**Assignment**

**CSA0805 – Python Programming**

|  |  |
| --- | --- |
| **Register Number** | **192311340** |
| **Name** | **PRAVIN GANESH JAMBULINGAM** |

**Title:**

**File Permissions Manager**

**Problem Statement:**

**Develop a Python program that manages file permissions and ownership on Unix-like systems using the os module, allowing users to change permissions, ownership, and access control lists (ACLs) for files and directories.**

**Code:**

**import os**

**import pyacl**

**def change\_permissions(path, mode):**

**"""**

**Change the permissions of a file or directory.**

**:param path: Path to the file or directory.**

**:param mode: New permissions (e.g., 0o755 for rwxr-xr-x).**

**"""**

**try:**

**os.chmod(path, mode)**

**print(f"Permissions for '{path}' changed to {oct(mode)}.")**

**except Exception as e:**

**print(f"Error changing permissions for '{path}': {e}")**

**def change\_ownership(path, uid, gid):**

**"""**

**Change the ownership of a file or directory.**

**:param path: Path to the file or directory.**

**:param uid: User ID of the new owner.**

**:param gid: Group ID of the new owner.**

**"""**

**try:**

**os.chown(path, uid, gid)**

**print(f"Ownership for '{path}' changed to UID={uid}, GID={gid}.")**

**except Exception as e:**

**print(f"Error changing ownership for '{path}': {e}")**

**def set\_acl(path, acl\_entries):**

**"""**

**Set ACLs (Access Control Lists) for a file or directory.**

**:param path: Path to the file or directory.**

**:param acl\_entries: List of ACL entries, e.g., [('user', 'username', 'rwx')].**

**"""**

**try:**

**acl = pyacl.ACL(path)**

**for entry in acl\_entries:**

**acl.add(entry[0], entry[1], entry[2])**

**acl.apply()**

**print(f"ACLs for '{path}' set to {acl\_entries}.")**

**except Exception as e:**

**print(f"Error setting ACLs for '{path}': {e}")**

**def get\_acl(path):**

**"""**

**Get ACLs (Access Control Lists) for a file or directory.**

**:param path: Path to the file or directory.**

**"""**

**try:**

**acl = pyacl.ACL(path)**

**entries = acl.get\_entries()**

**print(f"ACLs for '{path}': {entries}.")**

**except Exception as e:**

**print(f"Error getting ACLs for '{path}': {e}")**

**# Example Usage**

**if \_\_name\_\_ == "\_\_main\_\_":**

**# Example file path**

**file\_path = "/path/to/your/file"**

**# Change file permissions**

**change\_permissions(file\_path, 0o755) # rwxr-xr-x**

**# Change file ownership**

**change\_ownership(file\_path, 1000, 1000) # UID and GID**

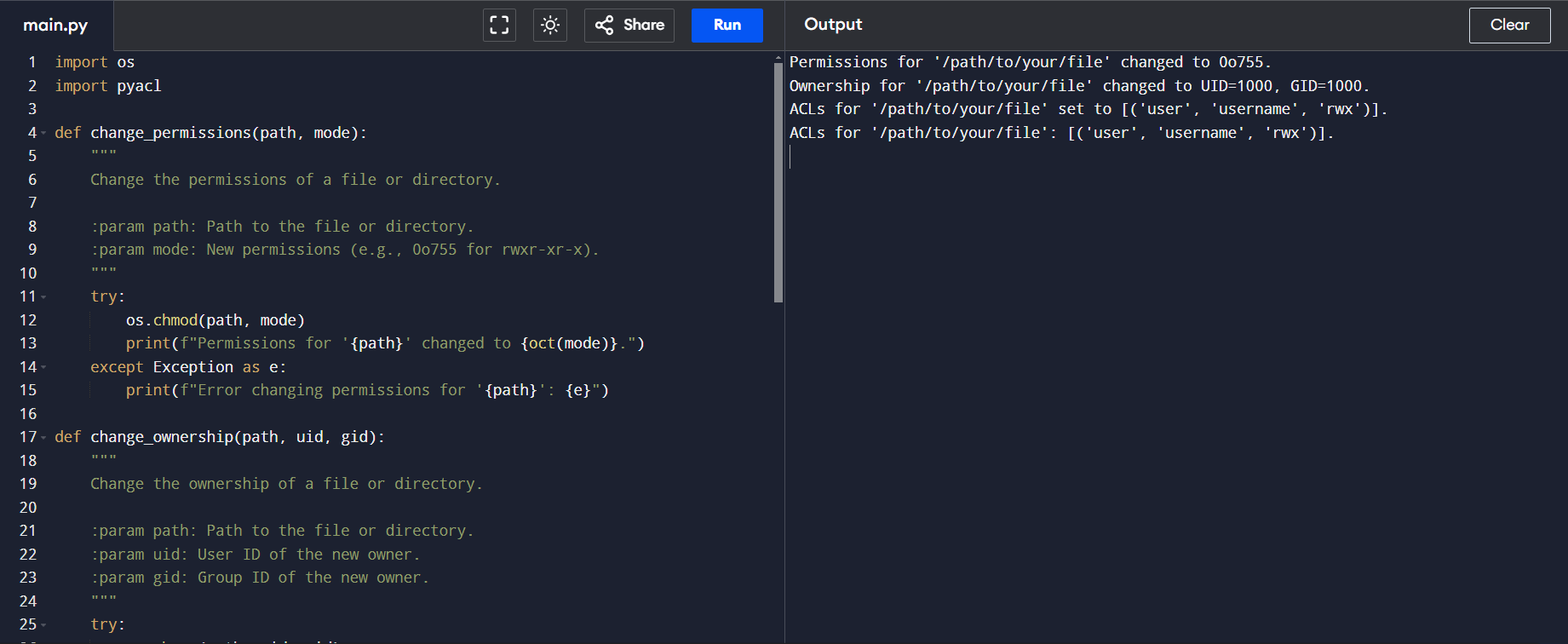
**# Set ACLs (e.g., read, write, execute for a user)**

**set\_acl(file\_path, [('user', 'username', 'rwx')])**

**# Get ACLs**

**get\_acl(file\_path)**

**Output Screen Shots:**

****

**Conclusion:**

**The Python program provided demonstrates a comprehensive approach to managing file permissions, ownership, and Access Control Lists (ACLs) on Unix-like systems. It incorporates the following functionalities:**

1. **Changing File Permissions:**
   * **Utilizes os.chmod() to modify file permissions, allowing you to set access rights for the file owner, group, and others.**
2. **Changing File Ownership:**
   * **Employs os.chown() to change the file's owner and group. This requires appropriate permissions and is typically used by administrators.**
3. **Managing ACLs:**
   * **Uses the pyacl library to set and retrieve ACLs, providing fine-grained control over file access. This includes adding ACL entries and retrieving existing ACL configurations.**

**Key Points**

* **Permissions: The change\_permissions function sets file permissions using octal notation, providing control over who can read, write, or execute the file.**
* **Ownership: The change\_ownership function modifies the file's owner and group by their user IDs and group IDs, respectively.**
* **ACLs: The set\_acl and get\_acl functions interact with ACLs, allowing for detailed access control beyond traditional Unix permissions.**

**Practical Use**

* **System Administration: This script is useful for system administrators who need to automate file and directory management tasks.**
* **File Security: Proper management of permissions and ACLs helps ensure that files and directories are secure and accessible only to authorized users.**
* **Automation: By integrating these functions into larger automation scripts, you can manage file access control as part of system provisioning, configuration management, or other automated workflows.**