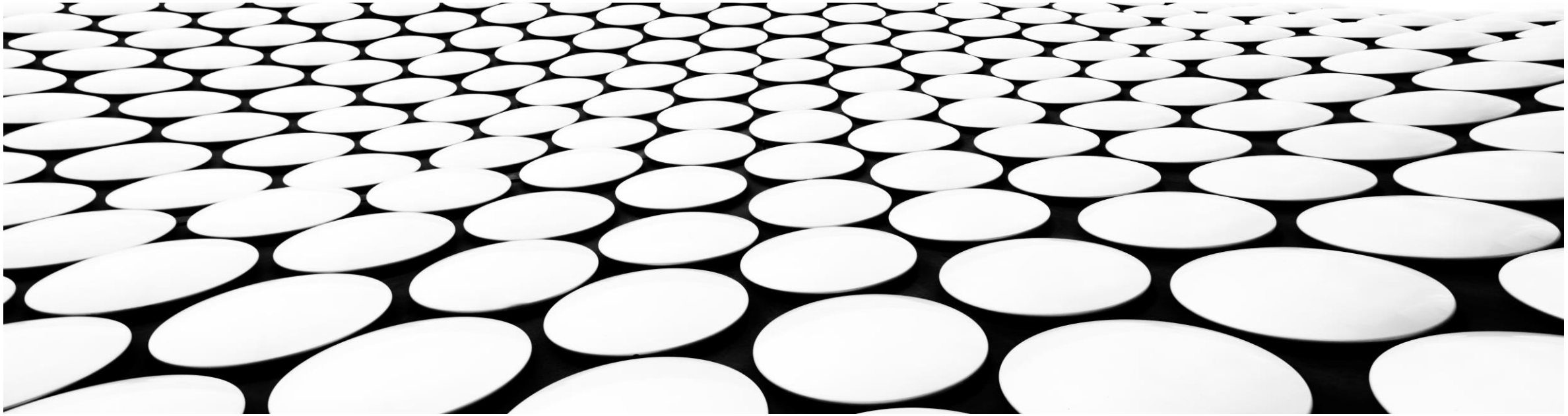
KĨ THUẬT LẬP TRÌNH PYTHON

NGUYỄN MẠNH HÙNG



CHƯƠNG 2.1: CÁC CÂU LỆNH ĐIỀU KHIỂN

1. BIỂU THỨC BOOLEAN VÀ TOÁN TỬ LOGIC

- **Biểu thức boolean**: là biểu thức mang giá trị **True** hoặc **False**.
- **Ví dụ**: (x==y) là một biểu thức boolean, == là toán tử so sánh và trả về True nếu hai giá trị bằng nhau và trả về False nếu ngược lại

```
print(5==5,5==6)
```

```
True False
```

Toán tử so sánh	Giải thích
x==y	x bằng y
x!=y	x không bằng y
x>y	x lớn hơn y
x<y	x nhỏ hơn y
x>=y	x lớn hơn hoặc bằng y
x<=y	x nhỏ hơn hoặc bằng y
x **is** y	x bằng y
x **is not** y	x không bằng y

- Có 3 toán tử logic: **and**, **or**, **not**. Xét 2 biểu thức boolean A và B, khi kết hợp với 3 toán tử logic, ta thu được các biểu thức boolean có giá trị như trong bảng dưới đây:

A	B	A **and** B	A **or** B	**not** A
True	True	True	True	False
True	False	False	True	False
False	True	False	True	True
False	False	False	False	True

- **Ví dụ**:

```
x=15
a = (x>0 and x<10)        # x lớn hơn 0, và x nhỏ hơn 10
b = (x%2==0 or x%3==0)    # x chia hết cho 2, hoặc x chia hết cho 3
c = (not x%4==3)          # không phải x chia 4 dư 3
print(a,b,c)            # in giá trị của biểu thức a, b, c
```

```
False True False
```

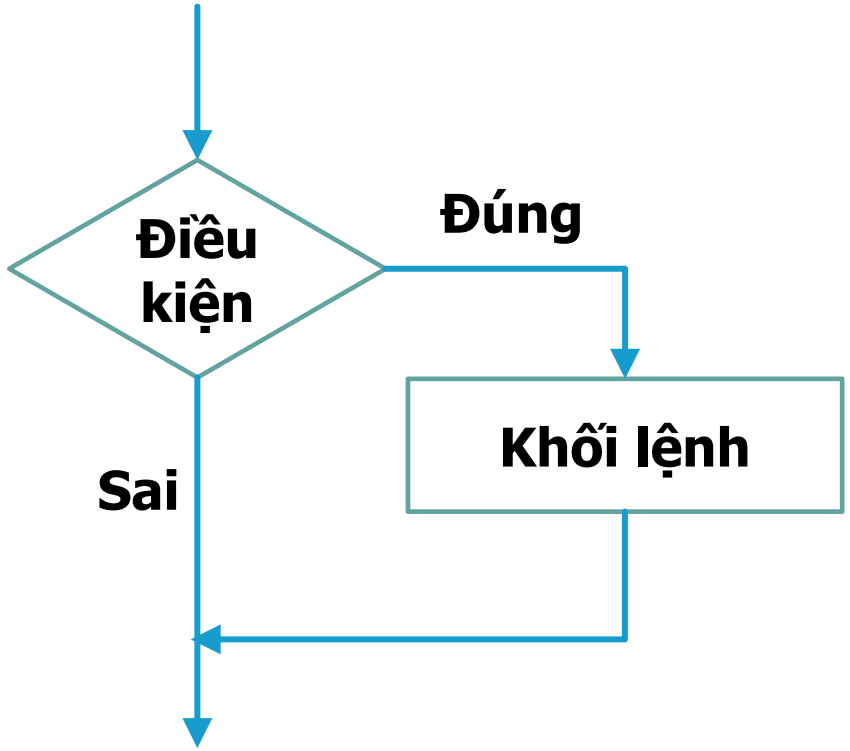
2. CÂU LỆNH IF

- **Cú pháp**:

```
if   <Điều kiện> :
     <Khối lệnh>
```

- **Giải thích**:
  - <Điều kiện> là một biểu thức boolean, theo sau bởi dấu hai chấm (:), các dòng bên dưới câu lệnh **if** là các lệnh muốn thực hiện.
  - Nếu <Điều kiện> là đúng thì <Khối lệnh> được thực thi, ngược lại thì bỏ qua.

Sơ đồ luồng điều khiển



<u>Ví dụ 1</u>: Viết một chương trình nhập vào 3 số thực x, y, z. Chương trình sẽ đưa ra kết luận x, y, z có phải là độ dài 3 cạnh của một tam giác hay không.

```
x=float(input("Nhập vào số thực x: "))
y=float(input("Nhập vào số thực y: "))
z=float(input("Nhập vào số thực z: "))
boolean = (x>0 and y>0 and z>0) and (x+y>z and x+z>y and y+z>x)
if boolean:
    print("x,y,z là độ dài 3 cạnh của một tam giác.")
if not boolean:
    print("x,y,z không phải là độ dài 3 cạnh của một tam giác.")
```

```
Nhập vào số thực x: -2
Nhập vào số thực y: 4
Nhập vào số thực z: 3
x,y,z không phải là độ dài 3 cạnh của một tam giác.
```

<u>Ví dụ 2</u>: Viết một chương trình nhập vào tên của 3 người. Chương trình sẽ in ra các tên này theo thứ tự bảng chữ cái.

Chẳng hạn,

John

Sebastian

Legatiuk

Sắp thứ tự: John Legatiuk Sebastian

```
a=str(input("Tên người thứ nhất: "))
b=str(input("Tên người thứ hai: "))
c=str(input("Tên người thứ ba: "))
if a<=b<=c:
    print("Tên được sắp thứ tự: ", a, b, c)
if a<=c<=b:
    print("Tên được sắp thứ tự: ", a, c, b)
if b<=a<=c:
    print("Tên được sắp thứ tự: ", b, a, c)
if b<=c<=a:
    print("Tên được sắp thứ tự: ", b, c, a)
if c<=a<=b:
    print("Tên được sắp thứ tự: ", c, a, b)
if c<=b<=a:
    print("Tên được sắp thứ tự: ", c, b, a)
```

```
Tên người thứ nhất: Yến
Tên người thứ hai: Vũ
Tên người thứ ba: Nam
Tên được sắp thứ tự:  Nam Vũ Yến
```

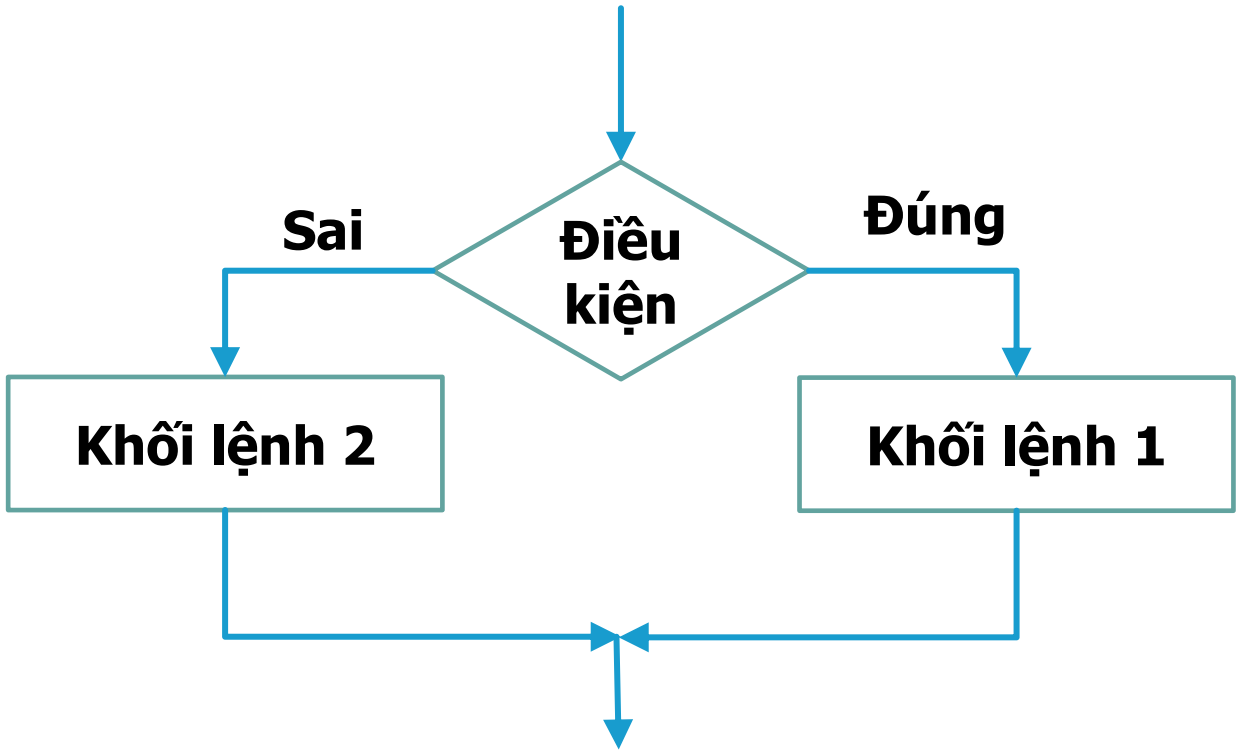
3. CÂU LỆNH IF ... ELSE

- **Cú pháp**:

```
if    <Điều kiện>:
      <Khối lệnh 1>
else :
      <Khối lệnh 2>
```

- **Giải thích**: Trong câu lệnh **if** ... **else**, có hai lựa chọn thực hiện khối lệnh và <điều kiện> sẽ xác định khối lệnh nào được thực thi.

Sơ đồ luồng điều khiển



<u>Ví dụ 3</u>: Viết một chương trình tính tiền lương tuần cho nhân viên. Chương trình sẽ nhập vào họ tên của nhân viên, số giờ làm của nhân viên, và tiền lương mỗi giờ. Theo quy định, nếu nhân viên làm nhiều hơn 40 giờ/tuần thì số giờ vượt mức sẽ được tính lương theo giờ gấp 1.5 lần. Chương trình tính và thông báo tiền lương được thanh toán cho nhân viên đó.

Chẳng hạn,

John

Số giờ làm: 45

Lương giờ: 9.5

Tiền lương: 451.25

```
st=str(input("Nhập họ tên của nhân viên: "))
x=float(input("Nhập số giờ làm việc: "))
y=float(input("Nhập lương cơ bản theo giờ: "))
if x<=40:
    print("Tiền lương phải trả cho nhân viên",st,"là: ",x*y)
else:
    print("Tiền lương phải trả cho nhân viên",st,"là: ",40*y+(x-40)*1.5*y)
```

```
Nhập họ tên của nhân viên: Nguyễn Mạnh Thắng
Nhập số giờ làm việc: 45
Nhập lương cơ bản theo giờ: 9.5
Tiền lương phải trả cho nhân viên Nguyễn Mạnh Thắng là:  451.25
```

4. CÂU LỆNH IF … ELIF … ELSE

- **Cú pháp**: **if** <Điều kiện 1>**:**

  <Khối lệnh 1>

  elif <Điều kiện 2>**:**

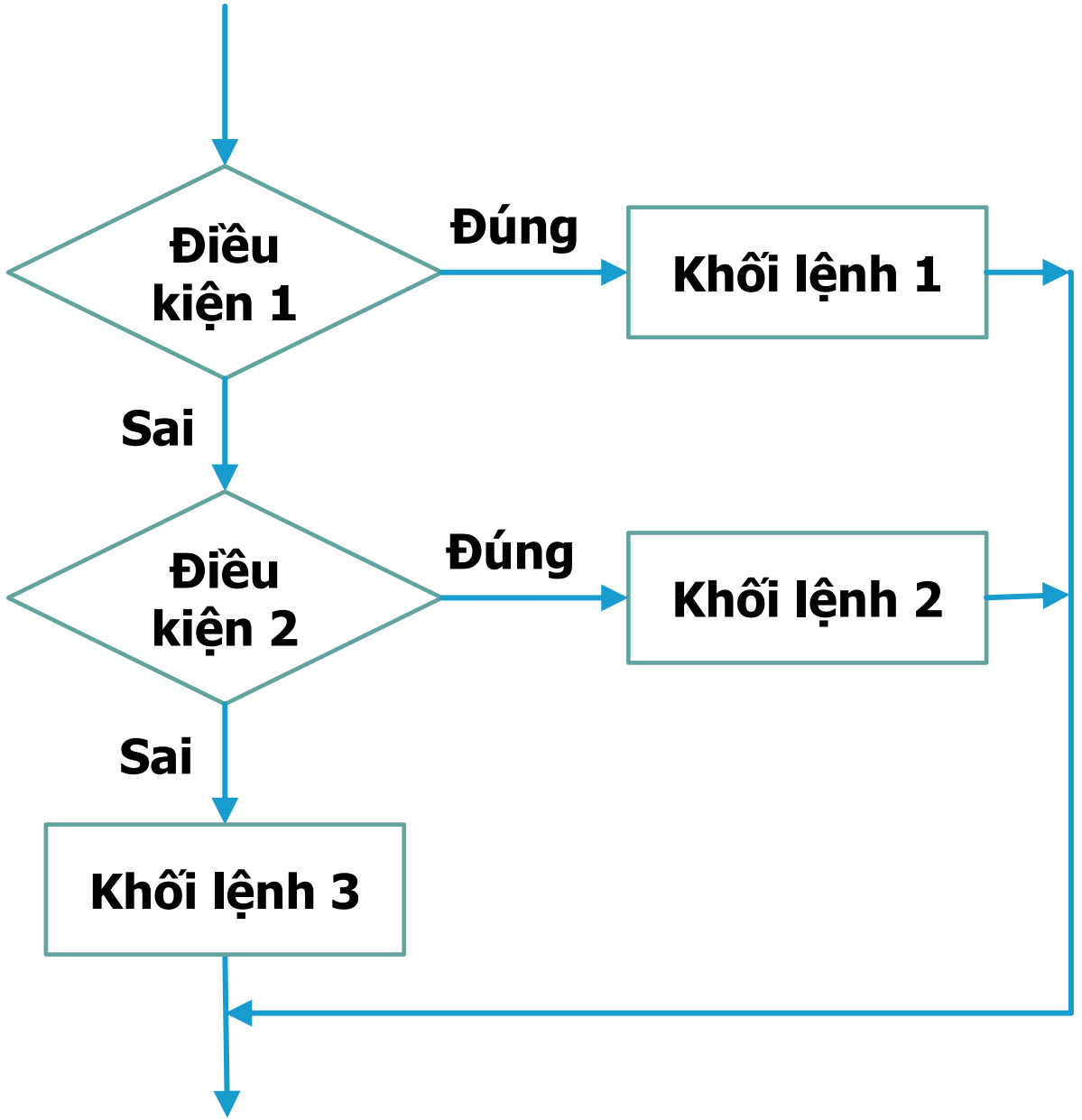
  <Khối lệnh 2>

  else :

  <Khối lệnh 3>

- **Giải thích**: câu lệnh **if ... elif ... else** cho phép xử lí những tình huống có nhiều hơn 2 lựa chọn thực hiện. Khi <Điều kiện 1> sai thì <Điều kiện 2> sẽ được kiểm tra để xác định khối lệnh thực hiện.

Sơ đồ luồng điều khiển



Ví dụ 4: Viết một chương trình nhập vào một số thực. Chương trình sẽ kiểm tra và đưa ra thông báo số đó là số dương, hoặc số âm, hoặc bằng 0.

```
x=float(input("Nhập vào một số thực: "))
if x>0:
    print("x là số dương")
elif x<0:
    print("x là số âm")
else:
    print("x bằng 0")
```

```
Nhập vào một số thực: -2.3
x là số âm
```

Ví dụ 5: Viết một chương trình nhập vào các hệ số của phương trình bậc hai. Chương trình sẽ in ra nghiệm của phương trình bậc hai hoặc thông báo phương trình vô nghiệm.

```
from math import *
print("Nhập các hệ số của phương trình bậc hai:")
a=float(input("a= "))
b=float(input("b= "))
c=float(input("c= "))
delta=b*b-4*a*c
if a==0:
    print("Không phải phương trình bậc hai!")
elif delta<0:
    print("Phương trình vô nghiệm!")
elif delta==0:
    print("Phương trình có nghiệm kép: x=",-b/(2*a))
else:
    print("Phương trình có hai nghiệm phân biệt: x1=",(-b+sqrt(delta))/(2*a),", x2=",(-b-sqrt(delta))/(2*a))
```

```
Nhập các hệ số của phương trình bậc hai:
a= -3
b= 4
c= 1
Phương trình có hai nghiệm phân biệt: x1= -0.21525043702153024 , x2= 1.5485837703548635
```

5. CÂU LỆNH TRY … EXCEPT

- **Cú pháp**:

```
try :
    <Khối lệnh 1>
except :
    <Khối lệnh 2>
```

- **Giải thích**: câu lệnh **try ... except** là cấu trúc thực thi có điều kiện cho phép bắt lỗi không mong muốn. Nó dựa trên ý tưởng là ta biết một số câu lệnh có thể có vấn đề và ta muốn thực thi một số lệnh nào đó khi có lỗi xảy ra.

Ví dụ 6: Viết một chương trình nhập vào một số thực là nhiệt độ Fahrenheit. Chương trình sẽ đổi sang nhiệt độ Celsius và in ra kết quả.

```
fahr=float(input("Nhập nhiệt độ Fahrenheit: "))
cel=(fahr-32.0)*5.0/9.0
print("Nhiệt độ Celsius tương ứng là", cel)
```

```
Nhập nhiệt độ Fahrenheit: không biết

---------------------------------------------------------------
ValueError                Traceback (most recent call last)
<ipython-input-11-dd2792e74571> in <module>
----> 1 fahr=float(input("Nhập nhiệt độ Fahrenheit: "))
      2 cel=(fahr-32.0)*5.0/9.0
      3 print("Nhiệt độ Celsius tương ứng là", cel)

ValueError: could not convert string to float: 'không biết'
```

```python
try:
    fahr=float(input("Nhập nhiệt độ Fahrenheit: "))
    cel=(fahr-32.0)*5.0/9.0
    print("Nhiệt độ Celsius tương ứng là", cel)
except:
    print("Có lỗi nhập dữ liệu. Xin hãy nhập một số.")
```

```
Nhập nhiệt độ Fahrenheit: không biết
Có lỗi nhập dữ liệu. Xin hãy nhập một số.
```

Thực hành:

- **Bài tập C2-1**: Viết chương trình nhập vào họ tên một sinh viên và điểm trung bình học tập nằm trong khoảng từ 0 đến 10. Nếu điểm nhập vào nằm ngoài khoảng trên thì chương trình đưa ra thông báo lỗi. Nếu điểm nhập vào nằm trong khoảng trên thì chương trình đưa ra thông báo điểm chữ theo bảng dưới đây:

Điểm số	<4	>=4	>=5.5	>=7	>=8.5
Điểm chữ	F	D	C	B	A

- **Bài tập C2-2**: Viết chương trình nhập vào họ tên một người, chiều cao H (m), và cân nặng W (kg). Chương trình sẽ tính chỉ số BMI = W/H^2 (Body mass index) của người đó và đưa ra thông báo về tình trạng béo phì của người đó theo bảng dưới đây:

BMI	<20	>=20	>=25	>=30.0
Kết luận	Gầy	Bình thường	Thừa cân	Béo phì

- **Bài tập C2-3**: Hãy viết lại chương trình tính tiền lương trong Ví dụ 3, sử dụng cấu trúc **try ... except** để xử lý những lỗi nhập dữ liệu về số giờ làm việc, lương giờ không phải số, bằng cách thông báo lỗi và thoát khỏi chương trình. Hai lỗi nhập dữ liệu được mô tả dưới đây:

  Số giờ làm việc: 20

  *Lương giờ: **chín***

  Lỗi, bạn phải nhập dữ liệu số

  *Số giờ làm việc: **bốn mươi***

  Lương giờ: 9

  Lỗi, bạn phải nhập dữ liệu số

6. VÒNG LẶP FOR

- **Cú pháp**:

  for <Tên biến> **in** [Tập các giá trị] **:**

  <Khối lệnh>

- **Giải thích**: Biến sẽ nhận lần lượt các giá trị trong [Tập các giá trị], với mỗi lần nhận giá trị của biến, <Khối lệnh> sẽ được thực hiện. Như vậy, <Khối lệnh> sẽ được thực hiện lặp lại bằng số giá trị trong [Tập giá trị]. Vòng lặp **for** có số lần lặp xác định.

```python
friends=["Hạnh","Nam","Nguyên","Quyền"]
for x in friends:
    print("Welcome to the Python course, ",x)
print("Kết thúc!")
```

```
Welcome to the Python course,  Hạnh
Welcome to the Python course,  Nam
Welcome to the Python course,  Nguyên
Welcome to the Python course,  Quyền
Kết thúc!
```

- **range(**n**)** : tạo ra tập gồm các số 0, 1, ..., n-1
- **range(**a,b**)** : tạo ra tập gồm các số a, a+1, ..., b-1
- **range(**a,b,c**)** : tạo ra tập gồm các số a, a+c, a+2c, ... nhưng nhỏ hơn b.

```
n=int(input("Nhập vào một số tự nhiên: "))
sumx=0
for x in range(n):
    sumx+=x+1
print("Tổng của",n,"số tự nhiên đầu tiên là ",sumx)
```

```
Nhập vào một số tự nhiên: 4
Tổng của 4 số tự nhiên đầu tiên là  10
```

```python
n=int(input("Nhập vào một số tự nhiên: "))
prod_odd=1
for x in range(1,n,2):
    prod_odd*=x
print("Tích của các số lẻ không vượt quá",n,"là :",prod_odd)
```

```
Nhập vào một số tự nhiên: 6
Tích của các số lẻ không vượt quá 6 là : 15
```

Ví dụ 7: Viết một chương trình nhập vào một số thực, sử dụng vòng lặp **for** để tính căn bậc hai của số được nhập với 20 bước lặp, bằng cách sử dụng phương pháp Newton. Chương trình sẽ đưa ra thông báo lỗi nếu số nhập vào là số âm.

Gợi ý:

Phương pháp Newton tính xấp xỉ căn bậc 2 như sau:

$$y = \sqrt{x} \Longrightarrow f(y) = y^2 - x = 0$$

$$\Longrightarrow y_{n+1} = y_n - \frac{f(y_n)}{f'(y_n)} = y_n - \frac{y_n^2 - x}{2y_n}$$

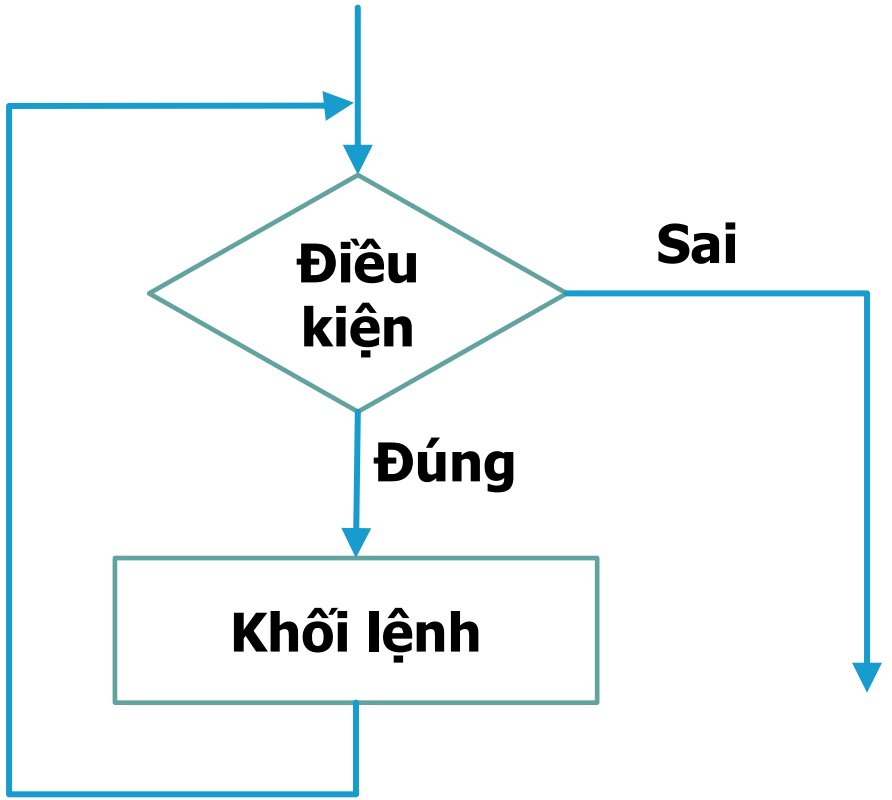
$$\Longrightarrow \boxed{y_{n+1} = \frac{1}{2}\left(y_n + \frac{x}{y_n}\right)}$$

```
x=float(input("Nhập vào một số thực: "))
if x<0:
    print("Không tồn tại căn bậc hai của",x)
elif x==0:
    print("Căn bậc hai của 0 là 0")
else:
    y=x/2
    for i in range(10):
        y=1/2*(y+x/y)
    print("Căn bậc hai của %f là %.15f" %(x,y))
```

```
Nhập vào một số thực: 23
Căn bậc hai của 23.000000 là 4.795831523312719
```

7. VÒNG LẶP WHILE

Sơ đồ luồng điều khiển



- **Cú pháp**:

  while <Điều kiện> **:**

  <Khối lệnh>

- **Giải thích**: Đầu tiên luồng điều khiển sẽ kiểm tra <Điều kiện>. Nếu <Điều kiện> đúng, thì thực thi <Khối lệnh>, sau đó quay lại kiểm tra <Điều kiện>. Nếu <Điều kiện> sai, thoát khỏi vòng lặp while và thực thi câu lệnh tiếp theo. Vòng lặp **while** có thể có số lần lặp **không xác định**.

<u>Ví dụ 8</u>: Viết một chương trình nhập vào một số thực, sử dụng vòng lặp **while** để tính căn bậc hai của số được nhập, bằng phương pháp Newton. Vòng lặp dừng khi

$$|y_{n+1} - y_n| < 1e - 5$$

```python
x=float(input("Nhập vào một số thực: "))
if x<0:
    print("Không tồn tại căn bậc hai của",x)
elif x==0:
    print("Căn bậc hai của 0 là 0")
else:
    boolean=True
    y1=x/2
    n=1
    while boolean:
        y2=1/2*(y1+x/y1)
        boolean=(abs(y2-y1)>1e-5)
        y1=y2
        n+=1
    print("Căn bậc hai của %f là %.15f" %(x,y1))
    print("Số lần lặp:",n)
```

```
Nhập vào một số thực: 23
Căn bậc hai của 23.000000 là 4.795831523317069
Số lần lặp: 6
```

8. VÒNG LẶP VÔ HẠN VÀ CÂU LỆNH BREAK

- Đôi khi ta không biết khi nào thì phải kết thúc vòng lặp cho đến khi thực thi được một phần chương trình. Trong trường hợp đó, ta có thể viết một vòng lặp vô hạn và sử dụng câu lệnh **break** để thoát ra khỏi vòng lặp.

- Bằng cách dùng vòng lặp **while** như trong chương trình dưới đây, ta có thể kiểm tra điều kiện dừng tại bất kì đâu trong vòng lặp.

Rõ ràng, vòng lặp là vô hạn vì biểu thức logic trong câu lệnh **while** luôn có giá trị **True**.

```
while True:
    line=input("Kết thúc vòng lặp? (c/k)")
    if line=="c":
        break
    print(line)
print("Kết thúc!")
```

```
Kết thúc vòng lặp? (c/k)k
k
Kết thúc vòng lặp? (c/k)a
a
Kết thúc vòng lặp? (c/k)c
Kết thúc!
```

9. KẾT THÚC LẶP VỚI CÂU LỆNH CONTINUE

- Đôi khi ta đang ở trong một vòng lặp và muốn kết thúc lần lặp hiện tại, đồng thời nhảy sang lần lặp tiếp theo. Trong trường hợp đó, ta có thể sử dụng câu lệnh **continue** mà không cần thực thi toàn bộ câu lệnh của lần lặp hiện tại.

```
while True:
    line=input("Kết thúc vòng lặp? (c/k) ")
    if line[0]=='#':
        continue
    if line=="c":
        break
    print(line)
print("Kết thúc!")
```

```
Kết thúc vòng lặp? (c/k) không
không
Kết thúc vòng lặp? (c/k) #alkdhflahs
Kết thúc vòng lặp? (c/k) c
Kết thúc!
```

10. CÁC VÒNG LẶP LỒNG NHAU

- Khối lệnh trong vòng lặp có thể chứa một vòng lặp khác. Khi đó, ta có các vòng lặp lồng nhau.

- Kiểu của các vòng lặp có thể khác nhau.

```
message=input("Nhập vào một thông điệp (gõ Enter để thoát) ")
while message!="":
    n=int(input("Bạn muốn lặp lại thông điệp bao nhiêu lần? "))
    for i in range(n):
        print(message)
    message=input("Nhập vào một thông điệp (gõ Enter để thoát) ")
print("Kết thúc!")
```

```
Nhập vào một thông điệp (gõ Enter để thoát) Cố lên Việt Nam!
Bạn muốn lặp lại thông điệp bao nhiêu lần? 5
Cố lên Việt Nam!
Cố lên Việt Nam!
Cố lên Việt Nam!
Cố lên Việt Nam!
Cố lên Việt Nam!
Nhập vào một thông điệp (gõ Enter để thoát)
Kết thúc!
```

Thực hành:

- **Bài tập C2-4**: Hãy viết một chương trình nhập vào liên tục các số, quá trình dừng khi nhập vào chuỗi “Hoàn thành”. Sau đó, chương trình đếm số các số được nhập vào, tính tổng và trung bình cộng của các số đó. Nếu người dùng nhập vào một chuỗi nào đó không phải số, chương trình sẽ bắt lỗi, sử dụng **try** ... **except**, và in ra thông báo lỗi, và tiếp tục nhập vào số tiếp theo.

Minh họa: *Nhập vào một số: 4*

Nhập vào một số : 5

*Nhập vào một số : **dữ liệu khác***

Không phù hợp

Nhập vào một số : 7

*Nhập vào một số : **Hoàn thành***

16 3 5.333333333333333

- **Bài tập C2-5**: Hãy viết một chương trình nhập vào một dãy số như trong **Bài tập 4**. Chương trình sẽ in ra giá trị lớn nhất và giá trị nhỏ nhất của dãy số đó.

Minh họa: *Nhập vào một số: 4*

Nhập vào một số : 5

*Nhập vào một số : **dữ liệu khác***

Không phù hợp

Nhập vào một số : 7

*Nhập vào một số : **Hoàn thành***

Giá trị lớn nhất : 7

Giá trị nhỏ nhất : 4

- **Bài tập C2-6**: Hãy viết một chương trình nhập vào một số tự nhiên $n < 1e + 1000$. Chương trình sẽ tính tổng các chữ số và tìm chữ số lớn nhất, nhỏ nhất của n.

Minh họa: *Nhập vào một số : 24157*

Tổng các chữ số : 2+4+1+5+7 = 19

Chữ số lớn nhất : 7

Chữ số nhỏ nhất : 1

- **Bài tập C2-7**: Giá trị của π (số Pi) có thể được xấp xỉ bởi chuỗi vô hạn sau đây:

$$\pi \approx 3 + \frac{4}{2 \times 3 \times 4} - \frac{4}{4 \times 5 \times 6} + \frac{4}{6 \times 7 \times 8} - \frac{4}{8 \times 9 \times 10} + \frac{4}{10 \times 11 \times 12} - \cdots$$

Hãy viết một chương trình hiển thị 15 giá trị xấp xỉ cho π. Giá trị xấp xỉ đầu tiên chỉ dùng số hạng đầu tiên trong chuỗi vô hạn. Giá trị xấp xỉ tiếp theo bằng giá trị trước cộng thêm một số hạng nữa trong khai triển chuỗi.

Minh họa: $\pi_1 \approx 3 + \frac{4}{2 \times 3 \times 4}$; $\pi_2 \approx 3 + \frac{4}{2 \times 3 \times 4} - \frac{4}{4 \times 5 \times 6}$; …