|  |  |
| --- | --- |
| **Ex.No:7** | **Merge Sort** |
|  |

***Aim:***

To develop a python program to perform sorting in a list using merge sort algorithm.

***Main Algorithm:***

|  |  |
| --- | --- |
| **Step 1:** | Start Process |
| **Step 2:** | Get a list of numbers from user and store id alist. |
| **Step 3:** | Print “Unsorted list” as alist |
| **Step 4:** | Call Merge Sort Algorithm with alist |
| **Step 5:** | Print “Sorted List” as alist |
| **Step 6:** | Stop Process |

***Merge Sort Algorithm:***

|  |  |
| --- | --- |
| **Step 1:** | Start Process |
| **Step 2:** | Receive alist |
| **Step 3:** | If length of list is less than 2 then return alist |
| **Step 4:** | Else compute mid = length of alist/2 and goto Step 5 |
| **Step 5:** | Split alist into two half’s as left sub list (l\_alist) and right sub list (r\_alist) |
| **Step 6:** | Call Merge Sort Algorithm with l\_list and assign to x |
| **Step 7:** | Call Merge Sort Algorithm with r\_ list and assign to y |
| **Step 8:** | Create empty list called out |
| **Step 9:** | Assign i and j as zero |
| **Step 10:** | If i is less than length of x and j is less than length of y then goto Step 12 |
| **Step 11:** | Else goto Step 15 |
| **Step 12** | If x[i] is greater than y[j] then append y[j] in out list and increment j |
| **Step 13:** | Else append x[i] in out list and increment i |
| **Step 14:** | Goto Step 10 |
| **Step 15:** | Append remaining elements of x list from i the position to out list |
| **Step 16:** | Append remaining elements of y list from j the position to out list |
| **Step 17:** | Return out list |
| **Step 18:** | Stop Process |

**Flow Chart:**

**Pseudo Code Merge Sort:**

START

RECIVE alist

IF length of alist < 2 THEN

RETURN alist

END IF

mid = length of alist / 2

l\_alist = alist [0, mid – 1]

r\_alist = alist [mid, end]

x = CALL Merge Sort with l\_alist

y = CALL Merge Sort with r\_alist

out = empty list

ASSIGN i as 0

ASSIGN j as 0

WHILE i < length of x and j < length of y THEN

IF x[i] > y[j]

ADD y[j] to out

INCREMENT j

ELSE

ADD x[i] to out

INCREMENT i

END IF

END WHILE

ADD x[i, end] to out

ADD y[j, end] to out

RETURN out

STOP

**Pseudo Code Main Algorithm:**

START

READ alist

PRINT “Unsorted List”, alist

alist = CALL Merge Sort with alist

PRINT “Sorted List”, alist

STOP

**Program:**

def mergesort(alist):

if (len(alist)<2):

return alist

mid = int(len(alist)/2)

x = mergesort(alist[:mid])

y = mergesort(alist[mid:])

out = []

i = 0

j = 0

while i<len(x) and j<len(y):

if(x[i]>y[j]):

out.append(y[j])

j+=1

else:

out.append(x[i])

i+=1

out = out + x[i:]

out = out + y[j:]

return out

print("Merge Sort")

print("----------")

alist = []

print ("enter any 5 numbers")

for i in range(5):

data = int(input())

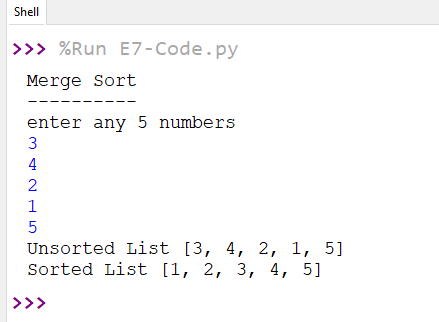
alist.append(data)

print("Unsorted List",alist)

alist = mergesort(alist)

print("Sorted List",alist)

**Output:**

****

***Result:***

Thus the program to perform sorting in a list using merge sort algorithm was developed and tested successfully.