PreLab 08

The While Loops Revision

There are two types of loops:

- Definite Loops ⇒ That loop for a definite number of times. These type of loops called **for** loop
- Indefinite Loops ⇒ That loop for an indefinite number of times, the code keeps looping as long as certain condition holds. These type of loops called **while** loop

Lets consider this example: “How long do I have to shower?” You have two possible replies:

1. 10 minutes
2. Until you are clean

When programming, the first answer would be portrayed as a for-loop because we know exactly how long the shower will continue:

```python
for minutes in range(0,10):
    shower
```

The second answer would be portrayed as a while-loop because the length of the shower is undetermined; we instead focus on the condition of cleanliness:

```python
while you are not clean:
    shower
```

The general form of a while loop:

```python
while <condition>:
    <body>
```

The condition is a Boolean expression. The while will keep looping executing the body as long as the condition is True

Lets see another example of while loop:

```python
count = 0
while count < 5:
    print(“The count is”, count)
    count = count + 1
print(“While loop ended”)
```

The output of this above code is:

```
The count is 0
The count is 1
```
The count is 2
The count is 3
The count is 4
While loop ended

**Integer Division**

Generally when we say 2 divides 10, or 10 is divisible by 2, this means that the result of the division operation 10/2 is a whole number (integer) with no remainder. On the other hand, we say that 10 is not divisible by 4 because 10/4 = 2.15 is not an integer.

Integer division operation have two results which should be INTEGERS; the quotient and the remainder. So that the general form of integer division is:

\[ \text{for } x/y: \text{ quotient} = \text{int}(x/y), \text{ remainder} = x - y \times \text{quotient} . \]

- E.g. The integer division: 10/4 has quotient = int(2.15)=2, and remainder = 10-4*2=2. Having remainder equals greater than zero means that 10 is not divisible by 4
- E.g. The integer division: 10/3 has quotient = int(3.3333)=3, and remainder = 10-3*3=1. Having remainder equals greater than zero means that 10 is not divisible by 3
- E.g. The integer division: 10/2 has quotient = int(5)=5, and remainder = 10-2*5=0. Having remainder equals zero means that 10 is divisible by 2

**Mod operation (%):**

The mod operations between x,y is \( x\%y \) which calculates the remainder of the integer division of \( x/y \) as we calculated in the previous examples.

- 10\%4=2
- 10\%3=1
- 10\%2=0

We can conclude that if \( x\%y \) equals to zero, then \( x \) is divisible by \( y \), otherwise \( x \) is not divisible by \( y \).

**Example**

You are required to write a python function that takes two parameters \( n,k \). The function should return the number times we can divide \( n \) by \( k \) with no remainder.

```python
def divisionCount(n, k):
    count = 0
    while (n%k == 0):
        count = count +1
        n = int(n/k)
    return count

print(divisionCount(40, 2)) #prints 3
```

http://courses.cs.purdue.edu/
Validating user input

Sometimes you want to make sure that values given by the user are in fact correct. For example if you are writing a program to calculate the square root for a number supplied by the user and you want to make sure that the user supply a non negative number. One way to do this is just have a simple if/else statement so that when the user supply a negative number your code will not calculate the square root and the program will terminate.

Another way, is to keep asking the user to enter a non negative number if the input number was negative, then your program should not terminate, instead, ask the user again to enter a non negative number. So that you code will keep asking the user for the correct input as long as the user supplies an invalid input.

```python
import math

x = -1
while x < 0:
    x = int(input('Enter a number: '))
print(math.sqrt(x))
```