QUESTION FORMAT: (ADD PLAG DECLAR & STAT OF COMPL. NO IMAGES OF CODE.)

**Question:**

**Source Code:**

**Sample Screen Dumps:**

**How this was tested:**

**Question: 1**

**Source Code:**

import random

arrA = []

sizeA = 300

arrB = []

sizeB = 400

for size in range(sizeA):

    arrA.append(random.randint(0,1024))

for size in range(sizeB):

    arrB.append(random.randint(0,1024))

temp = 0

print(f"Unsorted Array A: {arrA}")

print(f"Unsorted Array B: {arrB}")

# Shell Sort

gap = sizeA // 2

flag = True

while gap >= 1 and flag == True:

    flag = False

    for i in range(sizeA - gap):

        if arrA[i] > arrA[i + gap]:

            temp = arrA[i]

            arrA[i] = arrA[i + gap]

            arrA[i + gap] = temp

            flag = True

    if gap > 1:

        gap = gap // 2

# Quick Sort

def qSort(arr):

    quick(arr, 0, len(arr) - 1)

def quick(arr, first, last):

    if first < last:

        pivotPos = partition(arr, first, last)

        quick(arr, first, pivotPos - 1)

        quick(arr, pivotPos + 1, last)

def partition(arr, first, last):

    pivot = arr[first]

    u = first

    d = last

    while True:

        while u < last and arr[u] <= pivot:

            u += 1

        while arr[d] > pivot:

            d -= 1

        if u < d:

            temp = arr[u]

            arr[u] = arr[d]

            arr[d] = temp

        else:

            break

    temp = arr[first]

    arr[first] = arr[d]

    arr[d] = temp

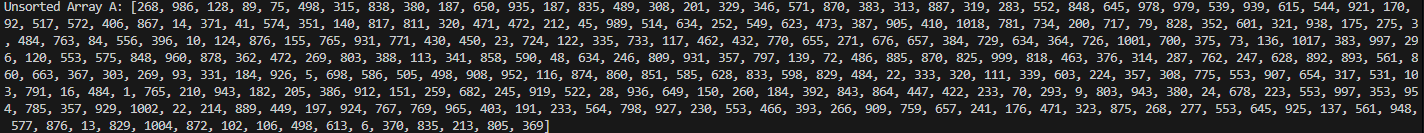
    return d

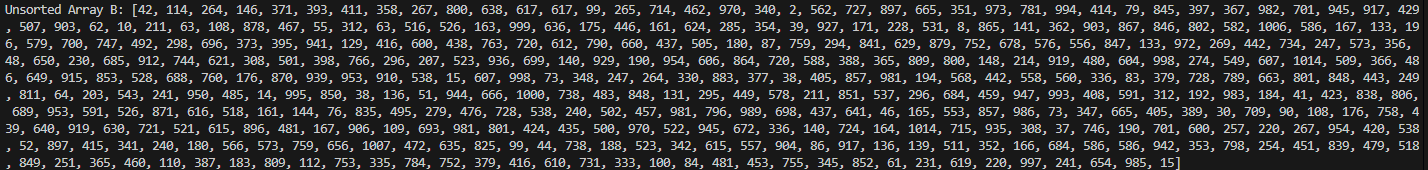
qSort(arrB)

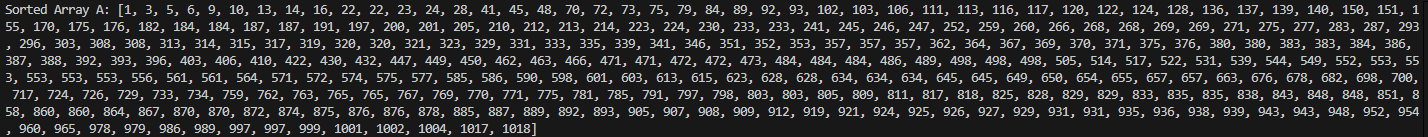
print(f"Sorted Array A: {arrA}")

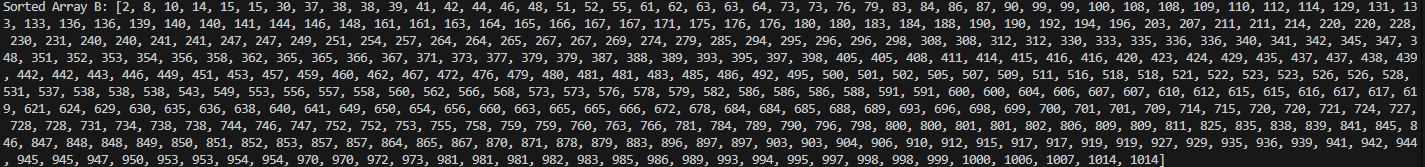
print(f"Sorted Array B: {arrB}")

**Sample Screen Dumps:**









**How this was tested:** The sizes of both arrays were initially set to smaller values to check the functionality of the code. Arrays of various sizes, including both odd and even, were tested to confirm that the program handled them properly. Additionally, both sorting algorithms were provided with an already sorted array to verify their correctness.

**Question:** 2

**Source Code (to be pasted after code in Question 1):**

arrC = []

ptr1 = 0

ptr2 = 0

while ptr1 < sizeA and ptr2 < sizeB:

    if arrA[ptr1] > arrB[ptr2]:

        arrC.append(arrB[ptr2])

        ptr2 += 1

    elif arrB[ptr2] > arrA[ptr1]:

        arrC.append(arrA[ptr1])

        ptr1 += 1

    else:

        arrC.append(arrA[ptr1])

        arrC.append(arrB[ptr2])

        ptr1 += 1

        ptr2 += 1

while ptr1 < sizeA:

    arrC.append(arrA[ptr1])

    ptr1 += 1

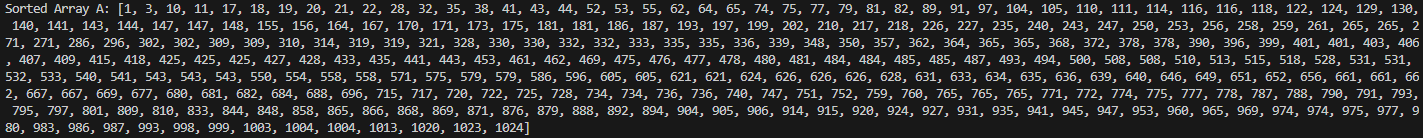
while ptr2 < sizeB:

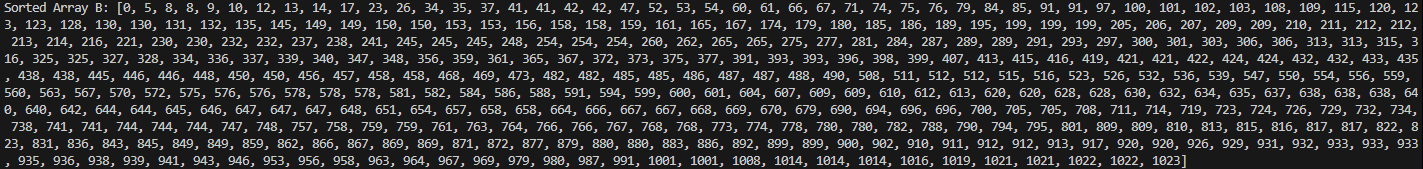
    arrC.append(arrB[ptr2])

    ptr2 += 1

print(f"Merged Array: {arrC}")

**Sample Screen Dumps:**

****

****

**A black screen with many small squares

AI-generated content may be incorrect.**

**How this was tested:** The sizes of both arrays were initially set to smaller values to check the functionality of the code. The lengths of arrays arrA, arrB and arrC were printed and as expected, the length of arrC was the combined lengths of arrays arrA and arrB.

**Question:** 3

**Source Code:**

import random

arr = []

eArr = [] # holds the extreme points

for i in range(10):

    arr.append(random.randint(0,20))

print(f"Unchecked array: {arr}")

def extremeCheck(arr):

    global epts

    epts = False

    for i in range(1, len(arr) - 1):

        if (arr[i] < arr[i-1] and arr[i] < arr[i+1]) or (arr[i] > arr[i-1] and arr[i] > arr[i+1]):

            eArr.append(arr[i])

            print(f"Value at index {i}: {arr[i]} is an extreme point")

            epts = True

extremeCheck(arr)

if epts:

    print(f"Line print of extreme points: {eArr}")

else:

    print("SORTED")

**Sample Screen Dumps:**

**A screenshot of a computer

AI-generated content may be incorrect.**

****

**How this was tested:** To test the extremeCheck function with an unsorted array, the code was simply run. As it’s highly unlikely for the random generated array to be sorted, a temporary sorted array was made and was passed to extremeCheck function.

**Do you agree that an array has no extreme points if and only if it is sorted? Explain your answer**

It is true that array has no extreme points if and only if it is sorted. If the array is sorted in ascending order, all elements (excluding first and last) will be larger than the previous element but smaller than the next element. Similarly, if the array is sorted in descending order, all elements (excluding first and last) will be smaller than the previous element but larger than the next element. Such conditions A[i-1] < A[i] < A[i+1] or A[i-1] > A[i] > A[i+1] does not comply with the stated conditions A[i− 1] < A[i] > A[i + 1] or A[i − 1] > A[i] < A[i + 1]