Q.1) Print first 10 natural numbers using while loop.

```
→ for i in range(1,11):
      print (i)
```

Q.2) Print the following pattern:

```
1 1 1 1
 1 1 1
  1 1
   1
```

Answer:-

```
def pattern(n):
    for i in range(n-1):
        for i in range(i+1):
            # printing spaces
            print(" ", end=" ")
        for k in range(n-i-1):
            # printing stars
            print("1", end=" ")
        print().
#n=4)   n=int(input("Enter the number
                     of rows:"))
pattern(n).
```

Q.3) Calculate the sum of all numbers from 1 to a given number.

```
⇒ a= int(input("Enter a given number"))
   sum=0
   for i in range(1,a):
       sum +=i
       print (sum).
```

Q.4) Write a program to print multiplication table of a given number.

```
⇒s=int (input ("Enter a given number"))
  for i in range(1,11):
      print (s*, i, "=" ,s*i)
```

Q.5) Display numbers from a list using loop.

```
list = ["apple","banana","Red","Pink"]
# print (list)
for i in range (list)
print (list).
```

Q.6) Count the total number of dig in a number.

```
→ n= 12345.
  digits = ~~Str~~(n)
  digits_count = len(digits)
  print("Total numbers of digits:",dig
```

Q.7) Print the following pattern: Diam Shape

```
⇒ Height =5
  for i in range(1,height+1):
      spaces = " " *(height-i)
      stars = "*" *(2*i-1)
      print(spaces + stars).
  for i in range (height-1,0,-1):
      spaces = " " *height-i)
      stars = "*"*(2*i-1)
      print (spaces+ stars).
```

Q.8) Print list in reverse order using a loop.

```
→ list =[1,2,3,4,5].
  for item in reversed(list):
      print (item)
```

Q.9) Display numbers from -10 to-1 using a loop.

```
⇒ for number in range(-10,0):
      print (number)
```

Q.10) Use else block to display a message "Done" after successful execution of for loop.

```
→ for number in range(-10,0):
      print (number)
  else:
      print ("Done").
```

Q.11) Write a program to display a message all prime numbers in within a range.

```
→ def is_prime (num):
      if num<=1 :
           return false
      elif num<=3
           return True
      elif num%2 ==0ornum%3==0
           return false
      i=5
      while i*i <=num:
        if num%i==0 ornum%(i+2)
                              ==0 :
           return false
        i+=6.
      return True.
s = int(input("Enter a start of the range:"))
e = int(input("Enter a end of the range:"))
print(f"Prime numbers between {s} and
                                   {e} :"))
for number in range (start,e+1):
    if is_prime(number):
         print (number).
```

Q.12). Display fibonacci series upto10 terms.

```
→ a,b = 0,1
  print("fibonacci Series(first10terms):")
  for _ in range(10).
     print(a,end="")
     a,b= b, a+b .
```

Q.13). find the factorial of agiven number.

```
n = int(input ("Enter a number:")).
f= 1
if n<0:
   print("factorial is not defined
                for negative numbers"))
elif n==0:
   print("The factorial of 0 is 1.")
else:
   for iin range(1,num+1):
     #factorial
       f *=i
   print(f"The factorial of {n}is
                              {f}")
```

14) Reverse in given integer number

```
→ N=int(input("Enter aninteger number:"))
  reversed number=int (str (number)[::-1]
  print("Reversed number:",reversed_num
```

15).Use a loop to display elements from a given list present at odd index positions

```
list = [10,20,30,4050,60,70,80,90]
print("Elements at odd index positions
for i in range(len(list));
      if i %2]=0:
           print(list[i]).
```

16) Calculate the cube of all numbers from 1 to a given number.

```
→ e_num=int(input("Enter anumber:
  print("Cubes of numbers from 1 to",e_num
  for i in range(1,e_num+1):
        cube =i**3
        print(f"{i} cubed is {cube}"
```

17). find the sum of the series upto n terms.

```
n= int(input("Enter the number of te
                                    (n):"
a=float(input("Enter the first term(a):"))
d =float (input("Enter the common differe
                    ce (d):"))
S_sum= (n/2)*(2*a+ (n-1)*d).
print(f"The sum of the series is: {s_sum
                                     "}.
```

Q.18) Print the following pattern

→ def pattern_number(n):

```
    for i in range(1,n+1):
            print("",end=" ")

     for j in range(1,i+1):
            print(j, end=" ")
     print(" ")

n=5
print("Pattern_number:")
pattern_number(n)
```

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
```

1) Calculate the Z score for the below data set assume sd = 1.5
How do U perform normalization (only formula).

data
2
3
1
3
2
4.

Answer → Z score formula.

$Z = \frac{x - \mu}{\sigma}$

where Z is the Z score.
X is individual data point
$\mu$ is mean of data set
$\sigma$ is the standard deviation of the data set.

Here $\sigma = 1.5$

to calculate Z score

$\mu = \frac{2+3+1+3+2+4}{6} = \frac{15}{6} = 2.5$

$Z = \frac{X - \mu}{\sigma}$

for each data point, substitute the values into the formula.

for X=2: $Z = \frac{2-2.5}{1.5} = -\frac{0.5}{1.5} = -0.333$

for X=3 $Z = \frac{3-2.5}{1.5} = \frac{.5}{1.5} = 0.333$

for X=1 $Z = \frac{1-2.5}{1.5} = \frac{-1.5}{1.5} = -1$

for X=3 : $Z = \frac{3-2.5}{1.5} = 0.333$

for X=2 $Z = \frac{2-2.5}{1.5} = -0.333$

for x = 4 $Z = \frac{4-2.5}{1.5} = \frac{1.5}{1.5} = 1$

So the Z-scores for data set are approximately:

-0.333, 0333, -1, 0.333, -0.333, 1.

2) What is One Hot encoding? Name the pandas function which perform OHE.

→ One Hot encoding is a technique used in machine learning and data preprocessing to represent categorical variables as binary vectors.

One Hot ecoding is used to quantify categorical data.

One Hot encoding can be implemented with pandas using

One Hot encoding get_dummies function take following parameters.

data: array-like, Series, or Dataframe -data containing categorical variables of which to get dummy indicators.

3) List all the transformers (function and power)?

→ 1) Function Transformer 2) Power transformer.

* Function transformer :
  * Function Transformer
    1) log Transformer
    2) Reciprocal Transformer
    3) Square Transformer.
    4) Square root transformer.
    5) Custom Transformer.
* Power Transformer
  1) Box Cos
  2) Yeo Johnson.

4) Explain all the assumptions of linear regression in 2 lines?

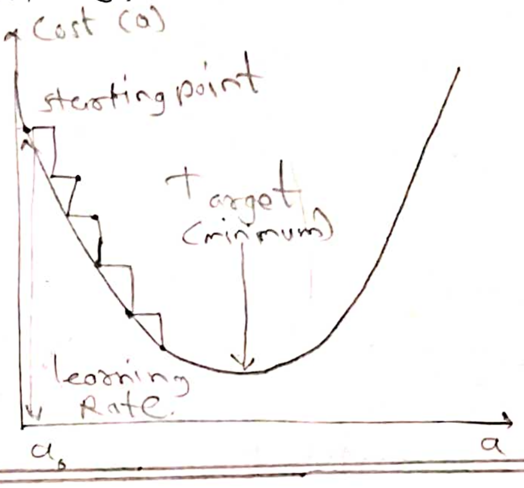
→ 1) linearity : Assumes a linear relationship between the independent and dependent variable, implying that the change in mean of dependent variable is proportional to a change in the independent variable.

2) Indepedance and Homoscedasticity: Assumes that error are independent and constant variance across all levels of indepedent variable ensuring the reliability of model predictions.

3) Multivariate Normality ÷ Normality of error distribution
4) Lack of multicollinearity ÷ (Predictors are not correlated with each other).
5) The outlier check.

Q.5) What is gradient descent algorithm? Explain with diagram

→ Gradient Descent is an optimization algorithm that is used to train machine learning models. The algorithm minimize the loss Function that measures the error between predicted values & actual values. The algorithm iteratively adjusts the model's parameters in the direction of the stepest descrease in the loss function.



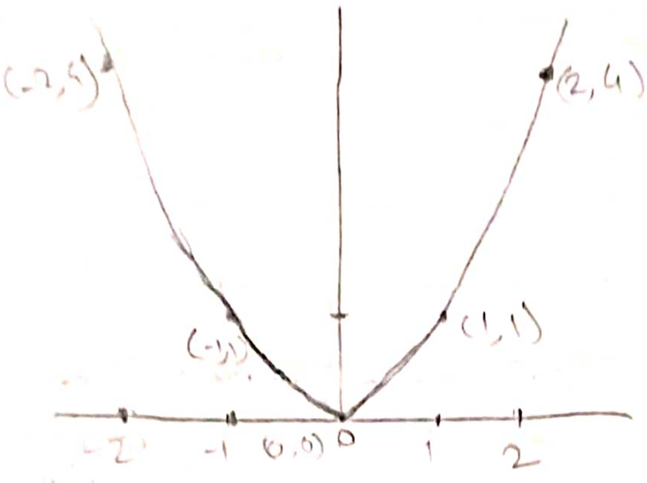
Q.6) What is pandas profilling? Write the suittable sytax.

→ Pandas profilling is a python library that performs an automated Exploratory Data analysis (EDA). It automatically generates a dataset profile report that gives valuable insights. The report is generated in an HTML format, which makes it easy to visualize.

```
from pandas_profilling.
profile = profile Report (df)
profile.to_file (index.html).
```

7) Draw the line for the following equation

y=x^2



8) Build the regression model for the "mpg" data set (Present in seaborn library).

import all the necessary libraries.
import dataset from seaborn library
check for missing value
split the data into train and test
fill the model
predict the model.

→ i) Import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from ~~[illegible]~~ sklearn.model_selection import train_test_split
from sklearn.linear_model import linearRegression

ii) df = sns.load_dataset ('mpg')

iii) (df.isnull().sum())
df.isnull().sum()/(len(df)))*100

iv) x= df.features
y = df.target
x_train, xtest, y_train, y_test = train_test split (x, y, test_size =0.2, random_state =42)

v) model = linearRegression()
model. fit (x_train, y_train)

vi) Y_pred = model.predict (x_test)

~~y~~ y_pred.