**TRAVERSING**

**DIRECTORIES**

**LAB # 0****9**

**Fall 2023**

**CSE-302L**

**Systems Programming Lab**

Submitted by: **AIMAL KHAN**

Registration No.: **21PWCSE1996**

Class Section: **A**

“On my honor, as student of University of Engineering and Technology, I have neither given nor received unauthorized assistance on this academic work.”



Student Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_

Submitted to:

**Engr. Abdullah Hamid**

Sunday, January 28, 2024

Department of Computer Systems Engineering

University of Engineering and Technology, Peshawar

**CSE 302L: SYSTEMS PROGRAMMING LAB**

**LAB ASSESSMENT RUBRICS**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Criteria & Point Assigned** | **Outstanding 2** | **Acceptable 1.5** | **Considerable 1** | **Below Expectations 0.5** | **Score** |
| **Attendance and**  **Attentiveness in Lab**  PLO08 | Attended in proper Time and attentive in Lab | Attended in proper Time but not attentive in Lab | Attended late but attentive in Lab | Attended late not attentive in Lab |  |
| **Capability of writing Program/Algorithm/Drawing Flow Chart**  PLO1, PLO2, PLO3, PLO5 | Right attempt/ no errors and well formatted | Right attempt/ no errors but not well formatted | Right attempt/ minor errors and not well formatted | Wrong attempt |  |
| **Result or Output/ Completion of target in Lab**  PLO9 | 100% target has been completed and well formatted. | 75% target has been completed and well formatted. | 50% target has been completed but not well formatted. | None of the outputs are correct. |  |
| **Overall, Knowledge**  PLO10, | Demonstrates excellent knowledge of lab | Demonstrates good knowledge of lab | Has partial idea about the Lab and procedure followed | Has poor idea about the Lab and procedure followed |  |
| **Attention to Lab Report**  PLO4, | Submission of Lab Report in Proper Time i.e., in next day of lab, with proper documentation. | Submission of Lab Report in proper time but not with proper documentation. | Late Submission with proper documentation. | Late Submission very poor documentation |  |

**Instructor:**

|  |  |
| --- | --- |
| Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ | Signature: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  |  |

**TRAVERSING DIRECTORIES**

Objectives:

The objectives of this lab are to gain a practical understanding of key system programming concepts, including

* Traversing Directories
* Traverse directory tree in depth-first order.
* Traverse directory tree in breadth-first order.

Tasks:

**Task 1** : Traverse directory tree in depth-first order

**Code in C:**

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <sys/stat.h>

#include <dirent.h>

#include <string.h>

void depthFirstTraverse(const char \*path, int depth)

{

struct stat entryStatistics;

struct dirent \*directoryEntry;

DIR \*dp;

if ((dp = opendir(path)) == NULL)

{

perror("Error while opening directory.\n");

return;

}

while ((directoryEntry = readdir(dp)) != NULL)

{

char newPath[1024];

if ((!strcmp(directoryEntry->d\_name, ".")) || (!strcmp(directoryEntry->d\_name, "..")))

continue;

for (int i = 0; i < depth; i++)

{

printf(" ");

}

S\_ISDIR(entryStatistics.st\_mode) ? printf("%s:\n", directoryEntry->d\_name) : printf("%s\n", directoryEntry->d\_name);

snprintf(newPath, sizeof(newPath), "%s/%s", path, directoryEntry->d\_name);

if (stat(newPath, &entryStatistics) == -1)

{

perror("Error while traversing statistics.\n");

return;

}

if (S\_ISDIR(entryStatistics.st\_mode))

{

depthFirstTraverse(newPath, depth + 1);

}

}

closedir(dp);

}

int main(int argc, char \*argv[])

{

if (argc != 2)

{

fprintf(stderr, "Need at exactly one arg. Usage:\n%s [DIR\_PATH]\n", argv[0]);

return 1;

}

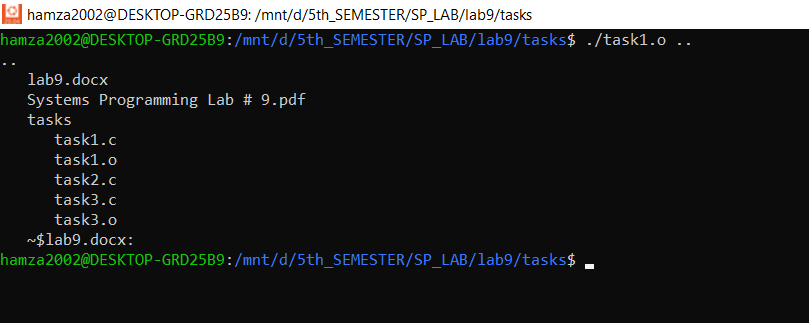
printf("%s\n", argv[1]);

depthFirstTraverse(argv[1], 1);

return 0;

}

**Output:**

****

**Task 2**: Traverse directory tree in breadth-first order.

**Code in C:**

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <sys/stat.h>

#include <dirent.h>

#include <string.h>

// TODO Complete this function

void breadthFirstTraverse(char \*path)

{

struct stat entryStatistics;

struct dirent \*directoryEntry;

DIR \*dp;

chdir(path);

if ((dp = opendir(path)) == NULL)

{

perror("Error while opening directory.\n");

return;

}

while ((directoryEntry = readdir(dp)) != NULL)

{

if ((!strcmp(directoryEntry->d\_name, ".")) || (!strcmp(directoryEntry->d\_name, "..")))

continue;

char newPath[1024];

snprintf(newPath, sizeof(newPath), "%s/%s", path, directoryEntry->d\_name);

if (stat(newPath, &entryStatistics) == -1)

{

perror("Error while traversing statistics.\n");

return;

}

if (S\_ISDIR(entryStatistics.st\_mode))

{

breadthFirstTraverse(newPath);

}

}

closedir(dp);

}

int main(int argc, char \*argv[])

{

if (argc != 2)

{

fprintf(stderr, "Usage: %s <directory\_path>\n", argv[0]);

return EXIT\_FAILURE;

}

char \*startPath = argv[1];

breadthFirstTraverse(startPath);

return 0;

}

**Output:**

**A computer screen with text on it

Description automatically generated**

**Task 3**: Implement the pfind utility

**Code in C:**

#include <stdio.h>

#include <stdlib.h>

#include <unistd.h>

#include <sys/stat.h>

#include <dirent.h>

#include <string.h>

void searchFile(const char \*path, const char \*target)

{

struct stat entryStatistics;

struct dirent \*directoryEntry;

DIR \*dp;

// open path(directory)

if ((dp = opendir(path)) == NULL)

{

perror("Error while opening directory.\n");

return;

}

// read the directory entries one by one

while ((directoryEntry = readdir(dp)) != NULL)

{

char newPath[1024];

// skip . and ..

if ((!strcmp(directoryEntry->d\_name, ".")) || (!strcmp(directoryEntry->d\_name, "..")))

continue;

// S\_ISDIR(entryStatistics.st\_mode) ? printf("%s:\n", directoryEntry->d\_name) : printf("%s\n", directoryEntry->d\_name);

// update the path by appending the file name with it.

snprintf(newPath, sizeof(newPath), "%s/%s", path, directoryEntry->d\_name);

if (stat(newPath, &entryStatistics) == -1)

{

perror("Error while traversing statistics.\n");

return;

}

if (S\_ISDIR(entryStatistics.st\_mode)) // if directory search for file in subdirectories.

searchFile(newPath, target);

else if (S\_ISREG(entryStatistics.st\_mode)) // if regular file compare the name of entry and target.

if (!strcmp(target, directoryEntry->d\_name))

printf("%s/%s\n", path, directoryEntry->d\_name);

}

if (closedir(dp) == -1)

{

perror("Error while opening directory.\n");

return;

}

}

int main(int argc, char \*argv[])

{

if (argc != 3)

{

fprintf(stderr, "Need at exactly two arg. Usage:\n%s [DIR\_PATH] [TARGET\_FILE]\n", argv[0]);

return 1;

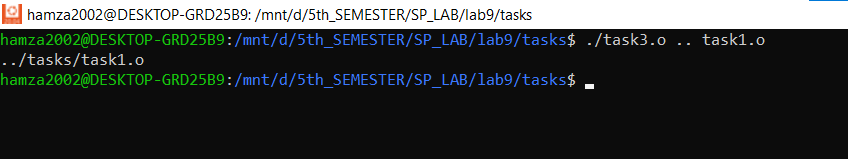
}

searchFile(argv[1], argv[2]);

return 0;

}

**Output:**

****

Reference:

To view my codes, please refer to my GitHub account:  [https://github.com/aimalexe/DCSE/tree/main/semester\_5\_(fall-23)/systems\_programming\_lab/lab\_reports](%20https://github.com/aimalexe/DCSE/tree/main/semester_5_(fall-23)/systems_programming_lab/lab_reports) .

Conclusion:

In summary, this laboratory experience has provided a comprehensive exploration of various fundamental system programming concepts, including traversing directories ,traverse directory tree in depth-first order and traverse directory tree in breadth-first order.

The End.