CS 330: Homework 3

Dirch

1 Description

Your goal in this homework is to wrap your head around the beauty and pain that is C programming. You will do this in the context of writing a utility to search for files with a certain name tucked away inside some directory.

2 Requirements

To receive full credit for this assignment, you must satisfy the following requirements:

- Create a directory named `dirch` and place all of your files inside of it.
- Declare in `dirch.h` a new `struct` type named `tree_t`. Variables of type `struct tree_t` have three fields:
  - a `char *` name (named `name`),
  - an `int` number of children (named `nchildren`),
  - and a `struct tree_t **` set of child nodes (named `children`).

  These names are part of the public interface; you must match them exactly. (Your instructor's test code expects them, so deviate at your peril.)
- Declare in your header file a function `make_path` that takes two parameters in this order: a `const char *` directory name and a `const char *` file name. It returns a new dynamically-allocated and null-terminated string in which the directory name and the file name are fused together, with a '/' separating them. For example, `make_path("backup/music", "dirge.mp3")` → `"backup/music/dirge.mp3"`.
- Declare in your header file a function `get_tree` that takes one parameter: a `const char *` name, which you may assume to be a path to either an existing file or directory. It returns a pointer to a dynamically-allocated `struct tree_t`, whose `name` field points to a *copy* of the path parameter and whose `nchildren` is 0 if the path refers to a file. If the path refers to a directory, `nchildren` is the number of files (including directories) contained just within it (the children only, no grandchildren or other descendants) and `children` is a set of dynamically-allocated nodes, one for each file or directory inside of it. All told, this function returns a hierarchical view of your computer's file system, starting at the specified path.

- Declare in your header file a function `find_in_tree` that takes two parameters in this order: a `const struct tree_t *` and `const char *` file name. Starting at the specified file hierarchy's root, walk through the tree depth first. Anytime a node is encountered whose name matches the specified file name, print its full path, starting at the root. For example, suppose we have this hierarchy:

  ```
  dir1
    butane
    test.txt
    dir2
      test.txt
      glycerin
  ```

  Function `find_in_tree`, when given a `tree_t` for `dir1` and "test.txt" as parameters, should print to stdout the text:

  ```
  dir1/test.txt
  dir1/dir2/test.txt
  ```

- Write in `dirch.c` the implementations of these methods and a `main` function that takes a single command-line argument for a path and creates the tree at the path. It then prompts the user to enter a name, searches the tree for any files with that name, prints out the path to those matches, and prompts for another name. When the user closes stdin (enters Control-D), the program ends, freeing all dynamically-allocated memory. Use your functions in your solution.
- Write a makefile named `makefile` that compiles your code and has a `clean` rule.
- Include a capture of a sample run of your program. Execute these commands:

  ```
  script capture.txt
  make clean
  make
  valgrind ./dirch /path/to/some/directory/of/your/choosing
  <Control-D>
  ```

  Make sure `valgrind` reports no memory leaks.

- The makefile and the `dirch` executable must work on `clark.cs.uwec.edu`.

3 Submission

Before submitting, you are encouraged to write test code and share it with the class on Piazza. Individuals who do so will have their names inserted into a lottery for a fabulous prize to be awarded at the end of the semester.

Please submit according to these instructions. Violators will be prosecuted.

1. One level above your `dirch` directory, run the command `zip -r dirch.zip dirch` to create a ZIP archive of your files.

2. Drop your `dirch.zip` file into your submission directory with the command:

   ```
   cp dirch.zip "/network_shares/w_drive/c s/CJohnson/cs330/<YOUR-USERNAME>"
   ```

   You may overwrite this file as often as you like before the deadline.