
    }
    char separator = separators[0];
    if (str != null) {
        int idx = str.length();
        for (char separator2 : separators) {
            int tmp = str.indexOf(separator2);
            if (tmp != -1 && tmp < idx) {
                idx = tmp;
                separator = separator2;
            }
        }
    }
    return parse(str, separator);
}
```

We have found that the fault is located among the lines:

```
if (tmp != -1 && tmp < idx) {
    idx = tmp;
    separator = separator2;
```

Please further locate the fault and fix only one line of the above code to make the entire function correct.

Justification

I directly ask GPT to locate the fault, and emphasize on "only one line" to make sure that the model focus on one line.

Below is an example of GPT response:

To fix the code, change the line `if (tmp != -1 && tmp < idx) {` to `if (tmp != -1 && tmp <= idx) {`. This ensures that if two separators are found at the same index, the first separator encountered is used as the separator for parsing the string. The corrected code is shown below:

```
/**
 * Parses a string to extract a map of name/value pairs.
 *
 * @param str the string containing a sequence of name/value pairs
 * @param separators the separators used to separate the name/value pairs
 * @return a map of name/value pairs
 */
public Map<String, String> parse(final String str, char[] separators) {
    if (separators == null || separators.length == 0) {
        return new HashMap<>();
    }
    char separator = separators[0];
    if (str != null) {
        int idx = str.length();
        for (char separator2 : separators) {
            int tmp = str.indexOf(separator2);
            if (tmp != -1 && tmp <= idx) {
                idx = tmp;
                separator = separator2;
            }
        }
        return parse(str, separator);
    }
}
```