Санкт-Петербургский национальный исследовательский университет ИТМО

Факультет систем управления и робототехники

Алгоритмы и структуры данных

Отчёт по лабораторной работе №11 (1067)

Преподаватель: Тропченко А. А.

Выполнил: Марухленко Д. С.

Группа: R3235

Санкт Петербург, 2021г.

1. Цель работы

Решить задачу №1067 на платформе Timus Online Judge
https://acm.timus.ru/problem.aspx?space=1&num=1067

2. Задача

Условие

Hacker Bill has accidentally lost all the information from his workstation's hard drive and he has no backup copies of its contents. He does not regret for the loss of the files themselves, but for the very nice and convenient directory structure that he had created and cherished during years of work. Fortunately, Bill has several copies of directory listings from his hard drive. Using those listings he was able to recover full paths (like "WINNT\SYSTEM32\CERTSRV\CERTCO~1\X86") for some directories. He put all of them in a file by writing each path he has found on a separate line. Your task is to write a program that will help Bill to restore his state of the art directory structure by providing nicely formatted directory tree..

Ограничение времени: 1.0 секунды
Ограничение памяти: 64 МБ

Исходные данные

The first line of the input contains single integer number N ($1 \le N \le 500$) that denotes a total number of distinct directory paths. Then N lines with directory paths follow. Each directory path occupies a single line and does not contain any spaces, including leading or trailing ones. No path exceeds 80 characters. Each path is listed once and consists of a number of directory names separated by a back slash (""). Each directory name consists of 1 to 8 uppercase letters, numbers, or the special characters from the following list: exclamation mark, number sign, dollar sign, percent sign, ampersand, apostrophe, opening and closing parenthesis, hyphen sign, commercial at, circumflex accent, underscore, grave accent, opening and closing curly bracket, and tilde ("!#$%&'()-@^_`{}~").

Результат

Write to the output the formatted directory tree. Each directory name shall be listed on its own line preceded by a number of spaces that indicate its depth in the directory hierarchy. The subdirectories shall be listed in lexicographic order immediately after their parent directories preceded by one more space than their parent directory. Top level directories shall have no spaces printed before their names and shall be listed in lexicographic order. See sample below for clarification of the output format.

Пример

Исходные данные	**Результат**
7 WINNT\SYSTEM32\CONFIG GAMES WINNT\DRIVERS	GAMES _DRIVERS HOME WIN

HOME WIN\SOFT GAMES\DRIVERS WINNT\SYSTEM32\CERTSRV\CERTCO~1\X86	_SOFT WINNT _DRIVERS _SYSTEM32 __CERTSRV ___CERTCO~1 ____X86 __CONFIG

3. Материалы работы

3.1. Объяснение алгоритма

Задача подразумевает вывод дерева каталогов. Для этого при записи данные сохраняются в структуру `directory`, ссодержащую map с вложенными каталогами (в формате имя: структура-каталог). Далее рекурсивно вызывается функция `getOrder()`, рекурсивно выводящая имя каталога со сдвигом – глубиной дерева.

3.2. Код программы.

```
1.  #include <iostream>
2.  #include <bits/stdc++.h>
3.
4.  using namespace std;
5.
6.  struct directory{
7.      map<string, directory *> children;
8.  };
9.
10. void getOrder(directory* const parent, string tab_current) {
11.     string tab = " ";
12.     tab += tab_current;
13.
14.     map<string, directory *> sorted_children(parent->children.begin(), parent-
    >children.end());
15.     map<string, directory *>::iterator it;
16.
17.         for (it = sorted_children.begin(); it != sorted_children.end(); it++) {
18.             cout << tab_current << it->first << endl;
19.             getOrder(it->second,tab);
20.         }
21. }
22.
23. directory *get_directory(string const name, directory* const parent) {
24.         if ((parent->children).find(name) != parent->children.end()) {
25.             return parent->children[name];
26.         } else {
27.             parent->children[name] = new directory{};
28.             return parent->children[name];
29.         }
30.     }
31.
32. int main() {
33.     int count;
34.     cin >> count;
35.     directory * root = new directory{};
36.     string s;
37.
```

```
    for (int i = 0; i <= count; i++) {
        getline(cin, s);
        stringstream path(s);

        directory *current = root;
        while (getline(path, s, '\\')) {
            current = get_directory(s,current);
        }
    }
    getOrder(root,"");
    return 0;
}
```

4. Результат выполнения и ссылка на репозиторий GitHub

ID	Дата	Автор	Задача	Язык	Результат проверки	№ теста	Время работы	Выделено памяти
9276484	00:22:05 22 мар 2021	Daniil Marukhlenko	1067	G++ 9.2 x64	Accepted		0.031	3 328 КБ

https://github.com/japersik/algorithms_and_data_structures/

5. Вывод

Работа выполнена, задача решена, полученный алгоритм, использующий представление структуры данных (дерева) удовлетворяет условиям.