FANSHAWE
School of Applied
Science & Technology

FILE ORGANIZER

PURPOSE

In this exercise, filesystem management is exercised. Files in a given folder are moved to appropriate folders based on the file type.

OBJECTIVES

After completing this exercise, you should be able to:

- Create directories in Python
- Move files and folders in Python
- Copy files and folders in Python
- Delete files and folders in Python

PROCEDURE

PREPARE SUBMISSION FILE

1. Create a copy of the submission template called COMP6060**INIT**Lab12.docx where **INIT** is replaced with your own initials. So if your name is John Smith, the document will be called COMP6060**JS**Lab12.docx

PREPARE PYTHON FILE

1. Download the Lab 12 data folder from FOL, and place it in C:/Users/Public/COMP-6060
2. Create a Python file called COMP6060**INIT**Lab12.py where **INIT** is replaced with your own initials. So if your name is John Smith, the document will be called COMP6060**JS**Lab12.py
3. Print out the following to the console, replacing NAME with your name:
   `Welcome to NAME's file organizer!`

PROMPT USER FOR FOLDER

1. Prompt the user for a folder name to organize, and store the result in a variable called `path_to_organize`. Use the following message:
   `Enter a path to a folder to organize:`
2. Verify that the folder exists.
   a. If the folder does not exist, print out a descriptive error statement then exit the program.

CREATE FOLDERS

1. Create a try-except block as follows:
```
try:
    # code goes here
except FileExistsError:
    print("Folder exists, not creating…")
```
2. Inside the try-except block, create a folder in `path_to_organize` called **Documents**
3. After the first try-except block, create a similar try-except block to create a folder in `path_to_organize` called **Copied_Images**

SCAN FOLDER

1. After creating all 3 folders, scan the contents of path_to_organize using a for loop and `os.scandir()`, and do the following:
    a. If the directory entry is a file:
        i. Retrieve the extension of the file using the following code: `file_extension = os.path.splitext(entry.path)[1]`
        ii. If `file_extension` is ".txt" or ".docx":
            1. Create the destination folder path by joining `path_to_organize` with Documents, creating the variable `dest`
               Hint: use `os.path.join()`
            2. Move the entry to `dest`
            3. Print the following message:
               `Moved path`
               where *path* is the entry path
        iii. Otherwise, if `file_extension` is ".pdb":
            1. Delete the entry
            2. Print the following message:
               `Deleted path`
               where *path* is the entry path
        iv. Otherwise, if `file_extension` is ".png" or ".jpeg":
            1. Create the destination folder path by joining `path_to_organize` with Documents, creating the variable `dest`
               Hint: use `os.path.join()`
            2. Copy the entry to `dest`
            3. Print the following message:
               `Copied path`
               where *path* is the entry path
        v. Otherwise, if the extension is not supported, print the following message:
           ext `is not supported..`
           Where *ext* is file_extension

FANSHAWE
School of Applied
Science & Technology

EXPECTED RESULTS

```
Welcome to Lynn's File Organizer!
Enter a path to a folder to organize: C:/test
Moved C:/test\data.txt
.mp4 is not supported...
Moved C:/test\data2.txt
Deleted C:/test\debug.pdb
Copied C:/test\image.jpeg
Copied C:/test\image.png
```

Show results to Instructor.

Student Name: _____ Instructor: _____

Date: _____