Frequently Asked Questions Python

1. Printing and Formatting:

2. Variables, Numbers, and Math:

3. if, elif, and else:

4. for Loops and while Loops:

5. Functions

6. Lists

Answers:
1.1 How do I print literal text?
Use the print function and make certain to put words inside quotes.
Example:

```
print('Hello World!')
```

1.2 How and when do I use the escape sequence (\) ?
Use the escape sequence, which is the `'\'` character to add the following to text:
`\n` - makes new line
`\t` - makes a tab (indent) in text
`\'` - makes a literal ' in text
`\\` - makes a literal \ in text

Example:

```
print('Hello\nWorld\tof Escape!')
```

Output:

```
Hello
World     of Escape!
```

1.3 How do I use formatters to format a string?
The syntax is as follows:

```
string.format(value1, value2, value3 …)
```

For each value argument given, the *`string`* must have placeholders `{ }`.

`{ }` is for strings
`{:d}` is for integer values
`{:f}` is for floating point numbers (real numbers)

Several additional formatting keys can be added in the place holder to control the format specifics.

Examples:

```
text = '{}'.format('hello')
print(text)
```

Output:

```
hello
```

```
text = '{} world of {}'.format('Hello', 'python')
print(text)
```

Output:

```
Hello world of python
```

```
text = 'xx{:10}xx'.format('stuff')
print(text)
```

Output:

```
xxstuff     xx
```

```
text = 'xx{:>10}xx'.format('stuff')
print(text)
```

Output:

```
xx     stuffxx
```

get it, right stuff?

```
text = 'xx{:0>10}xx'.format('right')  # right align fill zeros
print(text)
```

Output:

```
xx00000rightxx
```

```
text = 'xx{:^12}xx'.format('center')
print(text)
```

Output:

```
xx   center   xx
```

```
num = '{:d}'.format(42)
print(num)
```

Output:

```
42
```

```
num = '{:02d}'.format(4) # force 2 digits zero pad
print(num)
```

Output:

```
04
```

```
import math
```

```
floating_num = '{:f}'.format(math.pi)
print(floating_num)
```

Output:
```
3.141593
```

```
import math
floating_num = '{:.2f}'.format(math.pi)
print(floating_num)
```

Output:
```
3.14
```

1.4 How do I concatenate (add together) a string and an integer or float? I keep getting this error:
TypeError: can only concatenate str (not "int") to str
To combine a string and an int you must *cast* the int to a string using the `str(value)` function.

```
Example:
cost = 4.59
result = 'Your bill is' + str(cost)
```

1.5 How do I print without moving the cursor down to a new line?

The print function has an optional keyword argument called end. The end variable's default value is '\n' meaning make a new line. To stop this simply change this value.

Example:
```
print('hello', end = ' ')
print('world')
```

Output:
```
hello world
```

1.6 How do I print a mix of different types like string and int?

There are two ways you can do this.

Method 1: You can use commas in your print statement to separate the different types. Python will automatically convert each to string.

For example:
```
print('I have', 3, 'hamburgers left')
```

Output:
```
I have 3 hamburgers left.
```

As you can see, the print function separates each with a space. You can change that using the optional keyword argument `sep`. By default `sep = ' '`.

Method 2: You can use the `str(`*`value`*`)` function to convert non-string objects to strings.

Example:
```
print('I have ' + str(3) + ' hamburgers left')
```

Notice that when using concatenation like this, we need to include the spaces ourselves. This method is more work, but gives the coder greater control over the overall look of the output.

2.1 How do I assign a numeric value to a variable?
Make a variable name and be sure it is on the left side!
Example:
```
x = 4
print(x)
```

Output:
```
4
```

2.2 How do I increase a numeric value?
```
x = 4
x = x + 5
print(x)   # prints 9
```

You can also use the += assignment operator.
```
x += 5 # shortcut to increase x by 5
```

2.3 I think I am increasing my variable, but it does not change! Why?
You are likely making this error:
```
x = 4
```

```
x + 1    # this line does nothing
print(x) # prints 4 because x is still 4
```

You need to use
`x += 1`
or
`x = x + 1`

See question 2.2.

2.4 How do a I make an exponent like in x^2?
Use the ** operator.
Example
```
x = 4
y = x ** 2     # y gets value of 16
```

2.5 How do I take the square root of a number?
You can use the fact that the ½ power is square root.
`9 ** 0.5` will result in 3.0

Alternatively you can import from the math module.
```
from math import sqrt
x = 9
y = sqrt(x)
print(y)
```

2.6 How do I use trig functions like sin, cos, or tan?
These functions should be imported from the math module.

```
from math import sin, cos, tan, pi
print(sin(pi/2))
```

Note that python uses radians, not degrees. If you have an angle in degrees, convert it to radians before using the function.

```
from math import sin, radians
deg = 30
print(sin(radians(deg)))
```

3.1 How do I make python do something if a variable equals a certain value?

Use the `if` statement with the == operator. Note that two equal signs asks if a variable equals a value.

Examples

```
name = input('What is your name')
if name == 'Jed':
    print('That\'s my name too!')  # note the escape sequence
```

Remember that the input function always brings in text as a string. If you want it as a number you must use the `int(`*`string`*`)` or `float(`*`string`*`)` functions.

```
num = int(input('What is your favorite number?'))
if num == 7:
    print('That is my favorite number too!')
```

3.2 How to I make python do one thing if a variable equals a value and another thing if does not?

Use an `if – else` block of code.

Example:

```
grade = int(input('What score did you get on the quiz?'))
if grade >= 70:
    print('You passed!')
else:
    print('Oops - you failed this quiz.')
```

3.3 How do I use an if – elif – else to multiple cases?

Example:

```
age = int(input('How old are you?'))
if age < 42:
    print('You are younger than me')
elif age == 42:
    print('You are my age.')
else:
    print('You are older than me.')
```

4.1 How do I use a for loop to make Python repeat code a certain number of times?

The syntax to make a for loop to repeat something 5 times is as follows:

```
for i in range(5):
    print('hello')
```

Output:
```
hello
hello
hello
hello
hello
```

The variable `i` is used to count. It is traditional to use the letter `i` but you can use `k`, `x`, `counter` or any other variable name you want
To change the amount of times python repeats the code, simply change the number inside the ( ) of the range function.

Note that you can make the loop repeat more than one line of code. The for loop block does not end until you dedent back to the left.

This snippet of code prints python rocks three times then goodbye once.

```
for i in range(3):
    print('python', end = ' ')
    print('rocks')
print('goodbye')
```

Output:
```
python rocks
python rocks
python rocks
goodbye
```

4.2 How exactly does the range function work? How can I count by different amounts?

The range function full syntax is as follows:
`range(start, stop, step)`

start is the value you start counting at, stop is the value you will go up to BUT NOT INCLUDE and step is what value the counting will change by.

To make using range easier, python allows you the option of giving only a stop value, a start and stop, or all three. The default value for start is 0 and step is 1.

For example:
`range(5)`

This is a range from 0 up to but not including 5 counting by 1.

```
range(5, 10)
```
This is a range from 5 up to but not including 10 counting by 1.

```
range(10, 20, 2)
```
This is a range from 10 up to but not including 20 counting by 2.

Consider this example:

```
for k in range(10, 19, 3):
    print(k)
```

Output:
```
10
13
16
```

4.3 How do I use a while loop?

```
while condition:
    # code…
```

Code in the body of a while loop will be executed over and over again while the *condition* is true.

For example:
```
x = 10
while x > 0:
    print(x, end = '  ')
    x -= 1
```

Output:
```
10  9  8  7  6  5  4  3  2  1
```

4.4 How do I make a forever loop?

If you want code to repeat indefinitely, you can use "while True" loop.

```
while True:
    # repeat forever
```

4.5 How can I stop looping early or stop a forever loop?

Use the break command

For example consider the following code:

```
while True:
    command = input('what is your bidding')
    if command == 'STOP':
        break
```

This code will continually ask the user what is your bidding until they type STOP.

5.1 How do I write a function?

Functions are declared using the def keyword. def means 'define'. Parameters, if any are listed inside the parenthesis.

Examples:

```
def say_hello():
    print("Hello! I'm a function!")


def print_sum(a, b):
    print(a + b)
```

5.2 How do I make my function return a value?
Use the keyword `return`.

Example:

```
def get_perimeter(len, width):
    return 2 * len + 2 * width
```

5.3 How do I call a function from inside my module?
To call a function that you wrote from within the same module, simply say the name of the function along with any needed arguments.

Example:
```
def say_hello():
    print("Hello! I'm a function!")


def print_sum(a, b):
    print(a + b)

# The code below will call the functions written above
say_hello()
print_sum(10, 20)
```

Output:
```
Hello! I'm a function!
30
```

If the function you are calling returns a value, be sure to use a variable to *catch* what that function returns. For example, to call the function `get_perimeter(len, width)` which returns the perimeter of a rectangle with sides of `len` and `width`, we might code the following:

```
rect_len = 10
rect_width = 20
per = get_perimeter(rect_len, rect_width)
print(per)  # prints 60
```

5.4 How do I import a function from another module?
To use a function that is written in a different module you must first import the function. This can be done in three ways. Consider how we can import and the sqrt function from the math module.

Option 1: import only the module name
```
import math
x = math.sqrt(100)        # must include name of module 'dot' function
```

Option 2: import the function from the module
from math import sqrt
```
x = sqrt(100)            # only have to say sqrt()
```

Option 3: import all the function from math madule.

```
from math import *
```

This import will import all of the functions from the math module and allow you to directly call them such as

```
x = sqrt(100)
```

5.5 How do I call a function I imported from another module?

If you imported only the module name, you must call the function on the module name.

For example consider the following functions from the turtle module:

```
import turtle
turtle.forward(100)
turtle.left(90)
```

However if you import the function from the module you may call it directly. For example:

```
from turtle import forward, left
forward(100)
left(90)
```

You can also directly call function when you import the entire module using `from` *`module`* `import *`