Experiment # 1

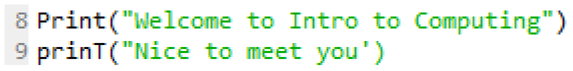
Implementation of PRINT statement in PYTHON

Things to be covered in this experiment

- Introduction to Jupyter Notebook software
- PRINT statement
- Use of **\n** and **end=””**

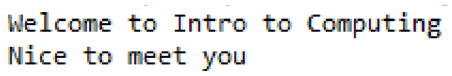
1. Eliminate all the syntax errors in the following code to get the output in the console window as shown below

Code



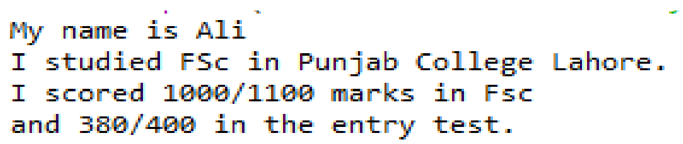
```
Print("Welcome to Intro to Computing")
prinT("Nice to meet you')
```

Output



```
Welcome to Intro to Computing
Nice to meet you
```

2. Write a program that prints the message as shown below. First use multiple print statements and then use a single print statement to get the output.

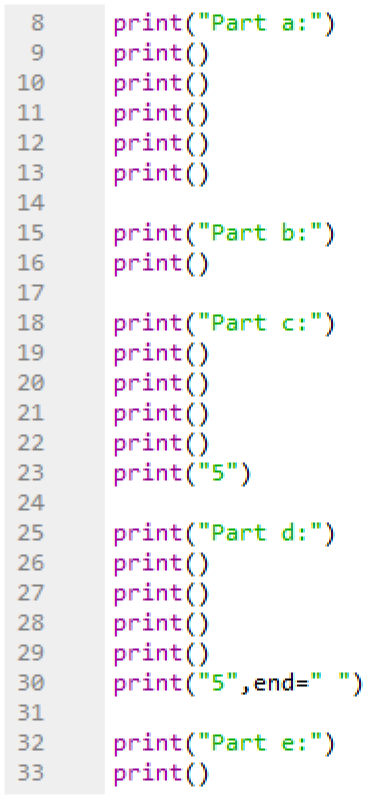


```
My name is Ali
I studied FSc in Punjab College Lahore.
I scored 1000/1100 marks in Fsc
and 380/400 in the entry test.
```

3. Complete the following code by following the steps below. The output obtained should look like the one given below the code.

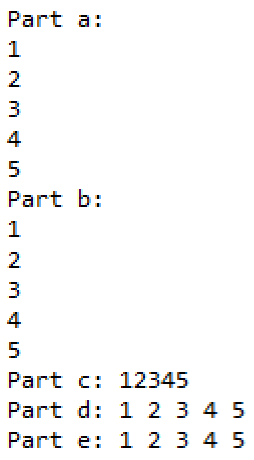
a. Use multiple print statements to print integers 1 to 5 vertically at the output window.
b. Use a single print statement and a single string to print integers 1 to 5 vertically at the output window.
c. Use multiple print statements to print integers 1 to 5 horizontally, without space, at the output window.
d. Use multiple print statements to print integers 1 to 5 horizontally, with space, at the output window.
e. Use a single print statement and a single string to print integers 1 to 5 horizontally at the output window.

The code to be completed



```python
print("Part a:")
print()
print()
print()
print()
print()

print("Part b:")
print()

print("Part c:")
print()
print()
print()
print()
print("5")

print("Part d:")
print()
print()
print()
print()
print("5",end=" ")

print("Part e:")
print()
```

OUTPUT



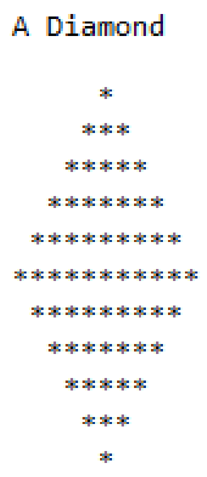
```
Part a:
1
2
3
4
5
Part b:
1
2
3
4
5
Part c: 12345
Part d: 1 2 3 4 5
Part e: 1 2 3 4 5
```

4. Write a program to print the diamond as shown below.

a. Use multiple print statements to complete the task.

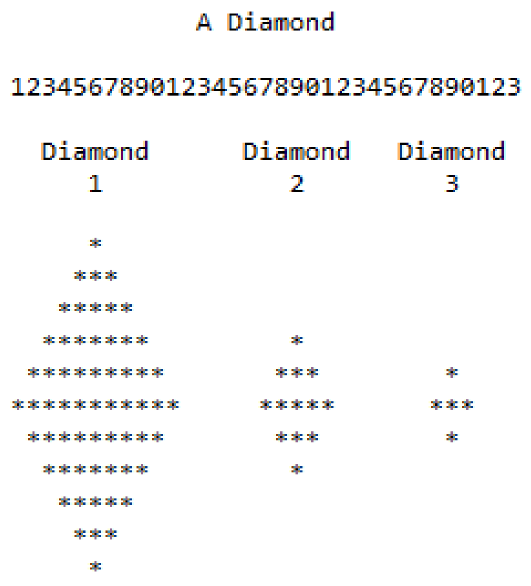
b. Use one print statement to complete the task.

Pay attention to the number of rows and the number of asterisks (*) in each row.



```
A Diamond

     *
    ***
   *****
  *******
 *********
***********
 *********
  *******
   *****
    ***
     *
```

5. Write a program to print three diamonds i.e. diamond 1, 2 and 3 as shown below. Use multiple print statements to complete the task. Pay special attention to the alignment and position of each text and diamond at the output. The integers “1234….0123” will help you to align and position the texts and diamonds properly. Also pay attention to the number of rows and the number of asterisks (*) in each row of each diamond.



```
            A Diamond

123456789012345678901234567890123

  Diamond      Diamond   Diamond
     1            2         3

     *
    ***
   *****
  *******         *
 *********       ***       *
***********     *****     ***
 *********       ***       *
  *******         *
   *****
    ***
     *
```

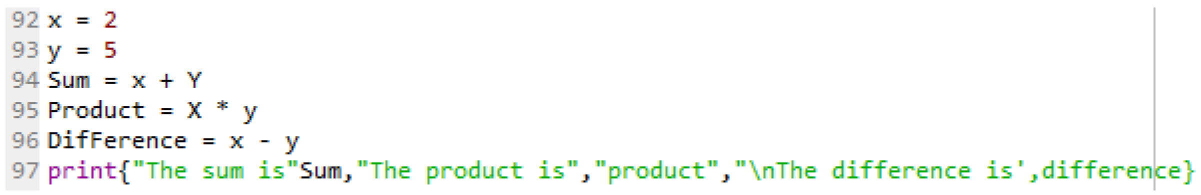
Experiment # 2

Implementation of Arithmetic operations and the INPUT function in Python

Topic to be covered in this Lab Manual

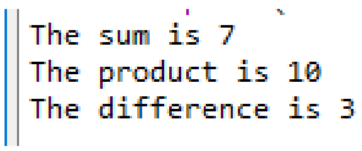
- Comments
- Docstring
- Arithmetic operators
- Variables
- Triple quotation marks in a string
- INPUT statement

1. Eliminate all the syntax and logical errors in the following code to get the output in the console window as shown below. Note: 92, 93, …, 97 represent line numbers and are not part of the code. Do not change the values of x and y. Output should be displayed using one PRINT statement only.



```
92 x = 2
93 y = 5
94 Sum = x + Y
95 Product = X * y
96 Difference = x - y
97 print{"The sum is"Sum,"The product is","product","\nThe difference is',difference}
```

The output should look exactly like the one shown below



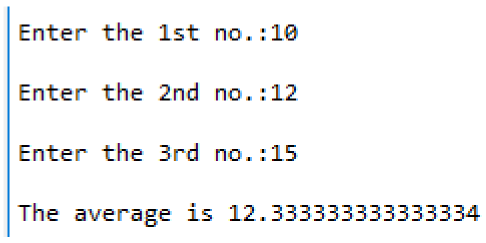
```
The sum is 7
The product is 10
The difference is 3
```

2. Implement the following recipe(algorithm) to find the average of three numbers entered by the user. The three numbers could also be floating point numbers. The script should begin with a description of what it does. Use docstring for that purpose.
   a. Prompt the user to enter the 1st number and store it in variable x.
   b. Prompt the user to enter the 2nd number and store it in variable y.
   c. Prompt the user to enter the 3rd number and store it in variable z.
   d. Calculate the average of three numbers.
   e. Display the average at the console window.

Insert the following comments for each step a, b, c, d and e.

Step	Comments to be written
a	The first number entered by the user
b	The second number entered by the user
c	The third number entered by the user
d	Average is calculated here
e	Average is displayed by this statement

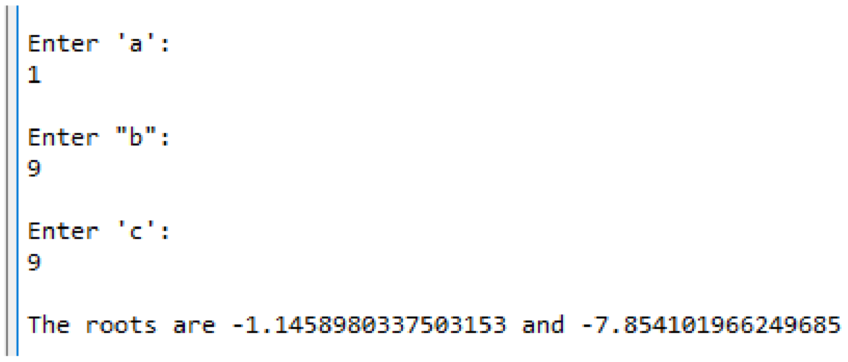
The output should look exactly like the one shown below



```
Enter the 1st no.:10

Enter the 2nd no.:12

Enter the 3rd no.:15

The average is 12.333333333333334
```

3. Write a program that calculates the roots of a quadratic equation $ax^2 + bx + c$. Prompt the user to enter the integer-type coefficients a, b and c. Then calculate the roots and display them at the output. The formula to calculate the roots is

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

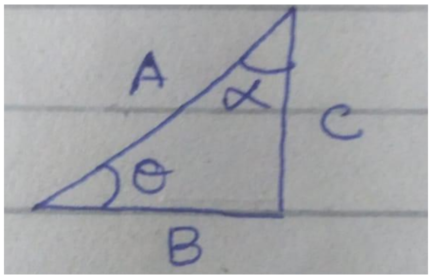


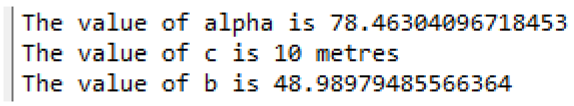
```
Enter 'a':
1

Enter "b":
9

Enter 'c':
9

The roots are -1.1458980337503153 and -7.854101966249685
```

Note:

a. The user enters the input at a new line.
b. Each variable is enclosed by quotation marks.
c. The program should work correctly for any number entered by the user.

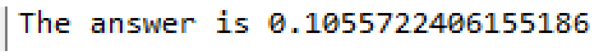
4. A right angle triangle having A = 50 metres and **C** = 10 metres is shown below. Write a program to find **α** and **B**. The program should work correctly for any **α** and **B**. The output should be as shown below.



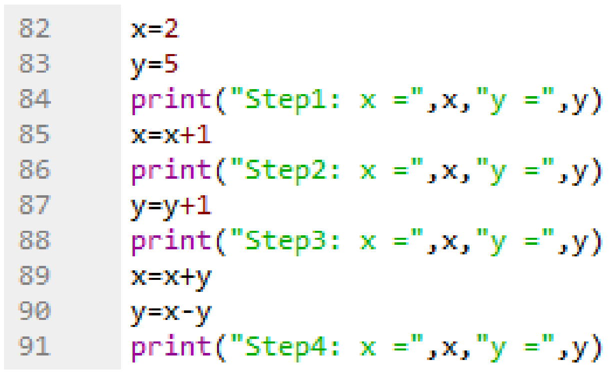

```
The value of alpha is 78.46304096718453
The value of c is 10 metres
The value of b is 48.98979485566364
```

5. Write a program to evaluate the following expression when a=10, b=5, c=15, d=20 and ***θ*** = 45°. The program should work correctly for any values of the variables. The output should be as shown below.

$$\frac{\sqrt{\left(\frac{a}{b}\times \sin\theta\right)^{\left(\frac{a}{b^2}\right)}+\left(\frac{b}{a}\times \cos\theta\right)^{\left(\frac{1}{a+b}\right)}}}{\left(c\times \tan^{-1}\theta\right)\div\left(\frac{d}{c}\right)}$$


```
The answer is 0.1055722406155186
```

6. Determine the output of the following program without using the computer.

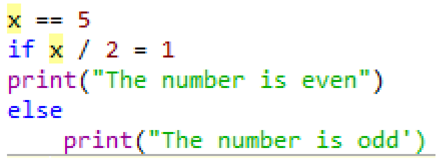

```python
x=2
y=5
print("Step1: x =",x,"y =",y)
x=x+1
print("Step2: x =",x,"y =",y)
y=y+1
print("Step3: x =",x,"y =",y)
x=x+y
y=x-y
print("Step4: x =",x,"y =",y)
```

Experiment # 3
Implementation of conditional statement in Python

Topics to be covered in this Lab Manual

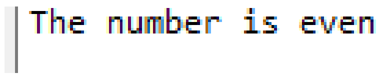
- If, If-Else and If-elif-else statements
- Logical and Relational Operators

1. Eliminate all the syntax and logical errors in the following code that displays “The number is even” or “The number is odd” if x is even or odd respectively.



```
x == 5
if x / 2 = 1
print("The number is even")
else
    print("The number is odd')
```

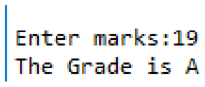
Example Output



```
The number is even
```

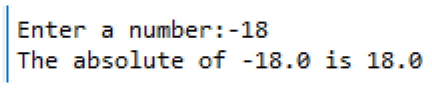
2. Implement the following recipe(algorithm) to find the grade of a student. Test your program to determine whether it works well for x ≥ 0.
   a. Prompt the user to enter the marks and store it in variable x. The marks can be a floating point number.
   b. If x is greater than 15 then display “The Grade is A”
   c. If x is less than equal to 15 and greater than 10 then display “The Grade is B”.
   d. If x is less than equal to 10 and greater than 5 then display “The Grade is C”.
   e. If x is less than equal to 5 and greater than equal to 0 then display “The Grade is F”
   f. If x does not satisfy any of the above mentioned conditions then print “Invalid Input”

The output should look exactly like the one shown below



```
Enter marks:19
The Grade is A
```

3. Write a program that takes any number from the user and displays its absolute value as shown below. The program should work correctly for any number entered by the user. Use IF statement only.

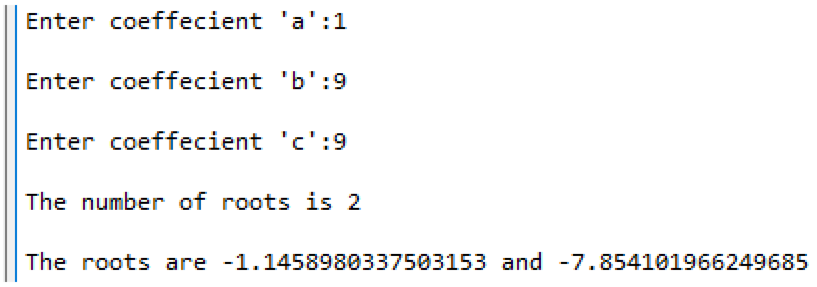


```
Enter a number:-18
The absolute of -18.0 is 18.0
```

4. Write a program that takes the coefficients *a, b and c* of a quadratic equation $ax^2 + bx + c$ as inputs from the user and then determines the number of roots and the values of the roots. The number of roots and their values should be displayed as shown below. The program should work correctly for any value of the coefficients. Test your program to determine whether it works well for all the possible values of the coefficients. Note:

Condition	Number of Roots
$(b^2 - 4ac) > 0$	2
$(b^2 - 4ac) < 0$	2
$(b^2 - 4ac) = 0$	1

Output

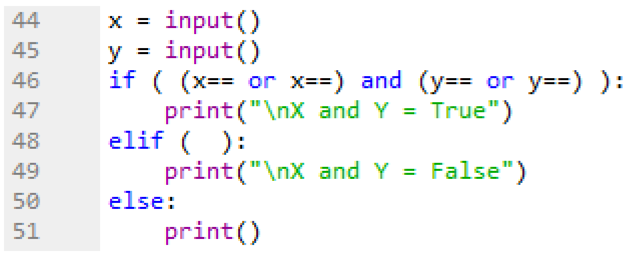


```
Enter coeffecient 'a':1

Enter coeffecient 'b':9

Enter coeffecient 'c':9

The number of roots is 2

The roots are -1.1458980337503153 and -7.854101966249685
```

5. Complete the following code to implement the truth table shown below. The output obtained should look like the one given below the code. Note:

g. The user can only enter T or t to represent true. The user can only enter F or f to represent false. If the user enters anything other than T, t, F or f then the program should display the message 'invalid input' after accepting both the inputs x and y. Note: Use only one IF, ELIF and ELSE statement.

The code to be completed

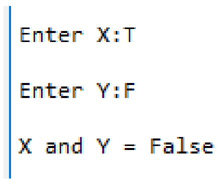


```
x = input()
y = input()
if ( (x== or x==) and (y== or y==) ):
    print("\nX and Y = True")
elif (  ):
    print("\nX and Y = False")
else:
    print()
```

X	Y	X and Y
True	True	True
True	False	False
False	True	False
False	False	False

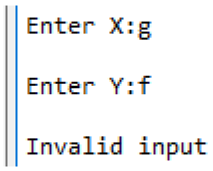
OUTPUTS

Output 1



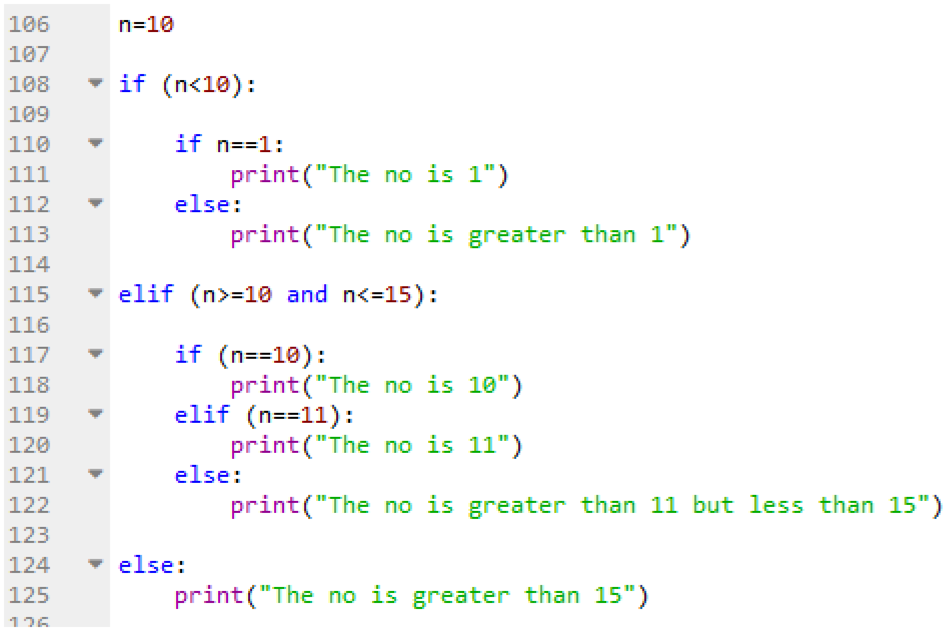
```
Enter X:T

Enter Y:F

X and Y = False
```

Output 2



```
Enter X:g

Enter Y:f

Invalid input
```

6. Determine the output of the following program without using a computer.



```
n=10

if (n<10):

    if n==1:
        print("The no is 1")
    else:
        print("The no is greater than 1")

elif (n>=10 and n<=15):

    if (n==10):
        print("The no is 10")
    elif (n==11):
        print("The no is 11")
    else:
        print("The no is greater than 11 but less than 15")

else:
    print("The no is greater than 15")
```

Experiment # 4
Implementation of Loops in Python

Topics to be covered in this Lab Manual

- WHILE loop
- FOR loop : Iteration over strings and over a sequence of numbers
- String indexing

1. Eliminate all the syntax and logical errors in the following code that asks the user for the number of times "Hello World" is to be displayed and then displays it.

```
z = input("Enter the number of times you want to print:")
y=1
While (y<=5)
    print("Hello World")
    y=y+2
```

2. Steps a to c below are part of an algorithm to find the absolute of a number three times. Do you think the program will work correctly or the order of the steps should be re-arranged to correctly run the program? If so, re-arrange the three steps and repeat them three times using a for-loop.

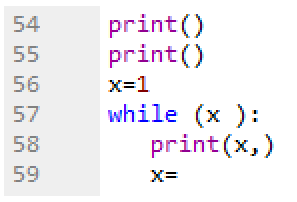
   Repeat the following steps 3 times using a for-loop:

   a. Display the message "The absolute is ___" where ___ is the value stored in x.
   b. Prompt the user to enter the number, the absolute of which is to be found and store it in variable x.
   c. If the number is negative, then multiply it by -1 and store it in x.

The output should look exactly like the one shown below

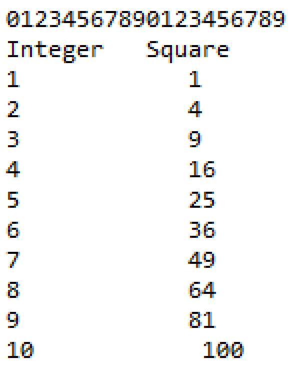
```
Enter a number:-3.5
The absolute is 3.5

Enter a number:2
The absolute is 2.0

Enter a number:0
The absolute is 0.0
```

3. Complete the following code to display the squares of integers from 1 to 10. Look at the output. Understand it thoroughly and then complete line 54, 55, 57, 58 and 59. Do not change anything else.



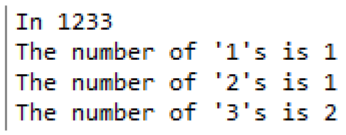
```
54  print()
55  print()
56  x=1
57  while (x ):
58      print(x,)
59      x=
```

Output



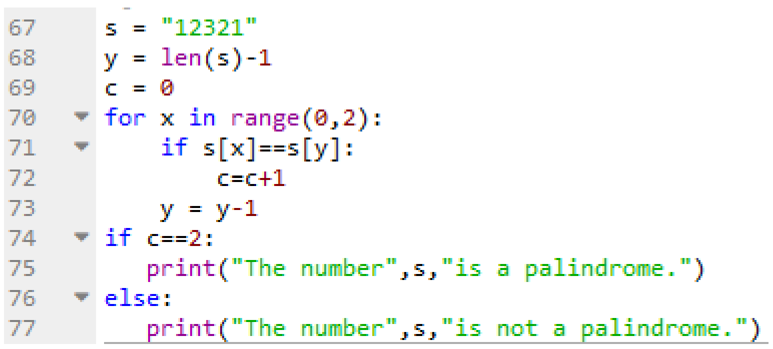
```
01234567890123456789
Integer   Square
1            1
2            4
3            9
4            16
5            25
6            36
7            49
8            64
9            81
10            100
```

4. Write a program to determine the number of 1s, 2s and 3s in a number. Store the number in a variable as a string and solve this task. The program should work correctly for any number containing only the digits 1, 2 and 3. The output should look like



```
In 1233
The number of '1's is 1
The number of '2's is 1
The number of '3's is 2
```

5. Determine the output of the following program without using the computer. What should be written in place of 2 at line 70 and 74 so that the program works correctly for any number of digits of the number stored in variable ‘s’.



```
67  s = "12321"
68  y = len(s)-1
69  c = 0
70  for x in range(0,2):
71      if s[x]==s[y]:
72          c=c+1
73      y = y-1
74  if c==2:
75     print("The number",s,"is a palindrome.")
76  else:
77     print("The number",s,"is not a palindrome.")
```

Experiment # 5

Implementation of Nested Loops in Python

1. Complete the following program to get the output as shown below.

```
for i in range (1,6):
    for j in range(1,):
        print(i)
```

Output

```
1
22
333
4444
55555
```

2. Complete the following program to get the output as shown below

```
for x in range(1, ,1):
    for y in range(1, ,1):
        print(x,y)
```

Output

```
( 1 , 1 ) ( 1 , 2 ) ( 1 , 3 )
( 2 , 1 ) ( 2 , 2 ) ( 2 , 3 )
( 3 , 1 ) ( 3 , 2 ) ( 3 , 3 )
```

3. Write a program that prompts the user to enter the number of rows of a pyramid and it then displays the pyramid at the output according to the number of rows given by the user. Use nested for-loops to get the output.

Output Sample 1

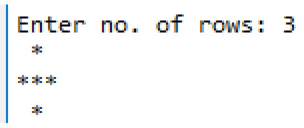
```
Enter no. of rows of the pyramid: 3
  *
 ***
*****
```

Output Sample 2

```
Enter no. of rows of the pyramid: 4
   *
  ***
 *****
*******
```

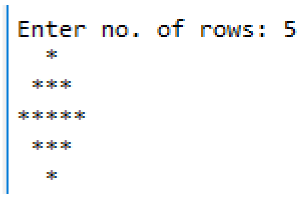
4. Write a program that prompts the user to enter the number of rows of a diamond and it then displays the diamond at the output according to the number of rows given by the user. Use nested for-loops to get the output.

Output Sample 1



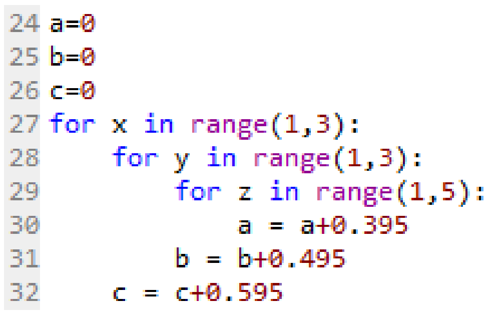
```
Enter no. of rows: 3
 *
***
 *
```

Output Sample 2



```
Enter no. of rows: 5
  *
 ***
*****
 ***
  *
```

5. Determine the output of the following program without using the computer. What are the values of a, b, c, x, y and z as soon as the 2nd for-loop ends for the very first time. What are the values of a, b, c, x, y and z after the program runs completely.



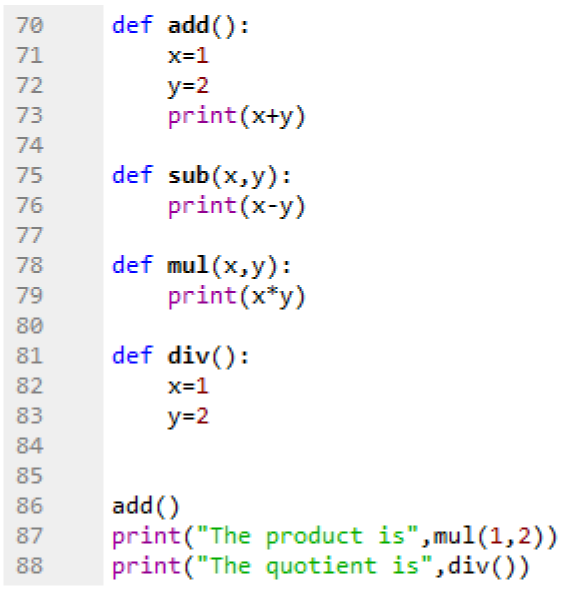
```python
a=0
b=0
c=0
for x in range(1,3):
    for y in range(1,3):
        for z in range(1,5):
            a = a+0.395
        b = b+0.495
    c = c+0.595
```

Experiment # 6
Implementation of Functions in Python

Topics to be covered in this Lab Manual

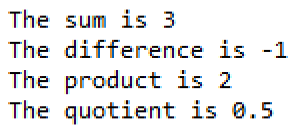
- Functions with and without parameters and return statements.
- Function(s) defined and used in another function.
- Function calling another function defined outside.
- Function used as function parameter/argument.

1. Complete the following code to get the output as shown below. The program should add, subtract, multiply and divide numbers 1 and 2. Do not change the instructions at line 86-88. You can add an instruction if necessary.



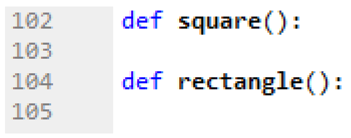
```
70  def add():
71      x=1
72      y=2
73      print(x+y)
74
75  def sub(x,y):
76      print(x-y)
77
78  def mul(x,y):
79      print(x*y)
80
81  def div():
82      x=1
83      y=2
84
85
86  add()
87  print("The product is",mul(1,2))
88  print("The quotient is",div())
```

Output



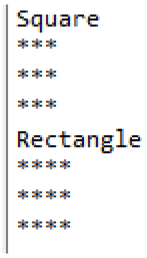
```
The sum is 3
The difference is -1
The product is 2
The quotient is 0.5
```

2. Complete the following program to print a square and a rectangle as shown below. The function *square* should print the square while the function *rectangle* should print the rectangle.



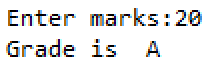
```
def square():

def rectangle():

```

Output



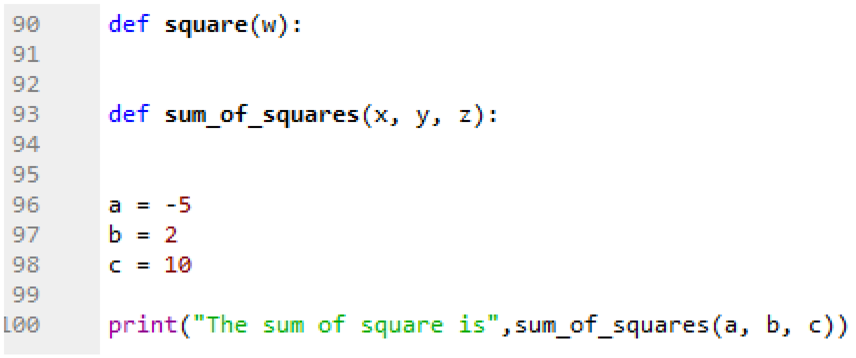
```
Square
***
***
***
Rectangle
****
****
****
```

3. Write a program that uses the user-defined function *grade* to display the grade of a student according to the marks obtained. The function *grade* should take *marks* as parameter and return the corresponding *grade* only. The marks should be entered by the user. The grades and marks range are given in question 2 , experiment 3.



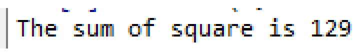
```
Enter marks:20
Grade is  A
```

4. Complete the following program to find the sum of squares of three numbers. The function *sum_of_squares* should call/use the function *square* to calculate and return the square of all the three numbers. The function *sum_of_squares* should then find the sum of the squares of all the three numbers and return it.



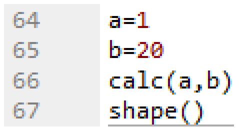
```python
def square(w):


def sum_of_squares(x, y, z):


a = -5
b = 2
c = 10

print("The sum of square is",sum_of_squares(a, b, c))
```

Output



```
The sum of square is 129
```

5. Complete the following program to print the outputs of the programs of question 1 and 2. The function *calc* should print the output of question 1 while the function *shape* should print the output of question 2. This question uses functions defined inside the other function. Use the codes of questions 1 and 2 in the definition of calc() and shape() functions..



```python
a=1
b=20
calc(a,b)
shape()
```

Output

```
The sum is 21
The difference is -19
The product is 20
The quotient is 0.05

Square
***
***
***

Rectangle
****
****
****
```

6. Write a program that asks the user to enter the number of rows of the pyramid and displays the pyramid. The program should continue to prompt the user to enter the number of rows and print the pyramid until the user types stop. The program should stop when user types stop. Use the user-defined function *pyr* to print the pyramid.

Output

```
Enter: 3
  *
 ***
*****

Enter: 4
   *
  ***
 *****
*******

Enter: 5
    *
   ***
  *****
 *******
*********

Enter: stop
```

7. Determine the output of the following program without using the computer.

```
def f(w):
    return 10*w

def g(x, y):
    return f(3*x) + y

def h(z):
    return f(g(z, f(z+1)))

print(h(1)) # hint: try the
```

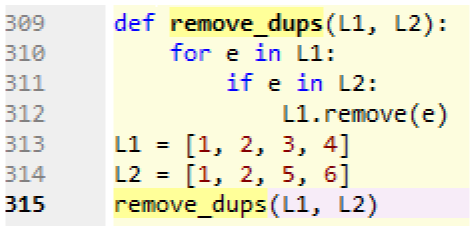
Experiment # 7
Implementation of List and Tuples in Python

Topics to be covered in this Lab Manual

- Tuples
- 1D and 2D lists.

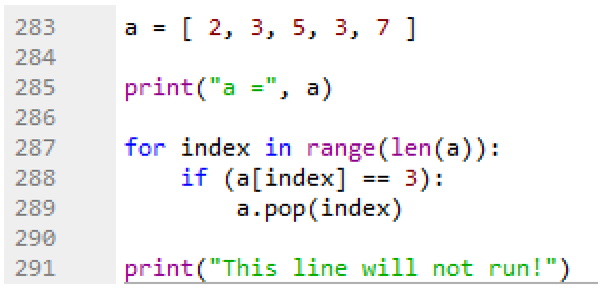
1. Determine the output of the following codes without using a computer. What problem did you notice while debugging each code?

Code 1



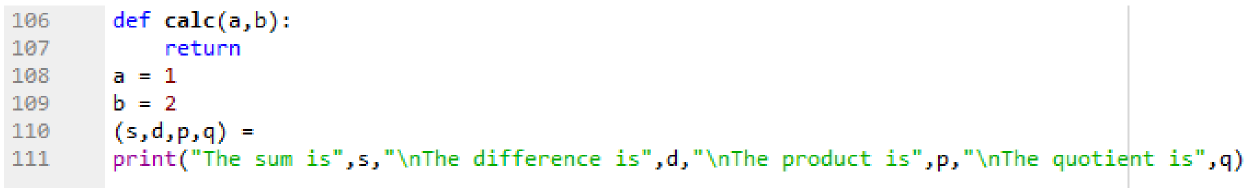
```python
def remove_dups(L1, L2):
    for e in L1:
        if e in L2:
            L1.remove(e)
L1 = [1, 2, 3, 4]
L2 = [1, 2, 5, 6]
remove_dups(L1, L2)
```

Code 2



```python
a = [ 2, 3, 5, 3, 7 ]

print("a =", a)

for index in range(len(a)):
    if (a[index] == 3):
        a.pop(index)

print("This line will not run!")
```

2. Complete the following code to get the output as shown below. The function *calc()* should return a tuple containing the sum, difference, product and quotient of variables a and b.



```python
def calc(a,b):
    return
a = 1
b = 2
(s,d,p,q) =
print("The sum is",s,"\nThe difference is",d,"\nThe product is",p,"\nThe quotient is",q)
```

Output

```
The sum is 3
The difference is -1
The product is 2
The quotient is 0.5
```

3. A. Write a program that uses a function *add_lists(L1,L2)* which takes two lists L1 and L2 and then calculates and prints their sum. The resultant sum must be stored in a third list L3. Use a for-loop to calculate the sum.

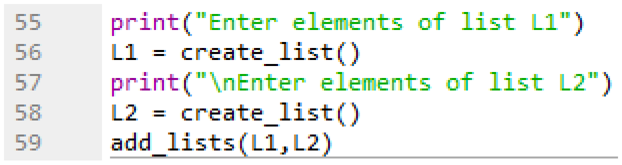
Output

```
The sum of [1, 2, 3, 6] and [3, 3, 3, 3] is [4, 5, 6, 9]
```

B. Write a program that stores the numbers entered by the user in a list until the user enters **'end'**. The list should then be displayed at the output as shown below.

```
enter:1

enter:5

enter:3

enter:end

The list is [1, 5, 3]
```

C. Complete the following code which adds two lists as in part a. The function *create_list()* should prompt the user to enter numbers until the user enters **'end'**. The function *create_list()* should return a list of numbers entered by the user.



```python
print("Enter elements of list L1")
L1 = create_list()
print("\nEnter elements of list L2")
L2 = create_list()
add_lists(L1,L2)
```

4. We can represent a polynomial as a list of its coefficients. For example, [2, 3, 0, 4] could represent the polynomial 2x**3 + 3x**2 + 4. With this in mind, write the function *multiplyPolynomials(p1, p2)* which takes two lists representing polynomials as just described, and returns a third list representing the polynomial which is the product of the two. For example, *multiplyPolynomials([2,0,3], [4,5])* represents the problem (2x**2 + 3)(4x + 5), and (2x**2 + 3)(4x + 5) = 8x**3 + 10x**2 + 12x + 15

And so this returns the list [8, 10, 12, 15].

5. Complete the following program that squares each element of a 2x2 2d list L1 without using any built-in function. The program should also display the number of rows and columns of the list L1.

```python
def square(l):
    L = [?]

    for row in range(0, ,1):
        for col in range(0, ,1):
            L[row][col] = ?

    print("The square of",l,"is",L)

l1 = [[1,2],[3,4]]
square(l1)
```

Output

```
The square of [[1, 2], [3, 4]] is [[1, 4], [9, 16]]
```

6. Write a program that uses the function *matrixMultiply(m1, m2)* that takes two 2d lists m1 and m2 (that we will consider to be matrices), each list can be of any number of rows and columns, and returns a new 2d list that is the result of multiplying the two matrices. Return **None** if the two matrices cannot be multiplied for any reason. Pay attention to the following concept to solve this task easily.

If $\mathbf{A}$ is an $m \times n$ matrix and $\mathbf{B}$ is an $n \times p$ matrix,

$$\mathbf{A} = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{pmatrix}, \quad \mathbf{B} = \begin{pmatrix} b_{11} & b_{12} & \cdots & b_{1p} \\ b_{21} & b_{22} & \cdots & b_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ b_{n1} & b_{n2} & \cdots & b_{np} \end{pmatrix}$$

the *matrix product* $\mathbf{C} = \mathbf{AB}$ (denoted without multiplication signs or dots) is defined to be the $m \times p$ matrix[6][7][8][9]

$$\mathbf{C} = \begin{pmatrix} c_{11} & c_{12} & \cdots & c_{1p} \\ c_{21} & c_{22} & \cdots & c_{2p} \\ \vdots & \vdots & \ddots & \vdots \\ c_{m1} & c_{m2} & \cdots & c_{mp} \end{pmatrix}$$

such that

$$c_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + \cdots + a_{in}b_{nj} = \sum_{k=1}^{n} a_{ik}b_{kj},$$

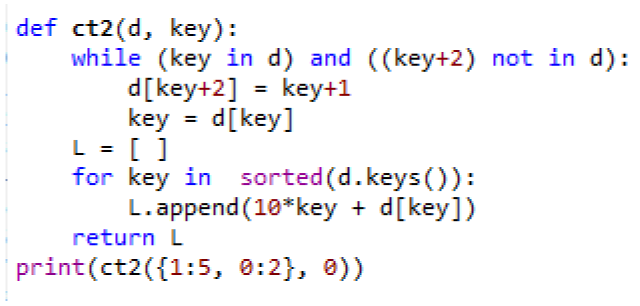
for $i = 1, \ldots, m$ and $j = 1, \ldots, p$.

Experiment # 8
Implementation of Dictionaries in Python

Topics to be covered in this Lab Manual

- Dictionary

1. Determine the output of the following program without using the computer.

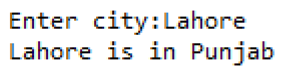


```python
def ct2(d, key):
    while (key in d) and ((key+2) not in d):
        d[key+2] = key+1
        key = d[key]
    L = [ ]
    for key in  sorted(d.keys()):
        L.append(10*key + d[key])
    return L
print(ct2({1:5, 0:2}, 0))
```

2. Create a dictionary of the table given below. Then prompt the user to enter the name of any city. If the city is in the dictionary, then print the city and its province, otherwise print the city is not in the dictionary.

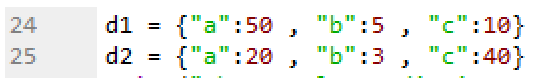
Keys	**Values**
Lahore	Punjab
Peshawar	KPK
Karachi	Sindh
Quetta	Balochistan
Gilgit	GB

Output



```
Enter city:Lahore
Lahore is in Punjab
```

3. Write a program that adds the values of two dictionaries of the same size. Create function *add()* which takes both the dictionaries as parameters and adds them. The result must be stored in another dictionary which must then be returned by the function *add()*



```python
d1 = {"a":50 , "b":5 , "c":10}
d2 = {"a":20 , "b":3 , "c":40}
```

Output

```
The resultant dictionary is {'a': 70, 'b': 8, 'c': 50}
```

4. Write a program that calculates and prints the average score of a student. The name of the student is stored as a key whereas the marks are stored in a list which itself is stored as a value in the dictionary as shown below.. The program should also print a dictionary containing the student names and their average marks. Create function *f()* to calculate and display the average marks. The program should work correctly for any size of the dictionary **D**.

```
D = {'Joe':[1,2,3] , 'John':[1,1,1] , 'Susan':[4,1,2,3]}
```

Output

```
The average of Joe is 2.0
The average of John is 1.0
The average of Susan is 2.5
{'Joe': 2.0, 'John': 1.0, 'Susan': 2.5}
```

5. Write a program that calculates and displays the number of students achieving a particular grade. Display the grade and the number of students in the form of a dictionary d1 as shown below. The program should work correctly for any of the grades A, B or C achieved by the student.

```
d  = {"Joe":"A" , "John":"A" , "Susan":"A"}
d1 = {"A":0 , "B":0 , "C":0}
```

Output

```
d =  {'Joe': 'A', 'John': 'A', 'Susan': 'A'}
d1 =  {'A': 3, 'B': 0, 'C': 0}
```

6. Write a program that prints the most frequent element in a list along with its frequency. The program must use a function ***mostFrequent(L)*** with list L as its parameter. The function's job is to

a. Print the list L and a dictionary containing all the elements as KEYS along with their frequencies as VALUES.

b. Print the most frequent element along with its frequency as shown below. The program should work correctly for any values in the list L of any length.

Note: You can use built-in function count() and max() in this program.

Output

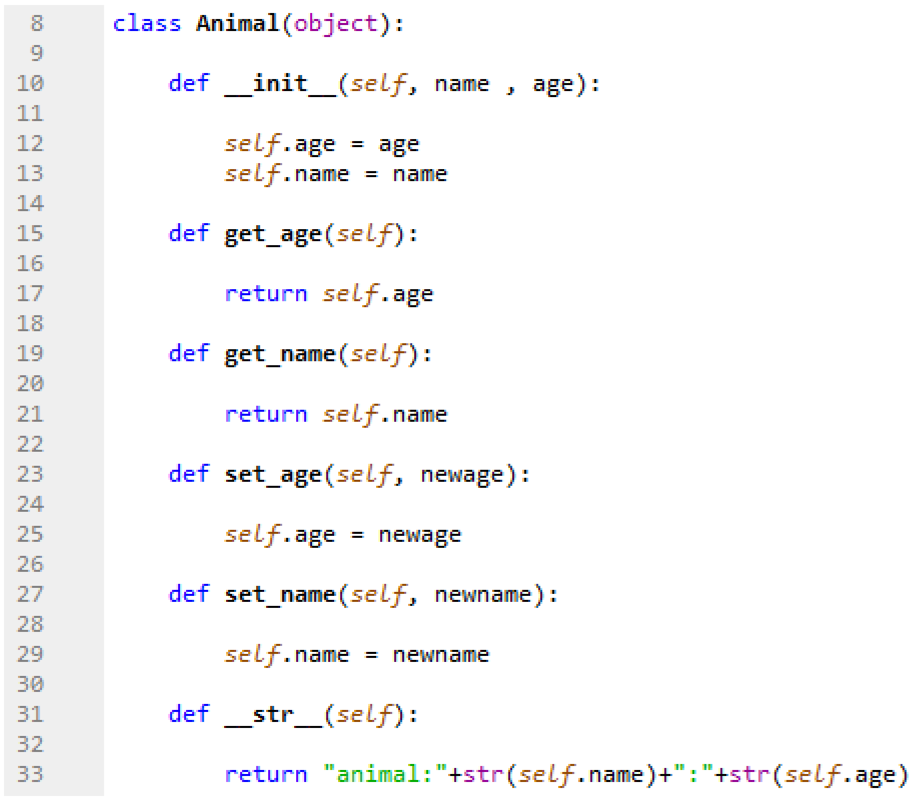
```
The list L is [5, 100, 21, 5, 1, 3, 10, 10, 3, 10, 5]
The dictionary is {5: 3, 100: 1, 21: 1, 1: 1, 3: 2, 10: 3}
The number(s) [5, 10] occurs more frequently and its the frequency is 3
```

Experiment # 9
Implementation of OOP in Python - Classes

Topics to be covered in this Lab Manual

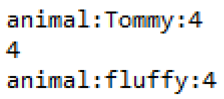
- Classes
- Getters and Setters
- Overriding special operators like print(), + , - , * , and float()
- Class variables

1. Complete the following code to get the output as shown below. The class Animal is complete and requires no changes.



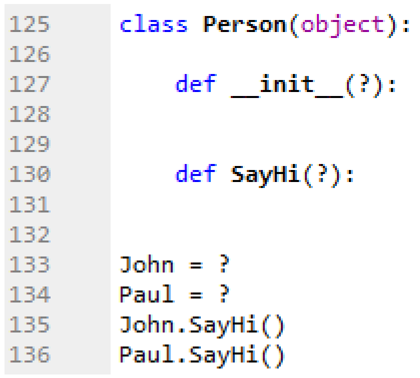
```python
class Animal(object):

    def __init__(self, name , age):

        self.age = age
        self.name = name

    def get_age(self):

        return self.age

    def get_name(self):

        return self.name

    def set_age(self, newage):

        self.age = newage

    def set_name(self, newname):

        self.name = newname

    def __str__(self):

        return "animal:"+str(self.name)+":"+str(self.age)
```

Output



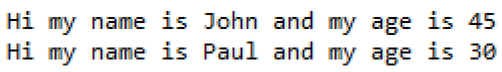
```
animal:Tommy:4
4
animal:fluffy:4
```

2. A. Complete the following program which uses a class Person that has two data attributes i.e. name and age. Write a method SayHi() so that the message “Hi, my name is John and my age is 45” is displayed when line 135 is executed. Similarly the message “Hi, my name is Paul and my age is 30” is displayed when line 136 is executed. Create two instances John and Paul of type Person.



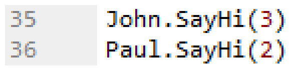
```
125    class Person(object):
126
127        def __init__(?):
128
129
130        def SayHi(?):
131
132
133    John = ?
134    Paul = ?
135    John.SayHi()
136    Paul.SayHi()
```

Output



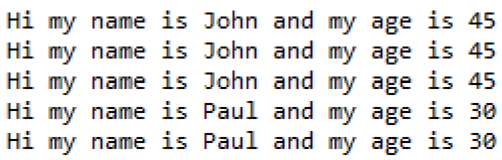
```
Hi my name is John and my age is 45
Hi my name is Paul and my age is 30
```

B. Modify the above program so that the method SayHi() accepts an integer representing the number of times the message, containing name and age of John and Paul, is displayed.



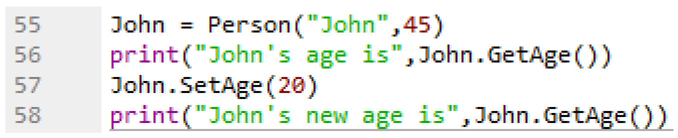
```
John.SayHi(3)
Paul.SayHi(2)
```

Output



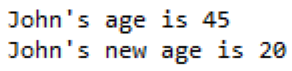
```
Hi my name is John and my age is 45
Hi my name is John and my age is 45
Hi my name is John and my age is 45
Hi my name is Paul and my age is 30
Hi my name is Paul and my age is 30
```

C. Write a method GetAge() that returns the age of the person. Write another method SetAge() to set the age of the person to a particular value.



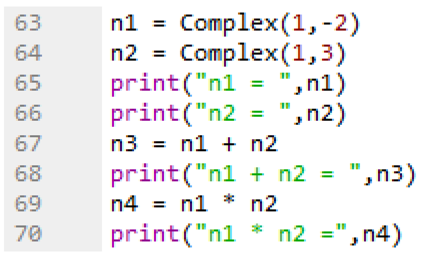
```python
John = Person("John",45)
print("John's age is",John.GetAge())
John.SetAge(20)
print("John's new age is",John.GetAge())
```

Output



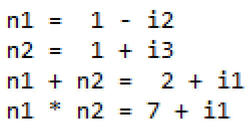
```
John's age is 45
John's new age is 20
```

3. Complete the following program which uses a class Complex that has two data attributes i.e. the real part of the complex number and the imaginary part of the complex number. The class Complex should contain three methods; one to print a complex number, second to add two complex numbers and third to multiply two complex numbers. Note: n3 and n4 are of type Complex in the code below



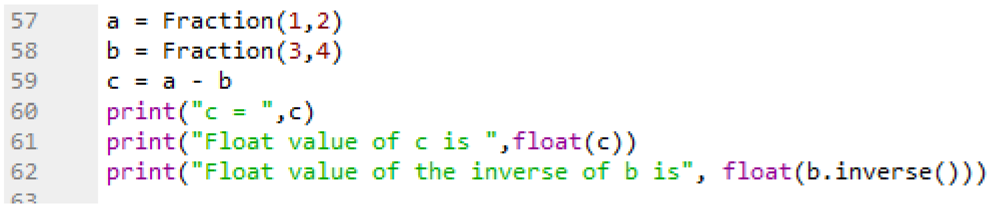
```python
n1 = Complex(1,-2)
n2 = Complex(1,3)
print("n1 = ",n1)
print("n2 = ",n2)
n3 = n1 + n2
print("n1 + n2 = ",n3)
n4 = n1 * n2
print("n1 * n2 =",n4)
```

Output



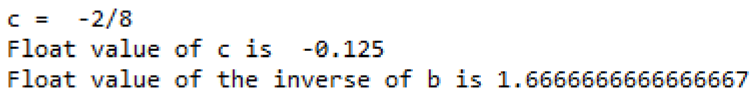
```
n1 =  1 - i2
n2 =  1 + i3
n1 + n2 =  2 + i1
n1 * n2 = 7 + i1
```

4. Complete the following program which uses a class Fraction that has two data attributes i.e. the numerator of a fraction and the denominator of the same fraction. The class Fraction should contain four methods; one to print a fraction, second to subtract two fractions, third to return the float value (in decimal point format) of the fraction and fourth to return the inverse of the fraction. Subtraction of two fractions means $\frac{(1*4)-(2*3)}{(2*4)}$ for a and b shown in the code below. Float of a fraction means $\frac{numerator+1}{denominator}$.
   Note: c is of type Fraction in the code below.



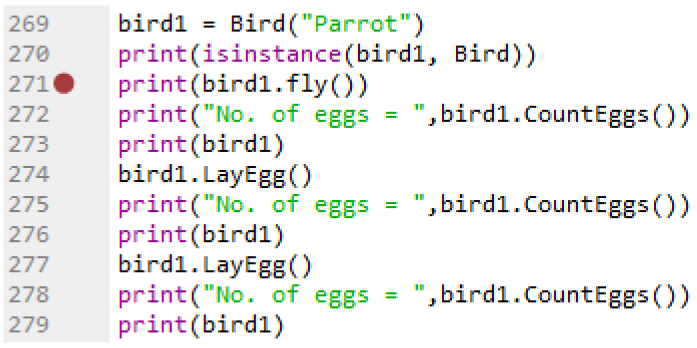
```
a = Fraction(1,2)
b = Fraction(3,4)
c = a - b
print("c = ",c)
print("Float value of c is ",float(c))
print("Float value of the inverse of b is", float(b.inverse()))
```

Output



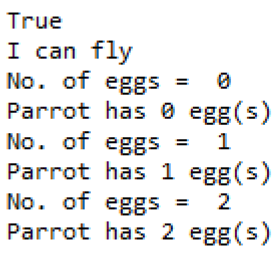
```
c =  -2/8
Float value of c is  -0.125
Float value of the inverse of b is 1.6666666666666667
```

5. Complete the following program to get the output as shown below. Figure out the class, its attributes and methods after analyzing the code and the output given below.



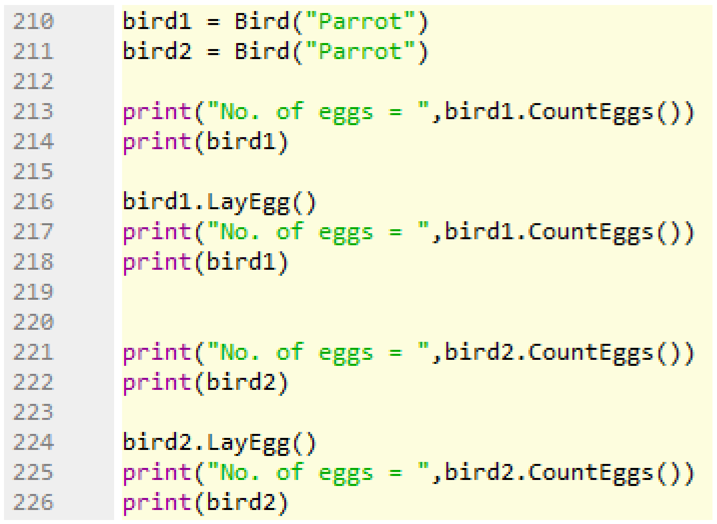
```
bird1 = Bird("Parrot")
print(isinstance(bird1, Bird))
print(bird1.fly())
print("No. of eggs = ",bird1.CountEggs())
print(bird1)
bird1.LayEgg()
print("No. of eggs = ",bird1.CountEggs())
print(bird1)
bird1.LayEgg()
print("No. of eggs = ",bird1.CountEggs())
print(bird1)
```

Output



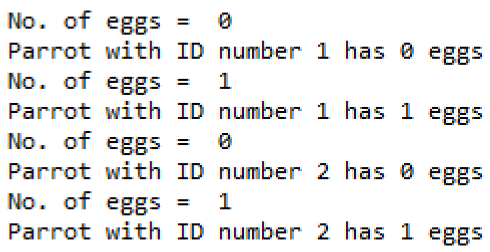
```
True
I can fly
No. of eggs =  0
Parrot has 0 egg(s)
No. of eggs =  1
Parrot has 1 egg(s)
No. of eggs =  2
Parrot has 2 egg(s)
```

6. Now modify the code in question 5 to include the identification number for each instance of type Bird. Use the concept of class variable to code this question.



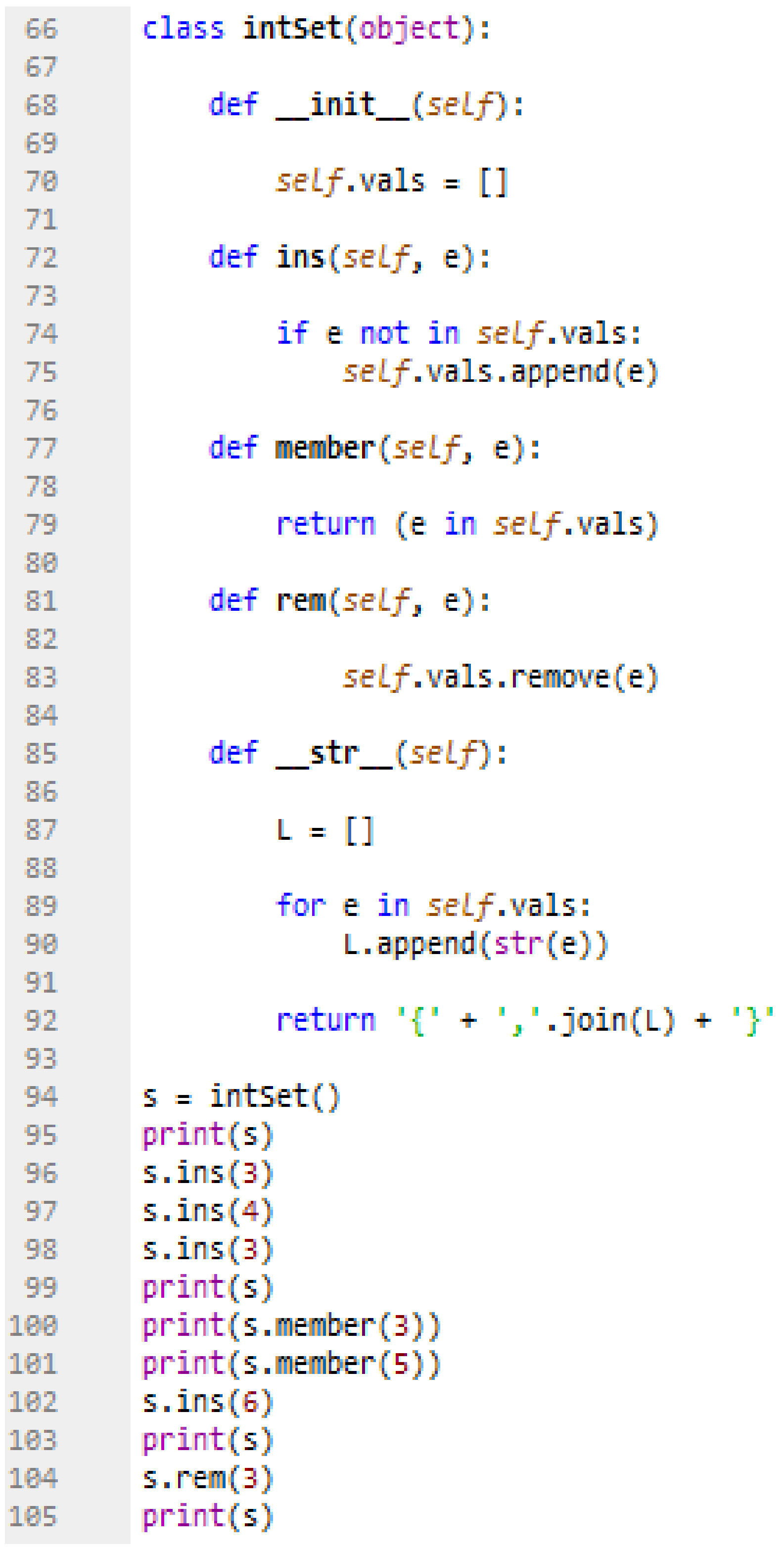
```python
bird1 = Bird("Parrot")
bird2 = Bird("Parrot")

print("No. of eggs = ",bird1.CountEggs())
print(bird1)

bird1.LayEgg()
print("No. of eggs = ",bird1.CountEggs())
print(bird1)


print("No. of eggs = ",bird2.CountEggs())
print(bird2)

bird2.LayEgg()
print("No. of eggs = ",bird2.CountEggs())
print(bird2)
```

Output



```
No. of eggs =  0
Parrot with ID number 1 has 0 eggs
No. of eggs =  1
Parrot with ID number 1 has 1 eggs
No. of eggs =  0
Parrot with ID number 2 has 0 eggs
No. of eggs =  1
Parrot with ID number 2 has 1 eggs
```

7. Determine the output of the following code without using the computer.



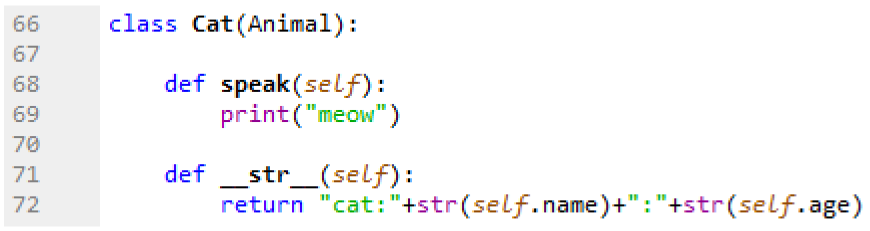
```python
class intSet(object):

    def __init__(self):

        self.vals = []

    def ins(self, e):

        if e not in self.vals:
            self.vals.append(e)

    def member(self, e):

        return (e in self.vals)

    def rem(self, e):

            self.vals.remove(e)

    def __str__(self):

        L = []

        for e in self.vals:
            L.append(str(e))

        return '{' + ','.join(L) + '}'

s = intSet()
print(s)
s.ins(3)
s.ins(4)
s.ins(3)
print(s)
print(s.member(3))
print(s.member(5))
s.ins(6)
print(s)
s.rem(3)
print(s)
```

Experiment # 10
Implementation of OOP in Python - Inheritance

Topics to be covered in this Lab Manual

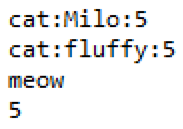
- Inherit methods of parent class
- Override parent class methods

1. Complete the following code to get the output as shown below. The subclass Cat **inherits** all the methods of the parent class Animal defined in question 1, experiment 9. The subclass Cat is completely defined and requires no changes.



```python
class Cat(Animal):

    def speak(self):
        print("meow")

    def __str__(self):
        return "cat:"+str(self.name)+":"+str(self.age)
```

Output



```
cat:Milo:5
cat:fluffy:5
meow
5
```

2. Write a subclass Person which inherits all the methods of the parent class Animal defined in question 1, experiment 9. The subclass Person should contain four additional methods; first method Speak() that prints "Hello", second method get_friends() that returns a list of the names of the friends, third method add_friend() that adds the name of a friend (duplicate names not allowed) to a list and fourth method to print any instance of type Person as shown below. Note that each instance p1 and p2 has its own list of friends. Also note that instance p1 and p2 have two parameters only i.e. name and age. Based on these two observations, write the __init__ method for the class Person.

```python
p1 = Person("jack", 30)
p2 = Person("jill", 25)
print("p1 name = ",p1.get_name())
print("p2 age = ",p2.get_age())
print(p1)
p1.speak()
print("p1 friends = ",p1.get_friends())
p1.add_friend("john")
print("p1 friends = ",p1.get_friends())
```

Output

```
p1 name =  jack
p2 age =  25
person:jack:30
hello
p1 friends =  []
p1 friends =  ['john']
```

3. Write a subclass Student which inherits all the methods of its parent class Person defined in question 2, experiment 10. It should contain some additional methods and attribute(s) which can be determined by analyzing the code and the output given below.

```python
s1 = Student('alice', 20, "EE")
s2 = Student('beth', 18 , "CS")
print(s1)
print(s2)
s2.change_dept("EE")
print(s2)
print(s1.get_name(),"says:", end=" ")
s1.speak()
print("s1 age = ",s1.get_age())
s1.set_age(30)
print("s1 age = ",s1.get_age())
print("s2 friends = ",s2.get_friends())
s2.add_friend("Kattie")
print("s2 friends = ",s2.get_friends())
```

Output

```
student:alice:20:EE
student:beth:18:CS
student:beth:18:EE
alice says: I am a student
s1 age =  20
s1 age =  30
s2 friends =  []
s2 friends =  ['Kattie']
```

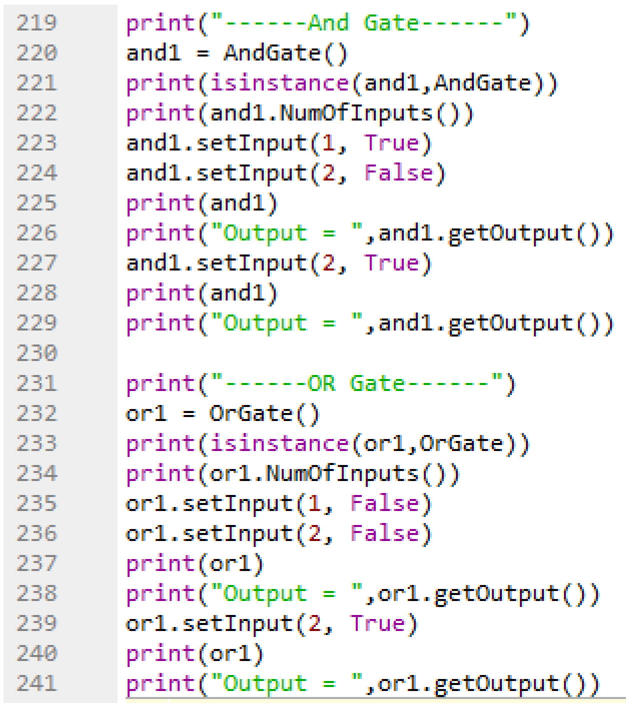
4. Complete the following code without changing lines 269 - 294. Bird and Penguin are class names. Study the code carefully to determine the parent class, the subclass, the attributes and the methods.

```
269    print("------Parrot Testing------\n")
270    bird1 = Bird("Parrot")
271    print(isinstance(bird1, Bird))
272    print(bird1.fly())
273    print("No. of eggs = ",bird1.CountEggs())
274    print(bird1)
275    bird1.LayEgg()
276    print("No. of eggs = ",bird1.CountEggs())
277    print(bird1)
278    bird1.LayEgg()
279    print("No. of eggs = ",bird1.CountEggs())
280    print(bird1)
281
282    print("\n------Penguin Testing------\n")
283    bird2 = Penguin("Emperor Penguin")
284    print(isinstance(bird2, Penguin))
285    print(bird2.fly())
286    print(bird2.swim())
287    print("No. of eggs = ",bird2.CountEggs())
288    print(bird2)
289    bird2.LayEgg()
290    print("No. of eggs = ",bird2.CountEggs())
291    print(bird2)
292    bird2.LayEgg()
293    print("No. of eggs = ",bird2.CountEggs())
294    print(bird2)
```

Output

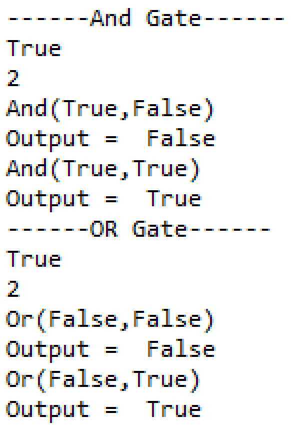
```
------Parrot Testing------

True
I can fly
No. of eggs =  0
Parrot has 0 egg(s)
No. of eggs =  1
Parrot has 1 egg(s)
No. of eggs =  2
Parrot has 2 egg(s)

------Penguin Testing------

True
No flying for me.
I can swim.
No. of eggs =  0
Emperor Penguin has 0 egg(s)
No. of eggs =  1
Emperor Penguin has 1 egg(s)
No. of eggs =  2
Emperor Penguin has 2 egg(s)
```

5. Complete the following code without changing lines 219 - 241. The class AndGate and OrGate should contain only one method getOutput(). The method getOutput() outputs True or False based on the type of gate. Write a separate class Gate to accommodate other methods shown in the code. Determine the parent class, subclass(es), the attributes and methods after studying the code and the output given below carefully.



```
219    print("------And Gate------")
220    and1 = AndGate()
221    print(isinstance(and1,AndGate))
222    print(and1.NumOfInputs())
223    and1.setInput(1, True)
224    and1.setInput(2, False)
225    print(and1)
226    print("Output = ",and1.getOutput())
227    and1.setInput(2, True)
228    print(and1)
229    print("Output = ",and1.getOutput())
230
231    print("------OR Gate------")
232    or1 = OrGate()
233    print(isinstance(or1,OrGate))
234    print(or1.NumOfInputs())
235    or1.setInput(1, False)
236    or1.setInput(2, False)
237    print(or1)
238    print("Output = ",or1.getOutput())
239    or1.setInput(2, True)
240    print(or1)
241    print("Output = ",or1.getOutput())
```

Output



```
------And Gate------
True
2
And(True,False)
Output =  False
And(True,True)
Output =  True
------OR Gate------
True
2
Or(False,False)
Output =  False
Or(False,True)
Output =  True
```

Experiment # 11
Data Manipulation and Analysis in Python

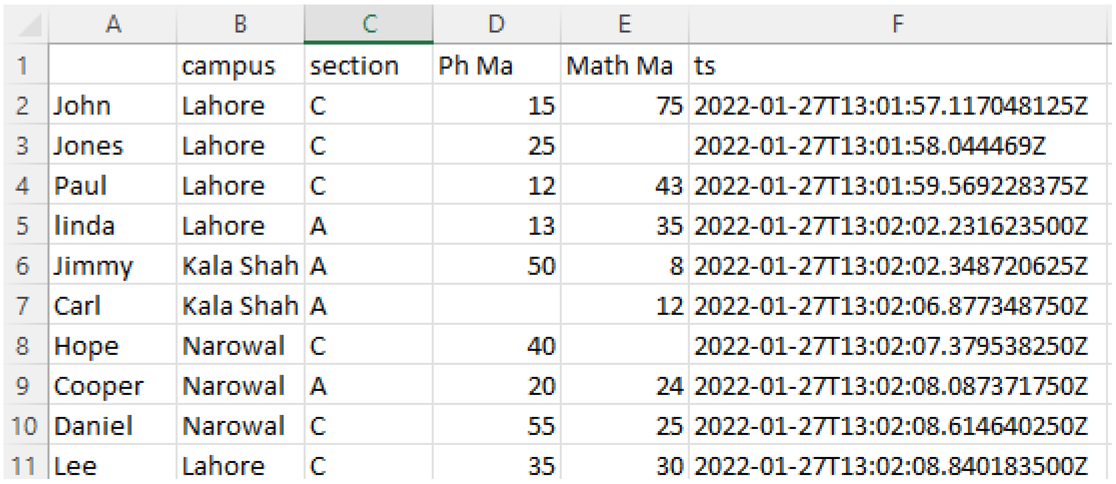
Topics to be covered in this Lab Manual

- Use PANDAS tool to manipulate, analyze and plot the data

1. Student related information is present in tabular form as shown below. The tabular data is stored in StudentList.csv format using Microsoft Excel. Use PANDAS library in Python to
    1. Rename the first row of each column.
    2. Count and display unique items of all columns or of a specific column.
    3. Rename each item of the campus column to its abbreviation eg. Lahore to LHR.
    4. Iterate over the column names.
    5. Create a new column cmp-sec which is the combination of columns campus and section.
    6. Rearrange the columns.
    7. Find the max, min and mean values of columns Ph Ma and Math Ma.
    8. Slice the dataframe.
    9. Group the columns campus and section together and count the number of items.
    10. Convert an object to dataframe.
    11. Convert the column ts to a recognizable format and set it as the index of the dataframe.
    12. Count the number of items of the cmp-sec column in a specified time interval say 10 sec.
    13. Reset the index of the dataframe.
    14. Plot data for easy visualization.

Note: The required output can be seen here. The StudentList.csv can be accessed here.

Data to be Manipulated and Analyzed

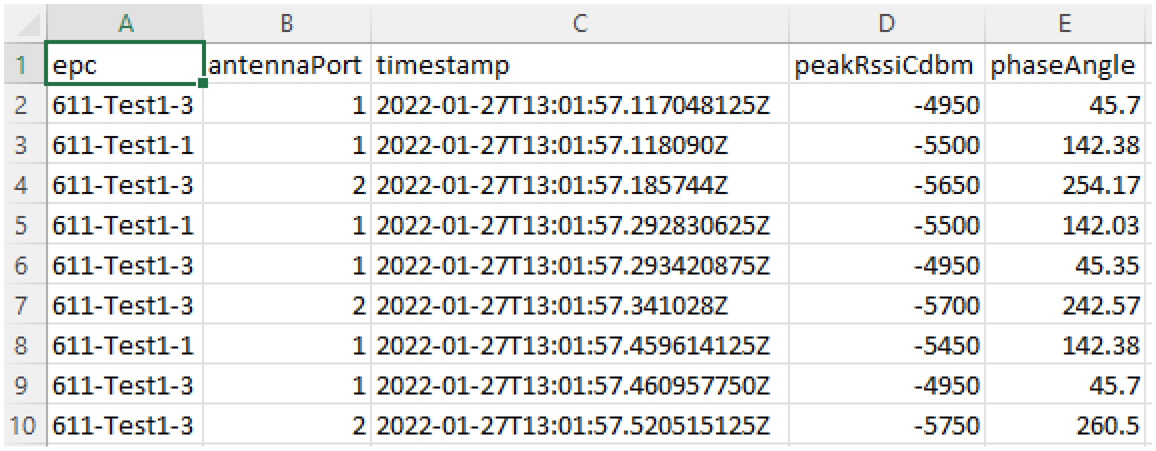


	A	B	C	D	E	F
1		campus	section	Ph Ma	Math Ma	ts
2	John	Lahore	C	15	75	2022-01-27T13:01:57.117048125Z
3	Jones	Lahore	C	25		2022-01-27T13:01:58.044469Z
4	Paul	Lahore	C	12	43	2022-01-27T13:01:59.569228375Z
5	linda	Lahore	A	13	35	2022-01-27T13:02:02.231623500Z
6	Jimmy	Kala Shah	A	50	8	2022-01-27T13:02:02.348720625Z
7	Carl	Kala Shah	A		12	2022-01-27T13:02:06.877348750Z
8	Hope	Narowal	C	40		2022-01-27T13:02:07.379538250Z
9	Cooper	Narowal	A	20	24	2022-01-27T13:02:08.087371750Z
10	Daniel	Narowal	C	55	25	2022-01-27T13:02:08.614640250Z
11	Lee	Lahore	C	35	30	2022-01-27T13:02:08.840183500Z

2. The data file normal.csv will be provided with the laboratory experiment. The file contains radio frequency (RF) signal values strengths (known as peakRssi) in dbm (decibel-milliwatts) and phase in degrees obtained from tags, known as RFID tags. The tags are attached to a chair. Tag Test-1 is at the seat of the chair, tag Test-2 is at the back of the chair and tag Test-3 is at the top of the top-rail of the chair. There are two antennas in the room: 1 and 2, which catch these signals from the tags, so every tag has two readings received at the two antennas. The tag signals may or may not reach an antenna, so all tags or antenna signals for those tags may not be present in the file. **Perform steps 1-14 as done in question 1.**

Note: The normal.csv file can be accessed here

Data to be Manipulated and Analyzed



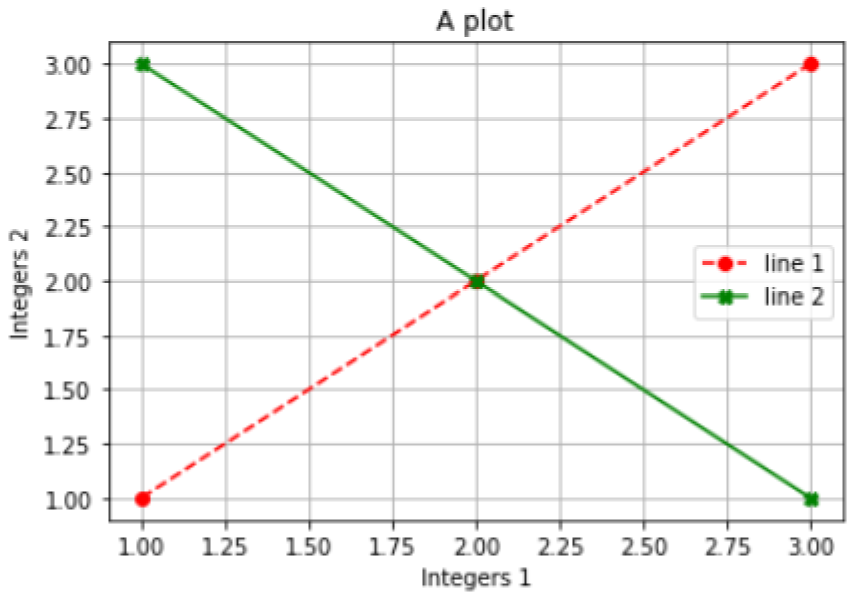
	A	B	C	D	E
1	epc	antennaPort	timestamp	peakRssiCdbm	phaseAngle
2	611-Test1-3	1	2022-01-27T13:01:57.117048125Z	-4950	45.7
3	611-Test1-1	1	2022-01-27T13:01:57.118090Z	-5500	142.38
4	611-Test1-3	2	2022-01-27T13:01:57.185744Z	-5650	254.17
5	611-Test1-1	1	2022-01-27T13:01:57.292830625Z	-5500	142.03
6	611-Test1-3	1	2022-01-27T13:01:57.293420875Z	-4950	45.35
7	611-Test1-3	2	2022-01-27T13:01:57.341028Z	-5700	242.57
8	611-Test1-1	1	2022-01-27T13:01:57.459614125Z	-5450	142.38
9	611-Test1-3	1	2022-01-27T13:01:57.460957750Z	-4950	45.7
10	611-Test1-3	2	2022-01-27T13:01:57.520515125Z	-5750	260.5

Experiment # 12
Plotting Data in Python

Topics to be covered in this experiment

- Use MATPLOTLIB and NUMPY tools to create and plot the data.

1. Write a program to get the output as shown below.



2. Write a program to display a sinusoidal wave as shown below. For this purpose, define a function Sine() that takes the amplitude, frequency, phase shift, stop time, label and color of the plot as inputs. Define another function SetPlot() to label the plot and display the grid. Complete the code shown below to get the required output. The general equation of a sinusoidal wave is

$$y = A * \sin((2*\pi*f*t) + \varphi)$$

where A = amplitude of the sine wave

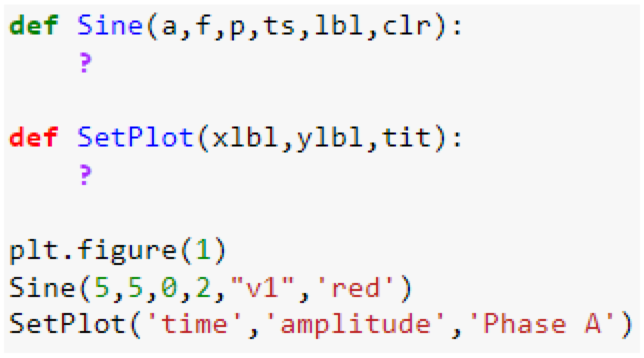
f = frequency of the sine wave in Hertz (Cycles/Second)

φ = Phase shift of the sine wave

t = time in seconds

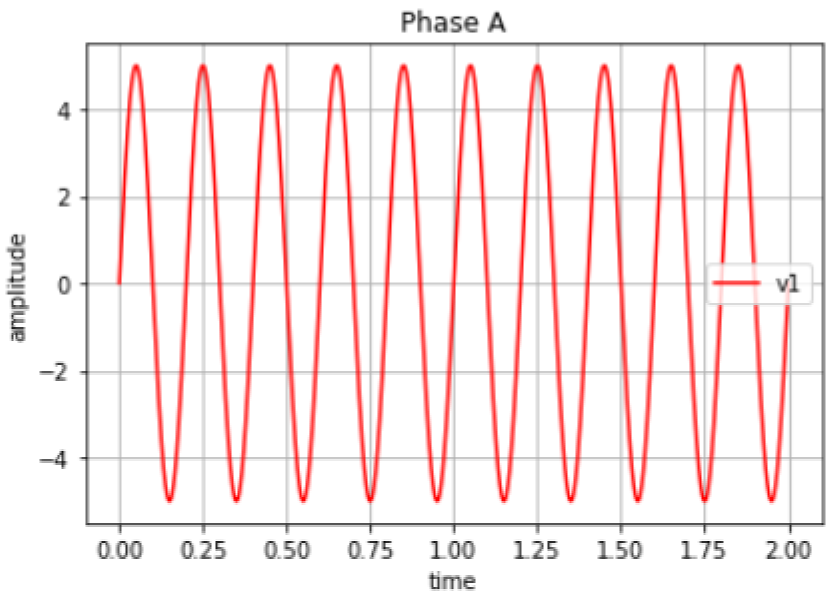
Note: Python considers the units of the angle of the function sin() to be radians i.e. sin(90) = 0.893 instead of sin(90) = 1.

Incomplete Code



```
def Sine(a,f,p,ts,lbl,clr):
    ?

def SetPlot(xlbl,ylbl,tit):
    ?

plt.figure(1)
Sine(5,5,0,2,"v1",'red')
SetPlot('time','amplitude','Phase A')
```

Output



3. Write a program to display three sinusoidal waves in one figure as shown below. For this purpose, define a function ***PrintSine()*** that takes the amplitude, frequency, phase shift, stop time and label of the plot as inputs. Complete the code below to get the output shown below. The general equation of the three sinusoidal waves are

v1 = A * sin(2*pi*f*t)

v2 = A * sin((2*pi*f*t) + 120)

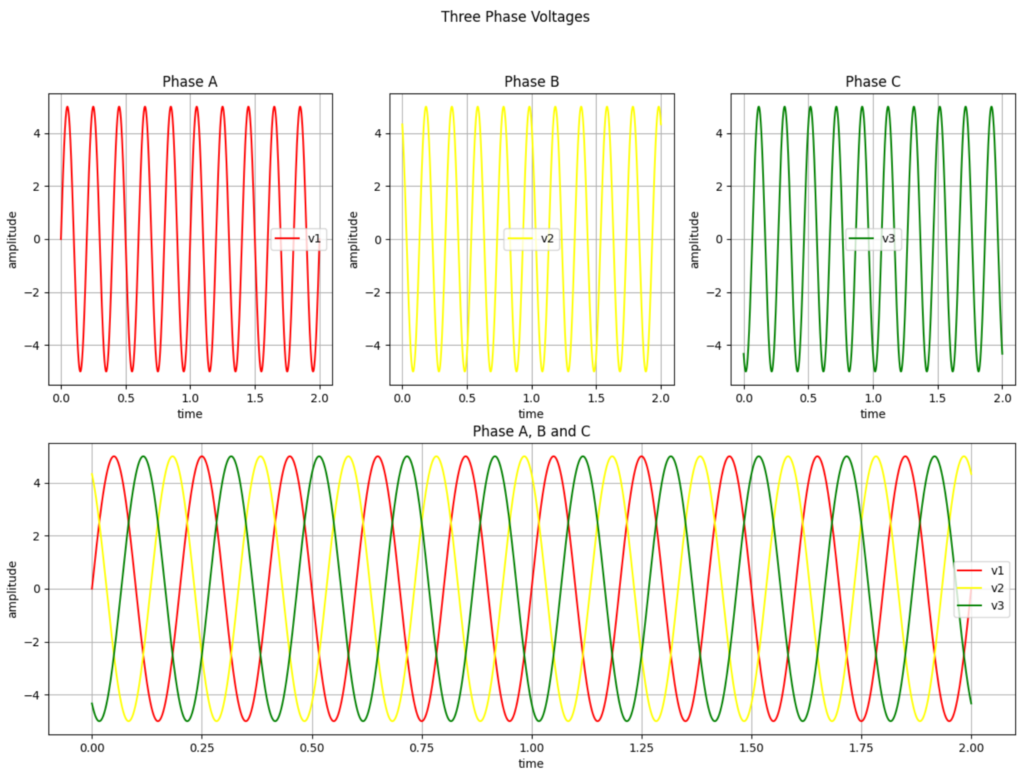
v3 = A * sin((2*pi*f*t) + 240)

Where A = Amplitude of the sine wave

f = Frequency of the sine wave in Hertz (Cycles/Second)

t = time in seconds

Output

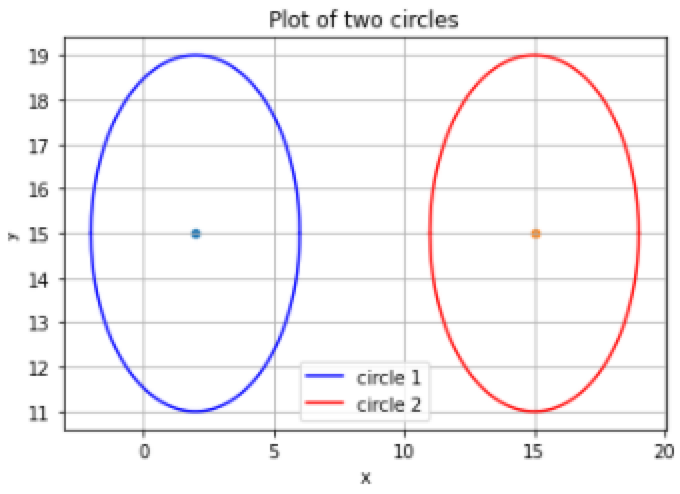


4. Write a program to display two circles with their centers as shown below. In the figure below, both circles have radius = 4 and centers at (2,15) and (15,15). The equation of a circle is

$$(x - a)^2 + (y - b)^2 = r^2$$

where 'a' and 'b' are the coordinates representing the center of the circle and 'r' is the radius of the circle. The inputs are a, b, r, color of the circle and name of the circle. It is better to define a function ***PrintCircle()*** that takes these inputs as parameters and prints the circle with the center.

Output

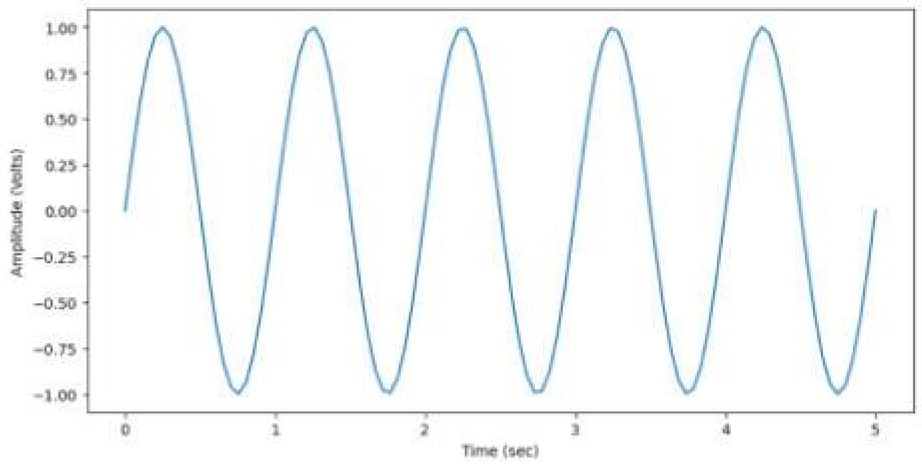


Experiment # 13
Signal Manipulation & Processing in Python

In this laboratory experiment, we will generate sine wave signals and add them to see how addition of different sine waves can result in different periodic signals, such as a square wave. You will also generate and listen to the audio corresponding to a sine wave. The audio corresponding to a sine wave sounds like a tone.

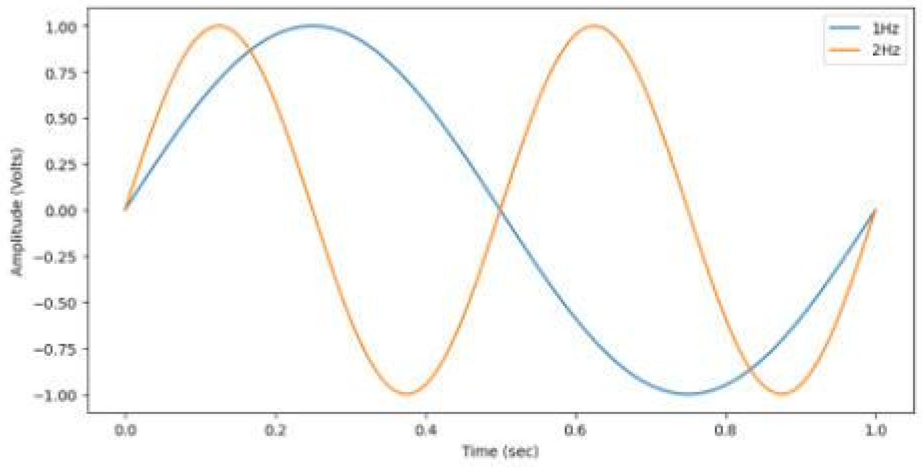
In [4]:
```
import numpy as np
import matplotlib.pyplot as plt


# Generate a sine wave of frequency f = 1Hz for x-axis (time) values from 0 to 1.
# The sine wave will have 100 samples and we will get one cycle for f = 1 Hz.

f = 1 # Use 1, 2, 3, Hz frequencies and check the output.

t = np.linspace(0,5,100) # 100 is the number of samples. Use 50, 100, 300, 500 sc

# if "t" is a numpy array the output "v" will also be a numpy array
v = np.sin(2*np.pi*f*t)

# Determine size of image before plotting width=10, height=5, and dots per inch (
plt.figure(figsize=(10, 5), dpi=100)

# plot function to plot, x-axis variable "t" comes first and y-axis variable "v"
plt.plot(t,v)

# For x-axis and y-axis labels
plt.xlabel("Time (sec)")
plt.ylabel("Amplitude (Volts)")
```

Out[4]:
```
Text(0, 0.5, 'Amplitude (Volts)')
```

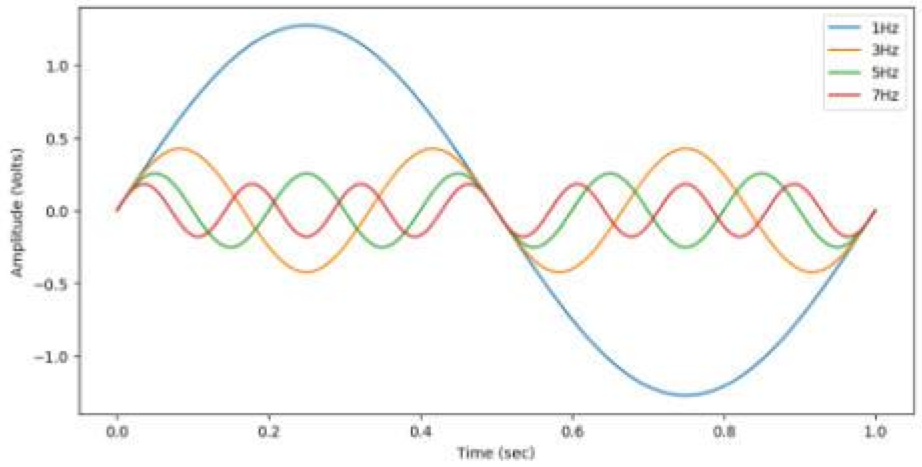


In [16]:
```
# plot sine waves with frequencies of multiples of 1Hz
t=np.linspace(0,1,200)


multiples = 2 # Select multiples = 1, 2, 3, 4 and check the output

plt.figure(figsize=(10, 5), dpi=100)

for k in range(1,multiples+1):
    f=k
    v = np.sin(2*np.pi*f*t)
    plt.plot(t,v,label=str(f)+'Hz') # label is used to assign a legend to a signa

# To display the legend, used in the end to display all
plt.legend()

plt.xlabel("Time (sec)")
plt.ylabel("Amplitude (Volts)")
```

Out[16]:
```
Text(0, 0.5, 'Amplitude (Volts)')
```

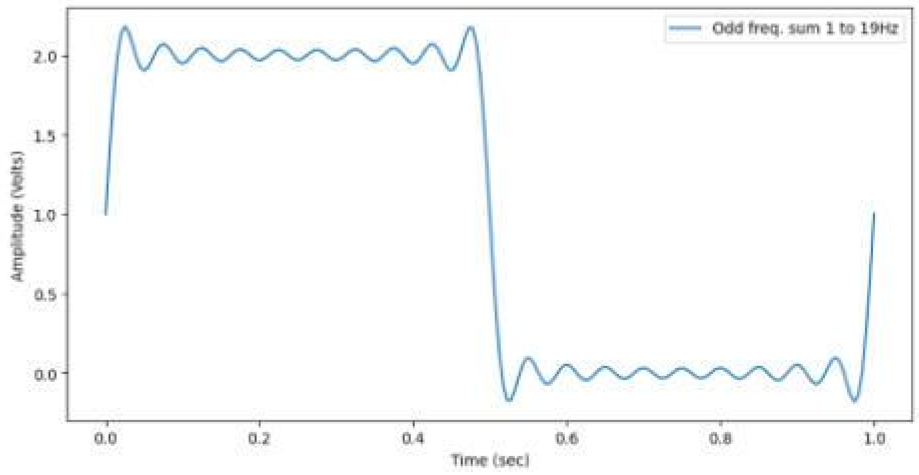


```
In [8]: # plot sine waves with frequencies of odd multiples of 1Hz and decreasing amplitu

        t=np.linspace(0,1,200)

        # check odd_multiples: Select 3, 4, 5 and 6 and then check the output
        multiples = 4

        plt.figure(figsize=(10, 5), dpi=100)

        for k in range(1,multiples+1):
            f=2*k-1                      # computes odd multiples
            magnitude = (4/np.pi)*(1/f) # change magnitude of each odd multiple

            v = magnitude*np.sin(2*np.pi*f*t)
            plt.plot(t,v,label=str(f)+'Hz')

        plt.legend()
        plt.xlabel("Time (sec)")
        plt.ylabel("Amplitude (Volts)")

Out[8]: Text(0, 0.5, 'Amplitude (Volts)')
```



In [11]:
```
# Let's add above odd multiples to see the resulting waveform
square=1
t=np.linspace(0,1,200)

# check a number of odd multiples of up to 1, 2, 3, 10, 50, 100, 200
no_of_multiples = 10

for k in range(1,no_of_multiples+1):
    f=2*k-1
    magnitude = (4/np.pi)*(1/f)
    v = magnitude*np.sin(2*np.pi*f*t)
    square = square +v

plt.figure(figsize=(10, 5), dpi=100)
plt.plot(t,square,label="Odd freq. sum 1 to "+str(f)+"Hz")
plt.legend()
plt.xlabel("Time (sec)")
plt.ylabel("Amplitude (Volts)")
```

Out[11]:
```
Text(0, 0.5, 'Amplitude (Volts)')
```

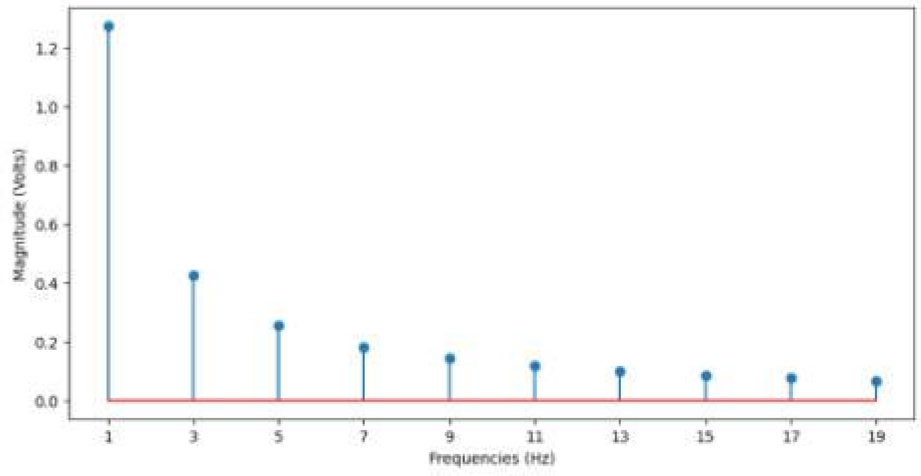


Adding sine waves makes a square wave

Adding different sine waves with odd multiples of 1Hz, produces a square wave of 1Hz. Notice, the sine waves must be added with a particular amplitude value to get a perfect square wave. The square wave becomes almost perfect with addition of 200 odd multiples of 1Hz sine wave. The following plots the sine wave frequencies on the x-axis to show, which different frequencies and their correponsing amplitudes make a square wave.

In [5]:
```
# Plot sine wave frequencies on the x-axis and their magnitudes (amplitudes) on t
square=1

t=np.linspace(0,1,200)


freq=[]
magn=[]
# check number of odd multiples 1, 2, 3, 10, 50, 100, 200


odd_multiples = 10

for k in range(1,odd_multiples+1):
    f=2*k-1
    magnitude = (4/np.pi)*(1/f)
    v = magnitude*np.sin(2*np.pi*f*t)

    square = square + v

    freq.append(f)
    magn.append(magnitude)

plt.figure(figsize=(10, 5), dpi=100)
# The stem plot draws vertical lines
plt.stem(freq, magn)
# prints exactly 1,3,5..Hz at the x-axis instead of default values
# uncomment plt.xticks below and execute again
plt.xticks(freq)

# prints x and y-axis labels
plt.xlabel("Frequencies (Hz)")
plt.ylabel("Magnitude (Volts)")
```

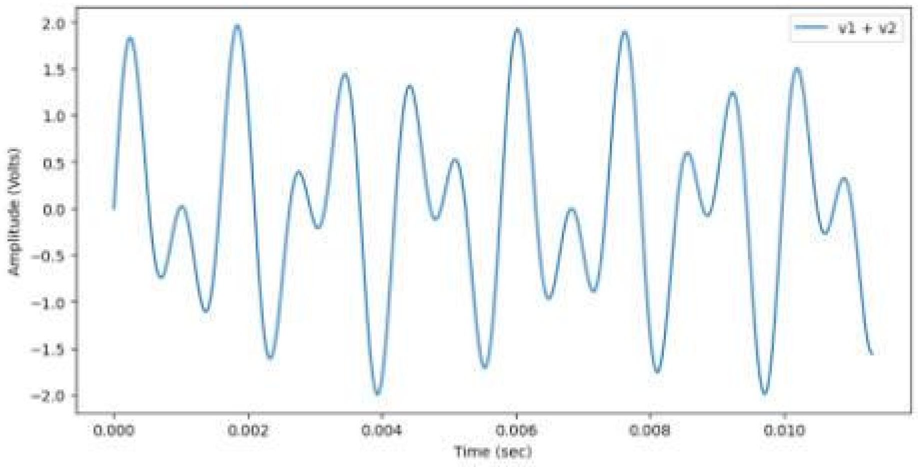
Out[5]:
```
Text(0, 0.5, 'Magnitude (Volts)')
```



In [13]:
```
t=np.linspace(0,1,44100)

fv1 = 697
fh1 = 1209

v1 = np.sin(2*np.pi*fv1*t)
v2 = np.sin(2*np.pi*fh1*t)

signal = v1 + v2

# Let's display first 500 samples of the signal
plt.figure(figsize=(10, 5), dpi=100)
plt.plot(t[:500],signal[:500],label="v1 + v2")

plt.legend()
plt.xlabel("Time (sec)")
plt.ylabel("Amplitude (Volts)")

from IPython.display import Audio
Audio(signal, rate=44100)  # Convert signal to audio
```

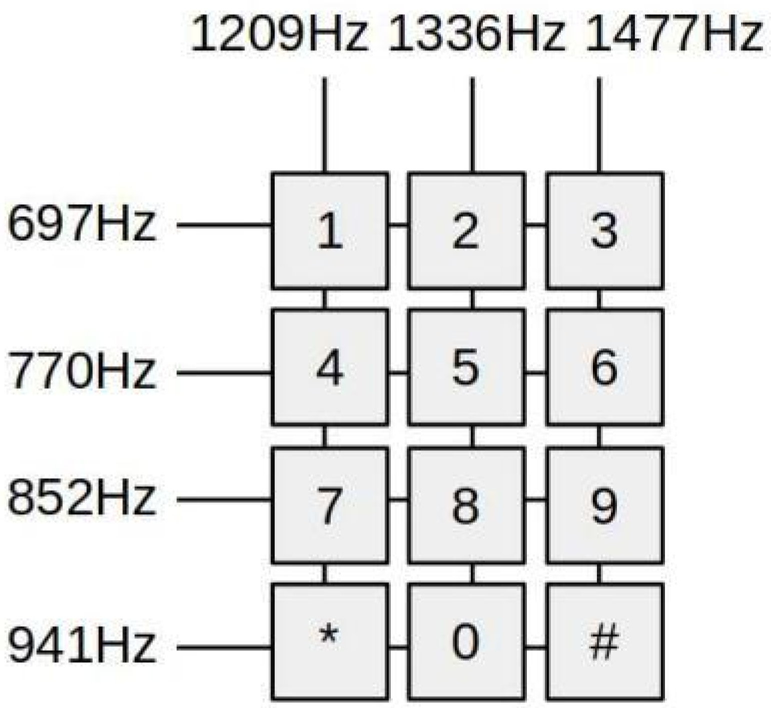
Out[13]:

0:00 / 0:01



Dual-Tone Multi-Frequency (DTMF) signals

DTMF signals belongs to a group of multi-frequency signals, which use two sine wave sounds. DTMF signals utilize 8 frequencies to represent 16 sounds for different digits/keys pressed over a telephone line. Seven of the eight frequencies are enough to represent the 10 digits from 0 to 9 along with * and # keys, as illustrated in Figure below:



For example, the digit 5 is a sum of two sine waves of frequencies 770Hz and 1336Hz. In this part, we will generate sine waves and add them to generate DTMF signals for digits 1 and 2. We will then multiply each of these signals with the "signal" in the previous section (which also represents a DTMF digit) and sum the resulting values. The higher the value of this sum of products computation, the greater the similarity between the two multiplied signals. We can use this technique to determine a digit in an unknown DTMF signal. The following section demonstrates the procedure.

In [7]:
```python
# DTMF signals for digits 1 and 2

fv1 = 697   # Frequency. See figure above showing the dial pad
fh1 = 1209  # Frequency. See figure above showing the dial pad
fh2 = 1336  # Frequency. See figure above showing the dial pad

sin_fv1 = np.sin(2*np.pi*fv1*t)
sin_fh1 = np.sin(2*np.pi*fh1*t)
sin_fh2 = np.sin(2*np.pi*fh2*t)

dtmf_digit1 = sin_fv1 + sin_fh1 # DTMF signal for digit 1
dtmf_digit2 = sin_fv1 + sin_fh2 # DTMF signal for digit 2


dtmf_digit_signals = [dtmf_digit1, dtmf_digit2]
value = []

# Multiply the dtmf digit 1 and 2 signals, each with the previous signal separate

for digit_signal in dtmf_digit_signals:
    out = np.sum(signal*digit_signal)
    value.append(out)


# Find the maximum sum of the two
print(value)
print('Maximum value = ',max(value))

#Find the index of the maximum sum, which belongs to the first signal correspondi
max_index = value.index(max(value))
print('Index of Maximum value = ',max_index)
print('The signal has the maximum similarity to digit 1.')
```

```
[44098.99999999998, 22049.499999999996]
Maximum value =  44098.99999999998
Index of Maximum value =  0
The signal has the maximum similarity to digit 1.
```

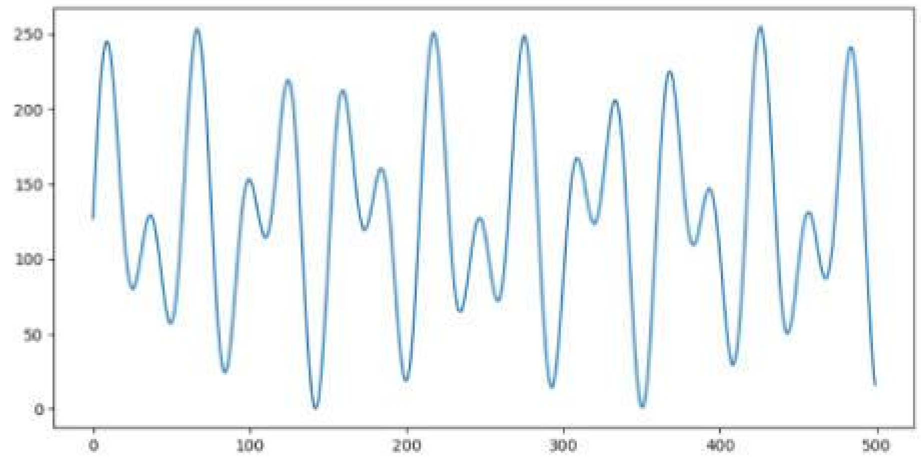
Loading an audio file

Load an audio file from disk containing the DTMF tone of a digit. Plot the signal and listen to the audio file

In [6]:
```
from scipy.io.wavfile import read #import the required function from the module t

samplerate, loaded_signal = read('dtmf_signal.wav')

# The signal amplitude is from 0 to 255
plt.figure(figsize=(10, 5), dpi=100)
plt.plot(loaded_signal[:500])


Audio(loaded_signal, rate=44100) # Listen to the audio signal
```

Out[6]:

0:00 / 0:00



Find out the digit in the loaded audio signal

Generate DTMF signals for different digits and compute the similarity with the loaded signal to find the digit in the loaded audio file. The process involves computing sum of products of each DTMF signal with the loaded signal as demonstrated above. The maximum value of sum will belong to the DTMF signal which will be present in the loaded signal.

In [ ]:

```
# Generate sin waves of DTMF frequencies.
???

# Generate the dtmf signals for all the digits from 0 to 9
# dtmf signals
dtmf_digit0 = ?
dtmf_digit1 = ?
dtmf_digit2 = ?
dtmf_digit3 = ?
dtmf_digit4 = ?
dtmf_digit5 = ?
dtmf_digit6 = ?
dtmf_digit7 = ?
dtmf_digit8 = ?
dtmf_digit9 = ?



dtmf_digit_signals = [dtmf_digit0, dtmf_digit1, dtmf_digit2, dtmf_digit3, dtmf_di

# Create a list 'value' containing the sums (as done in cell no. 7). Find the max

????


# The digit in the loaded_signal is

????
```