Data Structure Assignment-1

Question 1. Write algorithms for the following: A. Insertion Sort
B. Selection Sort
C. Merge Sort
D. Bubble Sort

Answer 1.

A. **Insertion Sort Algorithm**

To sort an array of size N in ascending order:

- Iterate from arr[1] to arr[N] over the array.
- Compare the current element (key) to its predecessor.
- If the key element is smaller than its predecessor, compare it to the elements before. Move the greater elements one position up to make space for the swapped element.

B. **Selection Sort Algorithm**

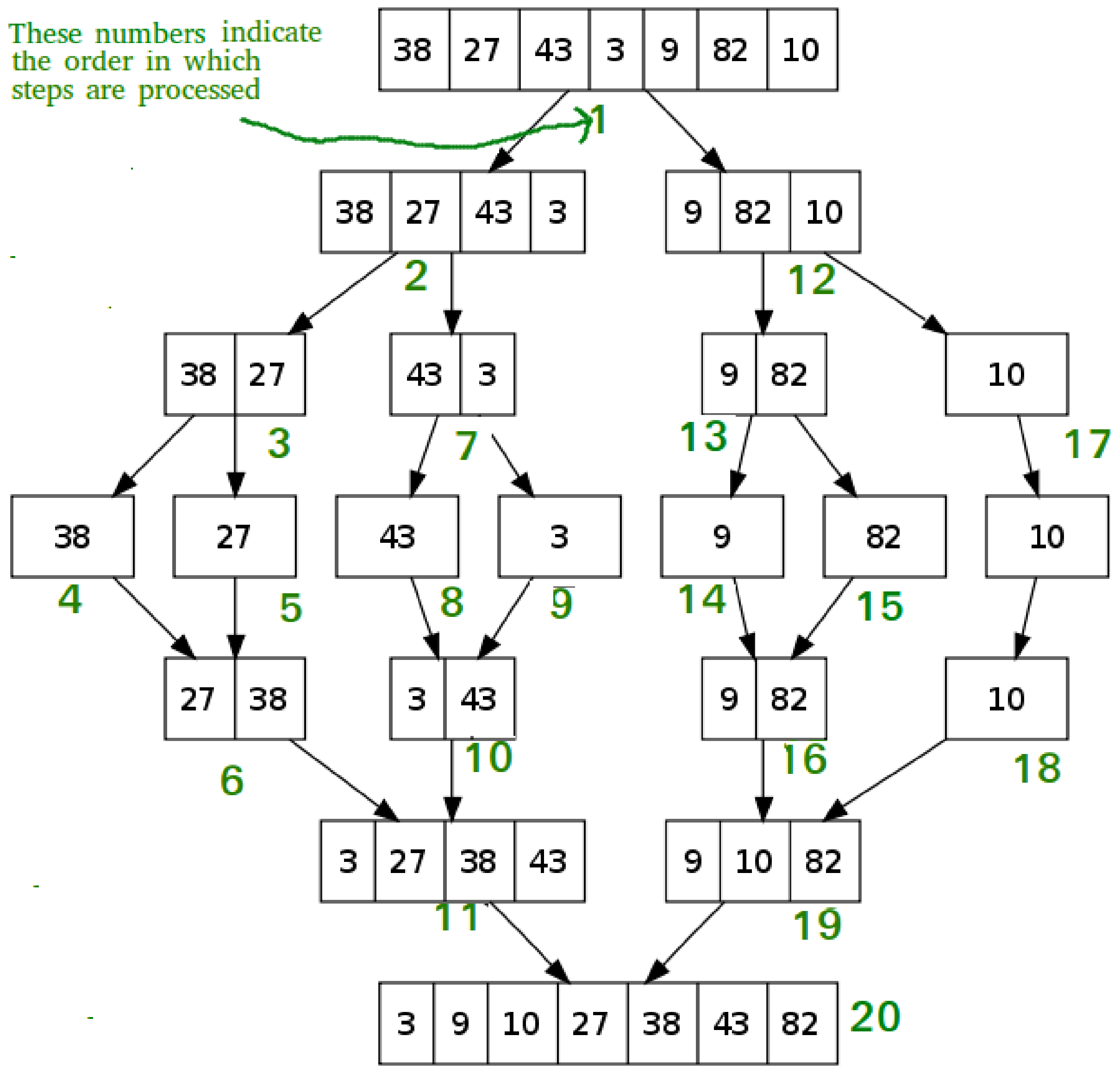
- Initialize minimum value(min_idx) to location 0
- Traverse the array to find the minimum element in the array
- While traversing if any element smaller than min_idx is found then swap both the values.
- Then, increment min_idx to point to the next element
- Repeat until the array is sorted

B. **Merge Sort Algorithm**

In the following algorithm, arr is the given array, beg is the starting element, and end is the last element of the array.

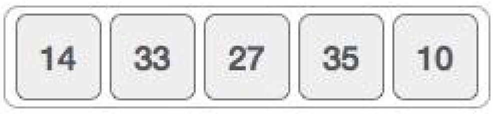
```
MERGE_SORT(arr, beg, end)

if beg < end
    set mid = (beg + end)/2
    MERGE_SORT(arr, beg, mid)
    MERGE_SORT(arr, mid + 1, end)
    MERGE (arr, beg, mid, end)
end of if

END MERGE_SORT
```

The important part of the merge sort is the MERGE function. This function performs the merging of two sorted sub-arrays that are A[beg…mid] and A[mid+1…end], to build one sorted array A[beg…end]. So, the inputs of the MERGE function are A[], beg, mid, and end.

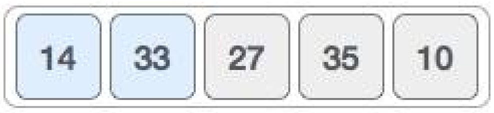

These numbers indicate the order in which steps are processed
38 27 43 3 9 82 10
1
38 27 43 3
9 82 10
2
12
38 27
43 3
9 82
10
3
7
13
17
38
27
43
3
9
82
10
4
5
8
9
14
15
27 38
3 43
9 82
10
6
10
16
18
3 27 38 43
9 10 82
11
19
3 9 10 27 38 43 82
20

B. Bubble Sort Algorithm

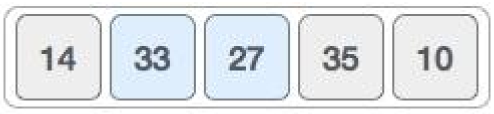
We take an unsorted array for our example. Bubble sort takes O(n2) time so we're keeping it short and precise.



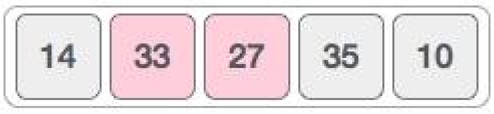
Bubble sort starts with the very first two elements, comparing them to check which one is greater.



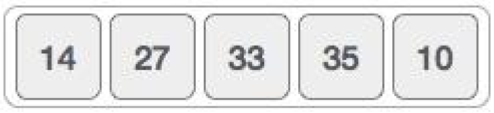
In this case, the value 33 is greater than 14, so it is already in sorted locations. Next, we compare 33 with 27.



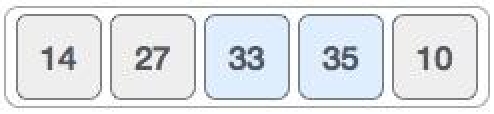
We find that 27 is smaller than 33 and these two values must be swapped.



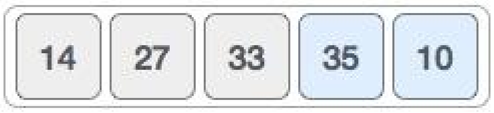
The new array should look like this –



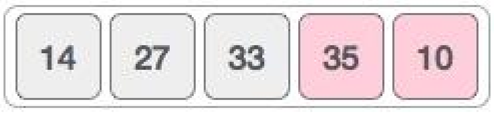
Next we compare 33 and 35. We find that both are in already sorted positions.



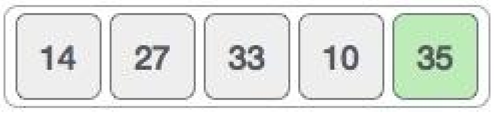
Then we move to the next two values, 35 and 10.



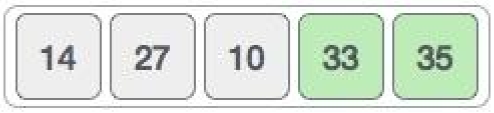
We know then that 10 is smaller than 35. Hence they are not sorted.



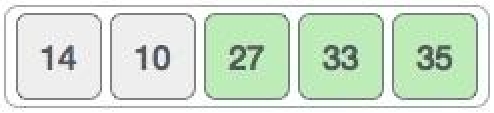
We swap these values. We find that we have reached the end of the array. After one iteration, the array should look like this −



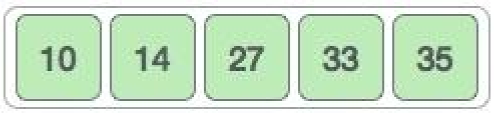
To be precise, we are now showing how an array should look like after each iteration. After the second iteration, it should look like this −



Notice that after each iteration, at least one value moves at the end.



And when there's no swap required, bubble sorts learns that an array is completely sorted.



Algorithm

We assume the list is an array of n elements. We further assume that the swap function swaps the values of the given array elements.

```
begin BubbleSort(list)
```

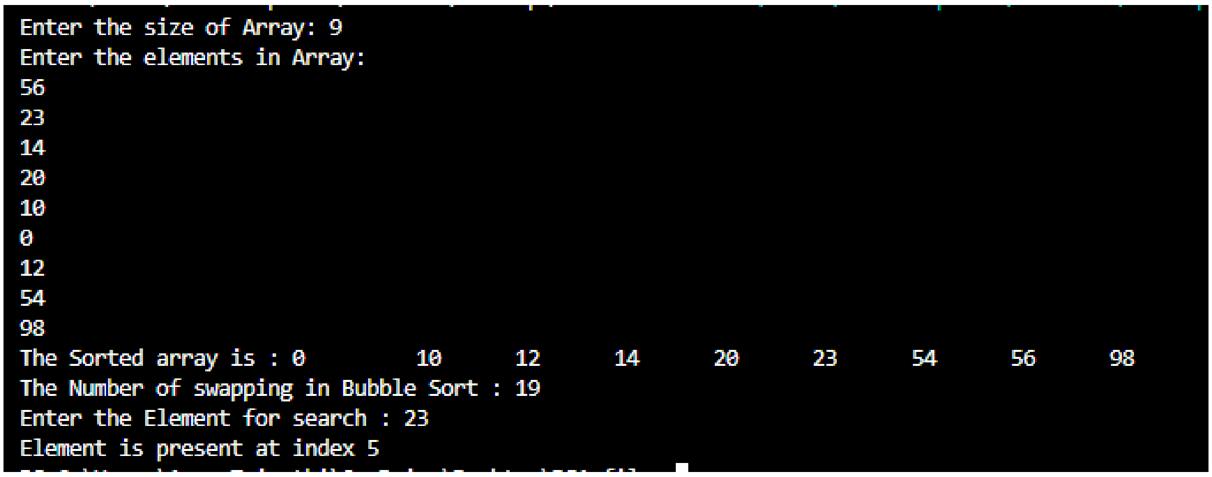
```
  for all elements of list
    if list[i] > list[i+1]
      swap(list[i], list[i+1])
    end if
  end for

  return list

end BubbleSort
```

Question 2. WAP to display the total number of comparisons and swapping made by each searching /sorting function for the given input N.

Answer 2.

```c
#include <stdio.h>
int c = 0;
void bubble(int a[], int n) // function to implement bubble sort
{
    int i, j, temp;
    for (i = 0; i < n; i++)
    {
        for (j = i + 1; j < n; j++)
        {
            if (a[j] < a[i])
            {
                temp = a[i];
                a[i] = a[j];
                a[j] = temp;
                c++;
            }
        }
    }
}

int search(int arr[], int n, int x)

{
    int i;
    for (i = 0; i < n; i++)
        if (arr[i] == x)
            return i;
    return -1;
}

int main(void)
{
    int size = 0;
    printf("Enter the size of Array: ");
    scanf("%d", &size);

    int arr[size];
    int n = sizeof(arr) / sizeof(arr[0]);
    printf("Enter the elements in Array: \n");
    for (int i = 0; i < size; i++)
    {
        scanf("%d" , &arr[i]);
    }

    bubble(arr,n);
    printf("The Sorted array is : ");
    for (int i = 0; i < n; i++)
    {
        printf("%d \t",arr[i]);
    }
    printf("\n");
    printf("The Number of swapping in Bubble Sort : %d\n",c);

    // Function call
    int x = 0;
    printf("Enter the Element for search : ");
    scanf("%d", &x);
    int result = search(arr, n, x);
    (result == -1)
        ? printf("Element is not present in array")
        : printf("Element is present at index %d", result);
    return 0;
}
```

Output:-



```
Enter the size of Array: 9
Enter the elements in Array:
56
23
14
20
10
0
12
54
98
The Sorted array is : 0         10      12      14      20      23      54      56      98
The Number of swapping in Bubble Sort : 19
Enter the Element for search : 23
Element is present at index 5
```

Question 3. Create a structure/class for a group of 50 students holding data for their Regn no., Name, Branch, CGPA

a) Call linear search function to display data of student with a particular Regn no.

b) Call bubble sort function to arrange data of students according to Regn

c) Apply binary search on the above output (part b) to display data of a student with a particular Regn no.

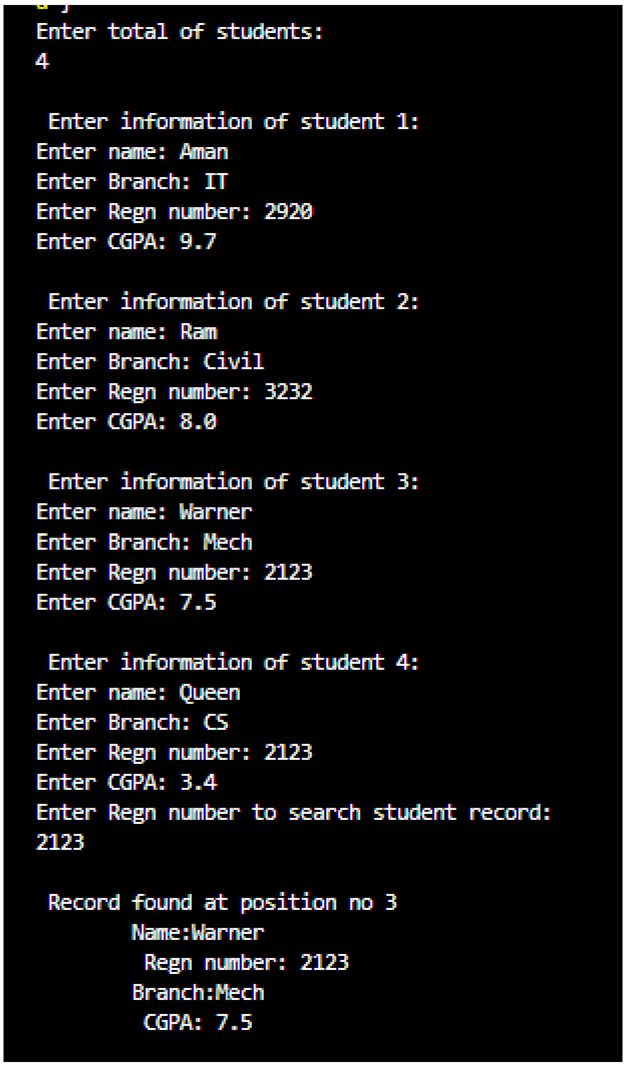
d) Use and modify Insertion sort logic to arrange data of students in descending order of CGPA.1

A.

```
// Question 3. Create a structure/class for a group of 50 students
holding data for their Regn no., Name, Branch, CGPA
// a) Call linear search function to display data of student with a
particular Regn no.
// b) Call bubble sort function to arrange data of students according
to Regn
// c) Apply binary search on the above output (part b) to display data
of a student with a particular Regn no.
// d) Use and modify Insertion sort logic to arrange data of students
in descending order of CGPA .1

#include <stdio.h>
struct student
{
char name[50];
int regn;
float cgpa;
char branch[30];
} s[100];

int main()
{
int i,n;
struct student s[100];

printf("Enter total of students:\n");
scanf("%d",&n);

for(i=0;i<n;i++)
{
printf("\n Enter information of student %d:\n",i+1);
printf("Enter name: ");
scanf("%s", s[i].name);
printf("Enter Branch: ");
scanf("%s", s[i].branch);

printf("Enter Regn number: ");
scanf("%d", &s[i].regn);
```

```c
printf("Enter CGPA: ");
scanf("%f", &s[i].cgpa);
}

int rn= 0; int k =0;
printf("Enter Regn number to search student record:\n");
scanf("%d",&rn);

for(i=0;i<n;i++)
{
if(s[i].regn==rn)
{
k=i;//for finding position of student

printf("\n Record found at position no %d \n",k+1);
printf("\tName:%s\n ",s[k].name);
printf("\t Regn number: %d\n",s[k].regn);
printf("\tBranch:%s\n ",s[k].branch);
printf("\t CGPA: %.1f\n\n",s[k].cgpa);
}
}
return 0;
}
```

Output:-



```
Enter total of students:
4

 Enter information of student 1:
Enter name: Aman
Enter Branch: IT
Enter Regn number: 2920
Enter CGPA: 9.7

 Enter information of student 2:
Enter name: Ram
Enter Branch: Civil
Enter Regn number: 3232
Enter CGPA: 8.0

 Enter information of student 3:
Enter name: Warner
Enter Branch: Mech
Enter Regn number: 2123
Enter CGPA: 7.5

 Enter information of student 4:
Enter name: Queen
Enter Branch: CS
Enter Regn number: 2123
Enter CGPA: 3.4
Enter Regn number to search student record:
2123

 Record found at position no 3
        Name:Warner
         Regn number: 2123
        Branch:Mech
         CGPA: 7.5
```

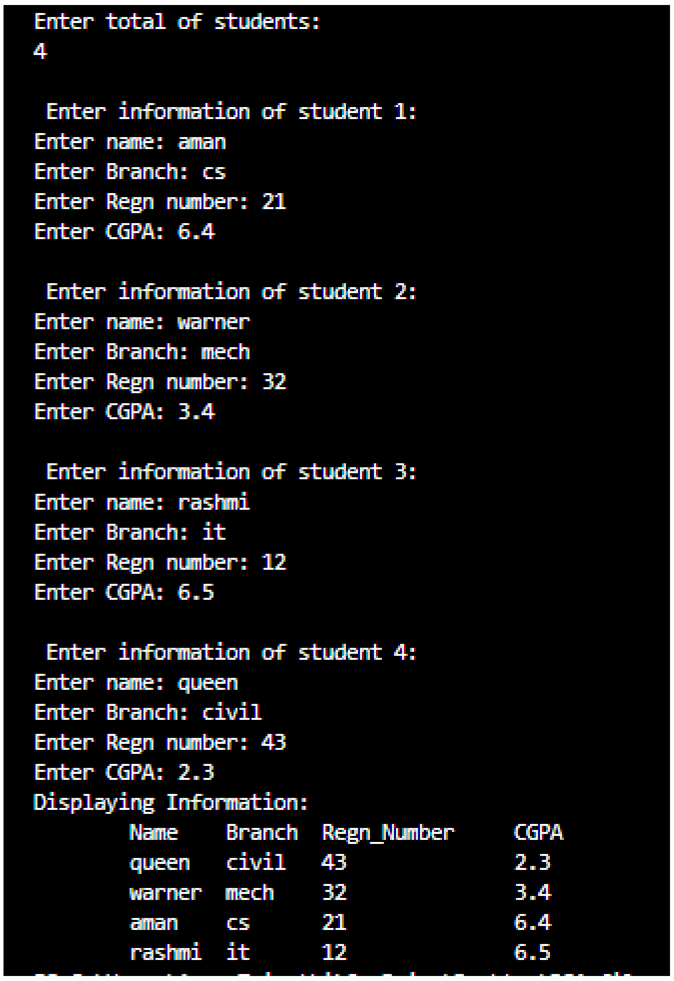
B.

```
// Question 3. Create a structure/class for a group of 50 students
holding data for their Regn no., Name, Branch, CGPA
// a) Call linear search function to display data of student with a
particular Regn no.
// b) Call bubble sort function to arrange data of students according
to Regn
// c) Apply binary search on the above output (part b) to display data
of a student with a particular Regn no.
// d) Use and modify Insertion sort logic to arrange data of students
in descending order of CGPA .1

#include <stdio.h>
```

```
void bubbleSortDesc();
struct student
{
char name[50];
int regn;
float cgpa;
char branch[30];
} s[100];

int main()
{
int i,n;
struct student s[100];

printf("Enter total of students:\n");
scanf("%d",&n);

for(i=0;i<n;i++)
{
printf("\n Enter information of student %d:\n",i+1);
printf("Enter name: ");
scanf("%s", s[i].name);
printf("Enter Branch: ");
scanf("%s", s[i].branch);
printf("Enter Regn number: ");
scanf("%d", &s[i].regn);



printf("Enter CGPA: ");
scanf("%f", &s[i].cgpa);
}
bubbleSortDesc(s, n);

printf("Displaying Information:\n");
printf("\tName  \tBranch \tRegn_Number \tCGPA\n");
for(i=0;i<n;i++)
{
printf("\t%s \t%s \t%d \t\t%.1f
\n",s[i].name,s[i].branch,s[i].regn,s[i].cgpa);
}

return 0;
```

```c
}

void bubbleSortDesc(struct student stud_list[100], int s)
{
int i, j;
struct student temp;

for (i = 0; i < s-1; i++)
{
for (j = 0; j < (s - 1-i); j++)
{
if (stud_list[j].regn < stud_list[j + 1].regn)
{
temp = stud_list[j];
stud_list[j] = stud_list[j + 1];
stud_list[j + 1] = temp;
}
}
}
}
```

Output:-

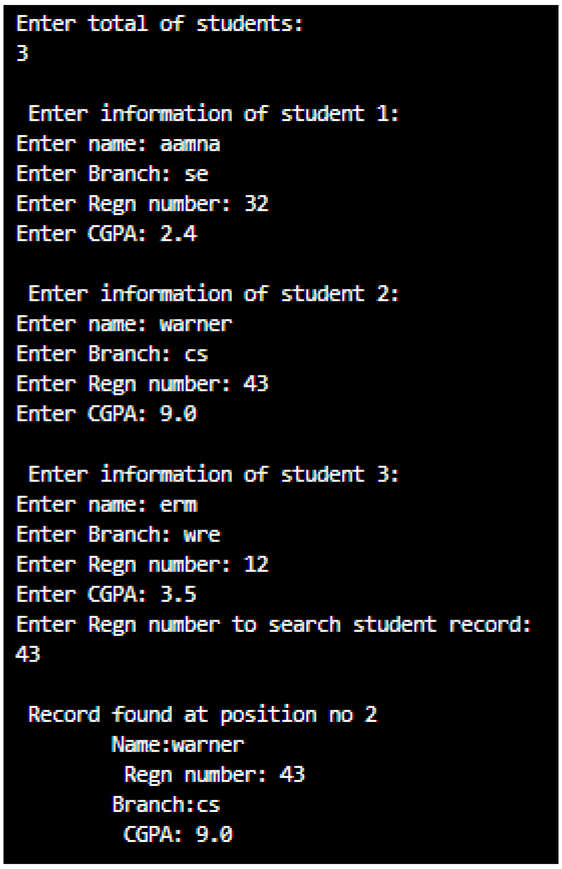


```
Enter total of students:
4

 Enter information of student 1:
Enter name: aman
Enter Branch: cs
Enter Regn number: 21
Enter CGPA: 6.4

 Enter information of student 2:
Enter name: warner
Enter Branch: mech
Enter Regn number: 32
Enter CGPA: 3.4

 Enter information of student 3:
Enter name: rashmi
Enter Branch: it
Enter Regn number: 12
Enter CGPA: 6.5

 Enter information of student 4:
Enter name: queen
Enter Branch: civil
Enter Regn number: 43
Enter CGPA: 2.3
Displaying Information:
        Name    Branch  Regn_Number     CGPA
        queen   civil   43              2.3
        warner  mech    32              3.4
        aman    cs      21              6.4
        rashmi  it      12              6.5
```

C.

```
// Question 3. Create a structure/class for a group of 50 students holding data for their Regn no., Name, Branch, CGPA
// a) Call linear search function to display data of student with a particular Regn no.
// b) Call bubble sort function to arrange data of students according to Regn
// c) Apply binary search on the above output (part b) to display data of a student with a particular Regn no.
// d) Use and modify Insertion sort logic to arrange data of students in descending order of CGPA .1

#include <stdio.h>
struct student
{
    char name[50];
    int regn;
    float cgpa;
    char branch[30];
} s[100];

int main()
{
    int i, n;
    struct student s[100];

    printf("Enter total of students:\n");
    scanf("%d", &n);

    for (i = 0; i < n; i++)
    {
        printf("\n Enter information of student %d:\n", i + 1);
        printf("Enter name: ");
        scanf("%s", s[i].name);
        printf("Enter Branch: ");
        scanf("%s", s[i].branch);

        printf("Enter Regn number: ");
        scanf("%d", &s[i].regn);

        printf("Enter CGPA: ");
        scanf("%f", &s[i].cgpa);
    }

    int rn = 0;
    int k = 0;
    printf("Enter Regn number to search student record:\n");
    scanf("%d", &rn);
    int beg = 0;
    int end = n;
    while (beg <= end)
    {
        int mid = (beg + end) / 2;
        if (s[mid].regn == rn)
        {
            k = mid; // for finding position of student

            printf("\n Record found at position no %d \n", k + 1);
            printf("\tName:%s\n ", s[k].name);
            printf("\t Regn number: %d\n", s[k].regn);
            printf("\tBranch:%s\n ", s[k].branch);
            printf("\t CGPA: %.1f\n\n", s[k].cgpa);
            return 1;
        }
        else if (s[mid].regn < rn)
            beg = mid + 1;
        else
            end = mid - 1;
    }
    return 0;
}
```

Output:-



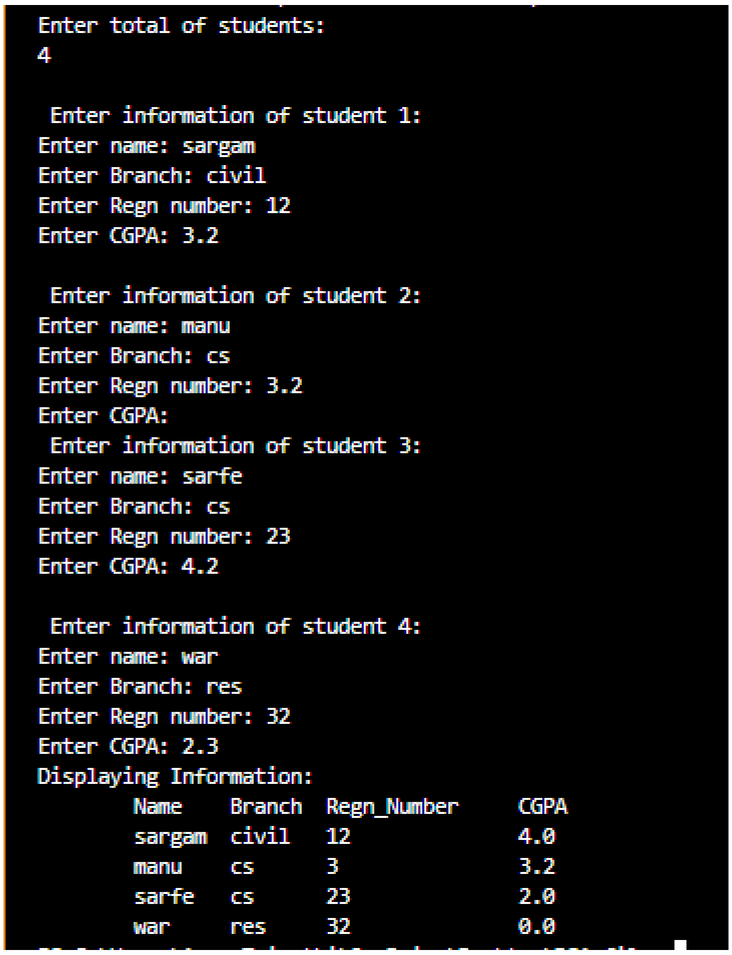
```
Enter total of students:
3

 Enter information of student 1:
Enter name: aamna
Enter Branch: se
Enter Regn number: 32
Enter CGPA: 2.4

 Enter information of student 2:
Enter name: warner
Enter Branch: cs
Enter Regn number: 43
Enter CGPA: 9.0

 Enter information of student 3:
Enter name: erm
Enter Branch: wre
Enter Regn number: 12
Enter CGPA: 3.5
Enter Regn number to search student record:
43

 Record found at position no 2
        Name:warner
         Regn number: 43
        Branch:cs
         CGPA: 9.0
```

D.

```
// Question 3. Create a structure/class for a group of 50 students holding data for their Regn no., Name, Branch, CGPA
// a) Call linear search function to display data of student with a particular Regn no.
// b) Call bubble sort function to arrange data of students according to Regn
// c) Apply binary search on the above output (part b) to display data of a student with a particular Regn no.
// d) Use and modify Insertion sort logic to arrange data of students in descending order of CGPA .1

#include <stdio.h>
void insertionSortDesc();
struct student
{
char name[50];
int regn;
float cgpa;
char branch[30];
} s[100];

int main()
{
int i,n;
struct student s[100];

printf("Enter total of students:\n");
scanf("%d",&n);

for(i=0;i<n;i++)
{
printf("\n Enter information of student %d:\n",i+1);
printf("Enter name: ");
scanf("%s", s[i].name);
printf("Enter Branch: ");
scanf("%s", s[i].branch);
printf("Enter Regn number: ");
scanf("%d", &s[i].regn);
```

```c
printf("Enter CGPA: ");
scanf("%f", &s[i].cgpa);
}
insertionSortDesc(s, n);

printf("Displaying Information:\n");
printf("\tName  \tBranch \tRegn_Number \tCGPA\n");
for(i=0;i<n;i++)
{
printf("\t%s \t%s \t%d \t\t%.1f
\n",s[i].name,s[i].branch,s[i].regn,s[i].cgpa);
}

return 0;
}

void insertionSortDesc(struct student stud_list[100], int s)
{
    for (int step = 1; step < s; step++) {
    int key = stud_list[step].cgpa;
    int j = step - 1;

    // Compare key with each element on the left of it until an element
smaller than
    // it is found.
    // For descending order, change key<array[j] to key>array[j].
    while (key > stud_list[j].cgpa && j >= 0) {
      stud_list[j + 1].cgpa = stud_list[j].cgpa;
      --j;
    }
    stud_list[j + 1].cgpa = key;
  }
}
```

Output:-



```
Enter total of students:
4

 Enter information of student 1:
Enter name: sargam
Enter Branch: civil
Enter Regn number: 12
Enter CGPA: 3.2

 Enter information of student 2:
Enter name: manu
Enter Branch: cs
Enter Regn number: 3.2
Enter CGPA:
 Enter information of student 3:
Enter name: sarfe
Enter Branch: cs
Enter Regn number: 23
Enter CGPA: 4.2

 Enter information of student 4:
Enter name: war
Enter Branch: res
Enter Regn number: 32
Enter CGPA: 2.3
Displaying Information:
        Name    Branch  Regn_Number     CGPA
        sargam  civil   12              4.0
        manu    cs      3               3.2
        sarfe   cs      23              2.0
        war     res     32              0.0
```

Aman tripathi
05221202021