|  |
| --- |
| Worksheet III |

|  |  |
| --- | --- |
| Expression | Meaning |
| x == y | True if x = y (mathematical equality, not assignment; otherwise, false |
| x < y | True if x < y; otherwise, false |
| x <= y | True if x ≤ y; otherwise, false |
| x > y | True if x > y; otherwise, false |
| x >= y | True if x ≥ y; otherwise, false |
| x != y | True if x ≠ y; otherwise, false |

Examples

 10 < 20 True

 10 >= 20 False

 x < 100 True if x Is less than 100; otherwise, False

 x != y True unless x and y are equal



Operator **not** has higher precedence than both **and** and **or**.

**and** has higher precedence than **or**.

**and**, **or** are left associative; **not** is right associative.

**and**, **or** have lower precedence than any other binary operator except assignment.

1. Fill in the blank

x = 10

y = 20

b = (x == 10) # assigns True to b

b = (x != 10) # assigns False to b

b = (x == 10 and y == 20) # assigns \_\_\_\_\_\_\_\_ to b

b = (x != 10 and y == 20) # assigns \_\_\_\_\_\_\_\_ to b

b = (x == 10 and y != 20) # assigns \_\_\_\_\_\_\_\_ to b

b = (x != 10 or y != 20) # assigns \_\_\_\_\_\_\_\_ to b

b = (x == 10 or y == 20) # assigns \_\_\_\_\_\_\_\_ to b

b = (x != 10 and y == 20) # assigns \_\_\_\_\_\_\_\_ to b

b = (x == 10 or y != 20) # assigns \_\_\_\_\_\_\_\_ to b

b = (x != 10 and y != 20) # assigns \_\_\_\_\_\_\_\_ to b

1. Given the following definitions

x = 13, y = 5

b1, b2, b3, b4 = True, False, x == 3, y < 3 # This statement assigns Boolean value in

# accordance with the order of the appearance, e.g., # b1 is assigned value of **True**. x == 3 is **False**,

# hence b3 is assigned value of **False**.

Evaluation the following Boolean expressions (True or False)

b1 or b2 and b3 ­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_

b1 and b2 and b3 ­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_

b1 or b2 or b3 ­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_

not b1 and b2 and b3 ­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_

not b1 or b2 or b3 ­­­ \_\_\_\_\_\_\_\_\_\_\_\_\_\_

not (b1 and b2 and b3) ­­­ \_\_\_\_\_\_\_\_\_\_\_\_\_\_

not (b1 or b2 or b3) ­­­ \_\_\_\_\_\_\_\_\_\_\_\_\_\_

not b1 and not b2 and not b3 ­­­ \_\_\_\_\_\_\_\_\_\_\_\_\_\_

not b1 or not b2 or not b3 ­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_

not (not b1 and not b2 and not b3) ­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_

not (not b1 or not b2 or not b3) ­­­\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. From the following Python code, what is the output (on the screen) if we enter …

Try to answer the above questions by yourself this before writing the code in the Jupyter notebook.



 a) 7, output = \_\_\_\_\_\_\_\_\_\_ b) 5, output = \_\_\_\_\_\_\_\_\_\_\_ c) 23, output = \_\_\_\_\_\_\_\_\_\_\_\_\_



 a) 4, output = \_\_\_\_\_\_\_\_\_\_\_\_ b) 7, output = \_\_\_\_\_\_\_\_\_\_\_

 Is there any integer value that causes the code to print **Superman** on the screen? If yes, which number? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ If no, why? \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1. Trace the value of each variable for the following Python code, while i = 2, 3 and 4 when the loop is executed. (Answer the question by yourself before writing the code on the Jupyter notebook to crosscheck your answer.)



|  |  |  |
| --- | --- | --- |
| n | i | output |
|  | 2 |  |
|  | 3 |  |
|  | 4 |  |

1. Trace the value of each variable for the following Python code (Answer the question by yourself before writing the code on the Jupyter notebook to crosscheck your answer.)



|  |  |  |  |
| --- | --- | --- | --- |
| Loop no. | i | output  | conditioni <= n |
| 1 |  |  |  |
| 2 |  |  |  |

1. Write a Python code that takes an integer, namely my\_num, and then iteratively print only an even number starting from my\_num to the last integer which is greater than 0.
2. To compute 3 + 7 + 11 + 15 +19 + 23, fill in the blank shown in the two codes below

 

 first blank = \_\_\_\_\_\_

 second blank = \_\_\_\_\_\_\_

 

 first blank = \_\_\_\_\_\_\_\_

 second blank = \_\_\_\_\_\_\_\_\_\_

 third blank = \_\_\_\_\_\_\_\_\_

1. The following Python code computes (1\*k) + (2\*k) + (3\*k) + … + (7\*k)

 

* Modify the code to compute (1\*k) + (2\*k) + (3\*k) + … + (n\*k), where *n* and *k* are two integer inputs.
* Write a Python code using while-loop to achieve the above computation.
1. Write a Python code to take three inputs, namely n, k and v and then compute

 $\frac{1\*2k}{v}+\frac{1\*2k}{v^{2}}+\frac{1\*2k}{v^{3}}…+\frac{1\*2k}{v^{n-1}}$

1. The following Python code computes the balance in the account with the interest rate of 10% (assume that the initial deposit is 1,000 baht). Hence after 1 year of deposit, your account balance is 1,100 baht (initial deposit = 1,000 + interest earned, hence account balance = 1,100).

Try the following Python code and see the result.



From the above example, the Python code is modified to compute the account balance for 5 years.



* **Modify the above Python code to take the number of years, interest rate and initial account balance, and compute the account balance accordingly.**
1. The following Python code prints the outputs (strings in the case) the following formatting. The purpose of adding a value after % sign is to tell Python how many spaces you want to add before printing out the value. Try the following codes to observe the results.



* Adjust the parameters in the second print( ) statement in the Python code below so that the output is shown as

 



1. Based on what you’ve learned from exercises 10 and 11,

Write a Python code using for-loop that computes the total balance in the account after depositing for *n* years with the initial deposit of *x* Baht, and the interest rate is fixed at y% per year. The interest is calculated at each completion of one year and the interest earned is deposited into the account.

Show the result in the following format. (In the following result, we assume that the first deposit you enter is 1000, interest rate is 10% and the number of years is 5)

 

1. Complete the exercise 12 again using while-loop.
2. Python provides a powerful function called eval( ) that is sometimes used to facilitate obtaining input. Try the following Python script and observe how eval( ) works.



* Write a Python code to read your height [in cm] and weight [in kg] using eval( ) and print your height and weight in pounds and inches, respectively. (Note that, all inputs entered must be separated by comma)
1. Modify the code in exercise 12 to use eval( ) function to read initial account balance, interest rate and years

 

Note: eval() function can also be used to take an input in a form of equation as shown in the example below.



1. Write a Python code that computes the total balance in the account after depositing for n years with the first deposit of x Baht, and the interest rate is fixed at y% in the first year and is increased 1% every year. The interest is calculated at each completion of one year and the interest earned is deposited into the account. The example of output is shown below.

