M110: Python Programming

Meeting #3

Control Structures-1: Decision Structures and Boolean Logic

AOU
الجامعة العربية المفتوحة
Arab Open University
Lebanon

Prepared by Dr. Ahmad Mikati

Contents

If Statement

Control structure:
A control structure is a logical design that controls the order in which a set of statements execute.
So far, we have used the sequence structure. A sequence structure is a set of statements that execute in the order in which they appear.
However, many programs require a different type of control structure: one that can execute a set of statements only under certain circumstances.
This can be accomplished with a ***decision structure***(also known as selection structures).
In a decision structure's simplest form, a specific action is performed only if a certain condition satisfied. If the condition is not satisfied, the action is not performed.

The if statement is used to create a decision structure, which allows a program to have more than one path of execution. The if statement causes one or more statements to execute only when a Boolean expression is true.

If statement

- The general form of an if statement is:

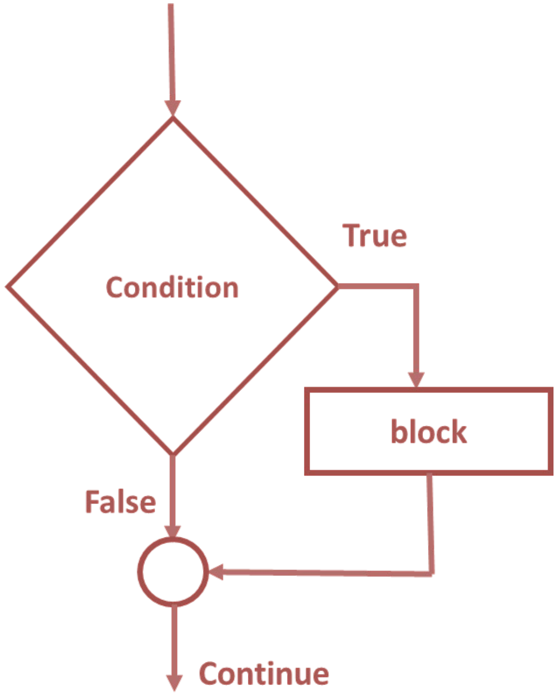
  if *condition*:

  block

- Example

```
if grade >=50:
    print (“pass”)
```

Needs indentation



- **The condition is a Boolean expression**
- **The block could be a series of statements**
- **If the condition evaluates to true, the block is executed**

Indentation-No Braces

- Python relies on **indentation**, using whitespace, to define scope in the code. Other programming languages often use curly-brackets for this purpose.
- All lines must be indented the same amount to be part of the scope (or indented more if part of an inner scope)
- This **forces** the programmer to use proper indentation since the indenting is part of the program!
- In the ***if*** statement, all the block statements are indented. This indentation is required because the Python interpreter uses it to tell where the block begins and ends.

Relational Operators and Boolean Expressions

- The if statement tests an expression to determine whether it is true or false. The expressions that are tested by the if statement are called **Boolean expressions**.
- Typically, the Boolean expression that is tested by an if statement is formed with a **relational operator**.
- A relational operator determines whether a specific relationship exists between two values. For example, the greater than operator (>) determines whether one value is greater than another. The equal to operator (==) determines whether two values are equal.

Relational Operators

Operator	**Meaning**
>	Greater than
<	Less than
>=	Greater than or equal to
<=	Less than or equal to
==	Equal to
!=	Not equal to

Relational Operators and Boolean Expressions

Python supports the usual logical conditions (Boolean expressions) from mathematics:

Boolean Expressions

Expression	**Meaning**
x>y	Is x greater than y?
x<y	Is x less than y?
x>=y	Is x greater than or equal to y?
x<=y	Is x less than or equal to y?
x==y	Is x equal to y?
x!=y	Is x not equal to y?

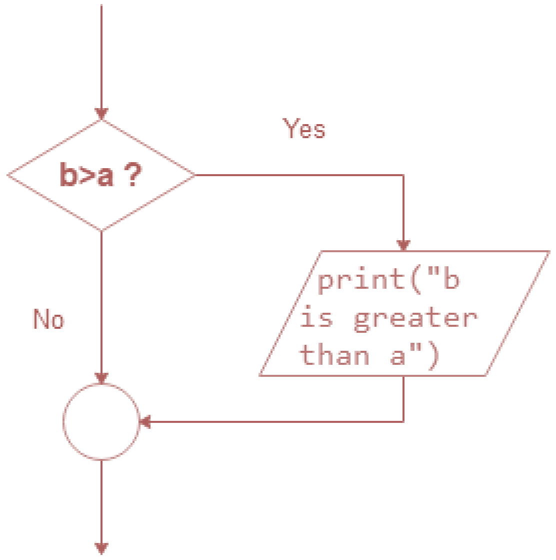
These conditions can be used in several ways, most commonly in "if statements" and loops.

Relational Operators and Boolean Expressions

The following is an example of an expression that uses the greater than (>) operator to compare two variables

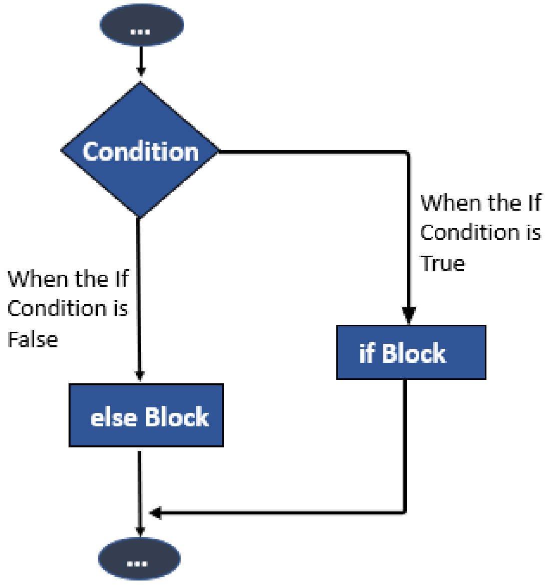
Example:

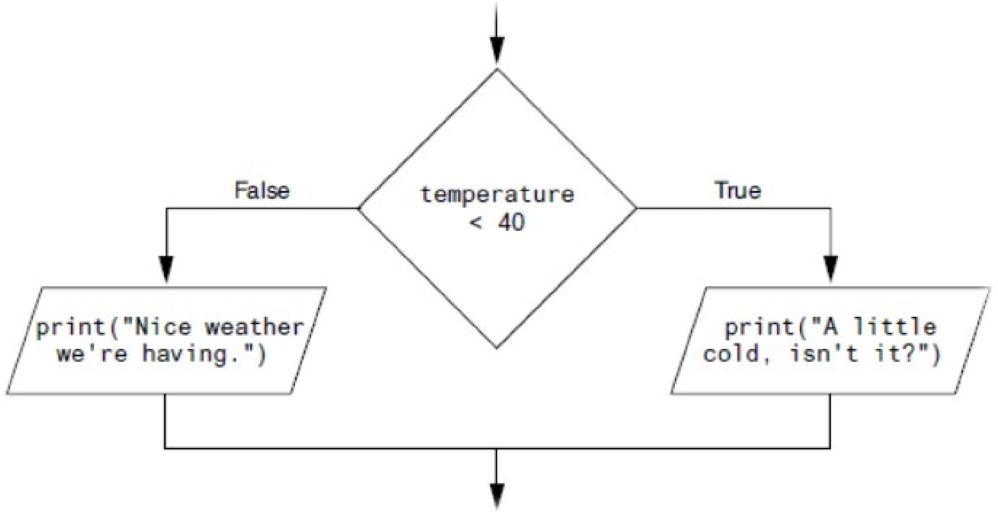
```
a = 33
b = 200
if b > a:
  print("b is greater than a")
```



The if-else Statement

- An **if-else** statement will execute one block of statements if its condition is true, or another block if its condition is false.
- Earlier, we introduced the single alternative decision structure (the if statement), which has one alternative path of execution.
- Dual alternative decision structure has two possible paths of execution—one path is taken if a condition is true, and the other path is taken if the condition is false.
- The below Figures show a flowchart for a dual alternative decision structure




False
temperature < 40
True
print("Nice weather we're having.")
print("A little cold, isn't it?")

The if-else Statement

The **else** keyword catches anything which isn't caught by the preceding condition.

Example:

```
a = 200
b = 33
if b > a:
  print("b is greater than a")
else:
  print(“b is less than or equal to
a")
```

Output:

```
b is less than or equal to a
```

Comparing Strings

Python allows you to compare strings. This allows you to create decision structures that test the value of a string.

Example:

```
name1 = ‘Majed'
name2 = ‘Majeda'
if name1 == name2:
   print('The names are the same.')
else:
   print('The names are NOT the
same.')
```

Output:

```
The names are NOT the same.
```

The == operator compares name1 and name2 to determine whether they are equal.

Because the strings ‘Majed' and ‘Majeda' are not equal, the else clause will display the message 'The names are NOT the same.'

String comparisons are case sensitive.

For example, the strings ‘october' and ‘October’ are not equal because the “o" is lowercase in the first string but uppercase in the second string.

Comparing Strings

Other String Comparisons

In addition to determining whether strings are equal or not equal, you can also determine whether one string is greater than or less than another string. This is a useful capability because programmers commonly need to design programs that sort strings in some order. Computers do not actually store characters, such as A, B, C, and so on, in memory. Instead, they store numeric codes that represent the characters.

ASCII (the American Standard Code for Information Interchange) is a commonly used character coding system.

In addition to establishing a set of numeric codes to represent characters in memory, ASCII also establishes an order for characters. The character “A” comes before the character “B”, which comes before the character “C”, and so on.

Comparing Strings

Other String Comparisons (cont'd)

When a program compares characters, it actually compares the codes for the characters.
For example, look at the following if statement:

```
if ‘d' < ‘e’:
    print('The letter d is less than the letter e.’)
```

This code determines whether the ASCII code for the character ‘d' is less than the ASCII code for the character ‘e'. The expression ‘d' < ‘e' is true because the code for ‘d’ is less than the code for ‘e’.

Moreover, if one of the strings in a comparison is shorter than the other, only the corresponding characters will be compared. If the corresponding characters are identical, then the shorter string is considered less than the longer string.

The Nested if Statement

When you have if statements inside if statements, this is called nested if statements.

Example:

```
x = 15
if x > 10:
    print(“The number is above ten,“, end=‘ ‘)
    if x > 20:
      print("and also above 20!")
    else:
      print("but not above 20.")
else:
      print("The number is less than or equal to
ten!")
```

Output:

```
The number is above ten, but not above
20.
```

The if-elif-else Statement

- The ***elif*** is short for **else if**. It allows us to check for multiple expressions.
- If the condition for **if** is False, it checks the condition of the next **elif** block and so on.
- If all the conditions are False, the body of **else** is executed.
- Only one block among the several if...elif...else blocks is executed according to the condition.
- The if block can have only one else block, but it can have multiple elif blocks.

The if-elif-else Statement

Flowchart of the if…elif…else statement

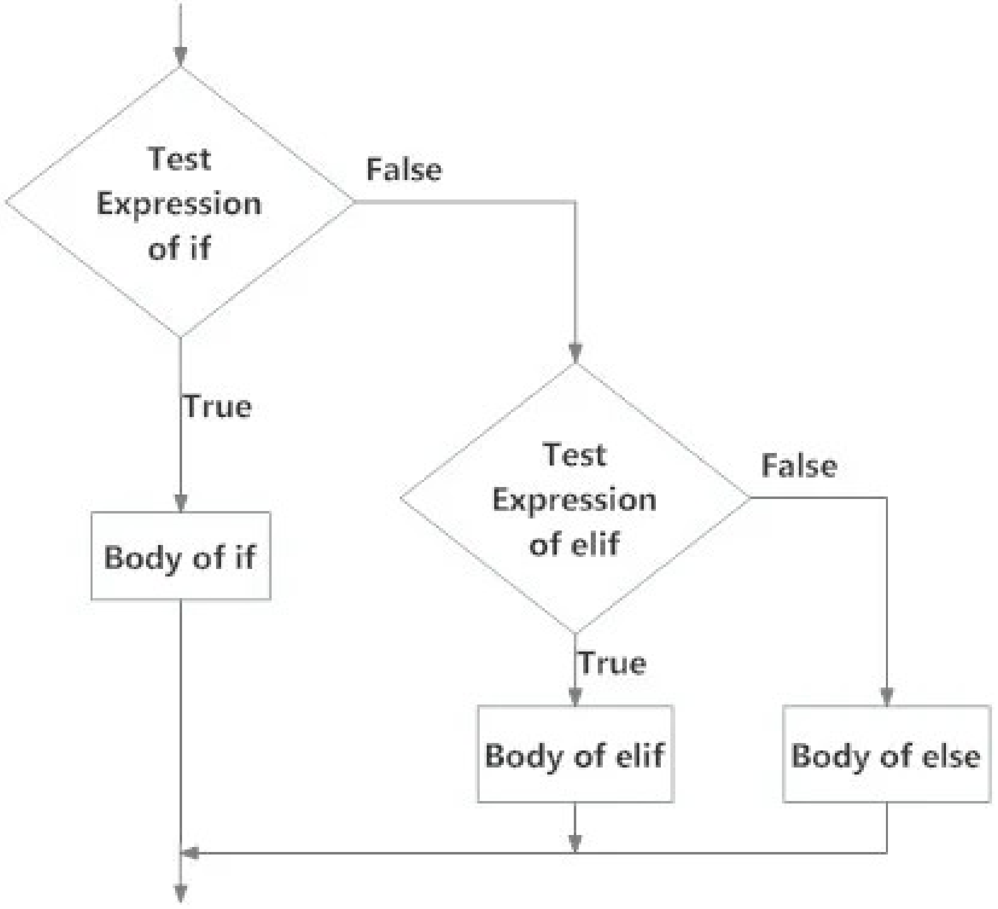


Fig: Operation of if…elif…else statement

The if-elif-else Statement

- The elif keyword is a python’s way of saying "if the previous conditions were not true, then try this condition".
- The **else** keyword catches anything which isn't caught by the preceding condition.

Example:

```
a = 33
b = 33
if b > a:
  print("b is greater than a")
elif a == b:
  print("a and b are equal")
else:
  print(“b is less than a")
```

Output:

```
a and b are equal
```

The if-elif-else Statement

Example: Given the following grading scale, you are asked to write a program that will allow a student to enter a test score and then display the proper message for that score.

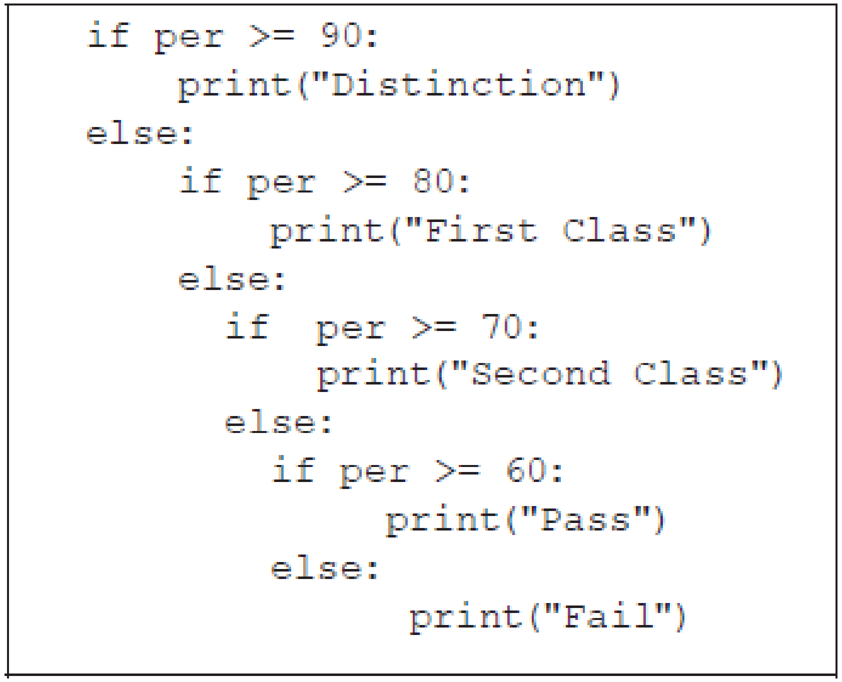
Percentage	Message
90 and above	Distinction
80–89	First Class
70–79	Second Class
60–69	Pass
Below 60	Fail

Below is the algorithm that you will use:

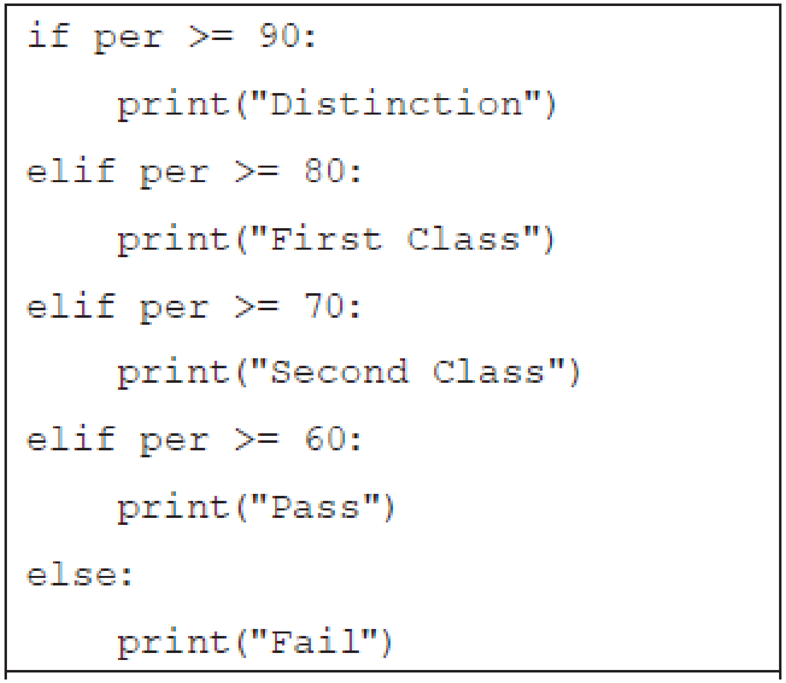
```
If the Percentage is greater than or equal to 90, then the message is Distinction.
  Else, if the Percentage is greater than or equal to 80, then the message is First Class.
    Else, if the Percentage is greater than or equal to 70, then the message is Second Class.
      Else, if the Percentage is greater than or equal to 60, then the message is Pass.
        Else, the message is Fail.
```

The if-elif-else Statement

Example: Below are two equivalent codes that you might write



```
if per >= 90:
    print("Distinction")
else:
    if per >= 80:
        print("First Class")
    else:
      if  per >= 70:
          print("Second Class")
      else:
        if per >= 60:
             print("Pass")
        else:
              print("Fail")
```



```
if per >= 90:

    print("Distinction")

elif per >= 80:

    print("First Class")

elif per >= 70:

    print("Second Class")

elif per >= 60:

    print("Pass")

else:

    print("Fail")
```

Which one is easier for a user to read?

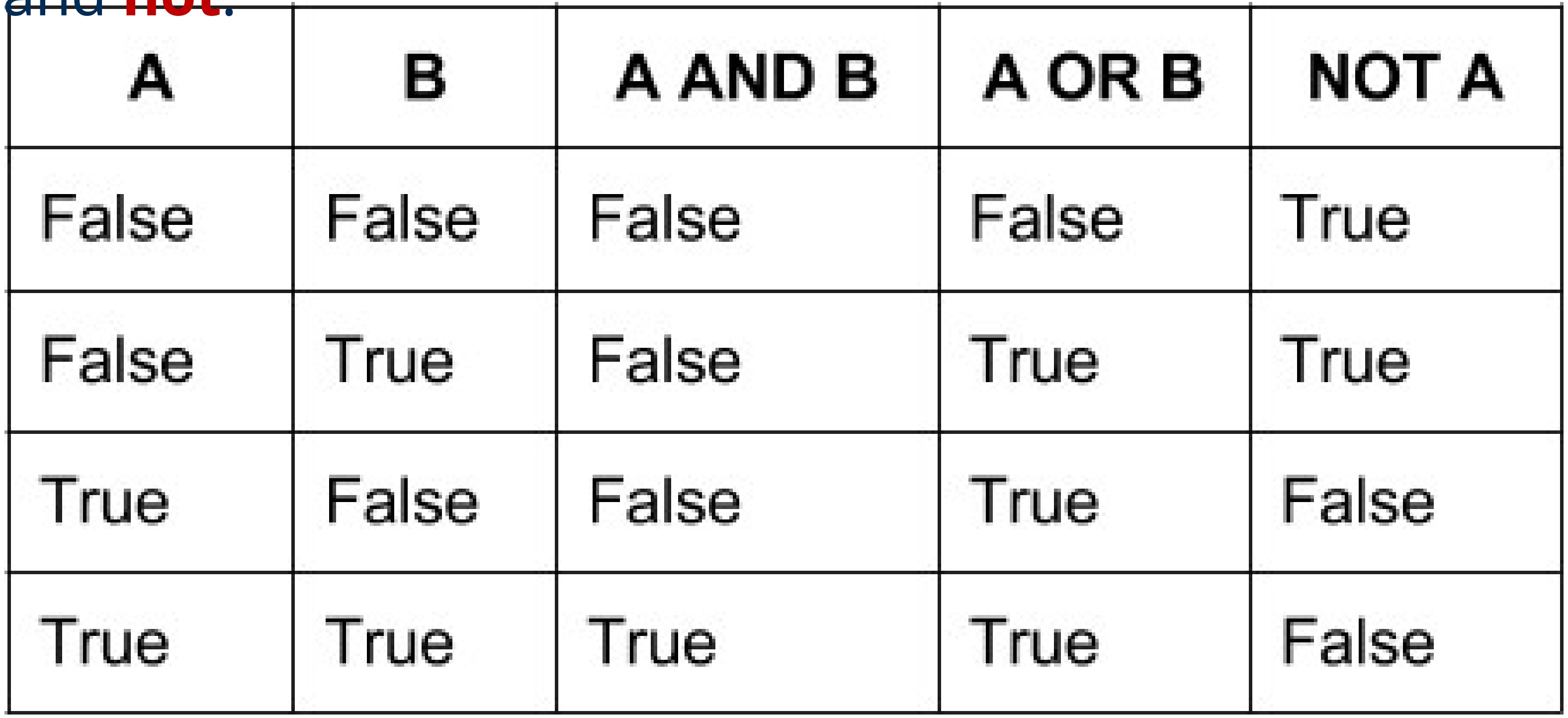
Logical Operators

- The logical and operator and the logical or operator allow you to connect multiple Boolean expressions to create a compound expression.
- Python provides a set of operators known as logical operators, which you can use to create compound Boolean expressions.

Operator	Meaning
and	The and operator connects two Boolean expressions into one compound expression. Both subexpressions must be true for the compound expression to be true.
or	One or both subexpressions must be true for the compound expression to be true. It is only necessary for one of the subexpressions to be true.
not	The not operator reverses the truth of its operand. If it is applied to an expression that is true, the operator returns false and vice versa.

Logical Operators

Here is the truth table of the Boolean operators: **and** , **or** and **not**:

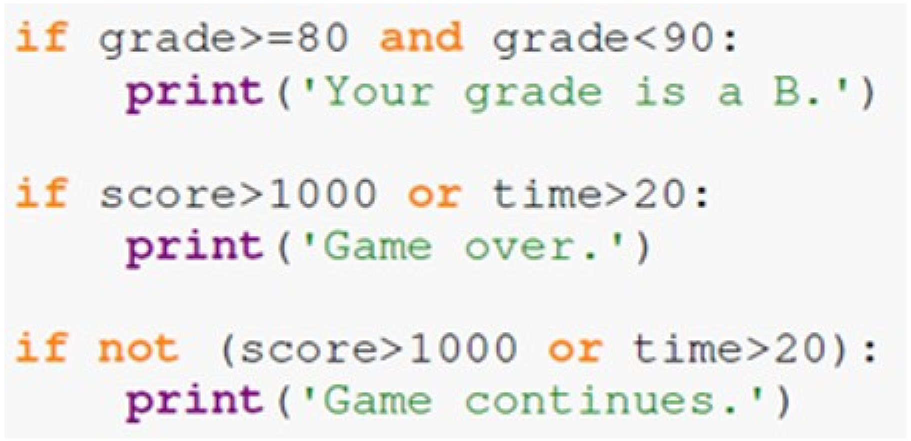


A	B	A AND B	A OR B	NOT A
False	False	False	False	True
False	True	False	True	True
True	False	False	True	False
True	True	True	True	False

Logical Operators

- Below are some examples of **Compound Boolean expressions** using logical operators

Expression	Meaning
x > y and x < z	Is x greater than y **AND** is x less than z?
x == y or x == z	Is x equal to y **OR** is x equal to z?
not (z > y)	Is the expression z > y **NOT** true?



```
if grade>=80 and grade<90:
    print('Your grade is a B.')

if score>1000 or time>20:
    print('Game over.')

if not (score>1000 or time>20):
    print('Game continues.')
```

Boolean Variables

In addition to **int**, **float**, and **str** (string) variables, Python also provides a **bool** data type which allows you to create Boolean variables.

A Boolean variable can reference one of two values: True or False.
Boolean variables are commonly used as flags, which indicate whether specific conditions exist.
A flag is a variable that signals when some condition exists in the program. When the flag variable is set to False, it indicates the condition does not exist. When the flag variable is set to True, it means the condition does exist.

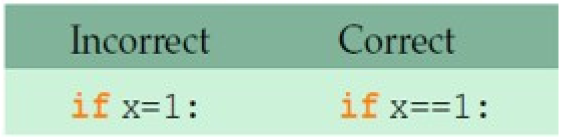
Example:

```
if sales >= 50000.0:
    sales_quota_met = True
else:
    sales_quota_met = False

if sales_quota_met == True:  # or you can just write  if sales_quota_met:
    print('You have met your sales quota!')
```

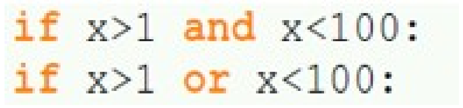
Common Mistakes

Mistake 1: The operator for equality consists of two equal signs. It really a common mistake to forget on of the equal signs



Incorrect	Correct
`if x=1:`	`if x==1:`

Mistake 2: A common mistake is to use and where or is needed and vice versa. Consider the following if statements.



```
if x>1 and x<100:
if x>1 or x<100:
```

Exercises

1. Write a program that asks the user to enter a length in centimeters. If the user enters a negative length, the program should tell the user that the entry is invalid. Otherwise, the program should convert the length to inches and print out the result. There are 2.54 centimeters in an inch.

Exercises

Solution of Ex 1:

```
length = float(input(“Enter the length in Centimeters: "))
if length <0:
    print("Length should be positive!!")
else:
    inch = length/ 2.54
    print(length, “Centimeters is: ",inch," Inches")
```

Output:

```
 Enter the length in Centimeters : 20
 20.0 Centimeters  is: 7.874015748031496  Inches
>>>
```

Note:

- You can use the round function to round the result:

```
print(length, “Centimeters is: ",round(inch,1)," Inches")
```

- In this case, the output will be rounded into **1** decimal place:

```
20.0 Centimeters  is:  7.9  Inches
```

Exercises

2. Ask the user for a temperature. Then ask them what units, Celsius or Fahrenheit, the temperature is in.

Your program should convert the temperature to the other unit. The conversion Formulas are:

- From Celsius to Fahrenheit: **F = 9/5 C + 32**
- From Fahrenheit to Celsius: **C = 5 (F -32) /9**

Exercises

Solution of Ex 2:

```
temp = float(input(“Enter the temperature: "))
unit = input(“Enter the unit: C/F: ")
if unit == "C":
    fah = 9/5 * temp + 32
    print(temp," Celsius is ",fah," Fahrenheit")
else:
    cel = 5/9*(temp -32)
    print(temp," Fahrenheit is ",cel," Celsius")
```

Output:

```
Enter the temperature: 50
Enter the unit: C/F: C
50.0  Celsius is  122.0  Fahrenheit
>>>
```

AOU
الجامعة العربية المفتوحة
Arab Open University
Lebanon

Exercises

3. Write a program that asks the user to enter the age of the customer , then display the proper message as follows:

- If the age is greater than or equal to 60, the message is: “ Senior Discount”.
- If the age is greater than or equal to 18 and less than 60 , the message is: “ No Discount”.
- If the age is less than 18, the message is: “ Junior Discount”.

Solution:

```
age= eval(input('Enter the age of the customer: '))
if age>=60:
    print('Senior Discount')
elif age>=18:
    print('No Discount')
else:
    print('Junior Discount')
```

Exercises

4. Scientists measure an object’s mass in kilograms and its weight in newtons. If you know the amount of mass of an object in kilograms, you can calculate its weight in newtons with the following formula:
***weight* = *mass* * 9.8**

Write a program that asks the user to enter an object’s mass, then calculates its weight. If the object weighs more than 500 newtons, display a message indicating that it is too heavy.
If the object weighs less than 100 newtons, display a message indicating that it is too light.

Exercise 4- Solution

```
# Global constants
MASS_MULTIPLIER = 9.8
TOO_HEAVY = 500.0
TOO_LIGHT = 100.0
# Local variables
weight = 0.0
mass = 0.0
# Get mass
mass = float(input("Enter the object's mass in kilograms: "))
# Calculate weight
weight = mass * MASS_MULTIPLIER
# Display weight evaluation
print ('Object Weight:’, round(weight,2))
if weight > TOO_HEAVY:
    print ('The object is too heavy. It weighs more than ', TOO_HEAVY,'
Newtons.')
elif  weight < TOO_LIGHT:
    print ('The object is too light. It weighs less than ‘ , TOO_LIGHT, '
Newtons.')
```

Output:

```
Enter the object's mass in kilograms: 100
Object Weight: 980.0
The object is too heavy. It weighs more than  500.0
Newtons.
```

Extra Exercises

1. Find the value of x and y after executing the following cod

```
x = 0
y = 5
if x>=0 and y <10:
    x+=1
    y+=1
else:
    x-=2
    y-=2
print(x)
print(y)
```

```
x = 0
y = 5
if not (x>=0 and y
<10):
    x+=1
    y+=1
else:
    x-=2
    y-=2
print(x)
print(y)
```

```
x = 2
y = 10
if x>5 or y >10:
    x*=3
    y*=3
else:
    x%=4
    y%=4
print(x)
print(y)
```

```
x = 2
y = 10
if not x>5 or y >10:
    x*=3
    y*=3
else:
    x%=4
    y%=4
print(x)
print(y)
```

Extra Exercises

2. Write a program that reads from the user an integer and displays on the screen a message if this integer is odd or even.
3. Write a program reads from the user 2 positive integers and displays on the screen a message if the first integer is a multiple of the second one or not.
4. Write a program that reads from the user an integer and displays on the screen a message if this integer is divisible by both 2 and 3 or not.
5. Write a program that reads from the user an integer and displays on the screen a message if this integer is positive, negative or zero.
6. Write a program that reads from the user the base and the height of a triangle. The program calculates and prints the area of the triangle using the formula: Area = $\frac{1}{2}$ base x height

   Note: if the base or height is zero or negative the program should print error message and the area should not be calculated.

Extra Exercises

7. Write a program that reads from the user the lengths of the bases and the heights of two triangles. The program should calculate the area of the two tringles and print them. Then, it should print a message that indicates which triangle has a larger area or if they have the same area.
8. Write a program that reads from the user a student's grades in TMA, MTA and final exam. The program should calculate the total grade and displays in the screen if the student passed or not. A student passes a course if the total grade is 50 or more.
9. Write a program that reads 3 scores (quiz, midterm, and final). Calculate the total score and determine and print the grade based on the following rules:
   - if the total score >= 90 → grade=A
   - if the total score >= 70 and <90 → grade=B
   - if the total score>=50 and <70 → grade=C
   - if the total score<50 → grade=F