|  |
| --- |
| TRƯỜNG ĐẠI HỌC VINH  **VIỆN KỸ THUẬT CÔNG NGHỆ** |



BÁO CÁO THỰC HÀNH

**KỸ THUẬT LẬP TRÌNH**

MÃ HỌC PHẦN: ELE20004

SVTH: TRƯƠNG QUANG MẠNH

MSSV: 235752021610116

GVHD: MAI THẾ ANH

**NGHỆ AN - 2024**

**MỤC LỤC**

Bài 1. Thực hiện các thuật toán bằng phần mềm Flowgorithm.............................2

Bài 2. Các cú pháp, kiểu dữ liệu, lệnh điều khiển trong lập trình Python.............4

Bài 3. Lập trình hàm trong Python.......................................................................10

Bài 4. Các kiểu dử liệu trong Python...................................................................15

Bài 5. Thiết kế module trong Python...................................................................25

Bài 6. Lập trình hướng đối tượng trong Python...................................................29

Bài 7. Thao tác trên tập tin và thư mục trong Python..........................................32

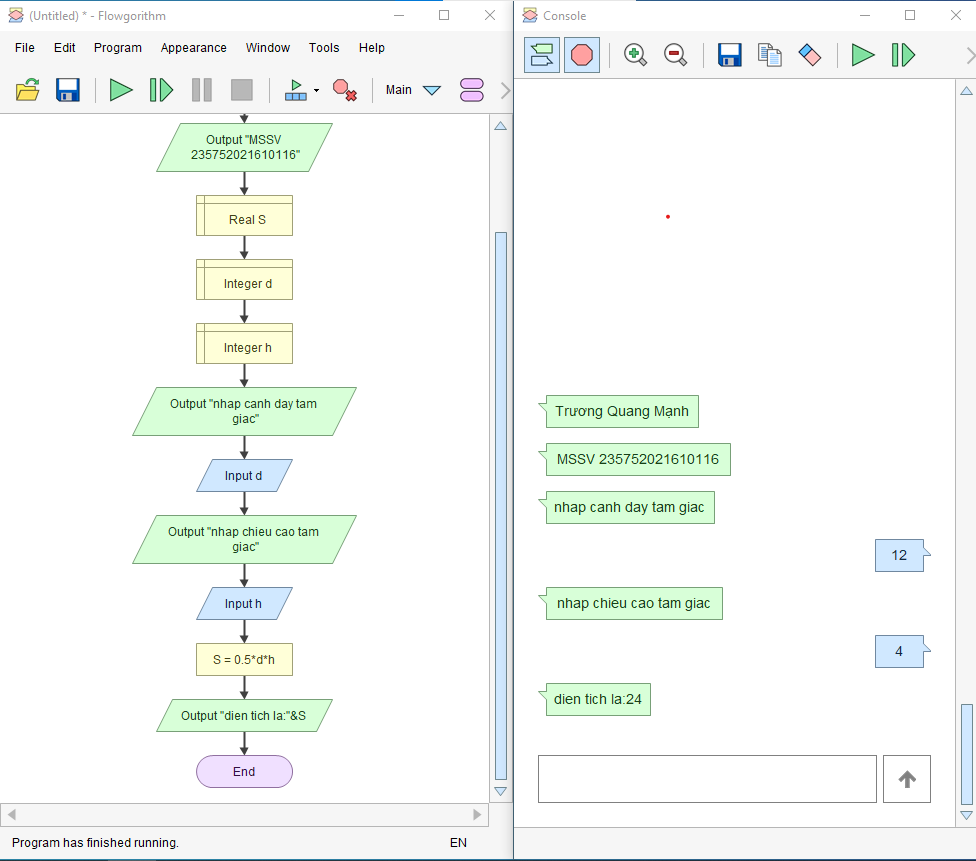
Bài 8. Lập trình giao diện trong Python ..............................................................36

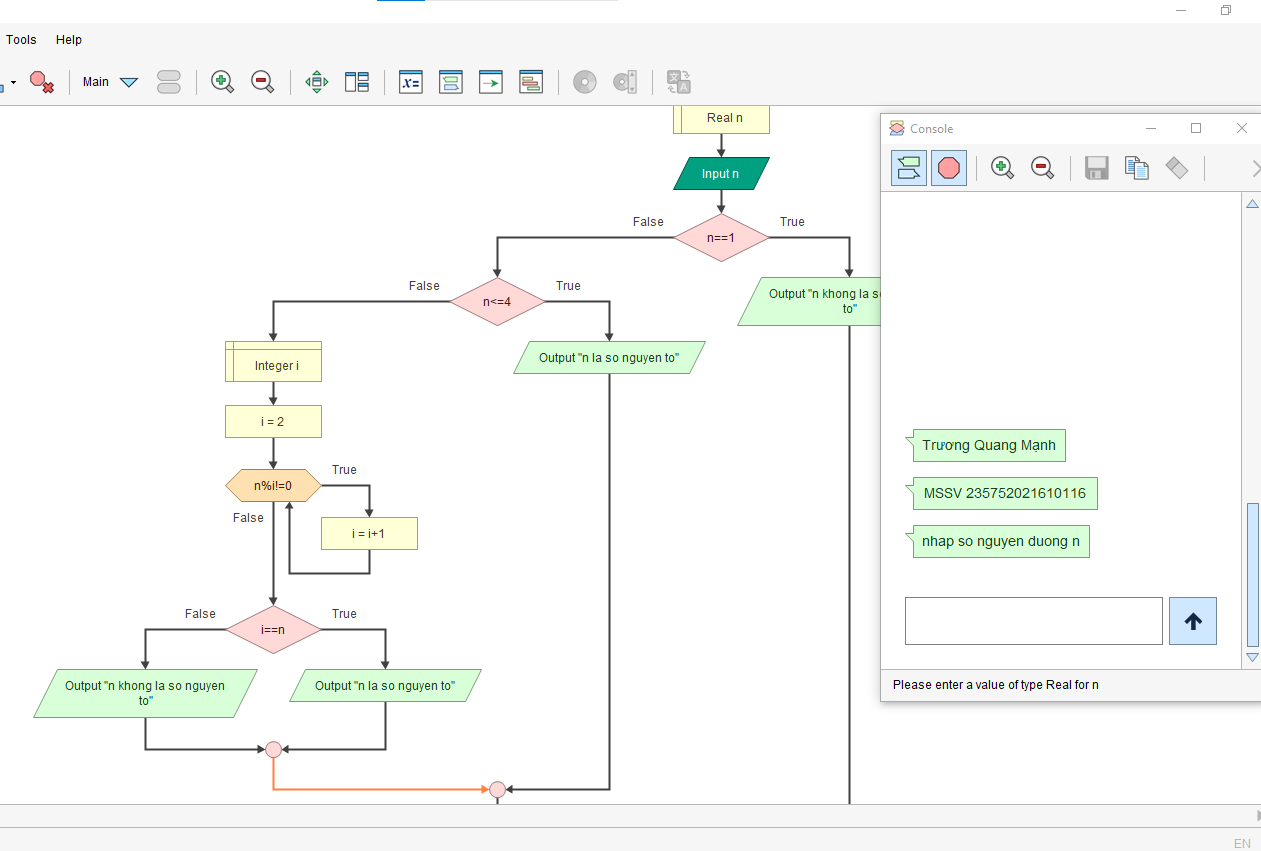
# **Bài 1. Thực hiện các thuật toán bằng phần mềm Flowgorithm**

* 1. **Mục đích**
* Sử dụng phần mềm Flowgorithm trong thiết kế và biểu diễn thuật toán;
* Xây dựng thuật toán cho các bài toán cụ thể trên Flowgorithm
  1. **Các bước thực hiện và kết quả**

*(Tóm tắt các bước thực hiện và kết quả nhận được trong mỗi bước, giải thích theo yêu cầu, trình bày ngắn gọn đúng trọng tâm)*

*(Có thể chụp màn hình nhưng phải bổ sung thêm khối in thông tin sinh viên như họ và tên, MSSV ra màn hình theo mẫu)*

**

**

* 1. **Câu hỏi kiểm tra**

Trả lời ngắn gọn các câu hỏi theo yêu cầu.

* 1. **Tài liệu tham khảo**

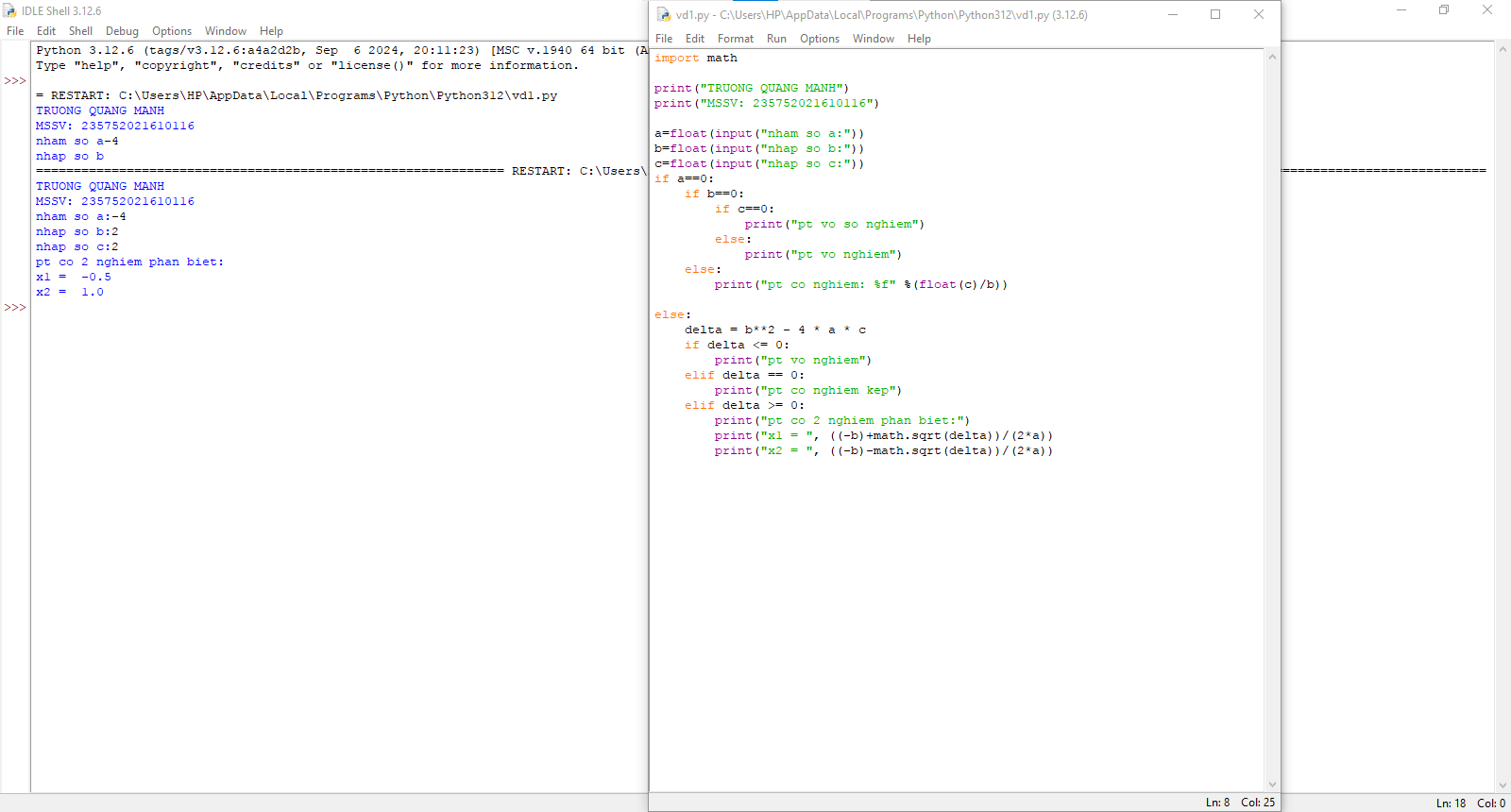
# **Bài 2. Các cú pháp, kiểu dữ liệu, lệnh điều khiển trong lập trình Python**

* 1. **Mục đích**

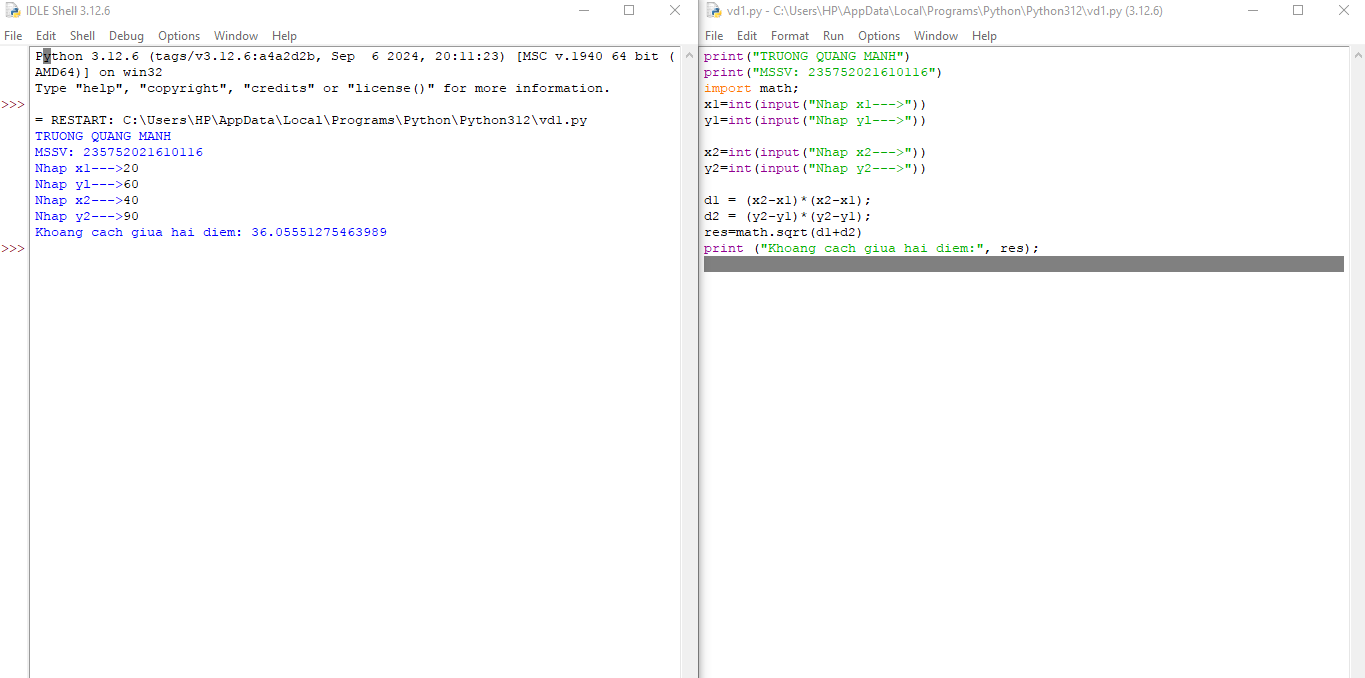
.

* 1. **Các bước thực hiện và kết quả**

**Bài 1. Viết chương trình giải phương trình bậc 2: ax2 +bx+c=0, với các hệ số a, b, c nhập từ bàn phím**



**Bài 2. Viết chương trình nhập hai điểm và tính khoảng cách**

****

**Các bước thực hiện:**

 **Nhập thư viện math**:

import math

* Thư viện math được sử dụng để sử dụng các hàm toán học, trong trường hợp này là hàm sqrt để tính căn bậc hai.

 **Nhập tọa độ của điểm 1**:

x1 = int(input("Nhập x1: "))

y1 = int(input("Nhập y1: "))

* Sử dụng input() để nhận tọa độ x1 và y1 của điểm đầu tiên từ người dùng. Hàm int() được dùng để chuyển đổi giá trị nhập vào thành số nguyên.

 **Nhập tọa độ của điểm 2**:

x2 = int(input("Nhập x2: "))

y2 = int(input("Nhập y2: "))

* Tương tự, nhận tọa độ x2 và y2 của điểm thứ hai.

 **Tính bình phương khoảng cách**:

d1 = (x2 - x1) \* (x2 - x1) # Khoảng cách theo trục x

d2 = (y2 - y1) \* (y2 - y1) # Khoảng cách theo trục y

* d1 là bình phương khoảng cách giữa x1 và x2.
* d2 là bình phương khoảng cách giữa y1 và y2.

 **Tính khoảng cách**:

res = math.sqrt(d1 + d2)

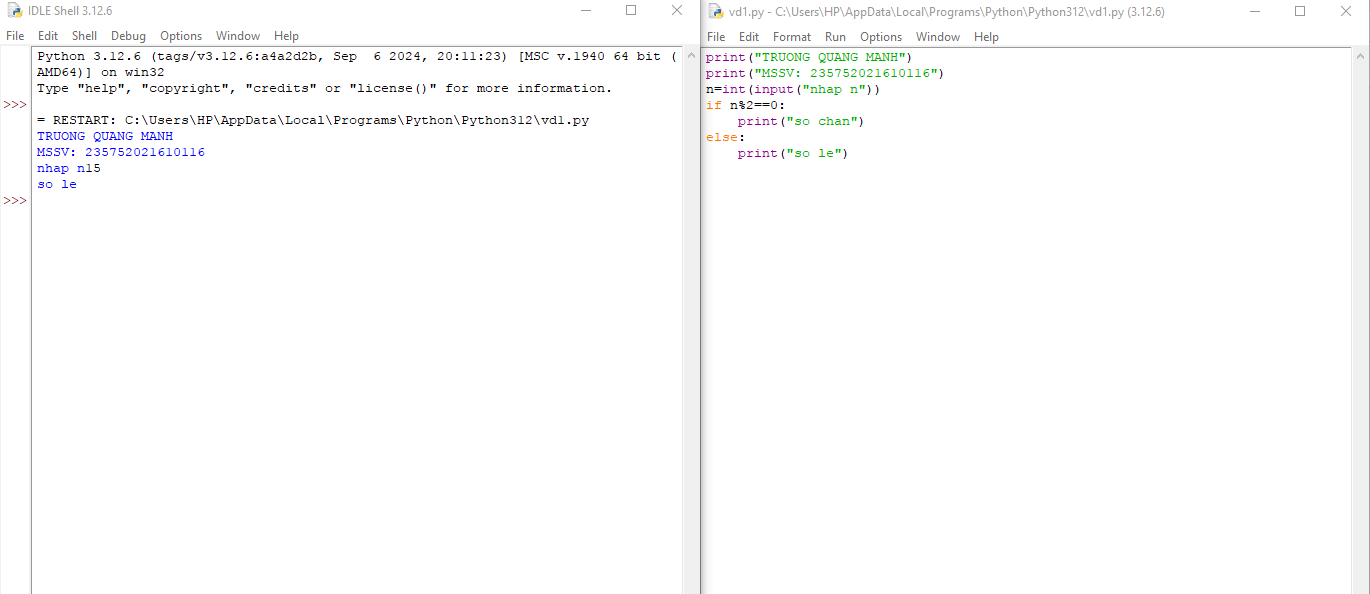
* Sử dụng hàm sqrt từ thư viện math để tính căn bậc hai của tổng bình phương khoảng cách d1 và d2, từ đó cho ra khoảng cách giữa hai điểm.

 **In kết quả**:

print("Khoảng cách giữa hai điểm:", res)

* In ra khoảng cách tính được.

**Bài 3. Viết chương trình nhặp vào một số và kiểm tra số đó chẵn hay lẻ**

****

Các bước thực hiện**:**

1. **Nhập dữ liệu**:

n = int(input("Nhập n: "))

* + Sử dụng hàm input() để nhận giá trị n từ người dùng. Hàm int() được dùng để chuyển đổi giá trị nhập vào thành số nguyên.

1. **Kiểm tra số chẵn hay lẻ**:

if n % 2 == 0:

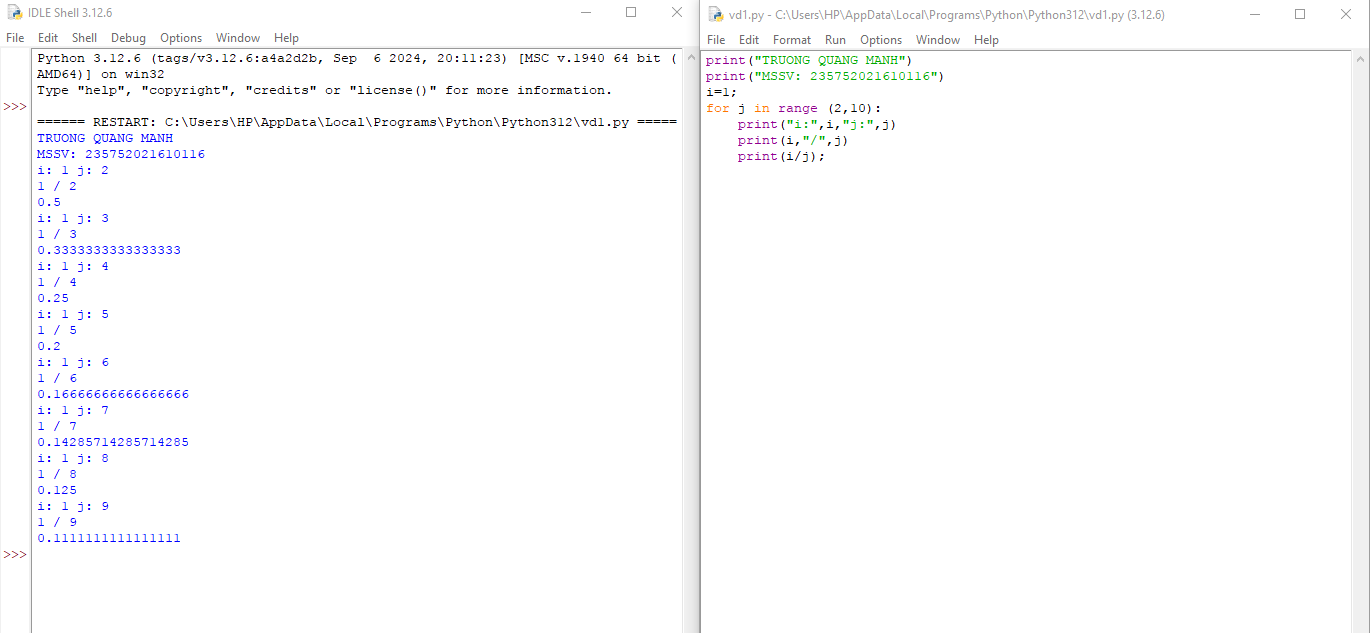
print("Số chẵn")

else:

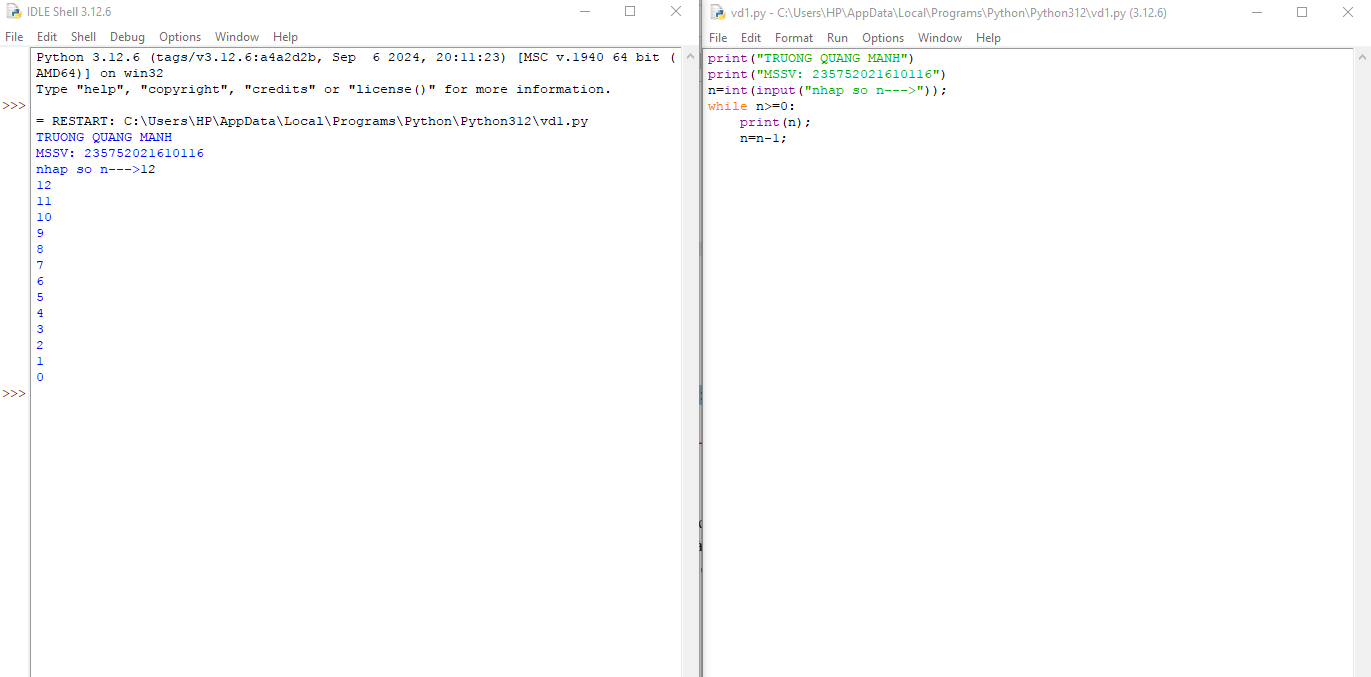
print("Số lẻ")

* + Sử dụng toán tử % (modulus) để kiểm tra phần dư khi chia n cho 2.
  + Nếu phần dư bằng 0, có nghĩa là n là số chẵn, và chương trình in ra "Số chẵn".
  + Nếu không, n là số lẻ, và chương trình in ra "Số lẻ".

**Bài 4. Viết chương trình in ra màn hình so nghịch đảo và in ra mành hình dưới dạng thập phân của một dãy số tự nhiên trong khoảng (a,b)**

****

**Bài 5. Viết chương trình nhập vào một số tự nhiên n > 0, in ra màn hinh các số tự nhiên giảm dần từ n đến 0, mỗi ký tự in trên 1 hàng**

****

**Các bước thực hiện:**

 **Nhập dữ liệu**:

n = int(input("Nhập số n: "))

* Dùng input() để nhận giá trị n từ người dùng và chuyển đổi thành số nguyên.

 **Vòng lặp while**:

while n >= 0:

* Vòng lặp sẽ tiếp tục chạy khi n lớn hơn hoặc bằng 0.

 **In giá trị n**:

print(n)

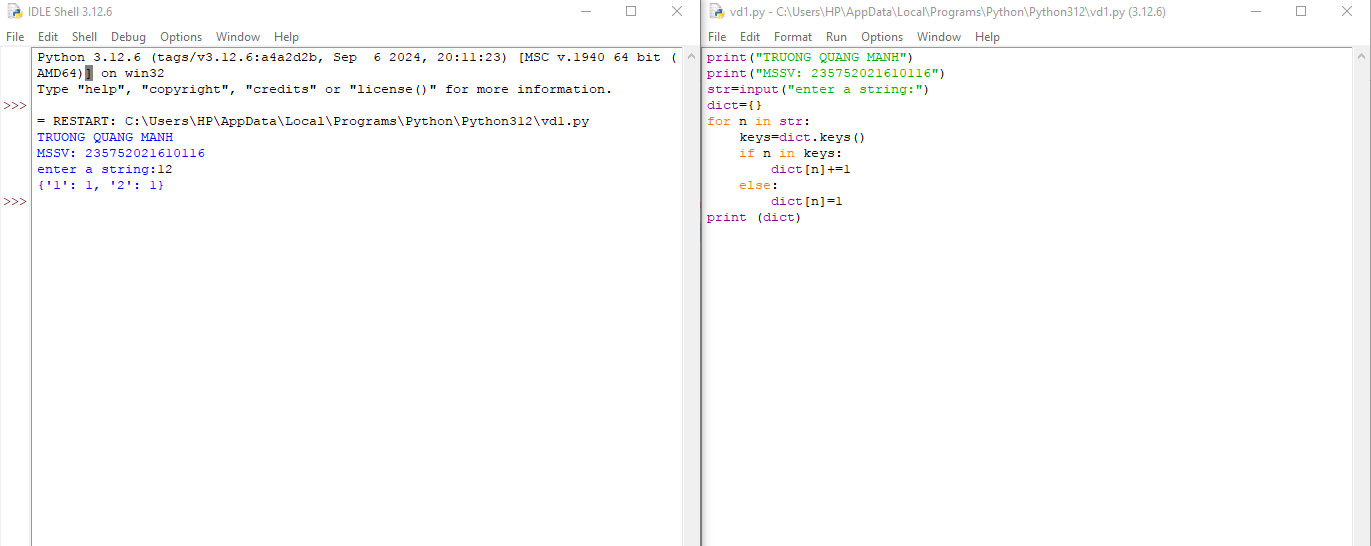
* In ra giá trị hiện tại của n.

 **Giảm giá trị của n**:

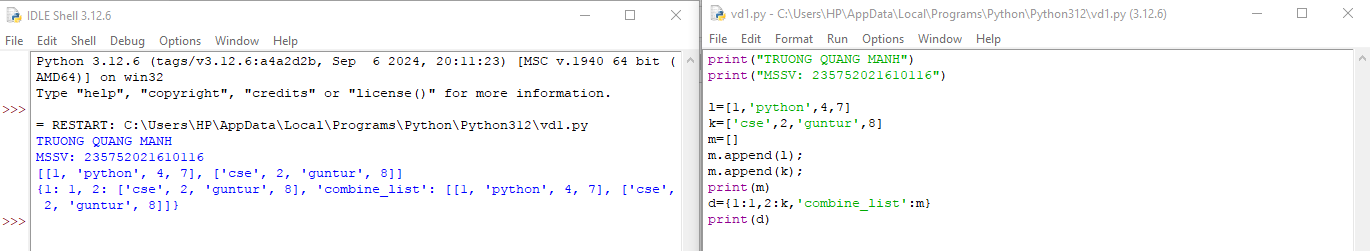
n=n-1

Câu lệnh này sẽ giảm giá trị của n đi 1 sau mỗi lần lặp

**Bài 6. Viết chương trình đếm số ký tự trong 1 xâu ký tự nhập vào từ bàn phim, lưu các ký tự vào cấu trúc từ điển**

****

**Bài 7. Viết chương trinh kết nối cac danh sach vào từ điển**



Các bước thực hiện:

 **Định nghĩa danh sách one**:

one = [1, 'python', 4, 7]

* Đây là một danh sách chứa các phần tử, bao gồm một số nguyên, một chuỗi, và hai số nguyên.

 **Định nghĩa danh sách k**:

k = ['cse', 2, 'guntur', 8]

* Danh sách này chứa một chuỗi, một số nguyên, một chuỗi khác, và một số nguyên.

 **Tạo danh sách rỗng m**:

m = []

 **Thêm phần tử vào danh sách m**:

m.append(1)

* Thêm số 1 vào danh sách m.

 **Thêm danh sách k vào danh sách m**:

m.append(k)

* Thêm danh sách k vào danh sách m. Sau bước này, danh sách m sẽ trở thành [1, ['cse', 2, 'guntur', 8]].

 **In danh sách m**:

print(m)

* Kết quả sẽ là: [1, ['cse', 2, 'guntur', 8]].

 **Định nghĩa từ điển d**:

d = {1: 1, 2: k, 'combine\_list': m}

* Tạo một từ điển d với các khóa là 1, 2, và 'combine\_list', trong đó:
  + Khóa 1 có giá trị là 1.
  + Khóa 2 có giá trị là danh sách k.
  + Khóa 'combine\_list' có giá trị là danh sách m.

 **In từ điển d**:

print(d)

* Kết quả sẽ là:

{1: 1, 2: ['cse', 2, 'guntur', 8], 'combine\_list': [1, ['cse', 2, 'guntur', 8]]}

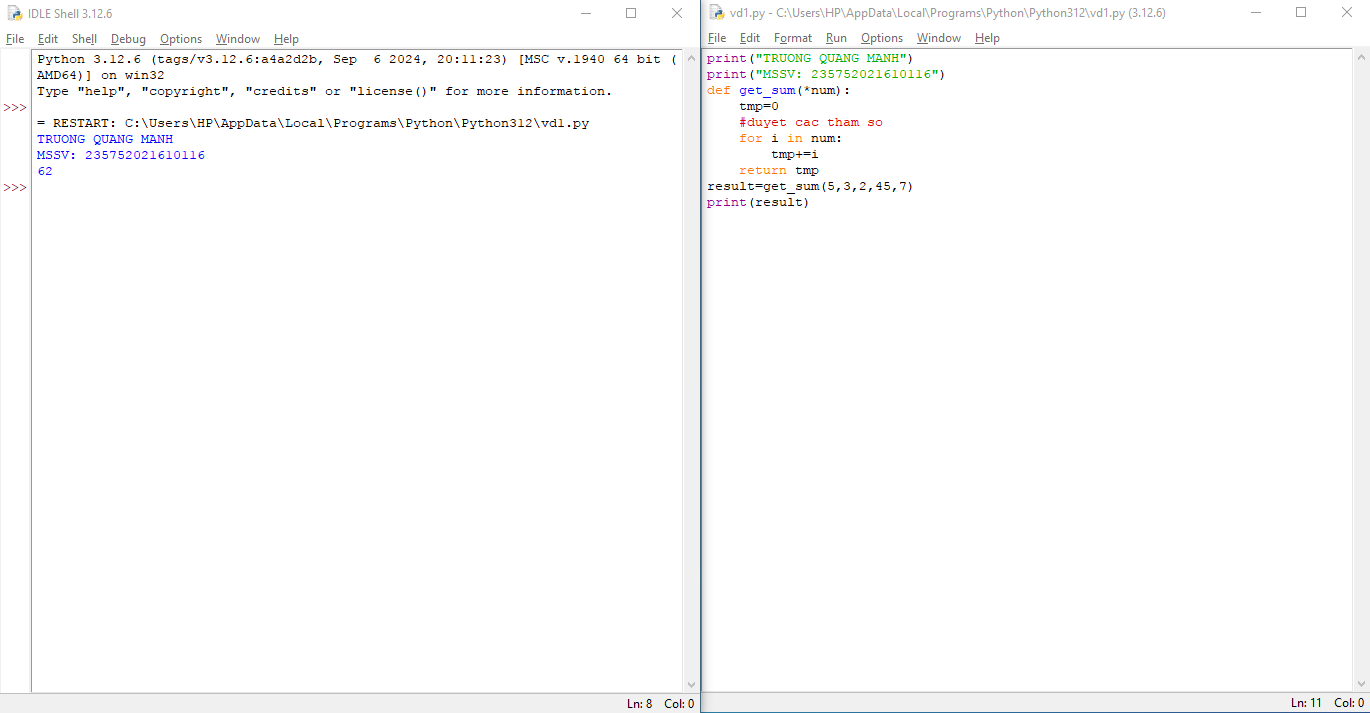
**Bài 3. Lập trinh hàm trong Python**

* 1. **Mục đích**

**Giúp sinh viên nắm bắt các kiến thức trong lập trình Python và sử dụng hàm**

* 1. **Các bước thực hiện và kết quả**

**Bài 1. Viết chương trình và giải thích việc truyền tham số của hàm**

****

**Các bước thực hiện:**

### 1. Định nghĩa hàm get\_sum(\*num):

* def get\_sum(\*num):: Dấu \* cho phép hàm nhận một số lượng tham số không xác định. Tất cả các tham số này sẽ được lưu trong một tuple có tên là num.

### 2. Khởi tạo biến tạm tmp:

* tmp = 0: Biến tmp được khởi tạo với giá trị 0. Đây sẽ là biến lưu tổng của các số.

### 3. Vòng lặp qua các tham số:

* for i in num:: Vòng lặp này duyệt qua từng phần tử trong tuple num. Mỗi phần tử sẽ được gán cho biến i trong mỗi lần lặp.
  + tmp += i: Câu lệnh này cộng giá trị của i vào tmp. Sau mỗi vòng lặp, tmp sẽ chứa tổng của tất cả các số đã được duyệt đến thời điểm đó.

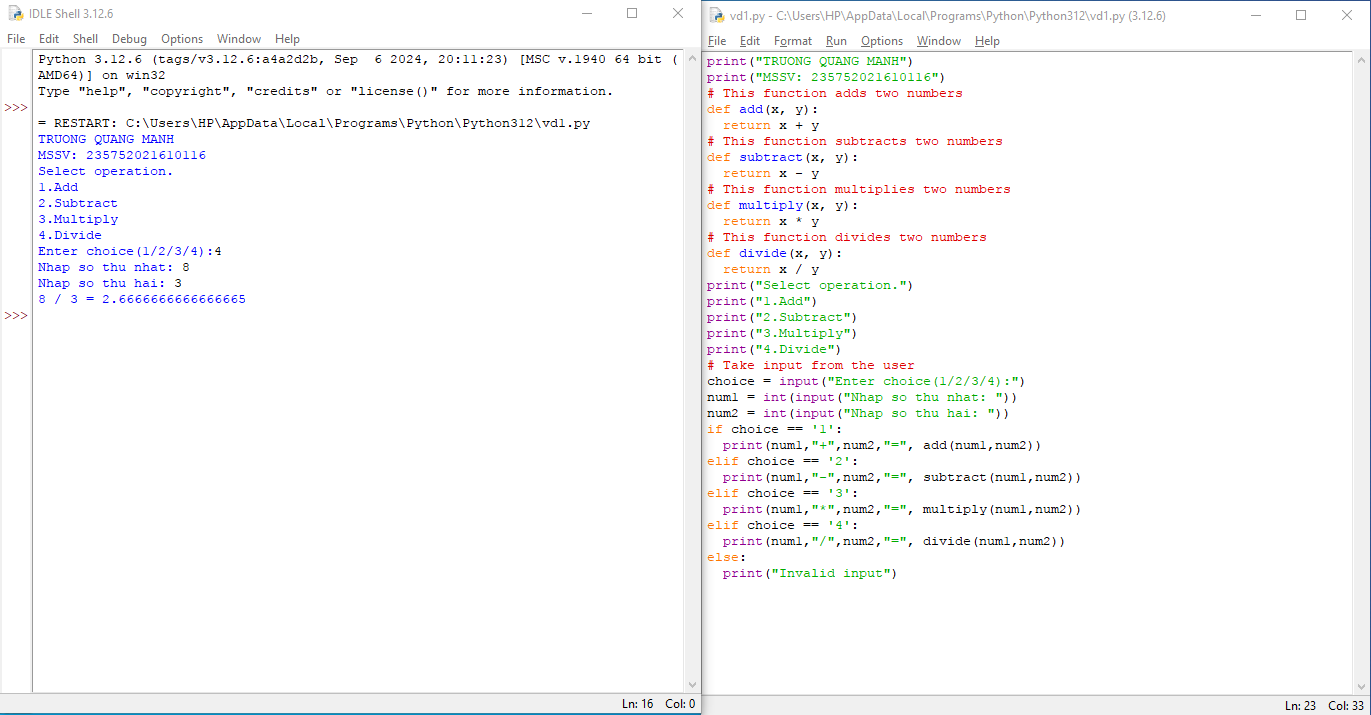
### 4. Trả về kết quả:

* return tmp: Sau khi vòng lặp kết thúc, hàm sẽ trả về giá trị của tmp, tức là tổng của tất cả các số mà bạn đã truyền vào.

### 5. Gọi hàm và in kết quả:

* result = get\_sum(5, 3, 2, 45, 7): Gọi hàm get\_sum với các tham số 5, 3, 2,45 và 7. Kết quả của phép tính tổng sẽ được lưu vào biến result.
* print(result): In ra kết quả của phép tính tổng. Trong trường hợp này, kết quả sẽ là 62.

**Bài 2. Chương trình thực hiện các phép tính đơn giản**

****

**Các bước thực hiện:**

### 1. Định nghĩa các hàm toán học:

* **Hàm add(x, y)**:
  + Được định nghĩa để nhận hai số và trả về tổng của chúng.
  + Cú pháp: return x + y.
* **Hàm subtract(x, y)**:
  + Nhận hai số và trả về hiệu của chúng.
  + Cú pháp: return x - y.
* **Hàm multiply(x, y)**:
  + Nhận hai số và trả về tích của chúng.
  + Cú pháp: return x \* y.
* **Hàm divide(x, y)**:
  + Nhận hai số và trả về thương của chúng.
  + Cú pháp: return x / y.

### 2. Giao diện người dùng:

* Chương trình hiển thị các tùy chọn phép toán cho người dùng:

print("Select operation.")

print("1.Add")

print("2.Subtract")

print("3.Multiply")

print("4.Divide")

### 3. Nhập dữ liệu từ người dùng:

* Chương trình yêu cầu người dùng nhập lựa chọn phép toán và hai số:

choice = input("Enter choice(1/2/3/4):")

num1 = int(input("Enter first number: "))

num2 = int(input("Enter second number: "))

* input được sử dụng để nhận dữ liệu từ người dùng, và int chuyển đổi dữ liệu nhập thành kiểu số nguyên.

### 4. Xử lý lựa chọn và thực hiện phép toán:

* Sử dụng cấu trúc điều kiện if-elif-else để thực hiện phép toán dựa trên lựa chọn của người dùng:
  + Nếu lựa chọn là '1', chương trình gọi hàm add và in ra kết quả:

if choice == '1':

print(num1,"+",num2,"=", add(num1,num2))

* + Tương tự cho các phép toán trừ, nhân và chia:

elif choice == '2':

print(num1,"-",num2,"=", subtract(num1,num2))

elif choice == '3':

print(num1,"\*",num2,"=", multiply(num1,num2))

elif choice == '4':

print(num1,"/",num2,"=", divide(num1,num2))

* Nếu lựa chọn không hợp lệ, chương trình in ra thông báo "Invalid input":

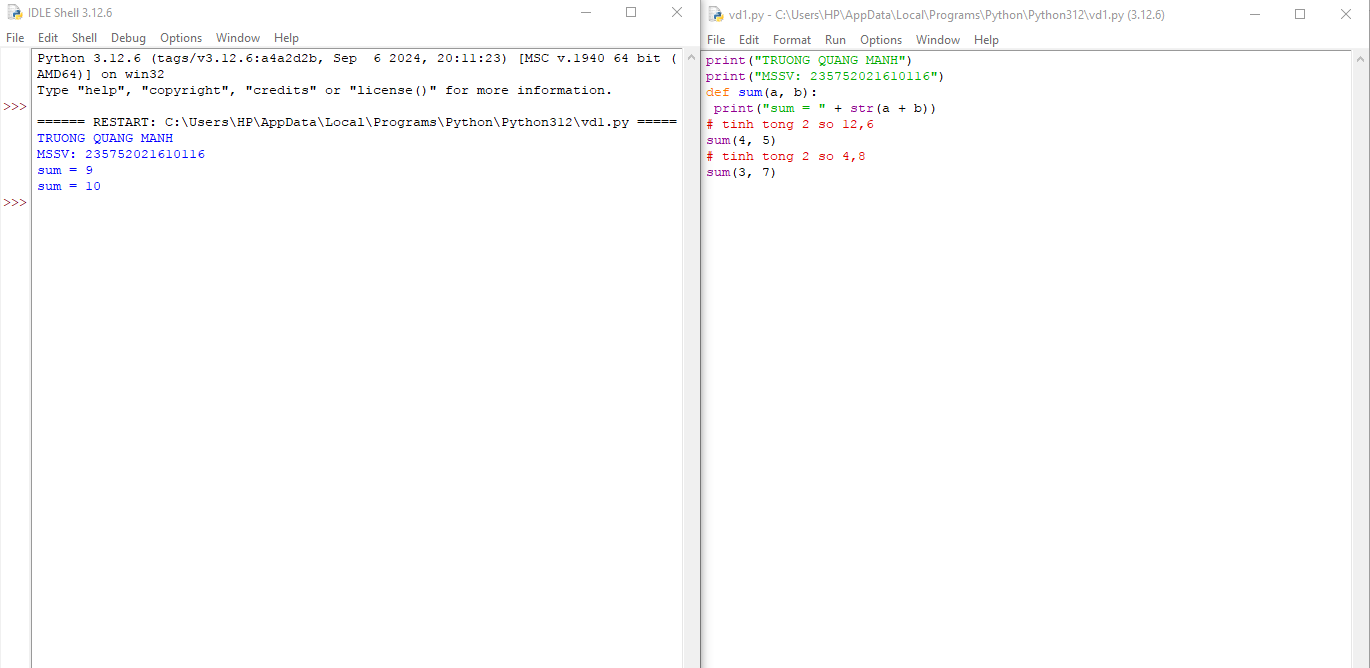
else:

print("Invalid input")

### 5. Kết thúc:

* Chương trình thực hiện phép toán dựa trên lựa chọn của người dùng và in ra kết quả tương ứng

**Bài 3. Viết hàm sum() tính tông hai số**

****

**Các bước thực hiện**

**1. Định nghĩa hàm sum(a, b)**:

-def sum(a, b): là cách bạn định nghĩa một hàm trong Python. Hàm này nhận vào hai tham số, a và b.

-return a + b là câu lệnh trả về tổng của hai tham số a và b. Khi gọi hàm, kết quả sẽ được trả về thay vì chỉ in ra màn hình.

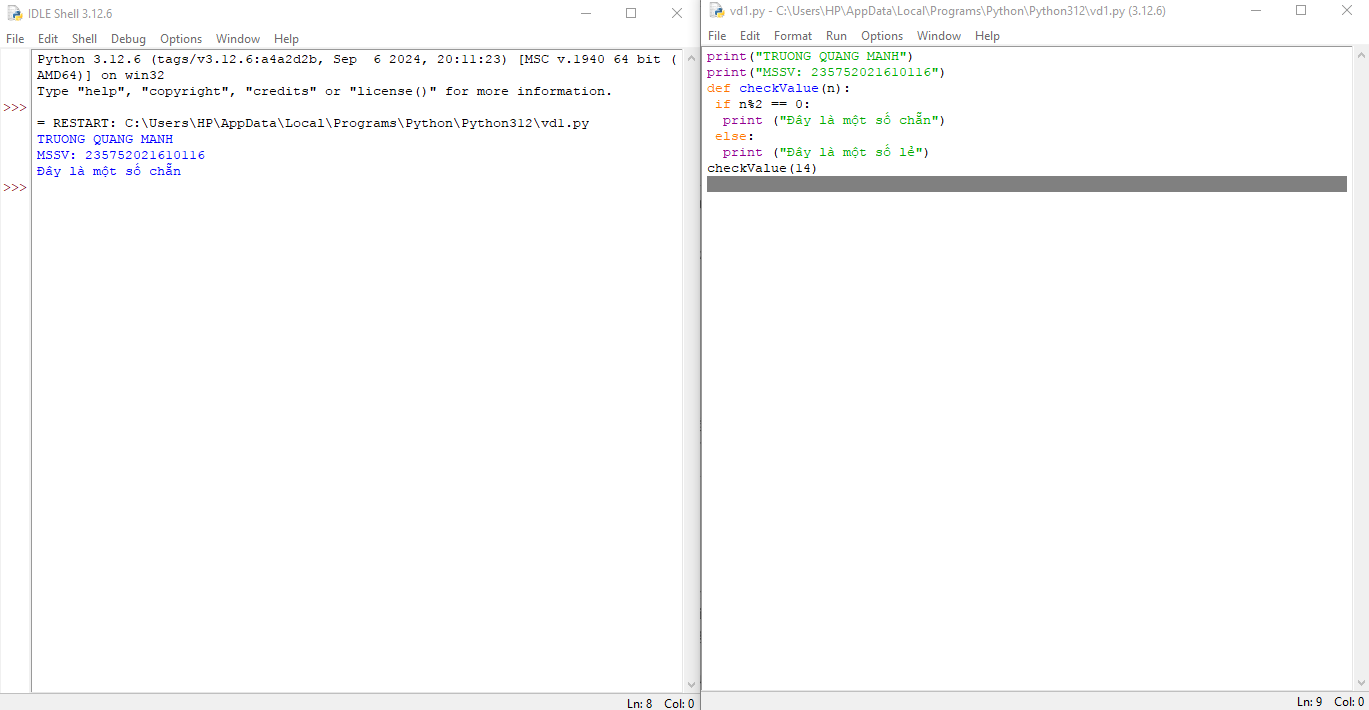
**2. Gọi hàm với các tham số**:

* + result1 = sum(4, 5) gọi hàm sum với các tham số 4 và 5. Hàm sẽ tính tổng là 9 và lưu vào biến result1.
  + print("Sum of 4 and 5 =", result1) in ra kết quả của phép tính, cụ thể là "Sum of 4 and 5 = 9".

**3. Lặp lại cho cặp số khác**:

* + Tương tự, result2 = sum(3, 7) gọi hàm sum với 3 và 7, và lưu kết quả là 10 vào result2.
  + print("Sum of 3 and 7 =", result2) in ra kết quả, tức là "Sum of 3 and 7 = 10".

**Bài 4. Kiểm tra số chẵn hay số lẻ**

****

**Các bước thực hiện:**

**1.** **Định nghĩa hàm checkValue(n)**:

* def checkValue(n): là cách bạn định nghĩa một hàm trong Python. Hàm này nhận một tham số n.

**2.** **Kiểm tra số chẵn hay lẻ**:

* if n % 2 == 0:: Đây là câu lệnh điều kiện. Hàm % (modulus) được sử dụng để tìm phần dư của phép chia n cho 2. Nếu phần dư bằng 0, tức là số n là số chẵn.
* Nếu điều kiện đúng, lệnh print("Đây là một số chẵn") sẽ được thực hiện.

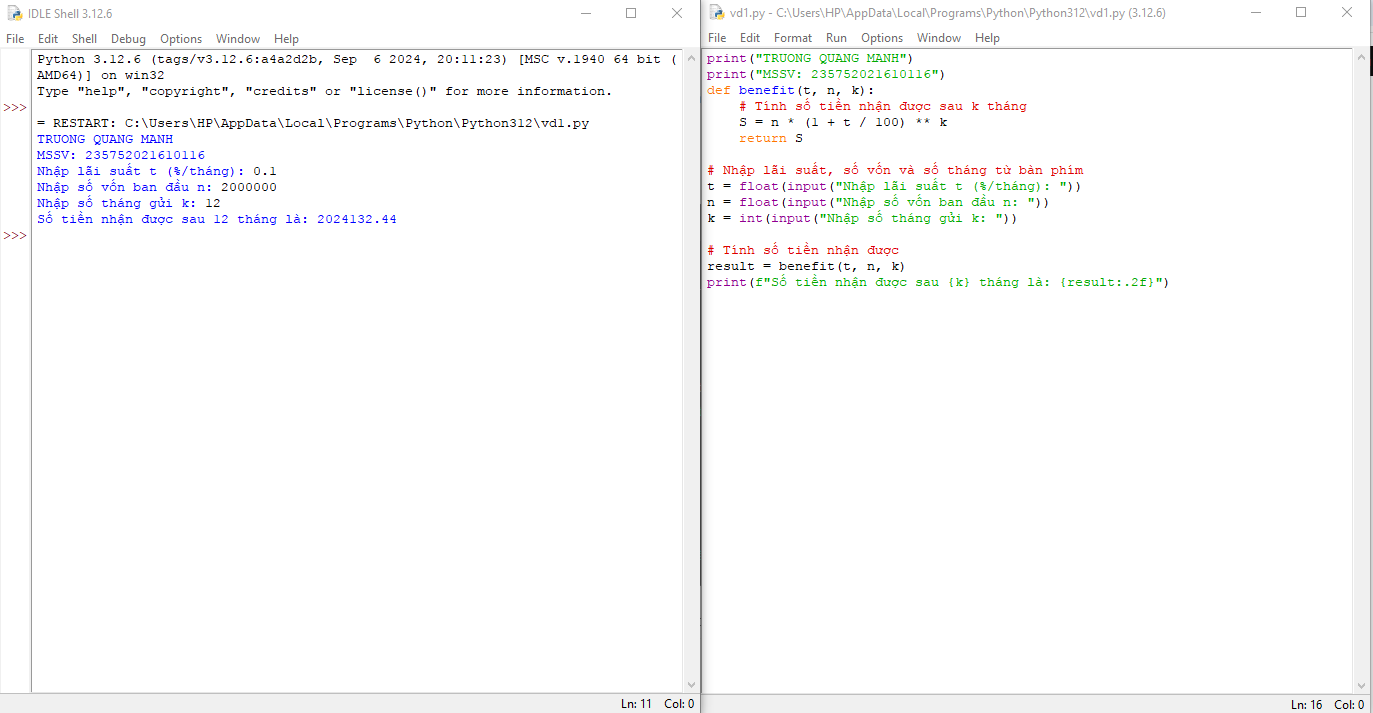
**3.** **Thực hiện nếu là số lẻ**:

* else:: Nếu điều kiện trên không đúng (tức là n là số lẻ), lệnh print("Đây là một số lẻ") sẽ được thực hiện.

**4.** **Gọi hàm**:

* checkValue(24): Gọi hàm checkValue với tham số là 24. Trong trường hợp này, 24 là số chẵn, nên kết quả in ra sẽ là "Đây là một số chẵn".

**Bài 5. Biết lãi suất tiết kiệm là t%/tháng (nhập t từ bàn phím). Nhập số vốn ban đầu n và số tháng gửi k. Tính số tiền nhận được sau k tháng sử dụng cấu trúc hàm def benefit(t,n,k):**

****

**Các bước thực hiện:**

1. **Định nghĩa hàm benefit(t, n, k)**:
   * Hàm này nhận ba tham số:
     + t: lãi suất theo phần trăm mỗi tháng.
     + n: số vốn ban đầu.
     + k: số tháng gửi.
   * Bên trong hàm, công thức tính tổng số tiền sau k tháng được thực hiện và kết quả được trả về.
2. **Nhập dữ liệu từ người dùng**:
   * t = float(input("Nhập lãi suất t (%/tháng): ")): Nhập lãi suất và chuyển đổi sang số thực.
   * n = float(input("Nhập số vốn ban đầu n: ")): Nhập số vốn ban đầu và chuyển đổi sang số thực.
   * k = int(input("Nhập số tháng gửi k: ")): Nhập số tháng gửi và chuyển đổi sang số nguyên.
3. **Tính số tiền nhận được**:
   * result = benefit(t, n, k): Gọi hàm benefit với các tham số đã nhập và lưu kết quả vào biến result.
4. **In kết quả**:
   * Sử dụng f-string để in ra số tiền nhận được sau k tháng với định dạng có 2 chữ số thập phân: print(f"Số tiền nhận được sau {k} tháng là: {result:.2f}").

**Bài 4. Các kiểu dữ liệu có cấu trúc trong Python**

* 1. **Mục đích**

Giúp sinh viên nắm bắt các kiểu dữ liệu có cấu trúc trong lập trình python.

* 1. **Cơ sở lý thuyết**

Xem các quy tắc sử dụng các kiểu dữ liệu: chuỗi, số, list, tuple, set và dictionary trong python.

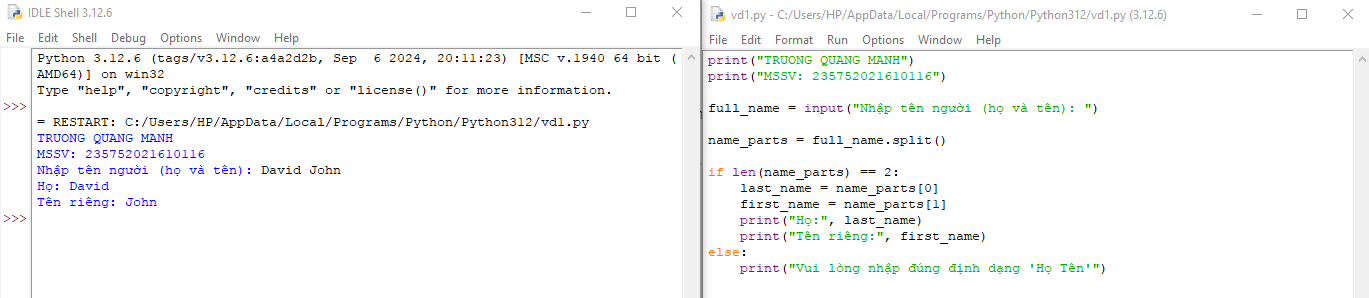
* 1. **Thiết bị thực hành thí nghiệm và vật tư tiêu hao**

Thiết bị thực hành thí nghiệm: Máy tính

Vật tư tiêu hao: Không

* 1. **Các bước tiến hành**

**Bài 1. Nhập một tên người từ bàn phím, hãy tách phần họ và tên riêng của người đó và in chúng ra màn hình (giả thiết họ và tên riêng chỉ gồm một âm).**



**Giải thích:**

**1.**  **Nhập tên người**: Dùng hàm input() để yêu cầu người dùng nhập tên. Tên này sẽ được lưu vào biến full\_name.

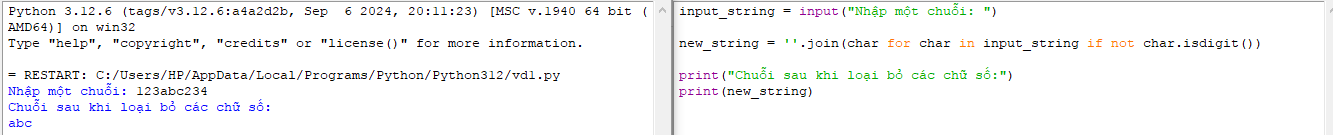
 **Tách tên**: Sử dụng phương thức split() để tách chuỗi full\_name thành một danh sách các phần (các từ), dựa trên dấu cách. Ví dụ, nếu người dùng nhập "Nguyễn Văn", danh sách parts sẽ có giá trị ["Nguyễn", "Văn"].

**2.** **Xác định họ và tên riêng**:

* Họ được lấy từ phần đầu tiên của danh sách parts (tức là parts[0]).
* Tên riêng được lấy từ phần thứ hai của danh sách parts (tức là parts[1]).

3. **In kết quả**: Sử dụng hàm print() để in ra họ và tên riêng.

**Bài 2. Nhập một chuỗi từ bàn phím, hãy loại bỏ tất cả các chữ số khỏi chuỗi và in lại nội dung chuỗi mới ra màn hình.**

****

**1.** **Nhập chuỗi**: Dùng input() để lấy chuỗi từ người dùng và lưu vào biến input\_string.

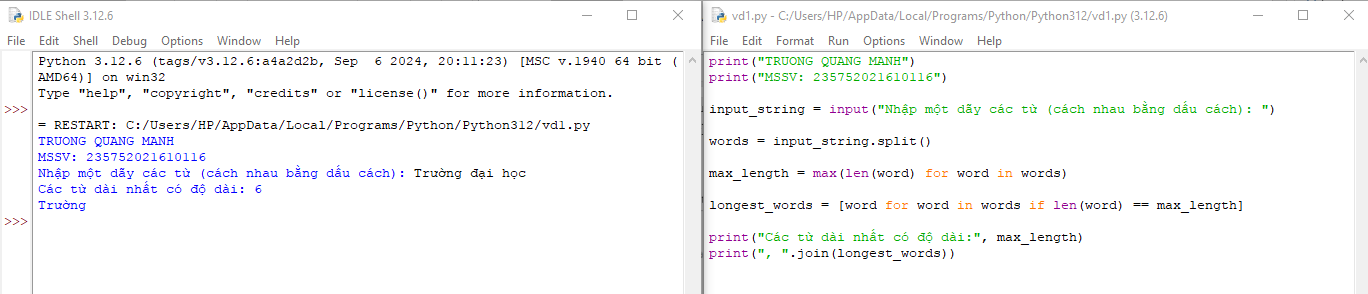
**2.** **Loại bỏ chữ số**:

* Sử dụng một biểu thức tạo danh sách (list comprehension) để tạo một chuỗi mới.
* char for char in input\_string if not char.isdigit() sẽ duyệt qua từng ký tự trong input\_string và chỉ giữ lại những ký tự không phải là chữ số.

**3.** **Kết hợp lại thành chuỗi**: Hàm join() sẽ kết hợp các ký tự lại thành một chuỗi mới mà không có chữ số.

**4.** **In kết quả**: Cuối cùng, in chuỗi mới ra màn hình.

**Bài 3. Nhập một dãy các từ từ bàn phím, hãy in ra từ dài nhất trong dãy vừa nhập, in ra mọi từ có cùng độ dài nhất.**

****

**Giải thích:**

**1.** **Nhập dãy từ**: Dùng input() để yêu cầu người dùng nhập một chuỗi các từ và lưu vào biến input\_string.

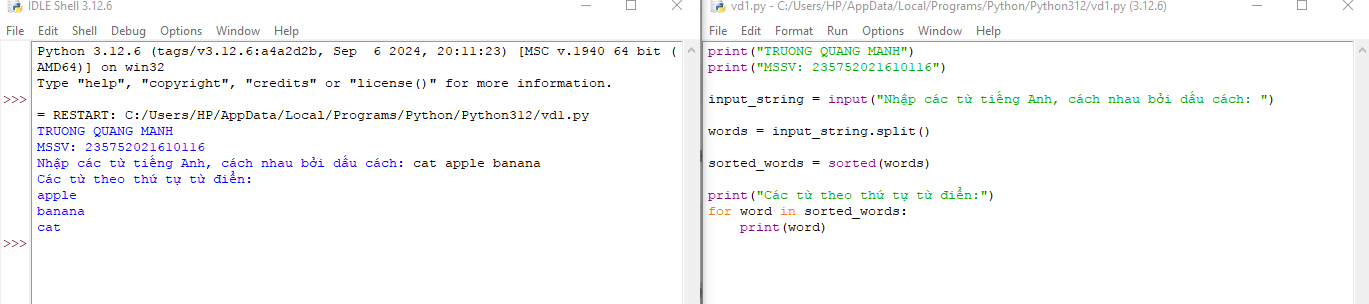
**2.** **Tách chuỗi thành danh sách từ**: Sử dụng phương thức split() để chia chuỗi thành danh sách các từ, lưu vào biến words.

**3.** **Tìm độ dài tối đa**: Sử dụng hàm max() kết hợp với một biểu thức tạo danh sách để tìm độ dài của từ dài nhất trong danh sách.

**4.** **Lấy các từ dài nhất**: Duyệt qua danh sách words và lưu tất cả các từ có độ dài bằng max\_length vào danh sách longest\_words.

**5.** **In kết quả**: Sử dụng join() để kết hợp các từ dài nhất thành một chuỗi, sau đó in ra.

**Bài 4. Người dùng nhập từ bàn phím liên tiếp các từ tiếng Anh viết tách nhau bởi dấu cách. Hãy nhập chuỗi đầu vào và tách thành các từ sau đó in ra màn hình các từ đó theo thứ tự từ điển.**

****

**1.** **Nhập chuỗi từ bàn phím**:

input\_string = input("Nhập chuỗi các từ tiếng Anh: ")

**2.** **Tách chuỗi thành danh sách các từ**:

words = input\_string.split()

**3.** **Sắp xếp danh sách các từ theo thứ tự từ điển**:

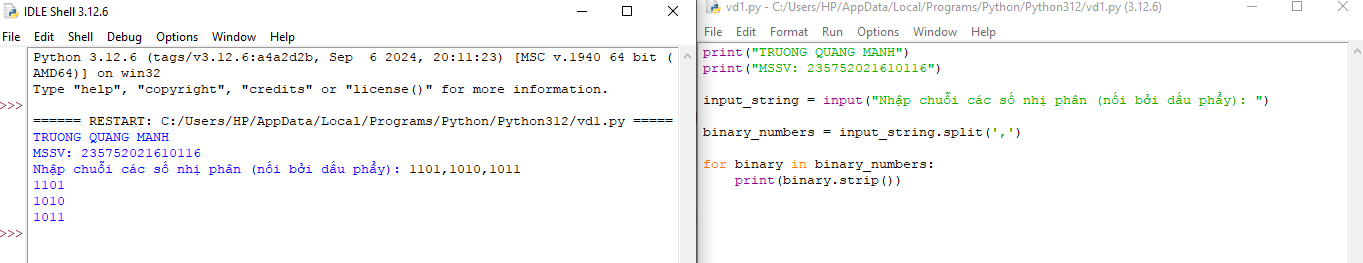
sorted\_words = sorted(words)

**4.** **In ra các từ đã được sắp xếp**:

for word in sorted\_words:

print(word)

**Bài 5. Người dùng nhập từ bàn phím chuỗi các số nhị phân viết liên tiếp được nối nhau bởi dấu phẩy. Hãy nhập chuỗi đầu vào sau đó in ra những giá trị được nhập.**

****

**Giải thích:**

**1. Nhập chuỗi từ bàn phím**

* **Hàm input()**: Dùng để nhận dữ liệu từ người dùng. Người dùng sẽ nhập một chuỗi các số nhị phân, nối với nhau bằng dấu phẩy.
* **Biến input\_string**: Lưu trữ chuỗi mà người dùng nhập vào.

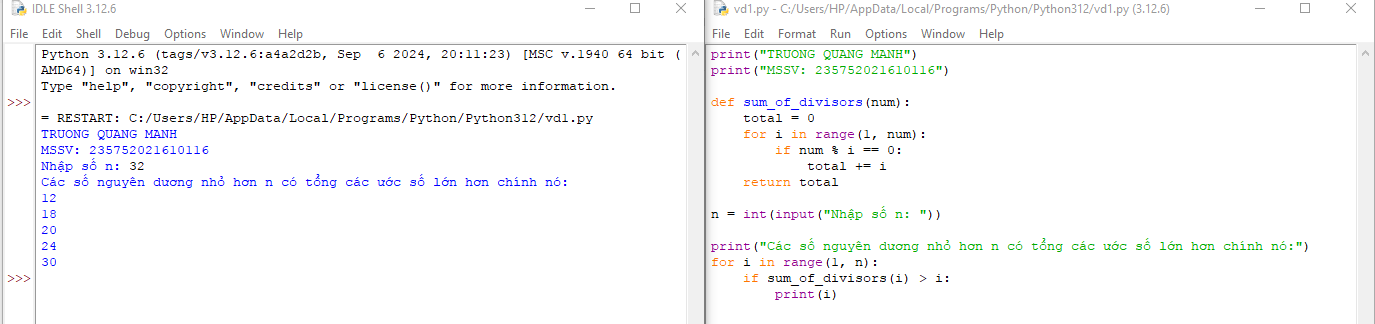
**2. Tách chuỗi thành danh sách**

* **Phương thức split(',')**: Tách chuỗi input\_string thành các phần tử trong danh sách dựa trên dấu phẩy. Mỗi số nhị phân sẽ trở thành một phần tử riêng trong danh sách.
* **Biến binary\_numbers**: Lưu trữ danh sách các số nhị phân.

**3. In ra các giá trị đã nhập**

* **Vòng lặp for**: Duyệt qua từng phần tử trong danh sách binary\_numbers.
* **print(binary.strip())**: In ra mỗi số nhị phân. Phương thức strip() loại bỏ khoảng trắng thừa ở đầu và cuối mỗi số, đảm bảo rằng kết quả in ra gọn gàng hơn.

**Bài 6. Nhập số n, in ra màn hình các số nguyên dương nhỏ hơn n có tổng các ước số lớn hơn chính nó.**

****

**Giải thích:**

**Hàm sum\_of\_divisors(num)**:

* Hàm này tính tổng các ước số của số num.
* Sử dụng vòng lặp để kiểm tra từng số từ 1 đến num - 1. Nếu số đó là ước số của num (tức là num % i == 0), nó sẽ được cộng vào tổng total.

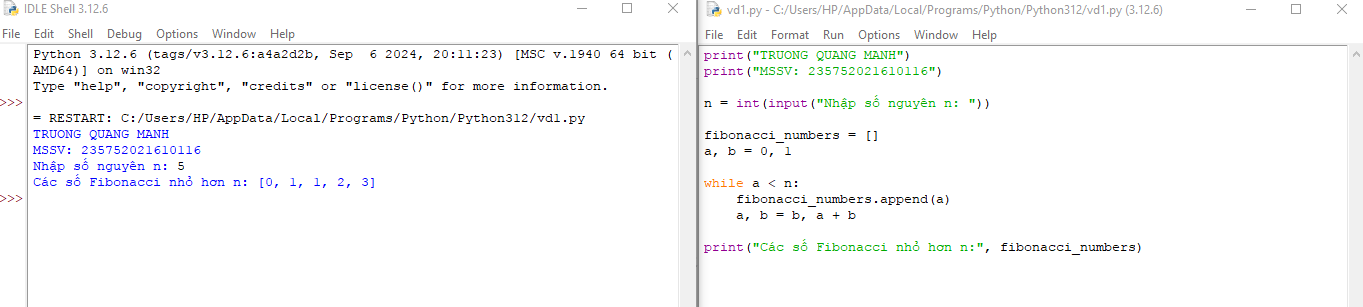
**Nhập số n**:

* Người dùng sẽ nhập vào một số nguyên n, và nó sẽ được chuyển đổi thành kiểu số nguyên.

**Vòng lặp để in ra các số**:

* Vòng lặp từ 1 đến n - 1 kiểm tra từng số nguyên dương i.
* Nếu tổng các ước số của i lớn hơn chính nó, số đó sẽ được in ra

**Bài 7. Hãy nhập số nguyên n, tạo một list gồm các số fibonacci nhỏ hơn n và in ra màn hình.**

**** **Nhập số nguyên n**:

Người dùng sẽ nhập vào một số nguyên nnn.

**Khởi tạo danh sách và các số Fibonacci**:

Tạo một danh sách rỗng fibonacci\_numbers để lưu các số Fibonacci.

Khởi tạo hai biến a và b với giá trị 0 và 1, là hai số Fibonacci đầu tiên.

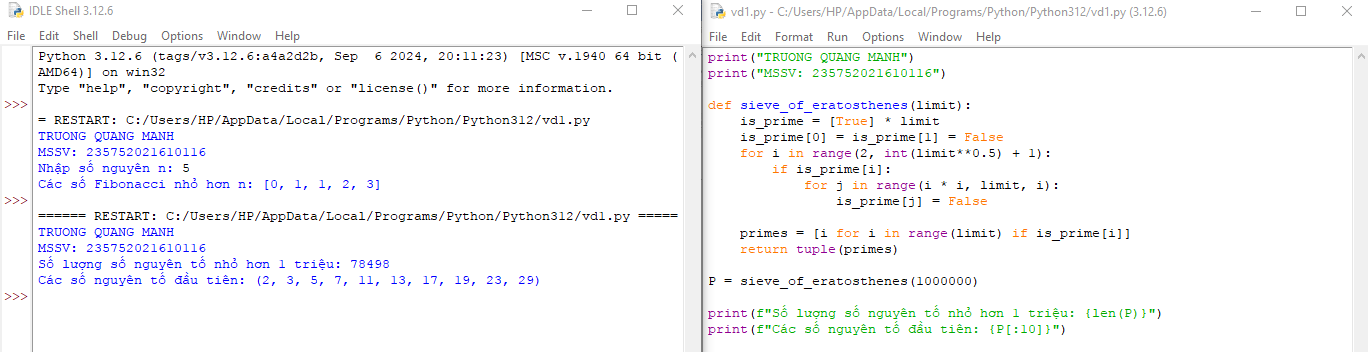
**Vòng lặp để tạo số Fibonacci**:

Vòng lặp while sẽ tiếp tục cho đến khi a (số Fibonacci hiện tại) lớn hơn hoặc bằng nnn.

Trong mỗi lần lặp, số Fibonacci a được thêm vào danh sách fibonacci\_numbers, sau đó cập nhật a và b để tính số Fibonacci tiếp theo.

**In ra danh sách**

**Bài 8. Hãy tạo ra tuple P gồm các số nguyên tố nhỏ hơn 1 triệu.**



**Giải thích:**

**1.** **Hàm sieve\_of\_eratosthenes(limit)**:

* Nhận một tham số limit, là giới hạn trên cho các số nguyên tố.
* Tạo một danh sách is\_prime với kích thước limit, khởi tạo tất cả các giá trị là True (mặc định cho là số nguyên tố).
* Đặt is\_prime[0] và is\_prime[1] là False, vì 0 và 1 không phải là số nguyên tố.

**2.** **Thuật toán Sieve of Eratosthenes**:

* Sử dụng vòng lặp từ 2 đến căn bậc hai của limit. Nếu i là số nguyên tố (tức là is\_prime[i] là True), thì đánh dấu tất cả các bội số của i (từ i2i^2i2 đến limit) là không phải là số nguyên tố.

**3.** **Tạo danh sách các số nguyên tố**:

* Sử dụng danh sách hiểu để tạo danh sách primes chứa tất cả các số có chỉ số là True trong is\_prime.

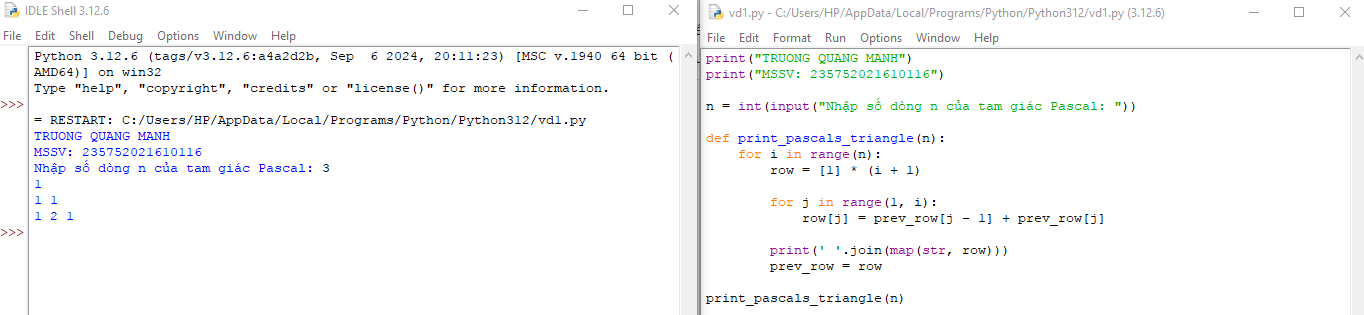
**4.** **Trả về tuple**:

* Chuyển đổi danh sách primes thành một tuple và trả về.

**5.** **In ra kết quả**:

* In ra số lượng số nguyên tố nhỏ hơn 1 triệu và một vài số nguyên tố đầu tiên.

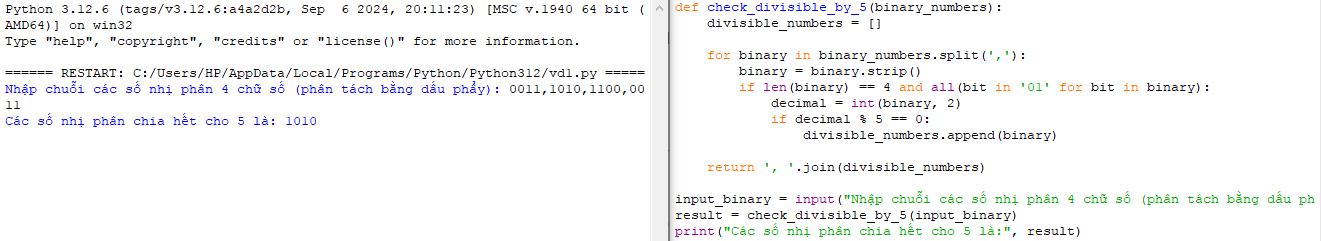
**Bài 9. Nhập n, in n dòng đầu tiên của tam giác pascal**



**Giải thích:**

* 1. **Nhập số dòng n**:
* Người dùng nhập vào số dòng mà họ muốn in ra.
  1. **Hàm print\_pascals\_triangle(n)**:
* Vòng lặp ngoài chạy từ 0 đến n−1n - 1n−1 để tạo từng dòng của tam giác Pascal.
* Dòng được khởi tạo với các giá trị bằng 1.
* Vòng lặp bên trong tính toán các giá trị giữa các giá trị 1 ở đầu và cuối dòng bằng cách cộng các giá trị từ dòng trước.
  1. **In dòng**:
* Mỗi dòng được in ra bằng cách sử dụng join() để nối các giá trị thành một chuỗi.

**Bài 10. Viết một chương trình chấp nhận đầu vào là chuỗi các số nhị phân 4 chữ số, phân tách bởi dấu phẩy, kiểm tra xem chúng có chia hết cho 5 không. Sau đó in các số chia hết cho 5 thành dãy phân tách bởi dấu phẩy.**

****

**Giải thích:**

**1. Định nghĩa hàm check\_divisible\_by\_5**

* Hàm này nhận đầu vào là chuỗi các số nhị phân và sẽ xử lý chúng để tìm ra các số chia hết cho 5.

**2. Khởi tạo danh sách để lưu các số chia hết cho 5**

* Tạo một danh sách rỗng divisible\_numbers để lưu các số nhị phân mà chúng ta tìm thấy là chia hết cho 5.

**3. Phân tách chuỗi đầu vào**

* Sử dụng phương thức split(',') để chia chuỗi đầu vào thành các phần tử riêng biệt, mỗi phần tử là một số nhị phân.

**4. Làm sạch và kiểm tra định dạng số nhị phân**

* strip() loại bỏ khoảng trắng thừa ở đầu và cuối chuỗi.
* len(binary) == 4 kiểm tra xem số nhị phân có đúng 4 chữ số hay không.
* all(bit in '01' for bit in binary) kiểm tra xem tất cả các ký tự trong chuỗi có phải là 0 hoặc 1 hay không, đảm bảo đây là số nhị phân hợp lệ.

**5. Chuyển đổi từ nhị phân sang thập phân**

* Sử dụng int(binary, 2) để chuyển đổi chuỗi nhị phân sang số thập phân. Tham số 2 cho biết rằng chuỗi đầu vào là hệ nhị phân.

**6. Kiểm tra chia hết cho 5**

* Kiểm tra xem số thập phân có chia hết cho 5 không bằng cách sử dụng toán tử chia lấy dư %.

**7. Lưu số nhị phân vào danh sách**

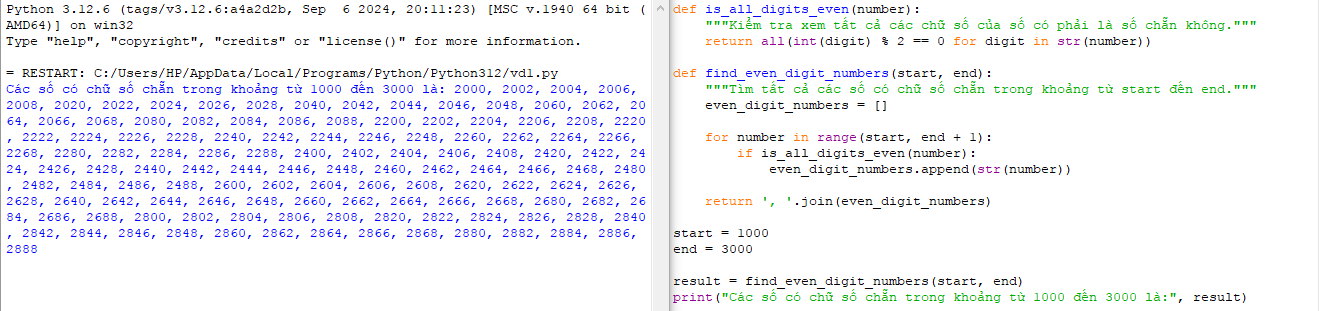
* Nếu số nhị phân chia hết cho 5, nó sẽ được thêm vào danh sách divisible\_numbers.

**8. In ra kết quả**

* Cuối cùng, sử dụng join() để nối các số nhị phân trong danh sách thành một chuỗi phân tách bằng dấu phẩy và trả về kết quả.

**9. Nhập đầu vào từ người dùng và hiển thị kết quả**

**Bài 11. Viết một chương trình tìm tất cả các số trong đoạn 1000 và 3000 (tính cả 2 số này) sao cho tất cả các chữ số trong số đó là số chẵn. In các số tìm được thành chuỗi cách nhau bởi dấu phẩy, trên một dòng**

****

**Giải thích:**

**1. Định nghĩa hàm is\_all\_digits\_even**

* Hàm này nhận một số nguyên number và kiểm tra xem tất cả các chữ số của nó có phải là số chẵn hay không.

**2. Kiểm tra các chữ số**

* **str(number)**: Chuyển số thành chuỗi để dễ dàng truy cập từng chữ số.
* **int(digit) % 2 == 0**: Kiểm tra xem chữ số hiện tại có phải là số chẵn hay không.
* **all(...)**: Nếu tất cả các chữ số đều là chẵn, hàm sẽ trả về True, ngược lại sẽ trả về False.

**3. Định nghĩa hàm find\_even\_digit\_numbers**

* Hàm này tìm tất cả các số trong khoảng từ start đến end mà tất cả các chữ số đều chẵn.

**4. Khởi tạo danh sách lưu số chẵn**

* Tạo danh sách rỗng even\_digit\_numbers để lưu các số thoả mãn điều kiện.

**5. Duyệt qua các số trong khoảng**

* Sử dụng vòng lặp for để duyệt qua từng số trong khoảng từ start đến end, bao gồm cả số end.

**6. Kiểm tra số chẵn**

* Gọi hàm is\_all\_digits\_even để kiểm tra xem số hiện tại có tất cả các chữ số chẵn hay không.
* Nếu điều kiện đúng, số đó sẽ được thêm vào danh sách even\_digit\_numbers.

**7. Trả về kết quả**

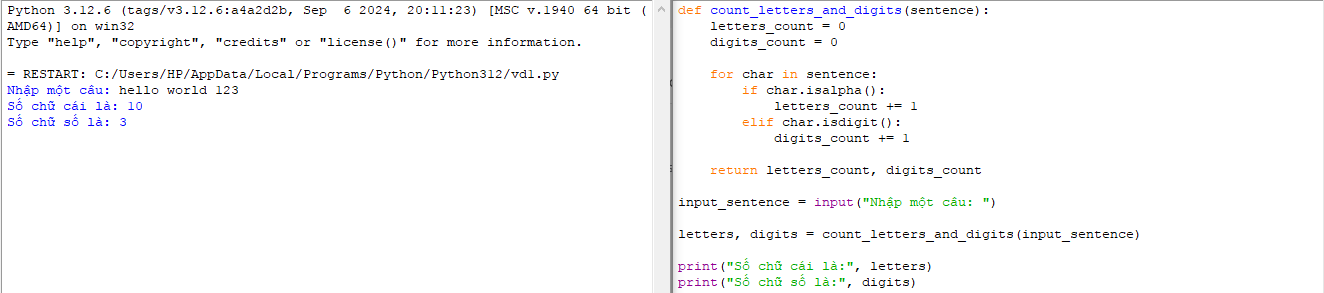
* Sử dụng join() để nối các số trong danh sách thành một chuỗi, các số được phân tách bởi dấu phẩy.

**8. Khởi tạo khoảng tìm kiếm**

* Đặt giá trị cho start là 1000 và end là 3000, xác định khoảng tìm kiếm cho chương trình.

**9. Gọi hàm và in kết quả**

**Bài 12. Viết một chương trình chấp nhận đầu vào là một câu, đếm số chữ cái và chữ số trong câu đó. Giả sử đầu vào sau được cấp cho chương trình: hello world! 123 Thì đầu ra sẽ là: Số chữ cái là: 10 Số chữ số là: 3**

****

**Giải thích:**

**1. Định nghĩa hàm count\_letters\_and\_digits**

* Hàm này nhận đầu vào là một chuỗi sentence (câu) và có nhiệm vụ đếm số chữ cái và chữ số trong chuỗi đó.

**2. Khởi tạo biến đếm**

* Tạo hai biến:
  + letters\_count: để đếm số chữ cái, khởi tạo bằng 0.
  + digits\_count: để đếm số chữ số, khởi tạo bằng 0.

**3. Duyệt qua từng ký tự trong câu**

* Sử dụng vòng lặp for để lặp qua từng ký tự (char) trong chuỗi sentence.

**4. Kiểm tra và đếm chữ cái**

* Sử dụng phương thức isalpha() để kiểm tra xem ký tự hiện tại có phải là chữ cái hay không.
* Nếu đúng, tăng letters\_count lên 1.

**5. Kiểm tra và đếm chữ số**

* Sử dụng phương thức isdigit() để kiểm tra xem ký tự hiện tại có phải là chữ số hay không.
* Nếu đúng, tăng digits\_count lên 1.

**6. Trả về kết quả**

* Sau khi duyệt qua toàn bộ ký tự trong câu, hàm sẽ trả về số lượng chữ cái và chữ số dưới dạng một tuple.

**7. Nhập câu từ người dùng**

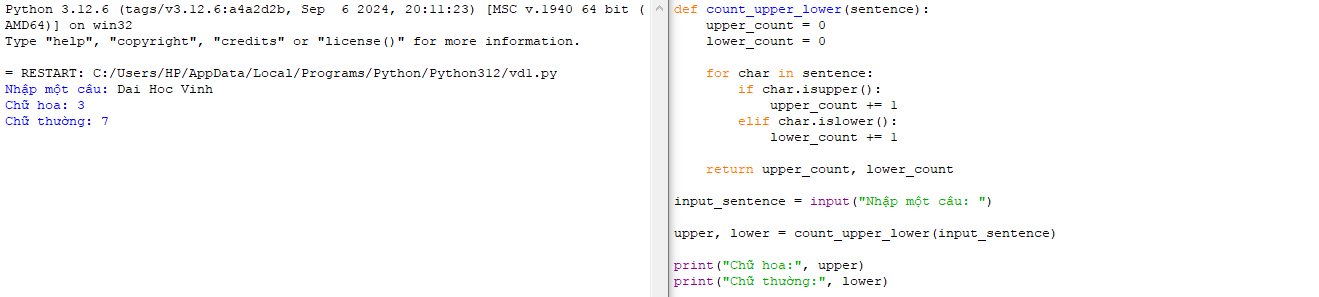
* Sử dụng input() để nhận một câu từ người dùng. Câu này sẽ được lưu vào biến input\_sentence.

**8. Gọi hàm và nhận kết quả**

* Gọi hàm count\_letters\_and\_digits với input\_sentence làm tham số.
* Kết quả trả về sẽ được gán cho hai biến: letters (số chữ cái) và digits (số chữ số).

**9. In kết quả**

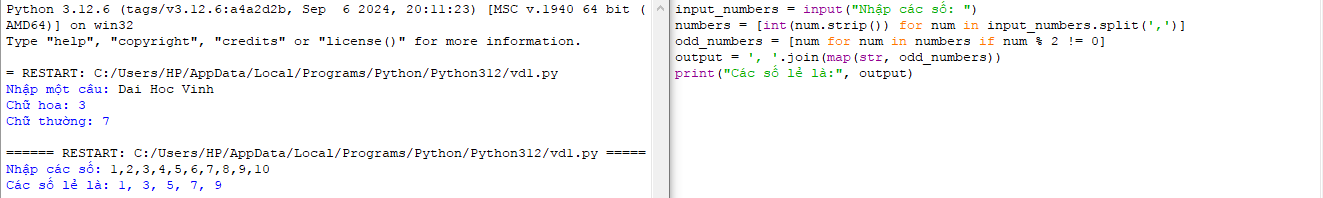
**Bài 13. Viết một chương trình chấp nhận đầu vào là một câu, đếm chữ hoa, chữ thường. Giả sử đầu vào là: Dai Hoc Vinh Thì đầu ra là: Chữ hoa: 3 Chữ thường: 7**

****

**Giải thích:**

* 1. **Định nghĩa hàm count\_upper\_lower**:
* Hàm này nhận đầu vào là một chuỗi sentence và sẽ đếm số lượng chữ hoa và chữ thường.
  1. **Khởi tạo biến đếm**:
* Tạo hai biến upper\_count và lower\_count để lưu số lượng chữ hoa và chữ thường, ban đầu khởi tạo bằng 0.
  1. **Duyệt qua từng ký tự trong câu**:
* Sử dụng vòng lặp for để kiểm tra từng ký tự (char) trong chuỗi sentence.
  1. **Kiểm tra và đếm chữ hoa**:
* Sử dụng phương thức isupper() để kiểm tra xem ký tự có phải là chữ hoa hay không. Nếu đúng, tăng upper\_count lên 1.
  1. **Kiểm tra và đếm chữ thường**:
* Sử dụng phương thức islower() để kiểm tra xem ký tự có phải là chữ thường hay không. Nếu đúng, tăng lower\_count lên 1.
  1. **Trả về kết quả**:
* Sau khi duyệt qua tất cả ký tự, hàm sẽ trả về số lượng chữ hoa và chữ thường.
  1. **Nhập câu từ người dùng**:
  2. **Gọi hàm và nhận kết quả**
  3. **In kết quả**:

**Bài 14. Sử dụng một danh sách để lọc các số lẻ từ danh sách được người dùng nhập vào. Giả sử đầu vào là: 1,2,3,4,5,6,7,8,9 thì đầu ra phải là: 1,3,5,7,9**

****

**Giải thích**

1. **Nhập chuỗi từ người dùng**:

* Nhận đầu vào từ người dùng dưới dạng chuỗi, yêu cầu người dùng nhập các số được phân tách bằng dấu phẩy.

1. **Chuyển đổi chuỗi thành danh sách các số nguyên**:

* Sử dụng split(',') để chia chuỗi thành các phần tử.
* Sử dụng strip() để loại bỏ khoảng trắng xung quanh mỗi số và int() để chuyển đổi chúng thành số nguyên.

1. **Lọc các số lẻ**:

* Sử dụng danh sách hiểu (list comprehension) để tạo một danh sách mới chỉ chứa các số lẻ. Điều kiện num % 2 != 0 kiểm tra xem số có phải là số lẻ không.

1. **Chuyển danh sách số lẻ thành chuỗi**:

* Sử dụng map(str, odd\_numbers) để chuyển mỗi số trong danh sách odd\_numbers thành chuỗi.
* Sử dụng join() để nối các chuỗi lại với nhau, cách nhau bởi dấu phẩy.

1. **In kết quả**:

**Bài 5. Thiết kế module trong Python**

**1.1. Mục đích**

Giúp sinh viên nắm bắt việc thiết kế module trong lập trình Python, sử dụng module thư viện numpy trong các ứng dụng

**1.2. Cơ sở lý thuyết**

Xem các quy tắc khai báo, thiết kế và sử dụng module trong python, các thuật toán tìm kiếm, sắp xếp, cài đặt và sử dụng thư viện numpy.

**1.3. Thiết bị thực hành và vật tư tiêu hao**

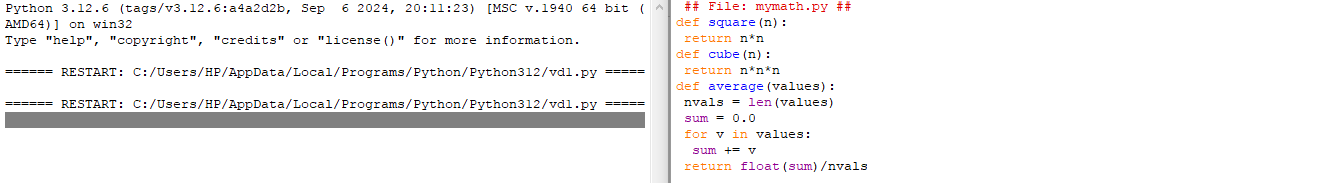
Thiết bị thực hành thí nghiệm: Máy tính

Vật tư tiêu hao: Không

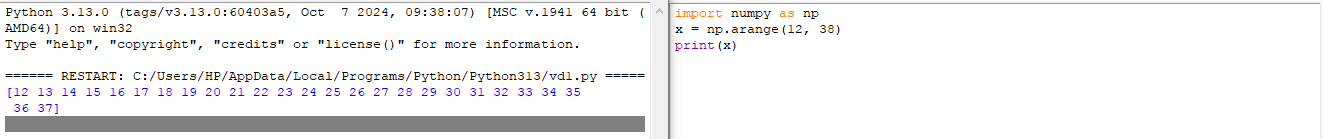
**1.4. Các bước tiến hành**

Thực hiện các bài tập sau sử dụng Python

**Bài 1. Sử dụng module. Định nghĩa một module toán học gọi là mymath và sử dụng module này từ một tập lệnh riêng biệt.**

****

**Bài 2. Viết chương trình sử dụng thư viện NumPy để tạo một mảng với các giá trị nằm trong khoảng từ 12 đến 38**

****

**Giải thích**

1. **Nhập thư viện NumPy**:

* import numpy as np: Nhập thư viện NumPy và gán cho nó tên viết tắt np.

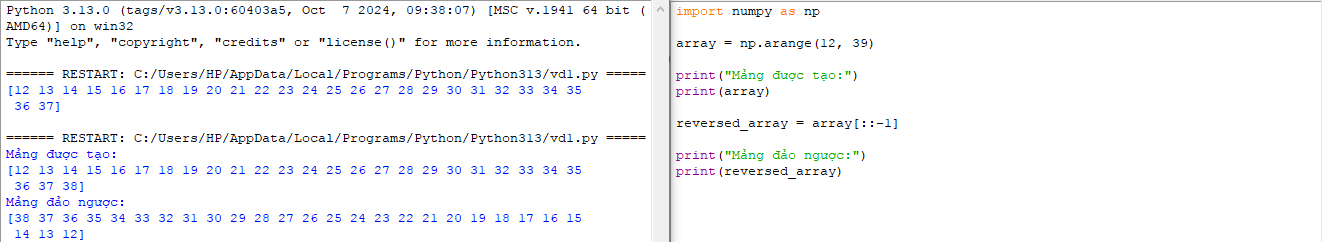
1. **Tạo mảng với np.arange**:

* x = np.arange(12, 38): Tạo một mảng NumPy chứa các giá trị từ 12 đến 37.

1. **In kết quả**:

* print(x): In mảng x ra màn hình.

**Bài 3. Viết chương trình để tạo một mảng với các giá trị nằm trong khoảng từ 12 đến 38 và đảo ngược mảng đã tạo (phần tử đầu tiên trở thành cuối cùng)**

****

**Giải thích**

* 1. **Nhập thư viện NumPy**:

-Nhập thư viện NumPy và gán cho tên viết tắt np.

* 1. **Tạo mảng với giá trị trong khoảng**:

-Sử dụng np.arange(12, 39) để tạo mảng chứa các số nguyên từ 12 đến 38.

* 1. **In mảng ban đầu**:

-In mảng vừa tạo ra.

* 1. **Đảo ngược mảng**:

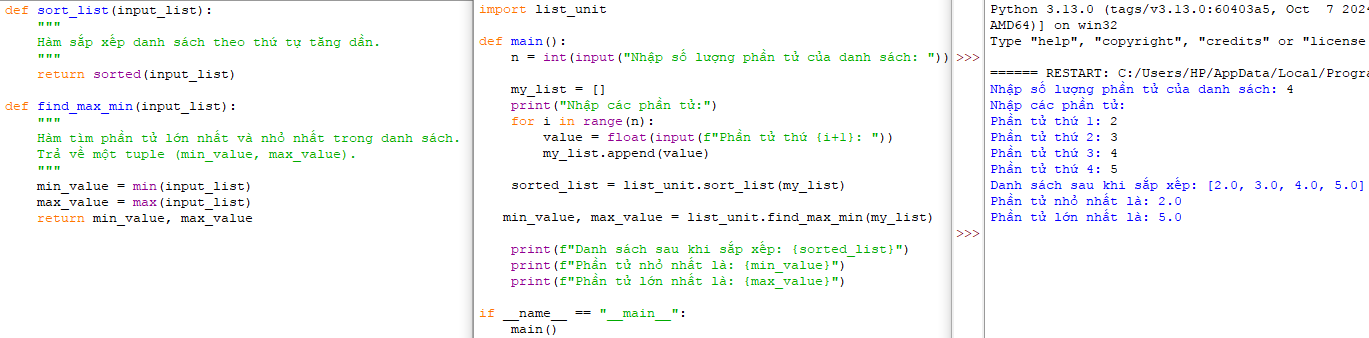
-Sử dụng slicing [::-1] để đảo ngược mảng. Cú pháp này tạo ra một mảng mới với thứ tự phần tử bị đảo ngược.

* 1. **In mảng đã đảo ngược**

**Bài 4. Viết chương trình tìm phần tử lớn nhất và nhỏ nhất của một danh sách**

**- Số lượng và giá trị của list được nhập từ bàn phím**

**- Phương thức sắp xếp và tìm phần tử lớn nhất được viết thành module**

****

**Giải thích:**

**1. Tạo 2 file riêng.**

**Module list\_unit.py:**

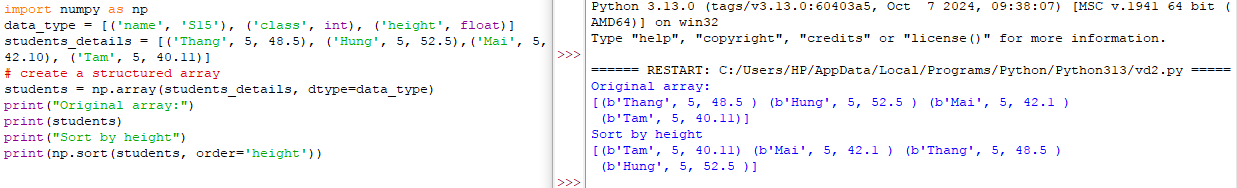
* **sort\_list(input\_list)**: Hàm này sử dụng hàm sorted() của Python để sắp xếp danh sách theo thứ tự tăng dần.
* **find\_max\_min(input\_list)**: Hàm này sử dụng các hàm min() và max() để tìm phần tử nhỏ nhất và lớn nhất trong danh sách và trả về chúng dưới dạng một tuple.

**Chương trình chính vd1.py:**

* Chương trình yêu cầu người dùng nhập số lượng phần tử của danh sách và các giá trị của chúng.
* Sau khi người dùng nhập dữ liệu, chương trình sẽ:
* Gọi hàm sort\_list() từ module để sắp xếp danh sách.
* Gọi hàm find\_max\_min() từ module để tìm phần tử nhỏ nhất và lớn nhất.
* Sau đó, kết quả sẽ được in ra bao gồm:
* Danh sách đã sắp xếp.
* Phần tử nhỏ nhất và phần tử lớn nhất.

**2. Chạy chương trình.**

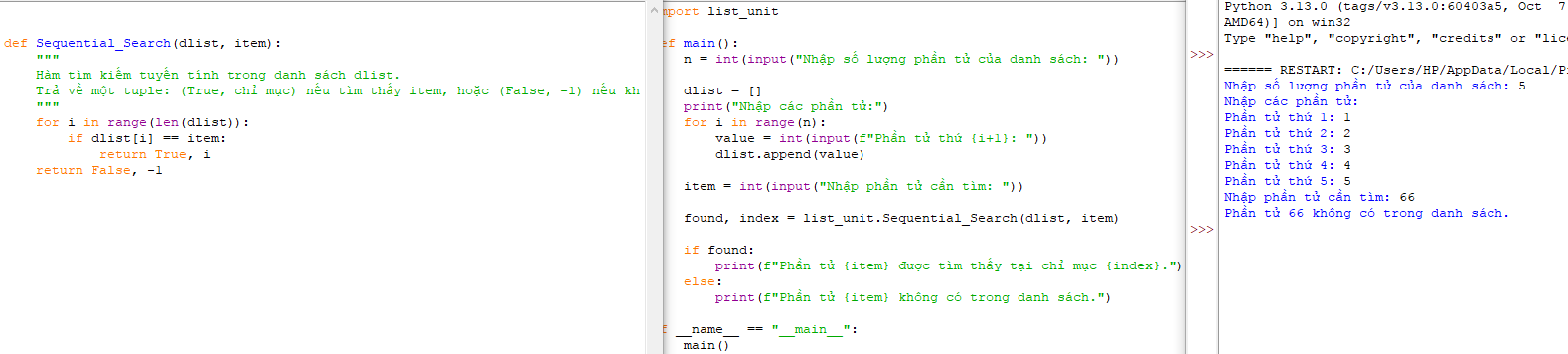
**Bài 5. Viết chương trình sử dụng thư viện NumPy để tạo một mảng có cấu trúc từ tên sinh viên, chiều cao, lớp và các kiểu dữ liệu của họ. Bây giờ sắp xếp các mảng theo chiều cao.**

****

**Giải thích:**

1. **data\_type**: Đây là kiểu dữ liệu cho mảng cấu trúc (structured array). Mỗi phần tử của mảng sẽ có 3 trường:
   * name: chuỗi ký tự (tối đa 15 ký tự).
   * class: kiểu số nguyên (int).
   * height: kiểu số thực (float).
2. **students\_details**: Đây là danh sách các bộ 3 giá trị chứa thông tin về sinh viên, bao gồm tên, lớp và chiều cao.
3. **students**: Đây là mảng NumPy được tạo từ students\_details và kiểu dữ liệu đã được định nghĩa trong data\_type.
4. **np.sort(students, order='height')**: Hàm np.sort sẽ sắp xếp mảng students theo trường height.

**Bài 6.Xây dựng hàm “Sequential\_Search(dlist, item)” (giải thuật tìm kiếm tuyến tính) dưới dạng module. Viết chương trình nhập một dlist n phần tử từ bàn phím và tìm kiếm phần tử item bất kỳ.**

****

**Giải thích:**

**Tạo 2 file**

**1.** **list\_unit.py**:

* Hàm Sequential\_Search(dlist, item) nhận vào một danh sách dlist và một phần tử item cần tìm. Hàm sẽ lặp qua từng phần tử trong danh sách và kiểm tra xem phần tử đó có bằng item không.
* Nếu tìm thấy phần tử, hàm trả về True và chỉ mục của phần tử.
* Nếu không tìm thấy, hàm trả về False và -1.

**2.** **vd2.py**:

* Chương trình này yêu cầu người dùng nhập số lượng phần tử và các giá trị của danh sách.
* Sau đó, chương trình yêu cầu người dùng nhập phần tử cần tìm.
* Hàm Sequential\_Search được gọi để thực hiện tìm kiếm phần tử trong danh sách.
* Kết quả được in ra màn hình: nếu tìm thấy phần tử, in ra chỉ mục của nó; nếu không, thông báo rằng phần tử không có trong danh sách.

**Bài 6. Lập trình hướng đối tượng trong Python**

**1.1. Mục đích**

Giúp sinh viên nắm bắt về lập trình hướng đối tượng trong Python

**1.2. Cơ sở lý thuyết**

Xem các quy tắc xây dựng class, các phương thức và thuộc tính của đối tượng

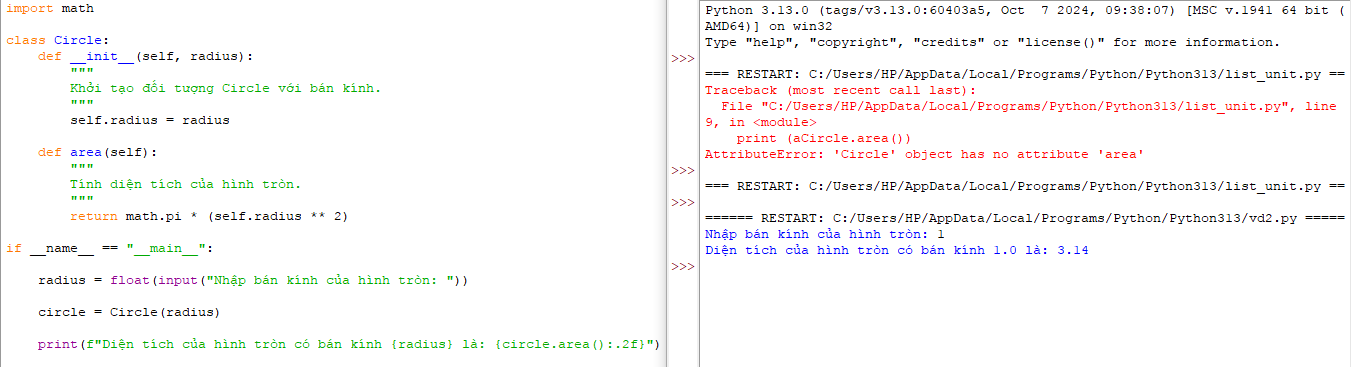
**1.3. Thiết bị thực hành và vật tư tiêu hao**

Thiết bị thực nghiệm: Máy tính

Vật tư tiêu hao: Không

**1.4. Các bước tiến hành**

**Bài 1. Định nghĩa một class có tên là Circle có thể được xây dựng từ bán kính. Circle có một method có thể tính diện tích.**

****

**Giải thích:**

**1.** **Khởi tạo đối tượng (\_\_init\_\_)**:

* Phương thức \_\_init\_\_(self, radius) là phương thức khởi tạo (constructor). Nó nhận một tham số radius (bán kính của hình tròn) và gán giá trị này cho thuộc tính self.radius.
* Khi bạn tạo một đối tượng của class Circle, phương thức \_\_init\_\_ sẽ tự động được gọi để khởi tạo đối tượng.

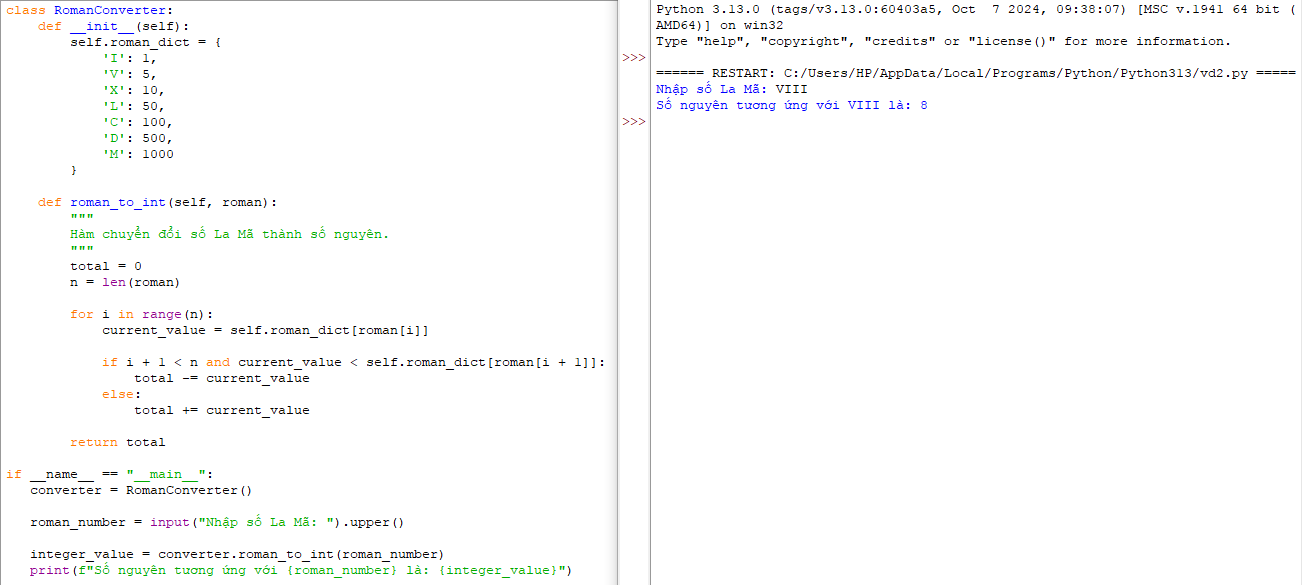
**2.** **Phương thức tính diện tích (area)**:

* Phương thức area(self) không nhận tham số nào ngoài self. Nó sử dụng thuộc tính self.radius để tính diện tích của hình tròn theo công thức: Diện tıˊch=π×baˊn kıˊnh2\text{Diện tích} = \pi \times \text{bán kính}^2Diện tıch=π×baˊn kıˊnh2
* math.pi cung cấp giá trị của π\piπ trong Python.

**3.** **Cách sử dụng**:

* Sau khi nhập bán kính từ người dùng, chương trình tạo đối tượng circle từ class Circle.
* Sau đó, gọi phương thức area() của đối tượng circle để tính diện tích và in kết quả ra màn hình.

**Bài 2. Định nghĩa class Nguoi và 2 class con của nó: Nam, Nu. Tất cả các class có method "getGender" có thể in "Nam" cho class Nam và "Nữ" cho class Nu.**

****

**Giải thích;**

**1.** **Class RomanConverter**:

* **\_\_init\_\_(self)**: Phương thức khởi tạo ánh xạ các ký tự La Mã với giá trị số nguyên tương ứng trong dictionary roman\_dict.
* **roman\_to\_int(self, roman)**: Phương thức này nhận vào một chuỗi La Mã roman và chuyển đổi nó thành một số nguyên. Phương thức lặp qua các ký tự của chuỗi La Mã và áp dụng nguyên lý cộng/trừ để tính giá trị số nguyên.

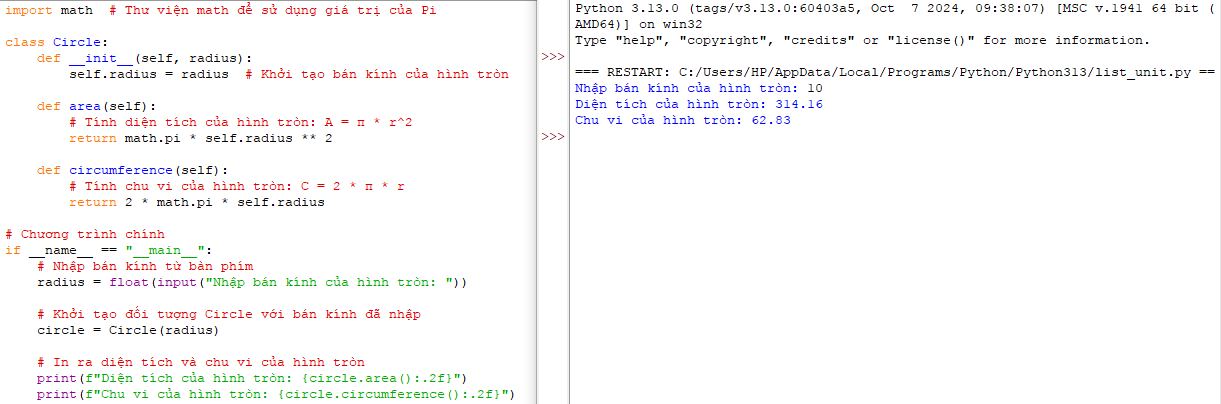
**2.** **Cách chuyển đổi**:

* Duyệt từng ký tự trong chuỗi La Mã.
* So sánh giá trị của ký tự hiện tại với ký tự kế tiếp. Nếu giá trị của ký tự hiện tại nhỏ hơn giá trị ký tự kế tiếp, ta trừ đi giá trị hiện tại (như trường hợp IV = 4).
* Ngược lại, ta cộng giá trị của ký tự vào tổng (như trường hợp VI = 6).

**3.** **Ví dụ sử dụng**:

* Chương trình yêu cầu người dùng nhập một chuỗi La Mã.
* Sau đó, nó chuyển đổi chuỗi La Mã này thành số nguyên và in kết quả.

**Bài 3. Viết một class Python có tên Circle được xây dựng theo bán kính và hai phương thức sẽ tính diện tích và chu vi của hình tròn.**

****

**Giải thích:**

**1.** **Khởi tạo lớp Circle**:

* Phương thức \_\_init\_\_(self, radius) sẽ nhận bán kính radius khi tạo đối tượng Circle và lưu trữ nó trong biến self.radius.

**2.** **Phương thức area(self)**:

* Phương thức này tính diện tích của hình tròn theo công thức A=π×r2A = \pi \times r^2A=π×r2, trong đó sử dụng math.pi để lấy giá trị của hằng số Pi.

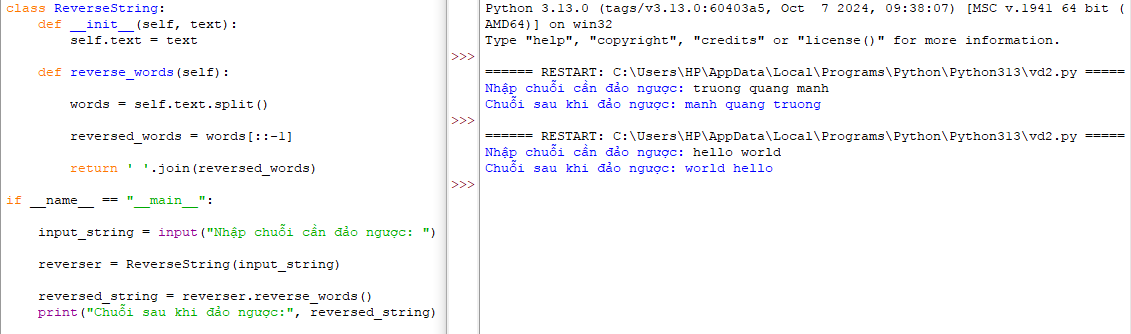
**3.** **Phương thức circumference(self)**:

* Phương thức này tính chu vi của hình tròn theo công thức C=2×π×rC = 2 \times \pi \times rC=2×π×r, cũng sử dụng math.pi cho Pi.

**4.** **Chương trình chính**:

* Người dùng nhập bán kính hình tròn.
* Khởi tạo đối tượng circle từ class Circle với bán kính đã nhập.
* In ra diện tích và chu vi của hình tròn đã được tính từ các phương thức area() và circumference().

**Bài 4. Viết chương trình Python dưới dạng class để đảo ngược chuỗi từ từng chữ.**

****

**1. Định nghĩa lớp**

* **Lớp ReverseString**: Đây là một lớp (class) định nghĩa các phương thức để xử lý chuỗi và đảo ngược các từ trong chuỗi đó.
* **\_\_init\_\_ method**: Phương thức khởi tạo (constructor) được gọi khi bạn tạo một đối tượng của lớp ReverseString. Phương thức này nhận một tham số text và gán giá trị đó cho thuộc tính self.text của đối tượng.

**2. Phương thức**

* **split()**: Phương thức này tách chuỗi thành một danh sách các từ, phân cách bằng các khoảng trắng.
* **' '.join(reversed\_words)**: Sau khi đảo ngược các từ, chúng ta sử dụng phương thức join() để nối lại các từ trong danh sách reversed\_words thành một chuỗi, với mỗi từ cách nhau một dấu cách.

**3. Chạy chương trình chính**

* **input()**: Dùng để nhận đầu vào từ người dùng. Hàm này sẽ yêu cầu người dùng nhập một chuỗi.
* **ReverseString(input\_string)**: Tạo một đối tượng reverser từ lớp ReverseString, truyền vào chuỗi người dùng đã nhập (input\_string).
* **reverser.reverse\_words()**: Gọi phương thức reverse\_words() của đối tượng reverser để đảo ngược thứ tự các từ trong chuỗi.
* **print()**: Cuối cùng, in ra chuỗi đã được đảo ngược.

**Bài 7. Thao tác trên tập tin và thư mục trong Python**

**1.1. Mục đích**

Giúp sinh viên có thể thao tác với các file văn bản sử dụng Python

**1.2. Cơ sở lý thuyết**

Xem các quy tắc mở, đóng file văn bản, đọc và ghi nội dung trong Python

**1.3. Thiết bị thực nghiệm và vật tư tiêu hao**

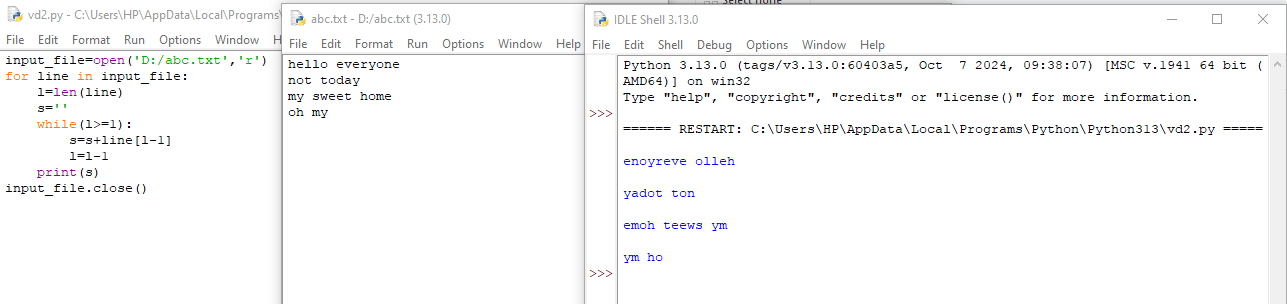
Thiết bị thực nghiệm: Máy tính

Vật tư tiêu hao: Không

**1.4. Các bước tiến hành**

Thực hiện các bài tập sau sử dụng Python

**Bài 1. Chương trình đọc file và in đảo ngược kết quả**

****

**1.**  **Mở file**:

Dòng này mở file có đường dẫn D:/a.txt trong chế độ đọc ('r'). Sau khi mở, đối tượng input\_file đại diện cho file và có thể được sử dụng để đọc nội dung từ file đó.

**2.** **Duyệt qua từng dòng của file**:

Dòng này thực hiện duyệt qua từng dòng trong file a.txt. Mỗi dòng của file sẽ được đọc và lưu vào biến line. Duyệt qua file này sẽ tự động dừng khi không còn dòng nào.

**3.** **Tính chiều dài dòng**:

Dòng này tính toán độ dài của dòng line (tức là số ký tự trong dòng) và gán giá trị đó vào biến l. Biến l sẽ được sử dụng để xác định vị trí của các ký tự trong dòng khi thực hiện đảo ngược.

**4.** **Khởi tạo chuỗi rỗng**:

Dòng này khởi tạo một chuỗi rỗng có tên s. Biến s sẽ được sử dụng để lưu trữ các ký tự của dòng line sau khi chúng bị đảo ngược.

**5.** **Vòng lặp while đảo ngược dòng**:

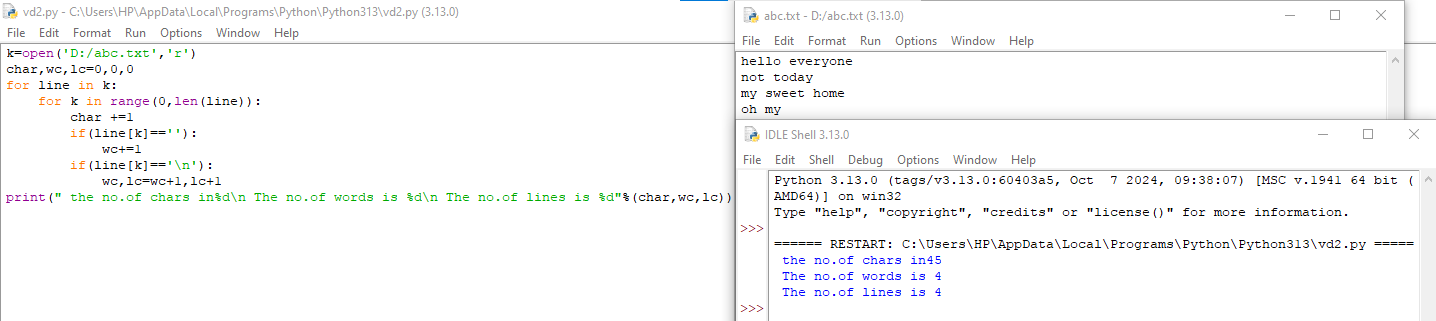
* Vòng lặp này sẽ tiếp tục thực hiện khi giá trị của l lớn hơn hoặc bằng 1.
* line[l - 1] lấy ký tự tại vị trí l-1 của dòng lines = s + line[l - 1] nối ký tự hiện tại vào chuỗi s. Điều này dần dần xây dựng chuỗi s với các ký tự của dòng line theo thứ tự ngược lại.
* l = l - 1 giảm giá trị của l đi 1 để di chuyển về phía trước trong dòng line và lấy ký tự tiếp theo từ cuối dòng.

**6.** **In kết quả**:

Sau khi vòng lặp while hoàn thành, chuỗi s sẽ chứa các ký tự của dòng line đã bị đảo ngược. Dòng này in chuỗi s ra màn hình.

**7.** **Đóng file**:

**Bài 2. Chương trình đọc một file, tính số ký tự, số từ và số dòng của file**

****

**1.** **Mở file**:

Đoạn mã này mở file abc.txt trong chế độ đọc ('r'). Biến k sẽ chứa đối tượng file.

**2. Khởi tạo các biến đếm:**

Các biến char, wc, và lc được khởi tạo để đếm số lượng ký tự (characters), từ (words) và dòng (lines), lần lượt.

**3.** **Duyệt qua từng dòng trong file**:

Đoạn mã này duyệt qua từng dòng trong file abc.txt. Biến line sẽ chứa nội dung của mỗi dòng.

**4.** **Vòng lặp qua từng ký tự trong dòng**:

Đây là một vòng lặp để duyệt qua từng ký tự trong dòng hiện tại, từ chỉ số 0 đến độ dài của dòng line.

**5.** **Đếm số ký tự**:

Mỗi khi vòng lặp qua một ký tự, biến char sẽ được tăng lên 1 để đếm số lượng ký tự trong toàn bộ file.

**6.** **Điều kiện kiểm tra từ**:

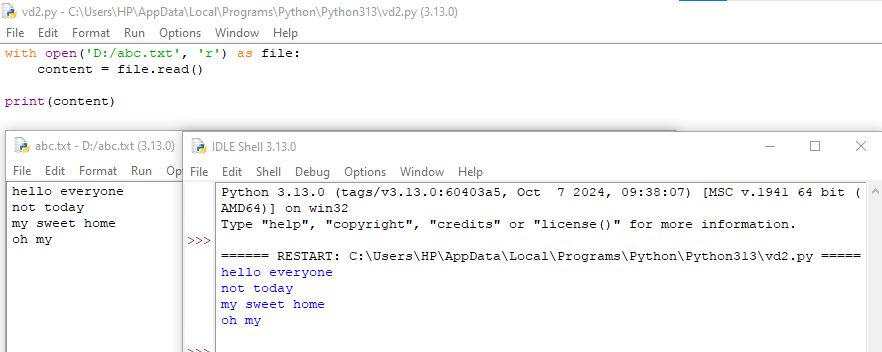
Dòng này có một lỗi: line[k] == '' sẽ không bao giờ đúng, vì line[k] là một ký tự trong dòng, và ký tự không thể bằng chuỗi rỗng ''. Để kiểm tra nếu một từ kết thúc hoặc một khoảng trắng giữa các từ, bạn cần phải kiểm tra xem ký tự là dấu cách ' ' (space).

**7.** **Điều kiện kiểm tra ký tự xuống dòng**:

Đây cũng có một số vấn đề. \n (ký tự xuống dòng) chỉ xuất hiện khi đọc từ file, nhưng bạn đang kiểm tra ký tự trong dòng. Dòng này không cần thiết vì vòng lặp trên file sẽ tự động xử lý dấu xuống dòng.

**8.** **In kết quả**:

**Bài 3. Viết chương trình Python để đọc toàn bộ file văn bản**

****

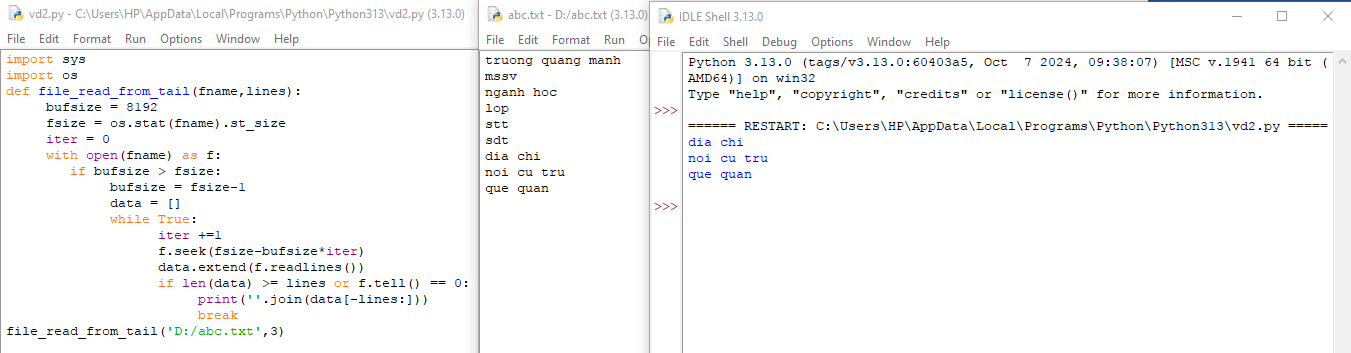
**1.** **open('D:/abc.txt', 'r')**: Mở tệp abc.txt ở chế độ đọc ('r').

**2.** **with**: Dùng with để tự động đóng tệp sau khi hoàn tất. Điều này rất tiện lợi vì bạn không cần gọi file.close().

**3.** **file.read()**: Đọc toàn bộ nội dung tệp và lưu vào biến content.

**4.** **print(content)**: In ra toàn bộ nội dung tệp.

**Bài 4. Viết chương trình Python để đọc n dòng cuối cùng của tệp**

****

**1.** **Đo kích thước tệp**: Dòng fsize = os.stat(fname).st\_size lấy kích thước của tệp để biết được tệp có bao nhiêu byte.

**2.** **Đọc từ cuối tệp**:

* Với bộ đệm bufsize (mặc định là 8192 byte), bạn sẽ đọc các phần của tệp từ cuối lên.
* Lệnh f.seek(fsize - bufsize \* iter) giúp dịch chuyển con trỏ tệp từ cuối tệp vào từng phần nhỏ (tương ứng với bộ đệm).

**3.** **Đọc các dòng**:

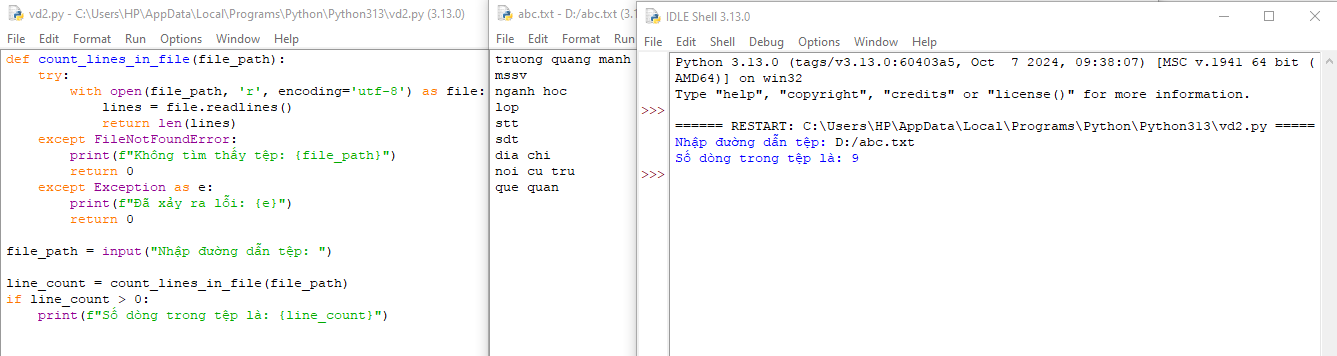
* Dùng f.readlines() để đọc các dòng từ vị trí con trỏ tệp hiện tại.
* Các dòng này được lưu vào danh sách data.

**4.** **Dừng khi đủ số dòng**:

* Khi số dòng đã đọc đạt đến yêu cầu (hoặc đã đọc hết tệp), vòng lặp dừng lại và in ra lines dòng cuối cùng của tệp.

**5.** **In kết quả**:

**Bài 5. Viết chương trình Python để đếm số dòng trong tệp văn bản**

****

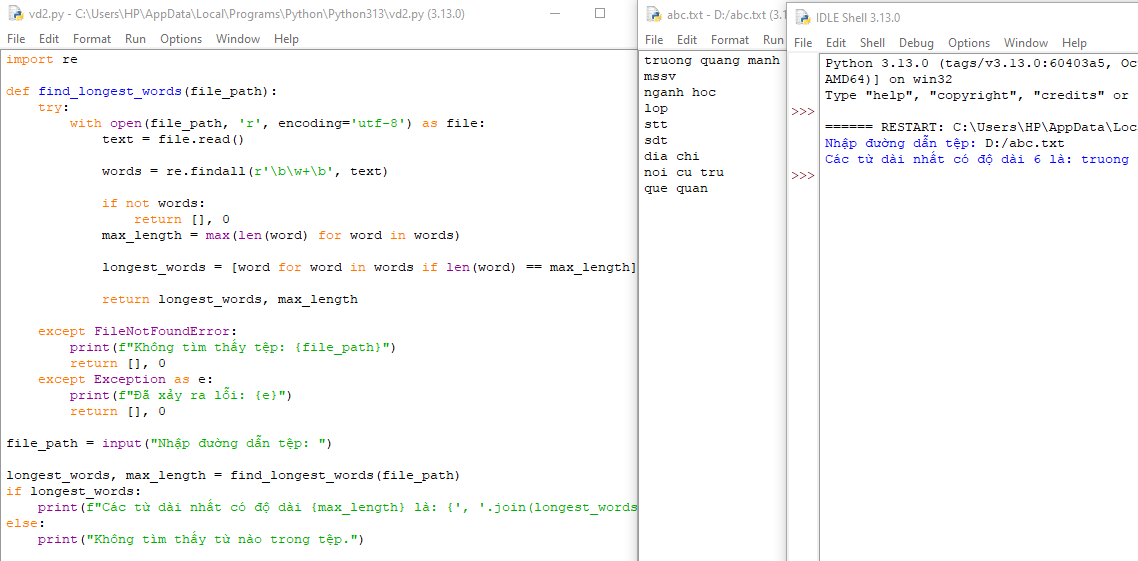
**1.** **open(file\_path, 'r', encoding='utf-8')**: Mở tệp ở chế độ đọc ('r') với mã hóa UTF-8.

**2.** **file.readlines()**: Đọc tất cả các dòng trong tệp và lưu chúng vào danh sách lines.

**3.** **len(lines)**: Trả về số phần tử trong danh sách lines, chính là số dòng trong tệp.

**4.** **try-except**: Đảm bảo chương trình xử lý các lỗi nếu tệp không tồn tại hoặc có vấn đề khi đọc tệp.

**Bài 6. Viết chương trình python để tìm những từ dài nhất trong văn bản**

****

**1.**  **re.findall(r'\b\w+\b', text)**: Sử dụng biểu thức chính quy để tách các từ trong văn bản. \b là ký tự biên giới từ, \w+ là một hoặc nhiều ký tự chữ cái và số.

**2.** **max(len(word) for word in words)**: Tìm độ dài của từ dài nhất trong danh sách các từ.

**3.** **[word for word in words if len(word) == max\_length]**: Tạo danh sách các từ có độ dài bằng với độ dài của từ dài nhất.

**4.** **file.read()**: Đọc toàn bộ nội dung của tệp.

**5.** **exception handling**: Đảm bảo chương trình không bị lỗi khi tệp không tồn tại hoặc gặp sự cố trong quá trình đọc tệp.

**Bài 8: Lập trình giao diện trong Python**

**1.1. Mục đích**

Giúp sinh viên làm quen với lập trình đồ họa và sử dụng thư viện turtle, Tkinter trong python.

**1.2. Cơ sở lý thuyết**

Xem các quy tắc xây dựng và quản lý layout,widget, hộp thoại, đồ họa trong python

**1.3. Thiết bị thực hành thí nghiệm và vật tư tiêu hao**

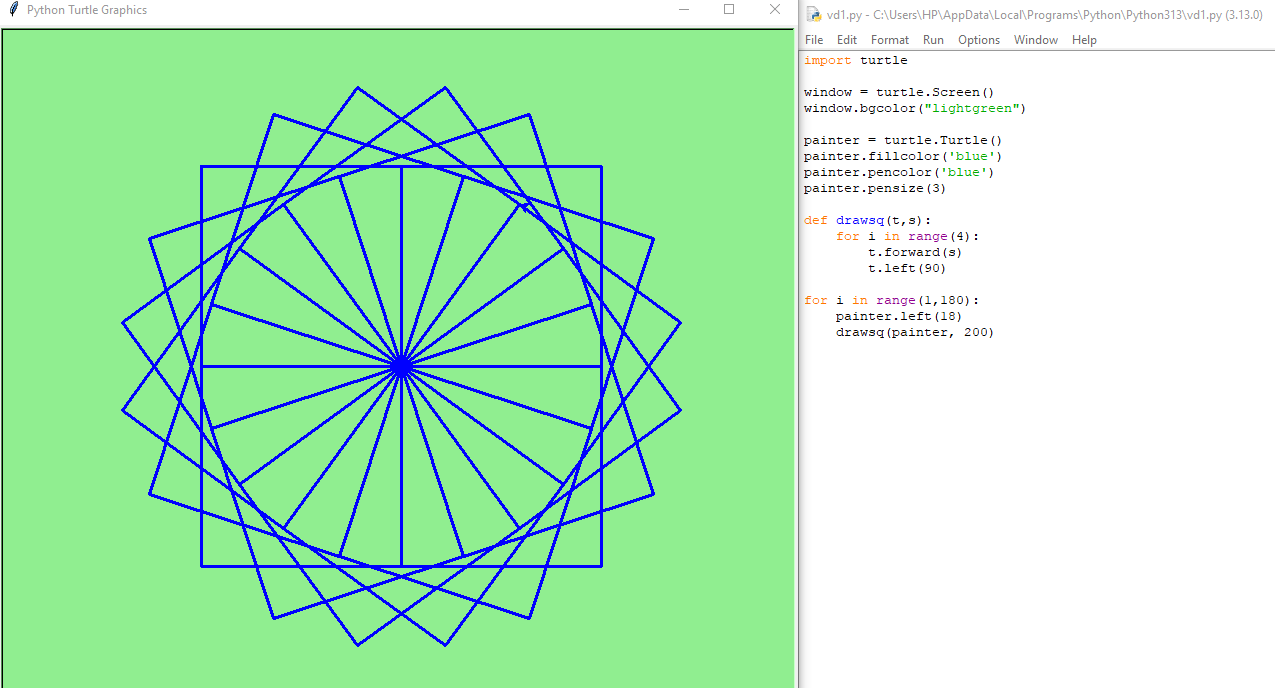
Thiết bị thực hành thí nghiệm: Máy tính

Vật tư tiêu hao: Không

**1.4. Các bước tiến hành**

Thực hiện các bài tập thực hành sử dụng python

**Bài 1. Viết chương trình đồ họa sử dụng thư viện turtle, kiểm tra kết quả và giải thích chương trình**

****

1. **Khởi tạo cửa sổ vẽ:**

**-window = turtle.Screen():** Lệnh này tạo ra một cửa sổ mới để hiển thị đồ họa.

**-window.bgcolor("lightgreen**"): Thiết lập màu nền của cửa sổ thành màu xanh nhạt (light green).

**2. Khởi tạo "Painter" (con rùa vẽ):**

**-painter = turtle.Turtle():** Tạo một đối tượng "turtle" (con rùa) với tên là painter để vẽ.

**-painter.fillcolor('blue')**: Đặt màu sắc của vùng tô bên trong các hình vẽ là màu xanh dương.

**-painter.pencolor('blue')**: Đặt màu sắc của đường vẽ (viền) là màu xanh dương.

**-painter.pensize(3)**: Đặt độ dày của đường vẽ (kích thước bút) là 3.

**3. Định nghĩa hàm vẽ hình vuông:**

-Hàm **drawsq(t, s)** nhận vào hai tham số:

**+t:** là đối tượng turtle (trong trường hợp này là painter).

**+s:** là độ dài mỗi cạnh của hình vuông.

-Trong hàm, vòng lặp for sẽ lặp 4 lần (vì hình vuông có 4 cạnh). Mỗi lần:

**+t.forward(s)**: Di chuyển turtle về phía trước với chiều dài bằng s (độ dài cạnh).

**+t.left(90)**: Quay turtle sang trái 90 độ để vẽ cạnh tiếp theo.

\*Kết quả là con rùa sẽ vẽ một hình vuông với các cạnh có độ dài là s.

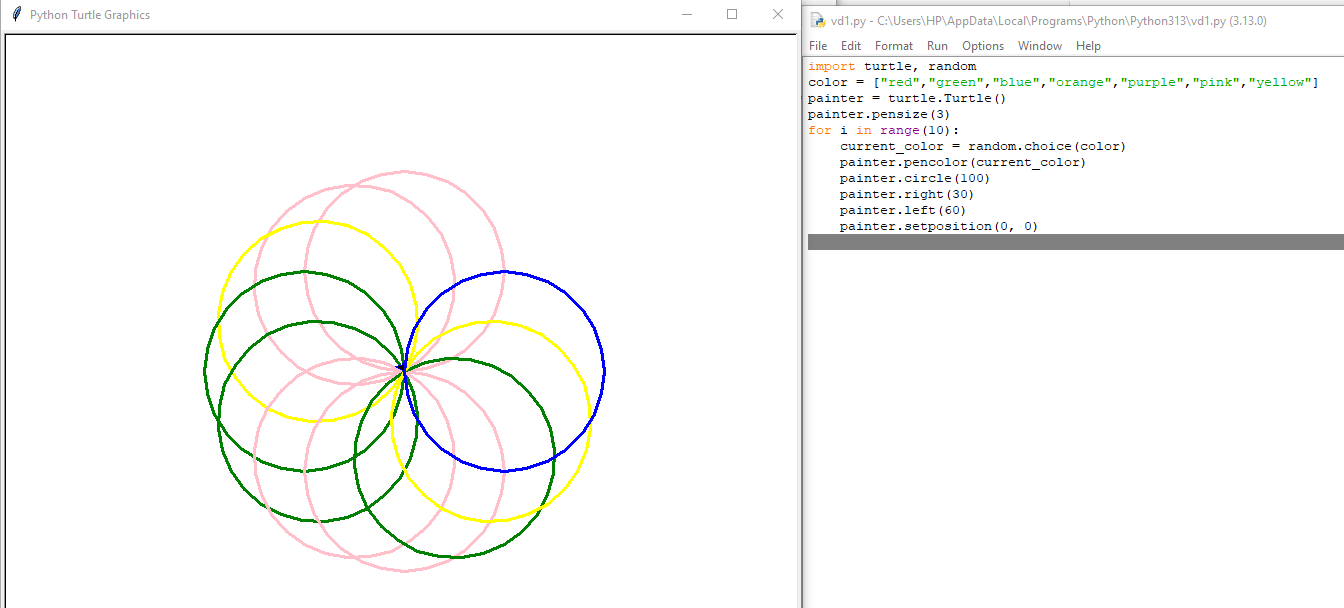
**4. Vòng lặp chính vẽ các hình vuông xoay quanh điểm trung tâm:**

**-for i in range(1, 180):** Vòng lặp này sẽ lặp lại 179 lần (từ 1 đến 179). Mỗi lần lặp, nó sẽ vẽ một hình vuông mới với góc xoay thay đổi.

**-painter.left(18):** Mỗi lần lặp, con rùa sẽ xoay sang trái một góc 18 độ. Điều này tạo ra hiệu ứng xoay liên tục giữa các hình vuông.

**-drawsq(painter, 200):** Gọi hàm drawsq để vẽ một hình vuông có chiều dài cạnh là 200 đơn vị.

**Bài 2. Viết chương trình đồ họa sử dụng thư viện turtle, kiểm tra kết quả và giải thích chương trình**

****

**1. Khởi tạo thư viện và màu sắc ngẫu nhiên:**

**-import turtle, random**: Nhập các thư viện cần thiết. turtle là thư viện vẽ đồ họa, còn random giúp lựa chọn màu sắc ngẫu nhiên từ danh sách color.

**-color = ["red","green","blue","orange","purple","pink","yellow"]**: Định nghĩa một danh sách chứa các màu sắc có sẵn để sử dụng cho đường vẽ.

**2.Khởi tạo đối tượng Turtle:**

**painter = turtle.Turtle()**: Tạo một đối tượng turtle có tên là painter để vẽ.

**painter.pensize(3)**: Đặt độ dày của đường vẽ (bút vẽ) là 3.

**3. Vòng lặp để vẽ các hình tròn:**

**-for i in range(10)**: Vòng lặp này sẽ chạy 10 lần, mỗi lần vẽ một hình tròn.

**-color = random.choice(color):** Mỗi lần lặp, chương trình sẽ chọn ngẫu nhiên một màu trong danh sách color và gán cho biến color.

**-painter.pencolor(color):** Đặt màu của đường vẽ thành màu đã chọn ngẫu nhiên.

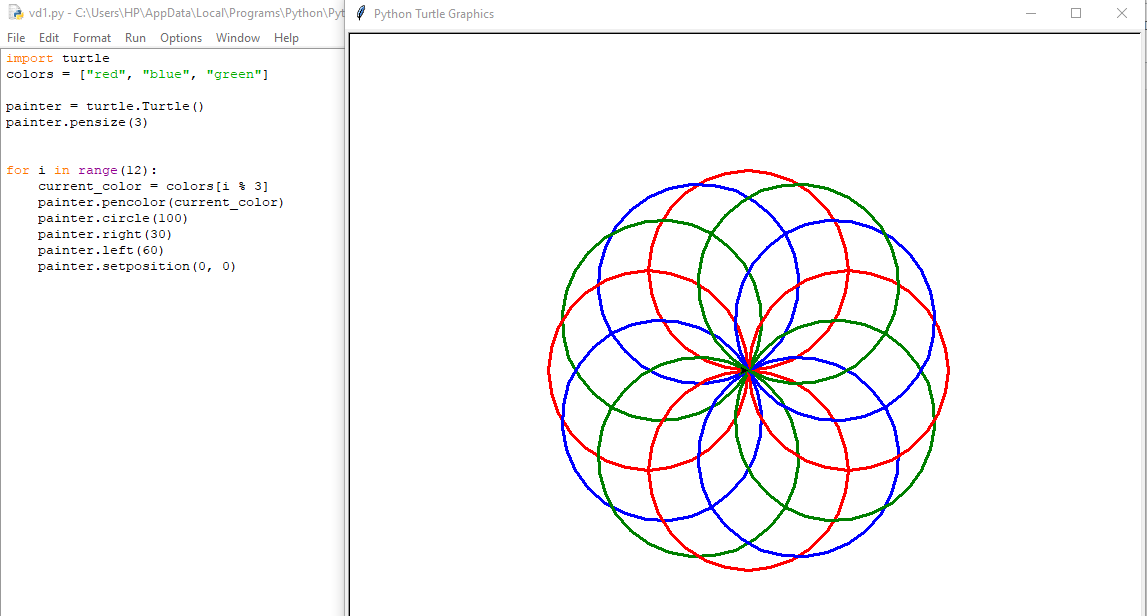
**-painter.circle(100):** Vẽ một hình tròn có bán kính 100 đơn vị.

**-painter.right(30)**: Sau khi vẽ xong mỗi hình tròn, turtle sẽ quay sang phải 30 độ.

**-painter.left(60):** Sau đó, turtle sẽ quay sang trái 60 độ.

**-painter.setposition(0, 0):** Di chuyển turtle trở lại vị trí gốc (tọa độ (0, 0)) để vẽ hình tròn tiếp theo từ điểm trung tâm.

**Bài 3. Dựa trên các kết quả đạt được từ các chương trình trên hãy viết chương trình hiển thị hình ảnh đồ họa sau**

****

**1. Khởi tạo thư viện và danh sách màu sắc:**

**-import turtle:** Nhập thư viện turtle để sử dụng các chức năng vẽ đồ họa.

**-colors = ["red", "blue", "green"]:** Tạo một danh sách có ba màu sắc: đỏ (red), xanh dương (blue), và xanh lá cây (green). Các màu này sẽ được sử dụng lần lượt trong mỗi vòng lặp để vẽ các hình tròn.

**2. Khởi tạo đối tượng Turtle (con rùa):**

**-painter = turtle.Turtle():** Tạo một đối tượng turtle có tên là painter để thực hiện các thao tác vẽ trên cửa sổ.

**-painter.pensize(3):** Đặt độ dày của bút vẽ là 3. Điều này quyết định độ dày của đường vẽ mà turtle sẽ tạo ra.

**3. Vòng lặp vẽ các hình tròn với màu sắc theo thứ tự:**

**-for i in range(12):** Vòng lặp này sẽ chạy 12 lần, từ i = 0 đến i = 9. Mỗi lần lặp sẽ vẽ một hình tròn.

**-current\_color = colors[i % 3]:**

Ở mỗi lần lặp, toán tử i % 3 sẽ chia chỉ số i cho 3 và lấy phần dư. Điều này giúp chọn một màu trong danh sách colors theo thứ tự:

+Khi i = 0, i % 3 = 0, chọn màu đầu tiên trong danh sách, là red.

+Khi i = 1, i % 3 = 1, chọn màu thứ hai trong danh sách, là blue.

+Khi i = 2, i % 3 = 2, chọn màu thứ ba trong danh sách, là green.

Sau đó, khi i tiếp tục tăng, nó sẽ lặp lại quá trình này, chọn lại màu từ đầu (red, blue, green,...).

**4. Đặt màu vẽ và vẽ hình tròn:**

**-painter.pencolor(current\_color):** Đặt màu sắc của đường vẽ (đường viền) thành màu current\_color, được chọn từ bước trên.

**-painter.circle(100):** Vẽ một hình tròn có bán kính là 100 đơn vị. Sau mỗi lần vẽ, turtle sẽ vẽ một hình tròn mới, sử dụng màu sắc đã chọn.

**5. Quay và di chuyển Turtle:**

**-painter.right(30):** Sau khi vẽ xong một hình tròn, turtle sẽ quay 30 độ sang bên phải để chuẩn bị vẽ hình tròn tiếp theo.

**-painter.left(60):** Sau đó, turtle quay thêm 60 độ sang bên trái. Điều này tạo ra một góc quay tổng cộng là 30 độ sang phải + 60 độ sang trái = 90 độ, giúp đảm bảo rằng mỗi hình tròn vẽ ra không bị trùng lên nhau, mà vẫn giữ được cấu trúc xoay.

**-painter.setposition(0, 0):** Di chuyển turtle về lại vị trí gốc (tọa độ (0, 0)) sau khi vẽ mỗi hình tròn, chuẩn bị cho việc vẽ hình tròn tiếp theo tại vị trí đó.

**6. Lặp lại quá trình 12 lần:**

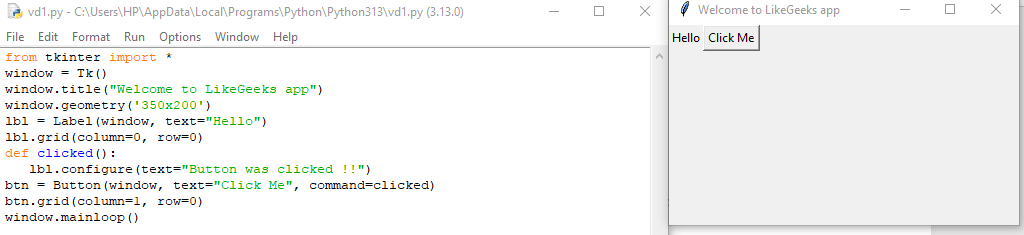
Vòng lặp sẽ tiếp tục chạy 12 lần. Mỗi lần, turtle sẽ chọn một màu sắc từ danh sách theo thứ tự red, blue, green, vẽ một hình tròn, rồi quay lại điểm gốc và tiếp tục cho đến khi vẽ đủ 12 hình tròn.

**Bài 4. Viết chương sử dụng thư viện đồ họa tkinter thực hiện:**

**a) Xây dựng cửa sổ đồ họa window form**

**b) Thêm một widget (button) vào window form**

**c) Xây dựng phương thức xử lý sự kiện phím bấm**

****

**1. Khởi tạo cửa sổ (Window):**

**window = Tk**(): Đây là bước khởi tạo cửa sổ chính của ứng dụng Tkinter. Lệnh này tạo một đối tượng cửa sổ (Tk()) để chứa tất cả các widget (các thành phần giao diện người dùng) của ứng dụng.

**2. Đặt tiêu đề và kích thước cửa sổ:**

**window.title("Welcome to LikeGeeks app"):** Đặt tiêu đề cho cửa sổ là "Welcome to LikeGeeks app". Tiêu đề này sẽ xuất hiện trên thanh tiêu đề của cửa sổ ứng dụng.

**window.geometry('350x200'**): Thiết lập kích thước cửa sổ là 350 pixels chiều rộng và 200 pixels chiều cao.

**3. Tạo và hiển thị nhãn (Label):**

**lbl = Label(window, text="Hello"):** Tạo một widget nhãn (label) có nội dung là "Hello", sẽ hiển thị trên cửa sổ window.

**lbl.grid(column=0, row=0):** Sử dụng phương thức grid để đặt widget nhãn này ở vị trí (0, 0) trên lưới (grid) của cửa sổ.

**4. Tạo một hàm xử lý sự kiện (clicked):**

def clicked(): Đây là một hàm mà sẽ được gọi khi người dùng nhấn vào nút "Click Me".

**lbl.configure(text="Button was clicked !!"):** Sau khi người dùng nhấn nút, hàm này sẽ thay đổi nội dung của nhãn (lbl) từ "Hello" thành "Button was clicked !!".

**5. Tạo và đặt nút (Button):**

**btn = Button(window, text="Click Me", command=clicked):** Tạo một widget nút (button) với nội dung là "Click Me". Khi nút này được nhấn, hàm clicked() sẽ được gọi.

**btn.grid(column=1, row=0):** Đặt nút ở vị trí (1, 0) trên lưới của cửa sổ.

**6. Chạy vòng lặp chính của ứng dụng Tkinter:**

**window.mainloop():** Đây là vòng lặp chính của ứng dụng Tkinter. Nó giữ cho cửa sổ mở và cho phép người dùng tương tác với các widget như nút, nhãn. Vòng lặp này sẽ tiếp tục chạy cho đến khi người dùng đóng cửa sổ.