1. Python Program for Topological Sorting

from collections import defaultdict

class Graph:

def \_\_init\_\_(self, vertices):

self.graph = defaultdict(list)

self.V = vertices

def addEdge(self, u, v):

self.graph[u].append(v)

def topologicalSortUtil(self, v, visited, stack):

visited[v] = True

for i in self.graph[v]:

if visited[i] == False:

self.topologicalSortUtil(i, visited, stack)

stack.insert(0, v)

def topologicalSort(self):

visited = [False] \* (self.V)

stack = []

for i in range(self.V):

if visited[i] == False:

self.topologicalSortUtil(i, visited, stack)

print("Topological Sorting:")

for i in stack:

print(i, end=" ")

# Example usage

g = Graph(6)

g.addEdge(5, 2)

g.addEdge(5, 0)

g.addEdge(4, 0)

g.addEdge(4, 1)

g.addEdge(2, 3)

g.addEdge(3, 1)

g.topologicalSort()

1. Python Program for Radix Sort

def countingSort(arr, exp):

n = len(arr)

output = [0] \* n

count = [0] \* 10

for i in range(n):

index = arr[i] // exp

count[index % 10] += 1

for i in range(1, 10):

count[i] += count[i - 1]

i = n - 1

while i >= 0:

index = arr[i] // exp

output[count[index % 10] - 1] = arr[i]

count[index % 10] -= 1

i -= 1

for i in range(n):

arr[i] = output[i]

def radixSort(arr):

max\_val = max(arr)

exp = 1

while max\_val // exp > 0:

countingSort(arr, exp)

exp \*= 10

# Example usage

my\_array = [170, 45, 75, 90, 802, 24, 2, 66]

radixSort(my\_array)

print(my\_array)

1. Python Program for Binary Insertion Sort

def countingSort(arr, exp):

n = len(arr)

output = [0] \* n

count = [0] \* 10

for i in range(n):

index = arr[i] // exp

count[index % 10] += 1

for i in range(1, 10):

count[i] += count[i - 1]

i = n - 1

while i >= 0:

index = arr[i] // exp

output[count[index % 10] - 1] = arr[i]

count[index % 10] -= 1

i -= 1

for i in range(n):

arr[i] = output[i]

def radixSort(arr):

max\_val = max(arr)

exp = 1

while max\_val // exp > 0:

countingSort(arr, exp)

exp \*= 10

# Example usage

my\_array = [170, 45, 75, 90, 802, 24, 2, 66]

radixSort(my\_array)

print(my\_array)

1. Python Program for Bitonic Sort

def compAndSwap(arr, i, j, direction):

if (arr[i] > arr[j] and direction == 1) or (arr[i] < arr[j] and direction == 0):

arr[i], arr[j] = arr[j], arr[i]

def bitonicMerge(arr, low, count, direction):

if count > 1:

k = count // 2

for i in range(low, low + k):

compAndSwap(arr, i, i + k, direction)

bitonicMerge(arr, low, k, direction)

bitonicMerge(arr, low + k, k, direction)

def bitonicSort(arr, low, count, direction):

if count > 1:

k = count // 2

bitonicSort(arr, low, k, 1)

bitonicSort(arr, low + k, k, 0)

bitonicMerge(arr, low, count, direction)

def sort(arr):

1. Python Program for Comb Sort

def getNextGap(gap):

gap = (gap \* 10) // 13

if gap < 1:

return 1

return gap

def combSort(arr):

n = len(arr)

gap = n

swapped = True

while gap != 1 or swapped:

gap = getNextGap(gap)

swapped = False

for i in range(0, n - gap):

if arr[i] > arr[i + gap]:

arr[i], arr[i + gap] = arr[i + gap], arr[i]

swapped = True

# Example usage

my\_array = [64, 34, 25, 12, 22, 11, 90]

combSort(my\_array)

print(my\_array)

1. Python Program for Pigeonhole Sort
2. Python Program for Cocktail Sort
3. Python Program for Gnome Sort
4. Python Program for Odd-Even Sort / Brick Sort
5. Python Program for BogoSort or Permutation Sort