```python
#creating a file
def create_file(filename):
    # Open the file in write mode
    with open(filename, 'w') as file:
        # Write some content to the file
        file.write("Hello, World!\n")
        file.write("This is a file handling example in Python.\n")
        file.write("Enjoy coding!")

    print(f"{filename} created and content written successfully.")

if __name__ == "__main__":
    create_file('example.txt')
```

```python
#reading a created file
def read_file(filename):
    # Open the file in read mode
    with open(filename, 'r') as file:
        content = file.read()  # Read the entire content
        print("File content:")
        print(content)

if __name__ == "__main__":
    create_file('example.txt')
    read_file('example.txt')
```

```python
#read the contents of the file
def read_file(filename):
    try:
        # Open the file in read mode
        with open(filename, 'r') as file:
            content = file.read()  # Read the entire content of the file
            print("File content:")
            print(content)
    except FileNotFoundError:
        print(f"The file '{filename}' does not exist.")
    except Exception as e:
        print(f"An error occurred: {e}")

if __name__ == "__main__":
    file_to_read = input("Enter the path of the file to read: ")
    read_file(file_to_read)
```

```
#locate a file
import os

def locate_file(filename, search_path):
    # Traverse the directory
    for dirpath, dirnames, files in os.walk(search_path):
        if filename in files:
            print(f"File found: {os.path.join(dirpath, filename)}")
            return
    print(f"File '{filename}' not found in '{search_path}'.")

if __name__ == "__main__":
    filename_to_search = input("Enter the name of the file to search: ")
    directory_to_search = input("Enter the directory to search in: ")
    locate_file(filename_to_search, directory_to_search)
```

```
#program to move file in file handling
import shutil
import os


def move_file(source, destination):
    try:
        # Check if the source file exists
        if not os.path.isfile(source):
            print(f"Source file '{source}' does not exist.")
            return

        # Move the file
        shutil.move(source, destination)
        print(f"File '{source}' moved to '{destination}' successfully.")
    except Exception as e:
```

```
        print(f"An error occurred: {e}")


if __name__ == "__main__":
    source_file = input("Enter the path of the file to move: ")
    destination_directory = input("Enter the destination directory: ")

    # Construct the full destination path
    destination_file = os.path.join(destination_directory, os.path.basename(source_file))

    move_file(source_file, destination_file)









#to delete python program in file handling

import os

def delete_file(file_path):
    try:
        # Check if the file exists
        if os.path.isfile(file_path):
            os.remove(file_path)  # Delete the file
            print(f"File '{file_path}' deleted successfully.")
        else:
            print(f"File '{file_path}' does not exist.")
    except Exception as e:
        print(f"An error occurred: {e}")

if __name__ == "__main__":
    file_to_delete = input("Enter the path of the file to delete: ")
    delete_file(file_to_delete)
```

Socket programming

```
import socket

def start_server():
    # Create a socket object
    server_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

    # Bind the socket to an address and port
    server_socket.bind(('localhost', 12345))

    # Listen for incoming connections
    server_socket.listen(5)
    print("Server listening on port 12345...")

    while True:
        # Accept a new connection
        client_socket, addr = server_socket.accept()
        print(f"Connection from {addr} has been established!")

        # Send a welcome message to the client
        client_socket.sendall(b"Hello, Client!")

        # Close the client socket
        client_socket.close()
```

```
if __name__ == "__main__":
    start_server()
```

```
#client
import socket

def start_client():
    # Create a socket object
    client_socket = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

    # Connect to the server
    client_socket.connect(('localhost', 12345))

    # Receive data from the server
    response = client_socket.recv(1024)
    print("Received from server:", response.decode())

    # Close the socket
    client_socket.close()

if __name__ == "__main__":
    start_client()
```