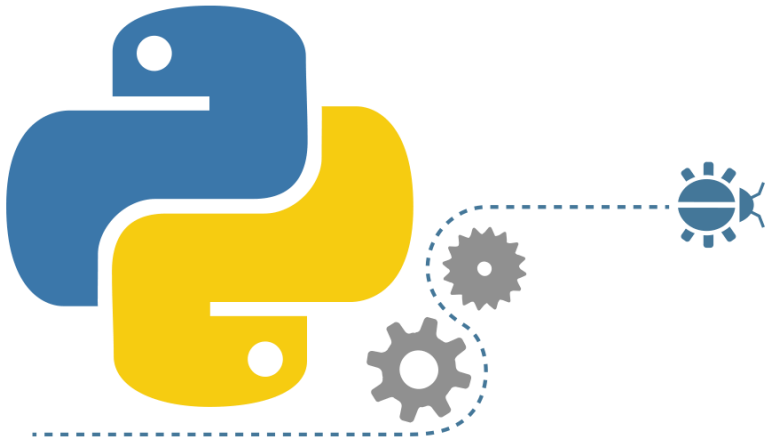
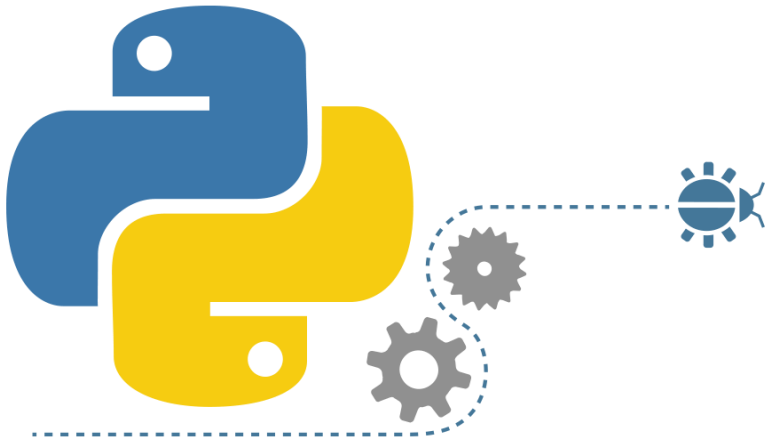
Chuyên đề ngôn ngữ lập trình
Chương 4



Cần Thơ, ngày 16 tháng 10 năm 2019

Lambda expression



Nội dung bài học

Python hỗ trợ kiểu khai báo hàm nặc danh thông qua Lambda expression:

```
lambda parameterlist : expression
```

lambda: là từ khóa

paramaterlist: tập hợp các **parameter** mà ta muốn định nghĩa

expression: biểu thức đơn trong Python (không nhập complex)

Nội dung bài học

- Ví dụ ta định nghĩa 1 hàm

```
1    def handle(f,x):
2        return f(x)
```

- Ta thấy đối số 1 là 1 hàm f nào đó
- Từ handle này ta có thể gọi tùy ý các giao tác:

```
ret1=handle(lambda x:x%2==0,7)
ret2=handle(lambda x:x%2!=0,7)
```

- Trước dấu 2 chấm là từ khóa lambda, đằng sau nó là số lượng các biến được khai báo trong handle (tính sau chữ f). Tức là nếu ta handle(f,x,y) thì viết lambda x, y:

```
def handle(f,x,y):
    return f(x,y)
sum=handle(lambda x,y:x+y,7,9)
print(sum)
```

Nội dung bài học

- Vì lambda expression không nhận cách viết complex, do đó nếu muốn complex thì ta nên định nghĩa các hàm độc lập rồi truyền vào cho Lamba expression. Ví dụ:
- Ở bên ta có 4 hàm, trong đó soChan, soLe, SoNguyenTo sẽ được gọi vào handle

```
1   def handle(f,x):
2       return f(x)
3   def soChan(x):
4       return x%2==0
5   def soLe(x):
6       return x%2==1
7   def soNguyenTo(x):
8       dem=0
9       for i in range(1,x+1):
10          if x % i is 0:
11              dem=dem+1
12      return dem==2
```

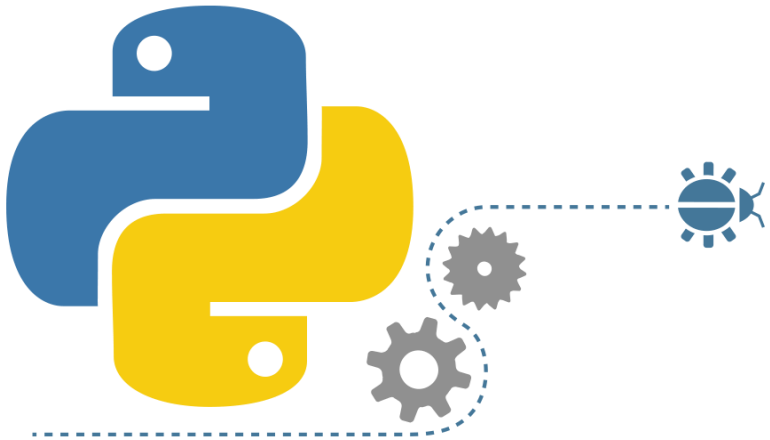
Nội dung bài học

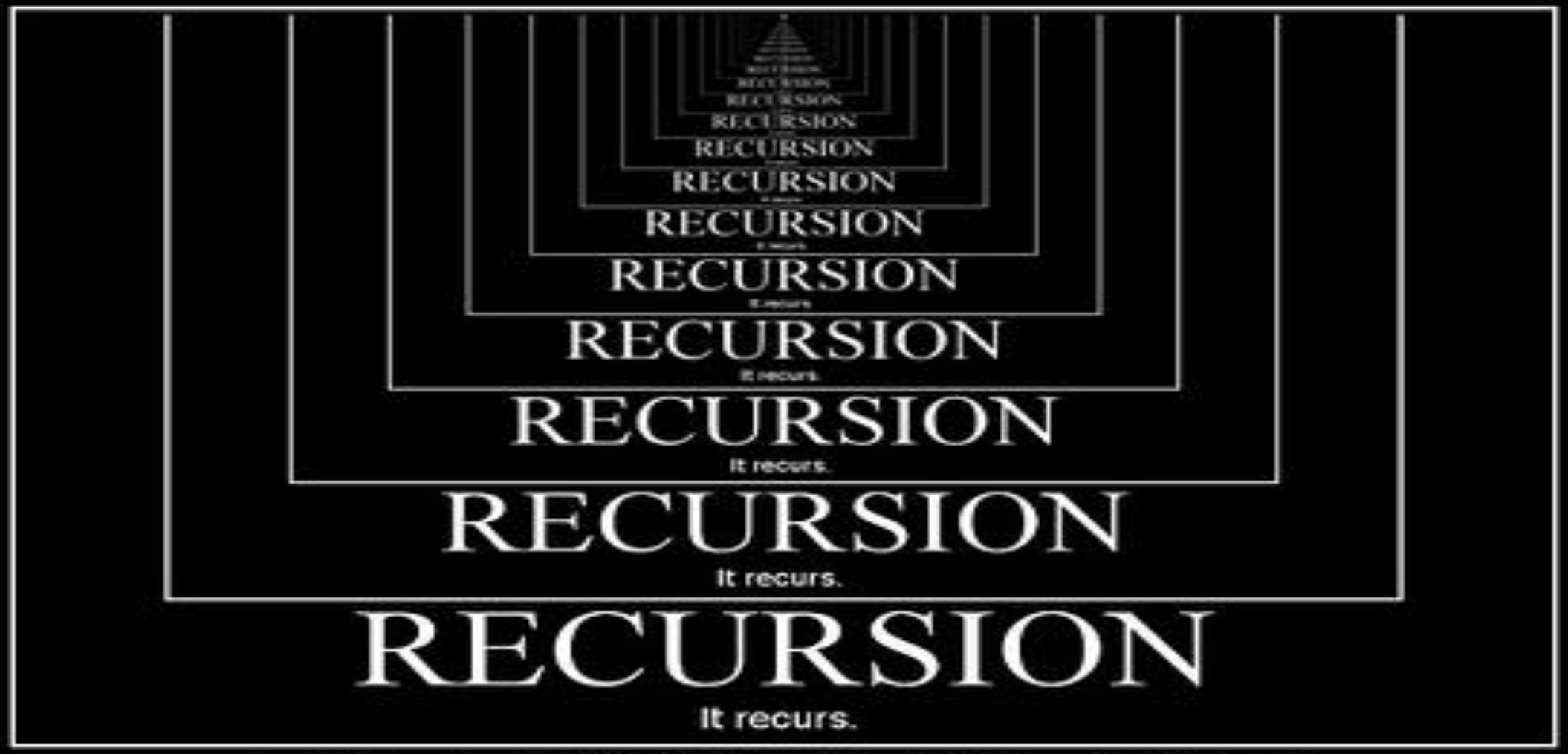
Một số Cách sử dụng:

```
ret1=handle(soChan,6)
print(ret1)
ret2=handle(lambda x:soChan(x),6)
print(ret2)
ret3=handle(soLe,6)
print(ret3)
ret4=handle(lambda x:soLe(x),7)
print(ret4)
ret5=handle(soNguyenTo,5)
print(ret5)
ret6=handle(lambda x:soNguyenTo(x),9)
print(ret6)
```

```
True
True
False
True
True
False
False
16
```

Giới thiệu về hàm đệ qui




RECURSION
It recurs.
RECURSION
It recurs.
RECURSION
It recurs.
RECURSION
It recurs.
RECURSION
It recurs.
RECURSION
It recurs.
RECURSION
It recurs.
RECURSION
It recurs.
RECURSION
It recurs.
RECURSION
It recurs.
RECURSION
It recurs.

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 hóc 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.

Nội dung bài học

Một vài ví dụ kinh điển về đệ qui như:

- Tính giai thừa: N!=N*(N-1)! ➔ Đệ qui: Nếu biết được (N-1)! Thì sẽ tính được N!
- Tính dãy số Fibonacci: F1=1, F2=1, $F_N=F_{N-1}+F_{N-2}$

Nội dung bài học

- Khi quyết định giải quyết bài toán theo Đệ Qui thì điều quan trọng phải nghĩ tới đó là:

1) Điểm dừng của bài toán là gì?
2) 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:

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

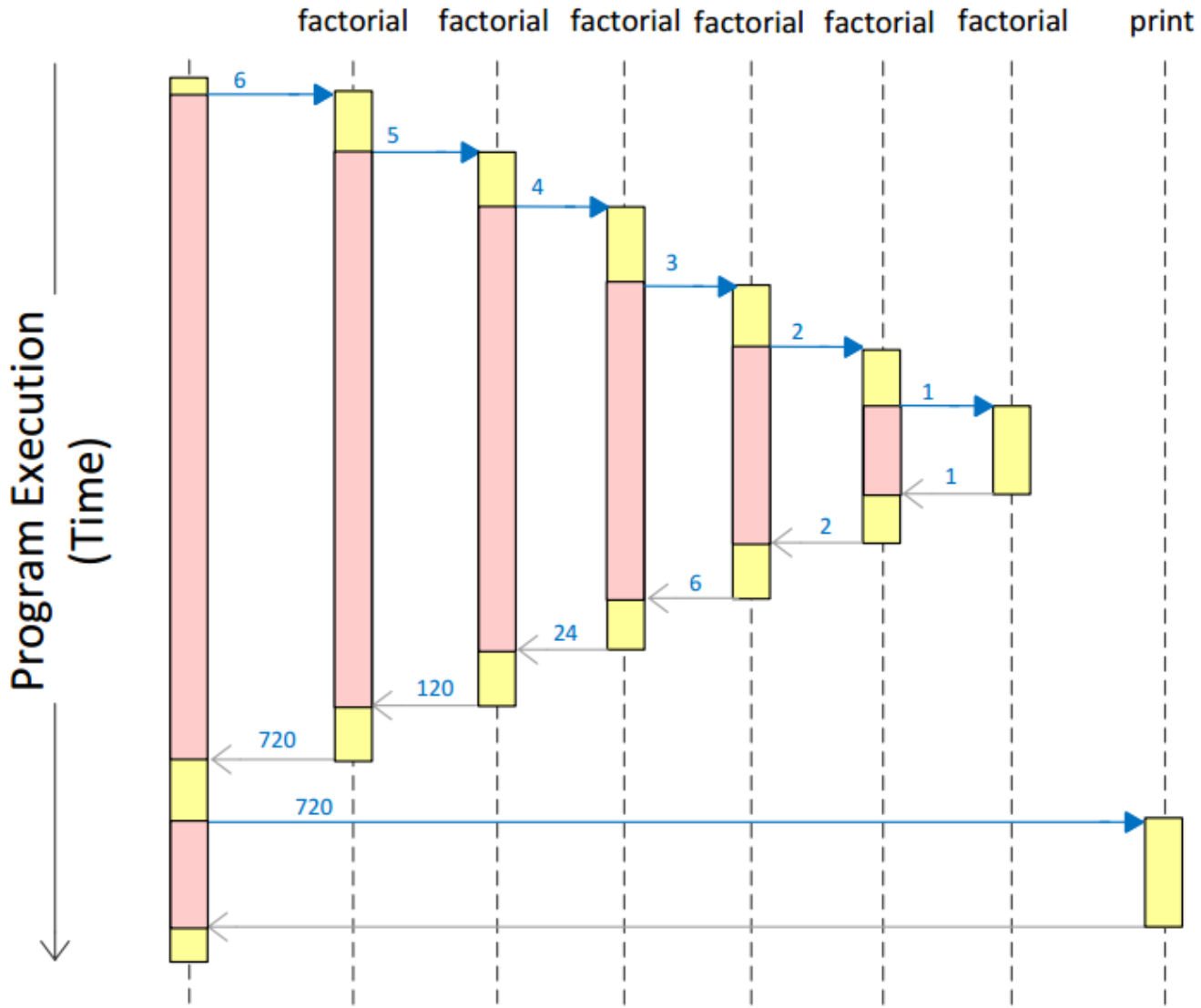
- Điểm dừng là khi n=0, quy luật là nếu biết (n-1)! Thì tính được N!, vì N!=N*(N-1)!

Nội dung bài học

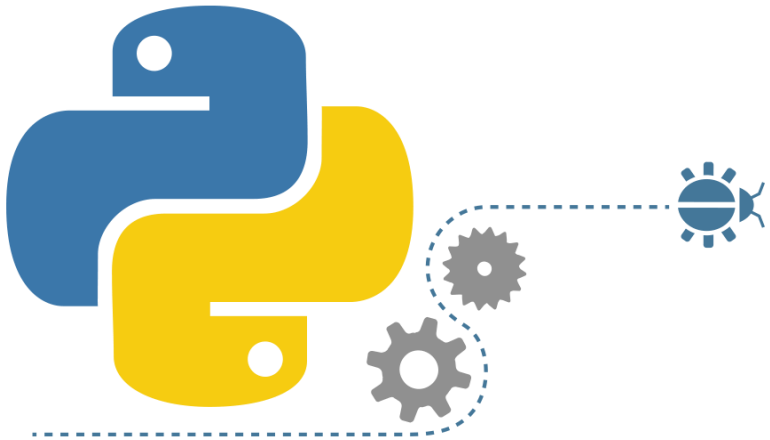
```
def factorial(n):
    """
    Hàm tính n!
    Trả về giai thừa của n
    """
    if n == 0:
        return 1
    else:
        return n * factorial(n - 1)

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
```

Nội dung bài học



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:

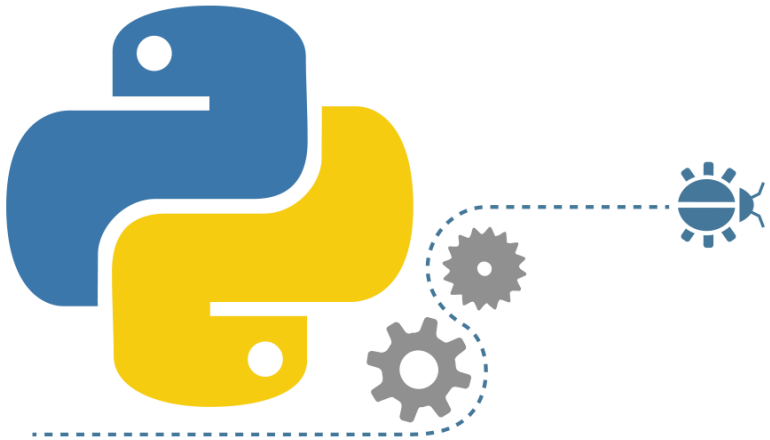
$$BMI = \frac{\text{Cân nặng (kg)}}{\text{Chiều cao x chiều cao (m)}}$$

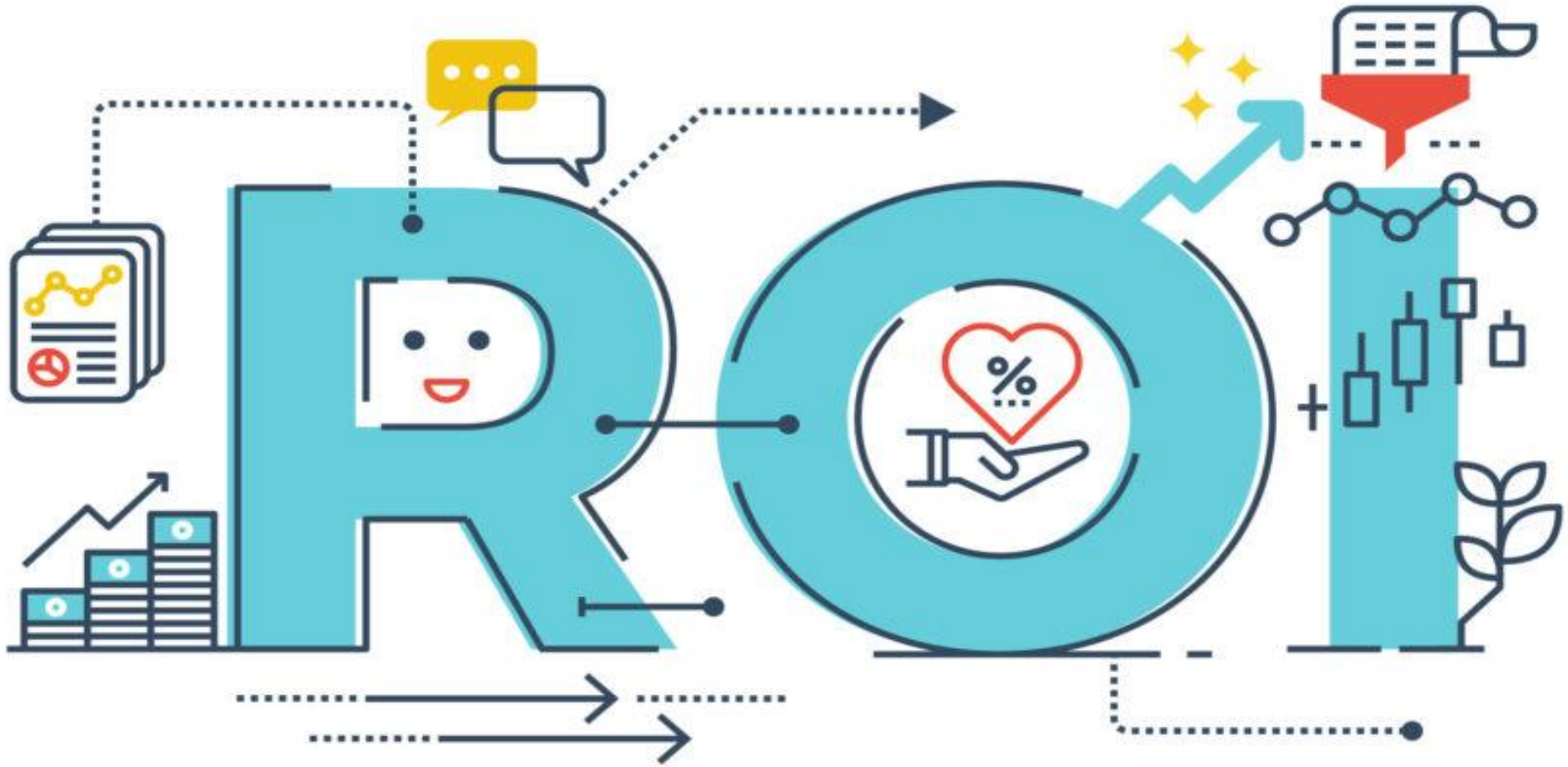
CHỈ SỐ KHỐI CƠ THỂ	PHÂN LOẠI	NGUY CƠ PHÁT TRIỂN BỆNH
< 18.5	Gầy	Thấp
18.5 - 24.9	Bình thường	Trung bình
25.0 - 29.9	Hơi béo	Cao
30.0 - 34.9	Béo phì cấp độ 1	Cao
35.0 - 39.9	Béo phì cấp độ 2	Rất cao
> 40.0	Béo phì cấp độ 3	Nguy hiểm

- Hãy thông báo phân loại

Và cảnh bảo nguy cơ cho họ

Viết hàm tính ROI




ROI

RETURN
ON
INVESTMENT

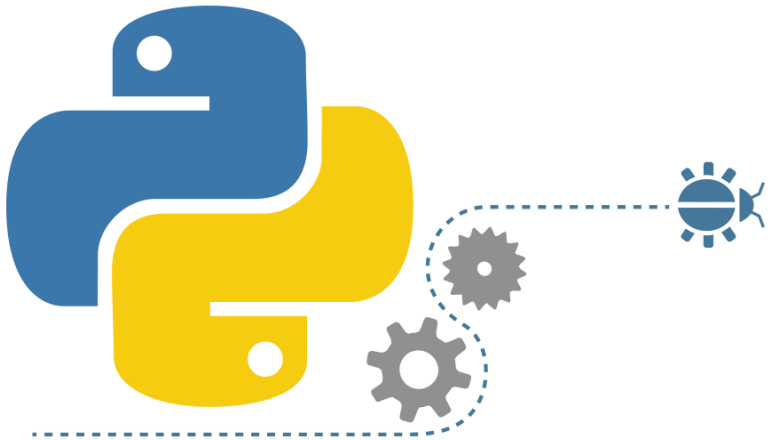
Nội dung bài học

- 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.

Nội dung bài học

- Vì ROI dựa vào các chỉ số cụ thể, nên nó cũng là một thước đo rất cụ thể:
- ROI = (Doanh thu – Chi phí)/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→1→2→3→5→8→13→21→34→55→89…
- Được định nghĩa theo công thức đệ qui như dưới đây:

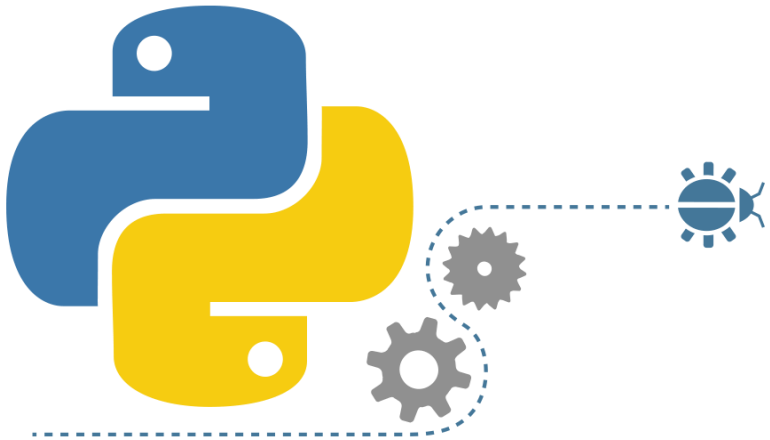
  Nếu N=1,N=2➔F_N=1

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

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:

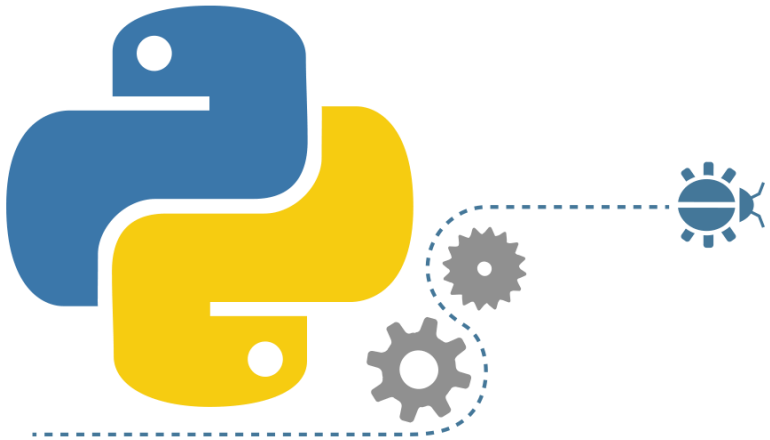
-3 3 -2 2 -1 1 0 0 1 -1 2 -2 3 -3 4 -4

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 ➔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 ➔12 là số thịnh vượng)

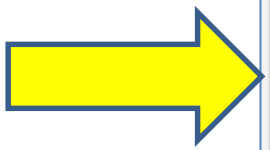
Khái niệm và cấu trúc của chuỗi



Nội dung bài học

Chuỗi là tập các ký tự nằm trong nháy đơn hoặc nháy đôi, hoặc 3 nháy đơn hoặc 3 nháy đôi. Chuỗi rất quan trọng trong mọi ngôn ngữ, hầu hết ta đều gặp xử lý chuỗi

```
1   s1='Hello Simon'
2   s2="Hello David"
3   s3="""
4   Quanh năm buôn bán ở mom sông
5   nuôi đủ năm con với một chồng
6       lặn lộ thân cò khi quãng vắng
7       eo sèo mặt nước buổi đò đông
8   """
9   s4='''Cha mẹ thói đời ăn ở bạc
10      có chồng hờ hững cũng như không'''
11  print(s1)
12  print(s2)
13  print(s3)
14  print(s4)
```



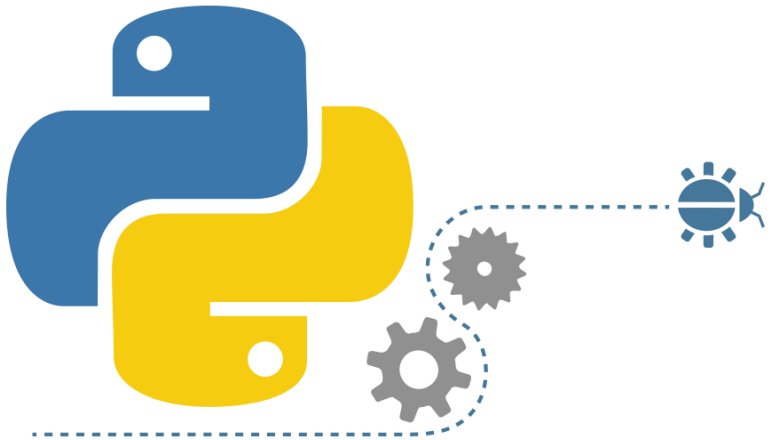
```
Hello Simon
Hello David

Quanh năm buôn bán ở mom sông
nuôi đủ năm con với một chồng
    lặn lộ thân cò khi quãng vắng
    eo sèo mặt nước buổi đò đông

Cha mẹ thói đời ăn ở bạc
    có chồng hờ hững cũng như không
```

Nội dung bài học

Chuỗi trong Python cũng là đối tượng, nó cung cấp một số hàm rất quan trọng:

Tên hàm	Mô tả
upper, lower	Xử lý in Hoa, in thường
rjust	Căn lề phải
ljust	Căn lề trái
center	Căn giữa
strip	Xóa khoảng trắng dư thừa
startswith	Kiểm tra Chuỗi có phải bắt đầu là ký tự ?
endswith	Kiểm tra Chuỗi có phải kết thúc là ký tự ?
count	Đếm số lần xuất hiện trong Chuỗi
find	Tìm kiếm Chuỗi con
format	Định dạng Chuỗi
__len__()	Trả về số lượng ký tự trong chuỗi, dùng index để lấy ký tự ra: str[index]

Hàm upper, lower -in HOA-thường



Nội dung bài học

Hàm upper ➔đưa Chuỗi về In HOA

Hàm lower➔ đưa Chuỗi về In thường

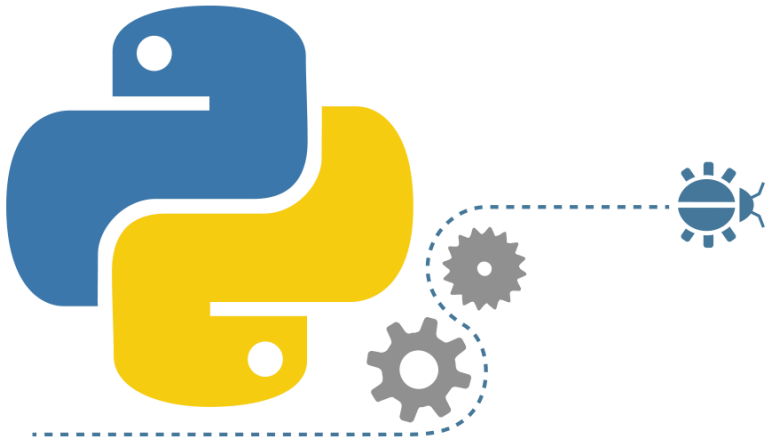
```
1    name = "Tram Vu Kiet"
2    print(name.upper())
3
```

➔ TRAM VU KIET

```
1    name = "KHOA CONG NGHE THONG TIN"
2    print(name.lower())
3
```

➔ Khoa cong nghe thong tin

Hàm căn lề rjust, ljust, center



Nội dung bài học

Hàm rjust ➔ căn lề phải

Hàm ljust ➔ căn lề trái

Hàm center ➔ căn giữa

Rjust

Hàm rjust sẽ căn phải Chuỗi, nếu truyền 1 đối số Python sẽ chèn khoảng trắng, nếu có đối số thứ 2 thì chèn nó vào trước.

```
word = "ABCD"
print(word.rjust(10, "*"))
print(word.rjust(3, "*"))
print(word.rjust(15, ">"))
print(word.rjust(10))
```

```
******ABCD
ABCD
>>>>>>>>>>>ABCD
      ABCD
```

Lưu ý nếu số ký tự chèn nhỏ hơn chuỗi gốc thì không có gì thay đổi (trường hợp rjust(3, “*”))

ljust

Hàm ljust sẽ căn trái Chuỗi, nếu truyền 1 đối số Python sẽ chèn khoảng trắng đằng sau, nếu có đối số thứ 2 thì chèn nó vào sau.

```
word="OBAMA"
print(word.ljust(1))
print(word.ljust(2))
print(word.ljust(3))
print(word.ljust(4))
print(word.ljust(5))
print(word.ljust(10))
print(word.ljust(10,'*'))
```

```
OBAMA
OBAMA
OBAMA
OBAMA
OBAMA
OBAMA*****
```

Lưu ý nếu số ký tự muốn chèn nhỏ hơn Chuỗi gốc thì không có gì thay đổi

center

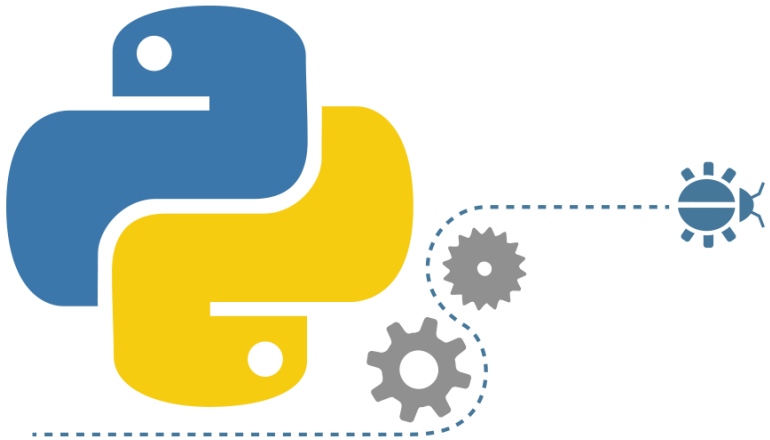
Hàm center căn giữa Chuỗi, nó tự đẩy khoảng trắng 2 bên sao cho tổng ký tự bằng giá trị muốn truyền vào. Nếu có đối số thứ 2 thì thay khoảng trắng bằng ký tự mới này

```
word="TRUMP"
print(word.center(10))
print(word.center(10,'*'))
print(word.center(21,'*'))
```

```
  TRUMP
**TRUMP***
********TRUMP********
```

Lưu ý: Nếu số lượng căn giữa mà nhỏ hơn số ký tự gốc thì không có gì thay đổi.

Hàm xóa khoảng trắng dư thừa strip



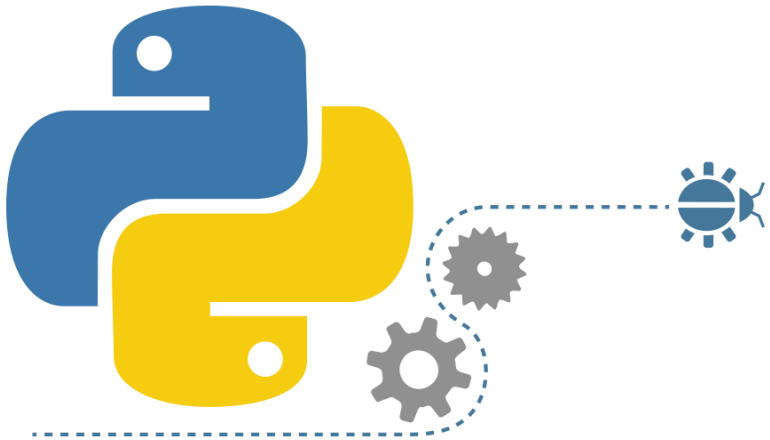
Nội dung bài học

Để xóa khoảng trắng dư thừa, Python hỗ trợ hàm trip

```
s = " ABCDEFGHBCDIJKLMNOPQRSBCDTUVWXYZ "
print(s)
print(s.__len__())
s=s.strip()
print(s)
print(s.__len__())
```

```
 ABCDEFGHBCDIJKLMNOPQRSBCDTUVWXYZ
34
ABCDEFGHBCDIJKLMNOPQRSBCDTUVWXYZ
32
```

Hàm startsWith, endsWith



Nội dung bài học

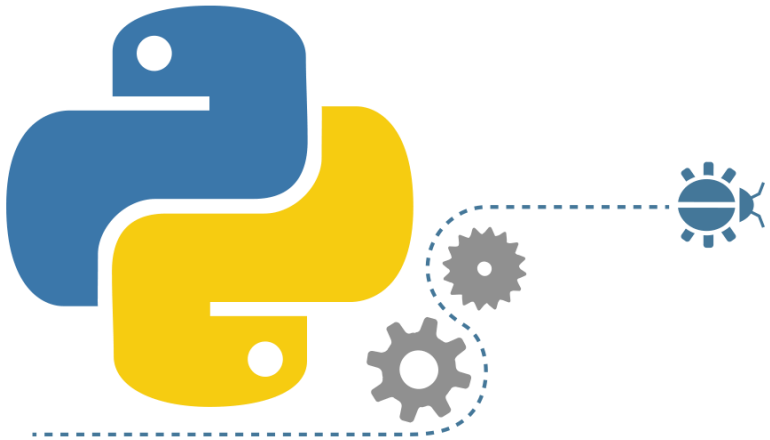
startswith để kiểm tra Chuỗi có bắt đầu bằng 1 chuỗi con nào đó hay không

endswith để kiểm tra Chuỗi có kết thúc bằng 1 chuỗi con nào đó hay không

```
s="#hello Python*"
print(s.startswith("#"))
print(s.startswith("*"))
print(s.endswith("#"))
print(s.endswith("*"))
```

```
True
False
False
True
```

Hàm find, count



Nội dung bài học

Hàm **find** trả về vị trí đầu tiên tìm thấy, hàm rfind trả về vị trí cuối cùng tìm thấy. Nếu không thấy sẽ trả về -1

```
s="hello hello hello"
x1=s.find('o')
print(x1)
x2=s.rfind('o')
print(x2)
x3=s.find('x')
print(x3)
```

```
4
16
-1
```

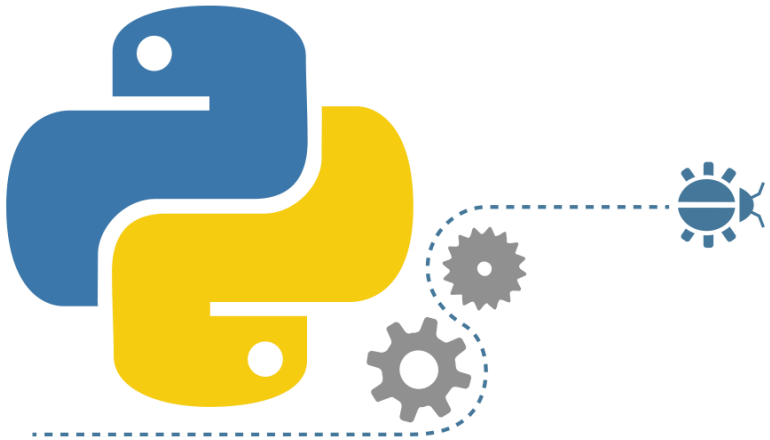
Nội dung bài học

Hàm **count** trả về số lần xuất hiện của Chuỗi con trong Chuỗi gốc, không tồn tại trả về 0

```
s="Obama likes Putin, Putin likes Kim Jong Un"
c1=s.count("Putin")
print(c1)
c2=s.count("Trump")
print(c2)
```

```
2
0
```

Hàm format, substring



Nội dung bài học

Hàm format sử dụng {} để dành chỗ xuất dữ liệu

```
a=5
b=9
c=a/b
s="{0}/{1}={2}".format(a,b,c)
print(s)
```

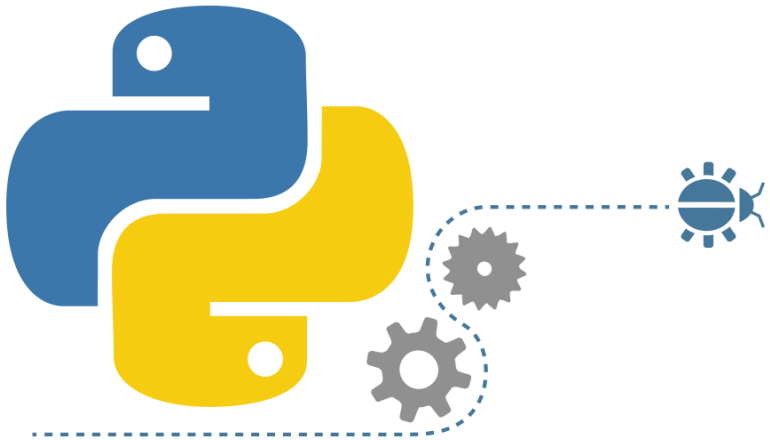
```
5/9=0.5555555555555556
```

Nội dung bài học

substring

```
x = "Hello World!"
print(x[2:])#"llo World!"
print(x[:2])#"He"
print(x[:-2])#"Hello Worl"
print(x[-2:])#"d!"
print(x[2:-2])#"llo Worl"
print(x[6:11])#"World"
```

Hàm tách chuỗi



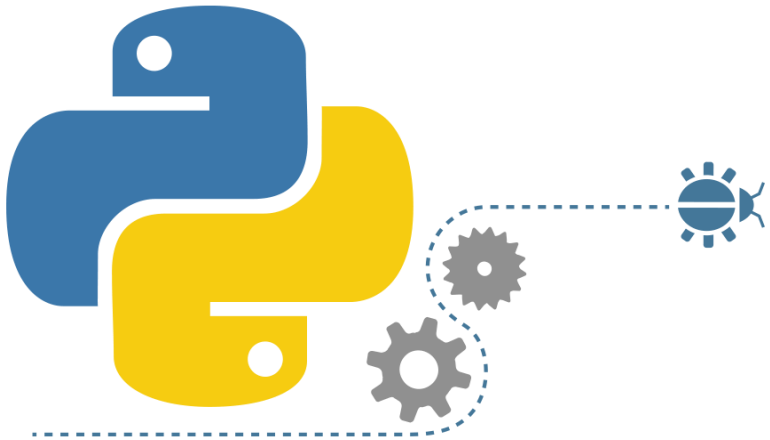
Nội dung bài học

Hàm split dùng để tách chuỗi thành mảng các chuỗi con

```
s="sv007;Nguyễn Thị Tẹt;1/1/1999"
arr=s.split(';')
for x in arr:
    print(x)
```

```
sv007
Nguyễn Thị Tẹt
1/1/1999
```

Hàm nối chuỗi



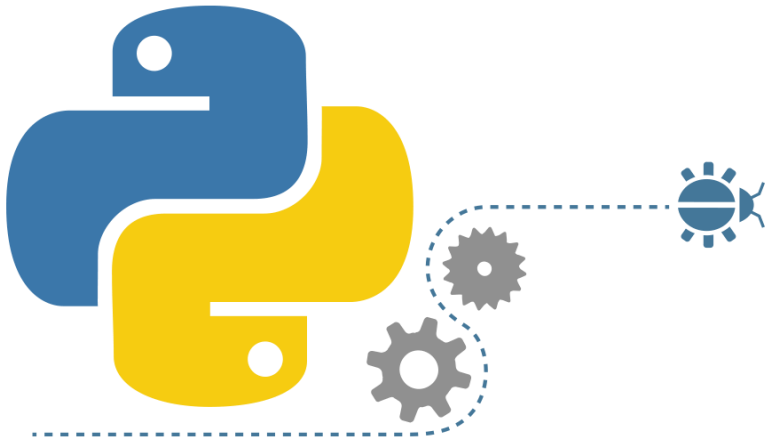
Nội dung bài học

Hàm join dùng để nối Chuỗi:

```
s="sv007;Nguyễn Thị Tẹt;1/1/1999"
arr=s.split(';')
for x in arr:
    print(x)
s2=","
s2=s2.join(arr)
print(s2)
```

```
sv007
Nguyễn Thị Tẹt
1/1/1999
sv007,Nguyễn Thị Tẹt,1/1/1999
```

Bài tập rèn luyện
-Kiểm tra chuỗi đối xứng



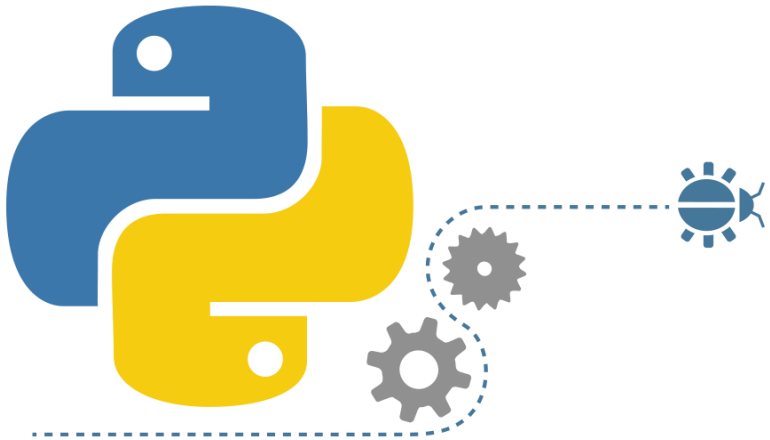
Nội dung bài học

Dùng vòng lặp while vĩnh cửu, cho phép Nhập vào một Chuỗi➔Xuất Chuỗi này có phải đối xứng hay không?

Hỏi người sử dụng có tiếp tục phần mềm.

Nếu tiếp tục thì nhập Chuỗi mới, còn không thì thoát và thông báo cảm ơn

Viết chương trình tối ưu chuỗi



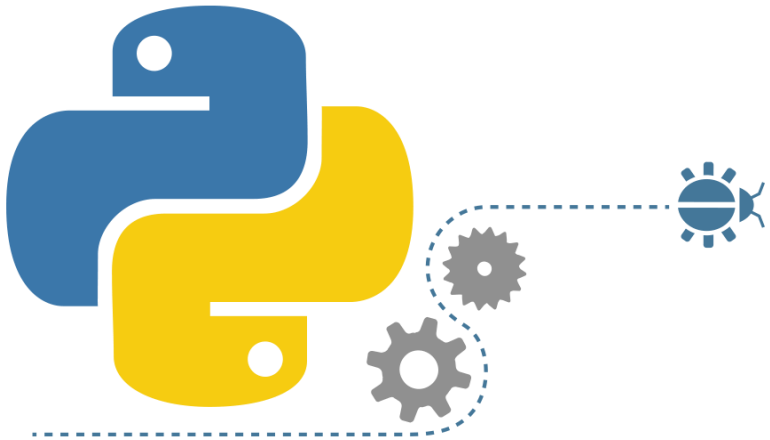
Nội dung bài học

Một Chuỗi được gọi là tối ưu khi:

1) Không chứa các khoảng trắng dư thừa,

2) Các từ cách nhau bởi một khoảng trắng.

Tách xử lý chuỗi

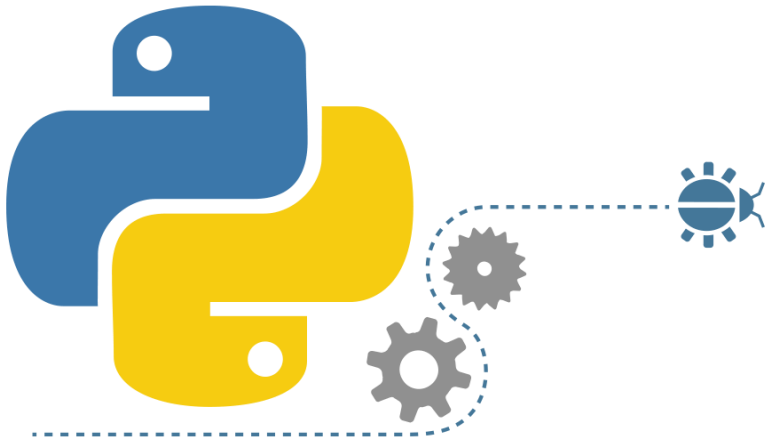


Nội dung bài học

Cho 1 Chuỗi như sau “5;7;8;-2;8;11;13;9;10”

- xuất các chữ số trên các dòng riêng biệt
- Xuất có bao nhiêu chữ số chẵn
- Xuất có bao nhiêu số âm
- Xuất có bao nhiêu chữ số nguyên tố
- Tính giá trị trung bình

Các bài tập tự rèn luyện



Nội dung bài học

Câu 1: Trình bày một số hàm quan trọng trong xử lý Chuỗi của Python

Câu 2: Viết chương trình cho phép nhập vào 1 chuỗi. Yêu cầu xuất ra:

- Bao nhiêu chữ IN HOA
- Bao nhiêu chữ in thường
- Bao nhiêu chữ là chữ số
- Bao nhiêu chữ là ký tự đặc biệt
- Bao nhiêu chữ là khoảng trắng
- Bao nhiêu chữ là Nguyên Âm
- Bao nhiêu chữ là Phụ âm

Nội dung bài học

Câu 3: Viết một hàm đặt tên là **NegativeNumberInStrings**(str). Hàm này có đối số truyền vào là một chuỗi bất kỳ. Hãy viết lệnh để xuất ra các số nguyên âm trong chuỗi.

Ví dụ: Nếu nhập vào chuỗi "**abc-5xyz-12k9l--p**" thì hàm phải xuất ra được 2 số nguyên âm đó là -5 và -12

Nội dung bài học

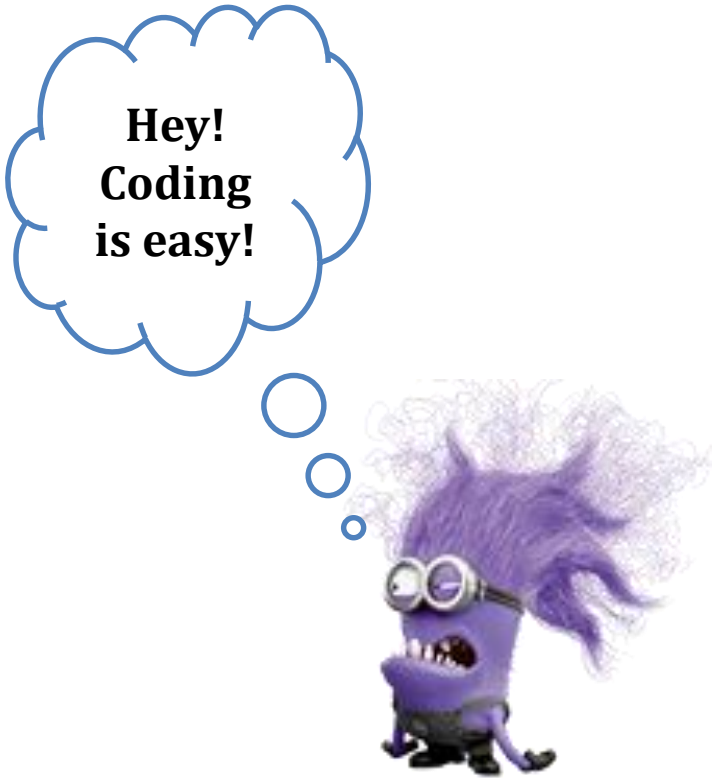
Câu 4: Viết chương trình tối ưu Chuỗi danh từ

Một Chuỗi được gọi là tối ưu khi: Không chứa các khoảng trắng dư thừa, các từ cách nhau bởi một khoảng trắng, Ký tự đầu tiên của các từ Viết Hoa.

Ví dụ:

Input “ TRầM vŨ kiỆt ”

Output “Trầm Vũ Kiệt”



END