**Q4. Solve the following system of three equations in three unknown:**

$x^2 + y^2 = 1$
$xy + yz = -1.1$
$y^2 + z^2 = 2$
A reasonsable guess for all variables is $x = y = z = 2$.

In [20]:

```
def f(xyz):
    x=xyz[0]
    y=xyz[1]
    z=xyz[2]

    f0=x**2+y**2-1
    f1=x*y+y*z+1.1
    f2=y**2+z**2-2

    return np.array([f0,f1,f1])

xyz0 = np.array([2.0,2,2])
xyz0
```

Out[20]:

```
array([2., 2., 2.])
```

# 10th September

# - Sequence

In [2]:

```
from sympy import SeqFormula, Symbol
n=Symbol('n')
s=SeqFormula(n**2,(n,0,5))
s.formula
```

Out[2]:

$n^2$

In [3]:

```
s.coeff(3)
```

Out[3]:

$9$

**supports slicing**

In [5]:

```
s[:]
```

Out[5]:

```
[0, 1, 4, 9, 16, 25]
```

In [6]:

```
s[1:4]
```

Out[6]:

```
[1, 4, 9]
```

**Sequence elements are displayed using list() command**

In [21]:

```
list(s)
```

Out[21]:

```
[0, 1, 4, 9, 16, 25]
```

# Lets discuss on few types of sequence in python

**1)Strings**

A string is a group of characters

**To declare an empty, use function str()**

**- declare a string**

In [24]:

```
name=str('Ann')
name
```

Out[24]:

```
'Ann'
```

**- Access a letter from the string**

In [25]:

```
name[2]
```

Out[25]:

```
'n'
```

**2)Lists**

A list is an ordered group of items. To declare it,we use square brackets.

#### Declare a list

In [26]:

```
groceries=['milk','bread','eggs']
groceries
```

Out[26]:

```
['milk', 'bread', 'eggs']
```

#### - Accessa element in the list

In [27]:

```
groceries[2]
```

Out[27]:

```
'eggs'
```

In [30]:

```
#slicing is possible
groceries[:2]
```

Out[30]:

```
['milk', 'bread']
```

#### Python list can hold all kinda ofitems,this is what makes it heteregoneous.

In [29]:

```
mylist=[1,'2',3.0,False]
mylist
#list is mutable we can change a value.
```

Out[29]:

```
[1, '2', 3.0, False]
```

#### 3)Tuples

- A tuple is an immutable group of item.
- we cant change a single value once we declare it.

-Declare a tuple

In [32]:

```
name=('Arushi','Kiran')
name
```

Out[32]:

```
('Arushi', 'Kiran')
```

We can also use the function tuple()

**4)Range**

A range()object lends us a range to iterate on;it gives us a list of numbers.

Syntax: range(start,stop,step)

In [34]:

```
for i in range(10,0,-3):
    print(i)
```

```
10
7
4
1
```

# 13th September

Q1.Program to generate sequence 3,5,7,9,11...,19 using range() command.

In [3]:

```
for n in range(1,10):
    print(2*n+1)
```

```
3
5
7
9
11
13
15
17
19
```

# Sequence Operations

**1.Concatenation**

The operator + is used to concatenate second element to the first.

In [5]:

```
[1,3,4]+[3,4,5]
```

Out[5]:

```
[1, 3, 4, 3, 4, 5]
```

In [6]:

```
['a','e','r']+['xd']
```

Out[6]:

```
['a', 'e', 'r', 'xd']
```

- Concatenate possible with oly same type of sequences

In [8]:

```
(1,2,3)+(4,5)
```

Out[8]:

```
(1, 2, 3, 4, 5)
```

**2.Repeat**

The operator *is used to repeat a sequence n number of times

In [10]:

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

Out[10]:

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

In [11]:

```
's'*3
```

Out[11]:

```
'sss'
```

In [13]:

```
'2'*3
```

Out[13]:

```
'222'
```

**3.Membership**

Membership operators **in** and **not in** are used to check whether an item is preset in a sequence or not.

Return Boolean values

In [15]:

```
var=str('Daisy')
'Da' in var
```

Out[15]:

```
True
```

In [16]:

```
var=str('Daisy')
'Da'  not in var
```

Out[16]:

```
False
```

# Sequence Functions

### 1.Len()

- length of sequence
- also counts space

In [18]:

```
d=[26,4,5,'F','abcd']
len(d)
```

Out[18]:

```
5
```

In [21]:

```
d='kfhhufgoh'
len(d)
```

Out[21]:

```
9
```

In [22]:

```
d=str('meow is the cat')
len(d)
```

Out[22]:

```
15
```

### 2.Min() and Max()

In [24]:

```
print(min([4,5,6,7]))
print(max([3,4,5,6]))
```

```
4
6
```

### 3.Index()

- searches an element and returns the index of the first occurence

In [26]:

```
var=str('athlete')
var.index('h')
```

Out[26]:

```
2
```

In [27]:

```
var.index('e') #e occurs twice but only first occurence is obtained.
```

Out[27]:

```
4
```

### 4.Count()

- counts number of times an element has occured in the sequence

In [28]:

```
var=str('athlete')
var.count('e')
```

Out[28]:

```
2
```

In [ ]:

```

```