FILE ORGANIZATION

September 15, 2019

Rwithik Manoj
College of Engineering, Trivandrum
Department of Computer Science and Engineering

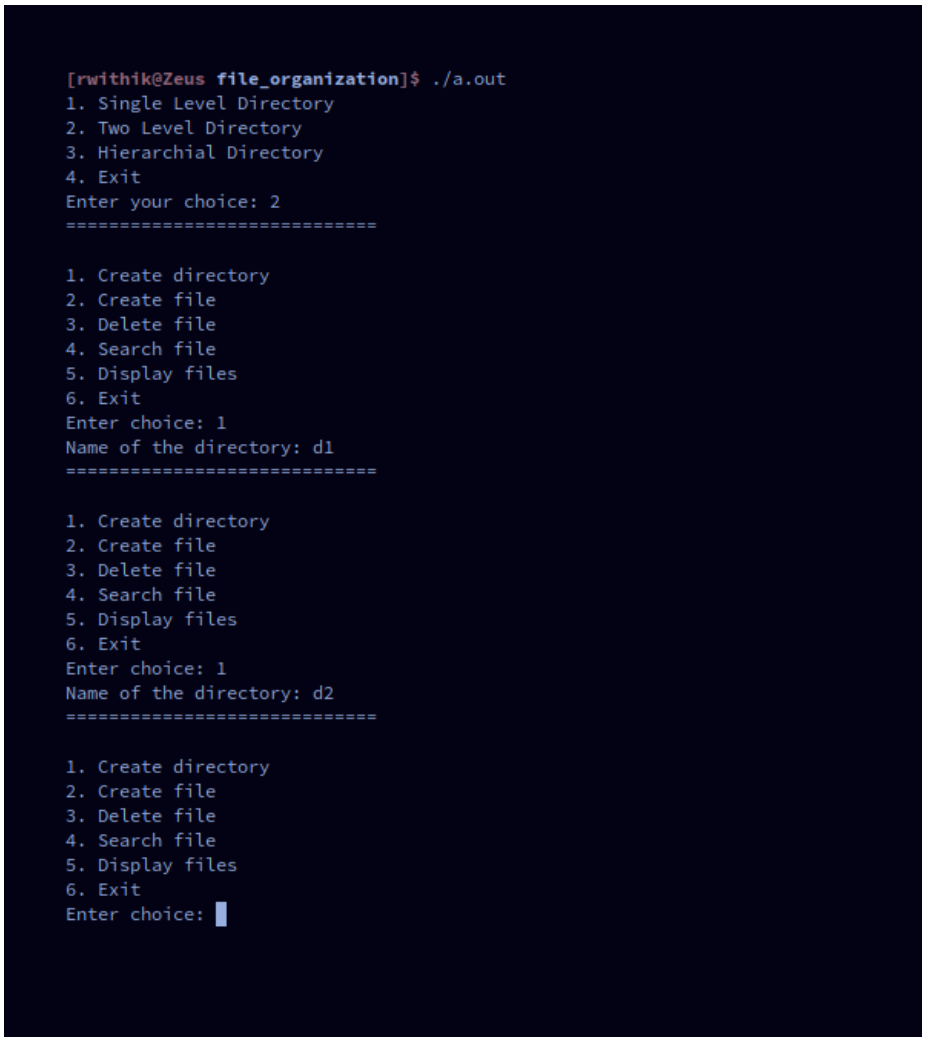
The Base Program

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

#include "single-level.h"
#include "two-level.h"
#include "hierarchial.h"

int main(int argc, char const *argv[]) {

    int choice;

    cout << "1. Single Level Directory" << endl;
    cout << "2. Two Level Directory" << endl;
    cout << "3. Hierarchial Directory" << endl;
    cout << "4. Exit" << endl;
    cout << "Enter your choice: ";
    cin >> choice;

    switch(choice){
      case 1: single();
      break;
      case 2: two();
      break;
      case 3: hierarchial();
      break;
      default: cout << "Invalid choice" << endl;
      break;
    }

    return 0;
  }
```

Output

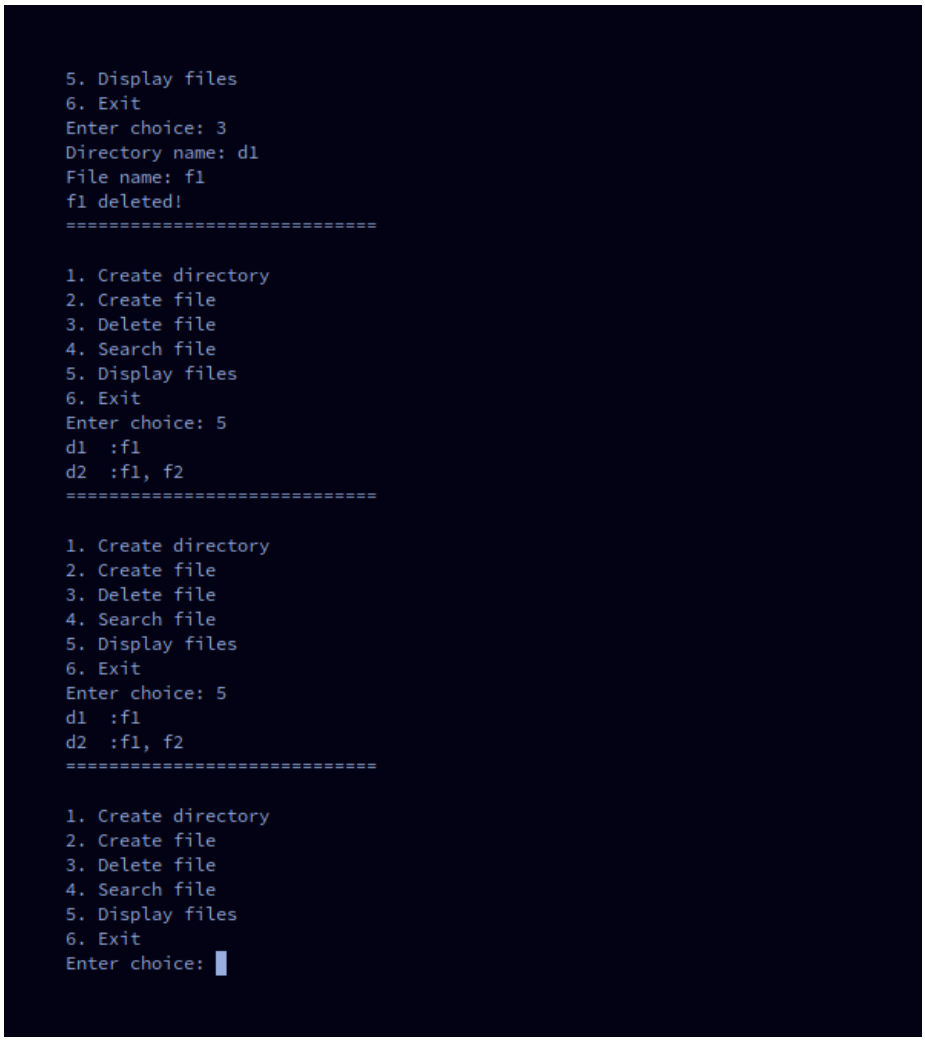
```
[rwithik@Zeus file_organization]$ g++ file_organization.cpp
[rwithik@Zeus file_organization]$ ./a.out
1. Single Level Directory
2. Two Level Directory
3. Hierarchial Directory
4. Exit
Enter your choice: 1
Enter the name of the base directory: root
============================

1. Create file
2. Delete file
3. Search file
4. Display file
5. Exit
Enter the choice: 1
Name of the file: f1
============================

1. Create file
2. Delete file
3. Search file
4. Display file
5. Exit
Enter the choice: 1
Name of the file: f2
============================

1. Create file
2. Delete file
3. Search file
4. Display file
5. Exit
Enter the choice: 2
```

```
Name of the file: f2
============================

1. Create file
2. Delete file
3. Search file
4. Display file
5. Exit
Enter the choice: 2
Name of the file: f1
============================

1. Create file
2. Delete file
3. Search file
4. Display file
5. Exit
Enter the choice: 4
f2
============================

1. Create file
2. Delete file
3. Search file
4. Display file
5. Exit
Enter the choice: 3
Name of the file: f2
f2 found!
============================
```


[rwithik@Zeus file_organization]$ ./a.out
1. Single Level Directory
2. Two Level Directory
3. Hierarchial Directory
4. Exit
Enter your choice: 2
============================
1. Create directory
2. Create file
3. Delete file
4. Search file
5. Display files
6. Exit
Enter choice: 1
Name of the directory: d1
============================
1. Create directory
2. Create file
3. Delete file
4. Search file
5. Display files
6. Exit
Enter choice: 1
Name of the directory: d2
============================
1. Create directory
2. Create file
3. Delete file
4. Search file
5. Display files
6. Exit
Enter choice:

```
1. Create directory
2. Create file
3. Delete file
4. Search file
5. Display files
6. Exit
Enter choice: 2
Directory name: d1
Name of the file: f1
File created
============================

1. Create directory
2. Create file
3. Delete file
4. Search file
5. Display files
6. Exit
Enter choice: 2
Directory name: d1
Name of the file: f2
File created
============================

1. Create directory
2. Create file
3. Delete file
4. Search file
5. Display files
6. Exit
Enter choice: 2
Directory name: d2
Name of the file: f1
File created
============================

1. Create directory
```

```
1. Create directory
2. Create file
3. Delete file
4. Search file
5. Display files
6. Exit
Enter choice: 2
Directory name: d2
Name of the file: f2
File created
============================

1. Create directory
2. Create file
3. Delete file
4. Search file
5. Display files
6. Exit
Enter choice: 4
File name: f1
Found in directory: d1
Found in directory: d2
============================

1. Create directory
2. Create file
3. Delete file
4. Search file
5. Display files
6. Exit
Enter choice: 5
d1  :f1, f2
d2  :f1, f2
============================

1. Create directory
```


5. Display files
6. Exit
Enter choice: 3
Directory name: d1
File name: f1
f1 deleted!
==========

1. Create directory
2. Create file
3. Delete file
4. Search file
5. Display files
6. Exit
Enter choice: 5
d1 :f1
d2 :f1, f2
==========

1. Create directory
2. Create file
3. Delete file
4. Search file
5. Display files
6. Exit
Enter choice: 5
d1 :f1
d2 :f1, f2
==========

1. Create directory
2. Create file
3. Delete file
4. Search file
5. Display files
6. Exit
Enter choice:

```
[rwithik@Zeus file_organization]$ ./a.out
1. Single Level Directory
2. Two Level Directory
3. Hierarchial Directory
4. Exit
Enter your choice: 3
Enter the name of file or directory to be created: root
Enter 1 for directory, 2 for file: 1
root created!
Enter the number of subdirectories and files of root: 1
Enter the name of file or directory to be created under root: rwithik
Enter 1 for directory, 2 for file: 1
rwithik created!
Enter the number of subdirectories and files of rwithik: 3
Enter the name of file or directory to be created under rwithik: Pictures
Enter 1 for directory, 2 for file: 1
Pictures created!
Enter the number of subdirectories and files of Pictures: 2
Enter the name of file or directory to be created under Pictures: Photos
Enter 1 for directory, 2 for file: 1
Photos created!
Enter the number of subdirectories and files of Photos: 0
Enter the name of file or directory to be created under Pictures: Screensh
ots
Enter 1 for directory, 2 for file: 1
Screenshots created!
Enter the number of subdirectories and files of Screenshots: 0
Enter the name of file or directory to be created under rwithik: Documents
Enter 1 for directory, 2 for file: 1
Documents created!
Enter the number of subdirectories and files of Documents: 3
Enter the name of file or directory to be created under Documents: Doc1.pd
f
Enter 1 for directory, 2 for file: 2
Doc1.pdf created!
Enter the name of file or directory to be created under Documents: Doc2.pd
f
```

```
rwithik created!
Enter the number of subdirectories and files of rwithik: 3
Enter the name of file or directory to be created under rwithik: Pictures
Enter 1 for directory, 2 for file: 1
Pictures created!
Enter the number of subdirectories and files of Pictures: 2
Enter the name of file or directory to be created under Pictures: Photos
Enter 1 for directory, 2 for file: 1
Photos created!
Enter the number of subdirectories and files of Photos: 0
Enter the name of file or directory to be created under Pictures: Screensh
ots
Enter 1 for directory, 2 for file: 1
Screenshots created!
Enter the number of subdirectories and files of Screenshots: 0
Enter the name of file or directory to be created under rwithik: Documents
Enter 1 for directory, 2 for file: 1
Documents created!
Enter the number of subdirectories and files of Documents: 3
Enter the name of file or directory to be created under Documents: Doc1.pd
f
Enter 1 for directory, 2 for file: 2
Doc1.pdf created!
Enter the name of file or directory to be created under Documents: Doc2.pd
f
Enter 1 for directory, 2 for file: 2
Doc2.pdf created!
Enter the name of file or directory to be created under Documents: Doc3.pd
f
Enter 1 for directory, 2 for file: 2
Doc3..pdf created!
Enter the name of file or directory to be created under rwithik: Desktop
Enter 1 for directory, 2 for file: 1
Desktop created!
Enter the number of subdirectories and files of Desktop: 0
[rwithik@Zeus file_organization]$
[rwithik@Zeus file_organization]$
```

Header Files

Single Level Directory Structure

```
void single(){
  string base, name;
  vector<string> files;
  vector<string>::iterator it;
  int choice;

  cout << "Enter the name of the base directory: ";
  cin >> base;

  while (true){
    cout << "============================" << endl << endl;
```

```
    cout << "1. Create file\n2. Delete file\n3. Search file\n4. Display file\n5. Exit" <<
    cout << "Enter the choice: ";
    cin >> choice;
    switch (choice) {
      case 1: cout << "Name of the file: ";
      cin >> name;
      files.push_back(name);
      break;

      case 2: cout << "Name of the file: ";
      cin >> name;
      it = find(files.begin(), files.end(), name);
      files.erase(it);
      break;

      case 3: cout << "Name of the file: ";
      cin >> name;
      it = find(files.begin(), files.end(), name);
      if (it != files.end()) cout << name << " found!" << endl;
      else cout << "Not found!" << endl;
      break;

      case 4:
      for (auto file: files){
        cout << file << endl;
      }
      break;

      case 5: exit(0);
      break;

      default: cout << "Invalid choice!" << endl;
    }
  }
}
```

Two Level Directory Structure

```
void two(){
  string name, dirname;
  int choice;

  map<string, vector<string>> fs;
  map<string, vector<string>>::iterator search;

  while(true){
    cout << "============================" << endl << endl;
    cout << "1. Create directory\n2. Create file\n3. Delete file\n4. Search file\n5. Displ
    cout << "Enter choice: ";
    cin >> choice;
```

```
switch (choice){
  case 1: cout << "Name of the directory: ";
  cin >> name;
  fs.insert(pair<string, vector<string>>(name, vector<string>()));
  break;

  case 2: cout << "Directory name: ";
  cin >> dirname;
  cout << "Name of the file: ";
  cin >> name;
  search = fs.find(dirname);
  if (search == fs.end())
    cout << "Directory does not exist!" << endl;
  else{
    cout << "File created" << endl;
    search -> second.push_back(name);
  }
  break;

  case 3: cout << "Directory name: ";
  cin >> dirname;
  cout << "File name: ";
  cin >> name;
  search = fs.find(dirname);
  if (search != fs.end()){
    auto contents = search -> second;
    auto file = find(contents.begin(), contents.end(), name);
    if (file != contents.end()){
      cout << name << " deleted!" << endl;
      fs[dirname].erase(file);
    }
    else{
      cout << "File not found!" << endl;
    }
  }
  else{
    cout << "Directory not found!" << endl;
  }
  break;

  case 4: cout << "File name: ";
  cin >> name;
  for (auto i: fs){
    auto found = find(i.second.begin(), i.second.end(), name);
    if (found != i.second.end()){
      cout << "Found in directory: " << i.first << endl;
    }
  }
  break;
```

```
      case 5:
      for (auto i: fs){
        cout << i.first << "\t:";
        for (auto j: i.second){
          cout << j << ", ";
        }
        cout << "\b\b " << endl;
      }
      break;

      case 6:
      exit(0);

      default: cout << "Invalid choice!" << endl;
    }
  }
}
```

Hierarchial Directory Structure

```
class Directory{
public:
  vector<Directory*> subdirs;
  vector<string> files;
  string path;
  Directory(string path){
    this -> path = path;
  }
};

void insert(Directory* root, string name, int choice);

void insert(Directory* root, string name, int choice){
  string subname;
  if (choice == 2){
    root -> files.push_back(name);
    cout << name << " created!" << endl;
  }
  else if (choice == 1){
    int n;
    Directory* subdir = new Directory(root -> path + "/" + name);
    root -> subdirs.push_back(subdir);
    cout << name << " created!" << endl;
    cout << "Enter the number of subdirectories and files of " << name << ": ";
    cin >> n;
    while(n--){
      cout << "Enter the name of file or directory to be created under " << name << ": ";
      cin >> subname;
      cout << "Enter 1 for directory, 2 for file: ";
```

```
      cin >> choice;
      insert(subdir, subname, choice);
    }
  }
}


void hierarchial(){
  string name;
  int choice;

  Directory* base = new Directory("/");

  cout << "Enter the name of file or directory to be created: ";
  cin >> name;
  cout << "Enter 1 for directory, 2 for file: ";
  cin >> choice;
  insert(base, name, choice);

}
```