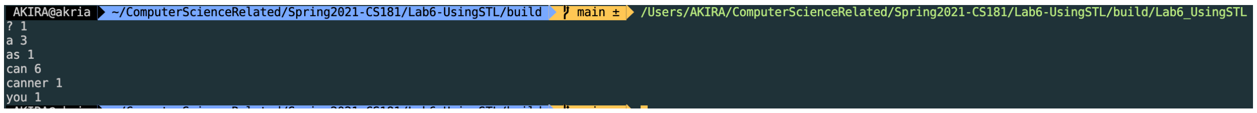
CS181 Lab6

Q1:

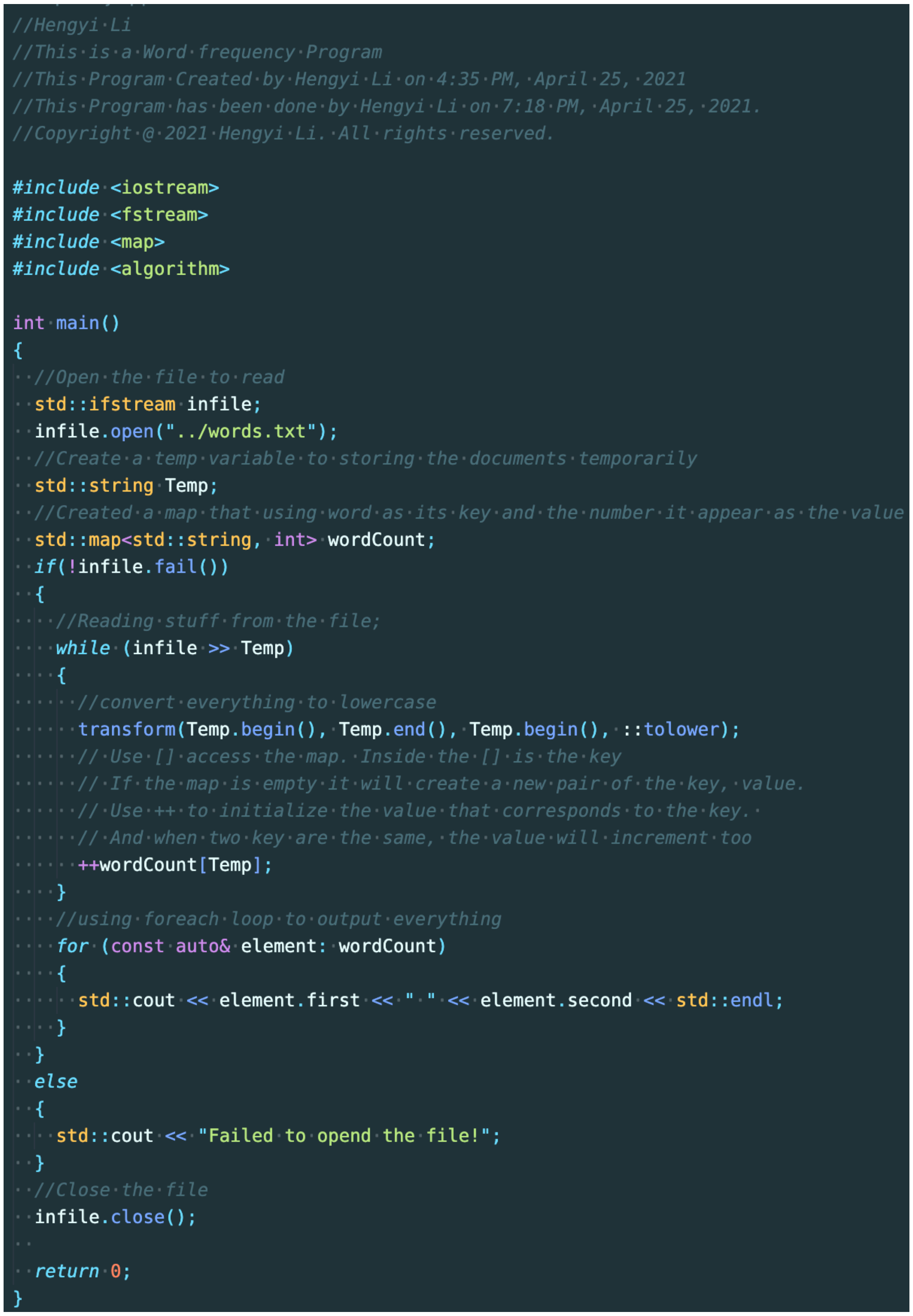
Input: Can you can a can as a canner can can a can?

Output:



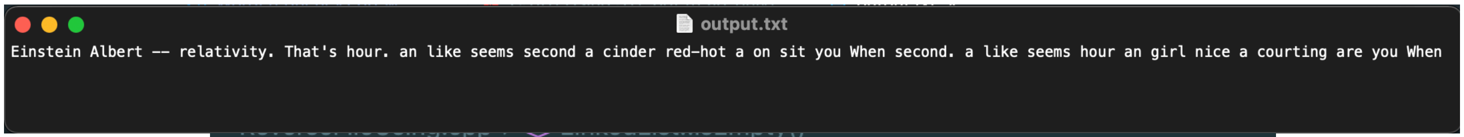
```
AKIRA@akria ~/ComputerScienceRelated/Spring2021-CS181/Lab6-UsingSTL/build  main ±  /Users/AKIRA/ComputerScienceRelated/Spring2021-CS181/Lab6-UsingSTL/build/Lab6_UsingSTL
? 1
a 3
as 1
can 6
canner 1
you 1
```

Program Screenshot:

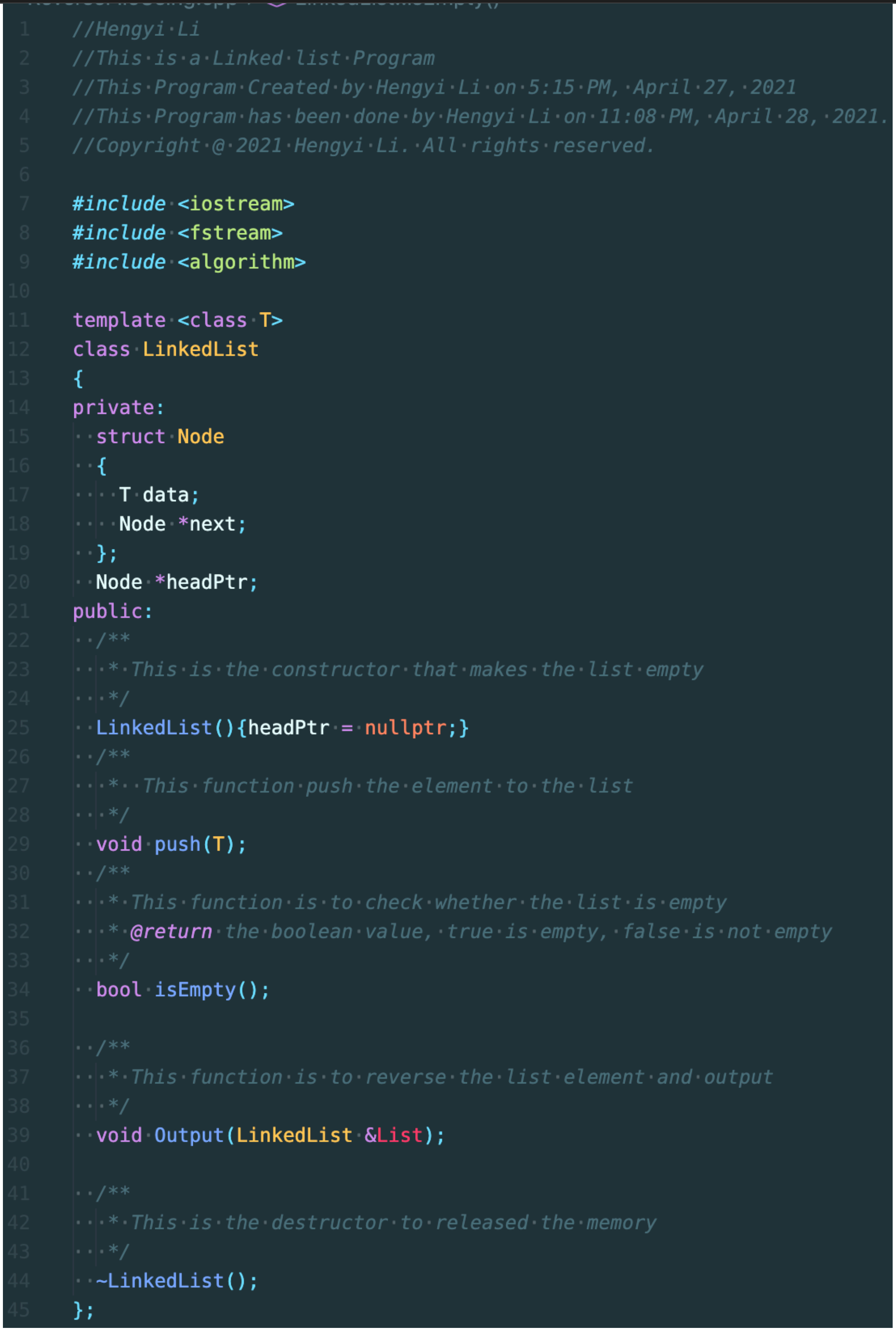


```cpp
//Hengyi Li
//This is a Word frequency Program
//This Program Created by Hengyi Li on 4:35 PM, April 25, 2021
//This Program has been done by Hengyi Li on 7:18 PM, April 25, 2021.
//Copyright @ 2021 Hengyi Li. All rights reserved.

#include <iostream>
#include <fstream>
#include <map>
#include <algorithm>

int main()
{
  //Open the file to read
  std::ifstream infile;
  infile.open("../words.txt");
  //Create a temp variable to storing the documents temporarily
  std::string Temp;
  //Created a map that using word as its key and the number it appear as the value
  std::map<std::string, int> wordCount;
  if(!infile.fail())
  {
    //Reading stuff from the file;
    while (infile >> Temp)
    {
      //convert everything to lowercase
      transform(Temp.begin(), Temp.end(), Temp.begin(), ::tolower);
      // Use [] access the map. Inside the [] is the key
      // If the map is empty it will create a new pair of the key, value.
      // Use ++ to initialize the value that corresponds to the key. 
      // And when two key are the same, the value will increment too
      ++wordCount[Temp];
    }
    //using foreach loop to output everything
    for (const auto& element: wordCount)
    {
      std::cout << element.first << " " << element.second << std::endl;
    }
  }
  else
  {
    std::cout << "Failed to opend the file!";
  }
  //Close the file
  infile.close();
  
  return 0;
}
```

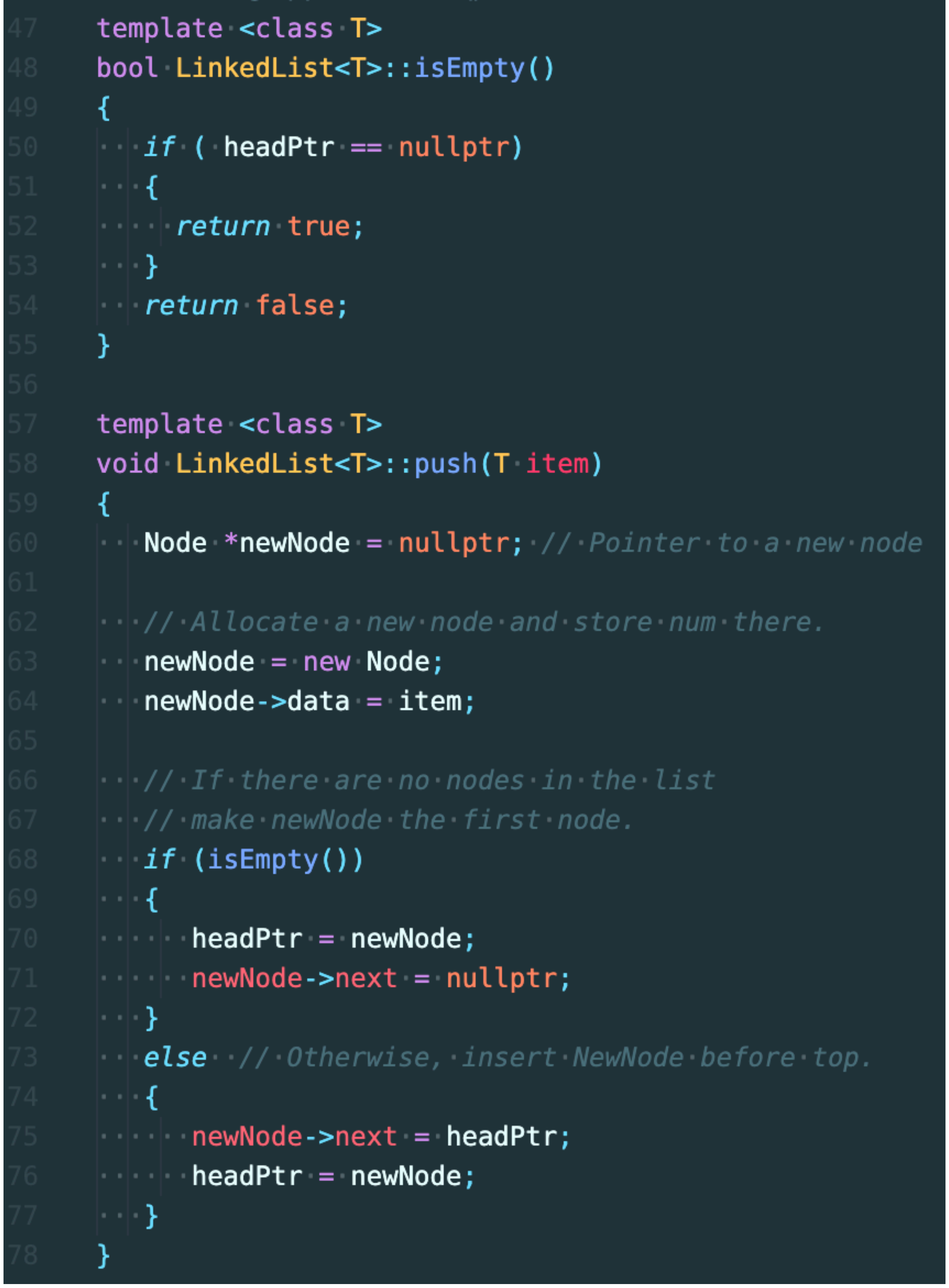
Q2:
Input: When you are courting a nice girl an hour seems like a second. When you sit on a red-hot cinder a second seems like an hour. That's relativity. -- Albert Einstein
Output:



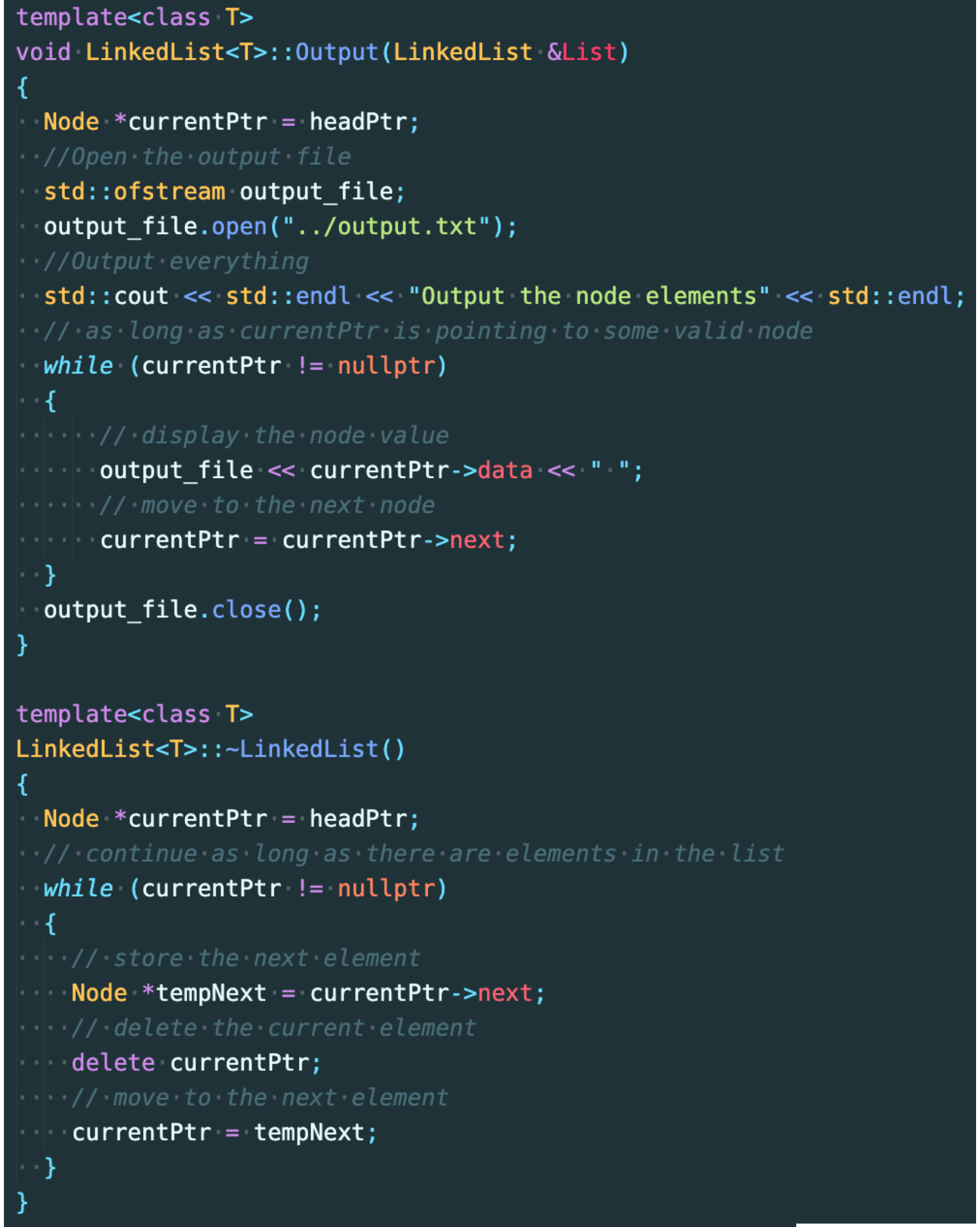
```
Einstein Albert -- relativity. That's hour. an like seems second a cinder red-hot a on sit you When second. a like seems hour an girl nice a courting are you When
```



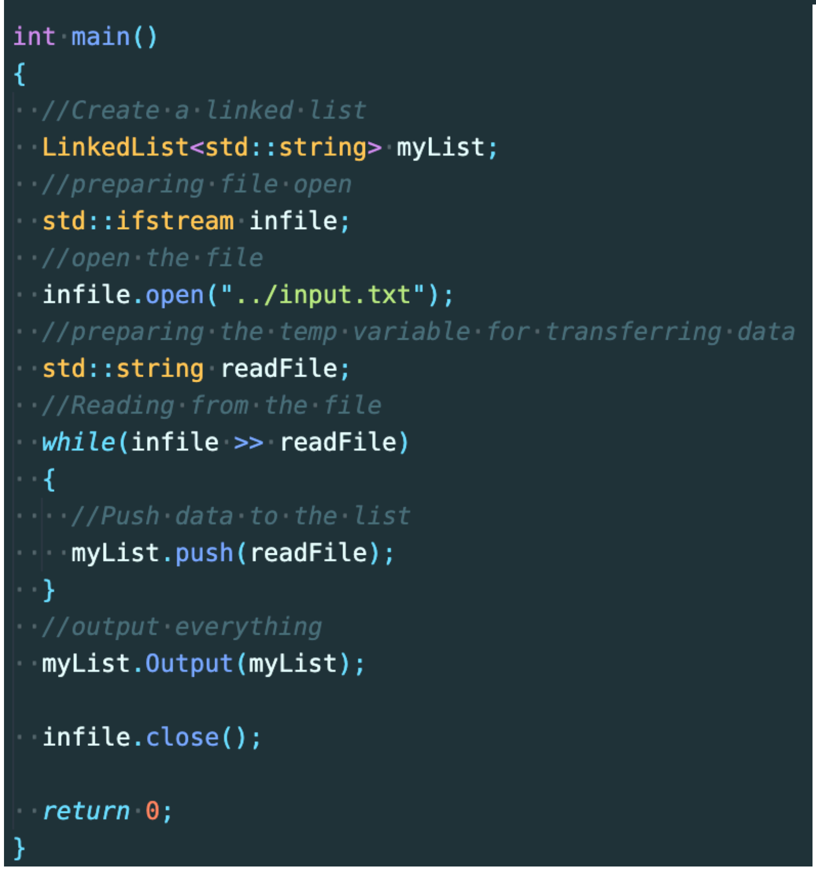
```cpp
//Hengyi Li
//This is a Linked list Program
//This Program Created by Hengyi Li on 5:15 PM, April 27, 2021
//This Program has been done by Hengyi Li on 11:08 PM, April 28, 2021.
//Copyright @ 2021 Hengyi Li. All rights reserved.

#include <iostream>
#include <fstream>
#include <algorithm>

template <class T>
class LinkedList
{
private:
  struct Node
  {
    T data;
    Node *next;
  };
  Node *headPtr;
public:
  /**
   * This is the constructor that makes the list empty
   */
  LinkedList(){headPtr = nullptr;}
  /**
   *  This function push the element to the list
   */
  void push(T);
  /**
   * This function is to check whether the list is empty
   * @return the boolean value, true is empty, false is not empty
   */
  bool isEmpty();

  /**
   * This function is to reverse the list element and output
   */
  void Output(LinkedList &List);

  /**
   * This is the destructor to released the memory
   */
  ~LinkedList();
};
```



```cpp
template <class T>
bool LinkedList<T>::isEmpty()
{
   if ( headPtr == nullptr)
   {
      return true;
   }
   return false;
}

template <class T>
void LinkedList<T>::push(T item)
{
   Node *newNode = nullptr; // Pointer to a new node

   // Allocate a new node and store num there.
   newNode = new Node;
   newNode->data = item;

   // If there are no nodes in the list
   // make newNode the first node.
   if (isEmpty())
   {
      headPtr = newNode;
      newNode->next = nullptr;
   }
   else  // Otherwise, insert NewNode before top.
   {
      newNode->next = headPtr;
      headPtr = newNode;
   }
}
```



```cpp
template<class T>
void LinkedList<T>::Output(LinkedList &List)
{
  Node *currentPtr = headPtr;
  //Open the output file
  std::ofstream output_file;
  output_file.open("../output.txt");
  //Output everything
  std::cout << std::endl << "Output the node elements" << std::endl;
  // as long as currentPtr is pointing to some valid node
  while (currentPtr != nullptr)
  {
      // display the node value
      output_file << currentPtr->data << " ";
      // move to the next node
      currentPtr = currentPtr->next;
  }
  output_file.close();
}

template<class T>
LinkedList<T>::~LinkedList()
{
  Node *currentPtr = headPtr;
  // continue as long as there are elements in the list
  while (currentPtr != nullptr)
  {
    // store the next element
    Node *tempNext = currentPtr->next;
    // delete the current element
    delete currentPtr;
    // move to the next element
    currentPtr = tempNext;
  }
}
```



```cpp
int main()
{
  //Create a linked list
  LinkedList<std::string> myList;
  //preparing file open
  std::ifstream infile;
  //open the file
  infile.open("../input.txt");
  //preparing the temp variable for transferring data
  std::string readFile;
  //Reading from the file
  while(infile >> readFile)
  {
    //Push data to the list
    myList.push(readFile);
  }
  //output everything
  myList.Output(myList);

  infile.close();

  return 0;
}
```