**CAPSTONE DOCUMENTATION**

**TEAM: DEMON SLAYERS**

**SECTION: A**

**GROUP MEMBERS: ANUSHKA RAWAT- 202301191**

**HARSHVIR SINGH- 202301080**

**PREXA PATEL- 202301181**

**SHAMBHAVI SINGH- 202301107**

**PROBLEM NAME: DA-IICT CLUB MANAGER**

**PROBLEM NUMBER: P6**

**PROBLEM STATEMENT: You need to build a manager for all the DA-IICT clubs. The manager ensures that a club member can be looked up in minimum time. A member can either be a faculty or a student. One should be able to search by name, ID, specific club name, or club category (i.e., arts, science & technology, sports, culture). Note that the user of this manager may not be a DA-IICT-ian and, therefore, may not know the clubs’ names.**

**GIT-HUB LINK:** [**PROJECT CAPSTONE**](https://github.com/Anushkarawat4/P-6-DEMON-SLAYER.git)

# **PSEUDOCODE**

* FUNCTION 1: readstudent\_info

Input- Input file which contains data of students

Output- First unordered map with key as student name and second unordered map with key as student ID

Start

Open file "membership\_data.txt" in read mode

Initialize an empty map named s\_names

Initialize an empty map named s\_ids

While (not end of file and current line is not "[Members]")

Read the next line from the file

EndWhile

While (not end of file and current line is not empty)

Read the next line from the file

If (the line does not contain "ID,Name")

Extract ID and Name from the line

If (extraction successful)

Add ID and Name to s\_names map

Add Name and ID to s\_ids map

Else

Print "Error reading line: " followed by the current line

EndIf

EndIf

EndWhile

Close the file

Stop

* FUNCTION 2: read\_stuMemberships

Input- Input file which contains data of students

Output- Unordered map with key as student ID

Start

Open file "membership\_data.txt" in read mode

Initialize an empty map named s\_membership

While (not end of file and current line is not "[Memberships]")

Read the next line from the file

EndWhile

While (not end of file and current line is not empty)

Read the next line from the file

If (the line does not contain "Member ID,Club Name,Role")

Extract ID, Club Name, and Role from the line

If (extraction successful)

Append Club Name and Role (if available) to the membership vector of the corresponding ID

Else

Print "Error reading line: " followed by the current line

EndIf

EndIf

EndWhile

Close the file

Stop

* FUNCTION 3: searchStudentByID

Input- Student Id

Output- Club information of the student

Start

Find the student with the given ID in s\_names

Find the student's club memberships in s\_membership

If (the student is found in s\_names)

Print "Student found:"

Print "ID: " followed by the student's ID

Print "Name: " followed by the student's name

If (the student is a member of any club)

Print "Memberships:"

For (each membership in the student's memberships)

Print the membership

EndFor

Else

Print "No memberships found."

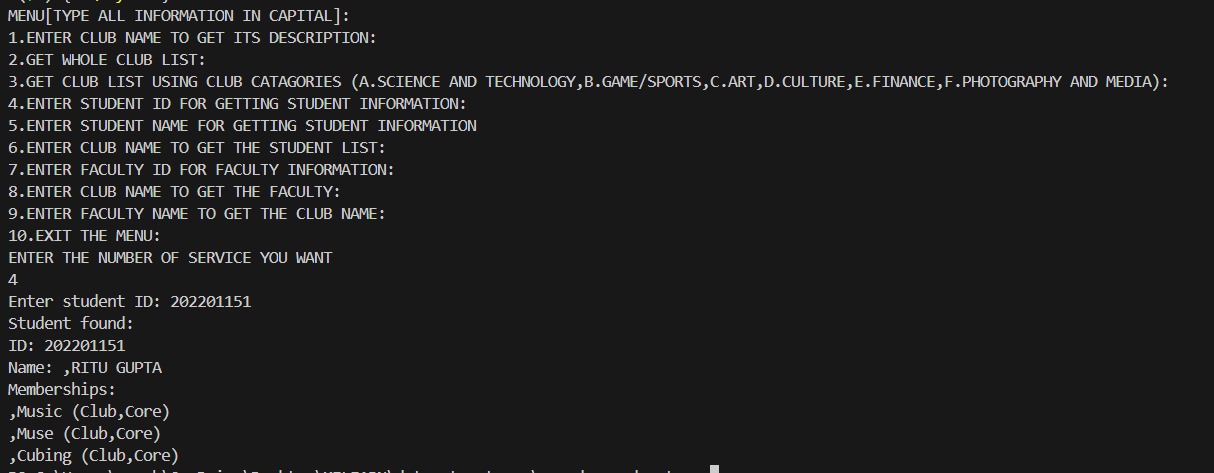
EndIf

Else

Print "Student with ID " followed by the given ID " not found."

EndIf

Stop



* FUNCTION 4: searchStudentByName

Input- Student name

Output- Club information of the student

Start

Initialize a boolean variable named found and set it to false

For (each student in s\_names)

If (the student's name contains the given name)

Get the ID of the student from s\_ids

Call searchStudentByID(s\_names, s\_membership, id) with the found ID

Set found to true

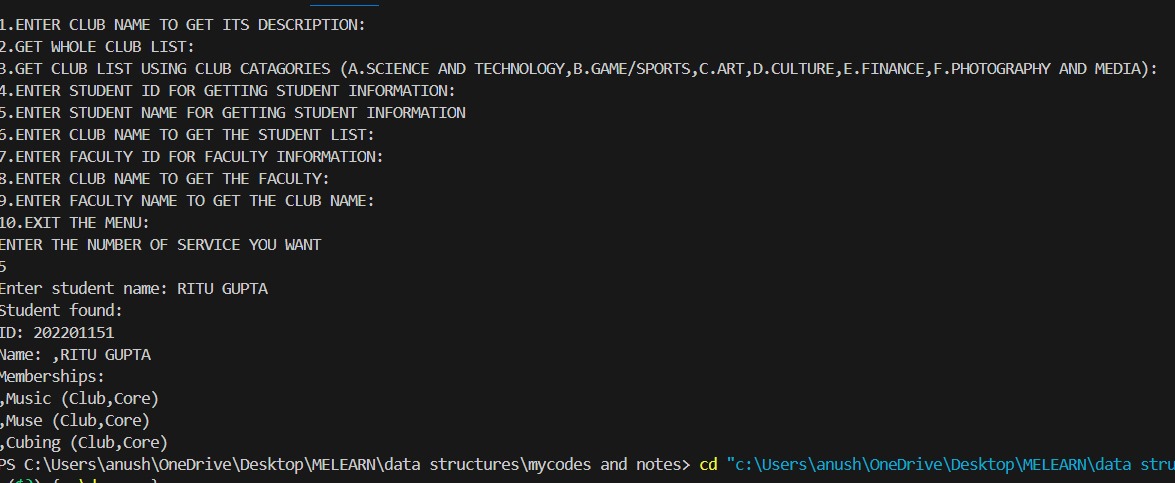
Break from the loop

If (found is false)

Print "Student with name " followed by the given name " not found."

End

Stop



* FUNCTION 5: displayClubDescription

Input- Club name

Output- Introduction to the given club

Start

Create an empty map named, “clubmap”

Open file "daiictclubmanager.txt" in read mode

While (file does not reach end)

Read a line and store it in string variable clubname

Read a line and store it in string variable description

Read a line and store it in string variable line

Add the key-value pair (clubname, description) to clubmap

EndWhile

Close file

Print "Enter the name of the club to get its description: "

Take input from user and store it in string variable inputclubname

Search for inputclubname in clubmap

If ( inputclubname matches a key in clubmap)

Print "Club Description: “

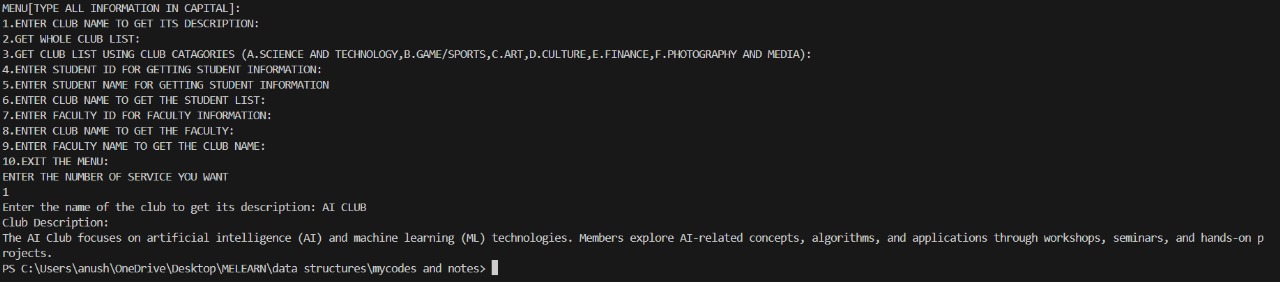
Print the value associated with the key

Else

Print "Club not found."

EndIf

Stop



* FUNCTION 6: displayclublist

Input- “Display List of all the clubs in DAIICT”

Output- List of all the clubs in DAIICT

Start

Open file "daiictclubmanager.txt" in read mode

For (integer variable i = 1; i less than or equal to 19)

Read a line and store it in string variable clubname

Read a line and store it in string variable description

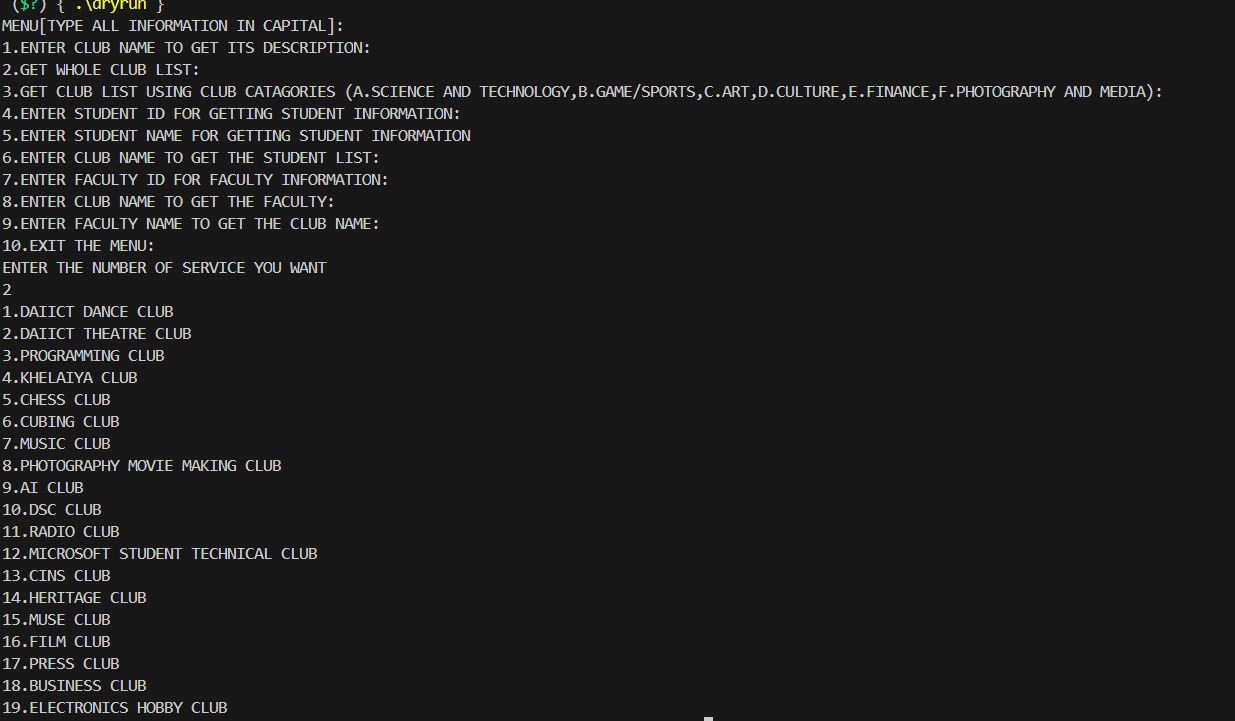
Read a line and store it in string variable line

Print “i. clubname”

EndFor

Close File

Stop



* FUNCTION 7: clubusingcat

Input- A, B, C, D, E, F

Output- Names of clubs categorized into the groups, Science and Technology, Sports, Art, Culture, Finance, Photography and Media

Start

Open file "daiictclubmanagercat.txt" in read mode

Print “ENTER (A,B,C,D,E,F):"

Take input from user and store it in character variable q

Read a line and store it in string variable s1

Read a line and store it in string variable s2

Read a line and store it in string variable s3

Read a line and store it in string variable s4

Read a line and store it in string variable s5

Read a line and store it in string variable s6

If (q is equal to ‘A’)

Print string variable s1

ElseIf (q is equal to ‘B’)

Print string variable s2

ElseIf (q is equal to ‘C’)

Print string variable s3

ElseIf (q is equal to ‘D’)

Print string variable s4

ElseIf (q is equal to ‘E’)

Print string variable s5

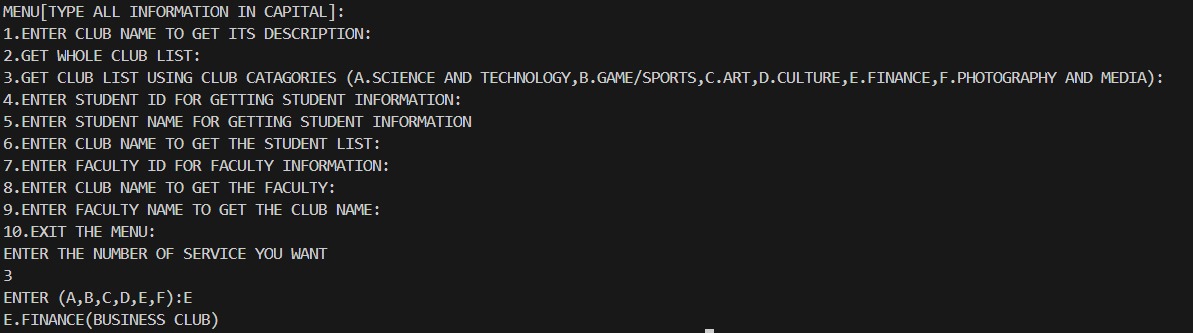
ElseIf (q is equal to ‘F’)

Print string variable s6

EndIf

Close file

Stop



* FUNCTION 8: getinfobyclubname

Input- Club name

Output- Information about faculty mentor of the given club

Start

Create an empty map named, “clubmap”

Open file " facultymentor.txt" in read mode

While (file does not reach end)

Read a line and store it in string variable clubname

Read a line and store it in string variable facultyname

Read a line and store it in string variable id

Store facultyname and id in string variable info

Add the key-value pair (clubname, info) to clubmap

EndWhile

Close file

Print " ENTER CLUB NAME IN CAPITAL TO GET FACULTY MENTOR INFORMATION: "

Take input from user and store it in string variable inputclubname

Search for inputclubname in clubmap

If (inputclubname matches a key in clubmap)

Print " FACULTY INFORMATION IS AS FOLLOWS: “

Print the value associated with the key

Else

Print " CLUB NAME NOT FOUND IN DATABASE."

EndIf

Stop



* FUNCTION 9: getinfobyfacultyname

Input- Faculty name

Output- Information about club of the given faculty

Start

Create an empty map named, “clubmap”

Open file " facultymentor.txt" in read mode

While (file does not reach end)

Read a line and store it in string variable clubname

Read a line and store it in string variable facultyname

Read a line and store it in string variable id

Store clubname and id in string variable info

Add the key-value pair (facultyname, info) to clubmap

EndWhile

Close file

Print ” ENTER FACULTY MENTOR'S NAME TO GET FACULTY INFORMATION: "

Take input from user and store it in string variable inputclubname

Search for inputclubname in clubmap

If ( inputclubname matches a key in clubmap)

Print " FACULTY INFORMATION IS AS FOLLOWS: “

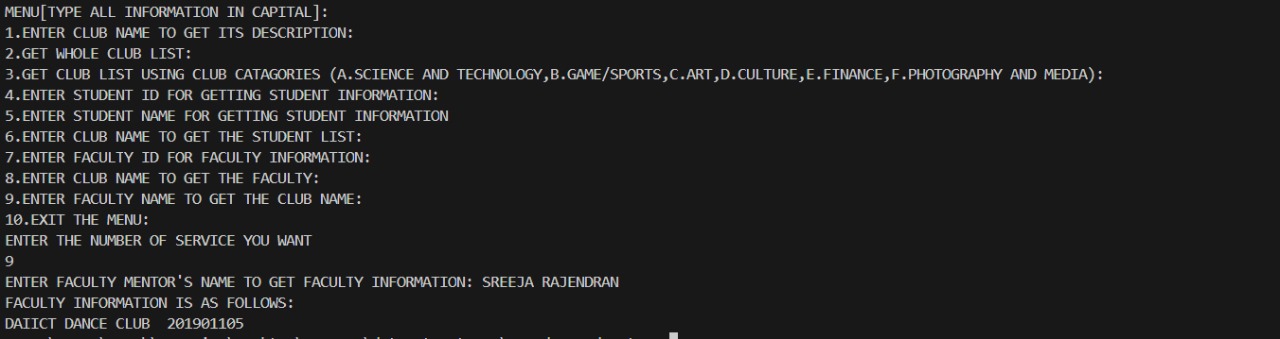
Print the value associated with the key

Else

Print " FACULTY NAME NOT FOUND IN DATABASE."

EndIf

Stop



* FUNCTION 10: getinfobyfacultyid

Input- Faculty ID

Output- Information about club of the given faculty ID

Start

Create an empty map named, “clubmap”

Open file " facultymentor.txt" in read mode

While (file does not reach end)

Read a line and store it in string variable clubname

Read a line and store it in string variable facultyname

Read a line and store it in string variable id

Store clubname and facultyname in string variable info

Add the key-value pair (facultyid, info) to clubmap

EndWhile

Close file

Print ” ENTER FACULTY MENTOR'S ID TO GET FACULTY INFORMATION: "

Take input from user and store it in string variable inputclubname

Search for inputclubname in clubmap

If ( inputclubname matches a key in clubmap)

Print " FACULTY INFORMATION IS AS FOLLOWS: “

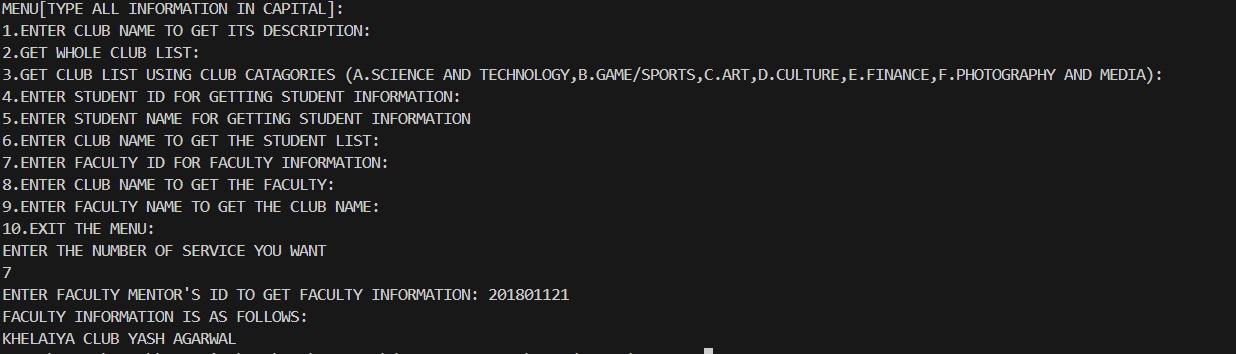
Print the value associated with the key

Else

Print " FACULTY ID NOT FOUND IN DATABASE."

EndIf

Stop



* FUNCTION 11: studentlistby\_club

Input- Club name

Output- Information about the members of the club

Start

In read mode Open file "student\_dataclub.txt" in read mode

Set found to false

While (read a line and store it in string variable line)

If (the line is equal to ("#" + club\_name))

Set found to true

Print "Student list for club '" followed by club\_name and "':" followed by a new line

Print "->" followed by a new line

While (read a line and store it in string variable line and it is not the end of the file)

Print "-> " followed by line and a new line

Find the position of comma in line

If (comma position is found)

Extract student information from line

Extract student ID from student information

Extract student name from line

Find student in s\_names using student ID

If (student is found)

Print "Student ID: " followed by student ID ", Student Name: " followed by student name and a new line

Break the loop

EndIf

EndIf

EndIf

EndWhile

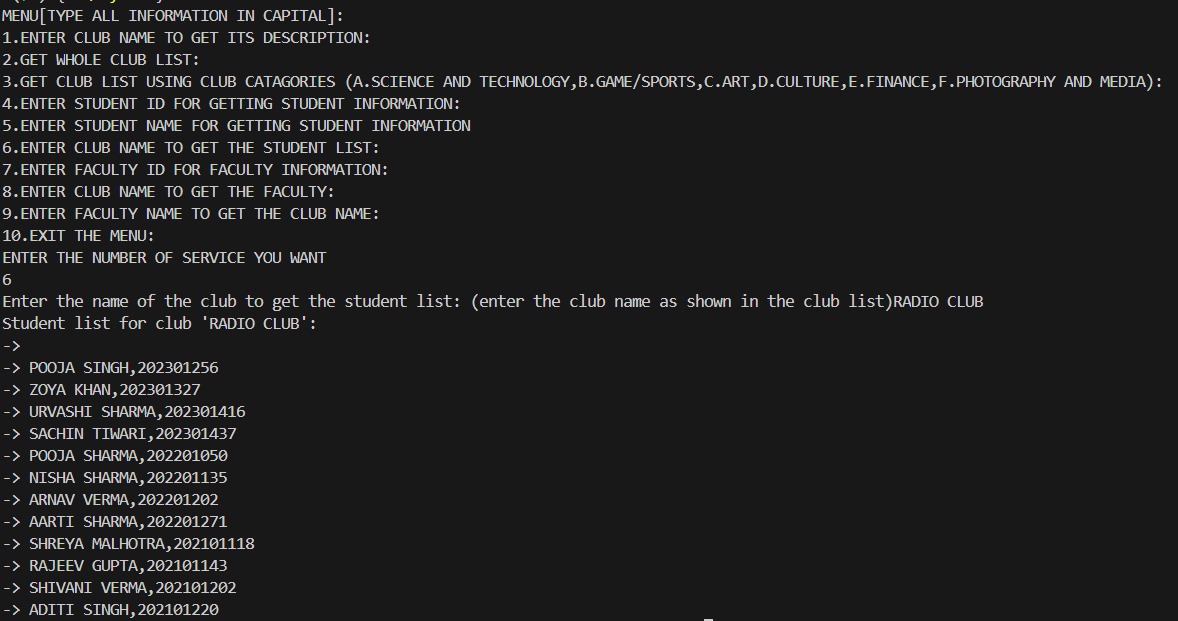
Close file

If (not found)

Print "Club '" followed by club\_name, "' not found or has no members." and a new line

EndIf

Stop



* FUNCTION 12: int main

Input- Number of the service (from 1 to 10) the user requires

Output- According to the service/services selected

Start

Create an empty map, named s\_names, to store student names

Create an empty map, named s\_memberships, to store names of club that students are part of

Create an empty map, s\_ids, to store student IDs

Call function readstudent\_info and pass s\_names and s\_ids as arguments

Call function read\_stuMemberships and pass s\_memberships as argument

While (option is not equal to 10)

Print "MENU[TYPE ALL INFORMATION IN CAPITAL]:"

Print "1.ENTER CLUB NAME TO GET ITS DESCRIPTION:"

Print "2.GET WHOLE CLUB LIST:"

Print "3.GET CLUB LIST USING CLUB CATAGORIES (A.SCIENCE AND TECHNOLOGY, B.GAME/SPORTS, C.ART,D.CULTURE, E.FINANCE,F.PHOTOGRAPHY AND MEDIA):”

Print "4.ENTER STUDENT ID FOR GETTING STUDENT INFORMATION:"

Print "5.ENTER STUDENT NAME FOR GETTING STUDENT INFORMATION" << endl;

Print "6.ENTER CLUB NAME TO GET THE STUDENT LIST:"

Print "7.ENTER FACULTY ID FOR FACULTY INFORMATION:"

Print "8.ENTER CLUB NAME TO GET THE FACULTY:" << endl;

Print "9.ENTER FACULTY NAME TO GET THE CLUB NAME: "

Print "10.EXIT THE MENU:"

Print "ENTER THE NUMBER OF SERVICE YOU WANT"

Take input option from user and store it in integer variable option

Switch(option)

Case 1:

Call the function displayClubDescription

Break from the loop

Case 2:

Call the function displayclublist

Break from the loop

Case 3:

Call the function clubusingcat

Break from the loop

Case 4:

Print "Enter student ID: "

Take input sid from user and store it in integer variable sid

Call the function searchStudentByID and pass s\_names, s\_memberships and sid as arguments

Break from the loop

Case 5:

Print "Enter student name: "

Read a line and store it in string variable sname

Call the function searchStudentByName and pass s\_names, s\_memberships, s\_ids and sname as arguments

Break from the loop

Case 6:

Print "Enter the name of the club to get the student list: (enter the club name as shown in the club list)"

Read a line and store it in string variable club\_name

Call the function studentlist\_byclub and pass s\_names and club\_name as arguments

Break from the loop

Case 7:

Call the function getinfobyfacultyid

Break from the loop

Case 8:

Call the function getinfobyclubname

Break from the loop

Case 9:

Call the function getinfobyfacultyname

Break from the loop

Case 10:

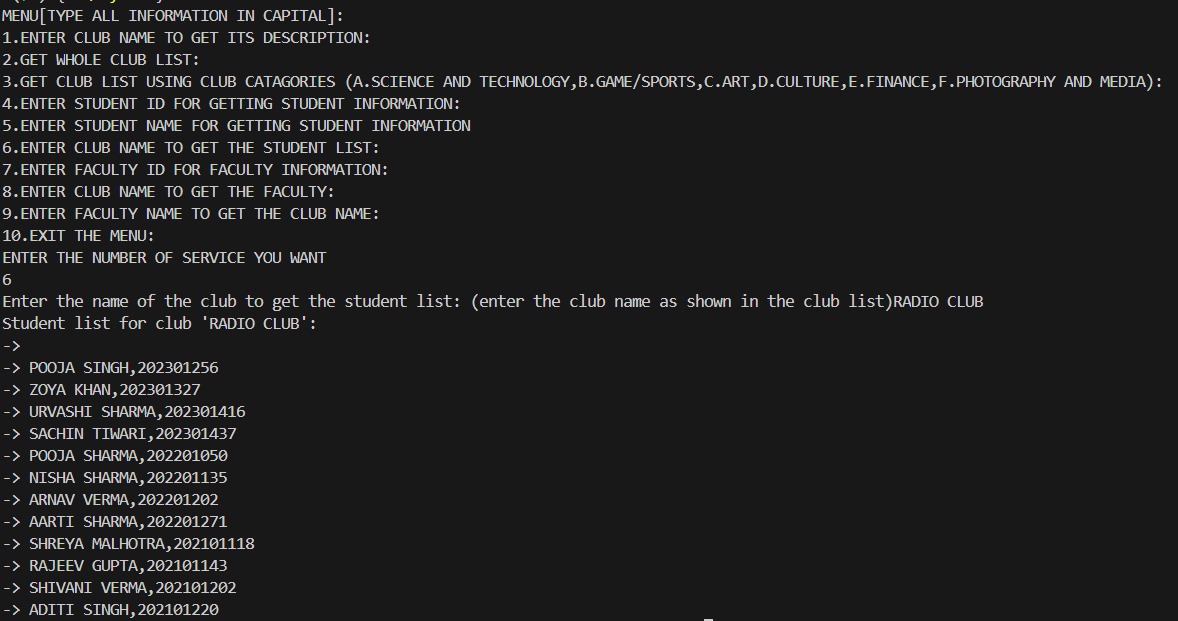
Print "EXITING MENU. THANKS FOR USING THE SERVICE"

Break from the loop

EndSwitch

EndWhile

Stop



# **TIME COMPLEXITY**

1. readstudent\_info:

O(n) where n is the number of lines in the input file, as the function reads the input file line by line and performs a constant amount of work

1. read\_stuMemberships

O(n) where n is the number of lines in the input file, as the function reads the input file line by line and performs a constant amount of work

1. searchStudentByID

O(1), as the function performs a lookup in the s\_names map, which has an average time complexity of O(1). It can be O(n) in the case where a student is part of more than 1 clubs and n is equal to the number of clubs the student is a member of.

1. searchStudentByName

O(1), as the function performs a lookup in the s\_names map, which has an average time complexity of O(1). It can be O(n) in the case where a student is part of more than 1 clubs and n is equal to the number of clubs the student is a member of.

1. displayClubDescription

O(1) for searching and O(n) for insertion where n is the number of lines in the input file, as no two club names are the same so there is no collision ensuring time complexity of O(1) and for insertion, the function reads the input file line by line and performs a constant amount of work.

1. displayclublist

O(n) where n is the number of lines in the input file, as the function traverses the whole input file and prints the club name line by line.

1. clubusingcat

O(1), as the function performs a constant amount of work regardless of the input

1. getinfobyclubname

O(1) for searching and O(n) for insertion where n is the number of lines in the input file, as no two club names are the same so there is no collision ensuring time complexity of O(1) and for insertion, the function reads the input file line by line and performs a constant amount of work.

1. getinfobyfacultyname

O(1), for searching and O(n) for insertion where n is the number of lines in the input file, as no two club names are the same so there is no collision ensuring time complexity of O(1) and for insertion, the function reads the input file line by line and performs a constant amount of work.

1. getinfobyfacultyid

O(1), for searching and O(n) for insertion where n is the number of lines in the input file, as no two club names are the same so there is no collision ensuring time complexity of O(1) and for insertion, the function reads the input file line by line and performs a constant amount of work.

1. studentlist\_byclub

In the worst case, the function may have to read and process every line in the file to find the club with the given name. So, the time complexity of the studentlist\_byname function is O(n), where n is the number of lines in the "student\_dataclub.txt" file.

# **SPACE COMPLEXITY**

1. readstudent\_info:

O(m) where m is the number of students in the input file.

1. read\_stuMemberships

O(m\*(k+l)) where m is the number of students IDs, k is the number of unique names and l is the number of roles in the input file.

1. searchStudentByID

O(1)

1. searchStudentByName

O(1)

1. displayClubDescription

O(m) where m is the number of clubs in the input file.

1. displayclublist

O(m) where m is the number of clubs in the input file.

1. clubusingcat

O(m) where m is the number of categories in the input file.

1. getinfobyclubname

O(m) where m is the number of clubs in the input file.

1. getinfobyfacultyname

O(m) where m is the number of faculty names in the input file.

1. getinfobyfacultyid

O(m) where m is the number of faculty IDs in the input file.

1. studentlist\_byclub

O(m+n) where m is the number of student IDs and n is the number of lines in the input file.

# **DATA STRUCTURE USED**

* The data structure used in this project is Hash Table. Hash table can be implemented in c++ using class and using standard template library (STL) as well. We have used STL in our project. The STL used to implement hash table is called unordered map.
* Hash table is a data structure which stores key-value pairs and each key is translated by a hash function into a distinct index in an array, so the key values themselves become the index of the array which stores the data. This ensures faster insertion and searching of a data element. While searching, the key is hashed and the resulting hash indicates where the corresponding value is stored.
* The main objective of our project was to minimise the search time of a club member. Unordered map insures very fast lookup and the time complexity in average case is O(1). Thus, unordered map was the best possible option to carry out our project.
* Unordered map has a worst-case complexity of O(n), however this usually happens when a collision occurs for every insertion in the map. This case is not very likely to happen in our project as all student IDs are unique and it is very less likely that a student is a member of all the clubs.
* Therefore, even if a collision is to happen, it can never be the case that a collision happens on every insertion. This implies that average time complexity for searching is O(1) in our project which serves our initial objective of minimised search time.

# **HARD PROBLEM STATEMENT**

Q. Design a money distribution system for the clubs at DAIICT, where funds are allocated among the clubs based on four factors: the number of large-scale events (weight: 5), medium-scale events (weight: 3), small-scale events (weight: 1.5), and competition bonus (weight: 0.5). Additionally, each club should receive at least 30% of the total funds divided by the number of clubs. Read data from a input file. Each club's data should include the number of large-scale events, medium-scale events, small-scale events, and competition bonus.

# **HARD PROBLEM SOLUTION**

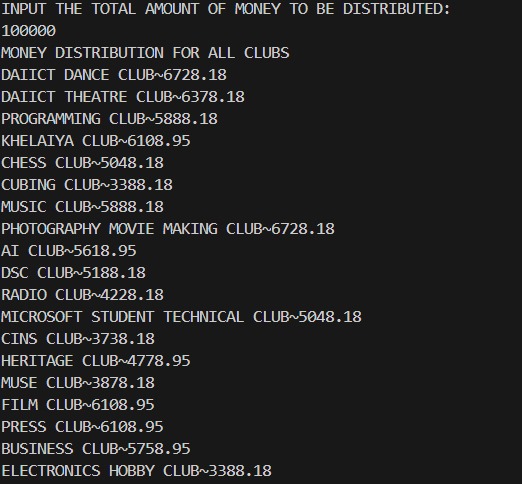
* We have provided solutions to our self-made problem statement using two different data structures.
* The first solution using a map has a time complexity of O(n) for reading the file and distributing the money, where n is the number of clubs. The space complexity is also O(n) due to the use of a map to store the club names and their corresponding money distribution.
* The second solution using a linked list has a time complexity of O(n) for reading the file and distributing the money, where n is the number of clubs. The space complexity is O(n) due to the use of a linked list to store the club information and an array to store the priority values.

# **OUTPUT OF HARD PROBLEM**

* SOLUTION 1: Using Class MoneyDivider

Input- Input file which contains data of club events

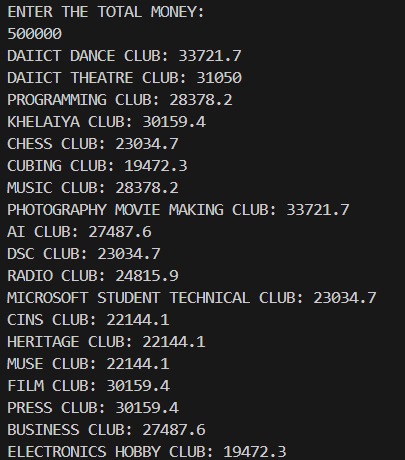
Output- Funds of each club



* SOLUTION 2: Using Singly linked list

Input- Input file which contains data of club events

Output- Funds of each club



# **ACKNOWLEDGEMENT**

We would like to express our sincere gratitude to all the people who have helped us in the completion of our project.

We would like to thank our course professor, Mr. Sourish Das Gupta and all our lab TAs.

And lastly to our team members, thank you to all teammates for their valuable efforts to achieving our project goal.

THANK YOU!

text here