SET FUNCTION

```
my_set = {1, 3, 4, 5, 6}
print(my_set)

# ADD
my_set.add(2)
print("ADD VALUE : ",my_set)

# update
my_set.update([6,7,8,9])
print("UPDATE",my_set)

# discard
my_set.discard(4)
print("DISCARD",my_set)
# remove
my_set.remove(6)
print("REMOVE",my_set)
my_set.remove(2)
print(my_set)
# discard
my_set.discard(2)
print(my_set)

# POP
my_set.pop()
print("POP",my_set)
my_set.pop()
print("POP",my_set)

# COPY
set1 = my_set.copy()
print("COPY",set1)


# CLEAR
print("CLEAR",set1.clear())


# SET OPERATION
A = {1, 2, 3, 4, 5}
B = {4, 5, 6, 7, 8}

C = {1, 2, 3, 4, 5}
D = {4, 5, 6, 7, 8}
```

```
E = {6,7,8,9,10}


# UNION
print("UNION")
print(A | B)
print(A.union(B))
print(B.union(A))

# Intersection
print("Intersection")
print(A & B)
print(A.intersection(B))
print(B.intersection(A))

# Intersection_update
C.intersection_update(D)
print("intersection_update",C)
D.intersection_update(C)
print("intersection_update",D)

# isdisjoint()
print("C TO D isdisjoint()",C.isdisjoint(D))
print("C TO E isdisjoint()",C.isdisjoint(E))

X = {1, 2, 3, 4, 5}
Y = {1,2,3}
# issubset()
print("SUB SET ",X.issubset(Y))
print("SUB SET ",Y.issubset(X))

# issuperset()
print("SUPER SET ",X.issubset(Y))
print("SUPER SET ",Y.issubset(X))


#Difference
print("Difference")
print(A - B)
A.difference(B)
print(B - A)
B.difference(A)

# Difference_update
print("C ",C)
print("D",D)
```

```
C.difference_update(D)
print("After C set\ndifference_update",C)
D.difference_update(C)
print("After D set\ndifference_update",D)


#Symmetric Difference
print("Symmetric Difference")
print(A ^ B)
print(A.symmetric_difference(B))
print(B.symmetric_difference(A))


Z= {6,8,1,3,9,10,4,7,5,2}
print("Z",Z)
print("ALL",all(Z))
print("ANY",any(Z))

Z1={}
print("ALL",all(Z1))
print("ANY",any(Z1))

print("SORTED", sorted(Z))
print("enumerate",type(enumerate(Z)))
print("LEN",len(Z))
print("MAX",max(Z))
print("MIN",min(Z))
print("SUM",sum(Z))
```

frozenset

```
A= frozenset([1, 2, 3, 4])
B = frozenset([3, 4, 5, 6])
print(A.isdisjoint(B))
print(A.difference(B))
print(A | B)
A.add(3) # Error
```

FILE HANDLING

P-1

```
fo = open("student.txt", "wb")
print ("Name of the file: ", fo.name)
print ("Closed or not : ", fo.closed)
print ("Opening mode : ", fo.mode)
fo.close()
```

P-2

```
file = open("e://student.txt","w")
str1="THIS IS FIRST PROGRAM TO FILE HANDLING"
file.write(str1)
file.close()
```

P-3

```
file = open("e://student.txt", "r")        # Open a file
for i in file :
   print(i)
file.close()
```

P-4

```
file = open("e://student.txt", "r")
str1 = file.read(30)
print ("Read String is : ", str1)
file.close()
```

P-4

```
file = open("e://student.txt", "r")        # Open a file
print(file.readline())
print(file.readline())
print(file.readline())
file.close()
```

P-5

```
file = open("e://student.txt", "r")        # Open a file
str = file.read(30)
print ("Read String is : ", str)

position = file.tell()                          # Check current position
print ("Current file position : ", position)
```

```
position = file.seek(0, 0)                    # Reposition pointer at the beginning once again
str = file.read(15)
print ("Again read String is : ", str)

file.close() # Close opened file
```

Function AS Object

```
def fact(n):
    """Assumes that n is an int > 0 Returns n!"""
    if n == 1:
        return n
    else:
        return n*fact(n - 1)

def apply(L, f):
    """Assumes L is a list, f a function Mutates L by replacing each element, e, of L by f(e)"""
    #print(type(f))
    for i in range(len(L)):
            L[i] = f(L[i])


L = [1, -4, 6.66]
print('L =', L)
print ('Apply abs to each element of L.')
apply(L, abs)
print('L =', L)

print ('Apply int to each element of', L)
apply(L, int)
print ('L =', L)

print ('Apply factorial to each element of', L)
apply(L, fact)
print ('L =', L)
```

MODULE -1

HEllO.py (File Save)	

def print_v(name):
  print("Hello", name)
  return
	import Hello # Hello File Import
	Hello.print_v("SANDIP")

MODULE -2

Number.py (File SAVE)	

def armstrong(n):
  sum1=0
  d=0
  ex=n
  while (n!=0):
  d=n%10
  sum1=sum1+(d*d*d)
  n=int(n/10)
 	
  if(ex==sum1):
  print("Number Is Artmstromg")
  else:
  print("Number is Not Armstyrong")
  return
 	
def digitsum(n):
  sum2=0
  d=0
  while (n!=0):
  d=n%10
  sum2=sum2+d
  n=int(n/10)
 	
  print(" Sum of Digit is %d" %sum2)
  return
	import number as n1 # import number
	n = int(input("Enter Number"))
	n1.armstrong(n)
	n1.digitsum(n)