



Lập trình Python cơ bản

Dành cho học sinh THCS

MỤC LỤC

LỜI NÓI ĐẦU.....	3
Bài 1: LÀM QUEN VỚI NGÔN NGỮ LẬP TRÌNH PYTHON	4
1. Giới thiệu.....	4
2. Cài đặt Python trên Windows	4
3. Viết chương trình Python.....	7
4. Lệnh print	8
5. Thực hành	11
Bài 2: CÁC PHÉP TOÁN SỐ HỌC TRONG PYTHON	13
1. Phép toán số học.....	13
2. Phép toán cộng và nhân.....	13
3. Phân toán trừ và chia.....	15
4. Thực hành	17
Bài 3: LŨY THỪA VÀ CĂN BẬC 2	20
1. Giao diện Jupyter Notebook	20
2. Phép toán lũy thừa.....	21
3. Phép toán tính căn bậc 2	22
4. Thực hành	22
Bài 4: BIỂU THỨC ĐIỀU KIỆN (IF)	25
1. Câu lệnh if	25
2. Câu lệnh if ... else	25
3. Điều kiện if ... elif ... else.....	28
4. Thực hành	29
Bài 5: VÒNG LẶP (LOOP)	31
1. Vòng lặp while.....	31
2. Vòng lặp for.....	33
3. Vòng lặp for và mảng	37
4. Câu lệnh break	37
5. Thực hành	38
Bài 6: LIST	41
1. Tổng quan về danh sách trong Python	41
2. Truy cập phần tử trong danh sách	42

3. Một số phép toán của tập hợp	43
4. Thực hành	48
Bài 7: CÁC LỆNH CƠ BẢN CỦA TURTLE GRAPHICS	51
1. Tọa độ màn hình	51
2. Turtle graphics	52
3. Các lệnh cơ bản của turtle	53
a. Mode standard.....	53
b. Turtle motion	56
4. Thực hành	67
Bài 8: VẼ HÌNH ROBOT VỚI TURTLE GRAPHICS	68
1. Color control	68
2. Filling	69
5. Thực hành	76
PHỤ LỤC 1: CÁC CÂU LỆNH CƠ BẢN CỦA TURTLE	78
PHỤ LỤC 2: CÁC MẪU ĐOẠN CODE MINH HỌA.....	79
TÀI LIỆU THAM KHẢO.....	83

LỜI NÓI ĐẦU

Tài liệu lập trình cơ bản với Python được biên soạn dành cho các em học sinh Trung Học Cơ Sở (THCS) yêu thích về lập trình, mong muốn tìm hiểu và khám phá thế giới lập trình. Tài liệu được trình bày với cấu trúc đơn giản, dễ hiểu, phù hợp với tất cả các em học sinh ở tất cả các trình độ của lứa tuổi THCS. Phần thí dụ và bài tập trong tài liệu này chủ yếu dựa vào tài liệu Toán học 6, tập một làm nền tảng.

Don't put off until tomorrow what you can do today!

(Đừng để tới ngày mai những việc bạn có thể làm hôm nay!)

Benjamin Franklin

Lập trình là một công việc thú vị đối với những bạn đam mê. Tuy nhiên, các em muốn đạt được đến một trình độ lập trình tốt thì ***các em phải đi từ nền tảng cơ bản, từng bước, không nên nôn nóng***. Tài liệu này được thiết kế gồm có phần lý thuyết, thí dụ và bài tập minh họa từng phần chi tiết, trong đó có những phần để trống. Yêu cầu là các em sẽ đọc từng phần theo trình tự: lý thuyết → thí dụ → thực hành theo thí dụ → suy nghĩ để hiểu, ghi giải thích hoặc ghi chú vào những mục để trống đó.

Tài liệu này được ngẫu hứng biên soạn dựa theo tinh thần tự học hỏi khám phá của các em học sinh lớp 6 Trường Trung học cơ sở Nguyễn An Khương – Hóc Môn. Đây là phiên bản đầu tiên được hoàn thành trong thời gian rất ngắn nên không tránh khỏi các sai sót, nên mong các em học sinh, quý vị phụ huynh nhiệt tình đóng góp.

Mọi góp ý xin vui lòng gửi về Huỳnh Thái Học, Khưu Minh Cảnh, Nguyễn Quang Huy địa chỉ email hoc.ht@vlu.edu.vn, Giảng viên Khoa CNTT Trường Đại học Văn Lang Tp.HCM.

Xin chân thành cảm ơn các em học sinh và quý vị phụ huynh!

Nhóm tác giả

Bài 1: LÀM QUEN VỚI NGÔN NGỮ LẬP TRÌNH PYTHON

1. Giới thiệu

Python là một ngôn ngữ lập trình (programming language), là một dạng ngôn ngữ được dùng để viết ra phần mềm (software) cho người sử dụng (users). Đây là ngôn ngữ được tạo bởi Guido Van Rossum, được phát hành lần đầu tiên vào tháng 2 năm 1991. Tên “Python” được lấy từ tên một phần trong sê-ri “Monty Python’s Flying Circus”, chương trình hài cuối những năm 1970.

Python có thể chạy dễ dàng trên môi trường của Hệ điều hành Windows lẫn môi trường Linux, Mac OS. Trong tài liệu này các em sẽ được hướng dẫn cài đặt, sử dụng và thực hành trong môi trường của Windows.

2. Cài đặt Python trên Windows

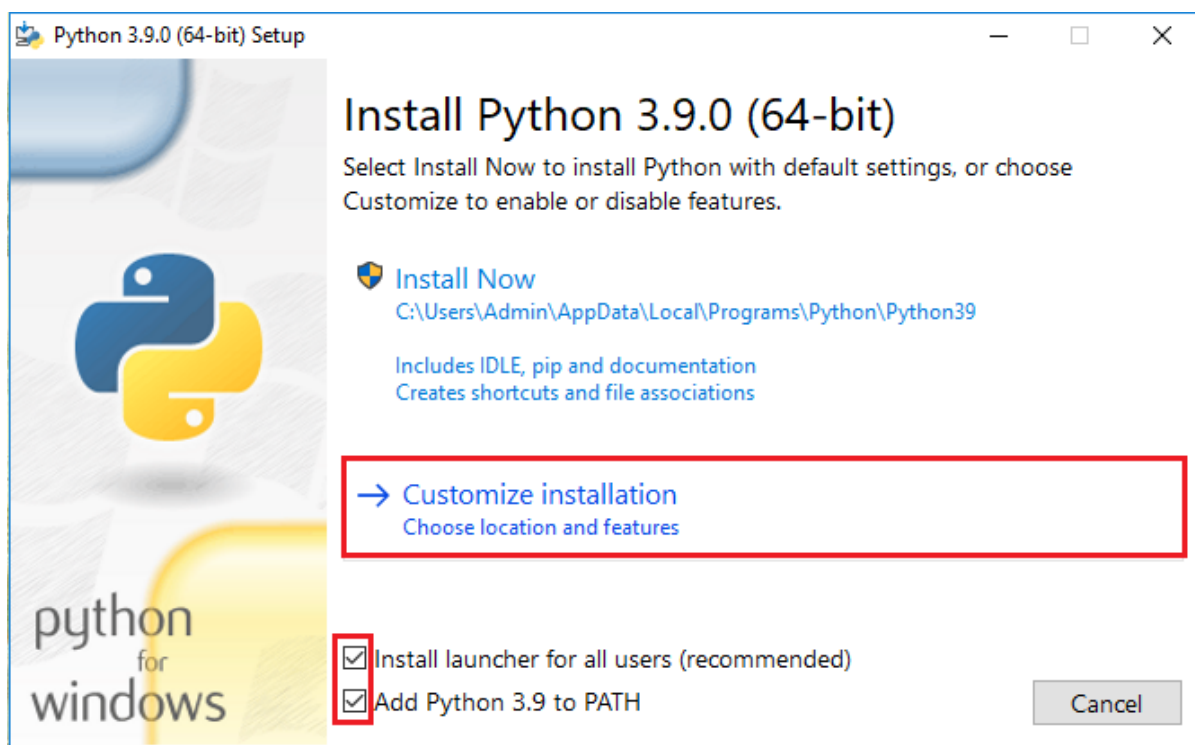
Các em truy cập liên kết <https://www.python.org/downloads/> để tải xuống bản phát hành mới nhất (latest version) của Python.



Hình 1. Giao diện của trang web www.python.org/downloads

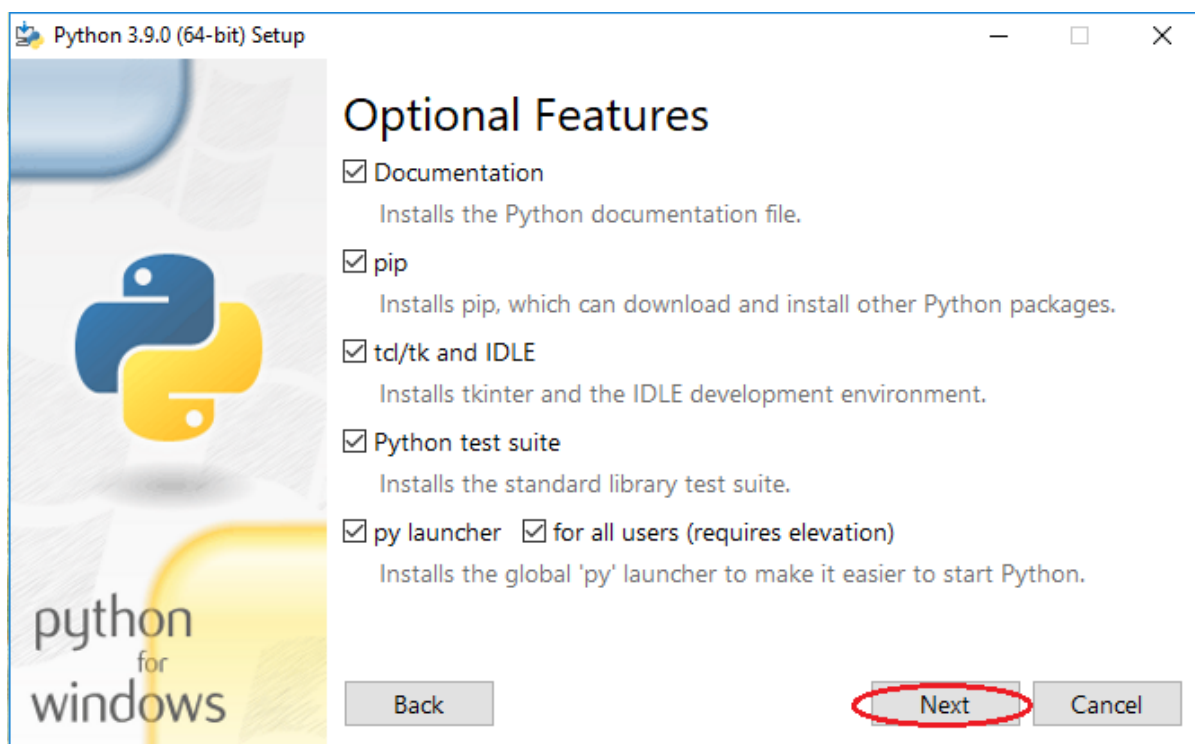
Click đúp chuột vào file vừa tải xuống để tiến hành cài đặt Python:

Chọn Customize installation:



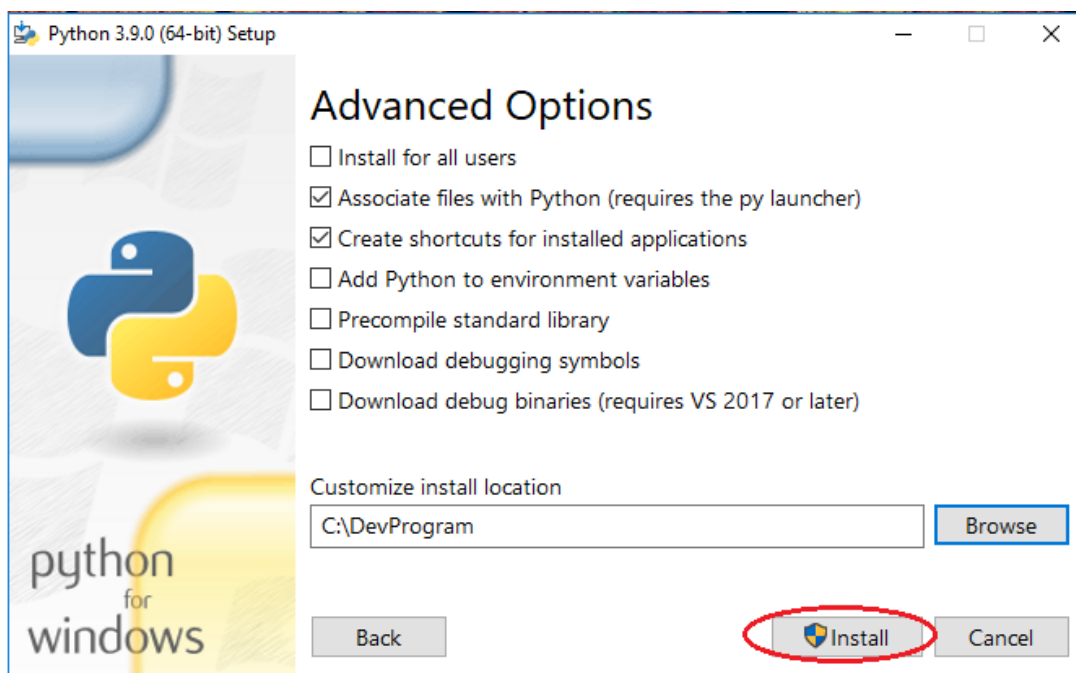
Hình 2. Các bước cài đặt Python trên Windows

Chọn tất cả các tính năng tùy chọn, click Next:



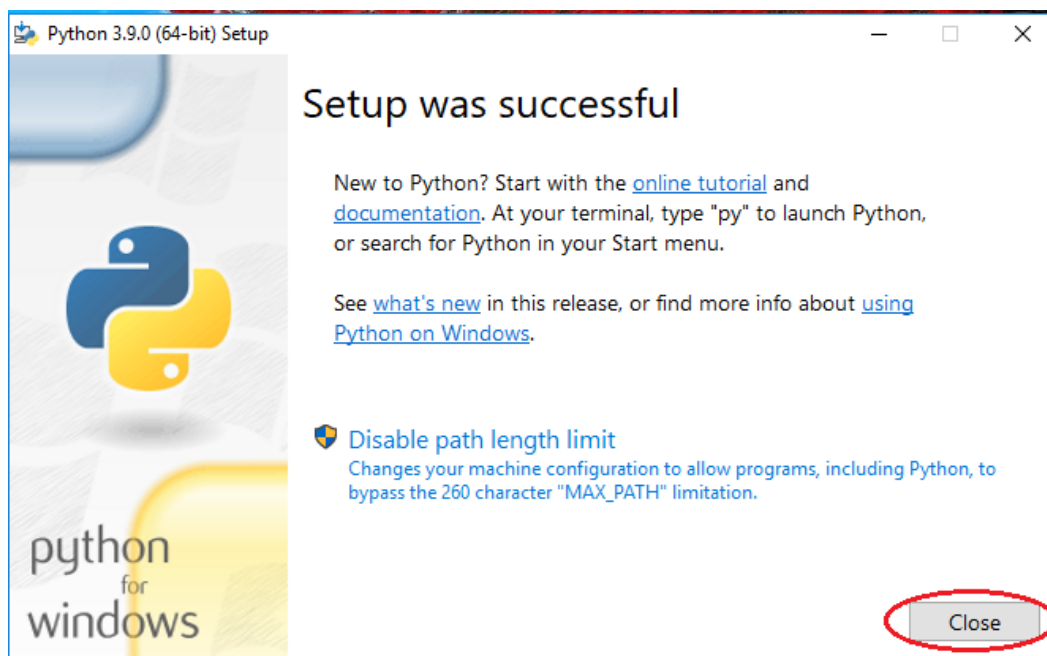
Hình 3. Các bước cài đặt Python trên Windows

Chọn vị trí mà Python sẽ được cài đặt sau đó click Install.



Hình 4. Các bước cài đặt Python trên Windows

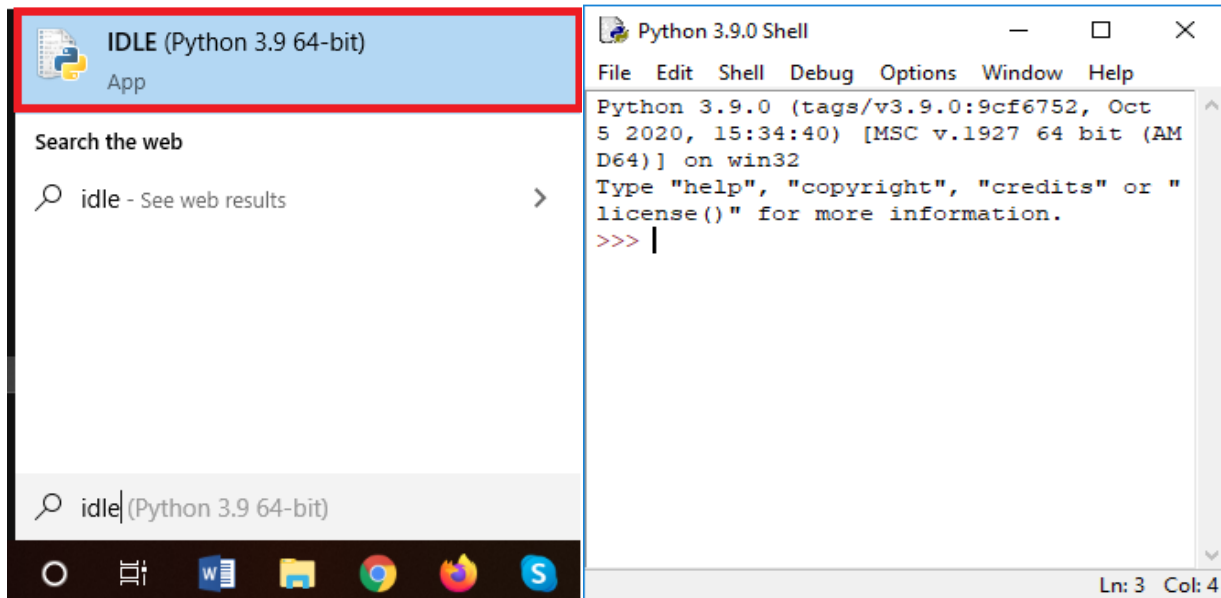
Click Close sau khi cài đặt thành công.



Hình 5. Các bước cài đặt Python trên Windows

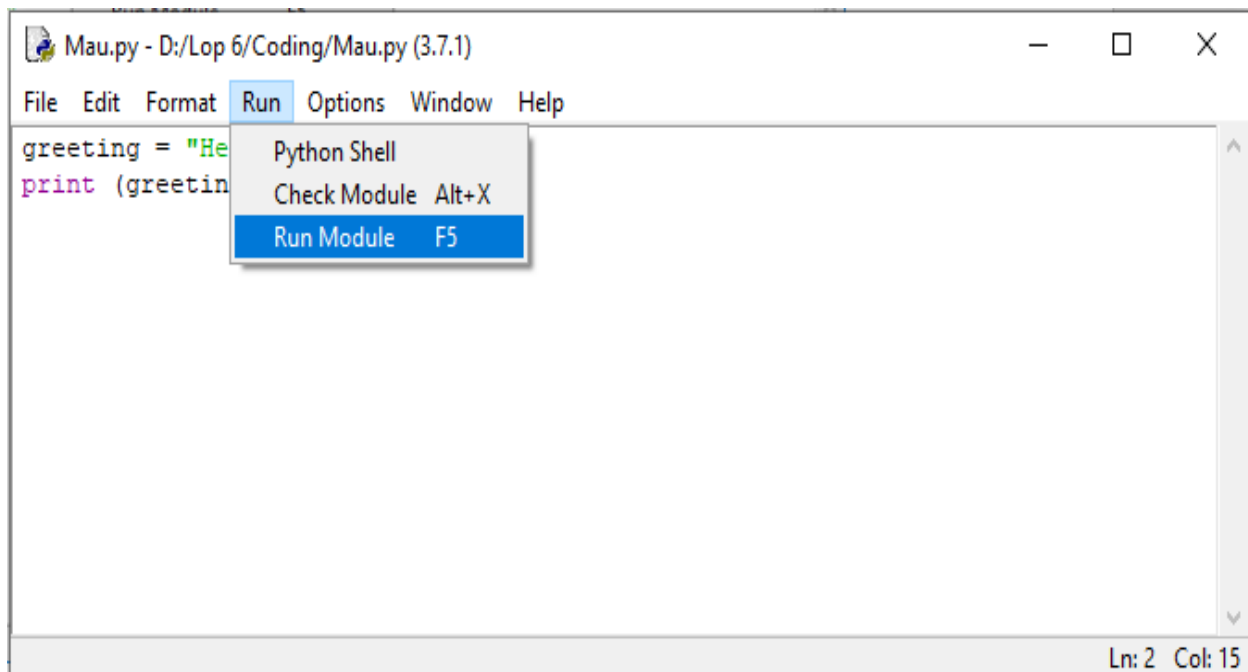
3. Viết chương trình Python

Mở Start Menu: tìm kiếm với từ khóa IDLE



Hình 6. Mở cửa sổ làm việc của Python

Vào menu File để tạo file mới với phần mở rộng .py. Ví dụ: helloworld.py



Hình 7. Cửa sổ làm việc của Python

Viết mã (code) Python trong file và lưu nó. Để chạy tệp, đi đến **Run > Run Module** hoặc đơn giản là nhấn **F5**



```
Python 3.7.1 (v3.7.1:260ec2c36a, Oct 20 2018, 14:57:15) [MSC v.1915 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: D:/Lop 6/Coding/Mau.py =====
Hello world
>>> |
```

Ln: 6 Col: 4

Hình 8. Giao diện hiển thị kết quả của Python

4. Lệnh print

Lệnh **print** hay còn gọi là hàm **print**. Đó là một lệnh được sử dụng để hiển thị một nội dung ra màn hình kết quả.

Cách viết:

print (phantu1, phantu2, phantu3,...)

Trong đó:

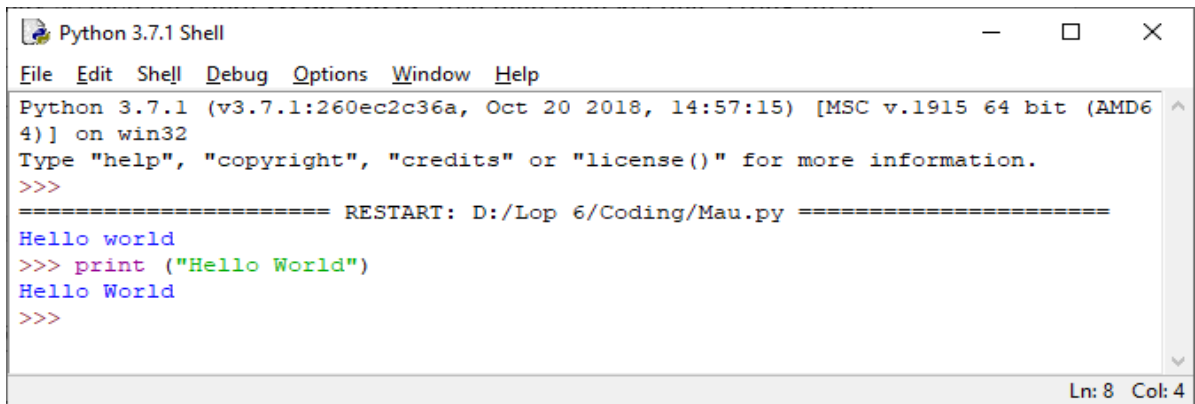
- phantu1, phantu2, phantu3,... : là chuỗi, số hay phép tính, *chuỗi là một tập hợp gồm nhiều ký tự.*
- Các phần tử cách nhau bởi dấu “,”.

Thí dụ 1:

```
print ("Hello world!")
```

Câu lệnh này sẽ hiển thị chuỗi **Hello world!** trên màn hình kết quả. Trong thí dụ này, hàm **print** chỉ có phantu1, nó là chuỗi, giá trị của nó là **Hello world!**

✓ Kết quả của thí dụ 1:



```
Python 3.7.1 Shell
File Edit Shell Debug Options Window Help
Python 3.7.1 (v3.7.1:260ec2c36a, Oct 20 2018, 14:57:15) [MSC v.1915 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: D:/Lop 6/Coding/Mau.py =====
Hello world
>>> print ("Hello World")
Hello World
>>>
```

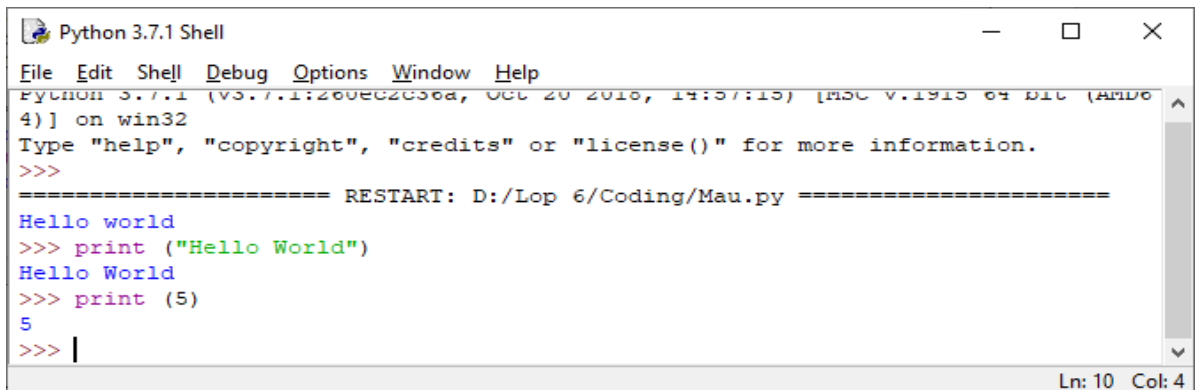
Hình 9. Kết quả hiển thị của thí dụ 1 - Bài 1

Thí dụ 2:

```
print (5)
```

Câu lệnh này sẽ hiển thị số **5** lên màn hình kết quả.

✓ Kết quả của thí dụ 2:



```
Python 3.7.1 Shell
File Edit Shell Debug Options Window Help
Python 3.7.1 (v3.7.1:260ec2c36a, Oct 20 2018, 14:57:15) [MSC v.1915 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: D:/Lop 6/Coding/Mau.py =====
Hello world
>>> print ("Hello World")
Hello World
>>> print (5)
5
>>> |
```

Hình 10. Kết quả hiển thị của thí dụ 2 - Bài 1

Thí dụ 3:

```
print ("6 + 5 = ", 6 + 5)
print ("3 * 4 = ", 3 * 4)
print ("100 - 1 = ", 100 - 1)
print ("30 / 3 = ", 30 / 3)
```

Hàm print trong thí dụ 3 có hai phần tử:

- **Dòng thứ nhất:**
 - Phần tử 1 là "6+5 =", đây là chuỗi, vì 6+5 = ở trong dấu nháy kép.
 - Phần tử 2 là phép toán cộng của hai số 6 và 5.
 - **Dòng thứ hai:**
 - Phần tử 1 là "3*4 =", đây là chuỗi, vì 3 * 4 = ở trong dấu nháy kép.
 - Phần tử 2 là phép toán nhân hai số 3 và 4.
 - **Dòng thứ ba:**
 - Phần tử 1 là , đây là , vì
 - Phần tử 2 là
 - **Dòng thứ tư:**
 - Phần tử 1 là , đây là , vì
 - Phần tử 2 là
- ✓ *Kết quả của thí dụ 3:*

The image shows two windows from a Python IDE. The top window, titled 'Bai1-Thidu3.py - D:/Lop 6/Coding/Bai1-Thidu3.py (3.7.1)', contains the following Python code:

```
print ("6 + 5 = ", 6 + 5)
print ("3 * 4 = ", 3 * 4)
print ("100 - 1 = ", 100 - 1)
print ("30 / 3 = ", 30 / 3)
```

The bottom window, titled 'Python 3.7.1 Shell', shows the output of the code after execution:

```
===== RESTART: D:/Lop 6/Coding/Bai1-Thidu3.py =====
6 + 5 =  11
3 * 4 =  12
100 - 1 = 99
30 / 3 = 10.0
>>>
```

Hình 11. Kết quả hiển thị của thí dụ 3 - Bài 1

5. Thực hành

Bài tập 1: Các em hãy sử dụng trang web được nêu ở mục 2 và thực hành theo thí dụ 1.

Bài tập 2: Các em hãy sử dụng trang web được nêu ở mục 2 và thực hành theo thí dụ 2.

Bài tập 3: Các em hãy sử dụng trang web được nêu ở mục 2 và thực hành theo thí dụ 3.

Bài tập 4: Các em hãy chép các dòng dưới đây vào **vùng 1**, thực hiện nút **Run**, xem và ghi lại kết quả ở **vùng 3**.

```
print ("*")
print ("**")
print ("***")
print ("****")
```

Kết quả:

.....
.....
.....
.....

Bài tập 5: Các em làm tương tự như bài 4, nhưng chép các dòng ở bài 4 theo thứ tự ngược lại (từ dưới lên trên), thực hiện và ghi lại kết quả.

Kết quả:

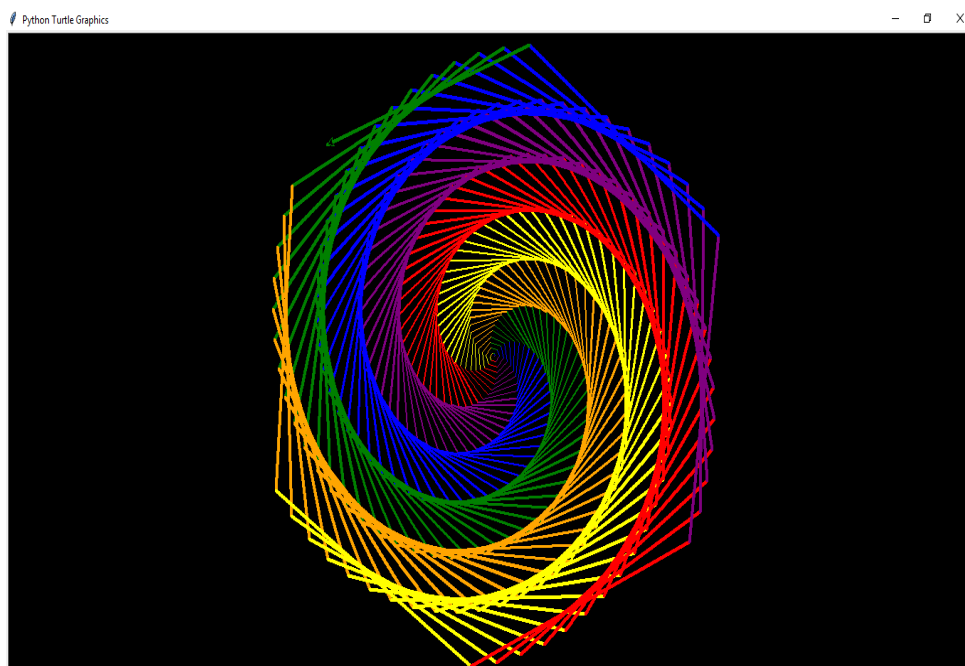
.....
.....
.....
.....

Bài tập 6: Các em hãy sử dụng hàm print, ký tự * và các khoảng trắng (nếu có) để in ra màn hình như hình minh họa dưới đây:

```
  *
 * * *
* * * * *
```

Gốc thư giản: Các em làm xong bài tập 6 có thể thư giản bằng cách chép đoạn code dưới đây vào chương trình **Python** và chạy để xem kết quả.

```
# Python program to draw  
# Rainbow Benzene  
# using Turtle Programming  
import turtle  
colors = ['red', 'purple', 'blue', 'green', 'orange', 'yellow']  
t = turtle.Pen()  
turtle.bgcolor('black')  
for x in range(360):  
    t.pencolor(colors[x%6])  
    t.width(x/100 + 1)  
    t.forward(x)  
    t.left(59)
```



Hình 12. Kết quả hiển thị của Bài 1 – thư giản

Bài 2: CÁC PHÉP TOÁN SỐ HỌC TRONG PYTHON

1. Phép toán số học

Phép toán số học (arithmetic operation) là một phép tính dựa trên các tham số đầu vào (gọi là toán hạng) để tạo ra một giá trị đầu ra. Các phép toán số học gồm: phép toán cộng, phép toán trừ, phép toán nhân và phép toán chia, phép tính lũy thừa, phép tính khai căn bậc hai.

Thí dụ 1:

$$5 + 7 = 12$$

Trong đó:

- Tham số đầu vào: **5** và **7**
- Phép toán: phép cộng (+)
- Đầu ra: **12**

2. Phép toán cộng và nhân

- Phép toán cộng hai số (+):

Thí dụ 2:

```
a = 6
b = 8
tong = a + b
print ("Tong hai so la: ", tong)
```

- ✓ Kết quả của thí dụ 2:

.....

.....

.....

- Phép toán cộng các chuỗi (+):

Thí dụ 3:

```

surname = "Gates"
name = "Bill"
fullname = name + " " + surname
print (" Full name: ", fullname)

```

- ✓ Kết quả của thí dụ 3

.....

- Phép toán nhân hai số (*):

Thí dụ 4:

```

n = 37
m = 3
tich = n * m
print ("Tich hai so la: ", tich)
print (n, "*", m, tich)

```

- ✓ Kết quả của thí dụ 4

.....

.....

Đố vui 1:

1. Không lập trình, giả sử các em đã biết được kết quả $37 \times 3 = 111$, các em hãy suy nghĩ sử dụng kết quả trên để tính 37×6 .

Gợi ý: sử dụng tính chất phân phối của phép nhân với phép cộng

$$a(b + c) = ab + ac$$

.....

2. Tương tự như bài 1, cho biết $15873 \times 7 = 111111$. Hãy tính nhanh

$$15873 \times 21$$

.....

Gốc thư giãn: Các em hãy thực thi đoạn chương trình sau và ghi lại kết quả hiện thị. Các em có thể thay đổi các con số để có các kết quả khác nhau.

```
from turtle import *  
forward(100)  
shape('turtle')  
right(45)  
forward(150)
```

.....

.....

.....

.....

.....

.....

3. Phân toán trừ và chia

- Phép toán trừ hai số (-):

Thí dụ 5:

```
a = 6  
b = 8  
hieu = a - b  
print (a , " - ", b , " = ", hieu)
```

- ✓ Kết quả của thí dụ 5:

.....

.....

.....

.....

.....

- Phép toán chia hai số (/):

Thí dụ 6:

```
a = 6
b = 8
thuong = a / b
print(a, "/", b, "=", thuong)
```

- ✓ Kết quả của thí dụ 6:
-

- Phép toán chia lấy dư hai số (%):

Thí dụ 7:

```
m = 5
n = 3
phandu = m % n
print(m, "% ", n, "=", phandu)
```

- ✓ Kết quả của thí dụ 7:
-
-

Ghi nhớ của phép chia:

1. Cho hai số tự nhiên m và n , giả sử m chia hết cho n (ký hiệu $m : n$), nghĩa là tồn tại k sao cho $m = k \times n$; k gọi là **thương** của phép chia, còn lại là ước số của m .

Chẳng hạn: $6 : 3$, và $6 = 2 \times 3$. Trong minh họa này $m = 6$, $n = 3$ và $k = 2$.

2. Trường hợp m không chia hết cho n (giả sử $m > n$), khi đó kết quả phép toán m chia n sẽ tồn tại số dư r , $r \neq 0$.

Ký hiệu:

$$m = k \times n + r \quad (1)$$

Thí dụ: $m = 7$, $n = 2$; 7 chia cho 2 sẽ được 3 và dư 1. Khi đó: $7 = 3 \times 2 + 1$

Trong đó: $k = 3$, $r = 1$

Đố vui 2: Các em hãy nghĩ cách tìm phần nguyên (k) của phép toán m chia cho n có dư r .

Gợi ý: Suy nghĩ theo công thức (1)

.....
.....

4. Thực hành

Bài tập 1: Cho hình chữ nhật có chiều dài là 8 và chiều rộng là 3, các em hãy suy nghĩ và lập trình để tính diện tích và chu vi của hình chữ nhật đó, in giá trị tính toán ra màn hình kết quả.

Gợi ý: Các em hãy dùng các **biến** để lưu lại giá trị của chiều dài và chiều rộng, sau đó áp dụng công thức tính diện tích và chu vi để thực hiện.

```
cdai = 8
crong = 3
dientich = cdai * crong
chuvi = (cdai + crong) * 2
print ("Diện tích của hình chữ nhật là: ", dientich)
print ("Chu vi của hình chữ nhật là: ", chuvi)
```

Bài tập 2: Các em hãy viết chương trình lần lượt xác định các phần tử của tập hợp M của các số tự nhiên x, biết rằng $x = a + b$. Trong đó:

$$a \in \{25 ; 38\}; b \in \{14 ; 23\};$$

Gợi ý:

```
M1 = 25 + 14
M2 = 25 + 23
M3 = 38 + 14
M4 = 38 + 23
Print ("Các phần tử của tập hợp M: ", M1, M2, M3, M4)
```

Bài tập 3: Biết bán kính Trái Đất là $R = 6370\text{km}$, các em hãy lập trình để tính bán kính Mặt Trăng (r), biết rằng bán kính Trái Đất gấp khoảng 4 lần bán kính Mặt Trăng.

Trước khi các em lập trình, hãy viết công thức tính bán kính Mặt Trăng theo R như trên.

.....

.....

.....

.....

.....

Bài tập 4: Năm nhuận có 366 ngày. Các em hãy ghi lời giải lập trình và sau đó hãy lập trình để cho biết năm nhuận gồm bao nhiêu tuần và còn dư mấy ngày?

.....

.....

.....

.....

.....

.....

.....

.....

.....

Bài tập 5: (Không lập trình) Một phép chia có tổng của số bị chia và số chia bằng 72. Biết rằng thương là 3 và số dư bằng 8. Tìm số bị chia và số chia.

.....

.....

.....

.....

.....

.....

.....

.....

.....

Bài tập 6: Các em hãy thực hành bài dưới đây và hãy đoán xem randint(1,100) nghĩa là gì?, các em hãy thử thay 1 hay 100 trong randint và thực thi lại đoạn code, xem lại kết quả.

```
from random import randint
def numberGame():
    #choose a random number
    #between 1 and 100
    number = randint(1,100)
    print(number)
```

.....

.....

.....

.....

.....

.....

.....

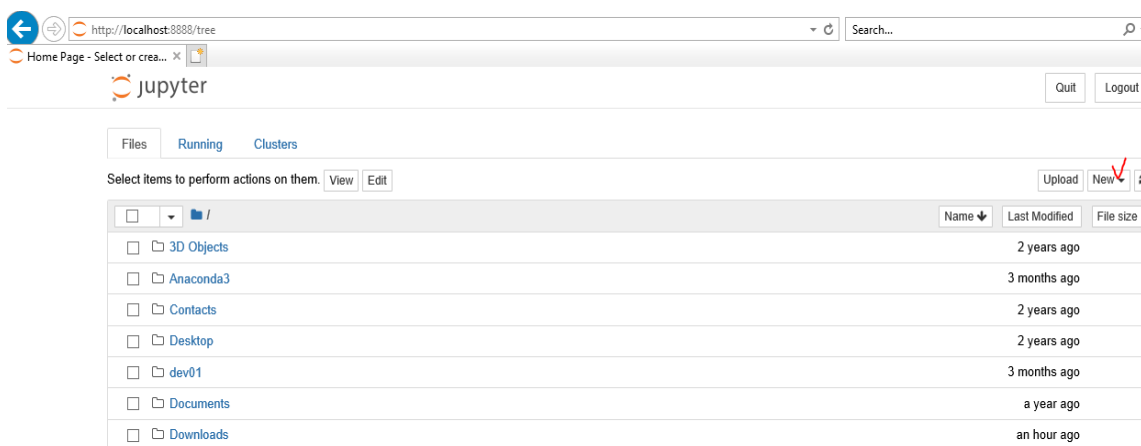
.....

.....

Bài 3: LỮY THỪA VÀ CĂN BẬC 2

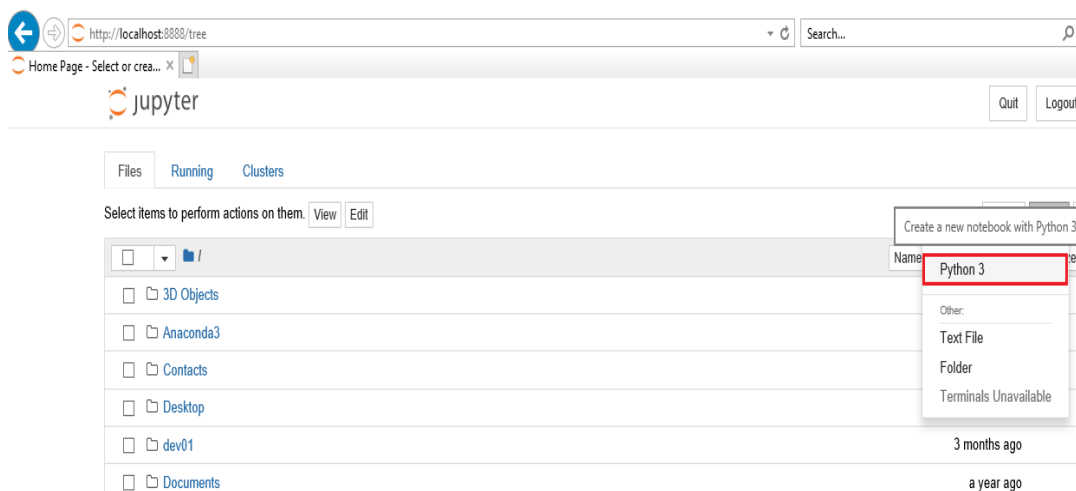
Các thí dụ minh họa trong bài ba sẽ được thể hiện bằng phần mềm **Jupyter Notebook** để tiện cho việc minh họa, đây cũng là một **IDE** (Integrated Development Environment) riêng cho lập trình Python. Đối với các em khi thực hành thì vẫn thực hành trên phần mềm Python đã được cài đặt. Các em cũng có thể tự tìm hiểu và cài đặt Jupyter Notebook, nhưng trong phần tài liệu này sẽ không trình bày các cài đặt.

1. Giao diện Jupyter Notebook

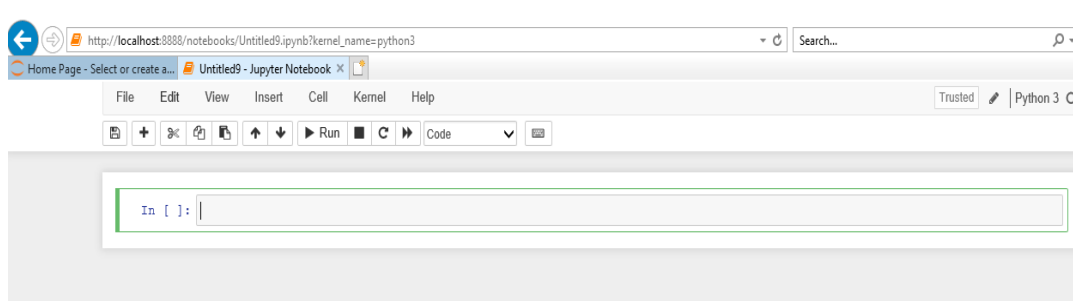


Hình 13. Giao diện của Jupyter Notebook

Để tạo mới một tập tin, các em chọn menu **New** (góc trên bên phải), chọn tiếp **Python3**.



Hình 14. Làm việc với Python3 trên Jupyter Notebook - 1



Hình 15. Làm việc với Python3 trên Jupyter Notebook - 2

2. Phép toán lũy thừa

- Cách 1: Sử dụng toán tử ******

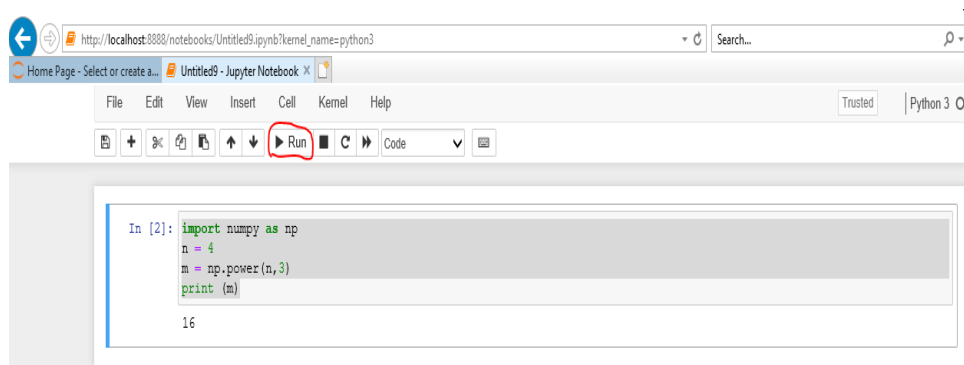
Thí dụ 1: Cho $n = 6$, tính giá trị của m , biết $m = n^2$

```
n = 6
m = n ** 2
print (m)
```

✓ Kết quả của thí dụ 1 (theo cách tính lũy thừa theo toán tử ******):

- Cách 2: Sử dụng hàm power của **thư viện numpy**

```
import numpy as np
n = 6
m = np.power(n,2)
print (m)
```

Hình 16. Làm việc với Python3 trên Jupyter Notebook - 3

✓ Kết quả của thí dụ 1 (theo cách sử dụng hàm **power** của thư viện **numpy**):

.....

3. Phép toán tính căn bậc 2

Sử dụng hàm **sqrt** của thư viện **numpy**

Thí dụ 2: Cho $n = 9$, tính giá trị căn bậc 2 của n .

```
import numpy as np
n = 16
m = np.sqrt(n)
print (m)
```

✓ Kết quả của thí dụ 2:

.....

4. Thực hành

Bài tập 1: Các em hãy thực hành các thí dụ có trong bài học, sau đó ghi lại công thức cách tính lũy thừa và căn bậc hai.

.....

.....

Bài tập 2: “**nan**” là một thông báo cho biết kết quả tính toán không hợp lệ vì một lỗi nào đó đã xảy ra trong quá trình tính toán. Các em hãy quan sát kết quả của bài dưới đây và cho biết lý do vì sao chúng ta lại có kết quả này. Hãy sửa lại giá trị của n hợp lệ và lập trình thực hiện lại bài tính này.

```
In [1]: import numpy as np
n = -9
m = np.sqrt(n)
print (m)
```

nan

C:\Users\Admin\Anaconda3\lib\site-packages\ipykernel_launcher.py:3: RuntimeWarning: invalid value encountered in sqrt
This is separate from the ipykernel package so we can avoid doing imports until

Em rút ra được gì từ thí dụ này?

.....

.....

Bài tập 3: Các em thử suy nghĩ và vận dụng lý thuyết toán học để chỉ cần sử dụng hàm hay toán tử lũy thừa vẫn có thể tính giá trị biểu thức của căn bậc hai được, hãy thực hành lại thí dụ 2, nhưng không sử dụng hàm sqrt để tính căn bậc 2.

Gợi ý: Ta đã biết (còn chưa biết thì học ☺), $\sqrt{n} = \sqrt[2]{n} = n^{\frac{1}{2}}$

.....

.....

.....

Bài tập 4: Các em hãy vận dụng phương pháp đã suy diễn ở bài tập 3, hãy viết đoạn chương trình nhập vào số nguyên m tùy ý và tính căn bậc 3 của số nguyên đó. Các em hãy chạy chương trình hai lần, lần thứ nhất nhập m dương và lần thứ hai nhập m âm, ghi lại kết quả nhìn được vào mục dưới đây.

.....

.....

.....

Bài 5: Tương tự như bài 4, nhưng các em hãy viết đoạn chương trình nhập vào số nguyên m tùy ý và tính căn bậc 4 của số nguyên đó. Các em hãy chạy chương trình hai lần, lần thứ nhất nhập m dương và lần thứ hai nhập m âm, ghi lại kết quả nhìn được vào mục dưới đây.

.....

.....

.....

.....

.....

.....

.....

.....

Gốc thư giản: Các em hãy thử thực thi đoạn code sau *Changing directions*, ghi lại kết quả hiện thị. Các em có thể thay đổi các số 100, 45, 150 thành các giá trị khác, thực thi và xem kết quả.

```
from turtle import *  
forward(100)  
shape('turtle ')  
right(45)  
forward(150)
```

.....

.....

.....

.....

.....

Bài 4: BIỂU THỨC ĐIỀU KIỆN (IF)

1. Câu lệnh if

Cú pháp:

```
if Biểu thức điều kiện:
```

```
    Câu lệnh*
```

Câu lệnh* được thực hiện khi “Biểu thức điều kiện” thỏa mãn (*biểu thức điều kiện có giá trị True hay còn gọi là điều kiện Đúng*).

Thí dụ 1:

```
n = 10
m = 5
sodu = n % m
print ("sodu = ", sodu)
if sodu == 0:
    print ("n chia hết cho m")
print ("-----")
```

✓ Kết quả của thí dụ 1:

.....
.....

2. Câu lệnh if ... else

```
if Biểu thức điều kiện:
```

```
    Câu lệnh 1
```

```
else:
```

```
    Câu lệnh 2
```

Câu lệnh 1 được thực hiện khi “Biểu thức điều kiện” có giá trị True, **Câu lệnh 2** được thực hiện khi “Biểu thức điều kiện” có giá trị False.

Thí dụ 2:

```
n = 10
m = 4
sodu = n % m
print ("sodu = ", sodu)
if sodu == 0:
    print (n, " chia het cho ", m)
else:
    print (n, "khong chia het cho ", m)
print ("-----")
```

✓ Kết quả của thí dụ 2:

.....
.....
.....

Thí dụ 2: Các em hãy sử dụng câu lệnh if ... else đã học để kiểm tra một số tự nhiên và mô tả lại phát biểu của nhà toán học người Đức **Christian Goldbach** “*Mọi số tự nhiên chẵn lớn hơn 2 có thể biểu diễn bằng tổng của hai số nguyên tố*”. Tham khảo tiểu sử của nhà toán học tại trang https://en.wikipedia.org/wiki/Goldbach%27s_conjecture#

```
n = 8
sodu = n % 2
print ("sodu = ", sodu)
if n >= 2 and sodu == 0 :
    print (n, " co the bieu dien bang tong cua hai so nguyen to ")
else:
    print (n, "co the khong bieu dien bang tong hai so nguyen to")
print ("-----")
```

✓ Kết quả của thí dụ 2:

.....

Đố vui 3: (Không lập trình) Các em hãy biểu diễn các số dưới đây dưới dạng tổng của hai số nguyên tố.

6 =

10 =

100 =

Thư giãn:

```
from turtle import *
from random import randint
speed(0)
def wander():
    while True:
        fd(3)
        if xcor() >= 200 or xcor() <= -200 or ycor() <= -200 or ycor() >= 200:
            lt(randint(90,180))
wander()
```

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

3. Điều kiện if ... elif ... else

if Biểu thức điều kiện 1:

Câu lệnh 1

elif Biểu thức điều kiện 2:

Câu lệnh 2

...

elif Biểu thức điều kiện n:

Câu lệnh n

else:

Câu lệnh else

Thí dụ 3:

```
n = int(input('Nhập số nguyên n: '))
m = int(input('Nhập số nguyên m: '))
if m == 0:
    print("Không thể thực hiện phép toán m chia cho n")
elif n % m == 0:
    print(n, "chia hết cho ", m)
else:
    print(n, "không chia hết cho ", m)
print("-----")
```

✓ Các em hãy thực thi thí dụ 3, và ghi lại kết quả:

.....

.....

.....

4. Thực hành

Bài tập 1: Các em hãy thực hành các thí dụ 1, 2 và 3.

Bài tập 2: Qua ba bài thí dụ, các em hãy cho biết, bài thí dụ nào yêu cầu chúng ta phải nhập giá trị m và giá trị n, từ đó các em hãy rút ra kết luận muốn nhập một số từ bàn phím thì phải viết như thế nào?

.....

.....

.....

Bài 3: Các em vận dụng biểu thức điều kiện để thực hành lại **Bài tập 2 trang 23**, bổ sung thêm điều kiện if để kiểm tra n, nếu n dương ($n \geq 0$) tính giá trị căn bậc hai của n, ngược lại in ra màn hình thông báo “Khong the tinh duoc can bac hai cua n do n nho hon 0.”

.....

.....

.....

.....

.....

Bài 4: Các em hãy viết chương trình, nhập vào một số tự nhiên tùy ý **n** và tính số tự nhiên liền sau của **n**. Người ta nói trong hai số tự nhiên liên tiếp có một số chia hết cho 2. Hãy sử dụng các câu lệnh đã học để in ra màn hình số tự nhiên đó.

.....

.....

.....

.....

.....

.....

.....

.....

Bài 5: Các em hãy thực thi đoạn code sau và ghi lại kết quả thực hiện. Các em có thể thay đổi giá trị 100, 60 thành các số nguyên dương khác, thực thi và xem lại kết quả. Lưu ý, riêng đối với số 3 các em vẫn có thể thay đổi được nhưng không nên quá lớn (≤ 20 hợp lý) vì sẽ ảnh hưởng đến thời gian chạy của đoạn mã.

```
def triangle(sidelength=100):  
    for i in range(3):  
        forward(sidelength)  
        right(60)  
triangle()
```

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Bài 5: VÒNG LẶP (LOOP)

1. Vòng lặp while

Cấu trúc vòng lặp while

```
while biểu thức điều kiện:  
    khối lệnh
```

Ghi chú: Khối lệnh là một khối gồm một hay nhiều câu lệnh.

Thí dụ 1:

```
i = 0  
while i <= 5:  
    print (i)  
    i = i + 1  
print("-----")
```

✓ Giải thích:

Biểu thức điều kiện: $i \leq 5$.

Khối lệnh gồm hai câu lệnh.

Câu lệnh 1: `print (i)`, in giá trị của `i` ra màn hình.

Câu lệnh 2: `i = i + 1`, tăng giá trị của `i` lên 1 đơn vị

✓ Kết quả hiện thị:

```
i = 0  
while i <= 5:  
    print (i)  
    i = i + 1  
print ("-----")  
0  
1  
2  
3  
4  
5  
-----
```

Đố vui 4: Các em hãy cho biết, nếu $i = i + 1$ bị xóa bỏ, thì đoạn chương trình trên xảy ra hiện tượng gì, vì sao?

.....

Thí dụ 2:

```
n = 2
while n <= 10:
    if n % 2 != 0:
        print (n)
        n = n + 1
print("-----")
```

Ghi chú: phép toán “!=” nghĩa là phép toán \neq trong toán học.

✓ Kết quả của thí dụ 2:

.....

Gốc thư giản: Tìm nghiệm của phương trình $2x + 5 = 13$.

Về mặt toán học, để tìm nghiệm của phương trình này, thì ta có thể biến đổi tương đương bằng cách chuyển hằng số 5 về vế phải của phương trình, sau đó ta thực hiện phép toán chia hai vế cho 2, thì phương trình trên tương đương như sau:

$$\frac{2x}{2} = \frac{13 - 5}{2} \Rightarrow x = \frac{13 - 5}{2} = \frac{8}{2} = 4$$

Đây là bài toán tìm x sao cho khi x nhân cho 2 và cộng thêm cho 5 thì trả về giá trị là 13. Do đó, các em có thể đoán trước giá trị x có thể nằm trong một khoảng nào đó. Thí dụ x có thể trong khoảng giữa hai số -100 và 100. Khi đó các em có thể sử dụng vòng lặp while để tìm x như sau:

```
x = -100 # start at -100
while x < 100 : # go up to 100
    if 2 * x + 5 == 13: # if it makes the equation true
        print ("x = ", x) # print it out
        break
    x +=1 #make x go up by 1 to test the next number
```

Lưu ý: Dấu `==` là phép toán so sánh, trong bài này so sánh $2*x + 5$ có bằng 13 hay không? Kết quả của phép toán sẽ trả về True hay False.

✓ Kết quả:

.....

.....

.....

.....

.....

.....

.....

2. Vòng lặp for

Cấu trúc vòng lặp **for**

```
for biến_vòng_lặp in [dãy số]:
    khối lệnh
```

Ghi chú: Khối lệnh là một khối gồm một hay nhiều câu lệnh.

Thí dụ 3:

```
for i in [1, 2, 3, 5, 7, 11, 13, 17, 19]:  
    print (i)  
print("-----")
```

✓ Giải thích:

Biến_vòng_lặp: **i**.

Dãy số: **[1, 2, 3, 5, 7, 11, 13, 17, 19]**.

Câu lệnh 1: print (i), in giá trị của i ra màn hình.

✓ Kết quả hiện thị:

```
for i in [1, 2, 3, 5, 7, 11, 13, 17, 19]:  
    print (i)  
print("-----")
```

```
1  
2  
3  
5  
7  
11  
13  
17  
19  
-----
```

Thí dụ 4:

```
for i in range (1,5):  
    print (i)  
print("-----")
```

✓ Giải thích:

Biến_vòng_lặp: **i**.

Dãy số: **range (1,5) = [1,2,3,4,5]**

Câu lệnh 1: print (i), in giá trị của i ra màn hình.

✓ Kết quả hiện thị:




```
for i in range(1,5):
    print (i)
print ("-----")
```

```
1
2
3
4
-----
```

Ghi chú: Hàm **range** được sử dụng để tạo ra một danh sách chứa các dãy số.

Cú pháp:

```
range (diem_batdau, diem_ketthuc, [ buoc_nhay])
```

-  diem_batdau: giá trị khởi đầu
-  diem_ketthuc: giá trị giới hạn đạt tới (không bao gồm số này)
-  buoc_nhay: là số bước nhảy/tăng lên của một số so với số trước đó

Thí dụ 5:

```
range (4)          # [ 0,1,2,3]
range (4,7)        # [4,5,6]
range (3,20,2)     #[3,5,7,9,11,13,15,17,19 ]
```

Đố vui 5: Các em hãy thực thi chương trình này và vẽ lại kết quả nhìn thấy được.

```
from turtle import *
shape('turtle')
for i in range(4):
    forward(100)
    right(90)
```

.....

.....

.....

.....

Đôi vui 6: *Function là gì? các em hãy xem đoạn code sau, và chỉ ra đâu là function, và làm thế nào để sử dụng được function?*

```
from turtle import *  
shape('turtle')  
# Định nghĩa Function square  
def square():  
    for i in range(4):  
        forward(100)  
        right(90)  
# Sử dụng Function square  
square()
```

Lưu ý:

```
def square():
```

 khởi lệnh

Đây là cách viết một **function** trong Python, cú pháp tổng quát như sau:

```
def tenFunction(thamso neu co):  
    Khởi lệnh
```

Các em hãy thực thi và ghi lại kết quả

.....

.....

.....

.....

.....

.....

.....

.....

3. Vòng lặp for và mảng

Thí dụ 6:

```
#Khai báo một mảng
congty=["Samsung","Apple","Oppo","Nokia"]
for i in congty:
    print("Tên công ty",i)
```

✓ Các em hãy thực thi thí dụ 5, và ghi lại kết quả:

.....

.....

.....

.....

4. Câu lệnh break

break là một lệnh của python dùng để thoát khỏi vòng lặp khi vòng lặp vẫn còn đang thực hiện (hay chưa kết thúc).

Thí dụ 7:

```
#In các số từ 0 đến 7
for i in range(1,10):
    if i > 7:
        break
    print(i)
```

✓ Các em hãy thực thi thí dụ 7, và ghi lại kết quả:

.....

.....

.....

.....

5. Thực hành

Bài tập 1: Các em hãy thực hành các thí dụ có trong bài.

Bài tập 2: Đoạn chương trình dưới đây dùng để tính tổng của biểu thức $A = \sum_{n=0}^{20} n^2 + 1$

```
num = 20
running_sum = 0
for i in range(num+1):
    running_sum += i**2 + 1
print (running_sum)
```

.....

.....

.....

.....

.....

Bài tập 3: Số *nguyên tố* là số **CHỈ** có thể chia hết cho 1 và chính nó, các em hãy viết đoạn chương trình nhập vào số nguyên **n**, và in ra màn hình thông báo “Day la so nguyen to” hoặc “Day khong phai la so nguyen to”.

Gợi ý:

Suy luận từ câu nói: Chỉ có thể chia hết cho số 1 và cho chính nó, nghĩa là nó không thể chia hết cho các số khác.

Hiển nhiên nó không thể chia hết cho số lớn hơn nó. Thí dụ 8 là số lớn hơn 4, nên 4 không chia hết cho 8.

Từ đó, các em chỉ cần dò các số từ 2 cho đến con số nhỏ hơn **n**, nếu **TỒN TẠI** một số trong đó mà **n** chia hết, thì **n** không phải là số nguyên tố.

Lời giải chi tiết

```
n = int (input("n = "))
for i in range(2,n):
    if n % i == 0:
        print ("n không phải là SNT")
        break
else:
    print("n là số nguyên tố")
```

Các em xem đoạn chương trình trên, có thể điểm lại gì so với phần lý thuyết không?

.....

.....

.....

.....

.....

.....

.....

Bài tập 4: Cho hàm square như sau, các em hãy sử dụng hàm này, với tham số truyền vào cho square, thực thi và xem kết quả.

```
def square(sidelength):
    for i in range(4):
        forward(sidelength)
        right(90)
```

.....

.....

.....

.....

.....

Bài tập 5: Các em hãy viết chương trình in ra các số tự nhiên n vừa chia hết cho 2, vừa chia hết cho 5, biết $136 < n < 182$.

.....

.....

.....

.....

.....

.....

.....

.....

.....

Bài tập 6: Các em hãy viết chương trình nhập số nguyên n , tính $n!$

Gợi ý:

Gọi p là kết quả của phép tính $n!$, nghĩa là $p = 1 \times 2 \times 3 \times \dots \times n$

- Đặt $p = 1$.
- $\text{range}(2, n + 1)$ sẽ là các phần tử gồm: $2, 3, \dots, n$.
- Sử dụng vòng lặp `for` để lấy từng phần tử trong $\text{range}(2, n + 1)$ nhân vào p , ta sẽ có kết quả của $n!$

Bài 6: LIST

1. Tổng quan về danh sách trong Python

Danh sách (List) là một tập hợp các phần tử, các phần tử đó có thể thuộc các kiểu dữ liệu giống nhau hay khác nhau.

Cú pháp khai báo danh sách:

```
tenDanhSach = [ phần tử 1, phần tử 2, ..., phần tử n]
```

Minh họa:

```
songuyento = [2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59]
lop6TH2 = ['Quan', 'Phu', 'Khoi', 'Ky', 'Ngoc', 'Han' ]
taphoprong = [ ]
```

- **songuyento**: là danh sách (hay còn gọi là tập hợp) gồm có 17 số nguyên tố đầu tiên từ 2 đến 60.
- **lop6TH2**: là danh sách (hay còn gọi là tập hợp) gồm **len(lop6TH2)** học sinh. **len(danhsach)** là câu lệnh để tính số phần tử của danh sách đó.
- **taphoprong**: là danh sách (hay còn gọi là tập hợp), tập hợp này không có phần tử nào.

Lưu ý: Danh sách chứa 0 phần tử gọi là danh sách rỗng. Khai báo danh sách rỗng bằng cặp dấu ngoặc vuông []. Từ đây về sau khi ta có thể dùng tên gọi tập hợp thay cho khái niệm danh sách để gần với khái niệm **Tập hợp** quen thuộc trong toán học.

Thí dụ 1:

```
name_list = ['Abe', 'Bob', 'Chloe', 'Daphne']
for i, name in enumerate(name_list):
    print(name, "has index", i)
```

2. Truy cập phần tử trong danh sách

Cú pháp:

```
phantu_index = tenDanhSach[ index]
```

Trong đó:

- **index**: vị trí (số thứ tự) của phần tử trong danh sách/tập hợp. Vị trí của phần tử đầu tiên trong danh sách/tập hợp là 0, phần tử thứ hai là 1, ... Vị trí của phần tử thứ n của danh sách/tập hợp là $n - 1$.
- `phantu_index`: là một biến (variable) được dùng để lấy phần tử tương ứng của tập hợp đó.

Lưu ý:

- Cú pháp `taphop[index1: index2]` với `index1`, `index2` là vị trí bắt đầu và kết thúc, lấy các phần tử từ `index1` đến `index2`. Nếu `index1` bị khuyết thì sẽ lấy mặc định `index1` là 0, `index2` khuyết thì mặc định `index2` là chiều dài danh sách.
- Nếu `index` vượt lớn hơn số phần tử có trong tập hợp, chương trình Python sẽ đưa ra thông báo lỗi *IndexError*.

Thí dụ 2:

```
chiaHetchoBa = [3, 6, 9, 12, 15, 18, 21]
print(chiaHetchoBa [0]) #Kết quả: 3
print(chiaHetchoBa [1]) #Kết quả: 6
print(chiaHetchoBa [-1]) #Kết quả: 21
print(chiaHetchoBa[1:3]) #Kết quả: [6, 9]
print(chiaHetchoBa [:3]) #Kết quả: [3, 6, 9]
```

✓ Các em hãy thực thi thí dụ 2, và ghi lại kết quả:

.....
.....
.....

3. Một số phép toán của tập hợp

Toán tử “in”:

Cú pháp

```
phantu in TapHop/DanhSach
```

Ý nghĩa: Toán tử này sẽ kiểm tra phantu có thuộc (True) tập hợp/danh sách đó hay không (False) thuộc tập hợp/danh sách đó.

Thí dụ 3:

```
danhsachlop=['Nhu','Quoc', 'Khanh', 'Ngoc', 'Hieu','Thai', 'Quan','Phu', 'Ky' ]  
print('Hoc' in danhsachlop) # Kết quả: False  
print( 'Phu' in danhsachlop) # Kết quả: True
```

✓ Các em hãy thực thi thí dụ 3, và ghi lại kết quả:

.....
.....

Thí dụ 4: Áp dụng vòng lặp for trong danh sách

```
danhsachlop=['Nhu','Quoc', 'Khanh', 'Ngoc', 'Hieu','Thai', 'Quan','Phu', 'Ky' ]  
for hocsinh in danhsachlop:  
    print (hocsinh, end = ' ')
```

✓ Các em hãy thực thi thí dụ 4, và ghi lại kết quả:

.....

Các em hãy xóa chữ màu đỏ trong thí dụ 4 và hãy thực thi lại, ghi lại kết quả và ghi nhớ `end = ' '` trong câu lệnh print.

.....
.....

Toán tử “+”:**Cú pháp**

```
Danhsach1 + Danhsach2
```

Ý nghĩa: Toán tử này sẽ gộp hai Các phần tử của Danh sách 1 và các phần tử của Danh sách 2 lại thành một danh sách.

Thí dụ 5:

```
nhom1=['Nhu','Quoc','Khanh','Ngoc','Hieu']  
nhom2=['Hieu','Thai','Quan','Phu','Ky']  
danhsachlop = nom1 + nom2  
print(danhsachlop)
```

✓ Các em hãy thực thi thí dụ 5, và ghi lại kết quả:

.....
.....

Toán tử “*”:**Cú pháp**

```
Danhsach * songuyen Hoặc songuyen * Danhsach
```

Ý nghĩa: Toán tử này sẽ tạo ra một danh sách mới với số phần tử được nhân lên songuyen lần so với ban đầu.

Thí dụ 6: Mỗi một bịch gồm có một trái Táo (Apple), một trái Xoài (Mango) và một trái Lê (Pear), có tổng cộng 5 bịch giống nhau như vậy, hãy lập một tập hợp gồm 5 bịch này.

```
motbich=['Apple','Mango','Pear']  
taphop5Bich = traicay * 5  
print(taphop5Bich)
```

- ✓ Các em hãy thực thi thí dụ 6, và ghi lại kết quả:

.....

.....

.....

Gốc thư giản:

```
from turtle import *  
shape('turtle')  
def octagon():  
    for i in range(8):  
        backward(100)  
        left(45)  
octagon()
```

.....

.....

.....

.....

.....

.....

Toán tử “del”:

Cú pháp

```
del Danh sách[vitrixoa]
```

Ý nghĩa: Toán tử này sẽ xóa một phần tử ứng với **vitrixoa** ra khỏi danh sách.

```
del Danh sách[vitridau : vitricuoi]
```

Ý nghĩa: Toán tử này sẽ xóa các phần tử trong phạm vi từ **vitridau** đến **vitricuoi** ra khỏi danh sách.

Thí dụ 7: Có 5 đội bóng đá tham gia 1 giải đấu Ngoại hạng Anh (NHA). Thứ tự xếp hạng của các đội lần lượt như sau: “Leicester City”, “Tottenham Hotspur”, “Liverpool”, “Southampton”, “Chelsea”. Đội xếp cuối phải xuống hạng. Hãy xóa đội xếp cuối ra khỏi danh sách NHA.

```
NHA=['Leicester City', 'Tottenham Hotspur', 'Liverpool', 'Southampton', 'Chelsea']  
del NHA[4] #  
print(NHA)
```

✓ Các em hãy thực thi thí dụ 7, và ghi lại kết quả:

.....
.....

Lệnh “append”:

Cú pháp

```
DanhSach.append(phantucanthemvao)
```

Ý nghĩa: Lệnh append sẽ thêm “phantucanthemvao” vào danh sách.

Thí dụ 8: Phân tích số 30 thành các thừa số.

```
"""returns a list of the factors of num"""  
def factor (num):  
    factorList = []  
    for i in range(1,num+1):  
        if num % i == 0:  
            factorList.append(i)  
    return factorList  
factorList = factor(30)  
print (factorList)
```

- ✓ Các em hãy thực thi thí dụ 8, và ghi lại kết quả:

.....

.....

.....

.....

Lệnh “remove”:

Cú pháp

```
Danhsach.remove(phantucanxoa)
```

Ý nghĩa: Lệnh remove sẽ xóa “phantucanxoa” ra khỏi danh sách.

Thí dụ 9: Cho tập hợp gồm các phần tử sau: 'Apple', 'Mango', 'Python', 'Pear'. Hãy xóa phần tử không phù hợp ra khỏi tập hợp.

```
Fruits=['Apple','Mango', 'Python', 'Pear']  
Fruits.remove('Python')  
print(Fruits)
```

- ✓ Các em hãy thực thi thí dụ 9, và ghi lại kết quả:

.....

.....

Lệnh gán:

Cú pháp

```
Danhsach[phan tu can thay doi gia tri] = gia tri moi
```

Ý nghĩa: Lệnh gán này sẽ được dùng để cập nhật giá trị mới cho một phần tử trong danh sách.

Thí dụ 10: Một học sinh trả lời 5 số nguyên tố đầu tiên lần lượt là: 2, 3, 5, 9, 11. Đây là câu trả lời chưa chính xác ở vị trí có index là 3, vì 9 là số chia hết cho 3, học sinh này đã nhầm lẫn con số 7 với 9. Đoạn mã sau đây sẽ thay giá trị 7 vào vị trí có index 3.

```
namSoNguyenTo=[2, 3, 5, 9, 11]
namSoNguyenTo [3]=7
print(namSoNguyenTo)
```

✓ Các em hãy thực thi thí dụ 9, và ghi lại kết quả:

.....

.....

.....

.....

4. Thực hành

Bài tập 1: Cho tập hợp: $D = [4, 15, 20, 13, 23]$. Các em hãy viết chương trình in ra màn hình các phần tử của tập hợp D, tính tổng các phần tử trong tập hợp, và tính trung bình cộng các phần tử trong tập hợp (trung bình cộng được tính theo công thức: $tbc = (\text{tổng các phần tử}) / (\text{số lượng phần tử})$)

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Bài tập 2: Cho tập tập hợp $A = \{2 \times Y\}$ với Y là số tự nhiên chẵn. Gọi B là tập con chứa 10 phần tử đầu tiên của tập A . Hãy in ra B .

.....

.....

.....

.....

.....

.....

.....

.....

.....

Bài tập 3: Các em hãy viết chương trình nhập vào một số tự nhiên. Sau đó cho biết số đó có bao nhiêu ước số và in ra các ước số.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Bài tập 5:

Cho các hàm số sau: $f(x) = \sqrt{x+1} - x + 3$

Hãy tính giá trị của hàm số tại các điểm sau:

$$f(0), f(1), f(\sqrt{2}), f(\sqrt{2} - 1)$$

Gợi ý:

```
import numpy as np
def f(x):
    return np.sqrt(x + 1) - x + 3
#list of values to plug in
for x in [0, 1, np.sqrt(2), np.sqrt(2) - 1]:
    print("f({:.3f}) = {:.3f}".format(x, f(x)))
```

(The last line just makes the output pretty while rounding all the solutions to three decimal places)

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Bài 7: CÁC LỆNH CƠ BẢN CỦA TURTLE GRAPHICS

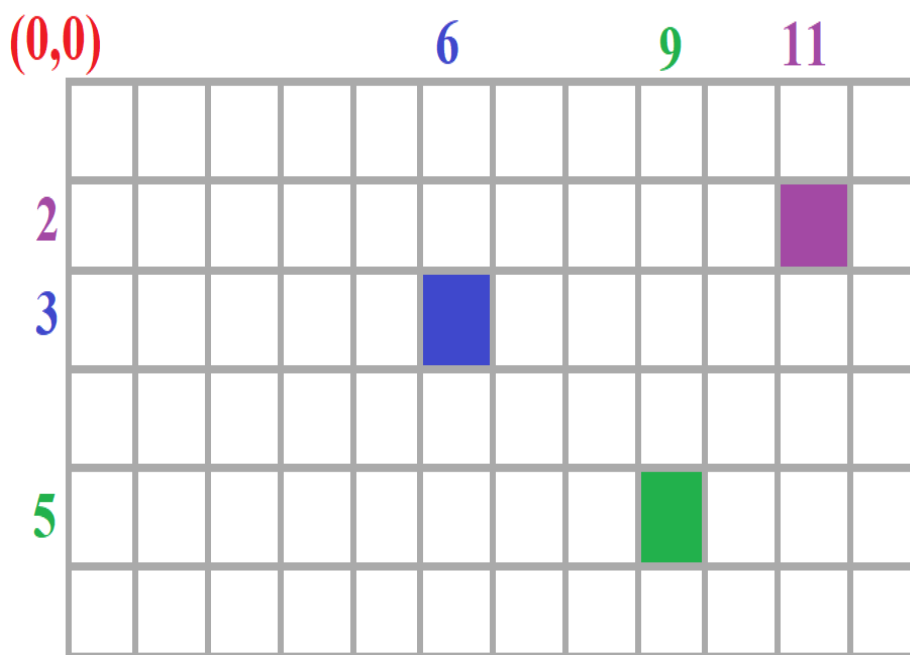
1. Tọa độ màn hình

Màn hình máy tính là một mặt phẳng hai chiều, trong toán học còn gọi là dạng của mặt phẳng Oxy. Đó là một mặt phẳng gồm có hai trục tọa độ, trục Ox, chiều nằm ngang và trục Oy, chiều thẳng đứng. Trong tin học, mặt phẳng Oxy thường gọi là mặt phẳng x-y.



Hình 17. Màn hình máy tính

Tọa độ màn hình là tọa độ mặt phẳng x-y với gốc tọa độ $(0, 0)$ được xác định. Thông thường, **gốc tọa độ màn hình** nằm ở vị trí phía trên bên trái của màn hình (các em xem hình minh họa bên dưới). Mỗi ô (pixel/cell) của tọa độ màn hình được xác định là phần giao vị trí của trục Ox và trục Oy. Thí dụ ô màu xanh dương có tọa độ là $(6, 3)$, ô màu xanh lá cây có tọa độ là $(9, 5)$, và tọa độ của ô màu tím là $(11, 2)$. Các tọa độ trong ví dụ như $(6, 3)$, $(9, 5)$, $(11, 2)$ còn gọi là các vị trí điểm ảnh (vị trí pixel). Trong lập trình đồ họa bằng ngôn ngữ C, Pascal,... gốc tọa độ màn hình được xác định theo quy ước này.



Hình 18. Xác định tọa độ màn hình máy tính trong lập trình đồ họa

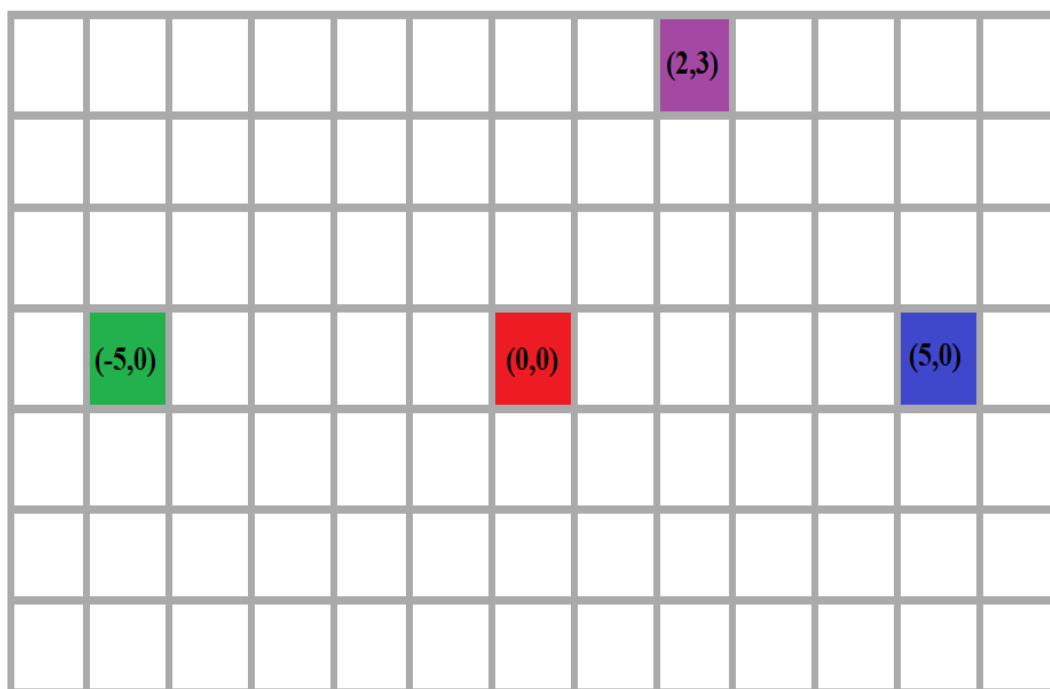
Tuy nhiên, trong thư viện **turtle** dưới đây các em sẽ được tìm hiểu, thì góc tọa độ màn hình được xác định ở **tâm của màn hình**. Điều này gần giống với gốc tọa trong trong mặt phẳng Oxy trong toán học, giúp các em dễ hình dung và xác định hơn trong lập trình.

Sự hiểu biết và nắm rõ về mặt phẳng tọa độ của màn hình sẽ giúp các em có thể định vị các vị trí tọa độ tốt hơn và từ đó giúp các em có thể lập trình để hiển thị các đối tượng đồ họa đúng các vị trí mà các em mong muốn.

2. Turtle graphics

Turtle graphics là một phần của ngôn ngữ lập trình Logo được phát triển bởi Wally Feurzeig, Seymour Papert và Cynthia Solomon năm 1967.

Các em hãy tưởng tượng con rùa robot (robotic turtle) bắt đầu ở vị trí (0, 0) trong tọa độ màn hình. Sau khi các em **import** thư viện **turtle** vào môi trường lập trình Python (Jupyter Notebook), các em có thể sử dụng lệnh **forward (5)**, khi đó turtle sẽ vẽ một mũi tên hướng đến vị trí điểm ảnh có tọa độ (5,0); hoặc **forward (-5)** turtle sẽ vẽ một mũi tên theo hướng ngược lại đến vị trí (-5,0).



Hình 19. Xác định vị trí tọa độ trong thư viện turtle

Cú pháp import thư viện turtle:

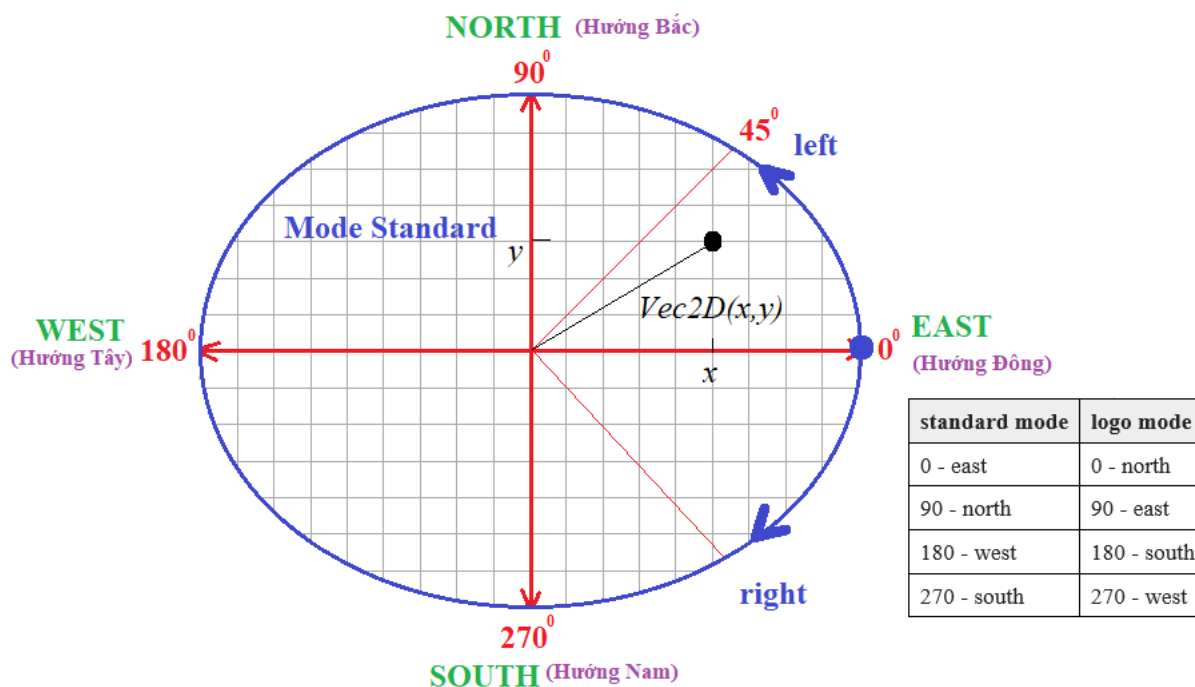
```
import turtle as t #import thư viện turtle và đặt tên ngắn gọn lại là t
```

Sau khi các em đã import thư viện **turtle** theo câu lệnh trên hoàn tất, khi đó, để sử dụng các câu lệnh có sẵn của thư viện, các em chỉ cần sử dụng tên ngắn gọn để thay thế turtle theo cú pháp: **t.caulenhTurtle (tham số)**. Các câu lệnh cơ bản của turtle sẽ được trình bày ở mục 3 của bài này.

3. Các lệnh cơ bản của turtle

a. Mode standard

Mode là một khái niệm trong thư viện của **turtle** dùng để xác định các hướng di chuyển của **turtle** trên mặt phẳng X-Y. **Turtle** có hai **mode** cơ bản đó là “**standard**” và “**logo**”. **Standard** là **mode** mặc định khi các em khai báo thư viện **turtle**, khi đó mũi tên của **turtle** sẽ chỉ sang *hướng Đông* (xem hình vẽ dưới đây).



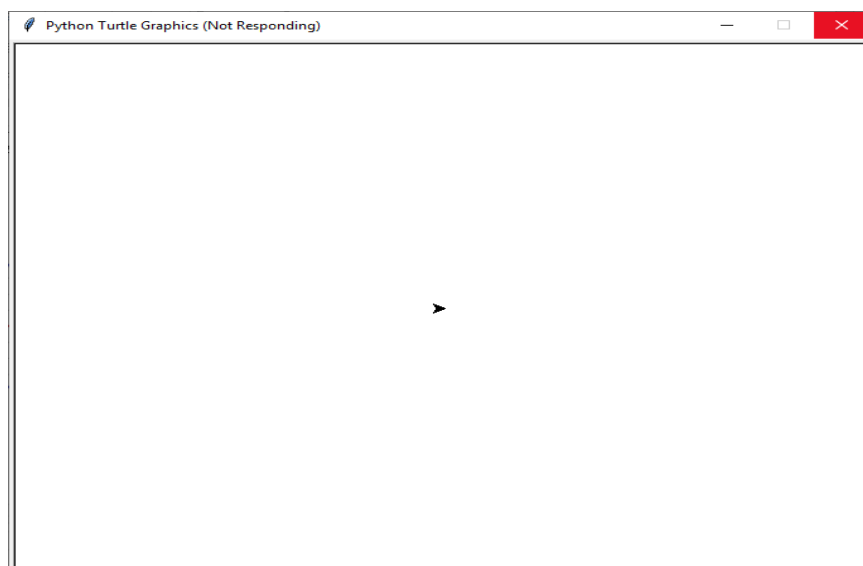
Hình 20. Mode Standard của thư viện turtle

Thí dụ 1:

```
In [1]: import turtle as t
```

```
In [2]: t.heading()
```

```
Out[2]: 0.0
```



Hình 21. Turtle đang chỉ đến hướng Đông

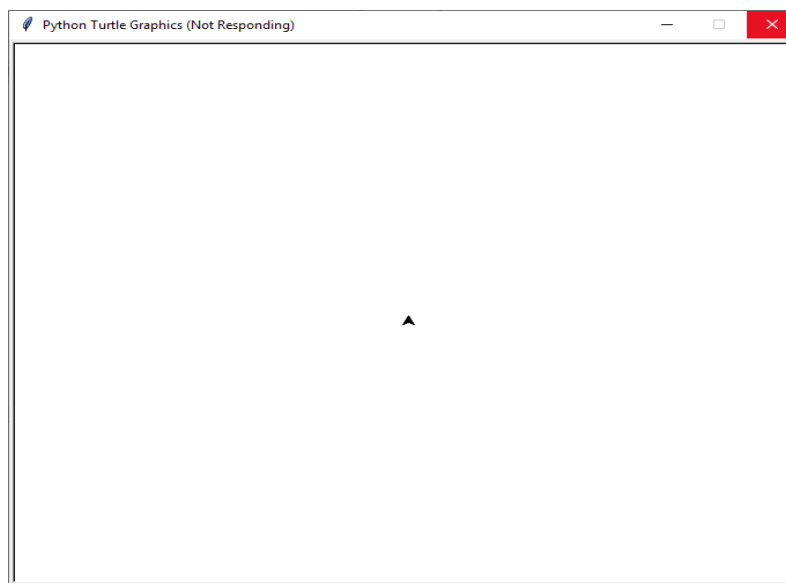
Giải thích:

- **Dòng 1:** Khai báo thư viện (hay còn gọi là nạp thư viện) vào chương trình Jupyter Notebook và đặt tên thư viện jupyter là **t** (các bạn có thể đặt một tên khác tùy ý).
- **Dòng 2: `t.heading()`** , câu lệnh này sẽ hiện thị hướng của turtle hiện tại. Như các bạn đã biết, hướng mặc định của turtle là hướng Đông, nên mũi tên chỉ hướng của turtle trên màn hình kết quả *Hình 21. Turtle đang chỉ đến hướng Đông*

Thí dụ 2:

```
In [1]: import turtle as t
```

```
In [2]: t.setheading(90)
```



Hình 22. Turtle đang chỉ đến hướng Bắc

Giải thích:

- **Dòng 1:** Tương tự như thí dụ 1.
- **Dòng 2: `t.setheading(90)`**, câu lệnh này sẽ chuyển hướng của turtle lên một góc 90^0 so với hướng hiện tại, thí dụ hướng hiện tại đang chỉ sang hướng Đông, thì `t.setheading(90)` sẽ chỉ hướng của turtle sang hướng Bắc (các em hãy xem mũi tên chỉ hướng trong hình 11). Các em có thể tùy chỉnh hướng của mũi tên theo góc tùy ý.

b. Turtle motion`t.position()`

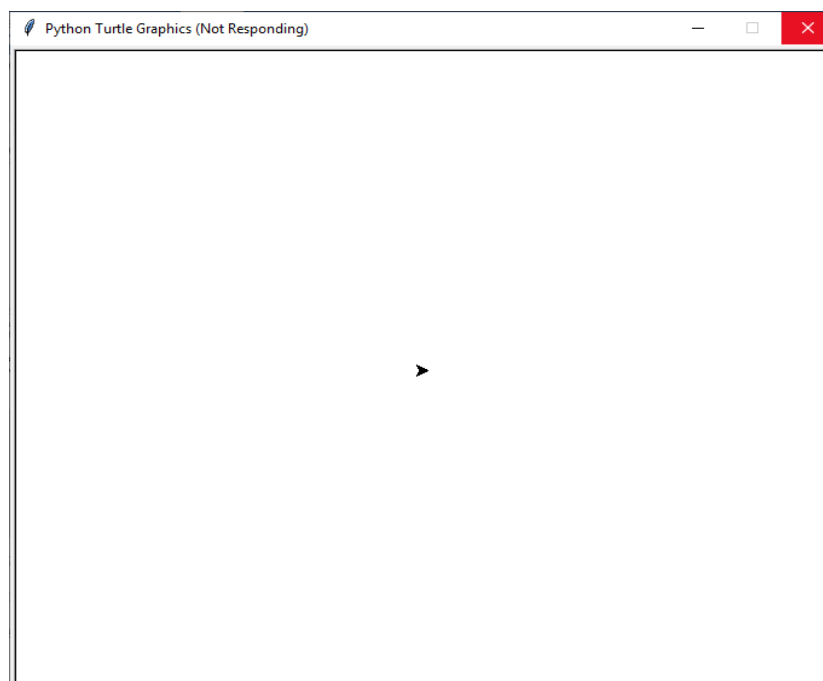
Ý nghĩa: Hiện thị tọa độ ở vị trí hiện tại.

Thí dụ 1:

```
In [1]: import turtle as t
```

```
In [2]: vitriTurtle = t.position()
        print(vitriTurtle)
```

```
(0.00,0.00)
```



Hình 23. Hướng hiện tại của turtle

Giải thích: Trong thí dụ này các em nhìn thấy rõ, dòng 1 là **import** thư viện **turtle** vào **Jupyter Notebook**, khi đó mặc định vị trí của **turtle** đang ở gốc tọa độ **(0,0)**. Ở dòng 2 là câu lệnh xác định vị trí của turtle, giá trị của vị trí này được gán vào cho biến **vitriTurtle**, hàm print (**vitriTurtle**) sẽ hiện thị giá trị của **vitriTurtle** hiện tại.

`t.forward(distance)` hoặc `t.fd(distance)`

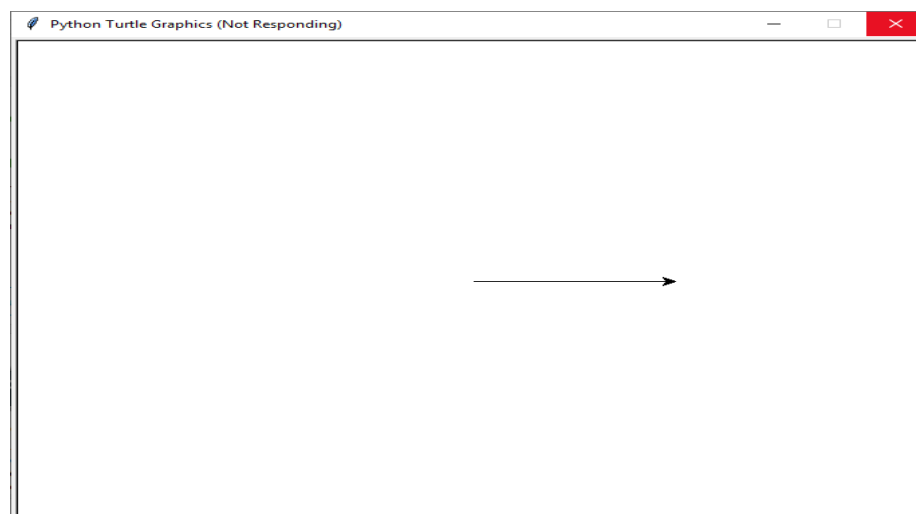
Ý nghĩa: Di chuyển **turtle** về phía trước theo khoảng cách (*distance*) xác định, theo hướng turtle đang hướng tới (thí dụ turtle đã chỉ đến hướng Đông thì mũi tên này sẽ chỉ theo hướng Đông)

Thí dụ 2:

```
In [1]: import turtle as t
In [2]: vitriTurtle = t.position()
        print(vitriTurtle)
        (0.00,0.00)
In [3]: t.forward(150)
In [4]: vitriTurtle = t.position()
        print(vitriTurtle)
        (150.00,0.00)
```

Giải thích:

1. Dòng 1 và dòng 2 của thí dụ 2 tương tự như thí dụ 1.
2. Dòng 3, di chuyển về phía trước theo khoảng cách 15 pixels, theo hướng của trục Ox.
3. Dòng 4, tiếp tục lấy vị trí của turtle như dòng 2. Tuy nhiên, vị trí của turtle lúc này đã khác so với dòng 2 trước đó, vì turtle đã được di chuyển 150 pixels, nên vị trí mới sẽ là (150, 0) như trong thí dụ trên.



Hình 24. Turtle vẽ một đường thẳng gồm 150 pixels

Đố vui 4: Các em hãy nhìn vào đoạn code dưới đây, và xem ở dòng số 6, sao khi lấy **vitriTurtle**, thì kết quả của hàm **print(vitriTurtle)** là bao nhiêu, giải thích?

```
In [1]: import turtle as t
```

```
In [2]: vitriTurtle = t.position()
print(vitriTurtle)

(0.00,0.00)
```

```
In [3]: t.forward(150)
```

```
In [4]: vitriTurtle = t.position()
print(vitriTurtle)

(150.00,0.00)
```

```
In [5]: t.forward(50)
```

```
In [6]: vitriTurtle = t.position()
print(vitriTurtle)
```

.....

.....

.....

t.backward(distance) hoặc t.bk(distance) hoặc t.back(distance)

Ý nghĩa: Di chuyển **turtle** về phía sau theo khoảng cách (*distance*), ngược lại với hướng turtle đang hướng tới.

Thí dụ 3:

```
In [1]: import turtle as t
```

```
In [2]: vitriTurtle = t.position()
print(vitriTurtle)

(0.00,0.00)
```

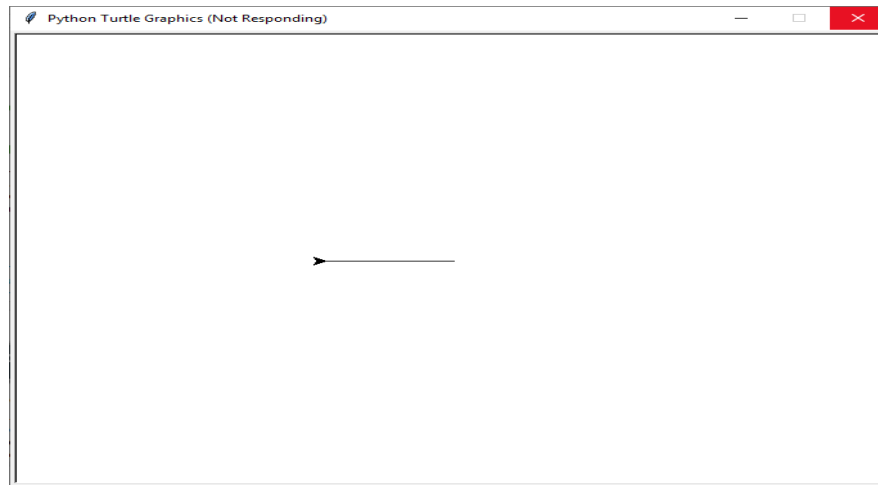
```
In [3]: t.backward(100)
```

```
In [4]: vitriTurtle = t.position()
print(vitriTurtle)

(-100.00,0.00)
```

Giải thích:

- Dòng 1 và dòng 2 của thí dụ 3 tương tự như thí dụ 1.
- Dòng 3, di chuyển về phía sau theo khoảng cách 100 pixels, theo hướng của trục Ox (xem kết quả hiển thị vị trí turtle trong Jupyter Notebook)



Hình 25. Turtle vẽ một đường thẳng gồm 100 pixels

4. Dòng 4, tiếp tục lấy vị trí của turtle như dòng 2, vì turtle đã được di chuyển ngược 100 pixels, nên vị trí mới sẽ là $(-100, 0)$ như trong thí dụ trên.

Đố vui: Các em hãy tiếp tục xem đoạn code dưới đây và giải thích vì sao ở dòng 6, giá trị hiển thị là $(-50,0)$.

```
In [1]: import turtle as t
In [2]: vitriTurtle = t.position()
        print(vitriTurtle)
        (0.00,0.00)
In [3]: t.backward(100)
In [4]: vitriTurtle = t.position()
        print(vitriTurtle)
        (-100.00,0.00)
In [5]: t.forward(50)
In [6]: vitriTurtle = t.position()
        print(vitriTurtle)
        (-50.00,0.00)
```

.....

.....

.....

`turtle.right(angle)` hoặc `turtle.rt(angle)`

Ý nghĩa: Quay **turtle** về bên phải một góc *angle*. Hướng của góc *angle* phụ thuộc vào *mode* của **turtle** (xem mode ở mục 3.1)

Thí dụ 4:

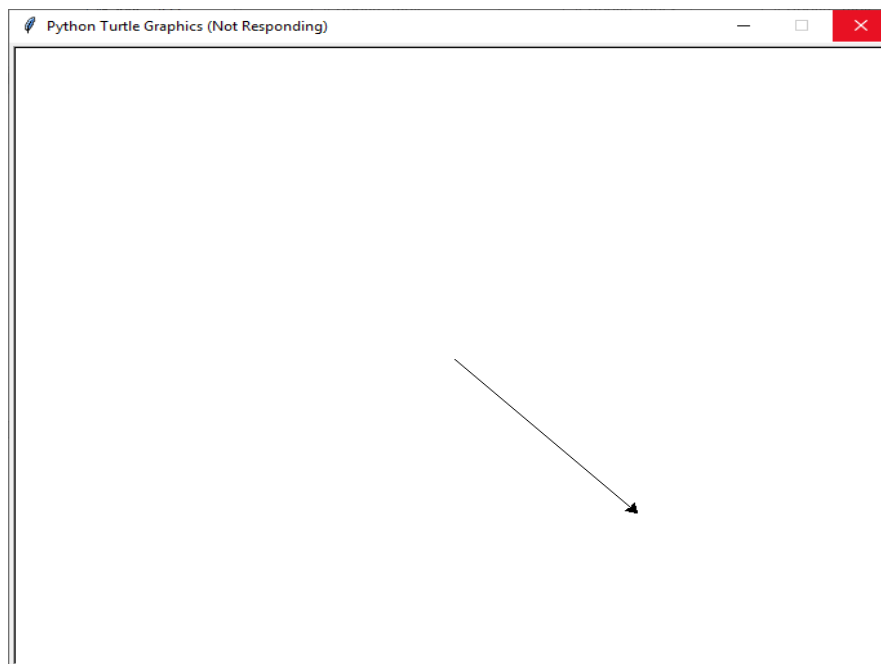
```
In [1]: import turtle as t
```

```
In [2]: t.right(45)
```

```
In [3]: t.forward(200)
```

Giải thích:

- **Dòng 2:** từ hướng hiện tại của turtle, quay turtle sang phải một góc 45 độ (các em xem lại Hình 20. Mode Standard của thư viện turtle).
- **Dòng 3:** dịch chuyển turtle theo hướng hiện tại 200 pixels (Các em xem kết quả hiện thì ở hình 14 dưới đây.



Hình 26 Kết quả của thí dụ 4 – Bài 7

`turtle.left(angle)` hoặc `turtle.lt(angle)`

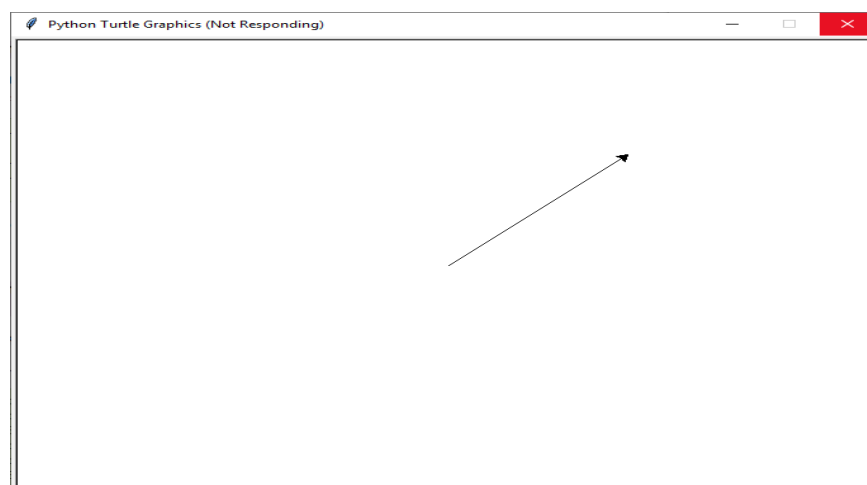
Ý nghĩa: Quay **turtle** về bên trái một góc *angle*. Hướng của góc *angle* phụ thuộc vào **mode** của **turtle** (xem mode ở mục 3.1)

Thí dụ 5:

```
In [1]: import turtle as t
```

```
In [2]: t.left(45)
```

```
In [3]: t.forward(200)
```



Hình 27 Kết quả của thí dụ 5 – Bài 7

Thí dụ 6: Các em hãy xem đoạn code dưới đây, sau đó các em hãy áp dụng **left** hoặc **right**, **forward** hoặc **backward** để có thể vẽ thêm một cạnh nữa để thành hình vuông.

```
In [1]: import turtle as t
```

```
In [2]: t.forward(200)
```

```
In [3]: t.left(90)
```

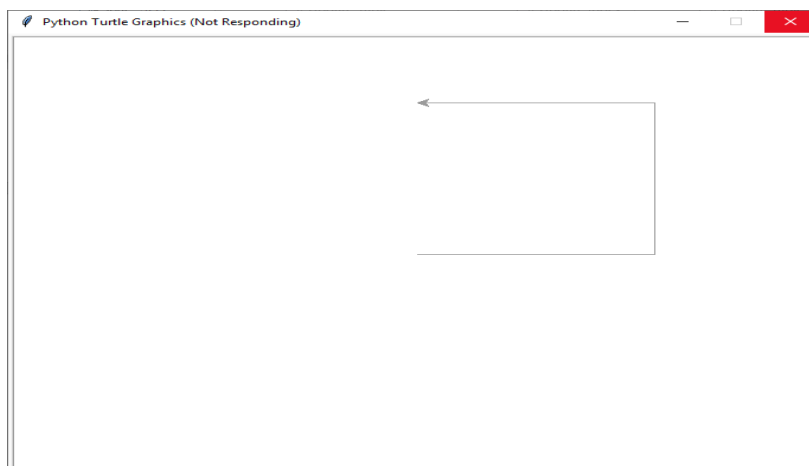
```
In [4]: t.forward(200)
```

```
In [5]: t.left(90)
```

```
In [6]: t.forward(200)
```

.....

.....



Hình 28 Kết quả của thí dụ 6 – Bài 7

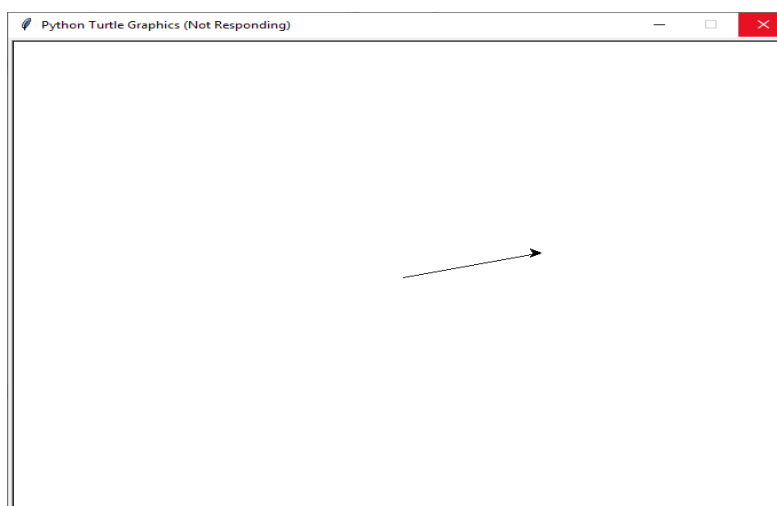
```
turtle.goto(turtle.vec2D(x, y))
```

Ý nghĩa: Di chuyển vị trí hiện tại của turtle sang một vị trí mới có tọa độ (x, y) trong mặt phẳng X-Y.

Thí dụ 7:

```
In [1]: import turtle as t
```

```
In [2]: t.goto(t.vec2D(120,30))
```



Hình 29 Kết quả của thí dụ 7 – Bài 7

Giải thích:

Từ tọa độ tâm của màn hình, turtle di chuyển x đến 120 pixels, đồng thời di chuyển y tăng lên 30 pixels (điểm A được minh họa ở Hình 20. *Mode Standard* của thư viện turtle mô phỏng cho thí dụ này).

```
turtle.setposition(turtle.Vec2D(x, y)) hoặc turtle.setpos(turtle.Vec2D(x, y))
```

Ý nghĩa: Tương tự như lệnh `turtle.goto(turtle.Vec2D(x,y))`.

```
turtle.setx(x)
```

Ý nghĩa: Đặt tọa độ đầu tiên của turtle thành x , giữ nguyên tọa độ thứ hai.

```
turtle.sety(y)
```

Ý nghĩa: Đặt tọa độ thứ hai của turtle thành y , giữ nguyên tọa độ đầu tiên.

```
turtle.home()
```

Ý nghĩa: Di chuyển turtle đến điểm gốc - tọa độ (0,0) - và đặt tiêu đề của nó về hướng bắt đầu (tùy thuộc vào mode standard hay logo).

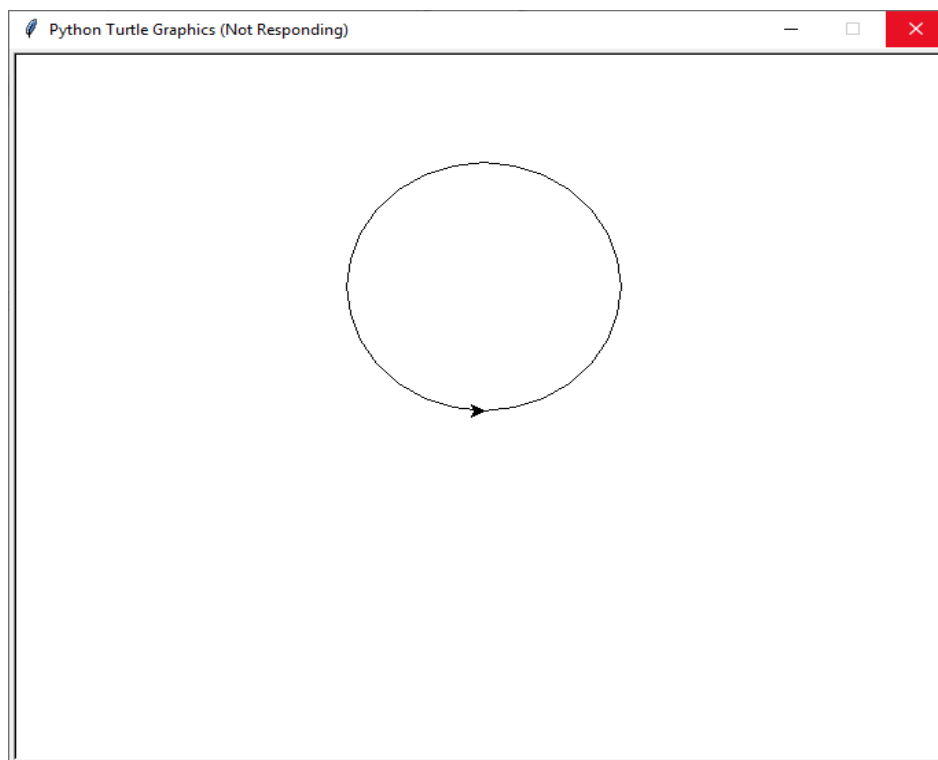
```
turtle.circle(radius, extent=None, steps=None)
```

Ý nghĩa: Vẽ một đường tròn với bán kính ($radius$) cho trước. Tâm là đơn vị bán kính bên trái của turtle. Nếu $radius$ dương, cung tròn được vẽ theo hướng ngược chiều kim đồng hồ, ngược lại theo chiều kim đồng hồ.

Thí dụ 8: Vẽ đường tròn bán kính 100.

```
In [1]: import turtle as t
```

```
In [2]: t.circle(100, extent=None, steps = None)
```



Hình 30 Kết quả của thí dụ 8 – Bài 7

Thí dụ 9: Các em hãy thay None ở extent và steps là những con số, thí dụ extent = 1200, steps = 10. Vẽ hình mà các em nhìn thấy.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

```
turtle.dot(size=None, *color)
```

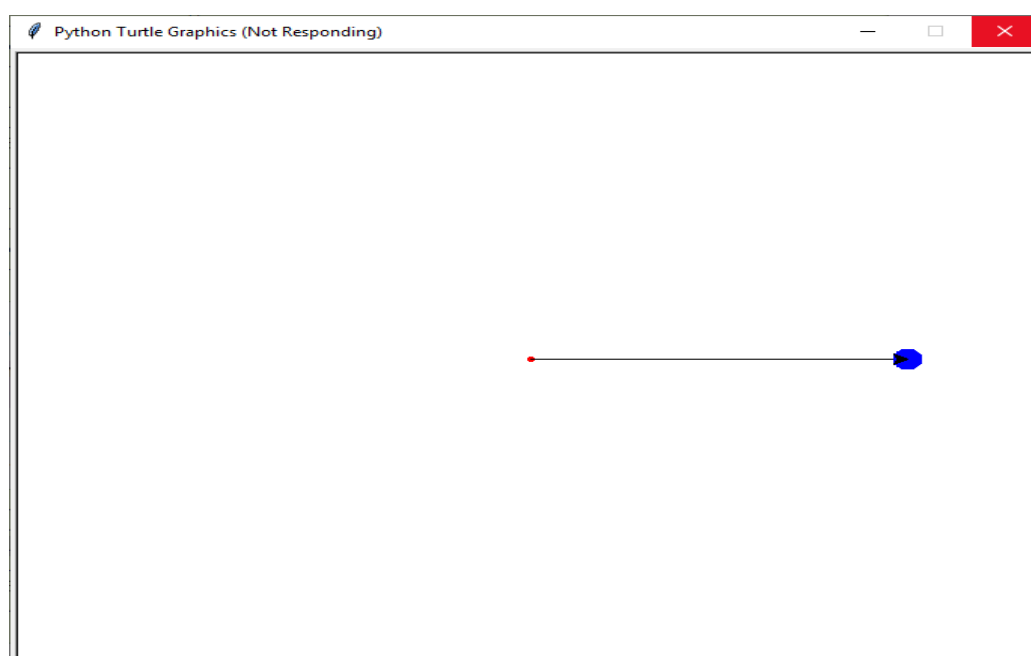
Ý nghĩa: Vẽ một chấm tròn với kích thước (size) được xác định là một số nguyên, trường hợp size = None thì kích thước mặc định.

Thí dụ 8: Sử dụng *dot* và *forward* (fd).

```
In [1]: import turtle as t
```

```
In [2]: t.dot(None, "red")
```

```
In [3]: t.fd(250)
         t.dot(20, "blue")
```



Hình 31 Kết quả của thí dụ 8 – Bài 7

Các em hãy xác định trên hình kết quả các đối tượng chấm tròn đỏ, đoạn thẳng và chấm tròn xanh dương là tương ứng các dòng nào trong thí dụ này

Tròn đỏ: Dòng số

Đoạn thẳng: Dòng số

Tròn xanh dương: Dòng số

```
turtle.undo()
```

Ý nghĩa: Undo (repeatedly) the last turtle action(s). Number of available undo actions is determined by the size of the undobuffer.

Thí dụ 9: Các em hãy thực thi đoạn chương trình dưới đây, quan sát kết quả thực hiện và diễn giải (giải thích từng dòng tương ứng với hình ảnh nhìn thấy).

```
In [1]: import turtle as t
```

```
In [2]: for i in range(4):
         t.fd(50); t.lt(80)
```

```
In [3]: for i in range(8):
         t.undo()
```

.....

.....

.....

.....

.....

.....

.....

turtle.speed(speed=None)

Ý nghĩa: Thiết lập tốc độ di chuyển của turtle từ nhanh nhất đến chậm nhất. speed là số nguyên trong phạm vi từ 0 (nhanh nhất) đến 10 (chậm nhất) hoặc chuỗi: “fastest”: 0, “fast”: 10, “normal”: 6, “slow”: 3, “slowest”: 1.

Attention: *speed = 0* means that *no* animation takes place. forward/back makes turtle jump and likewise left/right make the turtle turn instantly.

Thí dụ 10: Các em hãy quan sát thí dụ dưới đây, hãy thực thi và ghi lại các kết quả nhìn thấy, giải thích từng dòng lệnh tương ứng với kết quả có được.

```
In [1]: import turtle as t
```

```
In [2]: for i in range(10):
         t.fd(50)
         t.speed("slowest")
         t.lt(80)
```

.....

.....

.....

.....

4. Thực hành

Bài tập 1: Các em hãy sử dụng các câu lệnh đã học, hãy vẽ đường tròn có bán kính 200 pixels.

.....

.....

.....

.....

.....

Bài tập 2: Các em hãy sử dụng các câu lệnh đã học hãy vẽ hình chữ nhật kích thước 100 pixels x 200 pixels

.....

.....

.....

.....

.....

.....

Bài tập 3: Các em hãy sử dụng câu lệnh speed để điều chỉnh tốc độ vẽ của bài tập 2.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Bài 8: VẼ HÌNH ROBOT VỚI TURTLE GRAPHICS

1. Color control

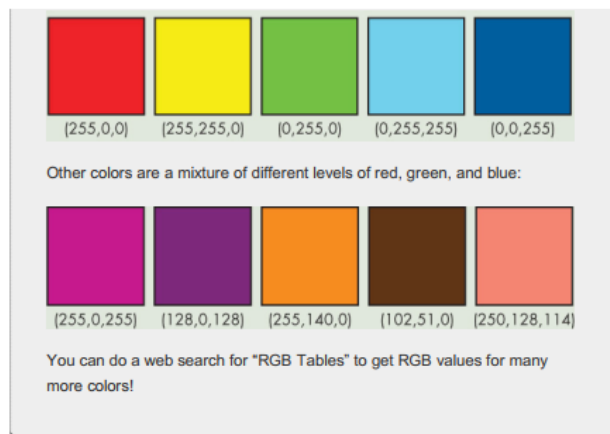
```
turtle.pencolor(*args)
```

pencolor()

Ý nghĩa: Trả lại màu viết (*pencolor*) hiện tại dưới dạng chuỗi đặc tả màu như “red”, “yellow” hoặc “#33cc8c”.

pencolor(colorstring), pencolor((r, g, b))

Ý nghĩa: Thiết lập giá trị cho màu viết (*pencolor*), *colorstring* là giá trị màu cần thiết lập, nó được mô tả dưới dạng chuỗi như “red”, “yellow” hoặc “#33cc8c”.



Hình 32 RGB Tables

```
turtle.fillcolor(*args)
```

fillcolor()

Ý nghĩa: tương tự như *pencolor()*, *fillcolor()* trả lại màu tô (*fillcolor*) hiện tại dưới dạng chuỗi như “red”, “yellow” hoặc “#33cc8c”.

fillcolor(colorstring)

Ý nghĩa: Thiết lập giá trị cho màu tô (*fillcolor*), *colorstring* là giá trị màu cần thiết lập, nó được mô tả dưới dạng chuỗi như “red”, “yellow” hoặc “#33cc8c”.

```
color(colorstring1, colorstring2)
```

Ý nghĩa: Tương đương với `pencolor(colorstring1)` và `fillcolor(colorstring2)`.

2. Filling

```
turtle.begin_fill()
```

Ý nghĩa: Chỉ được gọi trước khi một hình (shape) được tô màu.

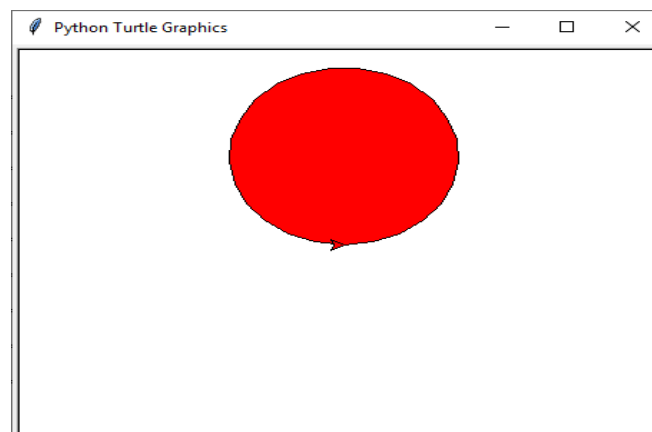
```
turtle.end_fill()
```

Ý nghĩa: Tô vào hình được vẽ sau lần cuối cùng gọi đến hàm `begin_fill()`.

Thí dụ 1:

```
import turtle as t
t.color("black", "red")
t.begin_fill()
t.circle(80)
t.end_fill()
```

✓ Kết quả của thí dụ 1:



Hình 33 Kết quả của thí dụ 1 – Bài 8

Giải thích:

.....

.....

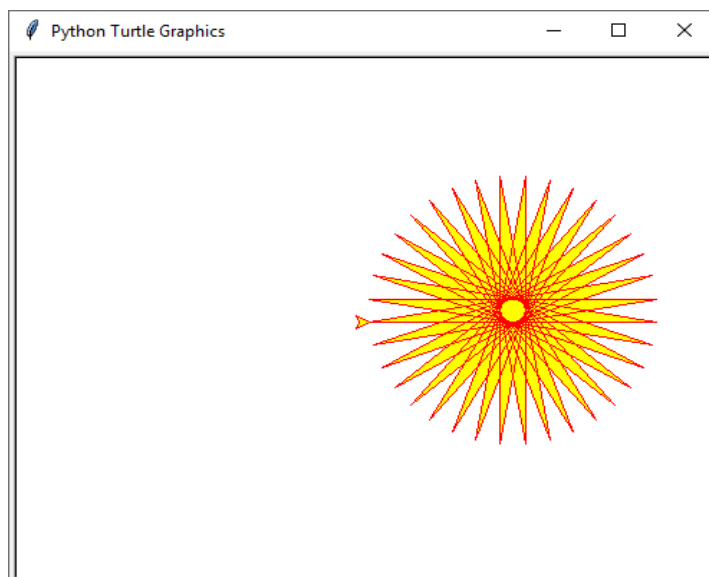
.....

.....

Thí dụ 2:

```
import turtle as t
t.color ('red', 'yellow')
t.begin_fill()
while True:
    t.forward(200)
    t.left(170)
    if abs(t.pos()) < 1:
        break
t.end_fill()
t.done()
```

✓ Kết quả của thí dụ 2:



Hình 34 Kết quả của thí dụ 2– Bài 8

Giải thích:

abs(number): Hàm lấy giá trị tuyệt đối của number. Ký hiệu toán học $|number|$. Thí dụ:

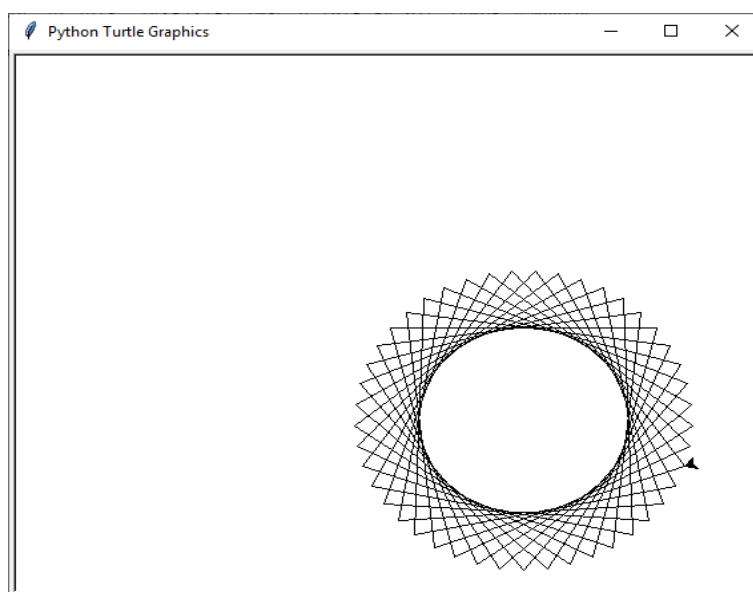
$$abs(5) = |5| = abs(-5) = |-5| = 5$$

.....
.....
.....
.....
.....

Thí dụ 3:

```
# Python program to draw star  
# using Turtle Programming  
import turtle as t  
for i in range(50):  
    t.forward(200)  
    t.right(104)  
t.done()
```

✓ Kết quả của thí dụ 3:



Hình 35 Kết quả của thí dụ 3 – Bài 8

Giải thích:

.....

.....

.....

.....

.....

.....

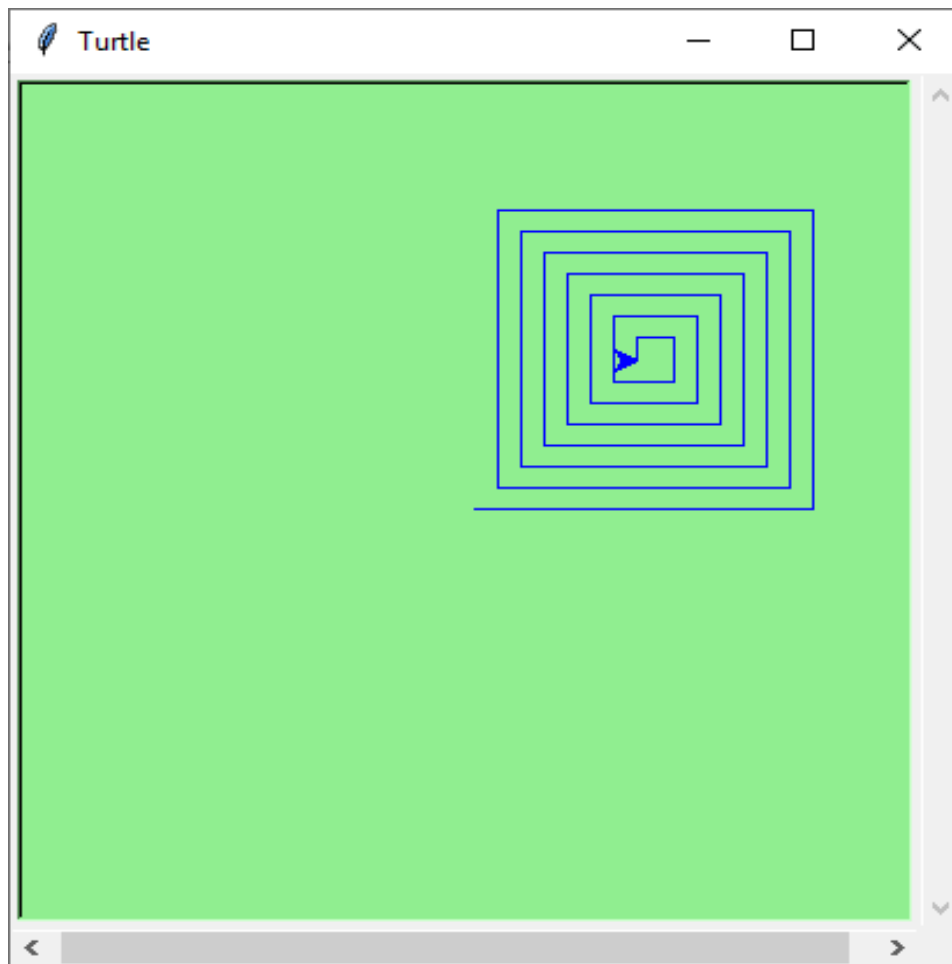
Thí dụ 4:

```
import turtle    #Outside_In
wn = turtle.Screen()
wn.bgcolor("light green")
wn.title("Turtle")
skk = turtle.Turtle()
skk.color("blue")

def sqrfunc(size):
    for i in range(4):
        skk.fd(size)
        skk.left(90)
        size = size-5

sqrfunc(146)
sqrfunc(126)
sqrfunc(106)
sqrfunc(86)
sqrfunc(66)
sqrfunc(46)
sqrfunc(26)
```

✓ Kết quả của thí dụ 4:



Hình 36 Kết quả của thí dụ 4 – Bài 8

Giải thích:

.....

.....

.....

.....

.....

.....

.....

.....

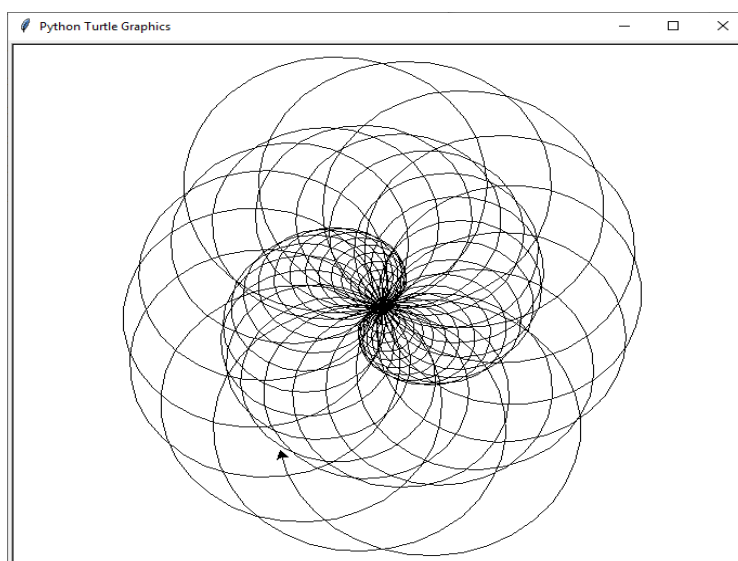
.....

Thí dụ 5:

```
import turtle
loadWindow = turtle.Screen()
turtle.speed(2)
for i in range(50):
    turtle.circle(5*i)
    turtle.circle(-5*i)
    turtle.left(i)

turtle.exitonclick()
```

✓ Kết quả của thí dụ 5:



Hình 37 Kết quả của thí dụ 5 – Bài 8

Giải thích:

.....

.....

.....

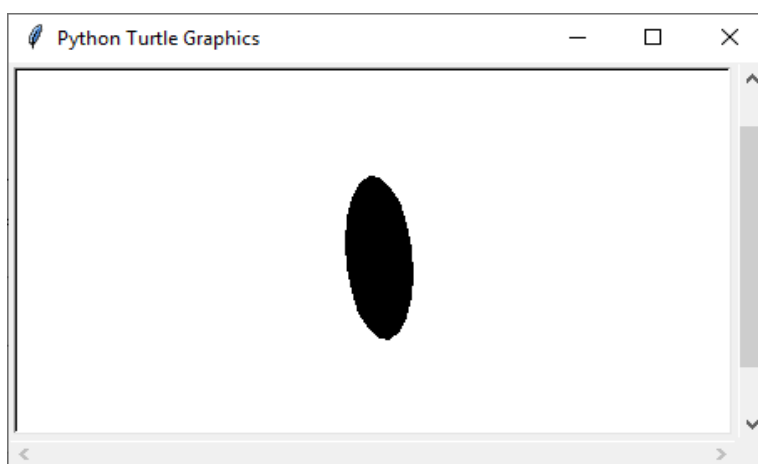
.....

.....

Thí dụ 6:

```
import turtle as t
t.shape("circle")
t.shapesize(5,2)
t.shearfactor(0.5)
t.shearfactor()
```

✓ Kết quả của thí dụ 6:



Hình 38 Kết quả của thí dụ 6 – Bài 8

Giải thích:

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

5. Thực hành

Bài tập 1: Các em hãy thực hành lại các thí dụ trong bài học.

.....

.....

.....

.....

.....

.....

Bài tập 2: Các em hãy thực hành lại thí dụ 2, trong đó hãy chọn một màu tùy thích khác để thay thế màu red và yellow.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Bài tập 3: Các em hãy bổ sung thêm các câu lệnh phù hợp để ô màu đường viền và màu nền cho hình của thí dụ 3.

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

[illegible]

PHỤ LỤC 1: CÁC CÂU LỆNH CƠ BẢN CỦA TURTLE

METHOD	PARAMETER	DESCRIPTION
Turtle()	None	Creates and returns a new turtle object
forward()	amount	Moves the turtle forward by the specified amount
backward()	amount	Moves the turtle backward by the specified amount
right()	angle	Turns the turtle clockwise
left()	angle	Turns the turtle counter clockwise
penup()	None	Picks up the turtle's Pen
pendown()	None	Puts down the turtle's Pen
up()	None	Picks up the turtle's Pen
down()	None	Puts down the turtle's Pen
color()	Color name	Changes the color of the turtle's pen
fillcolor()	Color name	Changes the color of the turtle will use to fill a polygon
heading()	None	Returns the current heading
position()	None	Returns the current position
goto()	x, y	Move the turtle to position x,y
begin_fill()	None	Remember the starting point for a filled polygon
end_fill()	None	Close the polygon and fill with the current fill color
dot()	None	Leave the dot at the current position
stamp()	None	Leaves an impression of a turtle shape at the current location
shape()	shapename	Should be 'arrow', 'classic', 'turtle' or 'circle'

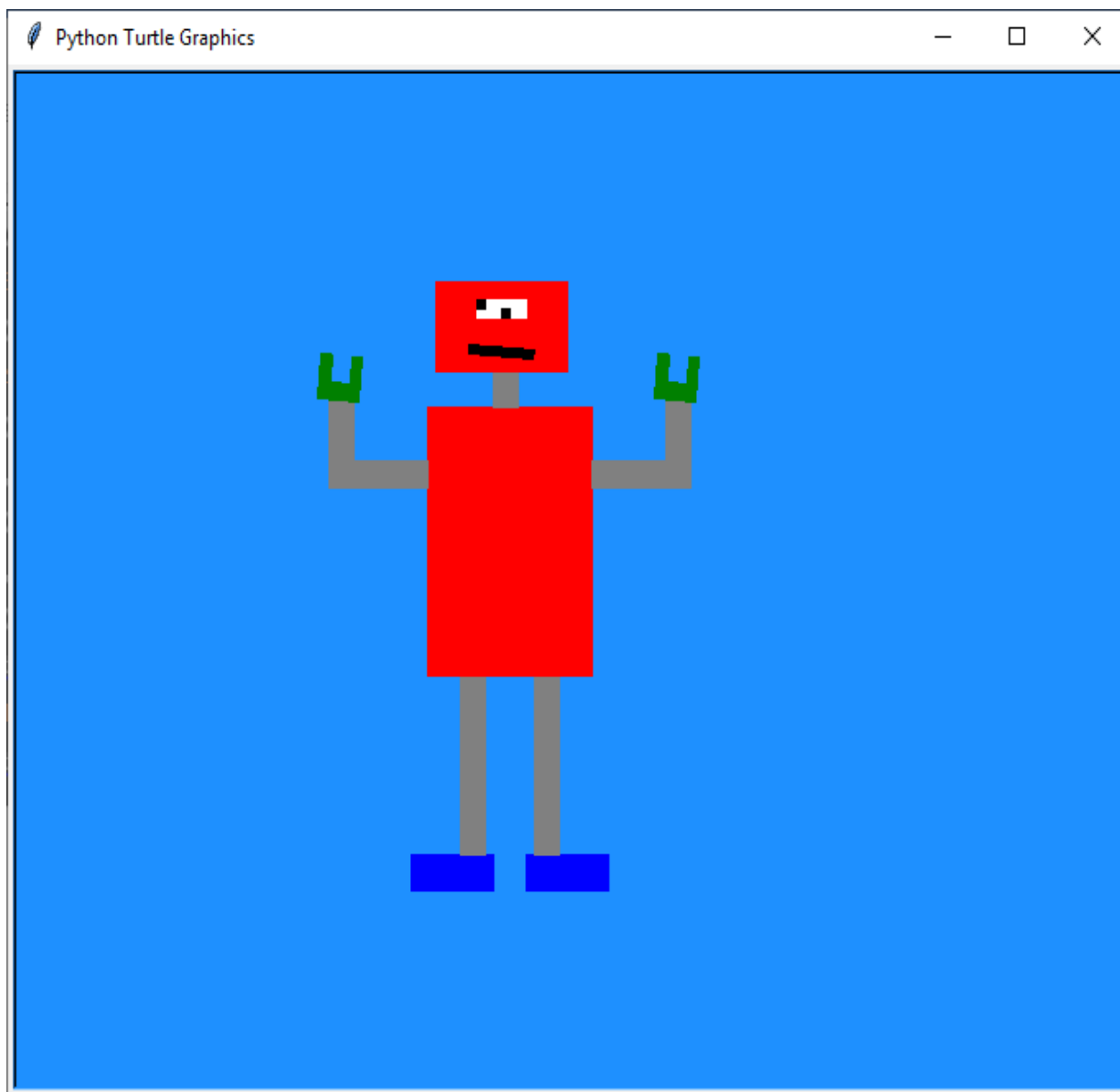
PHỤ LỤC 2: CÁC MẪU ĐOẠN CODE MINH HỌA

```
import turtle as t
import time as ti
#Định nghĩa ham rectangle:
def rectangle(hor,ver,col):
    t.pendown()
    t.pensize(1)
    t.color(col)
    t.begin_fill()
    for counter in range(1,3):
        t.forward(hor)
        t.right(90)
        t.forward(ver)
        t.right(90)
    t.end_fill()
    t.penup()
# Bat dau thuc hien cac thao tac ve
#0. -----
t.penup()
t.speed('slow')
#1. -----
t.bgcolor('Dodger blue')
#2. -----
t.goto(-100,-150)
rectangle(50,20,'blue')
#3. -----
t.goto(-30,-150)
rectangle(50,20,'blue')
```

```
#4. -----  
rectangle(10,15,t.bgcolor())  
t.goto(50,130)  
#5. -----  
rectangle(25,25,'green')  
t.goto(58,130)  
#5. -----  
t.goto(-25,-50)  
rectangle(15,100,'grey')  
t.goto(-55,-50)  
rectangle(-15,100,'grey')  
#6. -----  
t.goto(-90,100)  
rectangle(100,150,'red')  
#7. -----  
t.goto(-150,70)  
rectangle(60,15,'grey')  
#8. -----  
t.goto(-150,110)  
rectangle(15,40,'grey')  
#9. -----  
t.goto(10,70)  
rectangle(60,15,'grey')  
#10. -----  
t.goto(55,110)  
rectangle(15,40,'grey')  
#11. -----  
t.goto(-50,120)  
rectangle(15,20,'grey')
```

```
#12. -----  
t.goto(-85,170)  
rectangle(80,50,'red')  
#13. -----  
t.goto(-60,160)  
rectangle(30,10,'white')  
#14. -----  
t.goto(-60,160)  
rectangle(5,5,'black')  
#15. -----  
t.goto(-45,155)  
rectangle(5,5,'black')  
#16. -----  
t.goto(-65,135)  
t.right(5)  
rectangle(40,5,'black')  
#17. -----  
t.goto(-155,130)  
rectangle(25,25,'green')  
#18. -----  
t.goto(-147,130)  
rectangle(10,15,t.bgcolor())  
#19. -----  
t.goto(50,130)  
rectangle(25,25,'green')  
#20. -----  
t.goto(58,130)  
rectangle(10,15,t.bgcolor())  
#21. -----
```

```
t.hideturtle()  
ti.sleep(10)  
t.hideturtle()
```



Hình 39 Kết quả của phụ lục 2

TÀI LIỆU THAM KHẢO

1. Peter Farrell, *Math adventures with Python. An illutrated guide to exploring math with code*, No Starch press, Inc, 2019.
2. Tôn Thân, Vũ Hữu Bình, Phạm Gia Đức, Trần Luận, Phạm Đức Quang, *Bài tập toán 6 - tập 1*, NXB GDVN, 2011.

Các website

3. <https://docs.python.org/3/library/turtle.html>
4. <https://codelearn.io/sharing/ve-do-hoa-voi-turtle-graphic>
5. <https://codelearn.io/sharing/lap-trinh-game-flappybird-voi-python>
6. <https://www.geeksforgeeks.org/turtle-programming-python/>
7. https://www.tobiaskohn.ch/jython/turtlex_mario.html
8. https://www.tobiaskohn.ch/jython/turtlex_canon.html
9. <https://runestone.academy/runestone/books/published/thinkcspy/PythonTurtle/OurFirstTurtleProgram.html>
10. <https://runestone.academy/runestone/books/published/thinkcspy/PythonTurtle/OurFirstTurtleProgram.html>