CSA0676 – DAA

Chapter – 2 .

1. def handle\_empty\_list():

user\_input = input("Enter elements separated by spaces: ")

input\_list = user\_input.split()

if not input\_list:

print("Empty list entered.")

else:

print("List with elements:", input\_list)

handle\_empty\_list()

**2. selection sort**

def selection\_sort(arr):

n = len(arr)

for i in range(n):

min\_idx = i

for j in range(i+1, n):

if arr[j] < arr[min\_idx]:

min\_idx = j

arr[i], arr[min\_idx] = arr[min\_idx], arr[i]

return arr

arr = []

for i in range(6):

arr.append(int(input()))

print("Input Array:", arr)

print("Sorted Array:", selection\_sort(arr))

**3. bubble sort**

def bubble\_sort(arr):

n = len(arr)

for i in range(n):

already\_sorted = True

for j in range(n - i - 1):

if arr[j] > arr[j + 1]:

arr[j], arr[j + 1] = arr[j + 1], arr[j]

already\_sorted = False

if already\_sorted:

break

return arr

user\_input = input("Enter a list of numbers separated by spaces: ")

arr = list(map(int, user\_input.split()))

sorted\_arr = bubble\_sort(arr)

print("Sorted Array:", sorted\_arr)

**4. insertion sort**

def insertion\_sort(arr):

for i in range(1, len(arr)):

key = arr[i]

j = i - 1

while j >= 0 and key < arr[j]:

arr[j + 1] = arr[j]

j -= 1

arr[j + 1] = key

arr = list(map(int, input("Enter elements of the array separated by spaces: ").split()))

insertion\_sort(arr)

print("Sorted array with duplicate elements handled:", arr)

**5.kth positive integer**

def find\_kth\_missing(arr, k):

missing\_set = set(range(1, arr[-1] + k + 1)) - set(arr)

return sorted(missing\_set)[k - 1]

# Input from the user

arr = list(map(int, input("Enter the sorted array of positive integers: ").split()))

k = int(input("Enter the value of k: "))

result = find\_kth\_missing(arr, k)

print(f"The {k}th missing positive integer is: {result}")

**6. peak element**

def find\_peak\_element(nums):

left, right = 0, len(nums) - 1

while left < right:

mid = left + (right - left) // 2

if nums[mid] < nums[mid + 1]:

left = mid + 1

else:

right = mid

return left

# User input

nums = list(map(int, input("Enter the array elements separated by space: ").split()))

peak\_index = find\_peak\_element(nums)

print("Peak element index:", peak\_index)

**7. needle haystack.**

def find\_needle\_in\_haystack(needle, haystack):

try:

index = haystack.index(needle)

return index

except ValueError:

return -1

# Input from user

needle = input("Enter the needle string: ")

haystack = input("Enter the haystack string: ")

result = find\_needle\_in\_haystack(needle, haystack)

print("Index of the first occurrence of the needle in the haystack:", result)

**8. string words**

def find\_substrings(words):

substrings = [word for word in words if any(word in other\_word for other\_word in words if word != other\_word)]

return substrings

# Input from the user

words = input("Enter a list of words separated by spaces: ").split()

result = find\_substrings(words)

print("Substrings that are part of other words:", result)

**9. closest pair**

import math

def calculate\_distance(point1, point2):

return math.sqrt((point1[0] - point2[0])\*2 + (point1[1] - point2[1])\*2)

def closest\_pair(points):

pass

points = []

n = int(input("Enter the number of 2D points: "))

for \_ in range(n):

x, y = map(int, input("Enter x and y coordinates separated by space: ").split())

points.append((x, y))

closest\_points = closest\_pair(points)

print("Closest pair of points:", closest\_points)

11. convex hull.

import itertools

def orientation(p, q, r):

val = (q[1] - p[1]) \* (r[0] - q[0]) - (q[0] - p[0]) \* (r[1] - q[1])

if val == 0:

return 0

return 1 if val > 0 else -1

def convex\_hull(points):

n = len(points)

if n < 3:

return points

hull = []

for i in range(n):

for j in range(i+1, n):

count = 0

for k in range(n):

if k != i and k != j:

if orientation(points[i], points[j], points[k]) == -1:

count += 1

if count == n - 2:

if points[i] not in hull:

hull.append(points[i])

if points[j] not in hull:

hull.append(points[j])

return hull

# Input from the user

points = []

n = int(input("Enter the number of points: "))

for \_ in range(n):

x, y = map(int, input("Enter x and y coordinates separated by space: ").split())

points.append((x, y))

convex\_points = convex\_hull(points)

print("Convex Hull Points:", convex\_points)