

NAME : PREM SAH
PRN : 123B1B234
DIV : D

ASSIGNMENT NO 1

1. Consider a student database of SY COMP class (at least 15 records). Database contains different fields of every student like Roll No, Name and SGPA.
- Design a roll call list, arrange list of students according to roll numbers in ascending order using Insertion Sort
 - Arrange list of students alphabetically using shell sort
 - Arrange list of students to find out first ten toppers from a class using Radixsort

Code a)

```
#include <iostream>

#include <vector>

using namespace std;

class Student {

public:

    int rollNo;

    string name;

    float sgpa;

    Student(int rollNo, string name, float sgpa) {

        this->rollNo = rollNo;

        this->name = name;

        this->sgpa = sgpa;

    }

    void display() const {

        cout << "Roll No: " << rollNo << ", Name: " << name << ", SGPA: " << sgpa <<

endl;

    }

};

void insertionSort(vector<Student>& students) {

    int n = students.size();

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

        Student key = students[i];
```

```
int j = i - 1;

while (j >= 0 && students[j].rollNo > key.rollNo) {

    students[j + 1] = students[j];

    j = j - 1;

}

students[j + 1] = key;

}
```

```
int main() {

    vector<Student> students = {

        Student(12, "Aryan", 8.2),

        Student(7, "Pooja", 9.1),

        Student(15, "Rahul", 7.9),

        Student(9, "Neha", 8.5),

        Student(3, "Kiran", 8.0),

        Student(10, "Aditi", 9.0),

        Student(5, "Rohan", 8.8),

        Student(13, "Manoj", 7.5),

    };

}
```

```
cout << "Before Sorting:" << endl;

for (const auto& student : students) {

    student.display();

}

insertionSort(students);
```

```
cout << "\nAfter Sorting by Roll No:" << endl;

for (const auto& student : students) {

    student.display();

}
```

```
}
```

```
return 0;
```

```
}
```

Output :

Before Sorting:

Roll No: 12, Name: Aryan, SGPA: 8.2

Roll No: 7, Name: Pooja, SGPA: 9.1

Roll No: 15, Name: Rahul, SGPA: 7.9

Roll No: 9, Name: Neha, SGPA: 8.5

Roll No: 3, Name: Kiran, SGPA: 8

Roll No: 10, Name: Aditi, SGPA: 9

Roll No: 5, Name: Rohan, SGPA: 8.8

Roll No: 13, Name: Manoj, SGPA: 7.5

After Sorting by Roll No:

Roll No: 3, Name: Kiran, SGPA: 8

Roll No: 5, Name: Rohan, SGPA: 8.8

Roll No: 7, Name: Pooja, SGPA: 9.1

Roll No: 9, Name: Neha, SGPA: 8.5

Roll No: 10, Name: Aditi, SGPA: 9

Roll No: 12, Name: Aryan, SGPA: 8.2

Roll No: 13, Name: Manoj, SGPA: 7.5

Roll No: 15, Name: Rahul, SGPA: 7.9

main.cpp

```
1 #include <iostream>
2 #include <vector>
3 using namespace std;
4 class Student {
5 public:
6     int rollNo;
7     string name;
8     float sgpa;
9     Student(int rollNo, string name, float sgpa) {
10         this->rollNo = rollNo;
11         this->name = name;
12         this->sgpa = sgpa;
13     }
14     void display() const {
15         cout << "Roll No: " << rollNo << ", Name: " << name << ", SGPA: " << sgpa << endl;
16     }
17 };
18 void insertionSort(vector<Student>& students) {
19     int n = students.size();
20     for (int i = 1; i < n; ++i) {
21         Student key = students[i];
22         int j = i - 1;
23         while (j >= 0 && students[j].rollNo > key.rollNo) {
24             students[j + 1] = students[j];
25             j = j - 1;
26         }
27         students[j + 1] = key;
28     }
29 }
30 int main() {
31     vector<Student> students = {
32         Student(12, "Aryan", 8.2),
33         Student(7, "Pooja", 9.1),
34         Student(15, "Rahul", 7.9),
35         Student(9, "Neha", 8.5),
36         Student(3, "Kiran", 8.0),
37         Student(10, "Aditi", 9.0),
38         Student(5, "Rohan", 8.8),
39         Student(13, "Manoj", 7.5),
40     };
41
42     cout << "Before Sorting:" << endl;
43     for (const auto& student : students) {
44         student.display();
45     }
46
47     insertionSort(students);
48
49     cout << "\nAfter Sorting by Roll No:" << endl;
50     for (const auto& student : students) {
51         student.display();
52     }
53
54     return 0;
55 }
```

Output

```
/tmp/7FNZna0KnC.o
Before Sorting:
Roll No: 12, Name: Aryan, SGPA: 8.2
Roll No: 7, Name: Pooja, SGPA: 9.1
Roll No: 15, Name: Rahul, SGPA: 7.9
Roll No: 9, Name: Neha, SGPA: 8.5
Roll No: 3, Name: Kiran, SGPA: 8
Roll No: 10, Name: Aditi, SGPA: 9
Roll No: 5, Name: Rohan, SGPA: 8.8
Roll No: 13, Name: Manoj, SGPA: 7.5

After Sorting by Roll No:
Roll No: 3, Name: Kiran, SGPA: 8
Roll No: 5, Name: Rohan, SGPA: 8.8
Roll No: 7, Name: Pooja, SGPA: 9.1
Roll No: 9, Name: Neha, SGPA: 8.5
Roll No: 10, Name: Aditi, SGPA: 9
Roll No: 12, Name: Aryan, SGPA: 8.2
Roll No: 13, Name: Manoj, SGPA: 7.5
Roll No: 15, Name: Rahul, SGPA: 7.9

=== Code Execution Successful ===
```

Code b)

```
#include <iostream>
```

```
#include <vector>
```

```
using namespace std;
```

```
class Student {
```

```
public:
```

```
    int rollNo;
```

```
    string name;
```

```
    float sgpa;
```

```
    Student(int rollNo, string name, float sgpa) {
```

```
        this->rollNo = rollNo;
```

```
        this->name = name;
```

```

    this->sgpa = sgpa;

}

void display() const {

    cout << "Roll No: " << rollNo << ", Name: " << name << ", SGPA: " << sgpa << endl;

}

};

void shellSort(vector<Student>& students) {

    int n = students.size();

    for (int gap = n / 2; gap > 0; gap /= 2) {

        for (int i = gap; i < n; i++) {

            Student temp = students[i];

            int j;

            for (j = i; j >= gap && students[j - gap].name > temp.name; j -= gap) {

                students[j] = students[j - gap];

            }

            students[j] = temp;

        }

    }

}

int main() {

    vector<Student> students = {

        Student(12, "Aryan", 8.2),

```

```
Student(7, "Pooja", 9.1),
```

```
Student(15, "Rahul", 7.9),
```

```
Student(9, "Neha", 8.5),
```

```
Student(3, "Kiran", 8.0),
```

```
Student(10, "Aditi", 9.0),
```

```
Student(5, "Rohan", 8.8),
```

```
};
```

```
cout << "Before Sorting:" << endl;
```

```
for (const auto& student : students) {
```

```
    student.display();
```

```
}
```

```
shellSort(students);
```

```
cout << "\nAfter Sorting by Name:" << endl;
```

```
for (const auto& student : students) {
```

```
    student.display();
```

```
}
```

```
return 0;
```

```
}
```

Output :

Before Sorting:

Roll No: 12, Name: Aryan, SGPA: 8.2

Roll No: 7, Name: Pooja, SGPA: 9.1

Roll No: 15, Name: Rahul, SGPA: 7.9

Roll No: 9, Name: Neha, SGPA: 8.5

Roll No: 3, Name: Kiran, SGPA: 8

Roll No: 10, Name: Aditi, SGPA: 9

Roll No: 5, Name: Rohan, SGPA: 8.8

After Sorting by Name:

Roll No: 10, Name: Aditi, SGPA: 9

Roll No: 12, Name: Aryan, SGPA: 8.2

Roll No: 3, Name: Kiran, SGPA: 8

Roll No: 9, Name: Neha, SGPA: 8.5

Roll No: 7, Name: Pooja, SGPA: 9.1

Roll No: 15, Name: Rahul, SGPA: 7.9

Roll No: 5, Name: Rohan, SGPA: 8.8

main.cpp

```
1 #include <iostream>
2 #include <vector>
3 using namespace std;
4 class Student {
5 public:
6     int rollNo;
7     string name;
8     float sgpa;
9     Student(int rollNo, string name, float sgpa) {
10         this->rollNo = rollNo;
11         this->name = name;
12         this->sgpa = sgpa;
13     }
14     void display() const {
15         cout << "Roll No: " << rollNo << ", Name: " << name << ", SGPA: " << sgpa << endl;
16     }
17 };
18 void shellSort(vector<Student>& students) {
19     int n = students.size();
20     for (int gap = n / 2; gap > 0; gap /= 2) {
21         for (int i = gap; i < n; i++) {
22             Student temp = students[i];
23             int j;
24             for (j = i; j >= gap && students[j - gap].name > temp.name; j -= gap) {
25                 students[j] = students[j - gap];
26             }
27             students[j] = temp;
28         }
29     }
30 }
31 int main() {
32     vector<Student> students = {
33         Student(12, "Aryan", 8.2),
34         Student(7, "Pooja", 9.1),
35         Student(15, "Rahul", 7.9),
36         Student(9, "Neha", 8.5),
37         Student(3, "Kiran", 8.0),
38         Student(10, "Aditi", 9.0),
39         Student(5, "Rohan", 8.8),
40     };
41     cout << "Before Sorting:" << endl;
42     for (const auto& student : students) {
43         student.display();
44     }
45     shellSort(students);
46     cout << "\nAfter Sorting by Name:" << endl;
47     for (const auto& student : students) {
48         student.display();
49     }
50     return 0;
51 }
52
53
54
55
```

Output

```
/tmp/rhf69j0Gvc.o
Before Sorting:
Roll No: 12, Name: Aryan, SGPA: 8.2
Roll No: 7, Name: Pooja, SGPA: 9.1
Roll No: 15, Name: Rahul, SGPA: 7.9
Roll No: 9, Name: Neha, SGPA: 8.5
Roll No: 3, Name: Kiran, SGPA: 8
Roll No: 10, Name: Aditi, SGPA: 9
Roll No: 5, Name: Rohan, SGPA: 8.8

After Sorting by Name:
Roll No: 10, Name: Aditi, SGPA: 9
Roll No: 12, Name: Aryan, SGPA: 8.2
Roll No: 3, Name: Kiran, SGPA: 8
Roll No: 9, Name: Neha, SGPA: 8.5
Roll No: 7, Name: Pooja, SGPA: 9.1
Roll No: 15, Name: Rahul, SGPA: 7.9
Roll No: 5, Name: Rohan, SGPA: 8.8

|
=== Code Execution Successful ===
```

Code c)

```
#include <iostream>
```

```
#include <vector>
```

```
using namespace std;
```

```
class Student {
```

```
public:
```

```
    int rollNo;
```

```
    string name;
```

```
    float sgpa;
```

```
    Student() : rollNo(0), name(""), sgpa(0.0) {}
```

```
    Student(int rollNo, string name, float sgpa) : rollNo(rollNo), name(name), sgpa(sgpa) {}
```

```
    void display() const {
```

```
        cout << rollNo << " " << name << " " << sgpa << endl;
```

```

    }

};

float getMaxSGPA(vector<Student>& students) {

    float maxSGPA = students[0].sgpa;

    for (const auto& student : students) {

        if (student.sgpa > maxSGPA)

            maxSGPA = student.sgpa;

    }

    return maxSGPA;

}

```

```

void countingSort(vector<Student>& students, int exp) {

    int n = students.size();

    vector<Student> output(n);

    int count[10] = {0};

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

        int sgpaAsInt = int(students[i].sgpa * 100);

        count[(sgpaAsInt / exp) % 10]++;

    }

    for (int i = 1; i < 10; i++)

        count[i] += count[i - 1];
}

```

```

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

    int sgpaAsInt = int(students[i].sgpa * 100);

    output[count[(sgpaAsInt / exp) % 10] - 1] = students[i];

    count[(sgpaAsInt / exp) % 10]--;

}

for (int i = 0; i < n; i++)

    students[i] = output[i];

}

void radixSort(vector<Student>& students) {

    float maxSGPA = getMaxSGPA(students);

    for (int exp = 1; int(maxSGPA * 100) / exp > 0; exp *= 10)

        countingSort(students, exp);

}

int main() {

    vector<Student> students = {

        {12, "Aryan", 8.2}, {7, "Pooja", 9.1}, {15, "Rahul", 7.9}, {9, "Sneha", 8.5},

        {3, "Kiran", 8.0}, {10, "Aditi", 9.0}, {5, "Rohan", 8.8}, {13, "Manoj", 7.5},

        {6, "Priya", 9.2}, {11, "Nikhil", 7.8}, {1, "Tanvi", 8.6}, {8, "Suresh", 7.7},

        {4, "Geeta", 8.1}, {14, "Vikas", 8.4}, {2, "Neha", 9.0}

    };

```

```
radixSort(students);

for (int i = students.size() - 1, count = 0; i >= 0 && count < 10; i--, count++)

    students[i].display();

return 0;

}
```

Output :

6 Priya 9.2

7 Pooja 9.1

2 Neha 9

10 Aditi 9

5 Rohan 8.8

1 Tanvi 8.6

9 Sneha 8.5

14 Vikas 8.4

12 Aryan 8.2

4 Geeta 8.1

main.cpp

```
1 #include <iostream>
2 #include <vector>
3 using namespace std;
4
5 class Student {
6 public:
7     int rollNo;
8     string name;
9     float sgpa;
10    Student() : rollNo(0), name(""), sgpa(0.0) {}
11    Student(int rollNo, string name, float sgpa) : rollNo(rollNo), name(name), sgpa(sgpa) {}
12    void display() const {
13        cout << rollNo << " " << name << " " << sgpa << endl;
14    }
15 };
16
17 float getMaxSGPA(vector<Student>& students) {
18     float maxSGPA = students[0].sgpa;
19     for (const auto& student : students) {
20         if (student.sgpa > maxSGPA)
21             maxSGPA = student.sgpa;
22     }
23     return maxSGPA;
24 }
25
26 void countingSort(vector<Student>& students, int exp) {
27     int n = students.size();
28     vector<Student> output(n);
29     int count[10] = {0};
30     for (int i = 0; i < n; i++) {
31         int sgpaAsInt = int(students[i].sgpa * 100);
32         count[(sgpaAsInt / exp) % 10]++;
33     }
34     for (int i = 1; i < 10; i++)
35         count[i] += count[i - 1];
36     for (int i = n - 1; i >= 0; i--) {
37         int sgpaAsInt = int(students[i].sgpa * 100);
38         output[count[(sgpaAsInt / exp) % 10] - 1] = students[i];
39         count[(sgpaAsInt / exp) % 10]--;
40     }
41     for (int i = 0; i < n; i++)
42         students[i] = output[i];
43 }
44
45 void radixSort(vector<Student>& students) {
46     float maxSGPA = getMaxSGPA(students);
47     for (int exp = 1; int(maxSGPA * 100) / exp > 0; exp *= 10)
48         countingSort(students, exp);
49 }
50
51 int main() {
52     vector<Student> students = {
53         {12, "Aryan", 8.2}, {7, "Pooja", 9.1}, {15, "Rahul", 7.9}, {9, "Sneha", 8.5},
54         {3, "Kiran", 8.0}, {10, "Aditi", 9.0}, {5, "Rohan", 8.8}, {13, "Manoj", 7.5},
55         {6, "Priya", 9.2}, {11, "Nikhil", 7.8}, {1, "Tanvi", 8.6}, {8, "Suresh", 7.7},
56         {4, "Geeta", 8.1}, {14, "Vikas", 8.4}, {2, "Neha", 9.0}
57     };
58     radixSort(students);
59     for (int i = students.size() - 1; count = 0; i >= 0 && count < 10; i--, count++)
60         students[i].display();
61     return 0;
62 }
63
```

Output

/tmp/JSB71SYDCY.o

```
6 Priya 9.2
7 Pooja 9.1
2 Neha 9
10 Aditi 9
5 Rohan 8.8
1 Tanvi 8.6
9 Sneha 8.5
14 Vikas 8.4
12 Aryan 8.2
4 Geeta 8.1
```

=== Code Execution Successful ===