a) Write a c program to implement Insertion sort & Quick sort

Insertion sort

```
1. #include <stdio.h>
2.
3. void insert(int a[], int n) /* function to sort an aay with insertion
   sort */
4. {
5.     int i, j, temp;
6.     for (i = 1; i < n; i++) {
7.         temp = a[i];
8.         j = i - 1;
9.
10.             while(j>=0 && temp <= a[j])  /* Move the elements gr
   eater than temp to one position ahead from their current position*
   /
11.             {
12.                 a[j+1] = a[j];
13.                 j = j-1;
14.             }
15.             a[j+1] = temp;
16.         }
17.     }
18.
19.     void printArr(int a[], int n) /* function to print the array */
20.     {
21.         int i;
22.         for (i = 0; i < n; i++)
23.             printf("%d ", a[i]);
24.     }
25.
26.     int main()
27.     {
28.         int a[] = { 12, 31, 25, 8, 32, 17 };
```

```
29.         int n = sizeof(a) / sizeof(a[0]);
30.         printf("Before sorting array elements are - \n");
31.         printArr(a, n);
32.         insert(a, n);
33.         printf("\nAfter sorting array elements are - \n");
34.         printArr(a, n);
35.
36.         return 0;
37.     }
```

Quick sort

```
1. #include <stdio.h>
2. /* function that consider last element as pivot,
3. place the pivot at its exact position, and place
4. smaller elements to left of pivot and greater
5. elements to right of pivot.  */
6. int partition (int a[], int start, int end)
7. {
8.     int pivot = a[end]; // pivot element
9.     int i = (start - 1);
10.
11.         for (int j = start; j <= end - 1; j++)
12.         {
13.             // If current element is smaller than the pivot
14.             if (a[j] < pivot)
15.             {
16.                 i++; // increment index of smaller element
17.                 int t = a[i];
18.                 a[i] = a[j];
19.                 a[j] = t;
20.             }
21.         }
22.         int t = a[i+1];
```

```
23.          a[i+1] = a[end];
24.          a[end] = t;
25.          return (i + 1);
26.       }
27.
28.       /* function to implement quick sort */
29.       void quick(int a[], int start, int end) /* a[] = array to be sort
   ed, start = Starting index, end = Ending index */
30.       {
31.          if (start < end)
32.          {
33.             int p = partition(a, start, end); //p is the partitioning ind
   ex
34.             quick(a, start, p - 1);
35.             quick(a, p + 1, end);
36.          }
37.       }
38.
39.       /* function to print an array */
40.       void printArr(int a[], int n)
41.       {
42.          int i;
43.          for (i = 0; i < n; i++)
44.             printf("%d ", a[i]);
45.       }
46.       int main()
47.       {
48.          int a[] = { 24, 9, 29, 14, 19, 27 };
49.          int n = sizeof(a) / sizeof(a[0]);
50.          printf("Before sorting array elements are - \n");
51.          printArr(a, n);
52.          quick(a, 0, n - 1);
53.          printf("\nAfter sorting array elements are - \n");
54.          printArr(a, n);
```

55.
56. **return** 0;
}

b) Write a c program to sort the given n integers and perform following operations
i) Find the products of every two odd position elements
ii) Find the sum of every two even position elements
Explanation:
Input : 9
1 9 8 3 5 4 7 2 6
Output: 3 15 35 63
6 10 14

```
#include <stdio.h>

// Function to swap two integers

void swap(int *a, int *b) {

    int temp = *a;

    *a = *b;

    *b = temp;

}

// Function to perform bubble sort

void bubbleSort(int arr[], int n) {

    for (int i = 0; i < n-1; i++) {

        for (int j = 0; j < n-i-1; j++) {

            if (arr[j] > arr[j+1]) {

                swap(&arr[j], &arr[j+1]);

            }

        }

    }
```

```
}
int main() {
    int n;
    printf("Enter the number of integers: ");
    scanf("%d", &n);

    int arr[n];
    printf("Enter %d integers: ", n);
    for (int i = 0; i < n; i++) {
        scanf("%d", &arr[i]);
    }
    // Sorting the array
    bubbleSort(arr, n);
    // Finding products of every two odd position elements
    printf("Products of every two odd position elements: ");
    for (int i = 1; i < n; i += 2) {
        if (i + 1 < n) {
            printf("%d ", arr[i] * arr[i + 1]);
        }
    }
    printf("\n");
    // Finding sum of every two even position elements
    printf("Sum of every two even position elements: ");
    for (int i = 0; i < n; i += 2) {
```

```
        if (i + 1 < n) {
            printf("%d ", arr[i] + arr[i + 1]);
        }
    }
    printf("\n");

    return 0;
}
```