BLG 223E - Homework 1 Report

Melike Beşparmak - 150220061

April 2024

1 Introduction

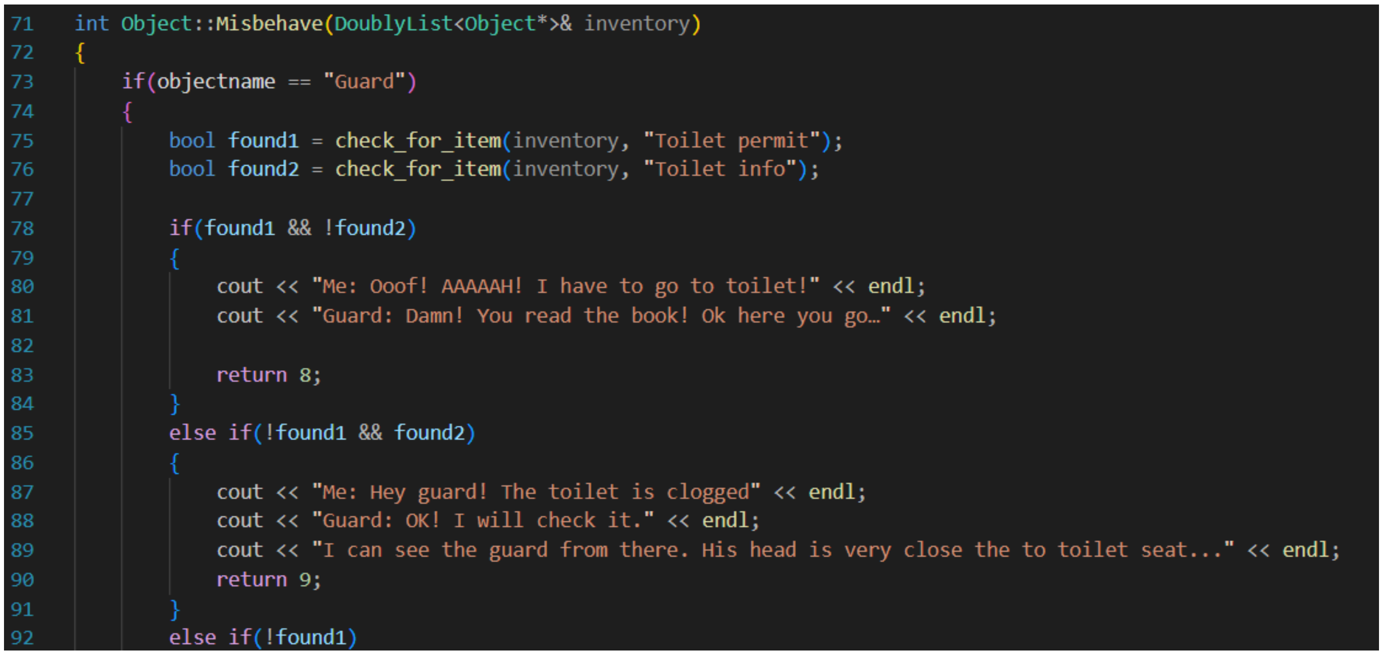
The homework aimed to escape the prison without any user input. Double linked list and objects implementations were provided and the the only missing part was the DFS algorithm to solve the problem.

2 Method

As stated in the assignment, stack, and the Depth First Search (DFS) algorithm had to be used. Therefore, I imitated a stack by using only *addBack* and *removeBack* functions of the given linked list implementation. The code tries every possible combination of actions and objects available in the room with two nested for loops. The advance function is used to detect if anything changed with the previous state. A positive return value indicates a change meanwhile zero means no change and -1 means game over. States with positive return values are added to stack and the last element is removed if no possible moves are left. The process repeats with the new last state. I preferred using two state variables for buffer state and current state. Since I want to add every possible action to the stack, I keep a current state and the changes are made to the buffer state. If the buffer state is affected, a new element is added to the stack, and the buffer is set to current. This process repeats until the win flag of the current state is 1.

3 Discussion

I suspect the objects file had a problem with the case where the return value is 9. The pattern we need to follow to escape the prison successfully is: 1 - 2 - 3 - 4 - 5 - 8 - 6 - 7 - 11 - 13 - 9 - 14 - 10 - 13 - 15 - 12. The transition from 13 to 9 never happens. The needed flags are Toilet permit and Toilet info, however when the states are updated Toilet permit vanishes and the statement is never satisfied. Consequently, an infinite loop occurs. Therefore, I changed the 85th line of the code given below. I believe a check for the second flag is enough but I did not want to delete the first flag since it is testable this way for both with and without '!'.



```cpp
71  int Object::Misbehave(DoublyList<Object*>& inventory)
72  {
73      if(objectname == "Guard")
74      {
75          bool found1 = check_for_item(inventory, "Toilet permit");
76          bool found2 = check_for_item(inventory, "Toilet info");
77
78          if(found1 && !found2)
79          {
80              cout << "Me: Ooof! AAAAAH! I have to go to toilet!" << endl;
81              cout << "Guard: Damn! You read the book! Ok here you go…" << endl;
82
83              return 8;
84          }
85          else if(!found1 && found2)
86          {
87              cout << "Me: Hey guard! The toilet is clogged" << endl;
88              cout << "Guard: OK! I will check it." << endl;
89              cout << "I can see the guard from there. His head is very close the to toilet seat..." << endl;
90              return 9;
91          }
92          else if(!found1)
```

Figure 1: Objects File - Line 85*