# Experiment- 1.4

**Student Name:** UTKARSH JOSHI
**UID:** 21BCS9158
**Branch:** CSE-Gen
**Section/Group:** ST 802-A
**Semester:** 5th
**Date of Performance:**24/08/23
**Subject Name:** Advanced Programming
**Subject Code:** 21CSP-314

## 1. Aim:

Solve the following problems on hackerrank:

1. Quicksort 1 – Partition
2. Closest Numbers

## 2. Objective: To perform Searching and Sorting.

## 3. Code :

### Program -1

```
#!/bin/python3

import math
import os
import random
import re
import sys
```

```
#
# Complete the 'quickSort' function below.
#
# The function is expected to return an INTEGER_ARRAY.
# The function accepts INTEGER_ARRAY arr as parameter.
#

def quickSort(arr):
    left = []
    right = []
    pivot  = arr[0]

    i = 0
    j= len(arr) - 1
    while i < j:
        #index i
        while i < j and arr[i] <= pivot:
            i += 1

        #index j
        while arr[j] > pivot:
            j -= 1

        if i < j:
            arr[i], arr[j] = arr[j], arr[i]
            i += 1
```

```
            j -= 1
        arr[0], arr[j] = arr[j], arr[0]

        return arr

if __name__ == '__main__':
    fptr = open(os.environ['OUTPUT_PATH'], 'w')

    n = int(input().strip())

    arr = list(map(int, input().rstrip().split()))

    result = quickSort(arr)

    fptr.write(' '.join(map(str, result)))
    fptr.write('\n')

    fptr.close()
```

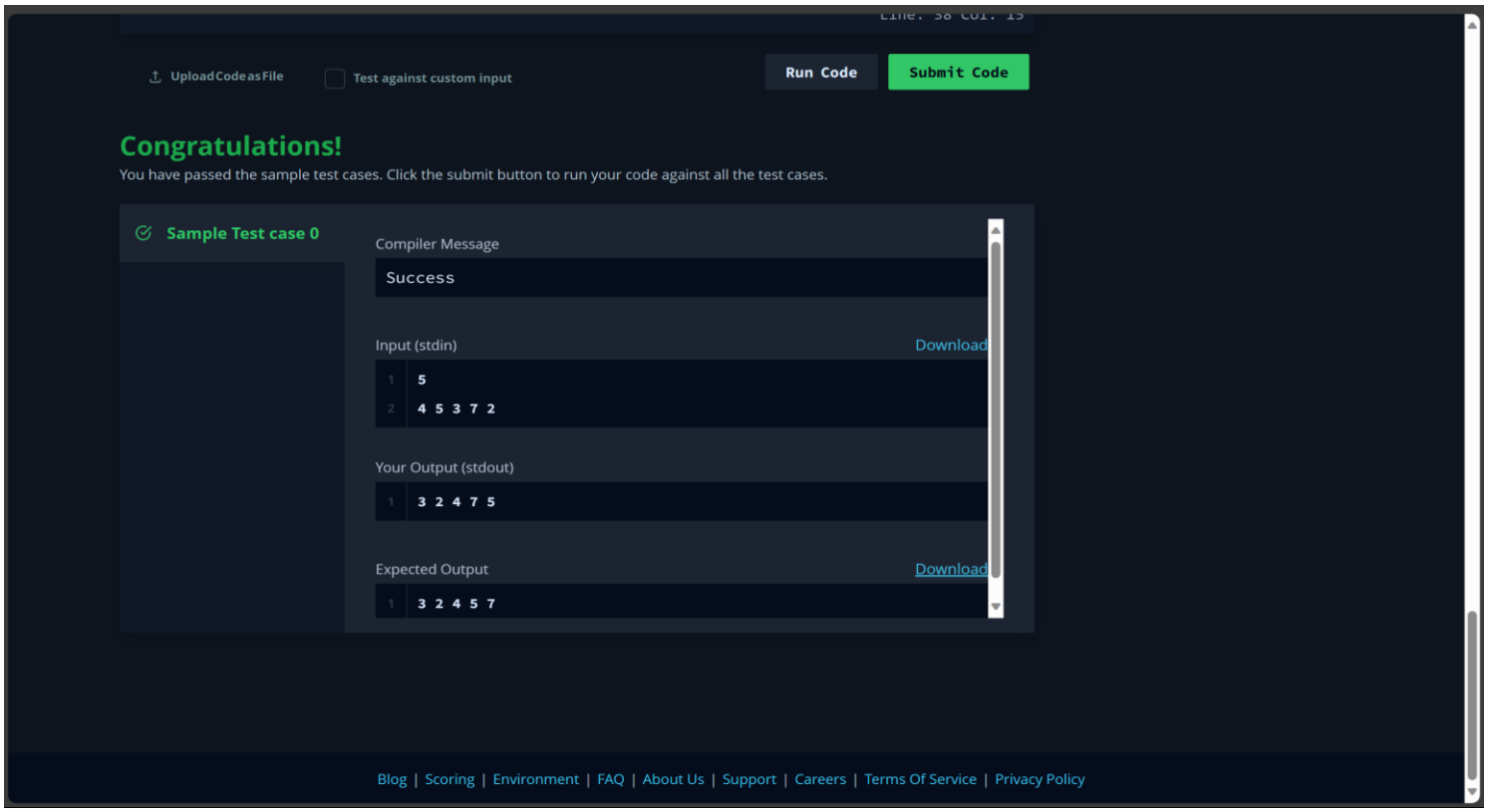
**Program -2**

```
#!/bin/python3

import math
import os
import random
import re
import sys

#
# Complete the 'closestNumbers' function below.
#
# The function is expected to return an INTEGER_ARRAY.
# The function accepts INTEGER_ARRAY arr as parameter.
#

def closestNumbers(arr):
    pairs = []
    mindiff = 99999999999
    arr.sort()

    # main logic
    for i in range(1, len(arr)):
        d = abs(arr[i-1] - arr[i])
        #new minimum difference
        if d < mindiff:
            mindiff = d
```

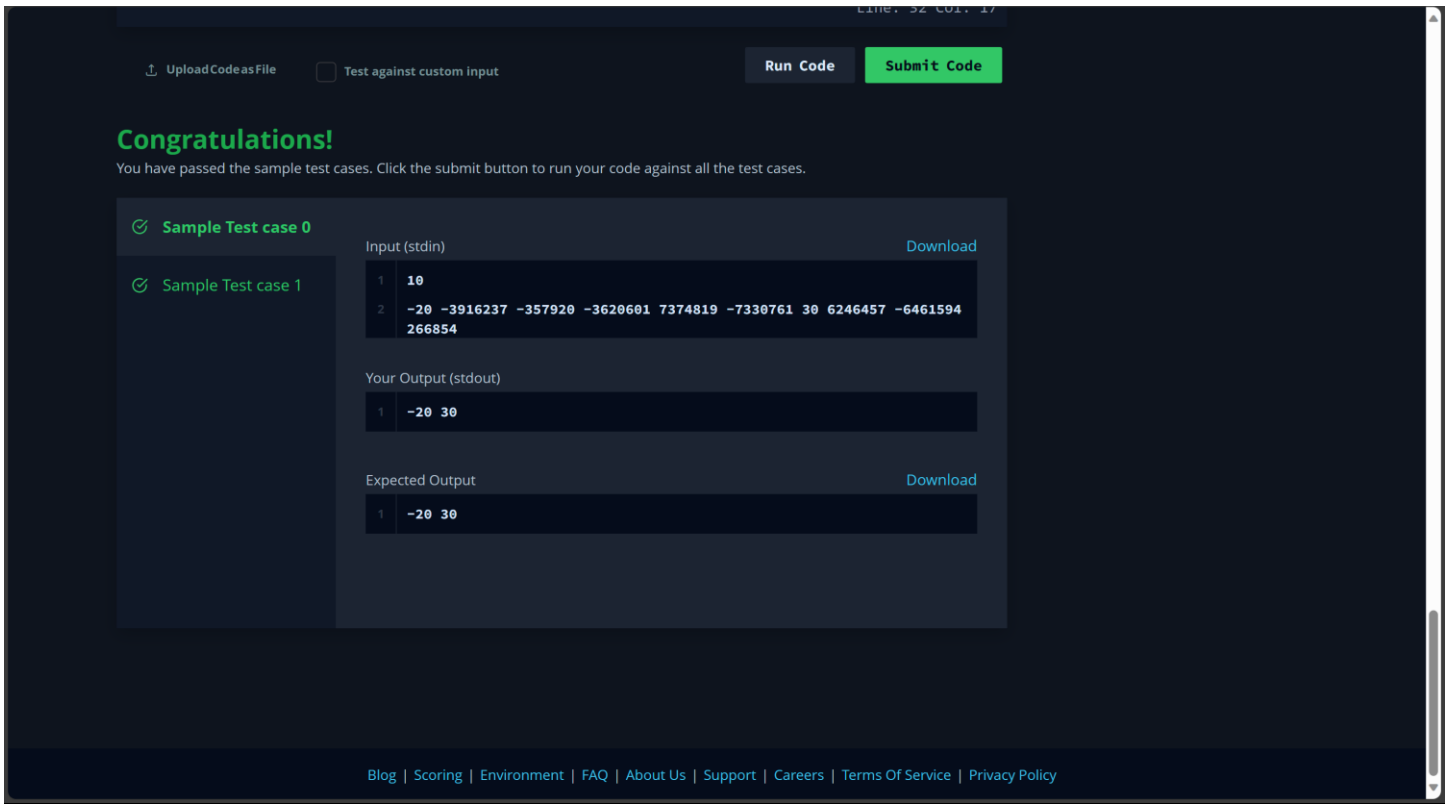
```
            pairs = [arr[i-1], arr[i]]
            #already exisiting minimum difference
        elif d == mindiff:
            pairs.extend([arr[i-1], arr[i]])

    return pairs

if __name__ == '__main__':
    fptr = open(os.environ['OUTPUT_PATH'], 'w')

    n = int(input().strip())

    arr = list(map(int, input().rstrip().split()))

    result = closestNumbers(arr)

    fptr.write(' '.join(map(str, result)))
    fptr.write('\n')

    fptr.close()
```

**5. Output:**

Program 1:


Upload Code as File
Test against custom input
Run Code
Submit Code
Congratulations!
You have passed the sample test cases. Click the submit button to run your code against all the test cases.
Sample Test case 0
Compiler Message
Success
Input (stdin)
Download
5
4 5 3 7 2
Your Output (stdout)
3 2 4 7 5
Expected Output
Download
3 2 4 5 7
Blog | Scoring | Environment | FAQ | About Us | Support | Careers | Terms Of Service | Privacy Policy

Program 2:




**Learning outcomes:**

1. Searching and sorting are important tasks in computer science.
2. Sorting algorithms are used to arrange elements of an array/list in a specific order
3. There are various sorting algorithms that can be used to complete this operation, such as bubble sort, selection sort, insertion sort, merge sort, quicksort, counting sort, radix sort, bucket sort, and heap sort
4. Binary search is a commonly used search algorithm that requires the array to be sorted beforehand.