**PS NO = 99004970**

**Python programs (part 1)**

**Q2.**

# Python3 program to find Closest number in a list

def closest(lst, K):

    return lst[min(range(len(lst)), key = lambda i: abs(lst[i]-K))]

# Driver code

lst = [3.64, 5.2, 9.42, 9.35, 8.5, 8]

K = 9.1

print(closest(lst, K))

**Q3.**

# Python3 Program to calculate speed,

# distance and time

# Function to calculate speed

def cal\_speed(dist, time):

    print(" Distance(km) :", dist);

    print(" Time(hr) :", time);

    return dist / time;

# Function to calculate distance traveled

def cal\_dis(speed, time):

    print(" Time(hr) :", time) ;

    print(" Speed(km / hr) :", speed);

    return speed \* time;

# Function to calculate time taken

def cal\_time(dist, speed):

    print(" Distance(km) :", dist);

    print(" Speed(km / hr) :", speed);

    return speed \* dist;

# Driver Code

# Calling function cal\_speed()

print(" The calculated Speed(km / hr) is :",

                     cal\_speed(45.9, 2.0 ));

print("");

# Calling function cal\_dis()

print(" The calculated Distance(km) :",

                   cal\_dis(62.9, 2.5));

print("");

# Calling function cal\_time()

print(" The calculated Time(hr) :",

              cal\_time(48.0, 4.5));

**Q5.**

# getMissingNo takes list as argument

def getMissingNo(A):

    n = len(A)

    total = (n + 1)\*(n + 2)/2

    sum\_of\_A = sum(A)

    return total - sum\_of\_A

# Driver program to test the above function

A = [1, 2, 4, 5, 6]

miss = getMissingNo(A)

print(miss)

**Q6.**

class Solution:

       def solve(self, L1, L2):

      L1.sort()

      L2.sort()

      ans = float("inf")

      i = j = 0

      while i < len(L1) and j < len(L2):

         ans = min(ans, abs(L1[i] - L2[j]))

         if L1[i] < L2[j]:

            i += 1

         else:

            j += 1

      return ans

ob = Solution()

L1 = [2, 7, 4] L2 = [16, 10, 11]

print(ob.solve(L1, L2))

**Q7.**

# find number of elements > k using for loop

# initializing list

test\_list = [1, 7, 5, 6, 3, 8]

# initializing k

k = 4

# printing list

print ("The list : " + str(test\_list))

# using for loop to get numbers > k

count = 0

for i in test\_list :

   if i > k :

      count = count + 1

# printing the intersection

print ("The numbers greater than 4 : " + str(count))

# find number of elements > k using list comprehension

# initializing list

test\_list = [1, 7, 5, 6, 3, 8]

# initializing k

k = 4

# printing list

print ("The list : " + str(test\_list))

# using list comprehension to get numbers > k

count = len([i for i in test\_list if i > k])

# printing the intersection

print ("The numbers greater than 4 : " + str(count))

# find number of elements > k using sum()

# initializing list

test\_list = [1, 7, 5, 6, 3, 8]

# initializing k

k = 4

# printing list

print ("The list : " + str(test\_list))

# using sum() to get numbers > k

count = sum(i > k for i in test\_list)

# printing the intersection

print ("The numbers greater than 4 : " + str(count))

**Python programs (part 2)**

Q3. Convert IP address

|  |
| --- |
|  |

# importing the module

import ipaddress

# converting IPv4 address to int

addr1 = ipaddress.ip\_address('191.255.254.40')

addr2 = ipaddress.ip\_address('0.0.0.123')

print(int(addr1))

print(int(addr2))

# converting IPv6 address to int

addr3 = ipaddress.ip\_address('2001:db7:dc75:365:220a:7c84:d796:6401')

print(int(addr3))

|  |
| --- |
| Q4.  # Python program to check  # if a word is isogram or not  def is\_isogram(word):        # Convert the word or sentence in lower case letters.      clean\_word = word.lower()        # Make an empty list to append unique letters      letter\_list = []        for letter in clean\_word:            # If letter is an alphabet then only check          if letter.isalpha():              if letter in letter\_list:                  return False              letter\_list.append(letter)        return True    if \_\_name\_\_ == '\_\_main\_\_':      print(is\_isogram("Machine"))      print(is\_isogram("isogram"))      print(is\_isogram("GeeksforGeeks"))      print(is\_isogram("Alphabet "))  Q5.  s='hello'  new=[]  for i, val in enumerate(s[:]):      up=s[i].upper()      c=s[:i] + up + s[i+1:]      new.append(c)  print(new)  Q6.  # Python program to implement  # the above approach    def maxnumber(n, k):  # Function to return the  # largest number possible        for i in range(0, k):          # Generate the largest number          # after removal of the least K digits          # one by one          ans = 0          i = 1            while n // i > 0:          # Remove the least digit          # after every iteration              temp = (n//(i \* 10))\*i + (n % i)              i \*= 10          # Store the numbers formed after          # removing every digit once            # Compare and store the maximum              if temp > ans:                  ans = temp          n = ans        # Return the remaining number      # after K removals      return ans;      n = 6358  k = 1  print(maxnumber(n, k))  Q7.  # Python program to print the maximum number  # from the set of digits of a given number    # Function to print maximum number  def printMaximum(inum):        # Hashed array to store count of digits      count = [0 for x in range(10)]        # Connverting given number to string      string = str(num)        # Updating the count array      for i in range(len(string)):          count[int(string[i])] = count[int(string[i])] +  1        # Result stores final number      result = 0      multiplier = 1        # traversing the count array      # to calculate the maximum number        for i in range(10):          while count[i] > 0:              result = result + ( i \* multiplier )              count[i] = count[i] - 1              multiplier = multiplier \* 10        # return the result      return result    # Driver code  num = 38293367  print (printMaximum(num))    Q8  # Python3 code to find frequency of each word  # function for calculating the frequency  def freq(str):        # break the string into list of words      str\_list = str.split()        # gives set of unique words      unique\_words = set(str\_list)        for words in unique\_words :          print('Frequency of ', words , 'is :', str\_list.count(words))    # driver code  if \_\_name\_\_ == "\_\_main\_\_":        str ='apple mango apple orange orange apple guava mango mango'        # calling the freq function      freq(str)  Q9.  import matplotlib  print(matplotlib.colors.to\_hex([ 0.47, 0.0, 1.0 ]))  print(matplotlib.colors.to\_hex([ 0.7, 0.321, 0.3, 0.5 ], keep\_alpha=True))  print(matplotlib.colors.to\_rgb("#aabbcc"))  print(matplotlib.colors.to\_rgb("#ddee9f")) |