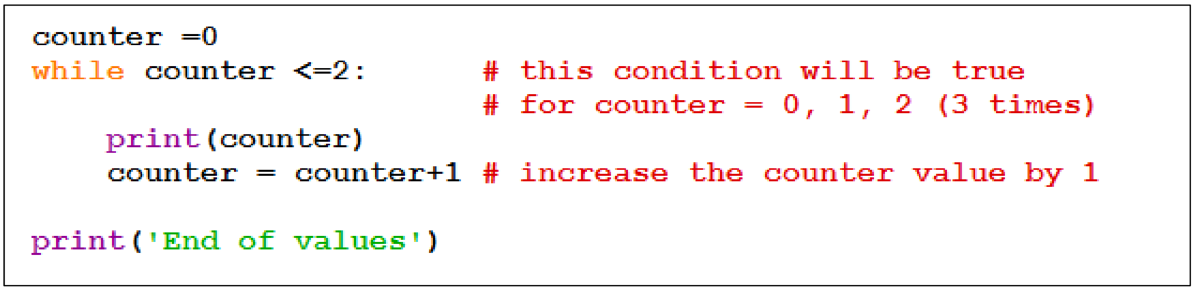
Lab 6: Iterative control structures

An iterative statement provides repeated execution of a set of instructions. One specific iterative statement is the **while** loop. When the iteration starts and when it stops is controlled by the condition of the while loop.

The general format of **while** statement:

```
while <condition>:          # must always end with a colon (:)
        <statement>         # can have one or many statements
        …
```

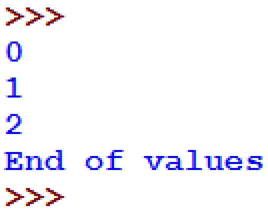
Here is an example that prints out the numbers 0, 1, and 2 to the screen:



```
counter =0
while counter <=2:       # this condition will be true
                         # for counter = 0, 1, 2 (3 times)
    print(counter)
    counter = counter+1  # increase the counter value by 1

print('End of values')
```

Sample Output:



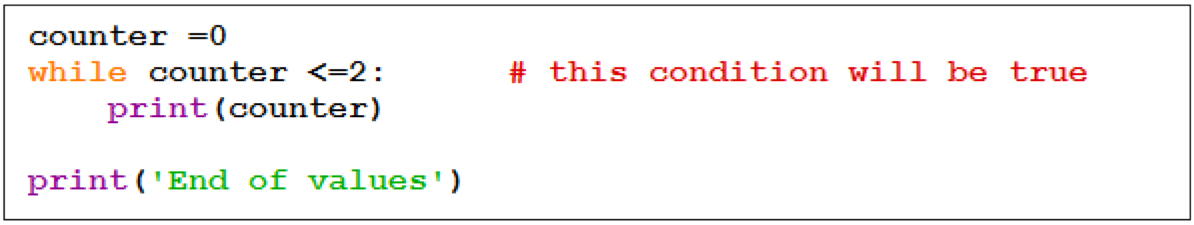
```
>>>
0
1
2
End of values
>>>
```

Working of the loop (and counter):

Loop variable	condition	Statements to repeat execution	
counter	counter<=2	print(counter)	counter=counter+1
0	True	0	counter = 0 + 1
1	True	1	counter = 1 + 1
2	True	2	counter = 2 + 1
3	False	-	-

Infinite loop

In the following loop, loop counter variable is not updated during the iterations; therefore, the condition stays true forever.

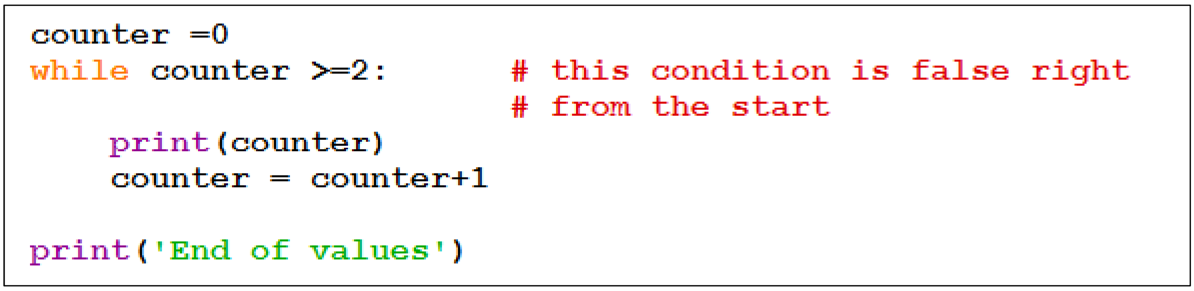


```
counter =0
while counter <=2:       # this condition will be true
    print(counter)

print('End of values')
```

Useless loop

In the following loop: loop condition is false from start - loop body is never executed



```
counter =0
while counter >=2:       # this condition is false right
                         # from the start
    print(counter)
    counter = counter+1

print('End of values')
```

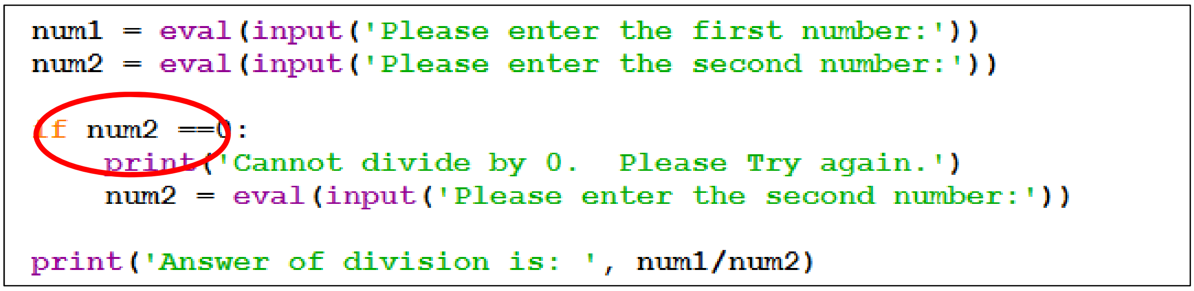
Therefore:

- the loop condition should normally be true to start off the loop
- there must be some code inside the loop that updates the condition and makes the condition false to stop the loop eventually

User input checking

The while statement is well suited for **input error checking**. Consider the following:

Exercise 1: Read two numbers from user and divide the first number by the second. If the second number is zero the user should be prompted for re-entry until satisfaction.



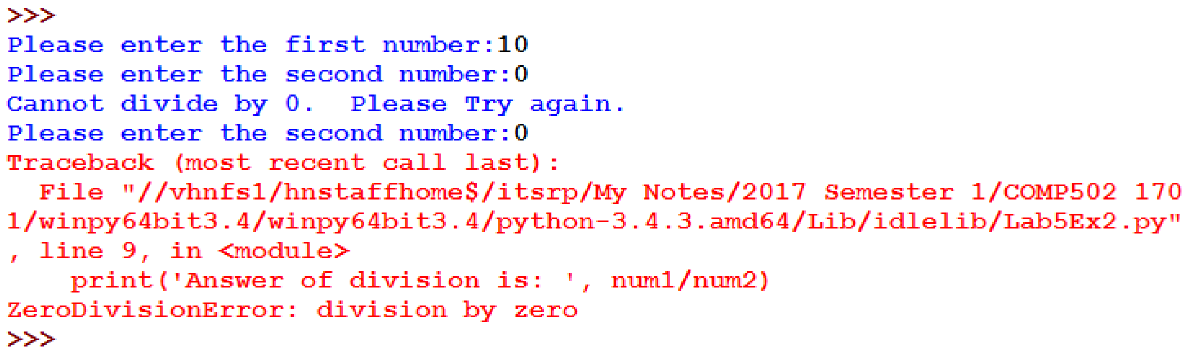
```
num1 = eval(input('Please enter the first number:'))
num2 = eval(input('Please enter the second number:'))

if num2 ==0:
   print('Cannot divide by 0.  Please Try again.')
   num2 = eval(input('Please enter the second number:'))

print('Answer of division is: ', num1/num2)
```

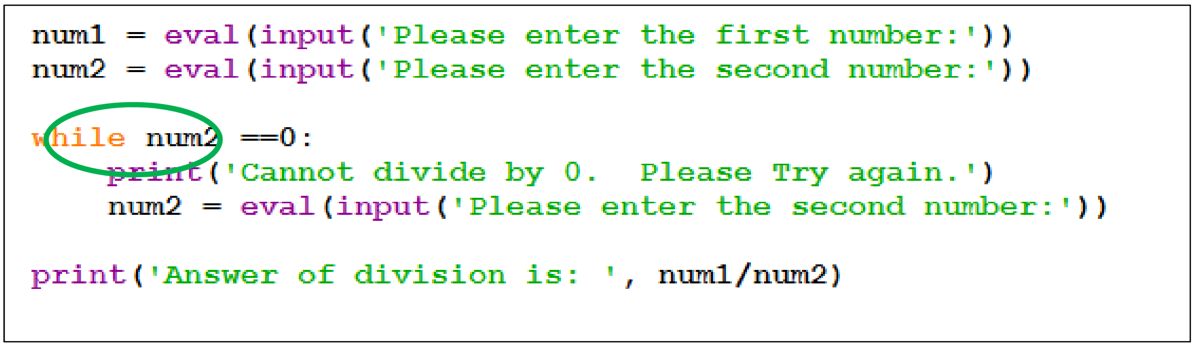
The above solution uses **if statement** for user input checking - which is insufficient (as the user can enter an incorrect value the second time).

Sample Output:



```
>>>
Please enter the first number:10
Please enter the second number:0
Cannot divide by 0.  Please Try again.
Please enter the second number:0
Traceback (most recent call last):
  File "//vhnfs1/hnstaffhome$/itsrp/My Notes/2017 Semester 1/COMP502 170
1/winpy64bit3.4/winpy64bit3.4/python-3.4.3.amd64/Lib/idlelib/Lab5Ex2.py"
, line 9, in <module>
    print('Answer of division is: ', num1/num2)
ZeroDivisionError: division by zero
>>>
```

The following solution uses while loop for user input checking - which guarantees the right input data:

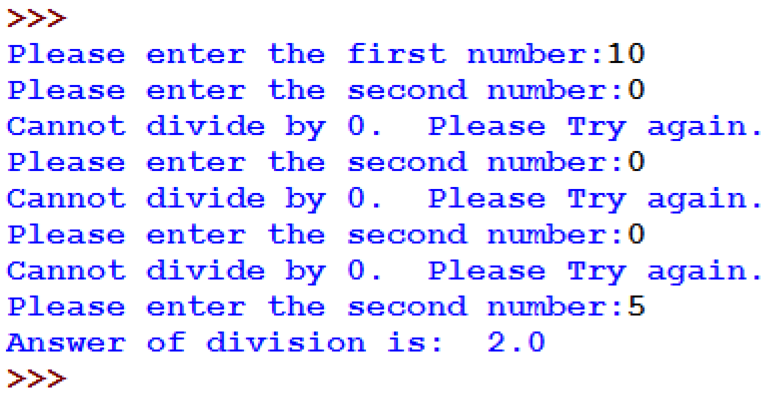


```
num1 = eval(input('Please enter the first number:'))
num2 = eval(input('Please enter the second number:'))

while num2 ==0:
    print('Cannot divide by 0.  Please Try again.')
    num2 = eval(input('Please enter the second number:'))

print('Answer of division is: ', num1/num2)
```

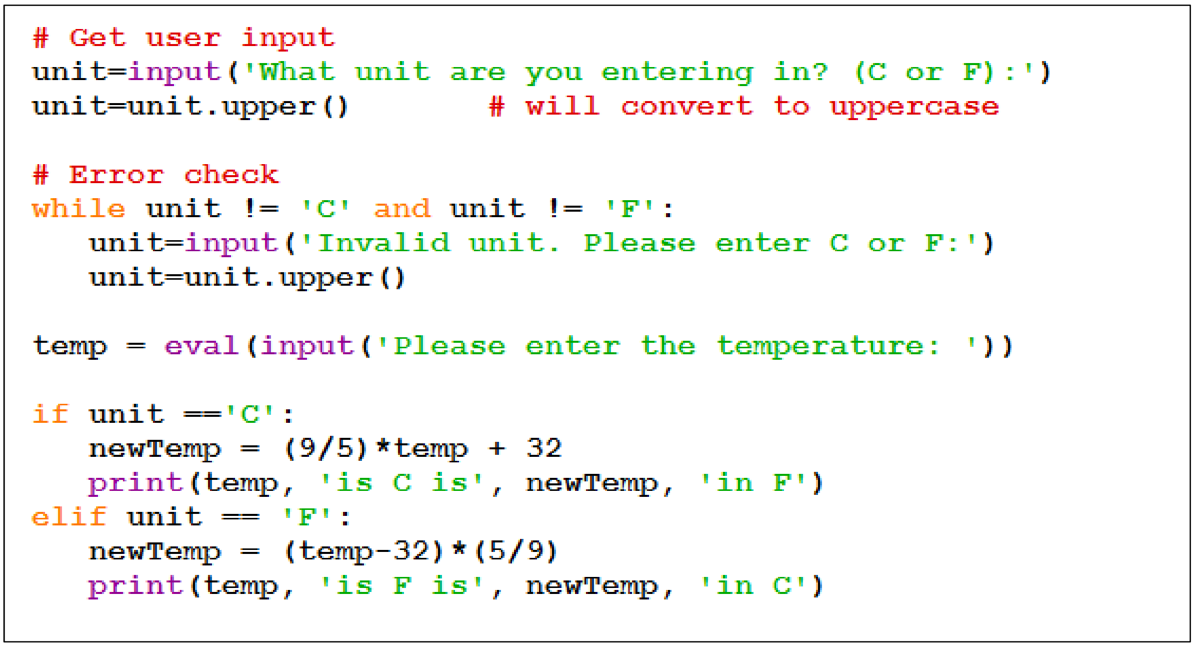
Sample output:



```
>>>
Please enter the first number:10
Please enter the second number:0
Cannot divide by 0.  Please Try again.
Please enter the second number:0
Cannot divide by 0.  Please Try again.
Please enter the second number:0
Cannot divide by 0.  Please Try again.
Please enter the second number:5
Answer of division is:  2.0
>>>
```

Task 1:

Save the temperature conversion program as **Lab6Task1.py**. Modify it so that it uses a while loop for user choice (C or F) validation.



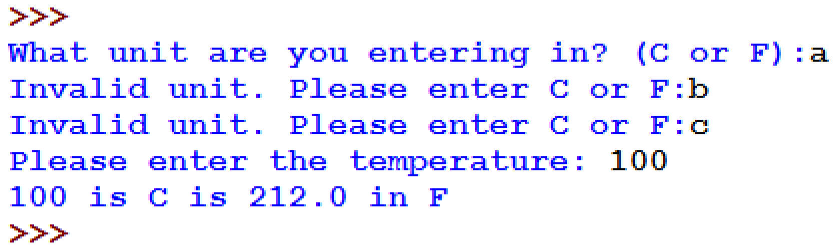
```
# Get user input
unit=input('What unit are you entering in? (C or F):')
unit=unit.upper()        # will convert to uppercase

# Error check
while unit != 'C' and unit != 'F':
   unit=input('Invalid unit. Please enter C or F:')
   unit=unit.upper()

temp = eval(input('Please enter the temperature: '))

if unit =='C':
   newTemp = (9/5)*temp + 32
   print(temp, 'is C is', newTemp, 'in F')
elif unit == 'F':
   newTemp = (temp-32)*(5/9)
   print(temp, 'is F is', newTemp, 'in C')
```

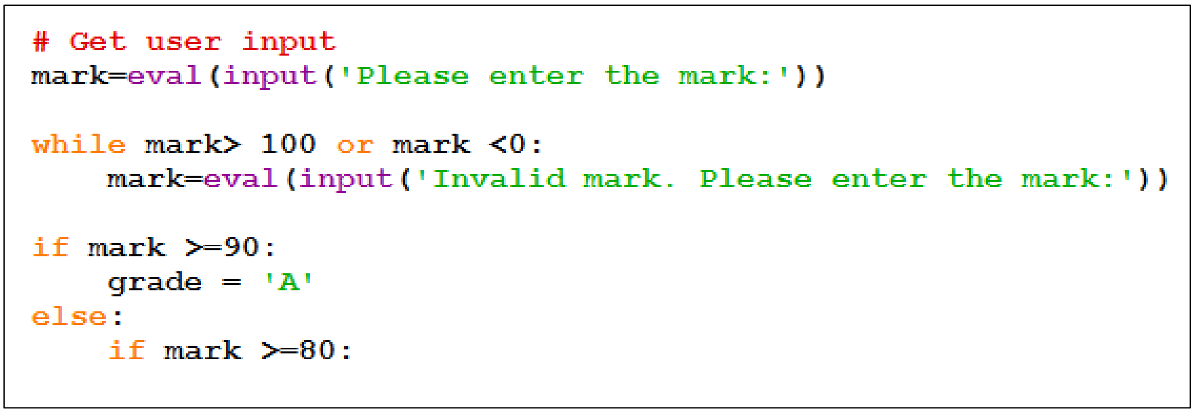
Sample output



```
>>>
What unit are you entering in? (C or F):a
Invalid unit. Please enter C or F:b
Invalid unit. Please enter C or F:c
Please enter the temperature: 100
100 is C is 212.0 in F
>>>
```

Task 2:

Save the grading program as **Lab6Task2.py**. Modify it so that it uses a while loop for marks entered. Only a mark between 0 and 100 should be accepted.



```
# Get user input
mark=eval(input('Please enter the mark:'))

while mark> 100 or mark <0:
    mark=eval(input('Invalid mark. Please enter the mark:'))

if mark >=90:
    grade = 'A'
else:
    if mark >=80:
```