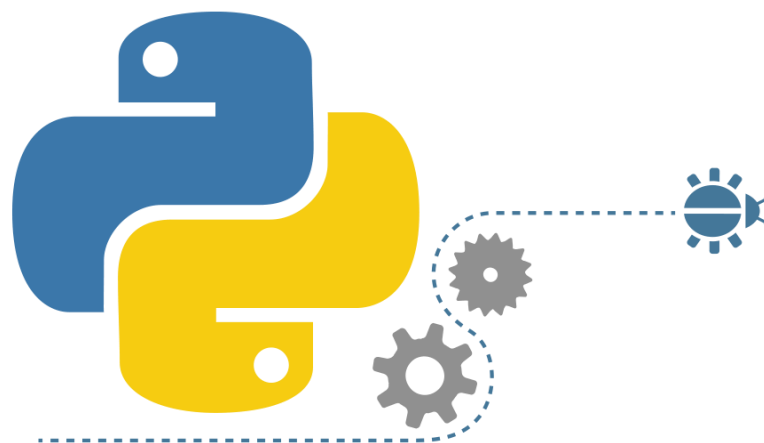


Khái niệm về Hàm



Nội dung bài học

- Hàm là một khối lệnh thực hiện một công việc hoàn chỉnh (module), được đặt tên và được gọi thực thi nhiều lần tại nhiều vị trí trong chương trình.
- Hàm còn gọi là chương trình con (*subroutine*)
- ***Nếu không viết hàm thì sẽ gặp những khó khăn gì?***
 - Rất khó để viết chính xác khi dự án lớn
 - Rất khó debug
 - Rất khó mở rộng

Nội dung bài học

- Có hai loại hàm:
 - *Hàm thư viện*: là những hàm đã được xây dựng sẵn. Muốn sử dụng các hàm thư viện phải khai báo thư viện chứa nó trong phần khai báo `from ... import`.
 - *Hàm do người dùng định nghĩa*.

Nội dung bài học

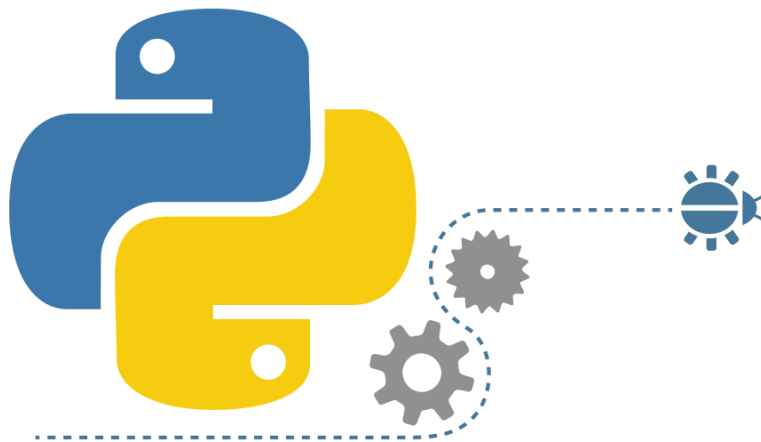
- Ví dụ hàm thư viện:

```
print("Chương trình tính điểm trung bình")
toan, ly, hoa = eval(input("Nhập điểm toán, lý, hóa: "))
print("Điểm toán=", toan)
print("Điểm lý=", ly)
print("Điểm hóa=", hoa)
dtb = (toan + ly + hoa) / 3
print("Điểm trung bình=", dtb)
print("Điểm làm tròn=", round(dtb, 2))
```

- Ví dụ hàm tự định nghĩa:

```
def cong(x, y):
    return x + y
```

Cấu trúc tổng quát của hàm



Nội dung bài học

- Python có cấu trúc tổng quát khi khai báo Hàm như sau:

```
def name ( parameter list ) :  
    block
```



- Dùng từ khóa **def** để định nghĩa hàm, các hàm có thể có đối số hoặc không. Có thể trả về kết quả hoặc không

Nội dung bài học

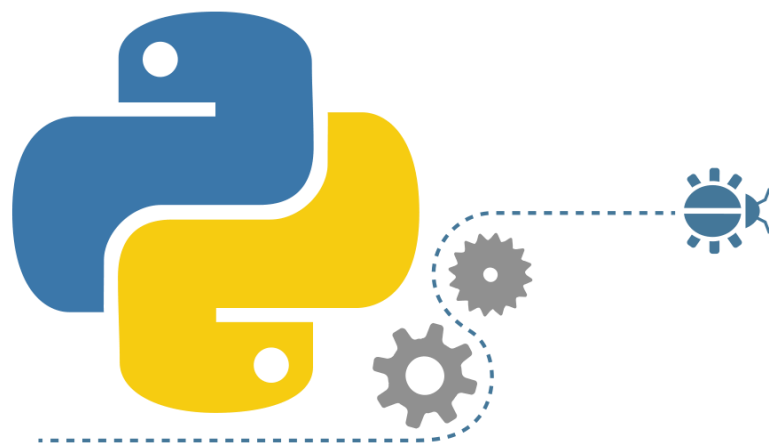
- Ví dụ : Viết hàm tính giải phương trình bậc 1

```
1  def PTB1(a,b):  
2      if a ==0 and b==0:  
3          return "Vô số nghiệm"  
4      elif a==0 and b!=0:  
5          return "Vô nghiệm"  
6      else:  
7          return "x={0}".format(round(-b/a,2))
```

- Ví dụ: Viết hàm xuất dữ liệu ra màn hình

```
 def XuatDuLieu(data):  
     print(data)
```

Cách gọi hàm



Nội dung bài học

- Để gọi hàm ta cũng cần phải kiểm tra Hàm đó được định nghĩa như thế nào?
 - ✓ Có đối số hay không?
 - ✓ Có trả về kết quả hay không?

Nếu có kết quả trả về:

Result=FunctionName ([parameter])

Nếu không có kết quả trả về:

FunctionName([parameter])

Nội dung bài học


- Ở bài học trước ta có ví dụ về phương trình bậc 1 và xuất dữ liệu

```
def PTB1(a,b):  
    if a ==0 and b==0:  
        return "Vô số nghiệm"  
    elif a==0 and b!=0:  
        return "Vô nghiệm"  
    else:  
        return "x={0}".format(round(-b/a,2))
```

- Hàm PTB1 vừa có đối số vừa có kết quả trả về. Ta có thể gọi như sau:
kq=PTB1(5,8)

Nội dung bài học

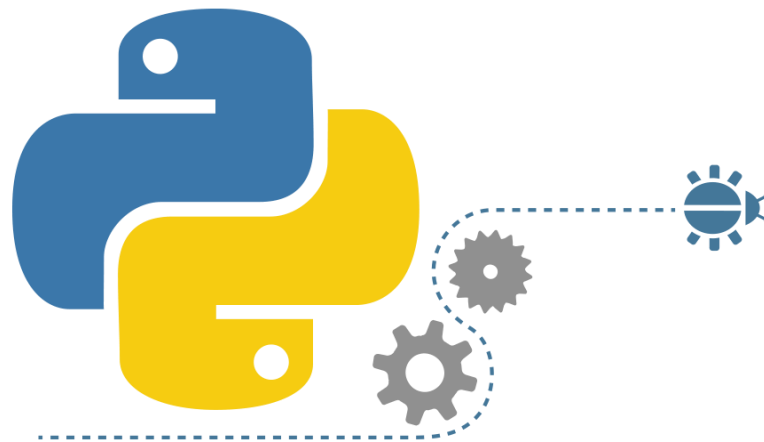
➤ Hàm xuất dữ liệu

```
 def XuatDuLieu(data):  
    print(data)
```

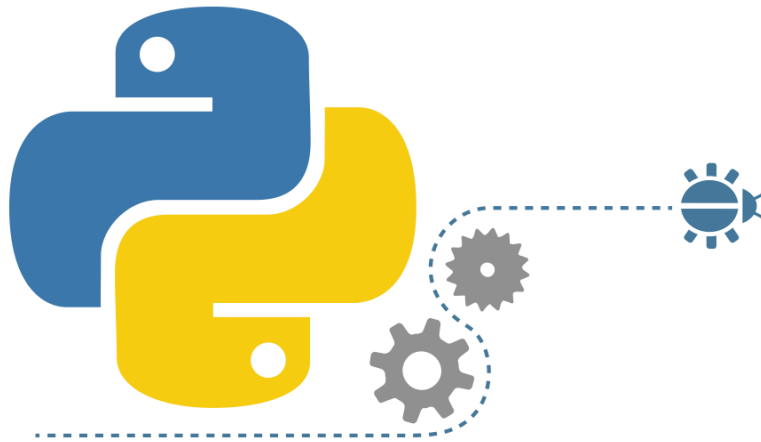
Ta có thể gọi:

XuatDuLieu("hello")

Nguyên tắc hoạt động của hàm



Global Variable



Nội dung bài học

- Tất cả các biến khai báo trong hàm chỉ có phạm vi ảnh hưởng trong hàm, các biến này gọi là biến local. Khi thoát khỏi hàm thì các biến này không thể truy xuất được.

➤ Xem ví dụ sau:

```
1 g=5
2 def increment():
3     g=2
4     g=g+1
5     increment()
6     print(g)
```

Global variable

Local variable

Global variable

- Theo bạn thì chạy xong 6 dòng lệnh ở trên, giá trị g xuất ra màn hình bao nhiêu?

Nội dung bài học

➤ Xem ví dụ 2 sau:

```
1 g=5
2 def increment():
3     global g
4     g=2
5     g=g+1
6     increment()
7     print(g)
```

- Theo bạn thì chạy xong 7 dòng lệnh ở trên, giá trị g xuất ra màn hình bao nhiêu?
- Global cho phép ta tham chiếu sử dụng được biến Global

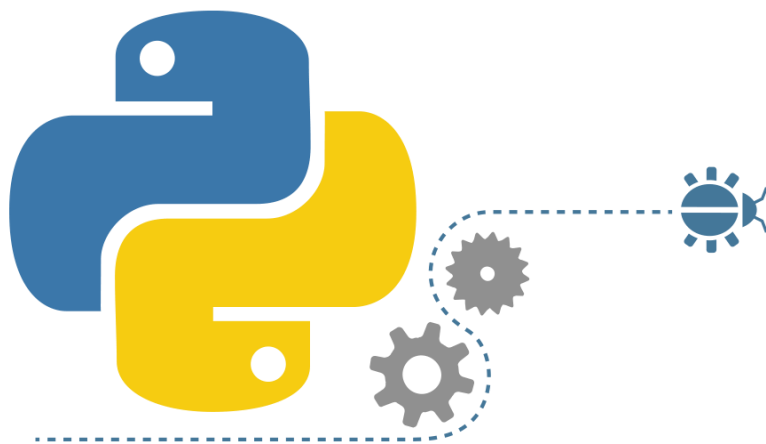
Nội dung bài học

- Xem ví dụ 3 sau:

```
1 g=5
2 def increment():
3     g=g+1
4     increment()
5     print(g)
```


- G dòng 3 Báo lỗi nha, vì g ở trong hàm không có lấy g ở ngoài (khai báo ở dòng 1)

Parameter mặc định



Nội dung bài học

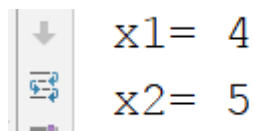
- Python cũng tương tự như C++, có hỗ trợ Parameter mặc định khi Khai báo hàm.
- Hàm print ta sử dụng cũng có các parameter mặc định

```
def print(self, *args, sep=' ', end='\n', file=None): #  
    """  
    print(value, ..., sep=' ', end='\n', file=sys.stdout
```

Nội dung bài học

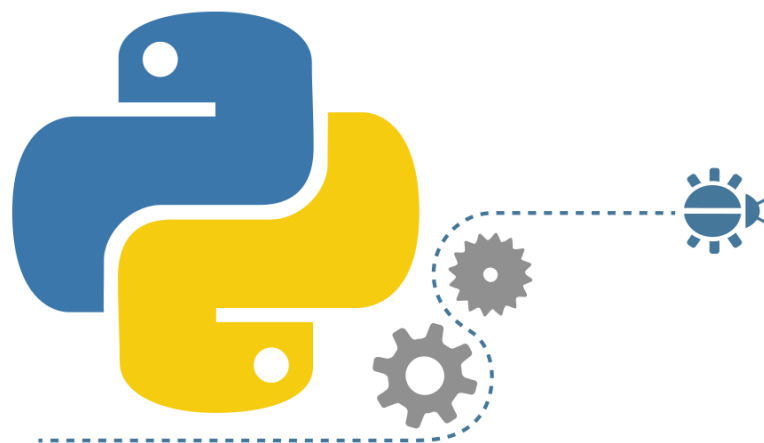
- Vậy nếu tự viết hàm thì ta sẽ định nghĩa các Parameter mặc định này như thế nào?

```
1  def SumRange (n, m=0) :  
2      sum=0  
3      for i in range (1, m+n, 1) :  
4          sum=i  
5      return sum  
6  
7  x1=SumRange (5)  
8  print ("x1=", x1)  
9  x2=SumRange (5, 1)  
10 print ("x2=", x2)
```



```
x1= 4  
x2= 5
```

Viết tài liệu cho hàm



Nội dung bài học

- Python hỗ trợ ta bổ sung tài liệu cho hàm, việc này rất thuận lợi cho đối tác sử dụng các function do ta làm ra. Dựa vào những tài liệu này mà Partner có thể dễ dàng biết cách sử dụng.
- Ta có thể sử dụng 3 dấu nháy kép hoặc 3 dấu nháy đơn để viết tài liệu cho hàm. Tuy nhiên theo kinh nghiệm thì các bạn nên dùng 3 dấu nháy kép.
- Các ghi chú (tài liệu) phải được viết ở những dòng đầu tiên khi khai báo hàm

Nội dung bài học

Ví dụ ta tạo 1 file

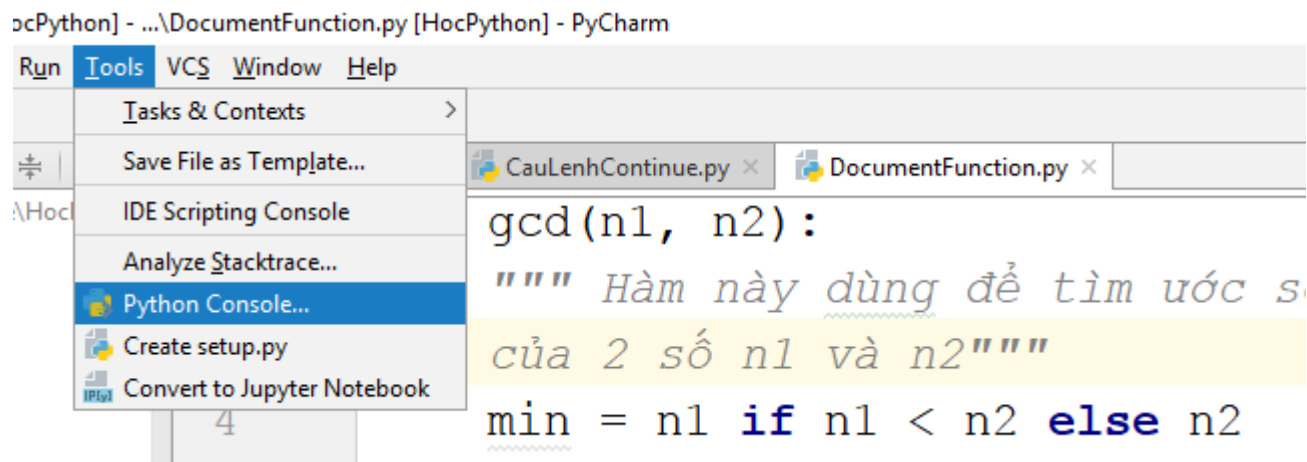
DocumentFunction.py

Có 2 hàm **gcd** và **ptb1**

```
1 def gcd(n1, n2):
2     """ Hàm này dùng để tìm ước số chung lớn nhất
3     của 2 số n1 và n2 """
4     min = n1 if n1 < n2 else n2
5     largest_factor = 1
6     for i in range(1, min + 1):
7         if n1 % i == 0 and n2 % i == 0:
8             largest_factor = i
9     return largest_factor
10 def ptb1(a,b):
11     """Giải phương trình bậc 1
12     ax+b=0"""
13     if a == 0 and b == 0:
14         return "Vô số nghiệm"
15     elif a == 0 and b != 0:
16         return "Vô nghiệm"
17     else:
18         return "x={0}".format(round(-b/a, 2))
```

Nội dung bài học

Ta chạy command line để xem cách lấy tài liệu cho các hàm trên. Ta vào menu tools/chọn Python Console...

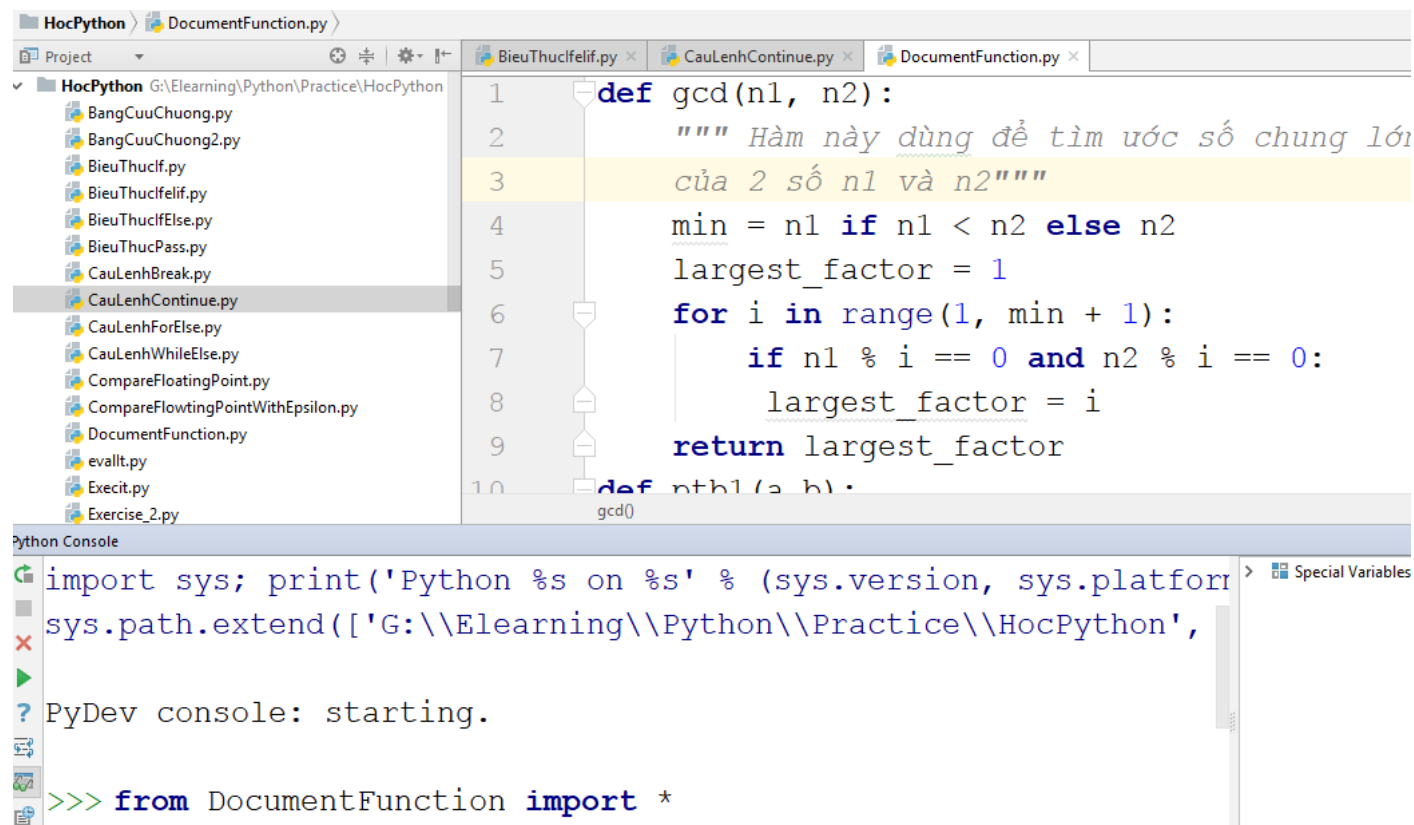


```
Python] - ...\DocumentFunction.py [HocPython] - PyCharm
Run Tools VCS Window Help
Tasks & Contexts >
Save File as Template...
IDE Scripting Console
Analyze Stacktrace...
Python Console...
Create setup.py
Convert to Jupyter Notebook

gcd(n1, n2):
    """ Hàm này dùng để tìm ước số
    của 2 số n1 và n2 """
    min = n1 if n1 < n2 else n2
```

Nội dung bài học

Ta chạy command line để xem cách lấy tài liệu cho các hàm trên. Ta vào menu tools/chọn Python Console...



The screenshot shows an IDE window with a project named 'HocPython'. The file explorer on the left lists several Python files, with 'CauLenhContinue.py' selected. The main editor displays the code for 'gcd(n1, n2)' in 'DocumentFunction.py'. The code is as follows:

```
1 def gcd(n1, n2):
2     """ Hàm này dùng để tìm ước số chung lớn
3     của 2 số n1 và n2 """
4     min = n1 if n1 < n2 else n2
5     largest_factor = 1
6     for i in range(1, min + 1):
7         if n1 % i == 0 and n2 % i == 0:
8             largest_factor = i
9     return largest_factor
10 def nth1(a, b):
    gcd()
```

Below the editor is the 'Python Console' window. It shows the following commands and output:

```
import sys; print('Python %s on %s' % (sys.version, sys.platform))
sys.path.extend(['G:\\Elearning\\Python\\Practice\\HocPython',
PyDev console: starting.
>>> from DocumentFunction import *
```

from DocumentFunction import *

Nội dung bài học

Muốn xem tài liệu của hàm nào thì gõ : **help**(function name)

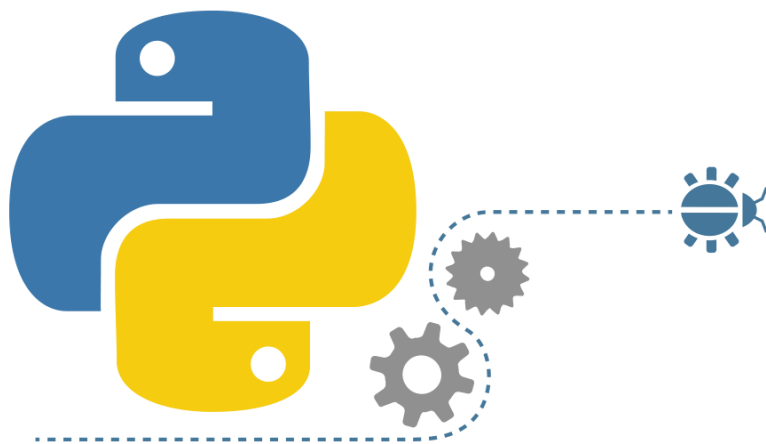
```
>>> from DocumentFunction import *
>>> help(gcd)
Help on function gcd in module DocumentFunction:

gcd(n1, n2)
    Hàm này dùng để tìm ước số chung lớn nhất
    của 2 số n1 và n2
```

```
>>> help(ptb1)
Help on function ptb1 in module DocumentFunction:

ptb1(a, b)
    Giải phương trình bậc 1
    ax+b=0
```

Giới thiệu về hàm đệ qui



Nội dung bài học

- Đệ qui là cách mà hàm tự gọi lại chính nó, trong nhiều trường hợp đệ qui giúp ta giải quyết những bài toán học búa và theo “tự nhiên”. Tuy nhiên đệ qui nếu xử lý không khéo sẽ bị tràn bộ đệm. Thông thường ta nên cố gắng giải quyết bài toán đệ qui bằng các vòng lặp, khi nào không thể giải quyết bằng vòng lặp thì mới nghĩ tới đệ qui. Do đó có những bài toán Khử đệ qui thì ta nên nghĩ về các vòng lặp để khử.
- Một vài ví dụ kinh điển về đệ qui như tính giai thừa, tính dãy số Fibonacci.
- $N! = N * (N-1)!$ ➔ Đệ qui: Nếu biết được $(N-1)!$ Thì sẽ tính được $N!$
- $F_1=1, F_2=1, F_N=F_{N-1}+F_{N-2}$

Nội dung bài học

- Khi quyết định giải quyết bài toán theo Đề Qui thì điều quan trọng phải nghĩ tới đó là: **Điểm dừng của bài toán là gì? + Biết được qui luật thực hiện của bài toán?** Nếu không tìm ra được 2 dấu hiệu này thì không thể giải quyết bài toán bằng cách dùng đệ qui.

Nội dung bài học

- Ví dụ ta thử phân tích bài giai thừa theo cách đệ qui?

$$n! = n \cdot (n-1) \cdot (n-2) \cdot (n-3) \cdots 3 \cdot 2 \cdot 1$$

- Viết lại dạng phương trình Đệ Qui có điều kiện:

$$n! = \begin{cases} 1, & \text{if } n = 0 \\ n \cdot (n-1)!, & \end{cases}$$

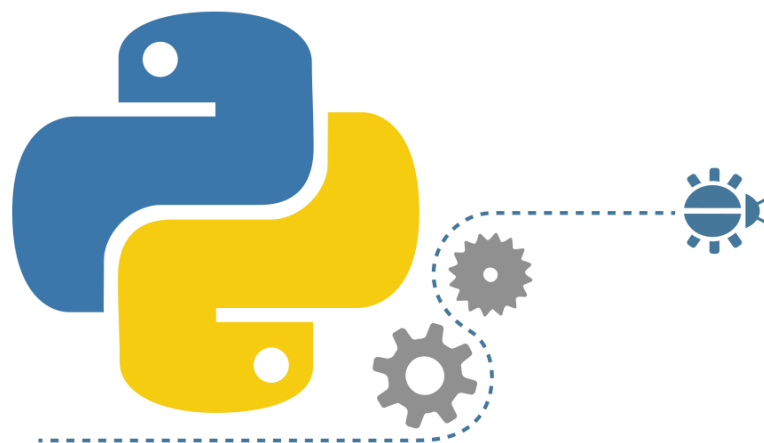
- Điểm dừng là khi $n=0$, quy luật là nếu biết $(n-1)!$ Thì tính được $N!$, vì $N! = N \cdot (N-1)!$

Nội dung bài học

```
def factorial(n):  
    """  
    Hàm tính n!  
    Trả về giai thừa của n  
    """  
    if n == 0:  
        return 1  
    else:  
        return n * factorial(n - 1)  
  
kq=factorial(6)  
print(kq)
```

```
factorial(6) = 6 * factorial(5)  
             = 6 * 5 * factorial(4)  
             = 6 * 5 * 4 * factorial(3)  
             = 6 * 5 * 4 * 3 * factorial(2)  
             = 6 * 5 * 4 * 3 * 2 * factorial(1)  
             = 6 * 5 * 4 * 3 * 2 * 1 * factorial(0)  
             = 6 * 5 * 4 * 3 * 2 * 1 * 1  
             = 6 * 5 * 4 * 3 * 2 * 1  
             = 6 * 5 * 4 * 3 * 2  
             = 6 * 5 * 4 * 6  
             = 6 * 5 * 24  
             = 6 * 120  
             = 720
```

Viết hàm tính BMI



Nội dung bài học

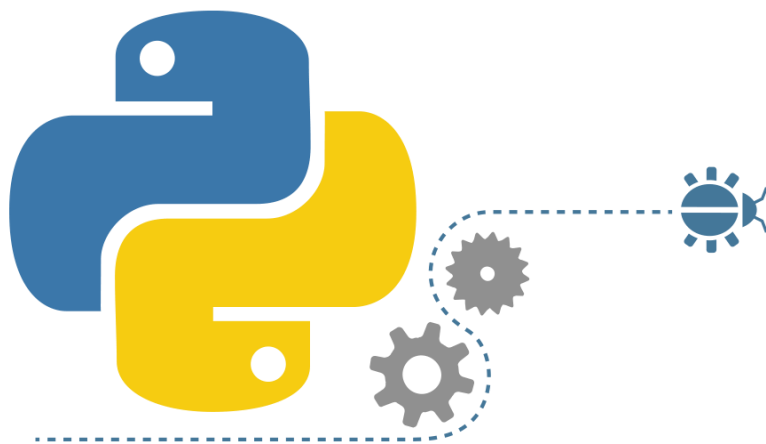
- Gọi BMI là chỉ số cân đối cơ thể. Yêu cầu đầu vào nhập là chiều cao và cân nặng, hãy cho biết người này như thế nào, biết rằng:

$$\text{BMI} = \frac{\text{Cân nặng (kg)}}{\text{Chiều cao} \times \text{chiều cao (m)}}$$

- Hãy thông báo phân loại
Và cảnh báo nguy cơ cho họ

CHỈ SỐ KHỐI CƠ THỂ	PHÂN LOẠI	NGUY CƠ PHÁT TRIỂN BỆNH
< 18.5	Gầy	Thấp
18.5 - 24.9	Bình thường	Trung bình
25.0 - 29.9	Hơi béo	Cao
30.0 - 34.9	Béo phì cấp độ 1	Cao
35.0 - 39.9	Béo phì cấp độ 2	Rất cao
> 40.0	Béo phì cấp độ 3	Nguy hiểm

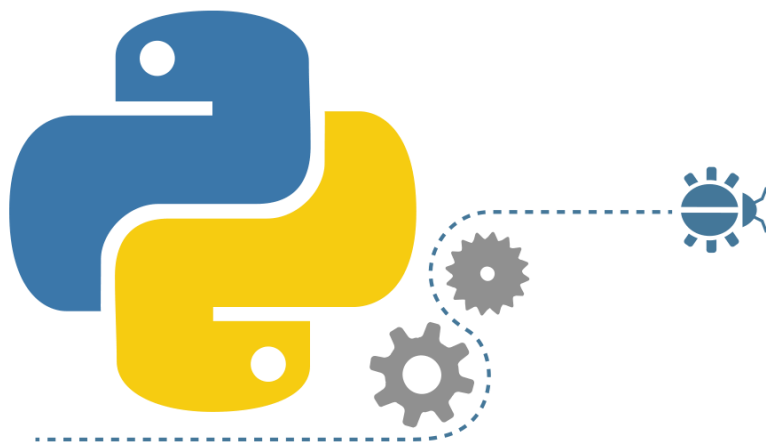
Viết hàm tính ROI



Nội dung bài học

- ROI (Return On Investment), một thuật ngữ quan trọng trong marketing, mà đặc biệt là SEO, tạm dịch là tỷ lệ lợi nhuận thu được so với chi phí bạn đầu tư. Có thể hiểu ROI một cách đơn giản chính là chỉ số đo lường tỷ lệ những gì bạn thu về so với những gì bạn phải bỏ ra.
- Hiểu đúng bản chất của ROI, bạn sẽ đo lường được hiệu quả đồng vốn đầu tư của mình cho các chi phí như quảng cáo, chạy Adwords, hay chi phí marketing online khác.
- Vì ROI dựa vào các chỉ số cụ thể, nên nó cũng là một thước đo rất cụ thể:
- $ROI = (\text{Doanh thu} - \text{Chi phí}) / \text{Chi phí}$
- Viết chương trình cho phép người dùng nhập vào Doanh thu và Chi phí và xuất ra tỉ lệ ROI cho người dùng, đồng thời hãy cho biết nên hay không nên đầu tư dự án khi biết ROI (giả sử mức tối thiểu $ROI = 0.75$ thì mới đầu tư).

Viết hàm đệ qui Fibonacci



Nội dung bài học

- Dãy Số Fibonacci là dãy số có dạng:
- $1 \rightarrow 1 \rightarrow 2 \rightarrow 3 \rightarrow 5 \rightarrow 8 \rightarrow 13 \rightarrow 21 \rightarrow 34 \rightarrow 55 \rightarrow 89 \dots$
- Được định nghĩa theo công thức đệ qui như dưới đây:

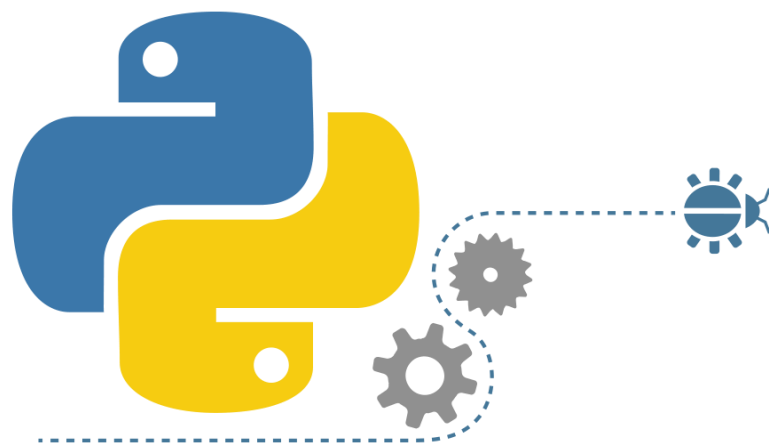
$$\text{Nếu } N=1, N=2 \rightarrow F_N=1$$

$$N>2 \quad F_N = F_{N-1} + F_{N-2}$$

Hãy viết 2 hàm:

- Hàm trả về số Fib tại vị trí thứ N bất kỳ
- Hàm trả về danh sách dãy số Fib từ 1 tới N

Các bài tập tự rèn luyện





Nội dung bài học

Câu 1: Cho 3 hàm dưới đây:

```
def sum1 (n) :  
    s = 0  
    while n > 0 :  
        s += 1  
        n -= 1  
    return s  
  
def sum2 () :  
    global val  
    s = 0  
    while val > 0 :  
        s += 1  
        val -= 1  
    return s
```

```
def sum3 () :  
    s = 0  
    for i in range (val, 0, -1) :  
        s += 1  
    return s
```

Nội dung bài học

Hãy cho biết kết quả sau khi gọi các lệnh trên:

Câu a)

```
def main():  
    global val  
    val = 5  
    print(sum1(5))  
    print(sum2())  
    print(sum3())  
  
main()
```

Câu b)

```
def main():  
    global val  
    val = 5  
    print(sum1(5))  
    print(sum3())  
    print(sum2())  
  
main()
```

Câu c)

```
def main():  
    global val  
    val = 5  
    print(sum2())  
    print(sum1(5))  
    print(sum3())  
  
main()
```



Nội dung bài học

Câu 3: Cho coding

```
for n in oscillate(-3, 5):  
    print(n, end=' ')  
print()
```

Hãy viết hàm oscillate để khi chạy phần mềm, nó ra kết quả:

-3 3 -2 2 -1 1 0 0 1 -1 2 -2 3 -3 4 -4

Nội dung bài học

Câu 4:Viết hàm tính tổng ước số để áp dụng chung cho 2 bài dưới đây:

- ❖ **4.1** : Kiểm tra số nguyên dương n có phải là số hoàn thiện (Pefect number) hay không? (Số hoàn thiện là số có tổng các ước số của nó (không kể nó) thì bằng chính nó. Vd: 6 có các ước số là 1,2,3 và $6=1+2+3 \rightarrow 6$ là số hoàn thiện)
- ❖ **4.2**: Kiểm tra số nguyên dương n có phải là số thịnh vượng (Abundant number) hay không? (Số thịnh vượng là số có tổng các ước số của nó (không kể nó) thì lớn hơn nó. Vd:12 có các ước số là 1,2,3,4,6 và $12<1+2+3+4+6 \rightarrow 12$ là số thịnh vượng)