Hơn 100 bài tập Python có lời giải (code mẫu)

Bài tập Python có rất nhiều trên Internet. Nhằm giúp bạn dễ **luyện code Python**, dưới đây là những bài tập tổng hợp kiến thức cơ bản nhất của ngôn ngữ lập trình này.

Lưu ý: Các code mẫu trong bài được viết trên Python 3.6.2, nếu bạn đang sử dụng phiên bản Python từ 2.5 trở xuống có thể không chạy được code vì trong bản Python mới nhiều lệnh, hàm đã được thay đổi.

Số bài tập Python này sẽ được chia thành 3 cấp độ dành cho cả người mới bắt đầu học lập trình, người đã học lập trình nhưng mới học Python và những người muốn nâng cao trình độ Python. Mỗi bài tập đều có đầy đủ các phần là yêu cầu của bài, gợi ý làm bài và lời giải (code mẫu), chính là code Python mẫu để bạn tham khảo. Bây giờ mời bạn đi vào các nội dung cụ thể nhé.

Bài tập Python thực hành với code mẫu

1. Mô tả cấp độ Python

Level 1: Người vừa trải qua khóa học tổng quan về Python, có thể giải quyết một số vấn đề với 1, 2 class hoặc hàm Python. Những bài tập thuộc cấp độ này có thể tìm thấy trong các sách giáo khoa, tài liệu hướng dẫn thông thường.

Level 2: Người mới học Python nhưng đã có nền tảng lập trình tương đối mạnh mẽ từ trước, có thể giải quyết các vấn đề liên quan tới 3 lớp hoặc hàm Python. Những bài tập này thường không tìm thấy trong sách giáo khoa.

Level 3: Nâng cao, sử dụng Python để giải quyết những vấn đề phức tạp hơn bằng cách sử dụng nhiều hàm, cấu trúc dữ liệu và thuật toán phong phú. Ở cấp độ này bạn có thể giải quyết các vấn đề sử dụng vài package Python tiêu chuẩn và những kỹ thuật lập trình nâng cao.

2. Cấu trúc bài tập Python

Mỗi bài tập Python trong trang này sẽ gồm có 3 phần như sau:

- Câu hỏi.
- Gợi ý.
- Code mẫu.

Mình sẽ để các Bài tập Python nguyên xi dạng gốc như thế này, bạn có thể xem câu hỏi, gợi ý sau đó tự thực hành trước khi kéo xuống xem code mẫu nhé.

3. Bài tập Python level 1

Bài 1:

Câu hỏi:

Viết chương trình tìm tất cả các số chia hết cho 7 nhưng không phải bội số của 5, nằm trong đoạn 2000 và 3200 (tính cả 2000 và 3200). Các số thu được sẽ được in thành chuỗi trên một dòng, cách nhau bằng dấu phẩy.

Gợi ý:

Để giải quyết bài toán này, bạn có thể sử dụng vòng lặp for để duyệt qua tất cả các số trong đoạn từ 2000 đến 3200. Sau đó, bạn sử dụng câu lệnh if để kiểm tra xem mỗi số có chia hết cho 7 không và không phải là bội số của 5 không. Nếu đúng, thì in số đó ra màn hình.

Code mẫu:

```
j=[] #Tạo một danh sách rỗng để lưu kết quả
for i in range(2000, 3201): #Duyệt qua tất cả các số trong đoạn từ 2000 đến 3200
    if (i%7==0) and (i%5!=0): #Kiểm tra xem số i có chia hết cho 7 và không phải là bội số của 5 không
        j.append(str(i)) #Nếu đúng, thì thêm số i vào danh sách result
print (','.join(j)) #In ra màn hình danh sách result, các phần tử cách nhau bằng dấu phẩy
```

Bài 2:

Câu hỏi:

Viết một chương trình có thể tính giai thừa của một số cho trước. Kết quả được in thành chuỗi trên một dòng, phân tách bởi dấu phẩy. Ví dụ, số cho trước là 8 thì kết quả đầu ra phải là 40320.

Gợi ý:

Trong trường hợp dữ liệu đầu vào được cung cấp, bạn hãy chọn cách để người dùng nhập số vào.

Code mẫu:

```
x=int(input("Nhập số cần tính giai thừa:"))
def fact(x):
    if x == 0:
        return 1
    return x * fact(x - 1)
print (fact(x))
```

Bài 3:

Câu hỏi:

Với số nguyên n nhất định, hãy viết chương trình để tạo ra một dictionary chứa (i, i*i) như là số nguyên từ 1 đến n (bao gồm cả 1 và n) sau đó in ra dictionary này.

Ví dụ: Giả sử số n là 8 thì đầu ra sẽ là: {1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64}.

Gợi ý:

- Viết lệnh yêu cầu nhập số nguyên n.

Code mẫu:

```
n=int(input("Nhập vào một số:"))
d=dict()
for i in range(1,n+1):
    d[i]=i*i
    #Code by TH&THCSTTQuan Chu
```

```
print (d)
```

Bài 4:

Câu hỏi:

Viết chương trình chấp nhận một chuỗi số, phân tách bằng dấu phẩy từ giao diện điều khiển, tạo ra một danh sách và một tuple chứa mọi số.

Ví dụ: Đầu vào được cung cấp là 34,67,55,33,12,98 thì đầu ra là:

['34', '67', '55', '33', '12', '98'] ('34', '67', '55', '33', '12', '98')

Gợi ý:

- Viết lệnh yêu cầu nhập vào các giá trị sau đó dùng quy tắc chuyển đổi kiểu dữ liệu để hoàn tất.

Code mẫu:

```
values=input("Nhập vào các giá trị:")
l=values.split(",")
t=tuple(l)
print (l)
print (t)
```

Bài 5:

Câu hỏi:

Định nghĩa một class có ít nhất 2 method:

- **getString:** để nhận một chuỗi do người dùng nhập vào từ giao diện điều khiển.
- **printString:** in chuỗi vừa nhập sang chữ hoa.

Thêm vào các hàm kiểm tra đơn giản để kiểm tra method của class.

Ví dụ: Chuỗi nhập vào là TH&THCSTTQuan Chu thì đầu ra phải là: TH&THCSTTQUAN CHU

Gợi ý:

- Sử dụng __init__ để xây dựng các tham số.

Code mẫu:

```
class InputOutString(object):
  def __init__(self):
    self.s = ""

  def getString(self):
    self.s = input("Nhập chuỗi:")
# Code by TH&THCSTTQuan Chu
  def printString(self):
    print (self.s.upper())
```

```
strObj = InputOutString()
strObj.getString()
strObj.printString()
```

Bài 6:

Câu hỏi:

Viết một method tính giá trị bình phương của một số.

Gợi ý:

- Sử dụng toán tử **.

Code mẫu:

```
x=int(input("Nhập một số:")) #nhập số cần tính bình phương từ giao diện
def square(num):  #định nghĩa bình phương của một số
  return num ** 2
# Code by TH&THCSTTQuan Chu
print (square(2)) #in bình phương của 2
print (square(3)) #in bình phương của 3
print (square(x)) #in bình phương của x
```

Vì đề bài không yêu cầu cụ thể bạn phải tính bình phương số có sẵn hay số nhập vào nên mình dùng cả hai.

Bài 7:

Câu hỏi:

Python có nhiều hàm được tích hợp sẵn, nếu không biết cách sử dụng nó, bạn có thể đọc tài liệu trực tuyến hoặc tìm vài cuốn sách. Nhưng Python cũng có sẵn tài liệu về hàm cho mọi hàm tích hợp trong Python. Yêu cầu của bài tập này là viết một chương trình để in tài liệu về một số hàm Python được tích hợp sẵn như abs(), int(), input() và thêm tài liệu cho hàm bạn tự định nghĩa.

Gợi ý:

- Sử dụng __doc__

Code mẫu:

```
print (abs.__doc__)
print (int.__doc__)
print (input.__doc__)
# Code by TH&THCSTTQuan Chu
def square(num):
 '''Trả lại giá trị bình phương của số được nhập vào.

 Số nhập vào phải là số nguyên.
 '''
 return num ** 2
```

```
print (square.__doc__)
```

Bài 8:

Câu hỏi:

Định nghĩa một lớp gồm có tham số lớp và có cùng tham số instance

Gợi ý:

- Khi định nghĩa tham số instance, cần thêm nó vào __init__
- Bạn có thể khởi tạo một đối tượng với tham số bắt đầu hoặc thiết lập giá trị sau đó.

Code mẫu:

```
class Person:
 # Định nghĩa lớp "name"
 name = "Person"
 # Code by TH&THCSTTQuan Chu
 def __init__(self, name = None):
 # self.name là biến instance
 self.name = name

jeffrey = Person("Jeffrey")
print ("%s name is %s" % (Person.name, jeffrey.name))

nico = Person()
nico.name = "Nico"
print ("%s name is %s" % (Person.name, nico.name))
```

Bài 9:

Câu hỏi: Tính tuổi dựa trên ngày tháng năm sinh nhập vào.

Ví dụ: Nhập vào ngày tháng năm sinh dạng y/m/d, hãy tính tuổi của người này.

Gợi ý: Sử dụng module datetime. Sử dụng datetime, chúng ta có thể tìm tuổi bằng cách trừ năm sinh cho năm hiện tại.

Code mẫu:

```
import datetime

print("Mời bạn vui lòng nhập ngày tháng năm sinh để tính tuổi")
birth_day = int(input("Ngày sinh:"))
birth_month = int(input("Tháng sinh:"))
birth_year = int(input("Năm sinh:"))

current_year = datetime.date.today().year
current_month = datetime.date.today().month
current_day = datetime.date.today().day
```

```
    age_year = current_year - birth_year
    age_month = abs(current_month-birth_month)
    age_day = abs(current_day-birth_day)

    print("### Tuổi của bạn chính xác là:### \n", age_year, " tuổi ", age_month, " tháng và ", age_day, " ngày")
```

Bài 10:

Viết chương trình nhập: số giờ làm mỗi tuần, thù lao trên mỗi giờ làm tiêu chuẩn, từ đó tính ra số tiền thực lĩnh của nhân viên. Biết rằng: số giờ tiêu chuẩn mỗi tuần là 44 giờ, và mỗi giờ vượt chuẩn được trả gấp rưỡi so với giờ làm chuẩn.

Code mẫu:

```
    so_gio_lam = float(input("Nhập số giờ làm mỗi tuần: "))
    luong_gio = float(input("Nhập thù lao trên mỗi giờ làm tiêu chuẩn: "))

    gio_tieu_chuan = 44  # Số giờ làm chuẩn mỗi tuần
    gio_vuot_chuan = max(0, so_gio_lam - gio_tieu_chuan)  # Số giờ làm vượt chuẩn mỗi tuần
    thuc_linh = gio_tieu_chuan * luong_gio + gio_vuot_chuan * luong_gio * 1.5  # Tính tổng thu nhập
    print(f"Số tiền thực lĩnh của nhân viên: {thuc_linh}")
```

4. Bài tập Python level 2

Bài 1:

Câu hỏi:

Viết chương trình và in giá trị theo công thức cho trước: Q = √([(2 * C * D)/H]) (bằng chữ: Q bằng căn bậc hai của [(2 nhân C nhân D) chia H]. Với giá trị cố định của C là 50, H là 30. D là dãy giá trị tùy biến, được nhập vào từ giao diện người dùng, các giá trị của D được phân cách bằng dấu phẩy.

Ví dụ: Giả sử chuỗi giá trị của D nhập vào là 100,150,180 thì đầu ra sẽ là 18,22,24.

Gợi ý:

- Nếu đầu ra nhận được là một số dạng thập phân, bạn cần làm tròn thành giá trị gần nhất, ví dụ 26.0 sẽ được in là 26.
- Trong trường hợp dữ liệu đầu vào được cung cấp cho câu hỏi, nó được giả định là đầu vào do người dùng nhập từ giao diện điều khiển.

Code mẫu:

```
    #!/usr/bin/env python
    import math
    c=50
    h=30
    value = []
```

```
items=[x for x in input("Nhập giá trị của d: ").split(',')]
for d in items:
    value.append(str(int(round(math.sqrt(2*c*float(d)/h)))))
# Code by TH&THCSTTQuan Chu
print (','.join(value))
```

Bài 2:

Câu hỏi:

Viết một chương trình có 2 chữ số, X, Y nhận giá trị từ đầu vào và tạo ra một mảng 2 chiều. Giá trị phần tử trong hàng thứ i và cột thứ j của mảng phải là i*j.

Lưu ý: i=0,1,...,X-1; j=0,1,...,Y-1.

Ví dụ: Giá trị X, Y nhập vào là 3,5 thì đầu ra là: [[0, 0, 0, 0, 0], [0, 1, 2, 3, 4], [0, 2, 4, 6, 8]]

Gợi ý:

- Viết lệnh để nhận giá trị X, Y từ giao diện điều khiển do người dùng nhập vào.

Code mẫu:

```
input_str = input("Nhập X, Y: ")
dimensions=[int(x) for x in input_str.split(',')]
rowNum=dimensions[0]
colNum=dimensions[1]
multilist = [[0 for col in range(colNum)] for row in range(rowNum)]
# Viết bởi TH&THCSTTQuan Chu
for row in range(rowNum):
    for col in range(colNum):
        multilist[row][col]= row*col

print (multilist)
```

Bài 3:

Câu hỏi:

Viết một chương trình chấp nhận chuỗi từ do người dùng nhập vào, phân tách nhau bởi dấu phẩy và in những từ đó thành chuỗi theo thứ tự bảng chữ cái, phân tách nhau bằng dấu phẩy.

Giả sử đầu vào được nhập là: without,hello,bag,world, thì đầu ra sẽ là: bag,hello,without,world.

Gợi ý:

Trong trường hợp dữ liệu đầu vào được nhập vào chương trình nó nên được giả định là dữ liệu được người dùng nhập vào từ giao diện điều khiển.

Code mẫu:

```
items=[x for x in input("Nhập một chuỗi: ").split(',')]
```

```
items.sort()
print (','.join(items))
```

Bài 4:

Câu hỏi:

Viết một chương trình chấp nhận chuỗi là các dòng được nhập vào, chuyển các dòng này thành chữ in hoa và in ra màn hình. Giả sử đầu vào là:

Hello world
Practice makes perfect

Thì đầu ra sẽ là:

HELLO WORLD
PRACTICE MAKES PERFECT

Gợi ý:

Trong trường hợp dữ liệu đầu vào được nhập vào chương trình nó nên được giả định là dữ liệu được người dùng nhập vào từ giao diện điều khiển.

Code mẫu:

```
lines = []

while True:
  s = input()
  if s:
    lines.append(s.upper())
  else:
    break;
# Bài Python 12, Code by TH&THCSTTQuan Chu
for sentence in lines:
   print (sentence)
```

Bài 5:

Câu hỏi:

Viết một chương trình chấp nhận đầu vào là một chuỗi các từ tách biệt bởi khoảng trắng, loại bỏ các từ trùng lặp, sắp xếp theo thứ tự bảng chữ cái, rồi in chúng.

Giả sử đầu vào là: hello world and practice makes perfect and hello world again

Thì đầu ra là: again and hello makes perfect practice world

Gợi ý:

- Trong trường hợp dữ liệu đầu vào được nhập vào chương trình nó nên được giả định là dữ liệu được người dùng nhập vào từ giao diện điều khiển.
- Sử dụng set để loại bỏ dữ liệu trùng lặp tự động và dùng sorted() để sắp xếp dữ liệu.

Code mẫu:

```
s = input("Nhập chuỗi của bạn: ")
words = [word for word in s.split(" ")]
print (" ".join(sorted(list(set(words)))))
```

Bài 6:

Câu hỏi:

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

Ví dụ đầu vào là: 0100,0011,1010,1001

Đầu ra sẽ là: 1010

Gợi ý:

Trong trường hợp dữ liệu đầu vào được nhập vào chương trình nó nên được giả định là dữ liệu được người dùng nhập vào từ giao diện điều khiển.

Code mẫu:

```
value = []

items=[x for x in input("Nhập các số nhị phân: ").split(',')]
for p in items:
    intp = int(p, 2)
    if not intp%5:
    value.append(p)
# Bài tập Python 14, viết bởi TH&THCSTTQuan Chu
print (','.join(value))
```

Bài 7:

Câu hỏi:

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

Gợi ý:

Trong trường hợp dữ liệu đầu vào được nhập vào chương trình nó nên được giả định là dữ liệu được người dùng nhập vào từ giao diện điều khiển.

Code mẫu:

```
values = []

for i in range(1000, 3001):
    s = str(i)
    if (int(s[0])%2==0) and (int(s[1])%2==0) and (int(s[2])%2==0) and (int(s[3])%2==0):
 values.append(s)
```

```
# Bài tập Python 15, Code by TH&THCSTTQuan Chu
print (",".join(values))
```

Bài 8:

Câu hỏi:

Viết một chương trình chấp nhận đầu vào là một câu, đếm số chữ cái và chữ số trong câu đó. Giả sử đầu vào sau được cấp cho chương trình: *hello world! 123*

Thì đầu ra sẽ là:

Số chữ cái là: 10
Số chữ số là: 3

Gợi ý:

Trong trường hợp dữ liệu đầu vào được nhập vào chương trình nó nên được giả định là dữ liệu được người dùng nhập vào từ giao diện điều khiển.

Code mẫu:

```
s = input("Nhập câu của bạn: ")
# Bài tập Python 16, Code by TH&THCSTTQuan Chu
d={"DIGITS":0, "LETTERS":0}
for c in s:
    if c.isdigit():
        d["DIGITS"]+=1
    elif c.isalpha():
        d["LETTERS"]+=1
    else:
        pass
print ("Số chữ cái là:", d["LETTERS"])
print ("Số chữ số là:", d["DIGITS"])
```

Bài 9:

Câu hỏi:

Viết một chương trình chấp nhận đầu vào là một câu, đếm chữ hoa, chữ thường.

Giả sử đầu vào là: Quản Trị Mạng

Thì đầu ra là:

- Chữ hoa: 3
- Chữ thường: 8

Gợi ý:

Trong trường hợp dữ liệu đầu vào được nhập vào chương trình nó nên được giả định là dữ liệu được người dùng nhập vào từ giao diện điều khiển.

Code mẫu:

```
s = input("Nhập câu của bạn: ")
```

```
d={"UPPER CASE":0, "LOWER CASE":0}
# Code by TH&THCSTTQuan Chu
for c in s:
    if c.isupper():
        d["UPPER CASE"]+=1
    elif c.islower():
        d["LOWER CASE"]+=1
    else:
        pass
print ("Chữ hoa:", d["UPPER CASE"])
print ("Chữ thường:", d["LOWER CASE"])
```

Bài 10:

Câu hỏi:

Viết một chương trình tính giá trị của a+aa+aaa+aaaa với a là số được nhập vào bởi người dùng.

Giả sử a được nhập vào là 1 thì đầu ra sẽ là: 1234

Gợi ý:

Trong trường hợp dữ liệu đầu vào được nhập vào chương trình nó nên được giả định là dữ liệu được người dùng nhập vào từ giao diện điều khiển.

Code mẫu:

```
a = input("Nhập số a: ")
n1 = int( "%s" % a )
n2 = int( "%s%s" % (a,a) )
n3 = int( "%s%s%s" % (a,a,a) )
n4 = int( "%s%s%s%s" % (a,a,a,a) )
# Bài tập Python 18, Code by TH&THCSTTQuan Chu
print ("Tổng cần tính là: ",n1+n2+n3+n4)
```

Bài 11:

Câu hỏi:

Sử dụng một danh sách để lọc các số lẻ từ danh sách được người dùng nhập vào.

Giả sử đầu vào là: 1,2,3,4,5,6,7,8,9 thì đầu ra phải là: 1,3,5,7,9

Gợi ý:

Trong trường hợp dữ liệu đầu vào được nhập vào chương trình nó nên được giả định là dữ liệu được người dùng nhập vào từ giao diện điều khiển.

Code mẫu:

```
values = input("Nhập dãy số của bạn, cách nhau bởi dấu phẩy: ")
numbers = [x for x in values.split(",") if int(x)%2!=0]
print (",".join(numbers))
```

Bài 12:

Câu hỏi:

Viết chương trình tính số tiền thực của một tài khoản ngân hàng dựa trên nhật ký giao dịch được nhập vào từ giao diện điều khiển.

Định dạng nhật ký được hiển thị như sau:

```
D 100
W 200

(D là tiền gửi, W là tiền rút ra).
```

Giả sử đầu vào được cung cấp là:

- D 300
- D 300
- W 200
- D 100

Thì đầu ra sẽ là: 500

Gợi ý:

Trong trường hợp dữ liệu đầu vào được nhập vào chương trình nó nên được giả định là dữ liệu được người dùng nhập vào từ giao diện điều khiển.

Code mẫu:

```
import sys
netAmount = 0
# Bài tập Python 20, Code by TH&THCSTTQuan Chu
while True:
    s = input("Nhập nhật ký giao dịch: ")
    if not s:
        break
    values = s.split(" ")
    operation = values[0]
    amount = int(values[1])
    if operation=="D":
        netAmount+=amount
    elif operation=="W":
        netAmount-=amount
    else:
        pass
print (netAmount)
```

Bài 13:

Viết chương trình Python nhập một dãy số nguyên, sau đó kiểm tra xem nó có khả đối xứng không? Nếu có, hãy biến đổi nó để được một dãy đối xứng.

Gợi ý:

- Bạn có thể dùng hàm **kiem_tra_doi_xung** để kiểm tra tính đối xứng của dãy số bằng cách so sánh dãy số ban đầu với dãy số đảo ngược. Nếu chúng bằng nhau, dãy số là đối xứng.
- Nếu dãy số chưa đối xứng, hãy thử dùng hàm **sx_doi_xung** để biến đổi dãy số thành dãy số đối xứng: bằng cách điều chỉnh giá trị của một phần tử ở một vị trí sao cho nó khớp với phần tử đối xứng của nó.

Code mẫu:

```
def kiem_tra_doi_xung(lst):
    return lst == lst[::-1]

def sx_doi_xung(lst):
    if kiem_tra_doi_xung(lst):
        return lst
    else:
        for i in range(len(lst)//2):
            if lst[i] != lst[-i-1]:
                for j in range(i+1, len(lst)):
                    if lst[j] == lst[-i-1]:
                        lst[i], lst[j] = lst[j], lst[i]
                        break
        return lst

lst = list(map(int, input("Nhập dãy số nguyên, cách nhau bởi dấu cách: ").split()))

if kiem_tra_doi_xung(lst):
    print("Dãy số đã đối xứng.")
else:
    new_lst = sx_doi_xung(lst)
    if kiem_tra_doi_xung(new_lst):
        print("Dãy số đã khả đối xứng và biến đổi thành dãy số đối xứng: ", new_lst)
    else:
        print("Dãy số không khả đối xứng.")
```

Bài 14:

Viết chương trình Python nhập một mảng hai chiều các số thực A (m hàng, n cột) từ bàn phím.

a. Tìm giá trị lớn nhất và nhỏ nhất trên mỗi cột

b. Tìm phần tử lớn nhất và phần tử nhỏ nhất của mảng A cùng các chỉ số hàng và cột của 2 phần tử này.

c. Trong mảng A có bao nhiêu phần tử bằng phần tử lớn nhất.

Gợi ý:

- Sử dụng thư viện **numpy** để khởi tạo mảng A và thực hiện các phép toán trên mảng.
- Câu lệnh **np.amax(A, axis=0)** trả về giá trị lớn nhất trên mỗi cột của mảng A, còn **np.amin(A, axis=0)** trả về giá trị nhỏ nhất trên mỗi cột.
- Để tìm phần tử lớn nhất và nhỏ nhất của mảng A cùng với chỉ số hàng và cột của 2 phần tử này, chúng ta sử dụng **np.amax(A)** và **np.amin(A)** để tìm giá trị lớn nhất và nhỏ nhất của mảng A, sau đó sử dụng hàm **np.unravel_index** để chuyển đổi chỉ số dạng 1 chiều thành chỉ số dạng 2 chiều.

Code mẫu dùng hàm numpy:

```
import numpy as np

# Nhập mảng A từ bàn phím
m, n = map(int, input("Nhập số hàng và số cột của mảng A (cách nhau bởi dấu cách): ").split())
A = np.zeros((m, n))
for i in range(m):
    row = list(map(float, input(f"Nhập dòng {i+1} của mảng A, cách nhau bởi dấu cách: ").split()))
    A[i, :] = row

# Tìm giá trị lớn nhất và nhỏ nhất trên mỗi cột
max_col = np.amax(A, axis=0)
min_col = np.amin(A, axis=0)
print("Giá trị lớn nhất trên mỗi cột:", max_col)
print("Giá trị nhỏ nhất trên mỗi cột:", min_col)

# Tìm phần tử lớn nhất và phần tử nhỏ nhất của mảng A cùng với chỉ số hàng và cột của 2 phần tử này
max_val = np.amax(A)
max_idx = np.unravel_index(np.argmax(A), A.shape)
min_val = np.amin(A)
min_idx = np.unravel_index(np.argmin(A), A.shape)
print("Phần tử lớn nhất của mảng A là", max_val, "ở dòng", max_idx[0]+1, "cột", max_idx[1]+1)
print("Phần tử nhỏ nhất của mảng A là", min_val, "ở dòng", min_idx[0]+1, "cột", min_idx[1]+1)

# Tính số phần tử bằng phần tử lớn nhất
```

```
count_max = np.count_nonzero(A == max_val)
print("Số phần tử bằng phần tử lớn nhất là:", count_max)
```

Code mẫu dùng các hàm thông thường:

```
# Hàm nhập mảng A từ bàn phím
def input_array(m, n):
    A = []
    for i in range(m):
        row = list(map(float, input(f"Nhập hàng {i+1}, cách nhau bởi dấu cách: ").split()))
        A.append(row)
    return A

# Hàm tìm giá trị lớn nhất và nhỏ nhất trên mỗi cột của mảng A
def find_max_min_per_col(A):
    m, n = len(A), len(A[0])
    max_per_col = [max([A[i][j] for i in range(m)]) for j in range(n)]
    min_per_col = [min([A[i][j] for i in range(m)]) for j in range(n)]
    return max_per_col, min_per_col

# Hàm tìm phần tử lớn nhất và nhỏ nhất của mảng A cùng các chỉ số hàng và cột
def find_max_min_index(A):
    max_val, min_val = A[0][0], A[0][0]
    max_row, max_col, min_row, min_col = 0, 0, 0, 0
    m, n = len(A), len(A[0])
    for i in range(m):
        for j in range(n):
            if A[i][j] > max_val:
                max_val = A[i][j]
                max_row, max_col = i, j
            if A[i][j] < min_val:
                min_val = A[i][j]
                min_row, min_col = i, j
    return max_val, max_row, max_col, min_val, min_row, min_col

# Hàm đếm số phần tử trong mảng A bằng phần tử lớn nhất
def count_max_value(A):
    max_val = max([max(row) for row in A])
    count = 0
    for row in A:
        count += row.count(max_val)
    return count

# Nhập mảng A từ bàn phím
```

```
    m, n = map(int, input("Nhập số hàng và số cột của mảng A, cách nhau bởi dấu cách: ").split())
    A = input_array(m, n)

    # Tìm giá trị lớn nhất và nhỏ nhất trên mỗi cột của mảng A
    max_per_col, min_per_col = find_max_min_per_col(A)
    print("Giá trị lớn nhất trên mỗi cột: ", max_per_col)
    print("Giá trị nhỏ nhất trên mỗi cột: ", min_per_col)

    # Tìm phần tử lớn nhất và nhỏ nhất của mảng A cùng các chỉ số hàng và cột
    max_val, max_row, max_col, min_val, min_row, min_col = find_max_min_index(A)
    print(f"Phần tử lớn nhất: {max_val}, hàng: {max_row+1}, cột: {max_col+1}")
    print(f"Phần tử nhỏ nhất: {min_val}, hàng: {min_row+1}, cột: {min_col+1}")

    # Đếm số phần tử trong mảng
    count = count_max_value(A)
    print(f"Số phần tử bằng giá trị lớn nhất ({max_val}) trong mảng A: {count}")
```

5. Bài tập Python level 3

Bài 1:

Câu hỏi:

Một website yêu cầu người dùng nhập tên người dùng và mật khẩu để đăng ký. Viết chương trình để kiểm tra tính hợp lệ của mật khẩu mà người dùng nhập vào.

Các tiêu chí kiểm tra mật khẩu bao gồm:

1. Ít nhất 1 chữ cái nằm trong [a-z]
2. Ít nhất 1 số nằm trong [0-9]
3. Ít nhất 1 kí tự nằm trong [A-Z]
4. Ít nhất 1 ký tự nằm trong [$ # @]
5. Độ dài mật khẩu tối thiểu: 6
6. Độ dài mật khẩu tối đa: 12

Chương trình phải chấp nhận một chuỗi mật khẩu phân tách nhau bởi dấu phẩy và kiểm tra xem chúng có đáp ứng những tiêu chí trên hay không. Mật khẩu hợp lệ sẽ được in, mỗi mật khẩu cách nhau bởi dấu phẩy.

Ví dụ mật khẩu nhập vào chương trình là: ABd1234@1,a F1#,2w3E*,2We3345

Thì đầu ra sẽ là: ABd1234@1

Gợi ý:

Trong trường hợp dữ liệu đầu vào được nhập vào chương trình nó nên được giả định là dữ liệu được người dùng nhập vào từ giao diện điều khiển.

Code mẫu:

```
import re
value = []
items=[x for x in input("Nhập mật khẩu: ").split(',')]
# Bài tập Python 21, Code by TH&THCSTTQuan Chu
for p in items:
    if len(p)<6 or len(p)>12:
        continue
    else:
        pass
    if not re.search("[a-z]",p):
        continue
    elif not re.search("[0-9]",p):
        continue
    elif not re.search("[A-Z]",p):
        continue
    elif not re.search("[$#@]",p):
        continue
    elif re.search("\s",p):
        continue
    else:
        pass
    value.append(p)
print (",".join(value))
```

Bài 2:

Câu hỏi:

Viết chương trình sắp xếp tuple (name, age, score) theo thứ tự tăng dần, name là string, age và height là number. Tuple được nhập vào bởi người dùng. Tiêu chí sắp xếp là:

Sắp xếp theo name sau đó sắp xếp theo age, sau đó sắp xếp theo score. Ưu tiên là tên > tuổi > điểm.

Nếu đầu vào là:

Tom,19,80
John,20,90
Jony,17,91
Jony,17,93
Json,21,85

Thì đầu ra sẽ là:

[('John', '20', '90'), ('Jony', '17', '91'), ('Jony', '17', '93'), ('Json', '21', '85'), ('Tom', '19', '80')]

Gợi ý:

Trong trường hợp dữ liệu đầu vào được nhập vào chương trình nó nên được giả định là dữ liệu được người dùng nhập vào từ giao diện điều khiển.

Sử dụng itemgetter để chấp nhận nhiều key sắp xếp.

Code mẫu:

```
from operator import itemgetter, attrgetter
# Bài tập Python 22 Code by TH&THCSTTQuan Chu
l = []
while True:
    s = input()
    if not s:
        break
    l.append(tuple(s.split(",")))

print (sorted(l, key=itemgetter(0,1,2)))
```

Bài 3:

Câu hỏi:

Xác định một class với generator có thể lặp lại các số nằm trong khoảng 0 và n, và chia hết cho 7.

Gợi ý:

Sử dụng yield.

Code mẫu:

```
def putNumbers(n):
    i = 0
    while i<n:
        j=i
        i=i+1
        if j%7==0:
            yield j
# Bài tập Python 23 Code by TH&THCSTTQuan Chu
for i in putNumbers (100):
    print (i)
```

Bài 4:

Câu hỏi:

Một Robot di chuyển trong mặt phẳng bắt đầu từ điểm đầu tiên (0,0). Robot có thể di chuyển theo hướng UP, DOWN, LEFT và RIGHT với những bước nhất định. Dấu di chuyển của robot được đánh hiển thị như sau:

UP 5

DOWN 3

LEFT 3

RIGHT 3

Các con số sau phía sau hướng di chuyển chính là số bước đi. Hãy viết chương trình để tính toán khoảng cách từ vị trí hiện tại đến vị trí đầu tiên, sau khi robot đã di chuyển một quãng đường. Nếu khoảng cách là một số thập phân chỉ cần in só nguyên gần nhất.

Ví dụ: Nếu tuple sau đây là input của chương trình:

UP 5
DOWN 3
LEFT 3
RIGHT 2

thì đầu ra sẽ là 2.

Gợi ý:

Trong trường hợp dữ liệu đầu vào được nhập vào chương trình nó nên được giả định là dữ liệu được người dùng nhập vào từ giao diện điều khiển.

Code mẫu:

```
import math
pos = [0,0]
while True:
    s = input()
    if not s:
        break
    movement = s.split(" ")
    direction = movement[0]
    steps = int(movement[1])
    if direction=="UP":
        pos[0]+=steps
    elif direction=="DOWN":
        pos[0]-=steps
    elif direction=="LEFT":
        pos[1]-=steps
    elif direction=="RIGHT":
        pos[1]+=steps
    else:
        pass
# Bài tập Python 24 Code by TH&THCSTTQuan Chu
print (int(round(math.sqrt(pos[1]**2+pos[0]**2))))
```

Bài 5:

Câu hỏi:

Viết chương trình tính tần suất các từ từ input. Output được xuất ra sau khi đã sắp xếp theo bảng chữ cái.

Giả sử input là: New to Python or choosing between Python 2 and Python 3? Read Python 2 or Python 3.

Thì output phải là:

2:2
3.:1
3?:1
New:1
Python:5
Read:1
and:1
between:1
choosing:1
or:2
to:1

Gợi ý:

Trong trường hợp dữ liệu đầu vào được cung cấp cho câu hỏi, nó phải được giả định là một input được nhập từ giao diện điều khiển.

Code mẫu:

```
freq = {} # frequency of words in text
line = input()
for word in line.split():
    freq[word] = freq.get(word,0)+1
    # Bài tập Python 25 Code by TH&THCSTTQuan Chu
words = sorted(freq.keys())
for w in words:
    print ("%s:%d" % (w,freq[w]))
```

6. Bài tập Python khác

Bài 1:

Câu hỏi:

Định nghĩa 1 hàm có thể tính tổng hai số.

Gợi ý:

Định nghĩa 1 hàm với 2 số là đối số. Bạn có thể tính tổng trong hàm và trả về giá trị.

Code mẫu:

```
def SumFunction(number1, number2): #định nghĩa hàm tính tổng
```

```
    return number1+number2
print (SumFunction(5,7)) #in tổng 2 số 5 và 7
```

Bài 2:

Câu hỏi:

Định nghĩa một hàm có thể chuyển số nguyên thành chuỗi và in nó ra giao diện điều khiển

Gợi ý:

Sử dụng str() để chuyển đổi một số thành chuỗi.

Code mẫu:

```
def printValue(n):
    print (str(n))
printValue(3)
```

Bài 3:

Câu hỏi:

Định nghĩa hàm có thể nhận hai số nguyên trong dạng chuỗi và tính tổng của chúng, sau đó in tổng ra giao diện điều khiển.

Gợi ý:

Sử dụng int() để chuyển đổi một chuỗi thành số nguyên.

Code mẫu:

```
def printValue(s1,s2):
    print (int(s1)+int(s2))
printValue("3","4") #Kết quả là 7
```

Bài 4:

Câu hỏi:

Định nghĩa hàm có thể nhận 2 chuỗi từ input và nối chúng sau đó in ra giao diện điều khiển

Gợi ý:

Sử dụng + để nối các chuỗi.

Code mẫu:

```
def printValue(s1,s2):
    print (s1+s2)
printValue("3","4") #Kết quả là 34
```

Bài 5:

Câu hỏi:

Định nghĩa một hàm có input là 2 chuỗi và in chuỗi có độ dài lớn hơn trong giao diện điều khiển. Nếu 2 chuỗi có chiều dài như nhau thì in tất cả các chuỗi theo dòng.

Gợi ý:

Sử dụng hàm len() để lấy chiều dài của một chuỗi

Code mẫu:

```
def printValue(s1,s2):
# Bài tập Python 30 Code by TH&THCSTTQuan Chu
    len1 = len(s1)
    len2 = len(s2)
    if len1>len2:
        print (s1)
    elif len2>len1:
        print (s2)
    else:
        print(s1)
        print (s2)
printValue("one","three")
```

Bài 6:

Câu hỏi:

Định nghĩa hàm có thể chấp nhận input là số nguyên và in "Đây là một số chẵn" nếu nó chẵn và in "Đây là một số lẻ" nếu là số lẻ.

Gợi ý:

Sử dụng toán tử % để kiểm tra xem số đó chẵn hay lẻ.

Code mẫu:

```
def checkValue(n):
    if n%2 == 0:
print ("Đây là một số chẵn")
    else:
       print ("Đây là một số lẻ")
checkValue(7)
```

Bài 7:

Câu hỏi:

Định nghĩa một hàm có thể in dictionary chứa key là các số từ 1 đến 3 (bao gồm cả hai số) và các giá trị bình phương của chúng.

Gợi ý:

- Sử dụng dict[key]=value để nhập mục vào dictionary.
- Sử dụng toán từ ** để lấy bình phương của một số.

Code mẫu:

```
def printDict():
    d=dict()
```

```
    d[1]=1
    d[2]=2**2
    d[3]=3**2
    print (d)
# Bài tập Python 32, Code by TH&THCSTTQuan Chu
printDict()
```

Chạy code trên bạn sẽ được kết quả là một dictionary như sau: {1: 1, 2: 4, 3: 9}. Nếu chưa hiểu lắm về kiểu dữ liệu dictionary này bạn hãy đọc lại bài:

Kiểu dữ liệu trong Python: chuỗi, số, list, tuple, set và dictionary

Bài 8:

Câu hỏi:

Định nghĩa một hàm có thể in dictionary chứa các key là số từ 1 đến 20 (bao gồm cả 1 và 20) và các giá trị bình phương của chúng.

Gợi ý:

- Sử dụng dict[key]=value để nhập mục vào dictionary.
- Sử dụng toán từ ** để lấy bình phương của một số.
- Sử dujnng range() cho các vòng lặp.

Code mẫu:

```
def printDict():
    d=dict()
    for i in range(1,21):
        d[i]=i**2
    print (d)
# Bài tập Python 33, Code by TH&THCSTTQuan Chu
printDict()
```

Kết quả khi chạy code trên là: {1: 1, 2: 4, 3: 9, 4: 16, 5: 25, 6: 36, 7: 49, 8: 64, 9: 81, 10: 100, 11: 121, 12: 144, 13: 169, 14: 196, 15: 225, 16: 256, 17: 289, 18: 324, 19: 361, 20: 400}

Bài 9:

Câu hỏi:

Định nghĩa một hàm có thể tạo dictionary, chứa các key là số từ 1 đến 20 (bao gồm cả 1 và 20) và các giá trị bình phương của chúng. Hàm chỉ in các giá trị mà thôi.

Gợi ý:

- Sử dụng dict[key]=value để nhập mục vào dictionary.
- Sử dụng toán từ ** để lấy bình phương của một số.
- Sử dụng range() cho các vòng lặp.
- Sử dụng keys() để di lặp các key trong dictionary. Có thể sử dụng item() để nhận cặp key/value.

Code mẫu:

```
def printDict():
```

```
    d=dict()
    for i in range(1,21):
 d[i]=i**2
    for (k,v) in d.items():
       print (v)
# Bài tập Python 34, Code by TH&THCSTTQuan Chu
printDict()
```

Kết quả bạn nhận được khi chạy code trên là các giá trị bình phương của số từ 1 đến 20.

Bài 10:

Câu hỏi:

Định nghĩa một hàm có thể tạo ra một dictionary chứa key là những số từ 1 đến 20 (bao gồm cả 1 và 20) và các giá trị bình phương của key. Hàm chỉ cần in các key.

Gợi ý:

Tương tự như bài 34.

Code mẫu:

```
def printDict():
 d=dict()
 for i in range(1,21):
 d[i]=i**2
 for k in d.keys():
       print (k)
# Bài Python 35, Code by TH&THCSTTQuan Chu
printDict()
```

Chạy code trên bạn sẽ nhận được các key trong dictionary, chính là các số từ 1 đến 20.

Bài 11:

Câu hỏi:

Định nghĩa một hàm có thể tạo và in list chứa các giá trị bình phương của các số từ 1 đến 20 (tính cả 1 và 20).

Gợi ý:

- Sử dụng toán tử ** để lấy giá trị bình phương.
- Sử dụng range() cho vòng lặp.
- Sử dụng list.append() để thêm giá trị vào list.

Code mẫu:

```
def printList():
    li=list()
    for i in range(1,21):
       li.append(i**2)
    print (li)
```

```
# Bài Python 36, Code by TH&THCSTTQuan Chu
printList()
```

Chạy code trên bạn sẽ nhận được một list chứa các giá trị bình phương của các số từ 1 đến 20.

Thụt đầu dòng trong Python rất quan trọng, nếu code trên bạn chỉ cần sửa 1 chút như sau:

```
def printList():
    li=list()
    for i in range(1,21):
        li.append(i**2)
        print (li)
# Bài Python 36, Code by TH&THCSTTQuan Chu
printList()
```

Thì sẽ nhận được output hình tháp khá đẹp như này:

```
[1]
[1, 4]
[1, 4, 9]
[1, 4, 9, 16]
[1, 4, 9, 16, 25]
[1, 4, 9, 16, 25, 36]
[1, 4, 9, 16, 25, 36, 49]
[1, 4, 9, 16, 25, 36, 49, 64]
[1, 4, 9, 16, 25, 36, 49, 64, 81]
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100]
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121]
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144]
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169]
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196]
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225]
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256]
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256, 289]
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256, 289, 324]
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256, 289, 324, 361]
[1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256, 289, 324, 361, 400]
>>>
```

Kết quả bài tập Python 36 khi thay đổi thụt đầu dòng

Bài 12:

Câu hỏi:

Định nghĩa một hàm có thể tạo list chứa các giá trị bình phương của các số từ 1 đến 20 (bao gồm cả 1 và 20) và in 5 mục đầu tiên trong list.

Gợi ý:

- Sử dụng toán tử ** để lấy giá trị bình phương.
- Sử dụng range() cho vòng lặp.
- Sử dụng list.append() để thêm giá trị vào list.
- Sử dụng [n1:n2] để cắt list

Code mẫu:

```
def printList():
    li=list()
    for i in range(1,21):
 li.append(i**2)
    print (li[:5])
# Bài Python 37, Code by TH&THCSTTQuan Chu
printList()
```

Chạy code trên bạn sẽ nhận được một list chứa giá trị bình phương của các số từ 1 đến 5.

Bài 13:

Câu hỏi:

Định nghĩa một hàm có thể tạo ra list chứa các giá trị bình phương của các số từ 1 đến 20 (bao gồm cả 1 và 20), rồi in 5 mục cuối cùng trong list.

Gợi ý:

Tương tự bài 37.

Code mẫu:

```
def printList():
    li=list()
    for i in range(1,21):
        li.append(i**2)
    print (li[-5:])
# Bài Python 38, Code by TH&THCSTTQuan Chu
printList()
```

Khi chạy code trên bạn sẽ nhận được list chứa giá trị bình phương của 16, 17, 18, 19, 20.

Bài 14:

Câu hỏi:

Định nghĩa một hàm có thể tạo list chứa giá trị bình phương của các số từ 1 đến 20 (bao gồm cả 1 và 20). Sau đó in tất cả các giá trị của list, trừ 5 mục đầu tiên.

Gợi ý:

Tương tư bài 37, 38.

Code mẫu:

```
def printList():
```

```
    li=list()
    for i in range(1,21):
        li.append(i**2)
    print (li[5:])
printList()
```

Kết quả:

```
[36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256, 289, 324, 361, 400]
```

Bài 15:

Câu hỏi:

Định nghĩa 1 hàm có thể tạo và in một tuple chứa các giá trị bình phương của các số từ 1 đến 20 (tính cả 1 và 20).

Gợi ý:

- Sử dụng toán tử ** để lấy giá trị bình phương.
- Sử dụng range() cho vòng lặp.
- Sử dụng list.append() để thêm giá trị vào list.
- Sử dụng tuple() để lấy giá tuple từ list.

Code mẫu:

```
def printTuple():
    li=list()
    for i in range(1,21):
        li.append(i**2)
    print (tuple(li))

printTuple()
```

Kết quả:

```
(1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256, 289, 324, 361, 400)
```

Bài 16:

Câu hỏi:

Với tuple (1,2,3,4,5,6,7,8,9,10) cho trước, viết một chương trình in một nửa số giá trị đầu tiên trong 1 dòng và 1 nửa số giá trị cuối trong 1 dòng.

Gợi ý:

Sử dụng [n1:n2] để lấy một phần từ tuple.

Code mẫu:

```
tp=(1,2,3,4,5,6,7,8,9,10)
tp1=tp[:5]
tp2=tp[5:]
print (tp1)
print (tp2)
```

Kết quả:

```
(1, 2, 3, 4, 5)
(6, 7, 8, 9, 10)
```

Bài 17:

Câu hỏi:

Viết một chương trình để tạo tuple khác, chứa các giá trị là số chẵn trong tuple (1,2,3,4,5,6,7,8,9,10) cho trước.

Gợi ý:

- Sử dụng for để lặp tuple.
- Sử dụng tuple() để tạo tuple từ list.

Code mẫu:

```
tp=(1,2,3,4,5,6,7,8,9,10)
li=list()
for i in tp:
 if tp[-i]%2==0:
      li.append(tp[i])

tp2=tuple(li)
print (tp2)
```

Kết quả:

```
(2, 4, 6, 8, 10)
```

Bài 18:

Câu hỏi:

Viết một chương trình để tạo ra và in tuple chứa các số chẵn được lấy từ tuple (1,2,3,4,5,6,7,8,9,10).

Gợi ý:

- Sử dụng "for" để lặp lại tuple.
- Sử dụng tuple() để tạo ra một tuple từ một danh sách.

Code mẫu:

```
tp=(1,2,3,4,5,6,7,8,9,10)
li=list()
for i in tp:
   if tp[i-1]%2==0:
      li.append(tp[i-1])
      tp2=tuple(li)
print (tp2)
```

Kết quả:

(2, 4, 6, 8, 10)

Bài 19:

Yêu cầu:

Viết một chương trình Python nhận chuỗi nhập vào bởi người dùng, in "Yes" nếu chuỗi là "yes" hoặc "YES" hoặc "Yes", nếu không in "No".

Gợi ý:

- Sử dụng lệnh if để kiểm tra điều kiện.

Code mẫu:

```
s = input ("Nhập chuỗi: ")
if s == "yes" or s == "YES" or s == "Yes":
   print ("Yes")
else:
    print ("No")
```

Bài 20:

Yêu cầu:

Viết chương trình Python có thể lọc các số chẵn trong danh sách sử dụng hàm filter. Danh sách là [1,2,3,4,5,6,7,8,9,10].

Gợi ý:

- Sử dụng filter() để lọc các yếu tố trong một list.
- Sử dụng lambda để định nghĩa hàm chưa biết.

Code mẫu:

```
li = [1,2,3,4,5,6,7,8,9,10]
evenNumbers = list(filter (lambda x: x% 2 == 0, li))
print (evenNumbers)
```

Kết quả:

[2, 4, 6, 8, 10]

Lưu ý: Trong các phiên bản Python trước, bạn chỉ cần dùng hàm filter sẽ được trả kết quả đầu ra là một danh sách. Nhưng từ Python 3, phải dùng list(filter()) thì kết quả trả về mới là list. Điều này cũng áp dụng với map().

Bài 21:

Yêu cầu:

Viết chương trình Python dùng map() để tạo list chứa các giá trị bình phương của các số trong [1,2,3,4,5,6,7,8,9,10].

Gợi ý:

- Sử dụng map() để tạo list.
- Sử dụng lambda để định nghĩa hàm chưa biết.

Code mẫu:

```
li = [1,2,3,4,5,6,7,8,9,10]
squaredNumbers = list(map (lambda x: x ** 2, li))
print (squaredNumbers)
```

Kết quả:

[1, 4, 9, 16, 25, 36, 49, 64, 81, 100]

Bài 22:

Yêu cầu:

Viết chương trình Python dùng map() và filter() để tạo list chứa giá trị bình phương của các số chẵn trong [1,2,3,4,5,6,7,8,9,10].

Gợi ý:

- Dùng map() để tạo list.
- Dùng filter() để lọc thành phần trong list.
- Dùng lambda để định nghĩa hàm chưa biết.

Code mẫu:

```
li = [1,2,3,4,5,6,7,8,9,10]
squareOfEvenNumbers = list (map (lambda x: x ** 2, filter (lambda x: x% 2 == 0, li)))
print (squareOfEvenNumbers)
```

Kết quả:

[4, 16, 36, 64, 100]

Bài 23:

Yêu cầu:

Viết chương trình Python dùng filter() để tạo danh sách chứa các số chẵn trong đoạn [1,20].

Gợi ý:

- Giống bài 45.

Code mẫu:

```
evenNumbers = list(filter (lambda x: x% 2 == 0, range (1,21)))
print (evenNumbers)
```

Kết quả:

[2, 4, 6, 8, 10, 12, 14, 16, 18, 20]

Bài 24:

Yêu cầu:

Viết chương trình Python sử dụng map() để tạo list chứa giá trị bình phương của các số trong đoạn [1,20].

Gợi ý:

- Giống bài 46.

Code mẫu:

```
squaredNumbers = list(map(lambda x: x ** 2, range (1,21)))
print (squaredNumbers)
```

Kết quả:

[1, 4, 9, 16, 25, 36, 49, 64, 81, 100, 121, 144, 169, 196, 225, 256, 289, 324, 361, 400]

Bài 25:

Yêu cầu:

Định nghĩa một class có tên là Vietnam, với static method là printNationality.

Gợi ý:

Sử dụng @staticmethod để định nghĩa class với static method.

Code mẫu:

```
class Vietnam (object):
    @staticmethod
    def printNationality ():
        print ("Vietnam")
# Bài Python 50, Code by TH&THCSTTQuan Chu
VietnamVodich = Vietnam ()
VietnamVodich.printNationality ()
Vietnam.printNationality ()
```

Bài 26:

Yêu cầu:

Định nghĩa một class tên Vietnam và class con của nó là Hanoi.

Gợi ý:

- Sử dụng `Subclass(ParentClass)` để định nghĩa một class con.

Code mẫu:

```
class Vietnam(object):
    pass

class Hanoi(Vietnam):
    pass
# Bài Python 51, Code by TH&THCSTTQuan Chu
VietnamVodich = Vietnam()
NguoiHanoi = Hanoi()
print (VietnamVodich)
print (NguoiHanoi)
```

Bài 27:

Yêu cầu:

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

Gợi ý:

Sử dụng `def methodName(self)` để định nghĩa method.

Code mẫu:

```
class Circle(object):
    def __init__(self, r):
        self.radius = r
# Bài Python 52, Code by TH&THCSTTQuan Chu
    def area(self):
        return self.radius**2*3.14

aCircle = Circle(2)
print (aCircle.area())
```

Trong code trên, ta thực hiện khai báo lớp Circle, và method tính diện tích cho hình tròn, với bán kính r=2, kết quả khi chạy code sẽ được là: 12.56.

Bài 28:

Yêu cầu:

Định nghĩa class có tên là Hinhchunhat được xây dựng bằng chiều dài và chiều rộng. Class Hinhchunhat có method để tính diện tích.

Gợi ý:

- Như bài 52.

Code mẫu:

```
class Hinhchunhat(object):
    def __init__(self, l, w):
        self.dai = l
        self.rong = w
# Bài Python 53, Code by TH&THCSTTQuan Chu
    def area(self):
        return self.dai*self.rong

aHinhchunhat = Hinhchunhat(10,2)
print (aHinhchunhat.area())
```

Trong code trên chiều dài hình nhữ nhật là 10, chiều rộng là 2. Chạy code ta được kết quả là 20.

Bài 29:

Yêu cầu:

Định nghĩa một class có tên là Shape và class con là Square. Square có hàm init để lấy đối số là chiều dài. Cả 2 class đều có hàm area để in diện tích của hình, diện tích mặc định của Shape là 0.

Gợi ý:

Để ghi đè một method trong super class, chúng ta có thể định nghĩa một method có cùng tên trong super class.

Code mẫu:

```
class Shape(object):
    def __init__(self):
        pass

    def area(self):
        return 0
# Bài Python 54, Code by TH&THCSTTQuan Chu
class Square(Shape):
    def __init__(self, l):
        Shape.__init__(self)
        self.length = l

    def area(self):
        return self.length*self.length

aSquare= Square(3)
print (aSquare.area())
```

Với chiều dài là 3, khi chạy code trên ta được kết quả là 9.

Bài 30:

Yêu cầu:

Đưa ra một RuntimeError exception.

Gợi ý:

- Sử dụng raise() để đưa ra exception.

Code mẫu:

Code đơn giản:

```
raise RuntimeError('something wrong')
```

Code phức tạp:

```
# Bài Python 55, Code by TH&THCSTTQuan Chu
class RuntimeError(Exception):
    def __init__(self, mismatch):
        Exception.__init__(self, mismatch)
try:
    print ("And now, the Vocational Guidance Counsellor Sketch.")
    raise RuntimeError("Does not have proper hat")
    print ("This print statement will not be reached.")
except RuntimeError as problem:
    print ("Vocation problem: {0}".format(problem))
```

Bài 31:

Yêu cầu:

Viết hàm để tính 5/0 và sử dụng try/exception để bắt lỗi.

Gợi ý:

- Sử dụng try/exception để bắt lỗi.

Code mẫu:

```
def throws():
    return 5/0
# Bài Python 56, Code by TH&THCSTTQuan Chu
try:
    throws()
except ZeroDivisionError:
    print ("Chia một số cho 0!")
except Exception as problem:
    print ('Bắt được một exception')
finally:
    print ('Phép tính bị hủy')
```

Kết quả khi chạy code trên ta nhận được như sau:

```
RESTART: C:/Users/[redacted]/AppData/Local/Programs/P
ython/Python36-32/Quantrimang.com.py
Chia một số cho 0!
Phép tính bị hủy
>>>
```

Bài 32:

Yêu cầu:

Định nghĩa một class exception tùy chỉnh, nhận một thông báo là thuộc tính.

Gợi ý:

- Để định nghĩa một class exception tùy chỉnh, chúng ta phải định nghĩa một class kế thừa từ Exception.

Code mẫu:

```
class MyError(Exception):
    """My own exception class
# Bài Python 57, Code by TH&THCSTTQuan Chu
    Attributes:
      msg -- explanation of the error
    """

    def __init__(self, msg):
      self.msg = msg

error = MyError("Có gì đó sai sai!")
```

```
print (error)
```

Khi chạy code trên, thông báo "Có gì đó sai sai!" sẽ được in ra màn hình.

Bài 33:

Yêu cầu:

Giả sử rằng chúng ta có vài địa chỉ email dạng username@companyname.com, hãy viết một chương trình để in username của địa chỉ email cụ thể. Cả username và companyname chỉ bao gồm chữ cái.

Ví dụ: Nếu cung cấp địa chỉ email QTM@TH&THCSTTQuan Chu thì đầu ra sẽ là: QTM.

Trong trường hợp dữ liệu đầu vào không có sẵn, ta giả định nó được người dùng nhập vào từ giao diện điều khiển.

Gợi ý:

Sử dụng \w để kiểm tra chữ cái.

Code mẫu:

```
# Bài Python 58, Code by TH&THCSTTQuan Chu
import re
emailAddress = input()
pat2 = "(\w+)@((\w+\.)+(com))"
re2 = re.match(pat2,emailAddress)
print (re2.group(1))
```

Khi chạy code trên, nó sẽ nhận email được nhập vào bởi người dùng và trả về username của email, nếu nhập vào qtm@TH&THCSTTQuan Chu bạn sẽ nhận được kết quả là qtm.

Bài 34:

Yêu cầu:

Tương tự như bài 58, nhưng lần này ta sẽ viết hàm để lấy companyname.

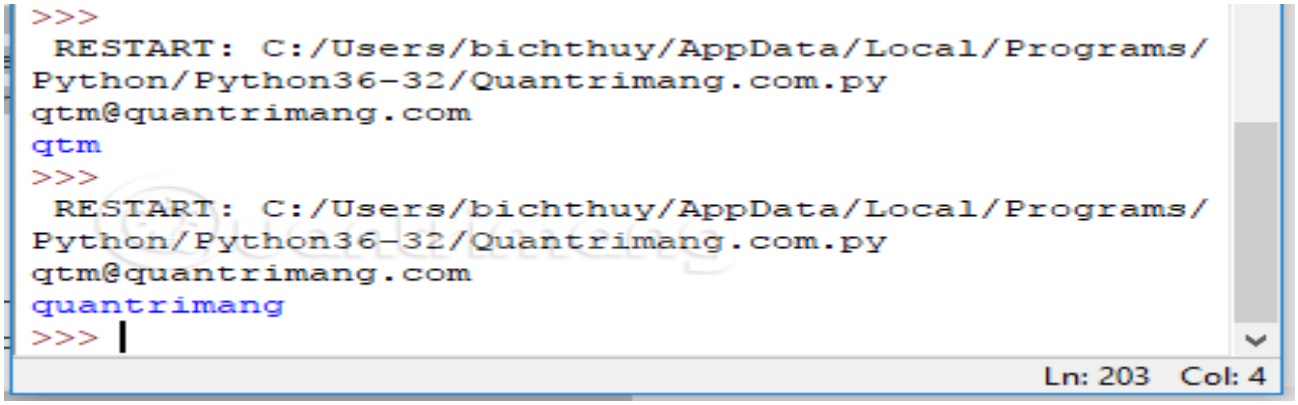
Gợi ý:

- Giống bài 58.

Code mẫu:

```
# Bài Python 59, Code by TH&THCSTTQuan Chu
import re
emailAddress = input()
pat2 = "(\w+)@(\w+)\.(com)"
r2 = re.match(pat2,emailAddress)
print (r2.group(2))
```

Đây là kết quả khi chạy code bài 58 và 59:



Bài 35:

Yêu cầu:

Viết một chương trình chấp nhận chuỗi từ được phân tách bằng khoảng trống và in các từ chỉ gồm chữ số.

Ví du: Nếu những từ sau đây là đầu vào của chương trình: 3 TH&THCSTTQuan Chu và 2 python. Đầu ra sẽ là ['3', '2']

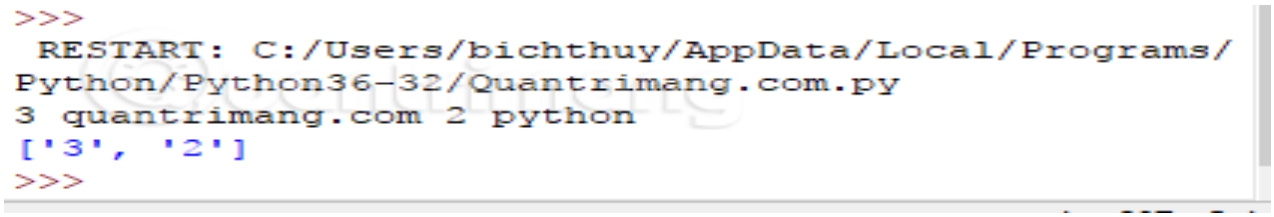
Gợi ý:

- Sử dụng `re.findall()` để tìm tất cả chuỗi con sử dụng regex (biểu thức tiêu chuẩn).

Code mẫu:

```
# Bài Python 60, Code by TH&THCSTTQuan Chu
import re
s = input()
print (re.findall("\d+",s))
```

Kết quả khi chạy code trên sẽ như sau:



Bài 36:

Yêu cầu:

In chuỗi Unicode "Hello world".

Gợi ý:

- Sử dụng định dạng u'string' để định nghĩa chuỗi Unicode.

Code mẫu:

```
# Bài Python 61, Code by TH&THCSTTQuan Chu
unicodeString = u"Hello world!"
print (unicodeString)
```

Bài 37:

Yêu cầu:

Viết chương trình để đọc chuỗi ASCII và chuyển đổi nó sang một chuỗi Unicode được mã hóa bằng UTF-8.

Gợi ý:

- Sử dụng hàm `encode()` để chuyển đổi.

Code mẫu:

```
# Bài Python 62, Code by TH&THCSTTQuan Chu
s = input()
v = s.encode() # có thể dùng v=s.encode('utf-8')
print (v)
```

Bài 38:

Yêu cầu:

Viết comment đặc biệt để chỉ định file code nguồn Python ở Unicode.

Code mẫu:

```
# -*- coding: utf-8 -*-
#Bài Python 63, Code by TH&THCSTTQuan Chu
```

Bài 39:

Yêu cầu:

Viết một chương trình tính 1/2 + 2/3 + 3/4 + ... + n/(n + 1) với một n là số được nhập vào (n> 0).

Ví dụ, nếu n là số sau đây được nhập vào:

5

Thì đầu ra phải là:

3.55

Gợi ý:

- Sử dụng float() để chuyển số nguyên sang số thập phân.

Code mẫu:

```
#Bài Python 64, Code by TH&THCSTTQuan Chu
n=int(input("Nhập số n >0: "))
sum=0.0
for i in range(1,n+1):
    sum += float(float(i)/(i+1))
print (sum)
```

Bài 40:

Yêu cầu:

Viết chương trình tính: f(n)=f(n-1)+100 khi n>0 và f(0)=1, với n là số được nhập vào (n>0).

Ví dụ: Nếu n được nhập vào là 5 thì đầu ra phải là 500.

Gợi ý:

- Chúng ta có thể định nghĩa hàm đệ quy trong Python.

Code mẫu:

```
def f(n):
    if n==0:
        return 0
    else:
        return f(n-1)+100
#Bài Python 65, Code by TH&THCSTTQuan Chu
n=int(input("Nhập số n>0: "))
print (f(n))
```

Bài 41:

Yêu cầu:

Dãy Fibonacci được tính dựa trên công thức sau:

f(n)=0 nếu n=0

f(n)=1 nếu n=1

f(n)=f(n-1)+f(n-2) nếu n>1

Hãy viết chương trình tính giá trị của f(n) với n là số được người dùng nhập vào. Ví dụ: Nếu n được nhập vào là 7 thì đầu ra của chương trình sẽ là 13.

Gợi ý:

Tương tự như bài 65, ta cũng sử dụng hàm đệ quy trong Python.

Code mẫu:

```
def f(n):
    if n == 0: return 0
    elif n == 1: return 1
    else: return f(n-1)+f(n-2)
#Bài Python 66, Code by TH&THCSTTQuan Chu
n=int(input("Nhập số n: "))
print (f(n))
```

Bài 42:

Yêu cầu:

Dãy Fibonacci được tính dựa trên công thức sau:

- f(n)=0 nếu n=0
- f(n)=1 nếu n=1
- f(n)=f(n-1)+f(n-2) nếu n>1

Hãy viết chương trình sử dụng list comprehension để in dãy Fibonacci dưới dạng tách biệt bằng dấu ",", n được người dùng nhập vào.

Ví dụ: Nếu n được nhập vào là 7 thì đầu ra của chương trình sẽ là: 0,1,1,2,3,5,8,13

Gợi ý:

- Chúng ta có thể định nghĩa hàm đệ quy trong Python.
- Sử dụng list comprehension để tạo ra list từ list hiện có.
- Sử dụng string.join() để nối danh sách các chuỗi.

Code mẫu:

```
def f(n):
    if n == 0: return 0
    elif n == 1: return 1
    else: return f(n-1)+f(n-2)
#Bài Python 67, Code by TH&THCSTTQuan Chu
n=int(input("Nhập số n: "))
values = [str(f(x)) for x in range(0, n+1)]
print (",".join(values))
```

Bài 43:

Yêu cầu:

Viết chương trình sử dụng generator để in số chẵn trong khoảng từ 0 đến n, cách nhau bởi dấu phẩy, n là số được nhập vào.

Ví dụ nếu n=10 được nhập vào thì đầu ra của chương trình là: 0,2,4,6,8,10

Gợi ý:

- Sử dụng yield để tạo ra giá trị kết tiếp trong generator.

Code mẫu:

```
def EvenGenerator(n):
    i=0
    while i<=n:
        if i%2==0:
            yield i
        i+=1

# Bài tập Python 68, Code by TH&THCSTTQuan Chu
n=int(input("Nhập n: "))
values = []
for i in EvenGenerator(n):
    values.append(str(i))

print ("Các số chẵn trong khoảng 0 và n là: ",",".join(values))
```

Bài 44:

Yêu cầu:

Viết chương trình sử dụng generator để in số chia hết cho 5 và 7 giữa 0 và n, cách nhau bằng dấu phẩy, n được người dùng nhập vào.

Ví dụ: Nếu n=100 được nhập vào thì đầu ra của chương trình là: 0,35,70.

Gợi ý:

Như bài 68.

Code mẫu:

```
def NumGenerator(n):
    for i in range(n+1):
        if i%5==0 and i%7==0:
            yield i
# Bài tập Python 69, Code by TH&THCSTTQuan Chu
n=int(input("Nhập n: "))
values = []
for i in NumGenerator(n):
    values.append(str(i))

print ("Các số chia hết cho 5 và 7 trong khoảng 0 và n là: ",",".join(values))
```

Bài 45:

Yêu cầu:

Viết các lệnh assert để xác minh rằng tất cả các số trong list [2,4,6,8] là chẵn.

Gợi ý:

- Sử dụng assert để khẳng định.

Code mẫu:

```
li = [2,4,6,8]
for i in li:
 assert i%2==0
# Code by TH&THCSTTQuan Chu
```

Bài 46:

Yêu cầu:

Viết chương trình chấp nhận biểu thức toán học cơ bản do người dùng nhập vào từ bảng điều khiển và in kết quả ước lượng ra ngoài màn hình.

Ví dụ: Nếu chuỗi sau là đầu vào của chương trình:

35 + 3

thì đầu ra sẽ lả:

38

Gợi ý:

- Sử dụng eval() để ước lượng biểu thức

Code mẫu:

```
expression = input("Nhập biểu thức cần tính: ")
# Code by TH&THCSTTQuan Chu
print (eval(expression))
```

Mấy bài này khá đơn giản và kết quả đầu ra cũng dễ hình dung nên mình không chụp kết quả nữa nhé, code thì test trên Python 3.6.2 đảm bảo chạy ngon rồi.

Bài 47:

Yêu cầu:

Viết hàm tìm kiếm nhị phân để tìm các item trong một list đã được sắp xếp. Hàm sẽ trả lại chỉ số của phần tử được tìm thấy trong list.

Gợi ý:

- Sử dụng if/elif để giải quyết các điều kiện.

Code mẫu:

```
import math
def bin_search(li, element):
    bottom = 0
    top = len(li)-1
    index = -1
    while top>=bottom and index==-1:
        mid = int(math.floor((top+bottom)/2.0))
        if li[mid]==element:
            index = mid
        elif li[mid]>element:
            top = mid-1
        else:
            bottom = mid+1
    return index
# Code by TH&THCSTTQuan Chu
li=[2,5,7,9,11,17,222]
print (bin_search(li,11))
print (bin_search(li,12))
```

Khi chạy code trên ta sẽ có kết quả đầu ra là 4 và -1, 4 là vị trí của 11 trong list li, và -1 nói lên rằng không có số 12 trong list li.

Bài 48:

Yêu cầu:

Tạo một số thập phân ngẫu nhiên, có giá trị nằm trong khoảng từ 10 đến 100 bằng cách sử dụng module math của Python.

Gợi ý:

- Sử dụng random.random() để tạo float ngẫu nhiên trong [0,1].

Code mẫu:

```
import random
print (random.random()*100)
# Code by TH&THCSTTQuan Chu
```

Vì hàm trên được sử dụng để tạo số thập phân ngẫu nhiên, nằm trong khoảng từ 10 đến 100, nên mỗi lần bạn chạy code sẽ cho ra một kết quả khác nhau, là các số thập phân ngẫu nhiên thỏa mãn điều kiện nằm trong khoảng 10 đến 100.

Bài 49:

Yêu cầu:

Tạo một số thập phân ngẫu nhiên, có giá trị nằm trong khoảng 5 đến 95, sử dụng module math của Python.

Gợi ý:

- Giống bài 73.

Code mẫu:

```
import random
print (random.random()*100-5)
# Code by TH&THCSTTQuan Chu
```

Code bài 73, 74 mình thấy chưa chuẩn lắm, mong nhận được góp ý của các bạn ở phần bình luận nhé!

Bài 50:

Yêu cầu:

Viết chương trình xuất ra một số chẵn ngẫu nhiên trong khoảng 0 đến 10 (bao gồm cả 0 và 10), sử dụng module random và list comprehension.

Gợi ý:

- Sử dụng random.choice() để tạo một phần tử ngẫu nhiên từ list.

Code mẫu:

```
import random
print (random.choice([i for i in range(11) if i%2==0]))
# Code by TH&THCSTTQuan Chu
```

Bài 51:

Yêu cầu:

Vui lòng viết chương trình để xuất một số ngẫu nhiên, chia hết cho 5 và 7, từ 0 đến 200 (gồm cả 0 và 200), sử dụng module random và list comprehension.

Gợi ý:

- Giống bài 75.

Code mẫu:

```
import random
print (random.choice([i for i in range(201) if i%5==0 and i%7==0]))
#Code by TH&THCSTTQuan Chu
```

Khi chạy code trên, bạn sẽ nhận được kết quả đầu ra là số bất kỳ, nằm trong đoạn [0;200] chia hết cho cả 5 và 7.

Bài 52:

Yêu cầu:

Vui lòng viết chương trình để tạo một list với 5 số ngẫu nhiên từ 100 đến 200.

Gợi ý:

- Sử dụng random.sample() để tạo list chứa các giá trị ngẫu nhiên.

Code mẫu:

```
import random
print (random.sample(range(100,201), 5))
#Code by TH&THCSTTQuan Chu
```

Khi chạy code trên bạn sẽ nhận được 1 list, có 5 giá trị ngẫu nhiên, nằm trong đoạn [100;200]. Nếu đề bài yêu cầu số ngẫu nhiên nằm trong đoạn [0;100] thì range() trong đoạn trên bạn chỉ cần viết là range(100).

Bài 53:

Yêu cầu:

Viết chương trình tạo ngẫu nhiên list gồm 5 số chẵn nằm trong đoạn [100;200].

Gợi ý:

- Giống bài 77.

Code mẫu:

```
import random
print (random.sample([i for i in range(100,201) if i%2==0], 5))
#Code by TH&THCSTTQuan Chu
```

Bài 54:

Yêu cầu:

Viết chương trình để tạo ngẫu nhiên một list gồm 5 số, chia hết cho 5 và 7, nằm trong đoạn [1;1000].

Gợi ý:

- Giống bài 77, 78.

Code mẫu:

```
import random
print (random.sample([i for i in range(1,1001) if i%5==0 and i%7==0], 5))
#Code by TH&THCSTTQuan Chu
```

Bài 55:

Yêu cầu:

Viết chương trình để in một số nguyên ngẫu nhiên từ 7 đến 15.

Gợi ý:

- Sử dụng random.randrange() để lấy số nguyên ngẫu nhiên trong một phạm vi nhất định.

Code mẫu:

```
import random
print (random.randrange(7,16))
#Code by TH&THCSTTQuan Chu
```

Bài 56:

Yêu cầu:

Viết chương trình để nén và giải nén string ""hello world!hello world!hello world!hello world!".

Gợi ý:

- Sử dụng zlib.compress() và zlib.decompress() để nén và giải nén string.

Code mẫu:

Với Python 2, code mẫu sẽ như sau:

```
import zlib
s = "hello world!hello world!hello world!hello world!"
t = zlib.compress(s)
print t
print zlib.decompress(t)
```

Tuy nhiên, trong Python 3, bạn phải gọi encode() và chỉ định kiểu mã hóa, giả sử là utf-8 thì yêu cầu trên sẽ được code như sau:

```
import zlib
s = "hello world!hello world!hello world!hello world!"
t = zlib.compress(s.encode("utf-8"))
print (t)
print (zlib.decompress(t))
#Code by TH&THCSTTQuan Chu
```

Bài 57:

Yêu cầu:

Bạn hãy viết một chương trình để in thời gian thực thi (running time of execution) phép tính "1+1" 100 lần.

Gợi ý:

- Sử dụng timeit() để đo thời gian chạy

Code mẫu:

```
from timeit import Timer
t = Timer("for i in range(100):1+1")
print (t.timeit())
```

Khi chạy code trên, bạn cần phải đợi để phép tính trên được thực hiện xong rồi chương trình mới in ra thời gian thực thi. Ban đầu khi mới chạy code, cảm giác như không có gì đang được thực thi.

Bài 58:

Yêu cầu:

Viết chương trình để trộn và in list [3,6,7,8].

Gợi ý:

- Sử dụng shuffle() để trộn list.

Code mẫu:

```
from random import shuffle
li = [3,6,7,8]

#Code by TH&THCSTTQuan Chu
shuffle(li)
print (li)
```

Khi code được thực thi, mỗi lần chạy sẽ cho ra một list với thứ tự các số được trộn ngẫu nhiên.

Bài 59:

Yêu cầu:

Viết một chương trình để tạo tất cả các câu có chủ ngữ nằm trong ["Anh","Em"], động từ nằm trong ["Chơi","Yêu"] và tân ngữ là ["Bóng đá","Xếp hình"].

Gợi ý:

- Sử dụng list[index] để lấy phần tử từ list.

Code mẫu:

```
chu_ngu=["Anh","Em"]
dong_tu=["Chơi","Yêu"]
tan_ngu=["Bóng đá","Xếp hình"]
# Code by TH&THCSTTQuan Chu
for i in range(len(chu_ngu)):
    for j in range(len(dong_tu)):
        for k in range(len(tan_ngu)):
            cau = "%s %s %s." % (chu_ngu[i], dong_tu[j], tan_ngu[k])
            print (cau)
```

Khi chạy code trên ta sẽ có kết quả như sau:

```
Anh Chơi Bóng đá.

Anh Chơi Xếp hình.

Anh Yêu Bóng đá.

Anh Yêu Xếp hình.

Em Chơi Bóng đá.

Em Chơi Xếp hình.

Em Yêu Bóng đá.

Em Yêu Xếp hình.
```

Bài 60:

Yêu cầu:

Viết chương trình in list sau khi xóa các số chẵn trong [5,6,77,45,22,12,24].

Gợi ý:

- Sử dụng list comprehension để xóa một loạt phần tử của list.

Code mẫu:

```
li = [5,6,77,45,22,12,24]
# Code by TH&THCSTTQuan Chu
li = [x for x in li if x%2!=0]
print (li)
```

Kết quả khi chạy code trên sẽ là:

```
[5, 77, 45]
```

Bài 61:

Yêu cầu:

Sử dụng list comprehension để viết chương trình in list sau khi đã loại bỏ các số chia hết cho 5 và 7 trong [12,24,35,70,88,120,155].

Gợi ý:

- Giống bài 85.

Code mẫu:

```
li = [12,24,35,70,88,120,155]
# Code by TH&THCSTTQuan Chu
li = [i for i in li if i%5!=0 or i%7!=0]
print (li)
```

Ta sẽ có kết quả như sau:

```
[12, 24, 88, 120, 155]
```

Bài 62:

Yêu cầu:

Viết chương trình in list sau khi đã xóa số thứ 0, thứ 2, thứ 4, thứ 6 trong [12,24,35,70,88,120,155].

Gợi ý:

- Sử dụng list comprehension để xóa một loạt phần tử trong list.
- Sử dụng hàm enumerate() để lấy index, value của tuple.

Code mẫu:

```
li = [12,24,35,70,88,120,155]
# Code by TH&THCSTTQuan Chu
a= [x for i,x in enumerate(li)if i%2!=0]
print (a)
```

Code trên sẽ trả về kết quả:

```
[24, 70, 120]
```

Bài 63:

Yêu cầu:

Viết chương trình tạo mảng 3D 3*5*8 có mỗi phần tử là 0.

Gợi ý:

- Sử dụng list comprehension để tạo mảng.

Code mẫu:

```
array = [[ [0 for col in range(8)] for col in range(5)] for row in range(3)]
print (array)
```

Kết quả:

```
[[[0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0]], [[0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0]], [[0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0],
[0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0], [0, 0, 0, 0, 0, 0, 0, 0]]]
```

Bài 64:

Yêu cầu:

Viết chương trình in list sau khi đã xóa số ở vị trí thứ 0, thứ 5, thứ 5 trong [12,24,35,70,88,120,155].

Gợi ý:

- Giống bài 87.

Code mẫu:

```
li = [12,24,35,70,88,120,155]
li = [x for (i,x) in enumerate(li) if i not in (0,4,5)]
print (li)
```

Kết quả:

```
[24, 35, 70, 155]
```

Bài 65:

Yêu cầu:

Viết chương trình in list sau khi đã xóa giá trị 24 trong [12,24,35,24,88,120,155].

Gợi ý:

- Sử dụng phương thức xóa của list để xóa giá trị.

Code mẫu:

```
li = [12,24,35,24,88,120,155]

#Code by TH&THCSTTQuan Chu
li = [x for x in li if x!=24]
print (li)
```

Kết quả:

```
[12, 35, 88, 120, 155]
```

Bài 66:

Yêu cầu:

Với 2 list cho trước: [1,3,6,78,35,55] và [12,24,35,24,88,120,155], viết chương trình để tạo list có phần tử là giao của 2 list đã cho.

Gợi ý:

Sử dụng set() và "&=" để thiết lập điểm giao.

Code mẫu:

```
list1=set([12,3,6,78,35,55,120])
list2=set([12,24,35,24,88,120,155])
# Code by TH&THCSTTQuan Chu
list1 &= list2
li=list(list1)
print (li)
```

Kết quả:

```
[120, 35, 12]
```

Bài 67:

Yêu cầu:

Viết chương trình in list từ list [12,24,35,24,88,120,155,88,120,155], sau khi đã xóa hết các giá trị trùng nhau.

Gợi ý:

- Sử dụng set() để lưu trữ các giá trị không bị trùng lặp.

Code mẫu:

```
def xoaTrung( li ):
```

```
    list_moi=[]
    xem = set()
    for i in li:
        if i not in xem:
            xem.add( i )
            list_moi.append(i)
            # Code by TH&THCSTTQuan Chu
    return list_moi

li=[12,12,15,24,35,35,24,88,120,155,88,120,155]
print ("List sau khi xóa giá trị trùng là:",xoaTrung(li))
```

Kết quả:

```
List sau khi xóa giá trị trùng là: [12, 15, 24, 35, 88, 120, 155]
```

Bài 68:

Yêu cầu:

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

Gợi ý:

- Sử dụng Subclass(Parentclass) để định nghĩa 1 class con.

Code mẫu:

```
class Nguoi(object):
    def getGender(self):
        return "Unknown"

class Nam(Nguoi):
    def getGender(self):
        return "Nam"
# Code by TH&THCSTTQuan Chu
class Nu(Nguoi):
    def getGender(self):
        return "Nữ"

aNam = Nam()
aNu= Nu()
print (aNam.getGender())
print (aNu.getGender())
```

Kết quả:

```
Nam

Nữ
```

Bài 69:

Yêu cầu:

Viết chương trình đếm và in số ký tự của chuỗi do người dùng nhập vào.

Ví dụ:

Nếu chuỗi nhập vào là TH&THCSTTQuan Chu thì đầu ra sẽ là:

q,1
u,1
a,2
n,2
t,1
r,1
i,1
m,2
g,1
.,1
c,1
o,1

Gợi ý:

- Sử dụng dict để lưu trữ các cặp key/value.
- Sử dụng dict.get() để tra cứu key với giá trị mặc định.

Code mẫu:

```
dic = {}

chuoi=input("Nhập chuỗi cần đếm ký tự: ")
# Code by TH&THCSTTQuan Chu
for c in chuoi:
    dic[c] = dic.get(c,0)+1
print ('\n'.join(['%s,%s' % (k, v) for k, v in dic.items()]))
```

Kết quả:

```
Nhập chuỗi cần đếm ký tự: TH&THCSTTQuan Chu

q,1
u,1
a,2
n,2
t,1
r,1
i,1
m,2
g,1
.,1
```

```
c,1
o,1
```

Bài 70:

Yêu cầu:

Viết chương trình nhận chuỗi đầu vào từ giao diện điều khiển và in nó theo thứ tự ngược lại.

Ví dụ nếu chuỗi nhập vào là:

i love you

Thì kết quả đầu ra là:

uoy evol i

Gợi ý:

- Sử dụng list[::-1] để lặp list theo thứ tự ngược lại.

Code mẫu:

```
chuoi=input("Nhập chuỗi vào đây: ")
# Code by TH&THCSTTQuan Chu
chuoi = chuoi[::-1]
print (chuoi)
```

Bài 71:

Yêu cầu:

Viết chương trình nhận chuỗi do người dùng nhập vào và in các ký tự có chỉ số chẵn.

Ví dụ: Nếu chuỗi sau được nhập vào: q1u2a3n4t5r6i7m8a9n4g5.6c7o8m, thì đầu ra sẽ là: TH&THCSTTQuan Chu.

Gợi ý:

- Sử dụng list[::2] để lặp list cách 2 vị trí.

Code mẫu:

```
chuoi=input("Nhập chuỗi vào đây: ")
# Code by TH&THCSTTQuan Chu
chuoi = chuoi[::2]
print (chuoi)
```

Kết quả:

```
Nhập chuỗi vào đây: q1u2a3n4t5r6i7m8a9n4g5.6c7o8m
TH&THCSTTQuan Chu
```

Bài 72:

Yêu cầu:

Viết chương trình in tất cả các hoán vị của [1,2,3].

Gợi ý:

- Sử dụng **itertools.permutations()** để lấy hết các hoán vị của list.

Code mẫu:

```
import itertools
print (list(itertools.permutations([1,2,3])))
# Code by TH&THCSTTQuan Chu
```

Kết quả:

```
[(1, 2, 3), (1, 3, 2), (2, 1, 3), (2, 3, 1), (3, 1, 2), (3, 2, 1)]
```

Bài 73:

Yêu cầu:

Viết chương trình để giải 1 câu đố cổ của Trung Quốc: Một trang trại thỏ và gà có 35 đầu, 94 chân, hỏi số thỏ và gà là bao nhiêu?

Gợi ý:

- Sử dụng vòng lặp for để lặp qua tất cả các giả thuyết có thể.

Code mẫu:

```
def giai(dau,chan):
    klg='Không có dáp án phù hợp!'
    for i in range(dau+1):
        j=dau-i
        if 2*i+4*j==chan:
            return i,j
    return klg,klg
# Code by TH&THCSTTQuan Chu
dau=35
chan=94
dap_an=giai(dau,chan)
print (dap_an)
```

Kết quả:

```
(23, 12)
```

Bài 74:

Yêu cầu:

Viết chương trình Python tính tổng tất cả số nguyên dương và âm trong một chuỗi được cung cấp.

Chuỗi gốc: `-100#^sdfkj8902w3ir021@swf-20`

Tính tổng tất cả số nguyên âm và dương nằm trong chuỗi sau:

- Giá trị dương: 9046
- Giá trị âm: -120

Cách 1:

Code Python:

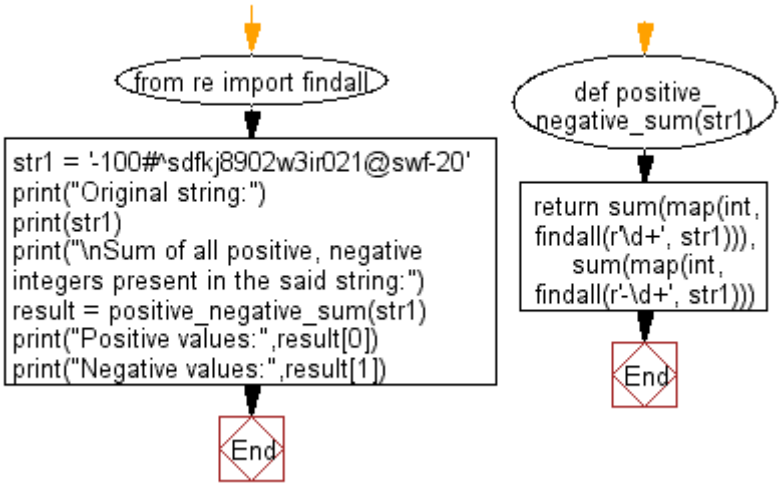
```
from re import findall
def positive_negative_sum(str1):
    return sum(map(int, findall(r'\d+', str1))), sum(map(int, findall(r'-\d+', str1)))
str1 = '-100#^sdfkj8902w3ir021@swf-20'
print("Original string:")
print(str1)
print("\nSum of all positive, negative integers present in the said string:")
result = positive_negative_sum(str1)
print("Positive values:",result[0])
print("Negative values:",result[1])
```

Kết quả mẫu:

```
Original string:
-100#^sdfkj8902w3ir021@swf-20

Sum of all positive, negative integers present in the said string:
Positive values: 9046
Negative values: -120
```

Sơ đồ



Cách 2:

Code Python:

```
import re
def positive_negative_sum(str1):
    positive_values = sum(int(n) for n in re.findall('\d+', str1))
    negative_values = sum(int(n) for n in re.findall('-\d+', str1))
    return positive_values, negative_values

str1 = '-100#^sdfkj8902w3ir021@swf-20'
print("Original string:")
```

```
print(str1)
print("\nSum of all positive, negative integers present in the said string:")
result = positive_negative_sum(str1)
print("Positive values:",result[0])
print("Negative values:",result[1])
```

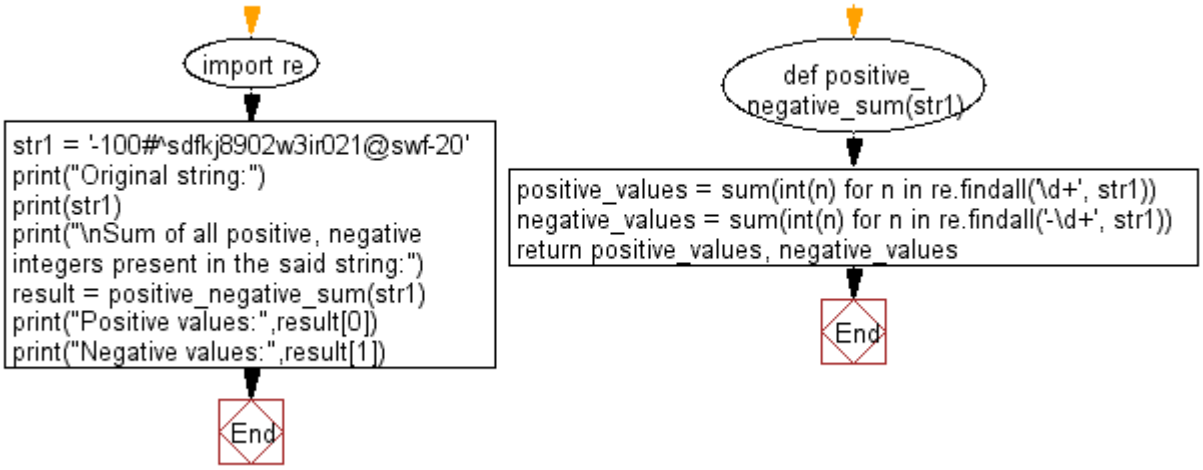
Kết quả mẫu:

```
Original string:

-100#^sdfkj8902w3ir021@swf-20

Sum of all positive, negative integers present in the said string:
Positive values: 9046
Negative values: -120
```

Sơ đồ:



The end.