2.1 What does the code below print? NB! The objective is to understand why Python gives certain results.

Therefore, give arguments!

```
print(8 % 2)          # a) 0 . % yields the remainder of floor division.
print(8 % 1.5)        # b) 0.5 . % yields the remainder of floor division.
print(4.0 * 2.0 / 4)  # c) 2.0. left to right
print(4 * 2 / 4)      # d) 2.0. left to right
print(2 / 4 * 4)      # e) 2.0. left to right
print(2 // 4 * 4)     # f) 0. Because // is the division that rounds down to nearest integer
print(int(1.99))      # g) 1
print(3 * 1.1)        # h) 3.3000000000000003. 1.1 is actually stored as something like 1.1000000000000000888, so 3 * 1.1 becomes about 3.3000000000000003
print(5 < 6)          # i) True. It is return as a boolean
print(2 + 5 < 6)      # j)False. Because 2 + 5 = 7 is greater than 6
print(2 + (5 < 6))    # k) 3. In arithmetic operations, bool values False and True behave like the values 0 and 1. So 2 + true = 2 + 1 = 3.
```

NB! You can run the above code in the interactive Python environment as follows:

1. Start the interactive Python interpreter by typing py to the Windows search box.

2. Copy/paste the code to the interpreter window.

2.2 How can we perform accurate arithmetic computations in Python (without installing any extra libraries)?

Using Decimal for precise decimal values

The decimal module contains the Decimal class for manipulating arbitrary-size and arbitrary precision decimal values. The Decimal class provides high-precision arithmetic needed in monetary calculations etc.

2.3 When should we use the elif keyword?

```
If ( logical_expression 1)
        #run if logical_expression is correct
elif (logical_expression 2)
```

```
#run when logical_expression 1 is incorrect and logical_expression 2 is
correct
```

2.4 How can we write a very short test on a single line of code to determine if the value of a variable called z is one of the following five values: 1, 2, 6, 11, and 22?

```
if z in (1, 2, 6, 11, 22)
```

2.5 In Python, there is no assignment operator. Instead, assignment is a statement in Python. Please explain in your own words how does assignment in Python differ from assignment in JavaScript (or Java).

In Python, assignment is just a statement that can assign value to variable. But in JS or Java, assignment can be an expression. Expressions always evaluate to a value, which statements do not.

2.6 In Python, comparison operators have the same precedence. What are the values of the expressions below? Explain in detail why these two expressions evaluate to different values.

```
1 < 2 == True

True == 1 < 2
```

In the first, the **middle value** is 2, so the second comparison is 2 == True → False. 1 < 2 is true, so true + false = false

In the second, the **middle value** is 1, so the second comparison is True == 1 → True. 1 < 2 is true, so true + true = true