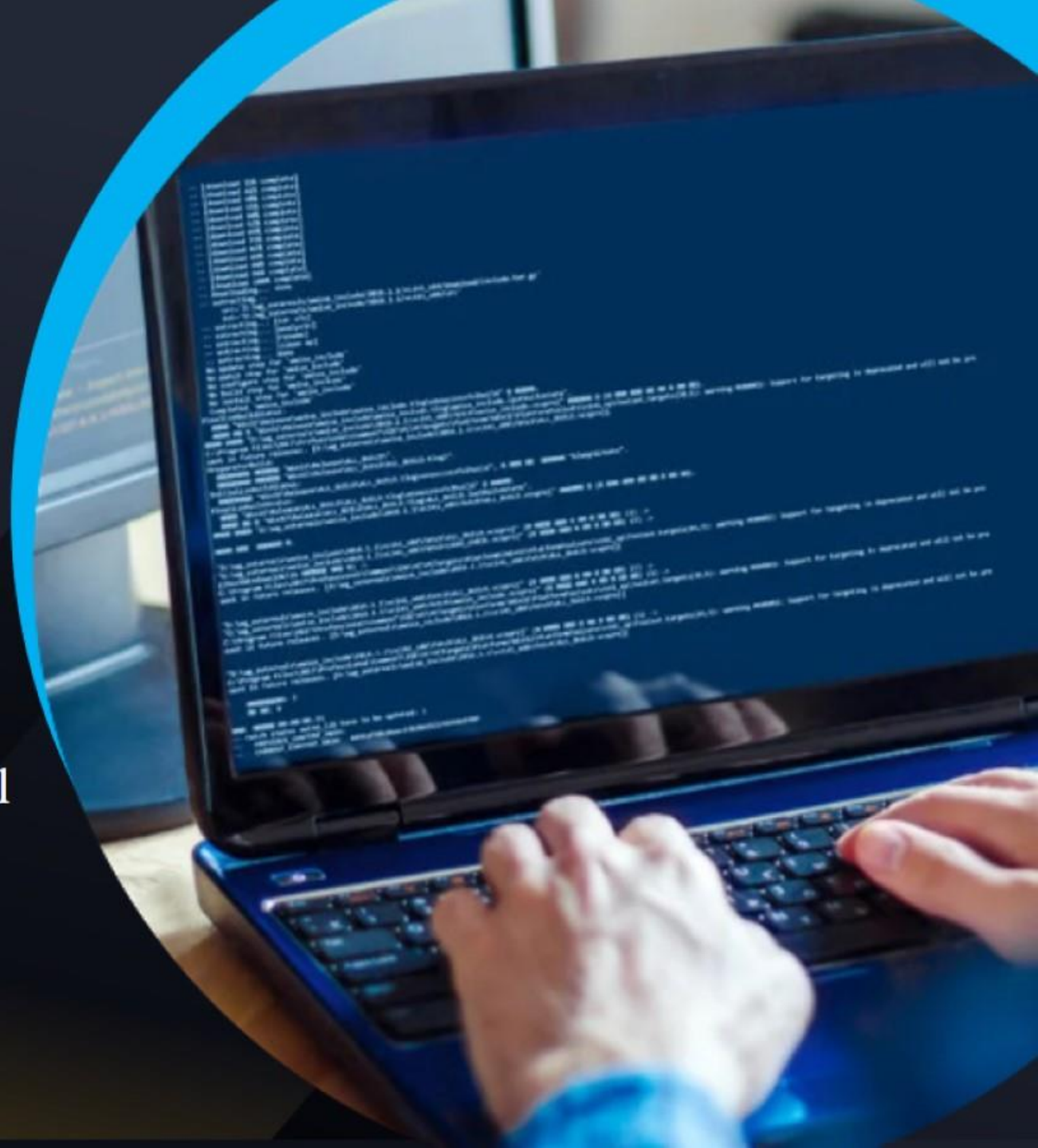


Programming Foundations in Python

Prepared by Dr. Hamada I. AbdulWakel

Lecture 3



Control and Iteration Statements

Lecture 3

Prepared by
Dr. Hamada I. AbdulWakel

www.midocean.edu.km

Lecture

03

RECAP: LECTURE 2

- The SDLC phases and models.
- The input and output operations.
- Data types in Python.

Lecture

03

LECTURE MAIN POINTS

- Controlling the program flow.
- The relational operators and comparison expressions.
- The selection statements.
- The iteration statements.

PROGRAM EXAMPLES

- Ex1: write a program that reads a three-digit number, separates this number into its individual digits and print them.
 - Program scenario:
Enter a number of three-digits: 234
Your number is 234 and its digits are:
First digit = 2
Second digit = 3
Third digit = 4

PROGRAM EXAMPLES

Cont'd

- Ex1: python code.

```
Number = int( input("Enter a number of three-digits: "))
D1 = Number // 100
New_number1 = Number % 100
D2 = New_number1 // 10
New_number2 = New_number1 % 10
print("Your number is", Number , " and its digits are: ")
print("First digit = ", D1)
print("Second digit = ", D2)
print("Third digit = ", New_number2)
```

PROGRAM EXAMPLES

Cont'd

- Ex2: write a program which inputs a temperature reading expressed in Fahrenheit and outputs its equivalent in Celsius, using the formula.

$$^{\circ}\text{C} = 5 / 9 * (^{\circ}\text{F} - 32)$$

- Program scenario:

Enter a temperature in Fahrenheit: 85

The equivalent Celsius temperature = 29.44

PROGRAM EXAMPLES

Cont'd

- Ex2: python code.

```
temp_fah = float( input("Enter a temperature in Fahrenheit: ") )
```

```
temp_cel = 5/9 * (temp_fah - 32)
```

```
print("The equivalent Celsius temperature = ", temp_cel)
```

BOOLEAN DATA TYPE AND COMPARISON OPERATORS

- Boolean data type consists of two values **TRUE** and **FALSE**.
- The **"is"** operator is used to test if two variables refer to the same value.
- The **"is not"** operator is used to compare if the objects are pointing to the same memory location or not.

COMPARISON OPERATOR	MEANING
==	Equals
!=	Not equals
<	Less than
>	Greater than
<=	Less than or equal
>=	Greater than or equal

BOOLEAN DATA TYPE AND COMPARISON OPERATORS

Cont'd

- Code examples:

```
print (15 > 3)
```

```
print(15 == 15)
```

```
x1 = 5
```

```
y1 = 5
```

```
print(x1 is not y1)
```

```
x2 = 'Hello'
```

```
y2 = 'Hello'
```

```
print(x2 is y2)
```

Lecture

03

COMPOUND BOOLEAN EXPRESSIONS

- The logical operators:
 - The **not** operator.
 - The **and** operator.
 - The **or** operator.

A	not A
True	False
False	True

A	B	A and B
True	True	True
True	False	False
False	True	False
False	False	False

A	B	A or B
True	True	True
True	False	True
False	True	True
False	False	False

COMPOUND BOOLEAN EXPRESSIONS

Cont'd

- Examples of compound Boolean expressions:

```
print( type(True) )
```

```
print( isinstance(True, int) )
```

```
print( True and True )
```

```
print( True and False )
```

```
print( 5 > 3 and 5 == 3 + 2 )
```

```
print( 5 < 3 and 5 == 3 and 5 != 3 )
```

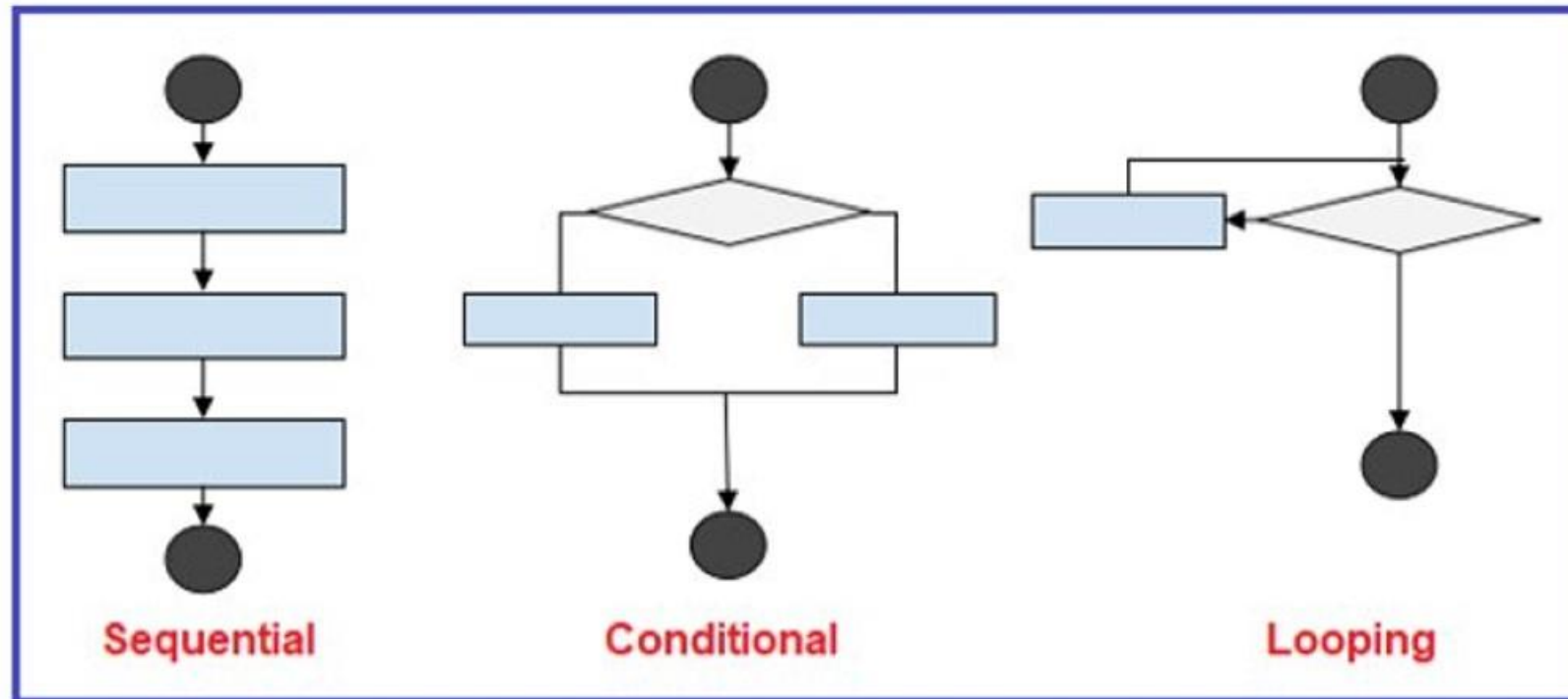
```
print( True or False )
```

```
print( 5 > 3 or 5 == 3 + 2 )
```

Lecture

03

PROGRAM FLOW



PROGRAM FLOW

Cont'd

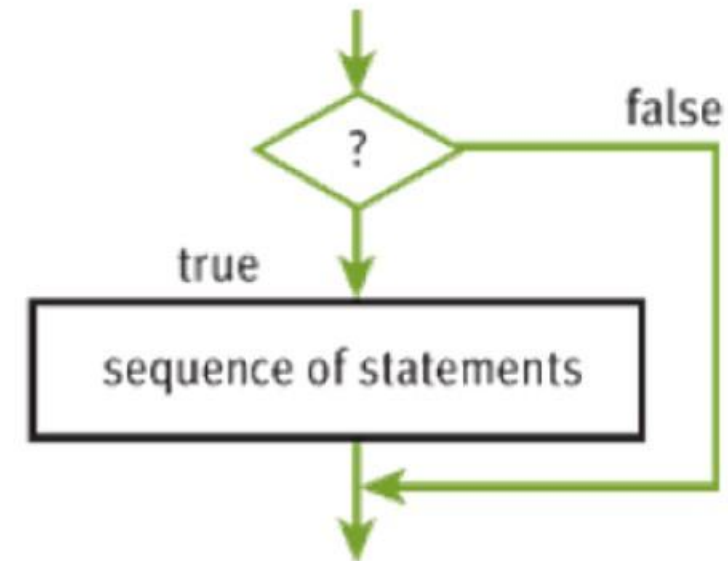
- A program's flow is the order in which the program's code executes.
- **The sequential flow:** a set of statements whose execution process happens in a sequence.
- **The conditional flow:** allows a program to test several conditions and execute instructions based on which condition is true.
- **The looping flow:** used to repeat a group (block) of programming instructions.

Lecture

03

THE CONDITIONAL FLOW
STATEMENTS

- The basic-selection statement (If statement).
 - Syntax:
if <condition> :
 <statement(s)>
 - Example:
 x = 5
 if x == 6 :
 print("Yes")



THE CONDITIONAL FLOW STATEMENTS

Cont'd

- The two-way selection statement (If...else statement).

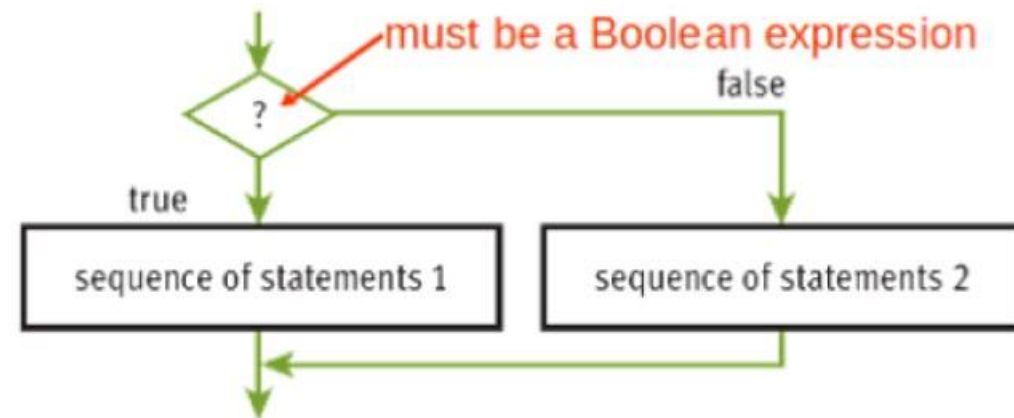
- Syntax:

```
if <condition> :  
    <true statement(s)>
```

```
else:  
    <false statement(s)>
```

- Example:

```
x = 5  
if x == 6 :  
    print( "Yes")  
else:  
    print("No")
```



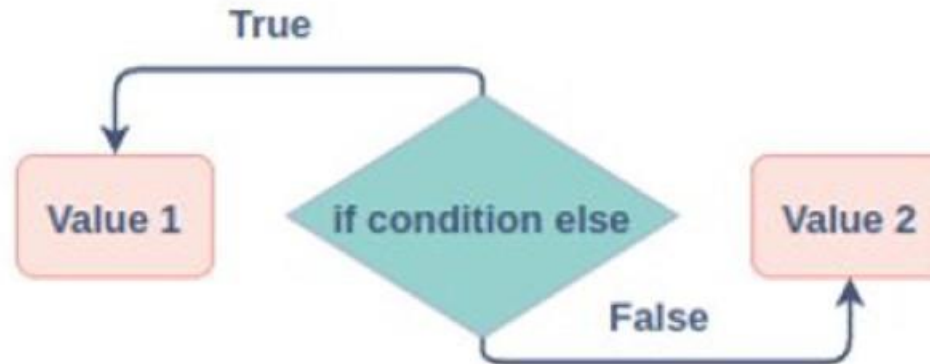
THE CONDITIONAL FLOW STATEMENTS

Cont'd

- The ternary operator statement (If...else statement).

- Syntax:

`<variable> = <true statement> if <condition> else <false statement>`



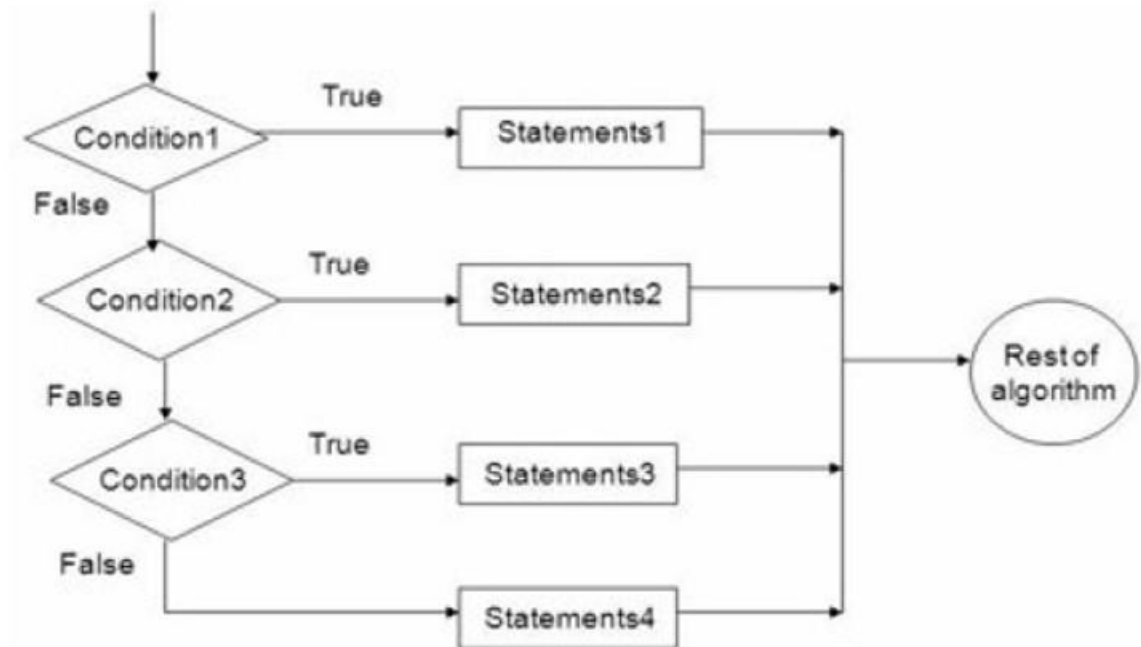
THE CONDITIONAL FLOW STATEMENTS

Cont'd

- The multi-way selection statement (If...elif statement).

- Syntax:

```
if <condition_1> :  
    <true cond_1 statement(s)>  
elif <condition_2> :  
    <true cond_2 statement(s)>  
else:  
    <otherwise statement(s)>
```



THE CONDITIONAL FLOW STATEMENTS

Cont'd

- The nested selection statement (If...if statement).

- Syntax:

if <condition1>:

if <condition2>:

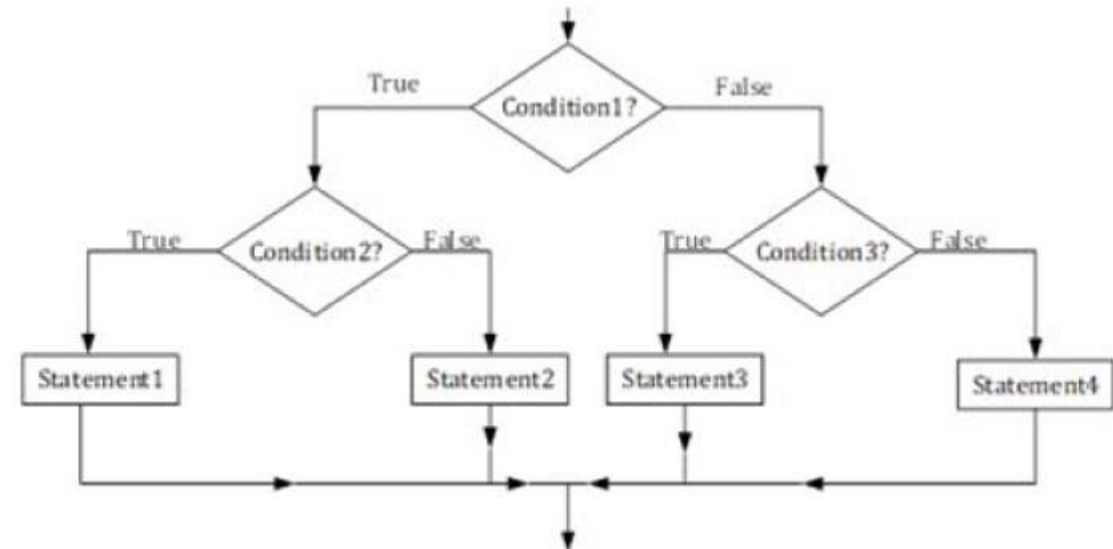
 <true cond2_statement(s)>

else:

 <otherwise statement(s)>

else:

 <otherwise statement(s)>



THE CONDITIONAL FLOW STATEMENTS: EXAMPLES

- Ex3: write a program which inputs a person's age and outputs one of the messages using the criteria specified.
 - You are child: age between 0 and 9
 - You are an adolescent: age between 9 and 18
 - You are an adult: age between 18 and 65
 - Golden ages: age above 65

THE CONDITIONAL FLOW STATEMENTS: EXAMPLES

Cont'd

- Ex3: python code.

```
age = int(input("Enter your age: "))  
if age >= 0 and age <= 9:  
    print("You are a child!")  
elif age > 9 and age <= 18:  
    print("You are an adolescent!")  
elif age > 18 and age <= 65:  
    print("You are an adult!")  
elif age > 65:  
    print("Golden ages!")
```

THE CONDITIONAL FLOW STATEMENTS: EXAMPLES

Cont'd

- Ex4: write a program that accepts the sides (a, b, c) of a triangle and calculates and display its area.

1. Work out $s = \frac{a+b+c}{2}$

2. Area = $\sqrt{s(s-a)(s-b)(s-c)}$

THE CONDITIONAL FLOW STATEMENTS: EXAMPLES

Cont'd

- Ex4: python code.
a = float(input('Enter first side: '))
b = float(input('Enter second side: '))
c = float(input('Enter third side: '))
s = (a + b + c) / 2
area = (s*(s-a)*(s-b)*(s-c)) ** 0.5
print("The area of the triangle = ", area)

THE CONDITIONAL FLOW STATEMENTS: EXAMPLES

Cont'd

- Ex5: write a program that check of a given number is divided by 3 and 4.
 - Python code:

```
Num = int(input())
if Num % 3 == 0 and Num % 4 == 0:
    print("The number is divided by both 3 and 4")
elif Num % 3 == 0 or Num % 4 == 0:
    print("The number is divided by either 3 or 4")
else:
    print("The number is not divided by any of 3 or 4")
```

THE CONDITIONAL FLOW STATEMENTS: EXAMPLES

Cont'd

- Ex6: write a program that check if a single digit integer is a positive or negative.
 - Python code:

```
x = int(input("Enter a single digit number: "))  
if x < 0 and x < 10:  
    print("This number is a single digit positive number")  
elif x < 0 and x > -10:  
    print("This number is a single digit negative number")  
else:  
    print("Invalid integer")
```

THE CONDITIONAL FLOW STATEMENTS: EXAMPLES

Cont'd

- Ex7: Python code for nested if.

```
var = 100
if var < 200:
    print ("Expression value is less than 200")
    if var == 150:
        print ("Which is 150")
    elif var == 100:
        print ("Which is 100")
    elif var < 50:
        print ("Expression value is less than 50")
else:
    print ("Could not find true expression")
```

THE CONDITIONAL FLOW STATEMENTS: EXAMPLES

Cont'd

- Ex8: Python code ternary operator.

```
a = 10
```

```
b = 20
```

```
min = a if a < b else b
```

```
print(min)
```

```
print ("Both a and b are equal" if a == b else "a is greater than b" if a > b else "b  
is greater than a")
```

ITERATION/LOOPING STATEMENTS

- An **iteration statement**, or loop, repeatedly executes a statement, known as the loop body, until the controlling expression is False.
- The **while statement** evaluates the **control expression** before executing the loop body.
- The **for statement** is used to repeat the execution of a set of statements for every element of a **sequence**.

ITERATION/LOOPING STATEMENTS

Cont'd

- The `range` function returns a sequence of numbers between the given range.
 - Examples:
 - `range(10)`
 - `range(-10)` #empty sequence
 - `range(2, 10)`
 - `range(-2, 10)`
 - `range(10, 2)` #empty sequence
 - `range(1, 10, 2)`
 - `range(10, 1, -1)`

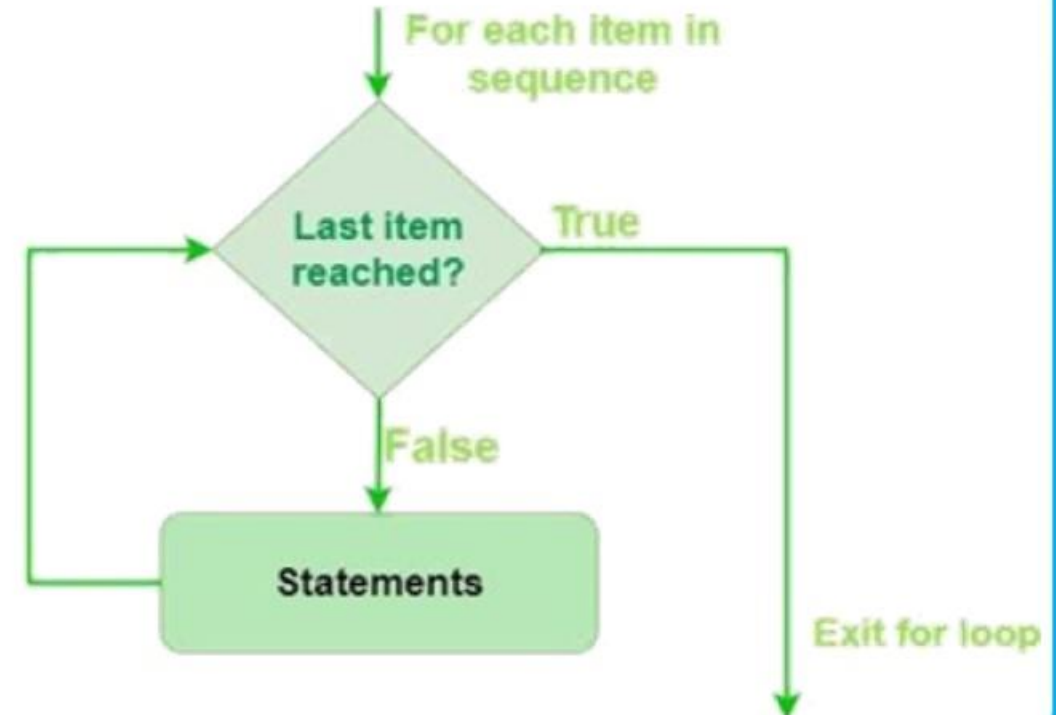
ITERATION/LOOPING STATEMENTS

Cont'd

- The **for** statement:

- Syntax:

```
for <variable> in <sequence> :  
    <statement 1>  
    .....  
    <statement n>
```

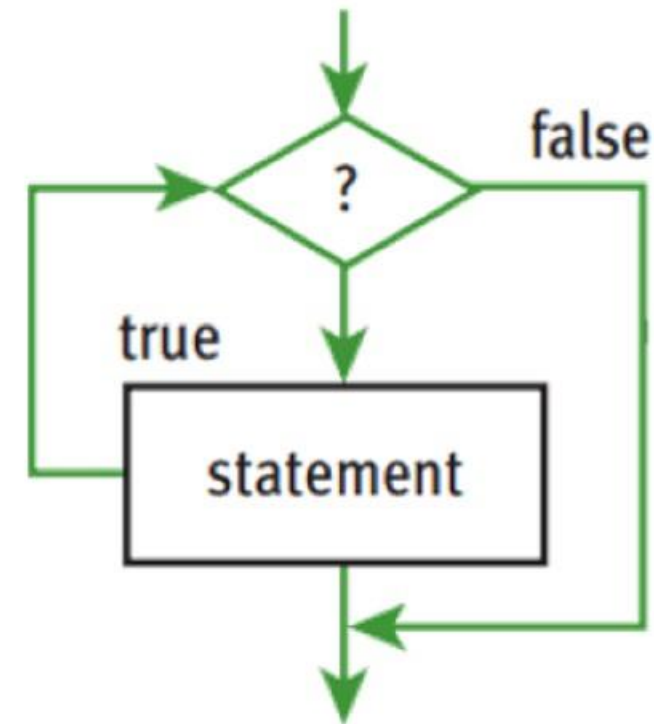


ITERATION/LOOPING STATEMENTS

Cont'd

- The **while** statement:
 - Syntax:
while <condition> :
 <statement 1>

 <statement n>



ITERATION/LOOPING STATEMENTS: EXAMPLES

- Ex9: write a program that reads the number of students in a class then reads their marks and finds and displays the average.

```
stdCount = int(input("Enter number of students in a class"))  
sum = 0.0  
for i in range(stdCount):  
    mark = float(input("Enter mark: "))  
    sum += mark  
print("The marks average = ", sum / stdCount)
```

ITERATION/LOOPING STATEMENTS: EXAMPLES

Cont'd

- Ex10: write a program that calculates the total of the following series of numbers:

$$1/30 + 2/29 + 3/28 + \dots + 30/1$$

```
min = 1
max = 6
sum = 0.0
for i in range(min, max):
    sum += min / (max-1)
    min += 1
    max -= 1
print(sum)
```

ITERATION/LOOPING STATEMENTS: EXAMPLES

Cont'd

- Ex11: write a program that calculates the average of a set of temperature. Data is terminated by the value 999.

```
sum = 0.0
count = 0
temp = float(input("enter a temperature "))
while temp != 999:
    count += 1
    sum += temp
    temp = float(input("enter another temp"))
print("temperature average = ", sum/count)
```

BREAK AND CONTINUE STATEMENTS

- **Break statement:** the loop with exits when a condition is met even if the loop condition or loop sequence does not finish.

- Syntax:

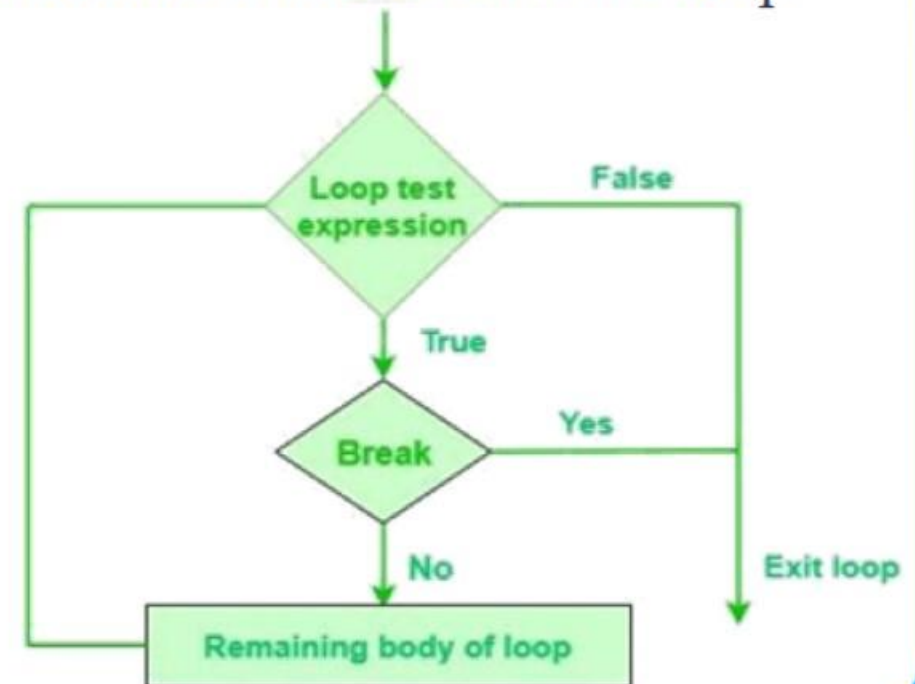
while <condition>:

 <statement 1>

if <condition>:

break

 <statement n>



BREAK AND CONTINUE STATEMENTS

Cont'd

- **Continue statement:** the loop will skip an iteration when a condition is met. The statements under the continue will not execute.

- Syntax:

```
while <condition>:
```

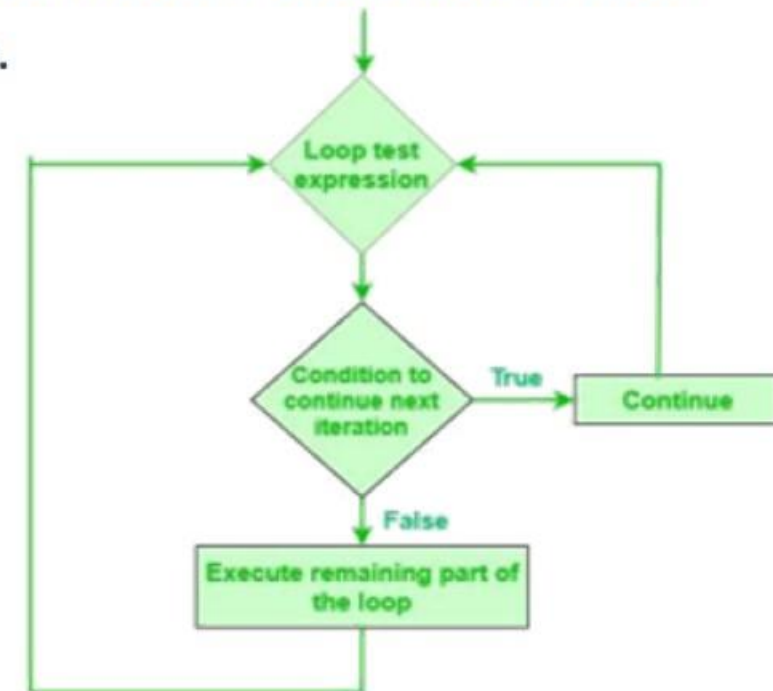
```
    <statement 1>
```

```
    .....
```

```
    if <condition>:
```

```
        continue
```

```
    <statement n>
```



Lecture

03

ITERATION/LOOPING STATEMENTS

- The **do-while** statement:

- Syntax:

```
while True :
```

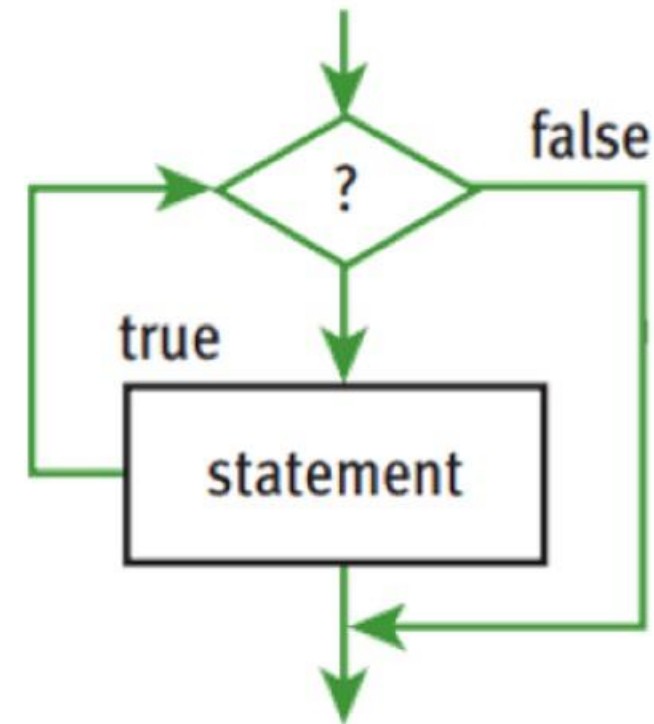
```
    <statement 1>
```

```
    .....
```

```
    if <condition>:
```

```
        break
```

```
    <statement n>
```



BREAK AND CONTINUE STATEMENTS: EXAMPLES

- Ex12: write a program to print numbers from 1 to 10. The program should stop and not printing the number 5.

```
count= 0
while count < 10:
    count += 1
    if count == 5:
        break
    print(count)
```

BREAK AND CONTINUE STATEMENTS: EXAMPLES

Cont'd

- Ex13: write a program to print numbers from 1 to 10 except the number 5.

```
count = 0
while count < 10
    count += 1
    if count == 5:
        continue
    print(count)
```

QUESTIONS FOR ONLINE DISCUSSION

- Write a program that calculates the occupancy rate for a hotel. The program should start by asking the user how many floors the hotel has. A loop should then iterate once for each floor. In each iteration, the loop should ask the user for the number of rooms on the floor and how many of them are occupied. After all the iterations, the program should display how many rooms the hotel has, how many of them are occupied, how many of them are unoccupied. The percentage may be calculated by dividing the number of rooms occupied by the number of rooms.



Thank You

