¡Felicitaciones! ¡Aprobaste!

PARA APROBAR 80 % o más

Continúa aprendiendo

CALIFICACIÓN
100 %

Practice Quiz: Managing Files & Directories

PUNTOS TOTALES DE 5

1. The create_python_script function creates a new python script in the current working directory, adds the line of comments to it declared by the 'comments' variable, and returns the size of the new file. Fill in the gaps to create a script called "program.py". **1 / 1 puntos**

```
1   import os
2
3   def create_python_script(filename):
4       comments = "# Start of a new Python program"
5       file = open('program.py', 'a')
6       file.write(comments)
7       file.close()
8       filesize = os.path.getsize(filename)
9       return(filesize)
10
11  print(create_python_script("program.py"))
12
```

Ejecutar

Restablecer

Correcto

Great work! Your new python script is now ready for some real code!

2. The new_directory function creates a new directory inside the current working directory, then creates a new empty file inside the new directory, and returns the list of files in that directory. Fill in the gaps to create a file "script.py" in the directory "PythonPrograms". **1 / 1 puntos**

```
1   import os
2
3   def new_directory(directory, filename):
4     # Before creating a new directory, check to see if it already exists
5       os.makedirs(directory, exist_ok=True)
6       os.chdir(directory)
7       with open(filename, 'w') as file:
8           file.write("")
9       os.chdir("..")
10      return(os.listdir(directory))
11
12
13  print(new_directory("PythonPrograms", "script.py"))
```

Ejecutar

Restablecer

Correcto

Well done, you! Working with files and directories can be a little tricky, and you're getting the hang of it!

3. Which of the following methods from the os module will create a new directory? **1 / 1 puntos**

- path.isdir()
- listdir()
- mkdir()
- chdir()

Correcto

Right on! os.mkdir() will create a new directory with the name provided as a string parameter.

4. The file_date function creates a new file in the current working directory, checks the date that the file was modified, and returns just the date portion of the timestamp in the format of yyyy-mm-dd. Fill in the gaps to create a file called "newfile.txt" and check the date that it was modified. **1 / 1 puntos**

```
1   import os
2   import datetime
3
4   def file_date(filename):
5     # Create the file in the current directory
6     with open (filename,'w') as file:
7       pass
8     timestamp = os.path.getmtime(filename)
9     c=datetime.datetime.fromtimestamp(timestamp)
10    # Convert the timestamp into a readable format, then into a string
11
12    # Return just the date portion
13    # Hint: how many characters are in “yyyy-mm-dd”?
14    return ("{}".format(c.strftime("%Y-%m-%d")))
15
16  print(file_date("newfile.txt"))
17  # Should be today's date in the format of yyyy-mm-dd
```

Ejecutar

Restablecer

Correcto

Way to go! You remembered the commands to convert timestamps and format strings, to get the results that were requested.

5. The parent_directory function returns the name of the directory that's located just above the current working directory. Remember that '..' is a relative path alias that means "go up to the parent directory". Fill in the gaps to complete this function. **1 / 1 puntos**

```
1   import os
2   def parent_directory():
3       # Create a relative path to the parent
4       # of the current working directory
5       dir = os.getcwd()
6       relative_parent = os.path.join(dir)
7
8       # Return the absolute path of the parent directory
9       return os.path.dirname(relative_parent)
10
11  print(parent_directory())
```

Ejecutar

Restablecer

Correcto

Excellent! You made all the right moves to print the path of the parent directory!