PSG Institute of Technology and Applied Research

Name: Adhithya J
Email: 24z108@psgitech.ac.in
Roll no: 715524104007
Phone: 8807303793
Branch: PSG iTech
Department: CSE
Batch: 2028
Degree: B.E CSE

Scan to verify results

2024_28_IV_CSE A_Algorithms Lab

2027_PSG_Algorithms_Week 4_COD

Attempt : 1
Total Mark : 40
Marks Obtained : 40

Section 1 : Coding

1. Problem Statement

Jake, a sales representative, recorded the number of sales made each week of the month. He needs to sort the sales figures at odd positions in descending order and those at even positions in ascending order using insertion sort. This sorting will help him analyze the performance trends effectively.

Example

Input:

10

25 36 96 58 74 14 35 15 75 95

Output:

96 14 75 15 74 36 35 58 25 95

Explanation

Odd positions (indices 0, 2, 4, 6, 8) in descending order: 25, 96, 74, 35, 75 becomes [96, 75, 74, 35, 25].

Even positions (indices 1, 3, 5, 7, 9) in ascending order: 36, 58, 14, 15, 95 becomes [14, 15, 36, 58, 95].

Combine the sorted odd and even positions: [96, 14, 75, 15, 74, 36, 35, 58, 25, 95].

Input Format

The first line contains a single integer N, representing the number of weeks.

The second line contains N space-separated integers, representing the number of sales made each week.

Output Format

The output displays the sorted array, where the sales numbers at odd positions (1st, 3rd, 5th, etc.) are sorted in descending order and those at even positions (2nd, 4th, 6th, etc.) are sorted in ascending order.

No trailing spaces added at the end.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 10
25 36 96 58 74 14 35 15 75 95

Output: 96 14 75 15 74 36 35 58 25 95

Answer

```
#include <stdio.h>
void insertionSort(int arr[], int n, int order){
   for (int i = 1; i < n; i++) {
      int key = arr[i];
      int j = i - 1;
```

```
        if (order == 1) {
            while (j >= 0 && arr[j] > key) {
                arr[j + 1] = arr[j];
                j--;
            }
        } else {
            while (j >= 0 && arr[j] < key) {
                arr[j + 1] = arr[j];
                j--;
            }
        }
        arr[j + 1] = key;
    }
}
int main() {
    int n;
    scanf("%d", &n);
    int sales[n];
    for (int i = 0; i < n; i++){
        scanf("%d", &sales[i]);
    }
    int odd[n / 2 + n % 2];
    int even[n / 2];
    int oddIndex = 0, evenIndex = 0;

    for (int i = 0; i < n; i++) {
        if (i % 2 == 0){
            odd[oddIndex++] = sales[i];
        }else{
            even[evenIndex++] = sales[i];
        }
    }
    insertionSort(odd, oddIndex, 0);
    insertionSort(even, evenIndex, 1);
    oddIndex = 0;
    evenIndex = 0;
    for (int i = 0; i < n; i++) {
        if (i % 2 == 0) {
            printf("%d", odd[oddIndex++]);
        } else {
            printf("%d", even[evenIndex++]);
        }
        if(i < n -1)
            printf(" ");
    }
    printf("\n");

    return 0;
}
```

Status : Correct ***Marks : 10/10***

2. Problem Statement

You are working on an optimization task for a sorting algorithm that uses insertion sort. Your goal is to determine the efficiency of the algorithm by counting the number of swaps needed to sort an array of integers.

Write a program that takes an array as input and calculates the number of swaps performed during the insertion sort process.

Example1

Input:

5

2 1 3 1 2

Output:

4

Explanation:

Step 1: [2, 1, 3, 1, 2] (No swaps)

Step 2: [1, 2, 3, 1, 2] (1 swap, element 1 shifts 1 place to the left)

Step 3: [1, 2, 3, 1, 2] (No swaps)

Step 4: [1, 1, 2, 3, 2] (2 swaps; element 1 shifts 2 places to the left)

Step 5: [1, 1, 2, 2, 3] (1 swap, element 2 shifts 1 place to the left)

Total number of swaps: 1 + 2 + 1 = 4

Example2

Input:

7

12 15 1 5 6 14 11

Output:

10

Explanation:

Step 1: [12, 15, 1, 5, 6, 14, 11] (No swaps)

Step 2: [12, 15, 1, 5, 6, 14, 11] (1 swap, element 15 shifts 1 place to the left)

Step 3: [12, 15, 1, 5, 6, 14, 11] (No swaps)

Step 4: [1, 12, 15, 5, 6, 14, 11] (2 swaps, element 1 shifts 2 places to the left)

Step 5: [1, 5, 12, 15, 6, 14, 11] (1 swap, element 5 shifts 1 place to the left)

Step 6: [1, 5, 6, 12, 15, 14, 11] (2 swaps, element 6 shifts 2 places to the left)

Step 7: [1, 5, 6, 12, 14, 15, 11] (1 swap, element 14 shifts 1 place to the left)

Step 8: [1, 5, 6, 11, 12, 14, 15] (3 swaps, element 11 shifts 3 places to the left)

Total number of swaps: 1 + 2 + 1 + 2 + 1 + 3 = 10

Input Format

The first line of input consists of an integer n, representing the number of elements in the array.

The second line of input consists of n space-separated integers, representing the elements of the array.

Output Format

The output prints the number of swaps performed during the insertion sort process as an integer.

No trailing spaces added at the end.

Refer to the sample output for the formatting specifications.

Sample Test Case

Input: 5
2 1 3 1 2

Output: 4

Answer

```
#include <stdio.h>

int main() {
   int n, j, key;
   int swaps = 0;
   scanf("%d", &n);
   int arr[n];
   for (int i = 0; i < n; i++) {
      scanf("%d", &arr[i]);
   }
   for (int i = 1; i < n; i++) {
      key = arr[i];
      j = i - 1;
      while (j >= 0 && arr[j] > key) {
         arr[j + 1] = arr[j];
         j = j - 1;
         swaps++;
      }
      arr[j + 1] = key;
   }
   printf("%d", swaps);
   return 0;
}
```

Status : Correct ***Marks : 10/10***

3. Problem Statement

Romit is developing software for a car rental service that wants to offer its customers a selection of the most fuel-efficient vehicles. He has data on the fuel efficiency (miles per gallon) of various car models and wants to sort them in descending order of efficiency.

Your task is to help him implement a function that takes an array of car fuel efficiencies and sorts them using the heap sort algorithm in descending order, allowing customers to choose eco-friendly options easily.

Input Format

The first line of input consists of an integer N, representing the number of cars.

The second line consists of N space-separated integers, representing the car fuel efficiencies.

Output Format

The output prints the car fuel efficiencies, sorted in descending order.

Refer to the sample output for formatting specifications.

Sample Test Case

Input: 4
90 87 78 92

Output: 92 90 87 78

Answer

```
#include <stdio.h>
#include <stdlib.h>

void heapify(int arr[], int n, int i){
    int largest = i;
    int l = 2 * i+1;
    int r = 2 * i+2;
    if(l < n && arr[l] > arr[largest])
        largest = l;
    if(r < n && arr[r] > arr[largest])
```

```
        largest = r;
    if(largest != i){
        int temp = arr[i];
        arr[i] = arr[largest];
        arr[largest] = temp;
        heapify(arr, n, largest);
    }
}
void heapSort(int arr[], int n){
    for(int i = (n/2)-1; i >= 0; i--)
        heapify(arr, n, i);
    for(int i = n-1; i>0; i--){
        int temp = arr[0];
        arr[0] = arr[i];
        arr[i] = temp;
        heapify(arr, i, 0);
    }
}

int main() {
    int n;
    scanf("%d", &n);

    int *arr = (int *)malloc(n * sizeof(int));

    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }

    heapSort(arr, n);

    for (int i = n - 1; i >= 0; i--) {
        printf("%d ", arr[i]);
    }

    free(arr);
    return 0;
}
```

Status : Correct   ***Marks : 10/10***

4. Problem Statement

A warehouse manager is responsible for managing shipments that arrive with unique tracking numbers. To streamline the processing, the manager wants to identify the Kth largest tracking number efficiently.

Implement a program that takes a list of tracking numbers as input and finds the Kth largest tracking number using the heap sort algorithm.

Input Format

The first line contains an integer N, representing the number of tracking numbers.

The second line contains N space-separated integers, each representing a tracking number.

The third line contains an integer K, representing the desired position of the Kth largest tracking number.

Output Format

If K is invalid (greater than N or less than or equal to 0), print "Invalid K value."

Otherwise, print a single line containing the Kth largest tracking number.

Refer to the sample output for the exact format.

Sample Test Case

Input: 5
1234 5678 9012 3456 7890
4
Output: The 4th largest ISBN number is: 3456

Answer

```
#include <stdio.h>

void swap(int *a,int *b){
  int t=*a;
```

```
    *a=*b;
    *b=t;
}

void heapify(int a[],int n,int i){
    int l=2*i+1;
    int r=2*i+2;
    int m=i;
    if(l<n&&a[l]>a[m])m=l;
    if(r<n&&a[r]>a[m])m=r;
    if(m!=i){
        swap(&a[i],&a[m]);
        heapify(a,n,m);
    }
}

int main(){
    int n,k;
    scanf("%d",&n);
    int a[n];
    for(int i=0;i<n;i++)
        scanf("%d",&a[i]);
    scanf("%d",&k);
    if(k<=0||k>n){
        printf("Invalid K value.");
        return 0;
    }
    for(int i=n/2-1;i>=0;i--)
        heapify(a,n,i);
    for(int i=n-1;i>=n-k+1;i--){
        swap(&a[0],&a[i]);
        heapify(a,i,0);
    }
    printf("The %dth largest ISBN number is: %d",k,a[0]);
    return 0;
}
```

Status : Correct ***Marks : 10/10***