class LinearProbing:

def \_\_init\_\_(self, size):

self.size = size

self.keys = [None] \* size

self.collisions = [0] \* size

def hash\_func(self, key):

return key % self.size

def insert\_lp(self, key):

i = self.hash\_func(key)

i = i

while self.keys[i] is not None:

i = (i + 1) % self.size

self.collisions[i] += 1

if i == i+1:

print("Hash table is full. Cannot insert.")

return

self.keys[i] = key

def search\_lp(self, key):

i = self.hash\_func(key)

i = i

while self.keys[i] is not None:

if self.keys[i] == key:

return i

i = (i + 1) % self.size

if i == i:

break

return None

def display(self):

print("| Index | Key | Collisions |")

print("|-------|-----|------------|")

for i in range(self.size):

print(f"| {i} | {self.keys[i]} | {self.collisions[i]} |")

def main():

while True:

print("\nMain Menu")

print("Select one of these options:")

print("1) Linear Probing")

print("2) Exit")

menu\_choice = int(input("Enter your choice: "))

if menu\_choice == 1:

size = int(input("Enter the size of the hash table: "))

:wq

submenu(linear\_table, "Linear Probing")

elif menu\_choice == 2:

break

else:

print("Invalid choice. Please enter a valid option.")

def submenu(hash\_table, technique):

while True:

print(f"\n{technique} Menu")

print("Select one of these options:")

print("1) Insert")

print("2) Search")

print("3) Display")

print("4) Return to Main Menu")

choice = int(input("Enter your choice: "))

:wq

:wq:wq

::Wq

if choice == 1:

value = int(input("Enter the value to insert: "))

hash\_table.insert\_lp(value)

elif choice == 2:

value = int(input("Enter the value to search: "))

result = hash\_table.search\_lp(value)

if result is not None:

print(f"{technique}: Value found at i {result}.")

else:

print(f"{technique}: Value not found.")

elif choice == 3:

hash\_table.display()

elif choice == 4:

break

else:

print("Invalid choice. Please enter a valid option.")

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

main():wq