FANSHAWE
School of Applied
Science & Technology

FOLDER ANALYZER

PURPOSE

In this exercise, filesystem access is practiced. The user is queried for a path, and after validation it is analyzed. The number of folders is counted, and the file names are printed out as well as their sizes. The program will also ask for a file size in bytes as a limit, which will then be used to count how many files exceed that limit.

OBJECTIVES

After completing this exercise, you should be able to:

- Iterate through a file system
- Create and manage file paths
- Retrieve a file name from a full path
- Retrieve file sizes
- Determine whether a directory entry is a folder or a file

PROCEDURE

PREPARE SUBMISSION FILE

1. Create a copy of the submission template called COMP6060**INIT**Lab11.docx where **INIT** is replaced with your own initials. So if your name is John Smith, the document will be called COMP6060**JS**Lab11.docx

PREPARE PYTHON FILE

1. Create a Python file called COMP6060**INIT**Lab11.py where **INIT** is replaced with your own initials. So if your name is John Smith, the document will be called COMP6060**JS**Lab11.py
2. Print out the following to the console, replacing NAME with your name: `Welcome to NAME's folder analyzer program!`
3. Import the `os` library

Prompt User For Path

1. Prompt the user for a path name to analyze, and store the path in a variable called `user_path`. Use the following prompt:
   `Enter a valid folder path to be analyzed:`
2. Validate `user_path` to ensure it exists.
   If the path doesn't exist, print the following message, then exit the program:
   `Error: The path /user/path does not exist`
   replacing `/user/path` with the value of `user_path`
3. Validate `user_path` to ensure it is a folder
   If the path is not a folder, print the following message, then exit the program:
   `Error: The path /user/path is not a folder`
   replacing `/user/path` with the value of `user_path`

Prompt User For Limit

1. Prompt the user for a file size limit, and store it in a variable called `file_size_limit_str`:
   `Enter a file size in bytes:`
2. Validate that `file_size_limit_str` contains a number
   a. If the string is not a number, print the following message, then exit the program:
      `Error: The value is not a number.`
3. Cast `file_size_limit_str` to an integer value, and store in a variable called `file_size_limit`

Change Current Working Directory

1. Print the current working directory:
   `Current working directory: /path/to/directory`
   where `/path/to/directory` is the current working directory
2. Change the current working directory to `user_path`
3. Print the current working directory after the change:
   `Changed current working directory: /path/to/directory`
   where `/path/to/directory` is the current working directory

Iterate Through User Path

1. Create the following variables, and assign them all to the 0:
    a. `folder_count`
    b. `file_count`
    c. `full_size`
    d. `file_limit_count`
2. Using the function `os.scandir()`, iterate through the current working directory in a for loop (use relative path). Use the variable name `entry` as the for loop variable
3. Inside the for loop, check the type of `entry`:
    a. If the `entry` is a folder, increment the `folder_count` variable
    b. If the `entry` is a file:
        i. Increment the `file_count` variable
        ii. Store the `entry` file size in a variable called `entry_size_bytes`
        Hint: Use the function `os.path.getsize()`
        iii. Add `entry_size_bytes` to `full_size`
        iv. Check if the file size is greater than or equal to `file_size_limit`. Increment `file_limit_count` if it is.
        v. Store the `entry` file name in a variable called `entry_filename`
        Hint: use the `os.path.split()` function

Print Results

1. Print the file count as follows:
   `Files found in /user/path: x`
   replacing */user/path* with the value of `user_path`
2. Print the folder count as follows:
   `Folders found in /user/path: x`
   replacing */user/path* with the value of `user_path`
3. `Print the full folder size as follows:`
   `All files in /user/path take up: x bytes`
   replacing */user/path* with the value of `user_path`
4. Print the number of files that exceeded the file size limit:
   `Files in /user/path that exceeded file_size_limit: y`
   replacing */user/path* with the value of `user_path`, x with the value of `file_size_limit`, and y with `file_limit_count`

FANSHAWE
School of Applied
Science & Technology

Expected Results

Valid Data

```
Welcome to Lynn's folder analyzer program!
Enter a valid folder path to be analyzed: C:/Test
Enter a file size in bytes: 100
Current working directory: C:\Users\Lynn\Documents\OneDrive\Fanshawe\2022
Changed current working directory: C:\Test
Files found in C:/Test: 5
Folders found in C:/Test: 2
All files in C:/Test take up: 67125964 bytes
Files in C:/Test that exceeded 100: 5
```

Path does not exist

```
Welcome to Lynn's folder analyzer program!
Enter a valid folder path to be analyzed: /does/not/exist
The path /does/not/exist does not exist
```

Path is not a folder

```
Welcome to Lynn's folder analyzer program!
Enter a valid folder path to be analyzed: C:/test/data.txt
The path C:/test/data.txt is not a folder
```

Invalid Limit Value

```
Welcome to Lynn's folder analyzer program!
Enter a valid folder path to be analyzed: C:/test
Enter a file size in bytes: one hundred
Error: value is not a number
```

Show results to Instructor.

Student Name: _____ Instructor: _____

Date: _____