Phép toán và các cấu trúc điều khiển, vòng lặp

Arithmetic operators

Phép toán	Ý nghĩa	Ví dụ
+	Add two operands or unary plus	x + y+ 2
-	Subtract right operand from the left or unary minus	x - y- 2
*	Multiply two operands	x * y
/	Divide left operand by the right one (always results into float)	x / y
%	Modulus - remainder of the division of left operand by the right	x % y (remainder of x/y)
//	Floor division - division that results into whole number adjusted to the left in the number line	x // y
**	Exponent - left operand raised to the power of right	x**y (x to the power y)

Arithmetic operators

```
x = 15
y = 4

# Output: x + y = 19
print('x + y =',x+y)

# Output: x - y = 11
print('x - y =',x-y)

# Output: x * y = 60
print('x * y =',x*y)

# Output: x / y = 3.75
print('x / y =',x/y)

# Output: x // y = 3
print('x // y =',x//y)

# Output: x ** y = 50625
print('x ** y =',x**y)
```

Comparison operators

Phép toán	Ý nghĩa	Ví dụ
>	Greater than - True if left operand is greater than the right	x > y
<	Less than - True if left operand is less than the right	x < y
==	Equal to - True if both operands are equal	x == y
!=	Not equal to - True if operands are not equal	x != y
>=	Greater than or equal to - True if left operand is greater than or equal to the right	x >= y
<=	Less than or equal to - True if left operand is less than or equal to the right	x <= y

Comparison operators

```
x = 10
y = 12

# Output: x > y is False
print('x > y is',x>y)

# Output: x < y is True
print('x < y is',x<y)

# Output: x == y is False
print('x == y is',x==y)

# Output: x != y is True
print('x != y is',x!=y)

# Output: x >= y is False
print('x >= y is',x>=y)

# Output: x <= y is True
print('x <= y is',x<=y)
```

Logical operators

Phép toán	Ý nghĩa	Ví dụ
and	True if both the operands are true	x and y
or	True if either of the operands is true	x or y
not	True if operand is false (complements the operand)	not x

```
x = True
y = False

print('x and y is',x and y)

print('x or y is',x or y)

print('not x is',not x)
```

Bitwise operators

Phép toán	Ý nghĩa	Ví dụ
&	Bitwise AND	x & y = 0 (0000 0000)
|	Bitwise OR	x | y = 14 (0000 1110)
~	Bitwise NOT	~x = -11 (1111 0101)
^	Bitwise XOR	x ^ y = 14 (0000 1110)
>>	Bitwise right shift	x >> 2 = 2 (0000 0010)
<<	Bitwise left shift	x << 2 = 40 (0010 1000)

Assignment operators

Phép toán	Ý nghĩa	Ví dụ
=	x = 5	x = 5
+=	x += 5	x = x + 5
-=	x -= 5	x = x - 5
*=	x *= 5	x = x * 5
/=	x /= 5	x = x / 5
%=	x %= 5	x = x % 5
//=	x //= 5	x = x // 5
**=	x **= 5	x = x ** 5
&=	x &= 5	x = x & 5
|=	x |= 5	x = x | 5
^=	x ^= 5	x = x ^ 5
>>=	x >>= 5	x = x >> 5
<<=	x <<= 5	x = x << 5

Identity operators

Operator	Meaning	Example
is	True if the operands are identical (refer to the same object)	x is True
is not	True if the operands are not identical (do not refer to the same object)	x is not True

- Kiểm tra nếu hai giá trị hoặc biến có cùng địa chỉ trong bộ nhớ hay không
- Hai biến cùng giá trị có thể khác địa chỉ (xem lại biến mutable)

```
x1 = 5
y1 = 5
x2 = 'Hello'
y2 = 'Hello'
x3 = [1,2,3]
y3 = [1,2,3]
# Output: False
print(x1 is not y1)
# Output: True
print(x2 is y2)
# Output: False
print(x3 is y3)
```

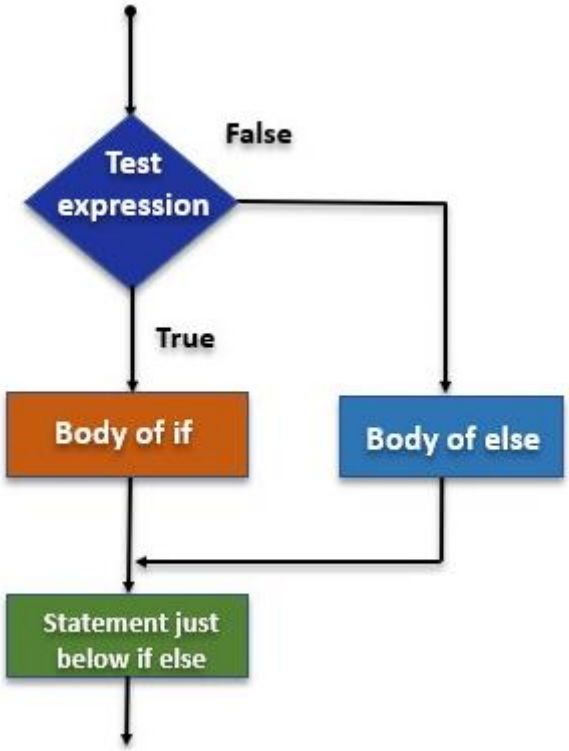
Membership operators

Kiểm tra giá trị hoặc biến có trong (string, list, tuple, set and dictionary)

Operator	Meaning	Example
in	True if value/variable is found in the sequence	5 in x
not in	True if value/variable is not found in the sequence	5 not in x

```
x = 'Hello world'
y = {1:'a',2:'b'}

# Output: True
print('H' in x)
# Output: True
print('hello' not in x)
# Output: True
print(1 in y)
# Output: False
print('a' in y)
```

Cấu trúc điều khiển if



lệnh if

```
if  <condition1>:
    <body1>
```

```
if  <condition1>:
    <body1>
else:
    <bodyelse>
```

```
if  <condition1>:
    <body1>
elif <condition2>:
    <body2>
...
elif <conditionn>:
    <bodyn>
...
else:
    <bodyelse>
```

```python
def get_week_day(argument):
    if(argument == 0):
        day="Sunday"
    elif(argument == 1):
        day="Monday"
    elif(argument == 2):
        day="Tuesday"
    elif(argument == 3):
        day="Wednesday"
    elif(argument == 4):
        day="Thursday"
    elif(argument == 5):
        day="Friday"
    elif(argument == 6):
        day="Saturday"
    else:
        day="Invalid day"
    return day
# Driver program
if __name__ == "__main__":
    print (get_week_day(6))
    print (get_week_day(8))
    print (get_week_day(0))
```

➔

Saturday
Invalid day
Sunday

Lập trình Switch statement với match

```
lang = input("What's the programming language you want to learn?
")
match lang:
    case "JavaScript":
        print("You can become a web developer.")
    case "Python":
        print("You can become a Data Scientist")
    case "PHP":
        print("You can become a backend developer")
        case "Solidity":
        print("You can become a Blockchain developer")
    case "Java":
        print("You can become a mobile app developer")
    case _:
        print("The language doesn't matter, what matters is
solving problems.")
```

Lập trình Switch statement với match

```
match variable_name:
    case ‘pattern1’: // statement1
    case ‘pattern2’: // statement2
    …
    case ‘pattern n’: // statement n
```

```
quit_flag = False
match quit_flag:
    case True:
        print("Quitting")
        exit()
    case False:
        print("System is on")
```

➔*System is on*

```
quit_flag = 4
 match quit_flag:
    case True:
        print("Quitting")
        exit()
    case False:
        print("System is on")
    case _:
        print("Boolean Value was not passed")
```

➔Boolean Value was not passed

Lập trình Switch statement với dictionary

```
switcher = {
    key_1: value_1/method_1(),
    key_2: value_2/method_2(),
    key_3: value_3/method_3(),
    ...
    key_n: value_n/method_n(),
    }
key = N
value = switcher.get(key, "default")
```

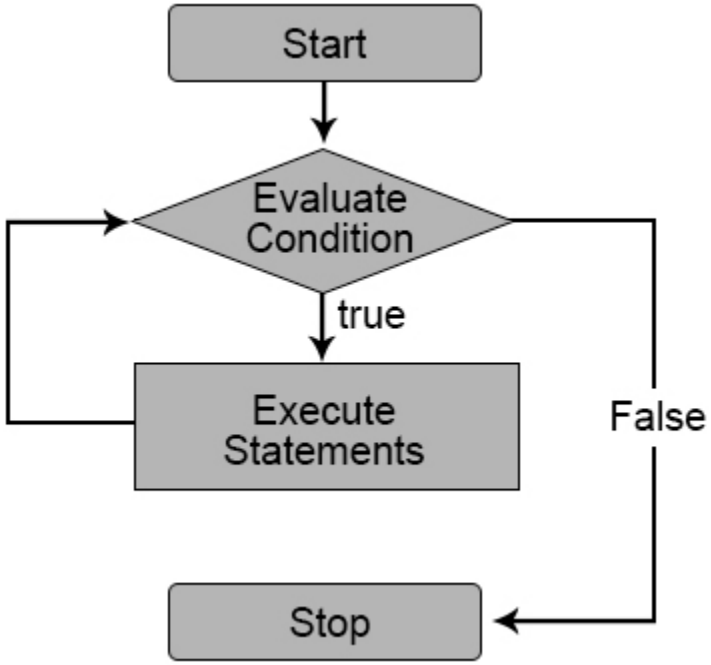
Lập trình Switch statement với dictionary

```
def get_week_day(argument):
    switcher = {
        0: "Sunday",
        1: "Monday",
        2: "Tuesday",
            3: "Wednesday",
            4: "Thursday",
            5: "Friday",
            6: "Saturday"
    }

    return switcher.get(argument, "Invalid day")
# Driver program
if __name__ == "__main__":

    print (get_week_day(6))
    print (get_week_day(8))
    print (get_week_day(0))
```

Saturday
➔ Invalid day
Sunday

Loop while

```
while expression:
    body of the loop
```

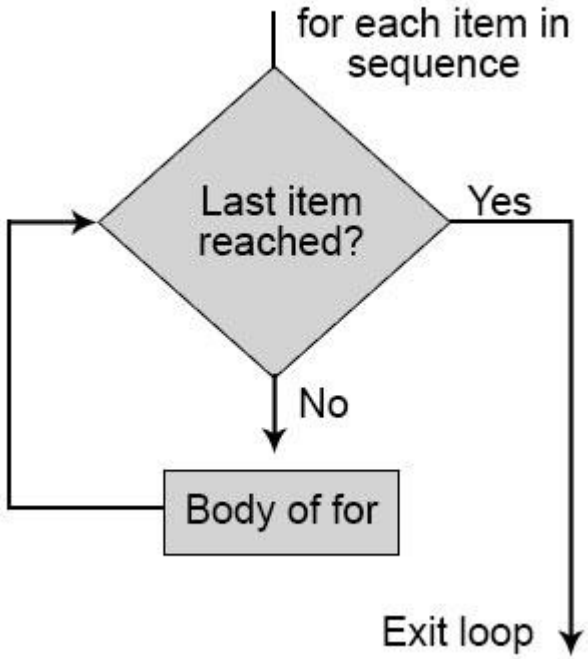


Loop while

```
n = 10
sum = 0
i = 1
while i <= n:
    sum = sum + i
    i = i+1
print("The sum is", sum)
```

➔The sum is 55

Loop for




```
for x in range(2, 6):
  print(x)
```

→
2
3
4
5

Lặp for

```
range(starting value, ending value, increment by)
```

```
for r in range(0,6,1):
    print(r)
```

➔
0
1
2
3
4
5

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

➔
5
4
3
2
1
0

Lặp for

```
greeting = "Hello Python"
# here we are taking for loop
for letter in greeting:
    print(letter)
```

```
greeting = "Hello Python"
# here we are taking for loop
for i in range(0, len(greeting)):
    print(greeting[i])
```

➔

H
e
l
l
o

P
y
t
h
o
n

Lặp for

```
color = ["green", "pink", "red"]
for c in color:
  print(c)
```

➔
green
pink
red

Lặp for

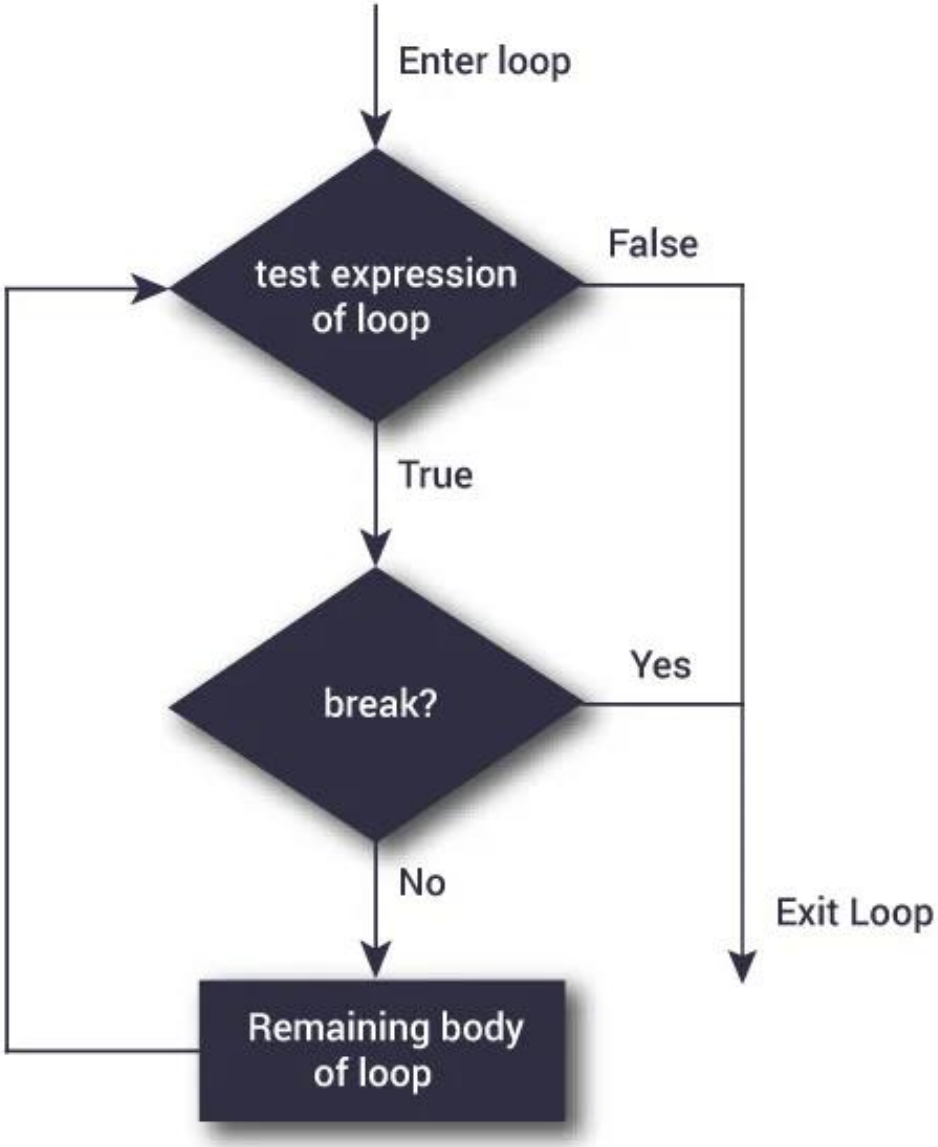
```
no = [1, 2]
color = ["red", "blue"]

for x in no:
  for y in color:
    print(x, y)
```

➔

1 red
1 blue
2 red
2 blue

break

- Sử dụng trong trường hợp cần thoát khỏi vòng lặp ngay lập tức khi thỏa điều kiện mong muốn nào đó và không cần phải lặp tiếp
- Ví dụ tìm thấy phần tử cần tìm hay tìm thấy lời giải bài toán
- Trong trường hợp có nhiều loop lồng nhau (nested loop), break chỉ thoát vòng lặp chứa nó mà thôi



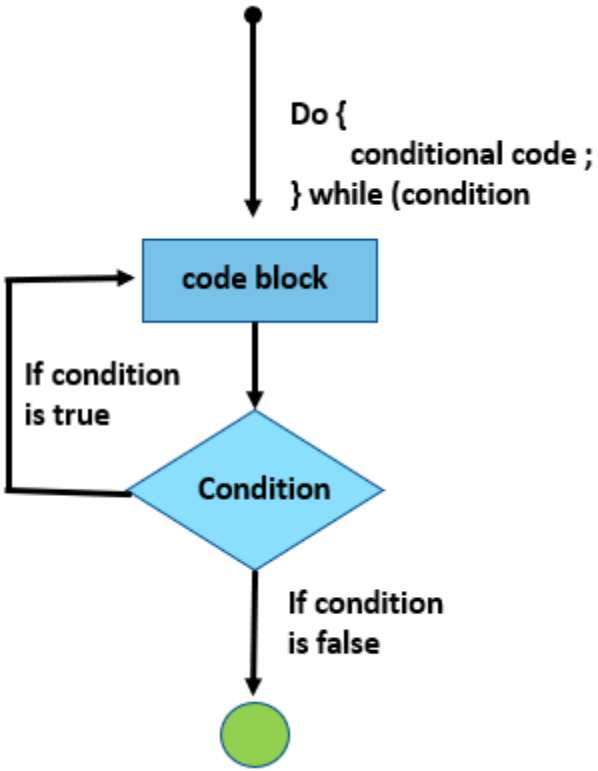
break

```
for var in sequence:
    # codes inside for loop
    if  condition:
        break
    # codes inside for loop

# codes outside for loop
```

```
while test expression:
    # codes inside while loop
    if  condition:
        break
    # codes inside while loop

# codes outside while loop
```

Lập trình do...while

```
while True:
    # statement (s)
    If not condition:
        break
```



break

```
""" from random import randint
day_so=[]
for i in range(0,10):
    so_nguyen=randint(0,1000)
    day_so.append(so_nguyen)
print(day_so) """
day_so=[394, 472, 815, 868, 95, 450, 15, 761, 12, 424]
for i in range(0,10):
    so_nguyen=day_so[i]
    if so_nguyen%3==0:
        print("So chia het cho 3 dau tien la:",so_nguyen,
      "vi tri", i)
        break
print ("i=", i)
```

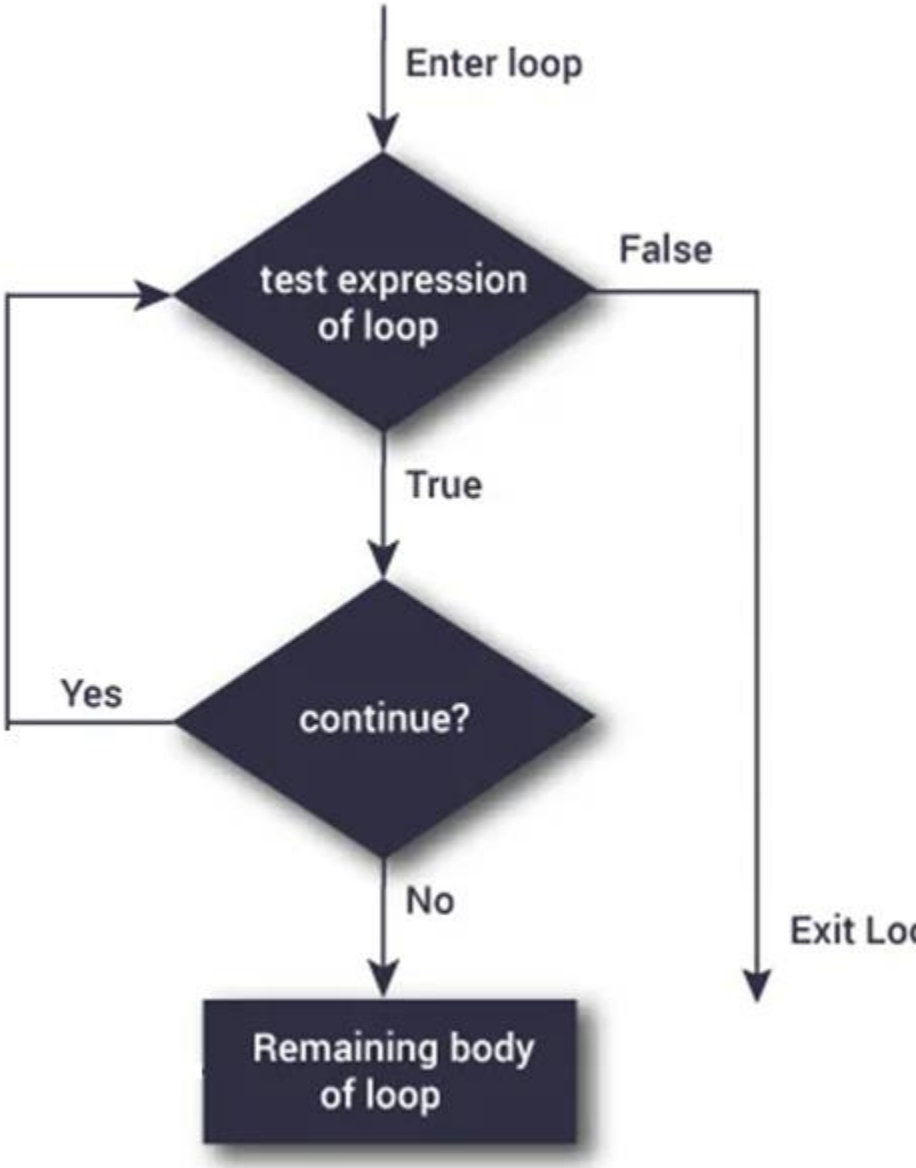
➔So chia het cho 3 dau tien la: 450 vi tri 5
i=5

continue

- Sử dụng trong trường hợp đi tiếp vòng lặp tiếp theo và bỏ qua tất cả các lệnh phía sau continue

- Ví dụ khi dữ liệu không đúng, hay không cần thiết phải tính toán cho lần lặp đó thì hãy lặp tiếp (continue)

- Trong trường hợp có nhiều loop lồng nhau, continue chỉ có tác dụng cho vòng lặp chứa nó mà thôi



continue

Kiểm tra điều kiện lặp trước

```
for var in sequence:
    # codes inside for loop
    if  condition:
        continue
    # codes inside for loop

# codes outside for loop
```

Kiểm tra điều kiện lặp trước

```
while test expression:
    # codes inside while loop
    if  condition:
        continue
    # codes  inside while loop

# codes outside while loop
```

continue

```
day_so = [394, 472, 815, 868, 95, 450, 15, 761, 12, 424]
for i in range(0,10):
    so_Nguyen = day_so[i]
    if so_Nguyen % 2 == 0:
        continue
    print("So le:",so_nguyen, "vi tri", i)
print ("i=", i)
```

➔

So le: 815 vi tri 2
So le: 95 vi tri 4
So le: 15 vi tri 6
So le: 761 vi tri 7
i=9

pass

- Giả sử chúng ta có một vòng lặp hoặc một hàm chưa được triển khai, nhưng chúng ta muốn triển khai nó trong tương lai

- Nhưng body của điều khiển, vòng lặp hay hàm không thể trống rỗng

➔ Câu lệnh pass để xây dựng một phần thân (body) mà không làm gì cả

pass

```
for i in range(0,10):
    pass

def function(args):
    pass

class Example:
    pass
```

Q & A

Thank you!