**Department of Computer Science & Engineering**

**DATA STRUCTURES**

**(13 ES 204)**

**PROJECT**

**Student Information System**

**S.NO ID NUMBER NAME**

**1 14003147 M.VISHNU PRIYA**

**2 14003347 V.VAISHNAVI**

**3 14003543 M.YAMINI PRIYANKA**

**4 14003424 A.V.GAUTHAM**

**ABSTRACT**

Student Information System (SIS) provides a simple interface for maintenance of student information. It can be used by educational institutes or colleges to maintain the records of students easily. The creation and management of accurate, up-to-date information regarding a students’ academic career is critically important in the university as well as colleges. Student information system deals with all kind of student details such as student rollno, student name ,student marks. The 80 students of a section can be taken into account. Their name, rollno and marks can be stored. The names can be printed in ascending order of average marks. The average marks can be calculated. The addition and/or deletion of new students can be updated.

Module 1:

|  |  |  |  |
| --- | --- | --- | --- |
| Serial  Number | Name of the  Module | Function  Number | Functions to be discharged |
| 1 | Create lists | #1. | Define list ADT |
| #2. | Create list1 with the rollnos |
| #3. | Create the list2 with names |
| #4. | Create list3 with marks |
| #5. | Display the individual lists |
| #6. | Merge all the lists |
| #7. | Display the merged list |
| #8. | Integrate all the above functions into a single program module. |

## Module-2 Specification: Search

|  |  |  |  |
| --- | --- | --- | --- |
| Serial  Number | Name of the  Module | Function  Number | Functions to be discharged |
| 2 | Search | #1. | Implement linear search |
| #2. | Search the details of a student given with the roll no from all the three lists using linear search. |
| #3. | Implement binary search |
| #4. | Search the details of a student given with the name from all the three lists using binary search. |
| #5. | For applying binary search select the sorting algorithm |
| #6. | Find the complexity in linear search case |
| #7. | Find the complexity in binary search case |
| #8. | Integrate both the searchings into a single program. |

## Module-3 Specification: **Sort**

|  |  |  |  |
| --- | --- | --- | --- |
| Serial  Number | Name of the  Module | Function  Number | Functions to be discharged |
| 3 | Sort | #1. | Define a sorting method |
| #2. | Sort the three lists as per name. |
| #3. | Display all these lists after sorting |
| #4. | Merge the lists into a single list |
| #5. | Sort as per as per marks |
| #6. | Display lists |
| #7. | Merge the lists into a single list |
| #8. | Keep all these together as a single program |

## Module-4 Specification: **Add/Delete**

|  |  |  |  |
| --- | --- | --- | --- |
| Serial  Number | Name of the  Module | Function  Number | Functions to be discharged |
| 4 | Add/Delete | #1. | Design the pseudo code for the Delete operation |
| #2. | Apply delete operation of one student |
| #3. | Display the lists |
| #4. | Add the new student into lists |
| #5. | Display lists |
| #6. | After adding new data merge all the three lists |
| #7. | Show the final merged list |
| #8. | Integrate all these into a single program |