**3. Repetition structures**

**3.1 While loop- condition controlled loop.**

While a condition is true, do some task.

True

while condition:

statement

statement

etc

When the while loop executes, the condition is tested. If condition is true, the statements that appear in the block following the while clause are executed. The loop then starts over.

**Question**

Does the while loop test the condition before or after performing an iteration.?

**Example**

# An example of while loop

# initialise the value of i

i=0

while (i < 6):

    i=i+1

    print(i)



i is initialised outside the loop and is incremented inside the loop.

**3.1.1 break and continue statements with while**

With the break statement, we stop the loop even when the condition i< is true.

# An example of while loop

# initialise the value of i

i=0

while (i < 6):

    i=i+1

    if i==3 :

        break

    print(i)



With continue statement, the iteration is skipped. Number 3 is skipped.

# An example of while loop

# initialise the value of i

i=0

while (i < 6):

    i=i+1

    if i==3 :

        continue

    print(i)



**3.2 For loop - A count controlled loop.**

A count controlled loop iterates a specific number of times. In Python you can use fro statement to write a count-controlled loop.

for variable in [value1,value2,value3. etc]:

Statement

Statement

Statement

An example of a simple for loop for listing a series of numbers.

print ('This programs displays all numbers from 1 to 6')

for num in [1,2,3,4,5,6]:

     print(num)



The variable *num* is assigned values from 1 to 6 in the list.

This program uses a series of strings with for statement

# This program a simple for loop that uses a series of strings

for name in ['Greg','Tom', 'James', 'William']:

    print(name)



**3.2.1 Using the range function with the for loop**

The range function creates a type of object known as an iterable. An iterable is an object that is similar to a list. This has a sequence of values to be iterated over a loop

for number in range (5)

print(num)

This is similar to

for number in [0,1,2,3,4]:

print(num)

The numbers start with 0 and goes upto 5 but does not include 5.

# range function used with a for loop

# Print a message 3 times

for i in range (3):

    print("Hello world")



A third value in range statement can be used as a step

range(1,10,2)

refers to numbers 1,3,5,7,9 with a step of 2.

**Exercise 3.2.2**

Make a program to display a table with numbers from 1 to 8 with the squares of the values.

Output should look like this

Number Square

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

1 1

2 4

3 9

.

.

.

8 64

(Hint: use a tab key to separate outputs (‘\t’)



**3.2.2 Practice with range statement**

* Use the range function to modify the program

The list is [0,1,2,3,4,5,6,7,8,9]

for y in [0,1,2,3,4,5,6,7,8,9]:

print(“I love programming”)

* What will this code display?

for x in range (5):

print(x)

***0  
1  
2  
3  
4***

* What will the code display

for count in range (1,5,2):

print(count)

***1  
3***

* What will be the output of the following code?

for x in range ( 10,5,-1):

print(x)

***10  
9  
8  
7  
6***

**3.2.3 Accumulator for a loop.**

If you want to have a total of a series of numbers, you need a loop that reads every number in the list and a variable that accumulates the total of the numbers.

This program sums a series of numbers entered by the user and prints the sum

# This program calculates the sum of a series of number entered.

#decide how many numbers you want

max\_nos = 4

# initialise the accumulator variable

sum=0.0

# This program calculates the sum of the numbers you enter

# get the numbers.

for count in range(max\_nos):

    num=int(input ("Enter the number: "))

    sum+=num

#Display the total

print("The sum of the numbers is "+str(sum))



**3.3 Sentinel**

When there is a long list of values there are two options.

* Ask the user at the end of each loop, if there is another value to process
* Ask the user at the beginning of the program how many items are in the sequence.

When processing a long sequence of value with a loop, it is a better technique to use a sentinel. Sentinel is a special value that you will add to mark the end of the sequence of value. Eg. If we are entering weight of a patient the sentinel can be zero since a patient cannot have a value of 0.

Here is an example of sentinel . Assume all the values are above 0.

# This program calculates the sum of a series of number entered.

# initialise the accumulator variable

sum=0.0

# This program calculates the sum of the numbers you enter

# get the numbers.

num=int(input("Enter a number or 0 to finish: "))

while num != 0:

    sum=sum+num

    num=int(input ("Enter the number: or 0 to finish: "))

6#Display the total

print("The sum of the numbers is "+str(sum))



The sentinel here is ‘0’.

Example of sentinel is on this link.

<https://www.youtube.com/watch?v=L3QZNaq9PCI>

**3.4 Input validation loops.**

Input validation is the process of inspecting data that has been input to program to make sure that the input data is valid before it is used in a computation.

# Get a test score.

score = int(input('Enter a test score: ' ))

while score < 0:

      print('You cannot have negative scores')

      score=int (input ('Enter the correct score:'))



**3.5 Nested loops**

Nested loop is a loop inside a loop.

description = ["tasty", "big", "nutritious"]

fruits = ["apple", "banana", "orange"]

for x in description:

  for y in fruits:

    print(x, y)



This is another example of a pattern with nested for loop.

This creates a triangular pattern

# This program displays a tringle pattern

base\_size=10

for count in range (base\_size):

    for y in range(count+1):

        print('x', end='')

    print('')



base\_size takes value of 0 in the outer loop . The inner loop, It prints ‘x’ in the first iteration in the outer loop and then 2 ‘x’s in the second iteration of the outer loop till it prints 10 ‘x’s in the final iteration of the outer loop.

base\_size =0

count = 1 prints one ‘x’

base size=1

count = 2 prints 2 ‘x’s

till base size = 9

count = 10 prints 10 ‘x’

**3.6. Programming exercises**

1. Write a loop that lets the user enter a number. The number should be multiplied by 10 and result will be assigned to a variable name **prod**. The loop should iterate as long as the product is less than 100



2. Write a loop that asks the user to enter a number. The loop will iterate 10 times and keep running total of the numbers entered.



3. Write a loop that calculates the total of the following series of numbers.

1/15 + 2/14 + 3/13 + 4/12 + - - - 15/1



**Hint: define two variables x and y and calculate the ratio x/y.**

4. Write a set of nested loops that display 10 rows of ‘\*’ characters. There should be 15 ‘\*’ characters in each row



5.Write a code that prompts the user to enter a number in the range of 1 through 100 and validates the input.

6. A pathology staff collects samples each day for 5 days. Write a program that keeps a running total of the number of samples collected during 5 days. The loop will ask for the number of samples collected for each day and when the loop is completed, the program will display the total number of samples collected .

7. **Sum of numbers**

Write program with a while loop that ask the user to enter a series of positive numbers . The user should enter a negative number to signal the end of the series. The program should display the sum of the numbers after it quits the loop.

8. **Calculating the factorial of a number**

In Mathematics the notation n! represents the factorial of a non-negative integer n.

The factorial of n is the prodicut of all non negative integers from 1 to n

Eg 4! = 1x2x3x4= 24

Write a program that lets te user enter a nonnegative integer and then uses the loop to calculate the factorial of that number. Print the factorial.

**3.7 Challenge exercises**

**a)Roulette wheel colours**

On a roulette wheel, the pockets are numbered from 0 to 36. The colours of the pockets are as follows:

Pocket 0 is green.

For pockets 1 through 10, the odd numbered pockets are blue and the even numbered pockets are black

For pockets 11 through 18, the odd numbered pockets are black and the even numbered pockets are blue

For pockets 19 through 28, the odd numbered pockets are blue and the even numbered pockets are black

For pockets 29 through 36, the odd numbered pockets are black and the even numbered pockets are blue.

Write a program that asks the user to enter a pocket number and displays where a pocket is green, blue or black

* Modify this program such that the program loops through using a while statement until the user enters a negative number. It also prints how many numbers were entered.

b) Design a program that simulates the toss of a coin. It generates 1 and 2 . Designate 1 as head and 2 as tail. Import *random* function and use *randint* option. Toss the coin 10 times. Count the number of heads and tails and print them.

* Modify the program and use a while loop with an option to let the user quit the loop when needed. Count the number of tosses and the number of times heads and tails appear.

Reference: **Reference: Starting out with Python : Third edition *Tony Gaddis*: Pearson**

**(Chapter 4)**