資料結構實習

11/03 作業報告

Stack 實作（e, s）segment

班級：資訊二甲

學號：D1109023

姓名：楊孟憲

Contents

1 引言

Stack 是一個只有一個開口的單向輸入輸出的資料結構，在日常生活中可以使用該資料結構維護許多演算法。插入以及拿取都只需要 $O(1)$ 的時間。以下是利用 LinkedList 實作 Stack 的方法。(以下是使用 C 實作 Stack 的 push/pop)

```c
/// stack
#include<stdio.h>
#include<stdlib.h>

typedef struct Node * NodePtr;

typedef struct Node {
  int val;
  NodePtr next; // Node *
} Node;

void push(NodePtr *head, int val) {
  NodePtr newNode = (NodePtr)malloc(sizeof(Node));
  newNode -> val = val;
  newNode -> next = (*head);
  *head = newNode;
  return;
}

void pop(NodePtr *head) {
  if(*head == NULL) {
    printf("The Stack is Empty!\n");
  } else {
    int topValue = (*head) -> val;
    printf("The top value of the stack: %d\n", topValue);
    NodePtr cur = *head;
    *head = (*head) -> next;
    free(cur);
  }
  return;
}

int main() {
  NodePtr head = NULL;

  push(&head, 20);
  push(&head, 30);
  pop(&head);

  while(head != NULL) {
    printf("Val: %d, nextPointer: %p\n", head -> val, head -> next);
    head = head -> next;
  }
  return 0;
}
```

2 題目敘述

題意說明： 如果在一個字串當中其頭一個字母為 E，最後一個字母為 S，而兩個字母中間不包含任何 E 或 S 字母的話，則稱為 ES 字串。請利用堆疊（Stacks）的原理，撰寫出一個程式，從檔案（input.txt）讀入一段文章或是字串，然後消去所有可能的 ES 字串，使得消去後的字串輸出不包含任何 ES 字串。

3 作法

使用 C++ 的 STL Stack 實作。遍歷字串的每一個字元，當前字元為's' 時，判斷 stack 是否為空並且如果前面有'e' 的話 (cnt > 0)，就不斷地把先前的字元從 stack 拿出來，直到 now == 'e'，並且將 cnt-1，這能夠確保之後的 e, s 區間能被正確的移除。否則，就把's' 放進 stack 裡。
如果當前字元不為's' 的話，就放進 stack 裡，並且判斷如果當前字元為'e' 的話，就 cnt++;

範例程式碼

```
#include<bits/stdc++.h>
using namespace std;

#define int long long
#define pb push_back
stack<char> st;

string get_ans(string ms) {
  int cnt = 0;
  for(int i = 0; i < ms.size(); i++) {
    if(ms[i] == 's') {
      if(cnt > 0) {
        while(!st.empty()) {
          char now = st.top();
          st.pop();
          if(now == 'e') {
            cnt--;
            break;
          }
        }
      } else {
        st.push(ms[i]);
      }
    } else {
      if(ms[i] == 'e') cnt++;
      st.push(ms[i]);
    }
  }

  string ans = "";
  while(!st.empty()) {
    ans = st.top() + ans;
    st.pop();
  }
  return ans;
}

void solve() {
  string ss, ms = "";
  ifstream ifs("input.txt", ifstream::in);

  if(!ifs.is_open()) {
    cout << "error\n";
    return;
  }

  while(getline(ifs, ss)) {
    if(ss == "") {
      cout << get_ans(ms) << "\n";
      ms = "";
    }
    ms += ss;
    ms += "\n";
  }
  cout << get_ans(ms) << "\n";
  return;
}

signed main() {
  solve();
  return 0;
}
```

4 執行結果

輸入輸出結果：

1. 輸入：

```
And then there’s the journey itself – retiring to bed as you
    clatter out of a big city and waking up in a new city, or
    even a new country, can create memories to last a lifetime.

At least that’s the theory – and why the new wave of night
    trains are being touted as one way to replace short or even
    medium-haul flights across Europe and the US.

So how’s that going?

Even before their renaissance, night trains could be a
    pleasant, memorable and sometimes economic way to cover long
    distances – but luck has always been a big factor.
```

2. 輸出：

```
And then ther the journelf – retiring to b you clatter out of
    a big city and waking up in a new city, or even a new
    country, can create mt a lifetime.


At lt that’s the theory – and why the new wav are being tout
    one way to replachort or  Europe and the US.


So how’s that going?


Even before thanc could be a plant, memorablom been a big
    factor.
```

5 心得與討論

這次的作業利用 Stack 的原理來完成，是一個常見的模板題，實作起來沒有什麼問題，這讓我更加了解 Stack 的運作原理。