```cpp
// /
//Name                          Sai Chaitanya Kilambi
//Course                        CPSC 131 Data Structures, Fall, 2022
//Assignment                    No.4 question:4
//Due date                      09/21/2022
// Purpose:
// This program prints out the the table of students with GPA > 3.0 and GPA <=3.0
//
// list of libraries
//
//importing the required libraries

#include<iostream>
#include<fstream>
#include<string>
#include<iomanip>

template <class T, int n>
class Stack
{
private:
    T Element[n];
    int counter=0;

public:
    void clearStack()
    {
        counter = 0;
    }
    bool emptyStack()
    {
        return (counter == 0 ? true :
            false);
    }
    bool fullStack()
    {
        return (counter == n ? true :
            false);
    }
    void pushStack(T x)
    {
        Element[counter] = x;
        counter++;
    }
    T popStack()
```

```
46      {
47          counter--;
48          return Element[counter];
49      }
50
51  };
52
53
54
55  int main(){
56      //  name of the file
57      std::string filename = "data.txt";
58      //  variables
59      std::string word;
60      float gpa;
61      //  stacks
62      Stack<std::string,80> lowStack;
63      Stack<std::string,80> highStack;
64      //  open file to read word by word
65      std::fstream file;
66      file.open(filename.c_str());
67      //  read file till EOF
68      while(file>>word){
69          //      read gpa of student 'word'
70          file>>gpa;
71          //      push student into stack accordingly
72          if(gpa<=3.0){
73              lowStack.pushStack(word);
74          }
75          else{
76              highStack.pushStack(word);
77          }
78      }
79      //  print output
80      std::cout<<"  GPA <= 3.0    GPA > 3.0"<<std::endl;
81      std::cout<<"_____________________"<<std::endl;
82      while(!lowStack.emptyStack() && !highStack.emptyStack())
83      {
84          std::cout<<std::setw(15)<<std::left<< lowStack.popStack();
85          std::cout<<std::setw(10)<<std::right<< highStack.popStack()
              <<std::endl;
86
87      }
88      //  if lowStack is still not empty
89      while(!lowStack.emptyStack())
90      {
91          std::cout<<std::setw(15)<<std::left<< lowStack.popStack()
              <<std::endl;
92
```

```
 93      }
 94      //  if highStack is still not empty
 95      while(!highStack.emptyStack())
 96      {
 97          std::cout<<std::setw(25)<<std::right<< highStack.popStack()
               <<std::endl;
 98
 99      }
100      //  close the file (IMPORTANT)
101      file.close();
102      return 0;
103  }
104
105
```