Университет ИТМО

Факультет программной инженерии и компьютерной техники

**Лабораторная работа №3**

по «Алгоритмам и структурам данных»

Timus

Выполнил:

Студент группы P3233

Фамилия И.О.

Шикунов Максим Евгеньевич

Преподаватели:

Косяков М.С.

Тараканов Д.С.

Санкт-Петербург

2024

Задача №1067 «Структура папок»

Код:

#include <iostream>  
#include <map>  
#include <sstream>  
  
using namespace std**;**struct directory {  
 map<string**,** struct directory\*> childDirectory**;**}**;**void print(struct directory\* dir**,** string spaces) {  
 string nextSpace = spaces + " "**;** if (!dir->childDirectory.empty()) {  
 for (auto it : dir->childDirectory) {  
 cout << spaces << it.first << endl**;** print(it.second**,** nextSpace)**;** }  
 }  
}  
  
int main() {  
 struct directory root = {}**;** struct directory\* current**;** string path**,** folderName**;** size\_t n**;** cin >> n**;** for (size\_t i = 0**;** i < n**;** i++) {  
 current = &root**;** cin >> path**;** stringstream ss(path)**;** while (getline(ss**,** folderName**,** '\\')) {  
 if (current->childDirectory.find(folderName) != current->childDirectory.end()) {  
 current = current->childDirectory[folderName]**;** } else {  
 current->childDirectory[folderName] = new struct directory**;** current = current->childDirectory[folderName]**;** }  
 }  
 }  
 print(&root**,** "")**;** return 0**;**}

Пояснение к примененному алгоритму:

Используем map, чтобы наши имена папок сортировались правильно, а также чтобы по ключу хранить ссылку на подзаголовки данной папки. Каждый итерацию мы принимаем путь, читаем название каждой папки, если нет такого названия, то создаем, иначе передаем существующую.В конце рекурсивно выводим наше дерево папок.

Сложность:

Задача №1494 «Монобильярд»

Код:

#include <iostream>  
#include <stack>  
using namespace std**;**int main() {  
 stack<size\_t> scoredPools**;** size\_t topPool = 0**;** size\_t n**;** cin >> n**;** size\_t pools[n]**;** for (size\_t i = 0**;** i < n**;** i++) {  
 cin >> pools[i]**;** }  
 for (size\_t i = 0**;** i < n**;** i++) {  
 if (pools[i] > topPool) {  
 size\_t pool = topPool + 1**;** while (pool < pools[i]) {  
 scoredPools.push(pool)**;** pool++**;** }  
 topPool = pools[i]**;** } else if (pools[i] == scoredPools.top()) {  
 scoredPools.pop()**;** } else {  
 cout << "Cheater"**;** return 0**;** }  
 }  
 cout << "Not a proof"**;** return 0**;**}

Пояснение к примененному алгоритму:

В данной задаче я проверяю, какой шар вытащили, если этот шар был больше чем “наибольший закатившийся”, то значит между ними должны были закатываться и шары от последнего до этого в порядке возрастания, для этого мы заносим в таком порядке в стек, чтобы они там хранились. Если приходит на проверку шар, который меньше чем наибольший, то он должен быть на вершине стека, иначе он читер.

Сложность: