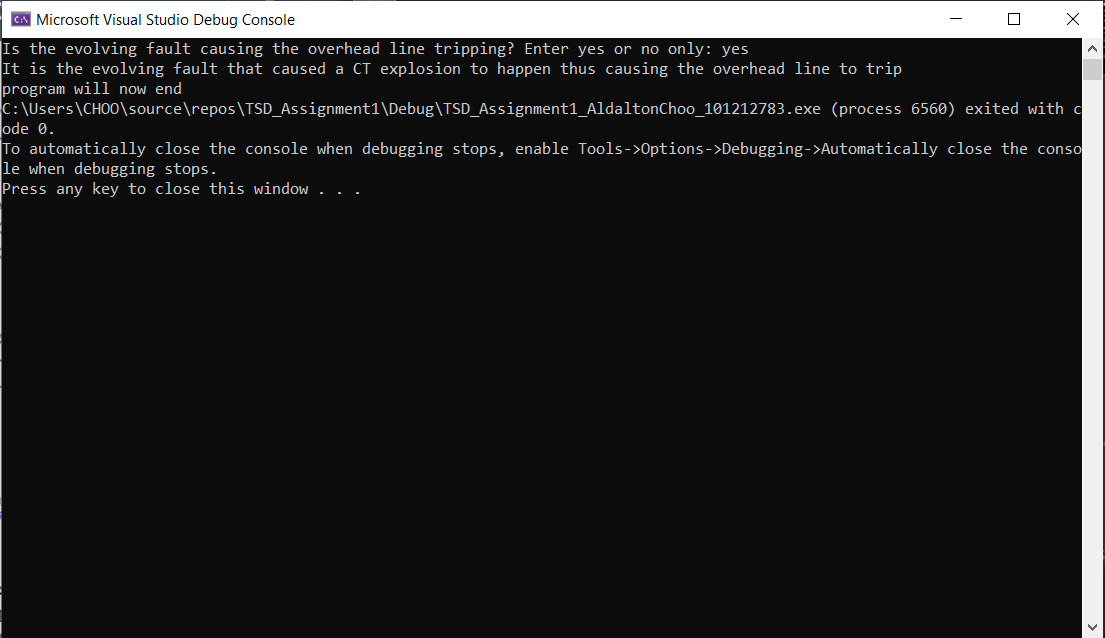
**Reporting Tasks**

|  |  |  |
| --- | --- | --- |
| Variable | Identifier | Type |
| Evolving Fault | e\_fault | String |
| 𝜆, Gradient or rate of change of the curve | v\_gradient | Double |
| Voltage Dip | v\_voltageDip | Double |
| Permanent Fault | p\_fault | String |
| Δ𝑡௙,Time interval between the last neutral current distortions and a flashover (real number) | v\_timeInterval | Double |
| Option to allow user to retry the program | retry | String |

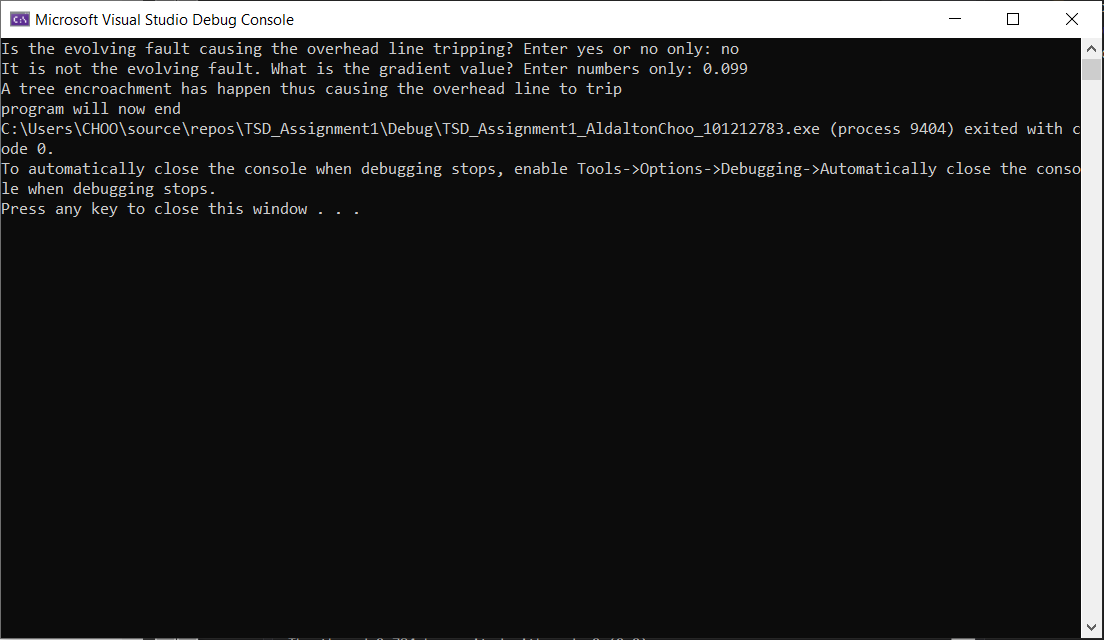
1. This program obtains the users input progressively when they are needed in each node in the decision tree. This is because each decision node is affected by different inputs enter by the user. For example, the user is asked whether it was the Evolving fault causing the overhead line tripping, if the user entered yes then it will show an output saying a CT explosion is the reason why the overhead line tripped and if the user entered no it will move on to the next decision process to find out the cause of the overhead line tripping. This allows the user to obtain the cause of overhead line tripping much faster rather than having to wait until the end of the program which is a waste of time when one of the earlier user inputs are the reason why the overhead line tripped.

**Testing Tasks**

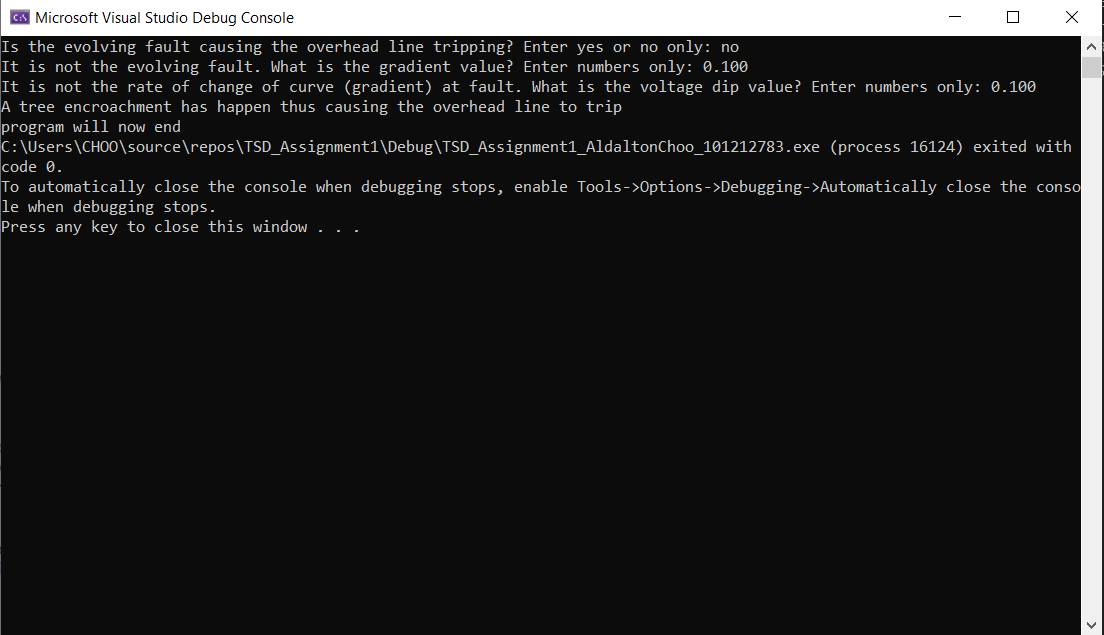
a)



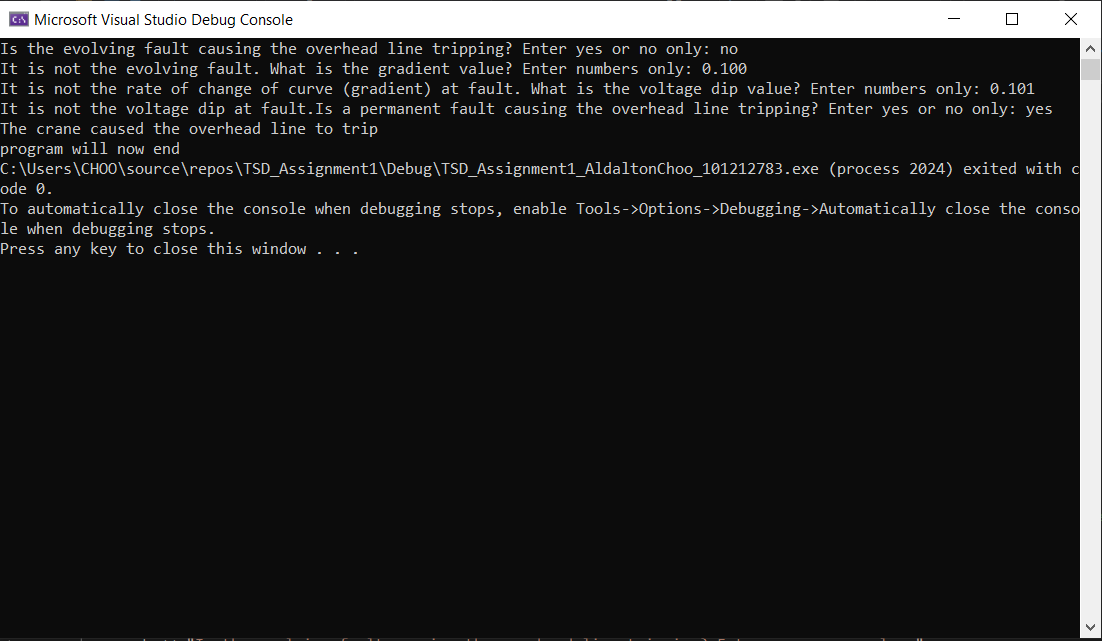
*Figure (1): Output 1 showing CT explosion is the cause for overhead line tripping*



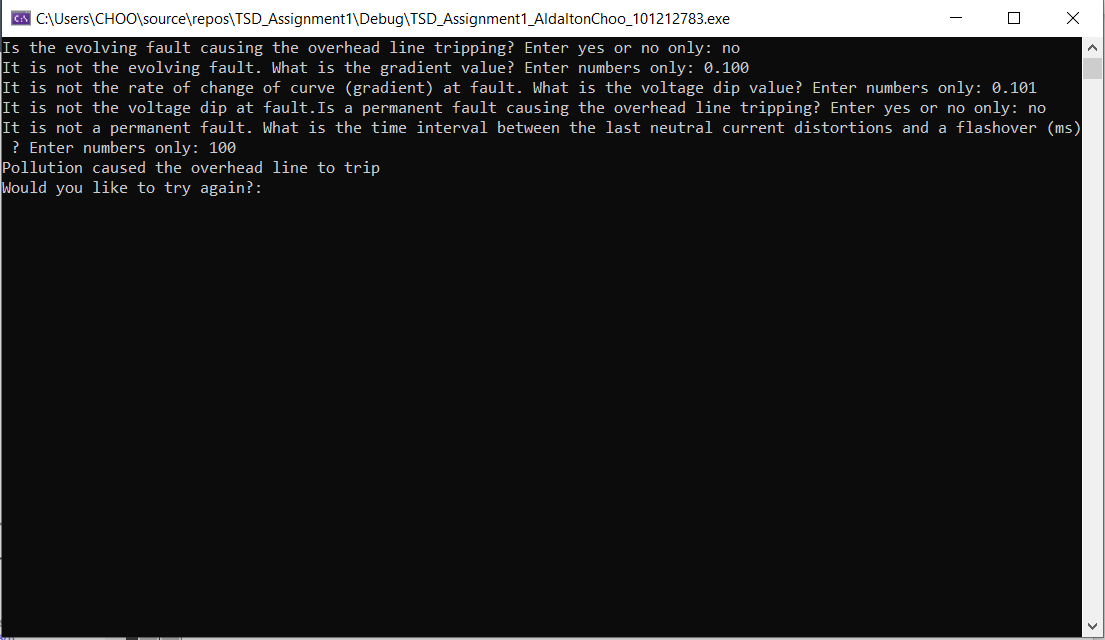
*Figure (2): Output 2 showing Tree encroachment is the cause for overhead line tripping*



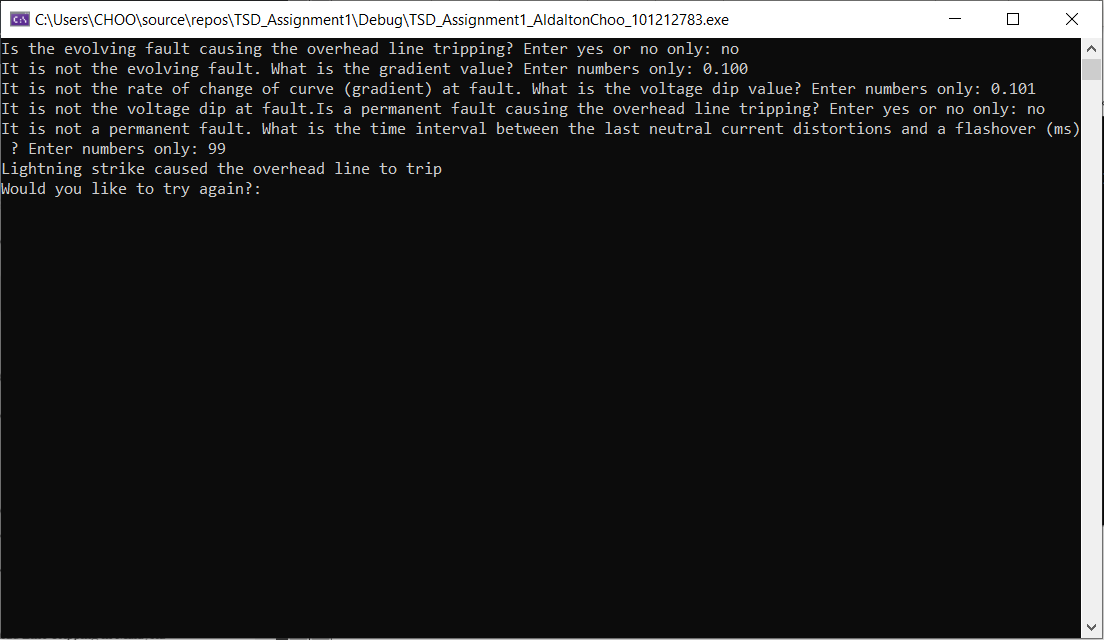
*Figure (3): Output 3 showing Tree encroachment is the cause for overhead line tripping*



*Figure (4): Output 4 showing A crane is the cause for overhead line tripping*

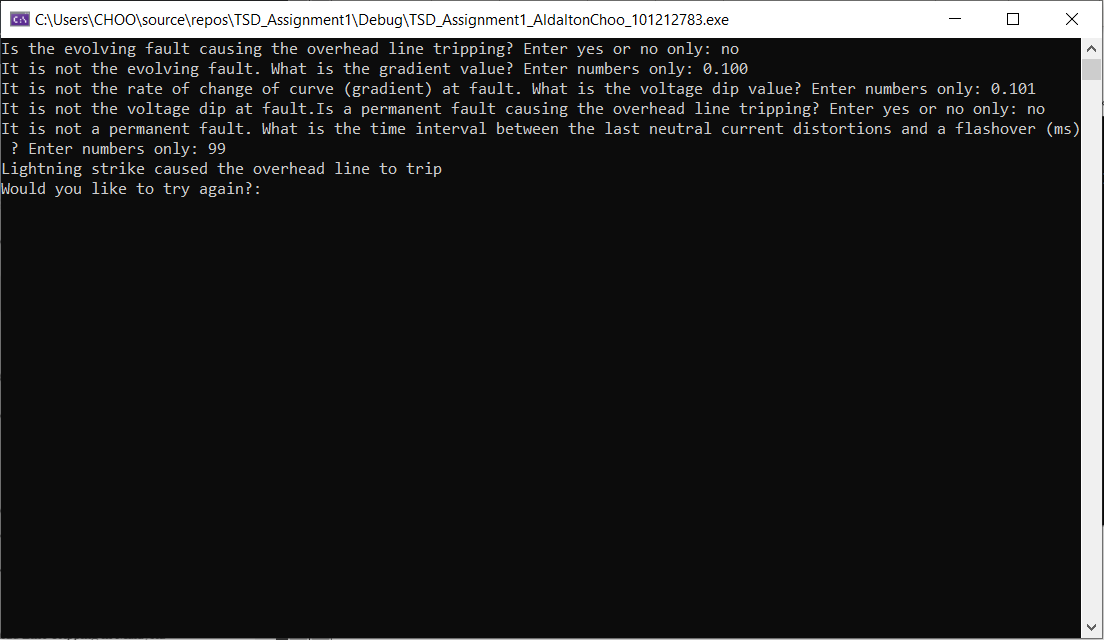


*Figure (5): Output 5 showing pollution is the cause for overhead line tripping*

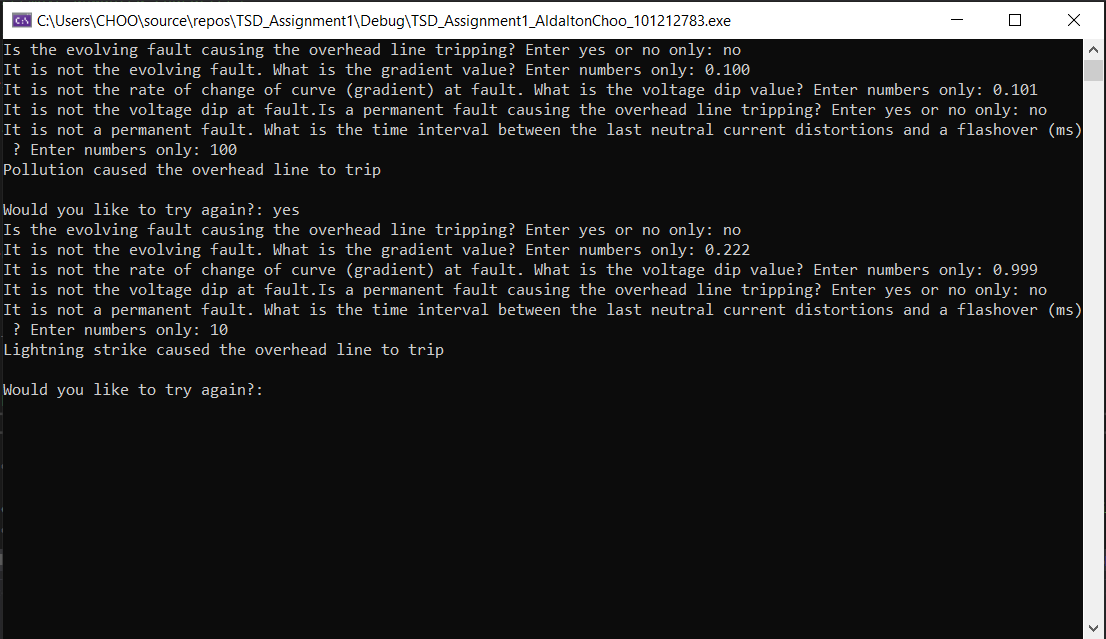


*Figure (6): Output 6 showing Lightning strike is the cause for overhead line tripping*

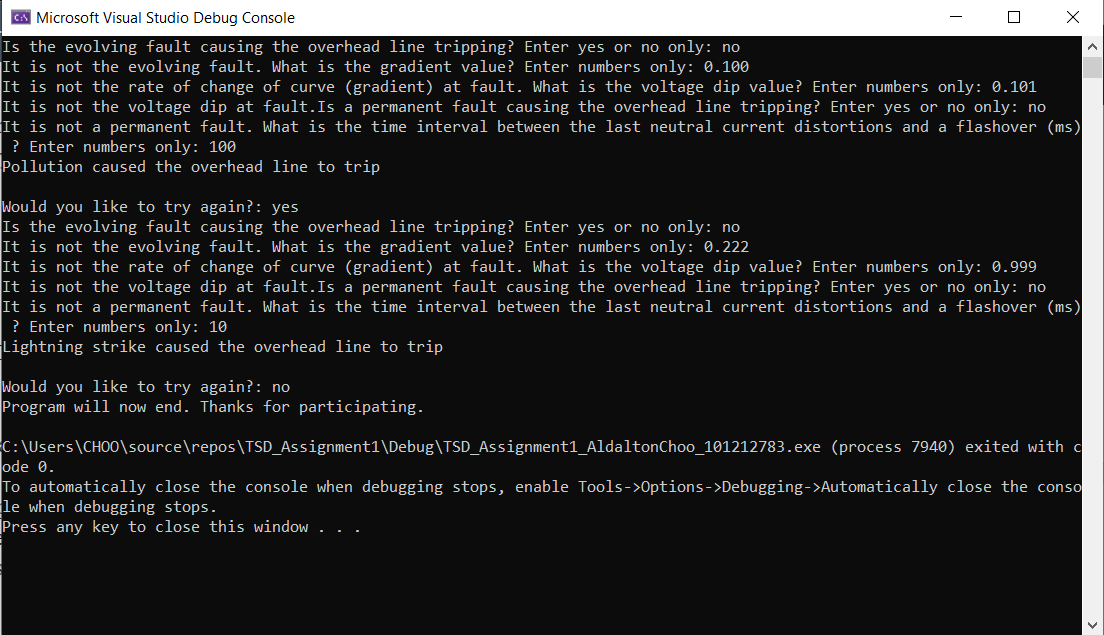
b)



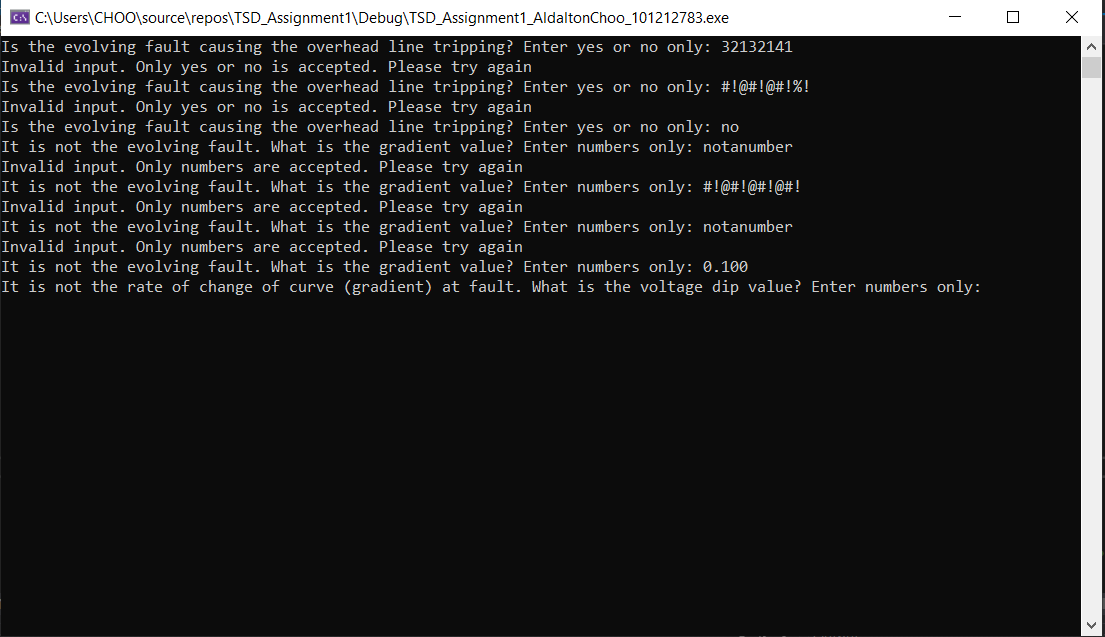
*Figure (7): Challenge task 1 part 1 Asking user if they want to repeat the process*



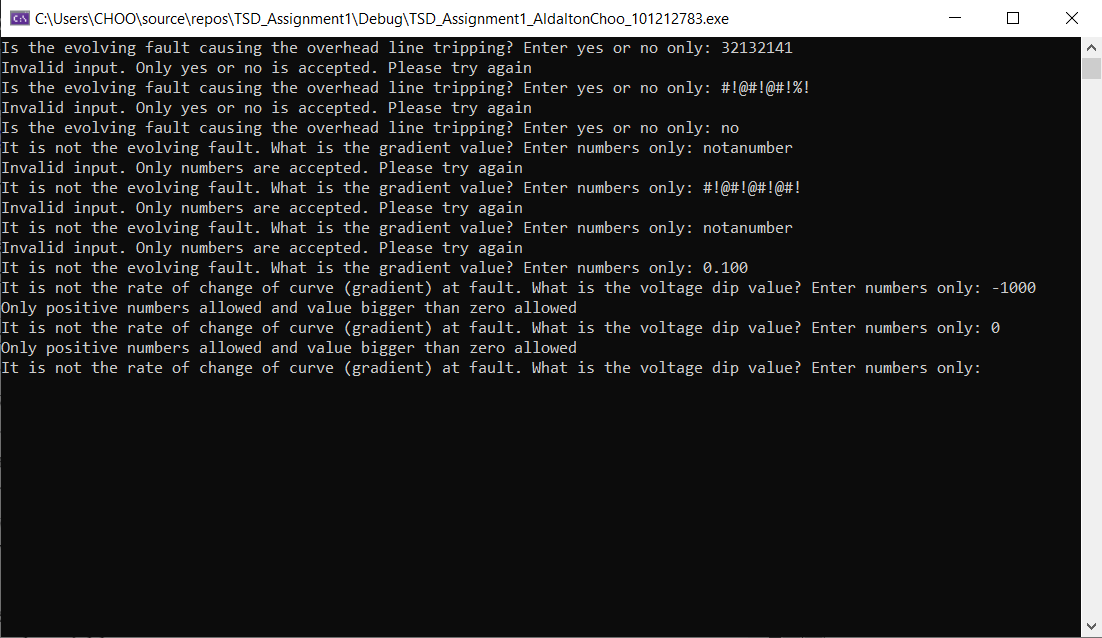
*Figure (8): Challenge task 1 part 2 If user enters yes, the whole process is repeated*



*Figure (9): Challenge task 1 part 3 If user enters no, the program will terminate*

**

*Figure (10): Challenge task 2 part 1 input validation for both binary value and real number input*



*Figure (11) Challenge task 2 part 2 users are not allowed to enter negative values or value smaller than zero.*