OS LAB 8

SAHIL BONDRE: U18CO021

1. To simulate the following file organization techniques

a) Single level directory

b) Two level directory

Description

In a single-level directory system, all the files are placed in one directory. There is a root directory which has all files. It has a simple architecture and there are no sub directories.

In the two-level directory system, each user has own user file directory (UFD). The system maintains a master block that has one entry for each user. This master block contains the addresses of the directory of the users. When a user job starts or a user logs in, the system's master file directory (MFD) is searched. When a user refers to a particular file, only his own UFD is searched. This effectively solves the name collision problem and isolates users from one another.

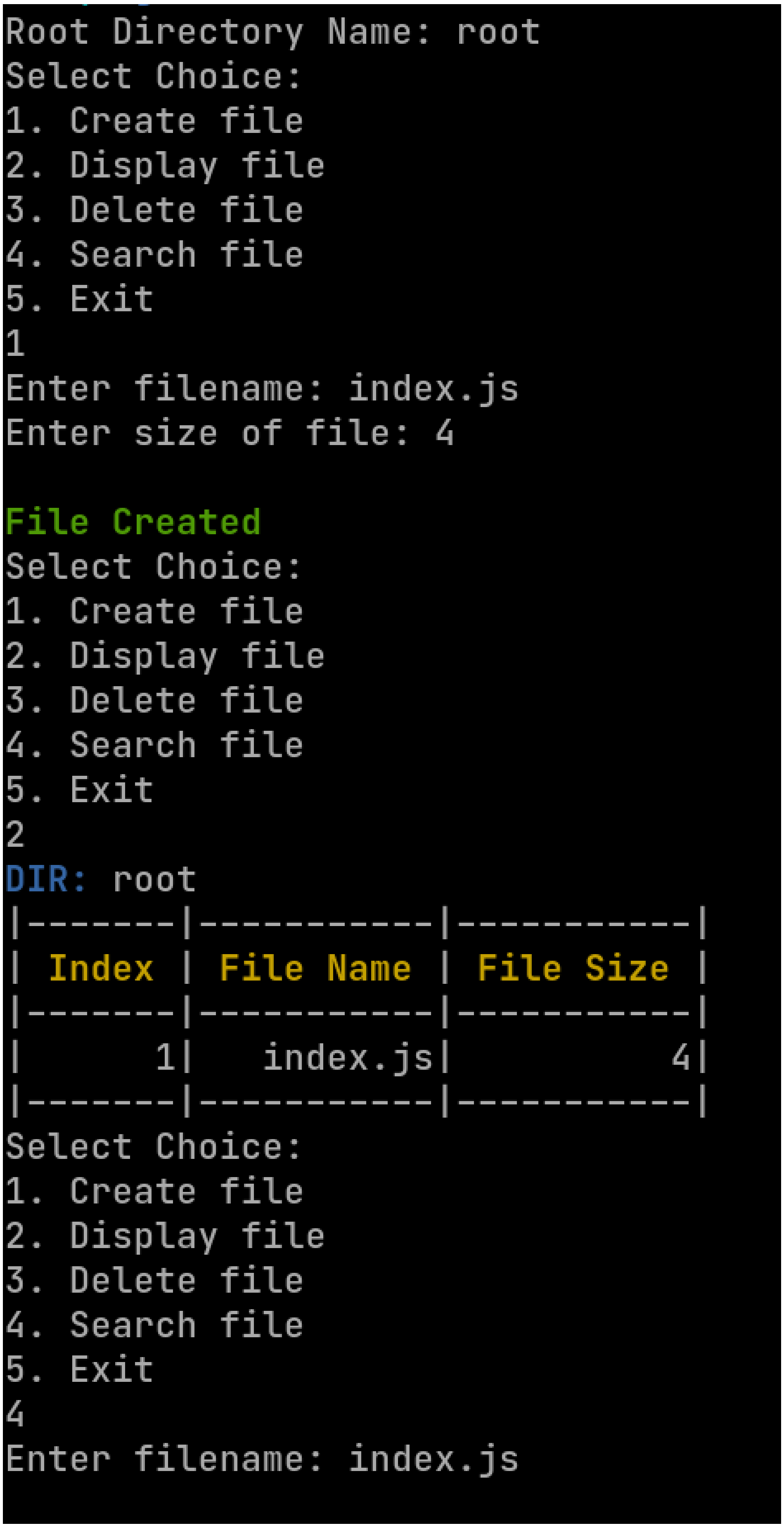
Operation: Create, Display, Delete, Search files

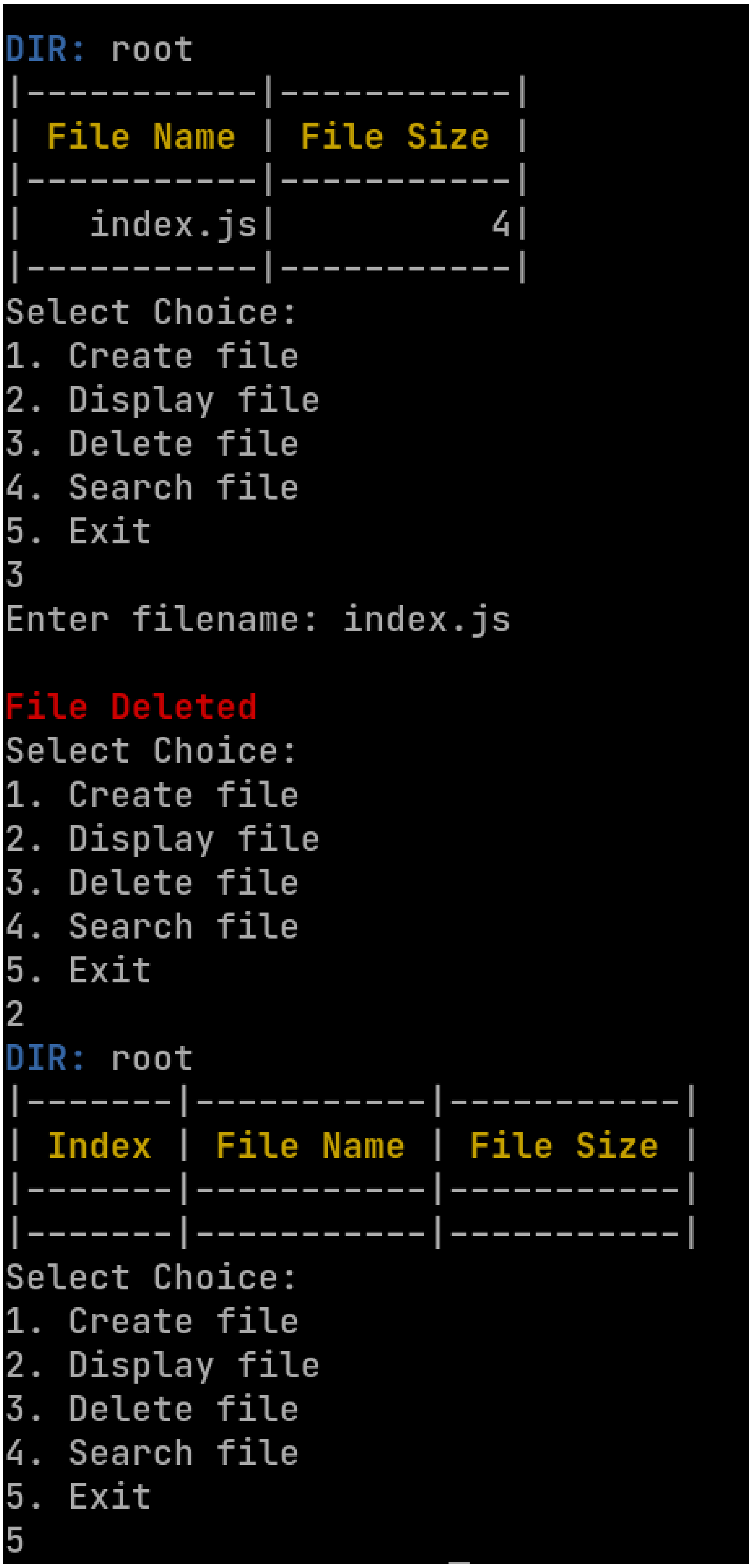
Single Level Directory:

```
#include <bits/stdc++.h>

using namespace std;

void createFile(File file) {
  bool fileExists = false;
  for (int i = 0; i < directory->files.size(); ++i) {
    if (directory->files[i].name == file.name) {
      printf("\x1B[1;31mERROR: File already exists in the
directory\033[0m\n");
      fileExists = true;
      break;
    }
  }
  if (!fileExists) {
    ++directory->numDirectory;
    directory->files.push_back(file);
    printf("\x1B[1;32mFile Created\033[0m\n");
  }
}
```

```
void displayFiles() {
  printf("\x1B[1;34mDIR: \033[0m");
  cout << directory->directory << endl;
  cout << "|-------|-----------|-----------|\n";
  cout << '|';
  printf("\x1B[1;33m Index \033[0m");
  cout << '|';
  printf("\x1B[1;33m File Name \033[0m");
  cout << '|';
  printf("\x1B[1;33m File Size \033[0m");
  cout << '|' << endl;
  cout << "|-------|-----------|-----------|\n";

  for (auto itr = directory->files.begin(); itr <
directory->files.end();
       ++itr) {
    int idx = itr - directory->files.begin();
    File file = directory->files[idx];
    cout << '|' << setw(7) << idx + 1 << '|' << setw(11) << file.name <<
'|'
         << setw(11) << file.size << '|' << endl;
  }
  cout << "|-------|-----------|-----------|\n";
}

void deleteFile(string fileName) {
  bool found = false;
  for (auto itr = directory->files.begin(); itr <
directory->files.end();
       ++itr) {
    int idx = itr - directory->files.begin();
    if (directory->files[idx].name == fileName) {
      directory->files.erase(itr);
      --directory->numDirectory;
      found = true;
    }
  }

  if (!found)
    printf("\x1B[1;31mERROR: File doesn't exist in the
directory\033[0m\n");
  else
    printf("\x1B[1;31mFile Deleted\033[0m\n");
}

void searchFile(string fileName) {
  bool found = false;
  File file;
  for (auto itr = directory->files.begin(); itr <
directory->files.end();
       ++itr) {
    int idx = itr - directory->files.begin();
    if (directory->files[idx].name == fileName) {
      file = directory->files[idx];
      found = true;
    }
  }

  if (!found)
    printf("\x1B[1;31mERROR: File doesn't exist in the
directory\033[0m\n");
  else {
    printf("\x1B[1;34mDIR: \033[0m");
    cout << directory->directory << endl;
    cout << "|-----------|-----------|\n";

    cout << '|';
    printf("\x1B[1;33m File Name \033[0m");
    cout << '|';
    printf("\x1B[1;33m File Size \033[0m");
    cout << '|' << endl;
    cout << "|-----------|-----------|\n";
    cout << '|' << setw(11) << file.name << '|' << setw(11) << file.size
<< '|'
         << endl;
    cout << "|-----------|-----------|\n";
  }
}

class File {
 public:
  string name;
  long long size;
};

class Directory {
 public:
  string directory;
  int numDirectory;
  vector<File> files;
};
```

```
Directory* directory = new Directory();

int main() {
  cout << "Root Directory Name: ";
  getline(cin, directory->directory);

  // single level
  int choice = 1;
  bool exit = false;

  while (!exit) {
    cout << "Select Choice: "
         << "\n"
         << "1. Create file"
         << "\n"
         << "2. Display file"
         << "\n"
         << "3. Delete file"
         << "\n"
         << "4. Search file"
         << "\n"
         << "5. Exit" << endl;
    cin >> choice;

    if (choice == 1) {
      File file;
      cout << "Enter filename: ";
      cin >> file.name;
      cout << "Enter size of file: ";
      cin >> file.size;
      cout << endl;
      createFile(file);

    } else if (choice == 2) {
      displayFiles();

    } else if (choice == 3) {
      string fileName;
      cout << "Enter filename: ";
      cin >> fileName;
      cout << endl;
      deleteFile(fileName);

    } else if (choice == 4) {
      string fileName;
      cout << "Enter filename: ";
      cin >> fileName;
      cout << endl;
      searchFile(fileName);

    } else if (choice == 5) {
      exit = true;
    } else {
      printf("\x1B[1;31mERROR: Invalid choice.\033[0m\n");
      exit = true;
    }
  }

  return 0;
}
```


```
Root Directory Name: root
Select Choice:
1. Create file
2. Display file
3. Delete file
4. Search file
5. Exit
1
Enter filename: index.js
Enter size of file: 4

File Created
Select Choice:
1. Create file
2. Display file
3. Delete file
4. Search file
5. Exit
2
DIR: root
|-------|-----------|-----------|
| Index | File Name | File Size |
|-------|-----------|-----------|
|      1|   index.js|          4|
|-------|-----------|-----------|
Select Choice:
1. Create file
2. Display file
3. Delete file
4. Search file
5. Exit
4
Enter filename: index.js
```



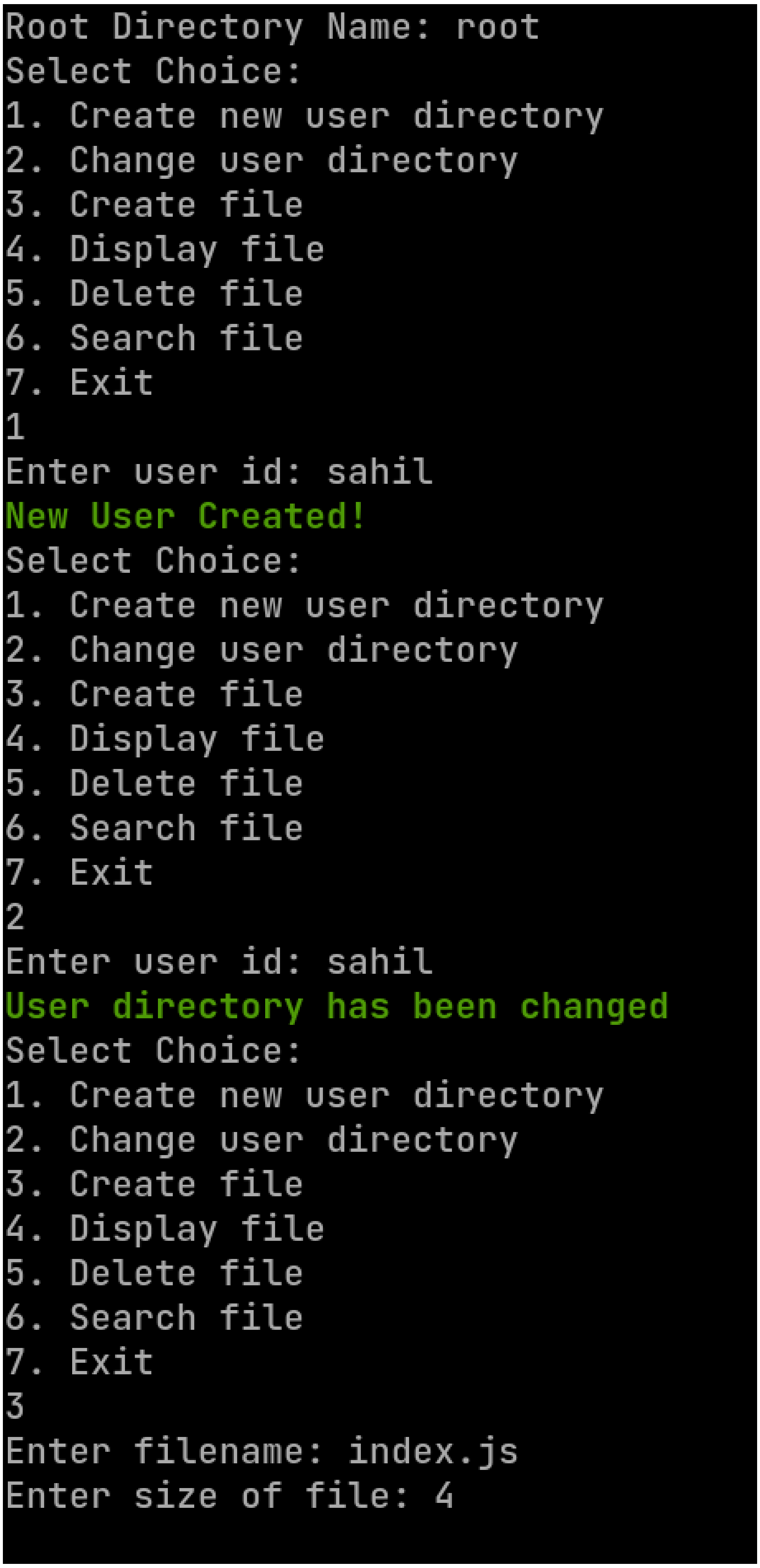
```
DIR: root
|-----------|-----------|
| File Name | File Size |
|-----------|-----------|
|   index.js|          4|
|-----------|-----------|
Select Choice:
1. Create file
2. Display file
3. Delete file
4. Search file
5. Exit
3
Enter filename: index.js

File Deleted
Select Choice:
1. Create file
2. Display file
3. Delete file
4. Search file
5. Exit
2
DIR: root
|-------|-----------|-----------|
| Index | File Name | File Size |
|-------|-----------|-----------|
|-------|-----------|-----------|
Select Choice:
1. Create file
2. Display file
3. Delete file
4. Search file
5. Exit
5
```

Double Level Directory:

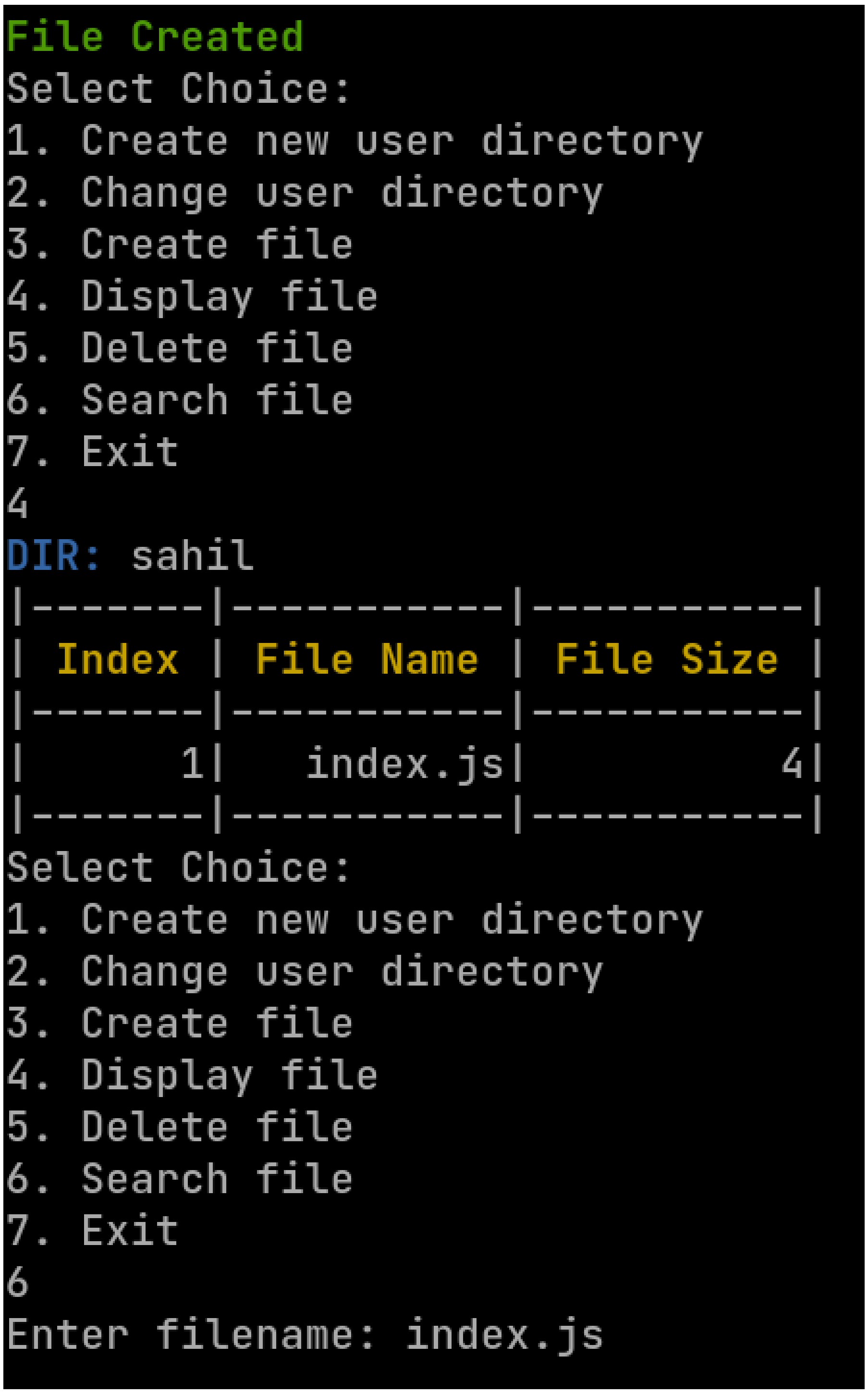
```cpp
#include <bits/stdc++.h>

using namespace std;

void createUserDirectory(string id) {
  userID = id;
  bool found = false;
  for (int i = 0; i < directory->userDirectories.size(); ++i) {
    if (directory->userDirectories[i].id == id) {
      found = true;
      printf(
          "\x1B[1;31mERROR: User directory already exists with this "
          "id\033[0m\n");
      break;
    }
  }
  if (!found) {
    UserFileDirectory userDirectory;
    userDirectory.id = id;
    userDirectory.name = id;
    userDirectory.files = {};
    directory->userDirectories.push_back(userDirectory);
    printf("\x1B[1;32mNew User Created!\033[0m\n");
  }
}

void changeUserDirectory(string id) {
  bool found = false;
  for (auto itr = directory->userDirectories.begin();
       itr < directory->userDirectories.end(); ++itr) {
    int idx = itr - directory->userDirectories.begin();
    if (directory->userDirectories[idx].id == id) {
      userID = id;
      printf("\x1B[1;32mUser directory has been changed\033[0m\n");
      found = true;
      break;
    }
  }

  if (!found) printf("\x1B[1;31mERROR: No such user
directory\033[0m\n");
}

void createFile(File file) {
  bool found = false, fileExists = false;
  for (auto itr = directory->userDirectories.begin();
       itr < directory->userDirectories.end(); ++itr) {
    int idx = itr - directory->userDirectories.begin();
    if (directory->userDirectories[idx].id == userID) {
      UserFileDirectory userDirectory = directory->userDirectories[idx];
      for (int i = 0; i < userDirectory.files.size(); ++i) {
        if (userDirectory.files[i].name == file.name) {
          printf(
              "\x1B[1;31mERROR: File already exists in the
directory\033[0m\n");
          found = true;
          fileExists = true;
          break;
        }
      }

      if (!fileExists) {
        userDirectory.files.push_back(file);
        directory->userDirectories[idx] = userDirectory;

        printf("\x1B[1;32mFile Created\033[0m\n");
        found = true;
        break;
      }
    }
  }

  if (!found) printf("\x1B[1;31mERROR: No such user directory
exists\033[0m\n");
}

void displayFiles() {
  for (auto itr = directory->userDirectories.begin();
       itr < directory->userDirectories.end(); ++itr) {
    int idx = itr - directory->userDirectories.begin();
    if (directory->userDirectories[idx].id == userID) {
      UserFileDirectory userDirectory = directory->userDirectories[idx];
      printf("\x1B[1;34mDIR: \033[0m");

      cout << userDirectory.name << endl;
      cout << "|-------|-----------|-----------|\n";

      cout << '|';
      printf("\x1B[1;33m Index \033[0m");
      cout << '|';
      printf("\x1B[1;33m File Name \033[0m");
      cout << '|';
      printf("\x1B[1;33m File Size \033[0m");
      cout << '|' << endl;
      cout << "|-------|-----------|-----------|\n";
      for (auto it = userDirectory.files.begin();
           it < userDirectory.files.end(); ++it) {
        int j = it - userDirectory.files.begin();
        File file = userDirectory.files[j];
        cout << '|' << setw(7) << j + 1 << '|' << setw(11) << file.name
<< '|'
             << setw(11) << file.size << '|' << endl;
      }
      cout << "|-------|-----------|-----------|\n";
    }
  }
}

void deleteFile(string fileName) {
  bool found = false;
  for (auto itr = directory->userDirectories.begin();
       itr < directory->userDirectories.end(); ++itr) {
    int idx = itr - directory->userDirectories.begin();
    if (found) break;
    if (directory->userDirectories[idx].id == userID) {
      UserFileDirectory userDirectory = directory->userDirectories[idx];
      for (auto it = userDirectory.files.begin();
           it < userDirectory.files.end(); ++it) {
        int i = it - userDirectory.files.begin();
        if (userDirectory.files[i].name == fileName) {
          userDirectory.files.erase(it);
          directory->userDirectories[idx] = userDirectory;
          printf("\x1B[1;31mFile Deleted\033[0m\n");
          found = true;
          break;
        }
      }
    }
  }

  if (!found)
    printf(
        "\x1B[1;31mERROR: File doesn't exist in the user
directory\033[0m\n");
}
```

```
void searchFile(string fileName) {
  bool found = false;
  File file;
  string directoryName;
  for (auto itr = directory->userDirectories.begin();
       itr < directory->userDirectories.end(); ++itr) {
    int idx = itr - directory->userDirectories.begin();
    if (found) break;
    if (directory->userDirectories[idx].id == userID) {
      UserFileDirectory userDirectory = directory->userDirectories[idx];
      for (auto it = userDirectory.files.begin();
           it < userDirectory.files.end(); ++it) {
        int i = it - userDirectory.files.begin();
        if (userDirectory.files[i].name == fileName) {
          file = userDirectory.files[i];
          directoryName = userDirectory.name;
          found = true;
          break;
        }
      }
    }
  }

  if (!found)
    printf("\x1B[31mFile doesn't exist in the user directory\033[0m\n");
  else {
    printf("\x1B[1;34mDIR: \033[0m");
    cout << directoryName << endl;
    cout << "|-----------|-----------|\n";

    cout << '|';
    printf("\x1B[1;33m File Name \033[0m");
    cout << '|';
    printf("\x1B[1;33m File Size \033[0m");
    cout << '|' << endl;
    cout << "|-----------|-----------|\n";
    cout << '|' << setw(11) << file.name << '|' << setw(11) << file.size
<< '|'
         << endl;
    cout << "|-----------|-----------|\n";
  }
}

class File {
 public:
  string name;
```

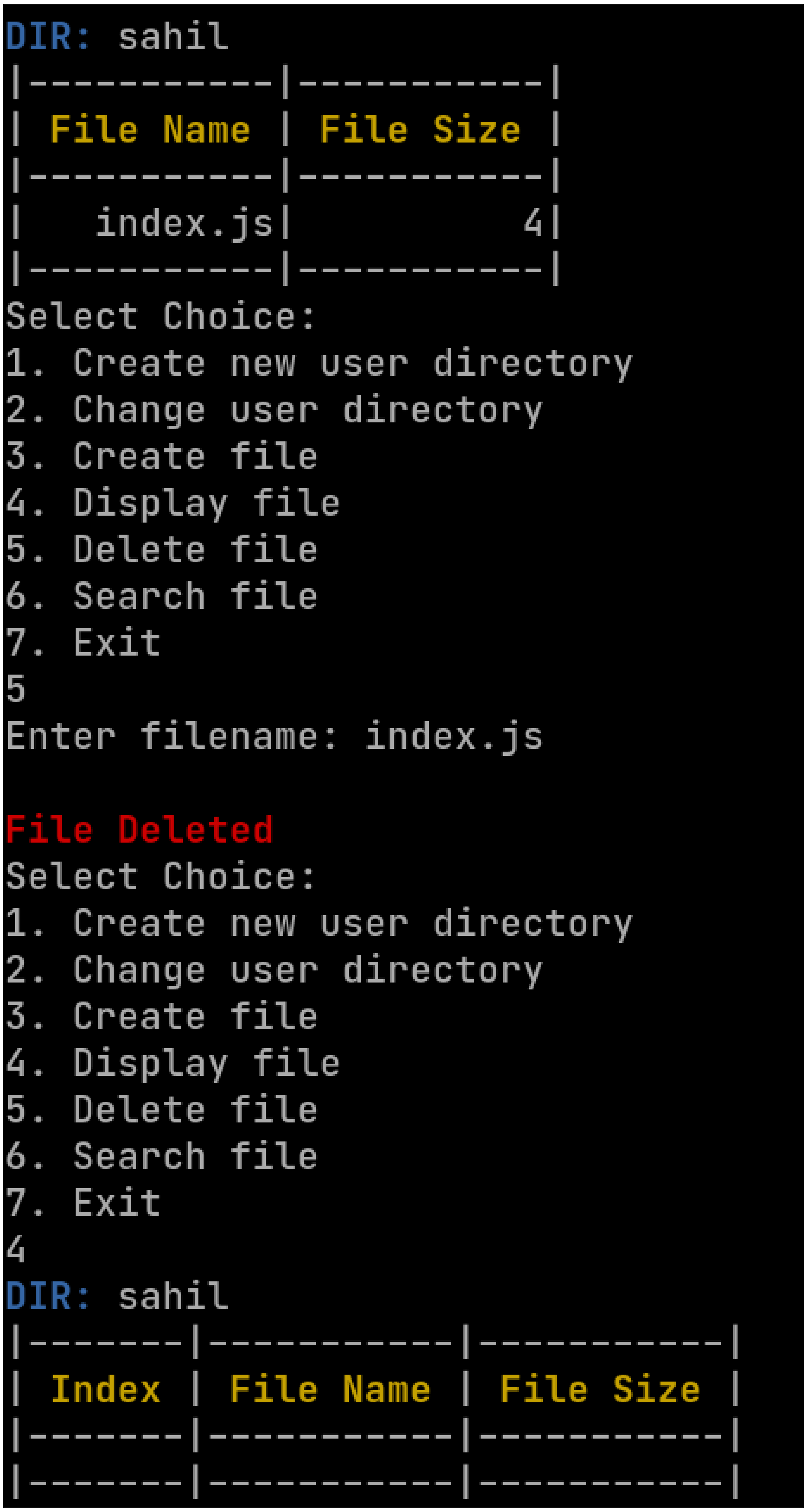
```cpp
  long long size;
};

class UserFileDirectory {
 public:
  string id;
  string name;
  vector<File> files;
};

class MasterDirectory {
 public:
  string name;
  vector<UserFileDirectory> userDirectories;
};

MasterDirectory* directory = new MasterDirectory();
string userID;


int main() {
  cout << "Root Directory Name: ";
  getline(cin, directory->name);

  // single level
  int choice = 1;
  bool exit = false;

  while (!exit) {
    cout << "Select Choice: "
         << "\n"
         << "1. Create new user directory"
         << "\n"
         << "2. Change user directory"
         << "\n"
         << "3. Create file"
         << "\n"
         << "4. Display file"
         << "\n"
         << "5. Delete file"
         << "\n"
         << "6. Search file"
         << "\n"
         << "7. Exit" << endl;
    cin >> choice;
    if (choice == 1) {
      string id;
      cout << "Enter user id: ";
      cin >> id;
      createUserDirectory(id);

    } else if (choice == 2) {
      string id;
      cout << "Enter user id: ";
      cin >> id;
      changeUserDirectory(id);

    } else if (choice == 3) {
      File file;
      cout << "Enter filename: ";
      cin >> file.name;
      cout << "Enter size of file: ";
      cin >> file.size;
      cout << endl;
      createFile(file);

    } else if (choice == 4) {
      displayFiles();

    } else if (choice == 5) {
      string fileName;
      cout << "Enter filename: ";
      cin >> fileName;
      cout << endl;
      deleteFile(fileName);

    } else if (choice == 6) {
      string fileName;
      cout << "Enter filename: ";
      cin >> fileName;
      cout << endl;
      searchFile(fileName);

    } else if (choice == 7) {
      exit = true;
    } else {
      printf("\x1B[1;31mERROR: Invalid choice.\033[0m\n");
      exit = true;
    }
  }
  return 0;
}
```



```
Root Directory Name: root
Select Choice:
1. Create new user directory
2. Change user directory
3. Create file
4. Display file
5. Delete file
6. Search file
7. Exit
1
Enter user id: sahil
New User Created!
Select Choice:
1. Create new user directory
2. Change user directory
3. Create file
4. Display file
5. Delete file
6. Search file
7. Exit
2
Enter user id: sahil
User directory has been changed
Select Choice:
1. Create new user directory
2. Change user directory
3. Create file
4. Display file
5. Delete file
6. Search file
7. Exit
3
Enter filename: index.js
Enter size of file: 4
```



```
File Created
Select Choice:
1. Create new user directory
2. Change user directory
3. Create file
4. Display file
5. Delete file
6. Search file
7. Exit
4
DIR: sahil
|-------|-----------|-----------|
| Index | File Name | File Size |
|-------|-----------|-----------|
|      1|   index.js|          4|
|-------|-----------|-----------|
Select Choice:
1. Create new user directory
2. Change user directory
3. Create file
4. Display file
5. Delete file
6. Search file
7. Exit
6
Enter filename: index.js
```



```
DIR: sahil
|-----------|-----------|
| File Name | File Size |
|-----------|-----------|
|   index.js|          4|
|-----------|-----------|
Select Choice:
1. Create new user directory
2. Change user directory
3. Create file
4. Display file
5. Delete file
6. Search file
7. Exit
5
Enter filename: index.js

File Deleted
Select Choice:
1. Create new user directory
2. Change user directory
3. Create file
4. Display file
5. Delete file
6. Search file
7. Exit
4
DIR: sahil
|-------|-----------|-----------|
| Index | File Name | File Size |
|-------|-----------|-----------|
|-------|-----------|-----------|
```