Name: ……………………………………………….. ( ) Class: ……… Date: …………………..

|  |  |  |
| --- | --- | --- |
| **4.1** | **Program Development** | **Arithmetic and Relational Operators** |

Python is a powerful calculator. You can carry out many arithmetic operations in Python.

**Arithmetic Operators**

1. Write down the code necessary to perform the following arithmetic operations in Python, then try out your code and record the results:

|  |  |  |
| --- | --- | --- |
| **Arithmetic Operation** | **Python Code** | **Actual Output** |
| Add 155 to 456 |  |  |
| Subtract 87 from 123 |  |  |
| Divide 256 by 8 |  |  |
| Multiply 47 by 13 |  |  |
| Divide 63 by 21, then add 6 |  |  |

1. Perform the following operations and record the results to find out what the //, \*\* and % operators in Python do:

|  |  |  |  |
| --- | --- | --- | --- |
| **Arithmetic Operation** | **Actual Output** | **Name of operator** | **Description** |
| 47 **//** 5 |  |  |  |
| 24 **%** 5 |  |  |  |
| 2 **\*\*** 4 |  |  |  |

1. Assign the following values to the variables a, b, c and d: a = 15, b = 7, c = 5, d = 2

Then, perform the following operations and record the results:

|  |  |  |
| --- | --- | --- |
| **No** | **Operation** | **Actual Output** |
| 1 | a + b |  |
| 2 | a - b |  |
| 3 | a / b |  |
| 4 | a \* b |  |
| 5 | c // d |  |
| 6 | c % d |  |
| 7 | c \*\* d |  |

**More Assignment Operators**

In Python, besides the = assignment operator, there are other assignment operators such as:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **+=** | **-=** | **\*=** | **/=** | **%=** | **\*\*=** | **//=** |

For example, x += 2 means the same thing as x = x + 2:

|  |  |  |
| --- | --- | --- |
| **Code** | **Actual Output** | **What it means** |
| >>> **x = 10** |  | x is assigned a value of 10 |
| >>> **x += 2** |  | x is assigned the result of x plus 2 |
| >>> **x** | 12 | x now has a value of 10 + 2, which is 12 |

1. Predict what will be printed for the following code, then write down the actual output:

|  |  |  |
| --- | --- | --- |
| **Code** | **Expected Output** | **Actual Output** |
| >>> **a = 9**  >>> **b = 3**  >>> **a -= b**  >>> **print(a)** |  |  |
| >>> **a = 9**  >>> **b = 3**  >>> **a /= b**  >>> **print(a)** |  |  |
| >>> **a = 3**  >>> **b = 4**  >>> **a \*\*= b**  >>> **print(a)** |  |  |
| >>> **a = 4**  >>> **b = 3**  >>> **a %= b**  >>> **print(a)** |  |  |

1. Write a program that asks the user for a triangle’s base and height (in cm), and then outputs the triangle’s area (in cm2). *(Hint: You will need to use the input() and float() functions.)*

Below are some samples of how the program should behave when it is working as intended:

|  |  |
| --- | --- |
| **Sample 1** | Enter base of triangle: **2**  Enter height of triangle: **3**  The area of the triangle is: 3.0 |
| **Sample 2** | Enter base of triangle: **0.4**  Enter height of triangle: **0.5**  The area of the triangle is 0.1 |

|  |
| --- |
|  |

1. Write a program based on the algorithm below. Run the program with different inputs to verify that the output is always 3. *(Hint: You will need to use the input() and int() functions.)*

|  |
| --- |
| **Step 1:** Ask the user for an integer from 1 to 10 inclusive  **Step 2:** Multiply the input by 2  **Step 3:** Add 6 to the result  **Step 4:** Divide the result by 2  **Step 5:** Subtract the original input from the result  **Step 6:** Output the result |

Below are some samples of how the program should behave when it is working as intended:

|  |  |
| --- | --- |
| **Sample 1** | Enter an integer from 1 to 10 (inclusive): **5**  The result is: 3.0 |
| **Sample 2** | Enter an integer from 1 to 10 (inclusive): **10**  The result is: 3.0 |

|  |
| --- |
|  |

**Order of Operations**

1. The order of operators in Python is important, as illustrated by the following example. Perform the following operations and write down the actual output:

|  |  |
| --- | --- |
| **Mathematical Operation** | **Actual Output** |
| 90 + 86 + 71 + 100 + 98 / 5 |  |
| (90 + 86 + 71 + 100 + 98) / 5 |  |

Which of the above operations correctly computes the average of 90, 86, 71, 100 and 98?

………………………………………………………………………………………………………………………………………………………

**Math Functions**

1. Enter and run the following program. Note that the first line imports the math module:

|  |
| --- |
| **import math**  **print("math.sqrt(9):", math.sqrt(9))**  **print("math.sqrt(16):", math.sqrt(16))** |

Write out the actual output of the program:

|  |
| --- |
|  |

1. Write down the Math symbol for math.sqrt() ………………………
2. Write a program that asks the user for a right-angled triangle’s base and height (in cm), and then outputs the right-angled triangle’s hypotenuse (in cm). *(Hint: You will need to use the input() and float() functions.)*

Below are some samples of how the program should behave when it is working as intended:

|  |  |
| --- | --- |
| **Sample 1** | Enter base of right-angled triangle: **3**  Enter height of right-angled triangle: **4**  The hypotenuse of the triangle is: 5.0 |
| **Sample 2** | Enter base of right-angled triangle: **1.0**  Enter height of right-angled triangle: **1.0**  The hypotenuse of the triangle is: 1.4142135623730951 |

|  |
| --- |
|  |

**Boolean Values**

A boolean variable stores one of only two possible values, either True or False. It is used in logical tests as well as to make decisions in programs.

1. Write down the expected output of the following Boolean statements. Then test the code in the Python shell and write down the actual output. If you expect an error to occur or if an actual error does occur, label the output as “Error”:

|  |  |  |
| --- | --- | --- |
| **Boolean Statement** | **Expected Output** | **Actual Output** |
| **False or False** |  |  |
| **False or True** |  |  |
| **TRUE or True** |  |  |
| **True or True** |  |  |

1. Write down the expected output of the following Boolean statements. Then test the code in the Python shell and write down the actual output. If you expect an error to occur or if an actual error does occur, label the output as “Error”.

|  |  |  |
| --- | --- | --- |
| **Boolean Statement** | **Expected Output** | **Actual Output** |
| **Not True** |  |  |
| **not False** |  |  |
| **NOT False** |  |  |

**Relational Operators**

There are six relational operators for comparing values that return a boolean value.

|  |  |
| --- | --- |
| **Relational Operator** | **What it means** |
| == | equals |
| != | not equals |
| < | less than |
| > | greater than |
| <= | less than or equal to |
| >= | greater than or equal to |

Note that == is different from =. == is used as an equality check which returns either True or False, while = is used for the assignment of a value to a variable.

1. Write down the expected output of the following Boolean statements. Then test the code in Python shell and write down the actual output.

|  |  |  |
| --- | --- | --- |
| **Boolean Statement** | **Expected Output** | **Actual Output** |
| **2 == 2 and 3 == 2** |  |  |
| **"hello" == "hello"** |  |  |
| **2 == 2 or 3 != 2** |  |  |
| **True and 1 == 1** |  |  |
| **False and 0 != 0** |  |  |
| **True or 1 == 1** |  |  |
| **"test" == "testing"** |  |  |
| **1 != 0 and 2 == 1** |  |  |
| **"test" != "testing"** |  |  |
| **"test" == 1** |  |  |
| **not (True and False)** |  |  |
| **not (1 == 1 and 0 != 1)** |  |  |
| **not (10 == 1 or 1000 == 1000)** |  |  |
| **not (1 != 10 or 3 == 4)** |  |  |
| **not ("testing" == "testing" and "Bala" == "Cool")** |  |  |
| **1 == 1 and (not ("testing" == 1 or 1 == 0))** |  |  |
| **"good" == "not bad" and (not (3 == 4 or 3 == 3))** |  |  |
| **5 == 5 and (not ("testing" == "testing" or "Python" == "Fun"))** |  |  |

1. The following program will check an integer for two possible outcomes: either it is "less than 5", or it is "equal to or more than 5".

However, the program's indentation and logic is incorrect. There are 4 errors.

|  |
| --- |
| **X = int(input("Enter an integer between 1 and 10, inclusive: "))**  **if x >= 6:**  **print("Equal to or more than 5")**  **else:**  **print("Less than 5")** |

Below are some samples of how the program should behave when it is working as intended:

|  |  |
| --- | --- |
| **Sample 1** | Enter an integer between 1 and 10, inclusive: **5**  Equal to or more than 5 |
| **Sample 2** | Enter an integer between 1 and 10, inclusive: **4**  Less than 5 |
| **Sample 3** | Enter an integer between 1 and 10, inclusive**: 10**  Equal to or more than 5 |
| **Sample 4** | Enter an integer between 1 and 10, inclusive**: 1**  Less than 5 |

Write down the corrected code below:

|  |
| --- |
|  |