

Table of Contents

[C++ Code: 1](#_Toc164729192)

[Pseudo Code: 5](#_Toc164729193)

[Flow Chart: 6](#_Toc164729194)

[GUI: 7](#_Toc164729195)

Table of figures

Figure 1 : Flow Chart 6

Figure 2 : GUI login 7

Figure 3 : GUI home 8

# C++ Code:

|  |
| --- |
| صورة تحتوي على نص, الإلكترونيات, الحاسوب, لقطة شاشة  تم إنشاء الوصف تلقائياً |

|  |
| --- |
| صورة تحتوي على نص, الإلكترونيات, لقطة شاشة, الحاسوب  تم إنشاء الوصف تلقائياً |

|  |
| --- |
| صورة تحتوي على نص, لقطة شاشة, عرض, برمجيات  تم إنشاء الوصف تلقائياً |

صورة تحتوي على نص, لقطة شاشة

تم إنشاء الوصف تلقائياً

# Pseudo Code:

structure for students containing the student's name,profession , grade, left , right.

Create a private method insert that inserts a new student into the database

Cheack is empty,create a new Student node with provided values

Cheack grade is less than node's semester, recursively insert into the left subtree.

Cheack grade is greater than node's semester, recursively insert into the right subtree.

Return the modified node.

Create a method findMin that finds the node with the minimum semester value in a subtree:

Traverse the left subtree until reaching the leaf node.

Return the node.

Create method remove that removes a student from the database:

Cheack is empty, return NULL.

Cheack grade is less than node's semester, recursively remove from the left subtree.

Cheack grade is greater than node's semester, recursively remove from the right subtree.

Cheack node is a leaf node, delete it and return NULL.

Cheack node has only one child, replace it with its child and delete the node.

Cheack node has both children, replace it with the minimum node from the right subtree and remove that node recursively.

Return the modified node.

Create methods for tree traversal

Create methods for inserting, removing, and traversing the tree:

insert Inserts a new student into the database.

remove Removes a student from the database.

preorder Performs a pre-order of the tree.

inOrder Performs an in-order of the tree.

postOrder Performs a post-order of the tree.

Create method method calculateAverageSemester

Calculate the average semester by dividing totalSemester by count.

Return the average semester.

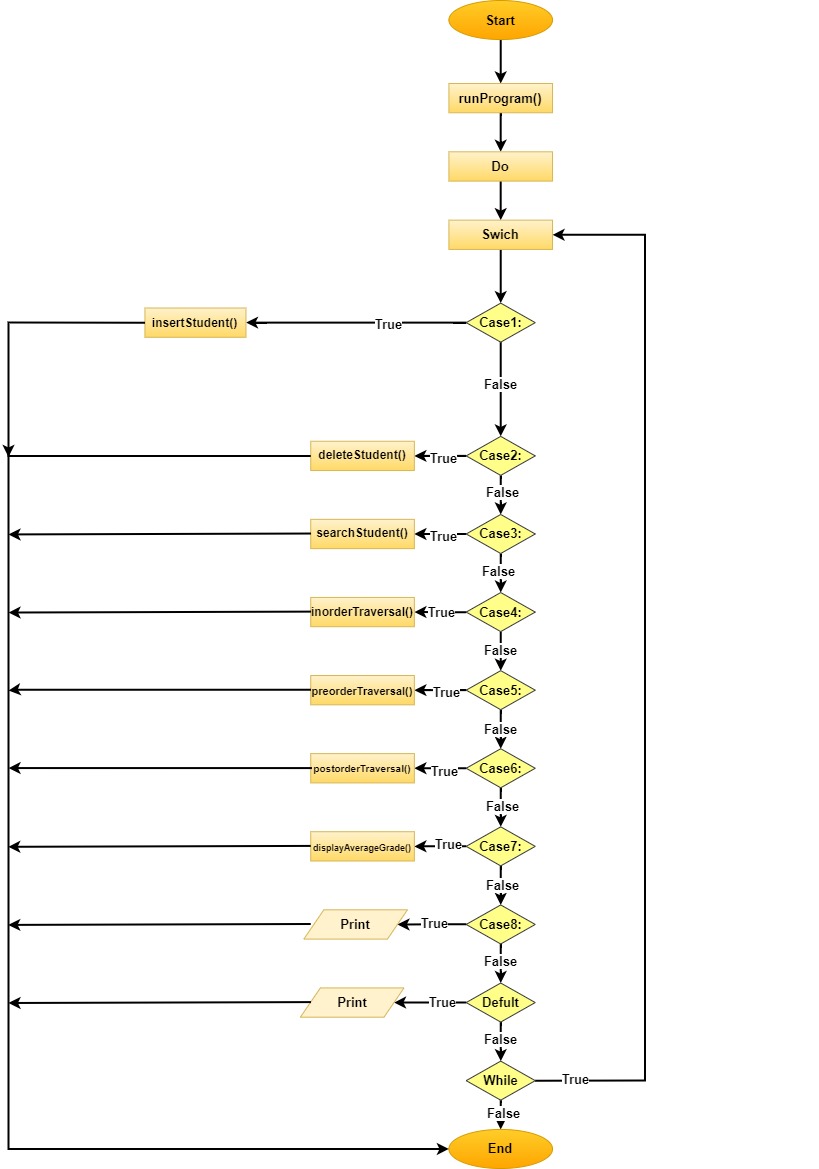
Create method traverseForAverage:

traverseForAverage for left subtree.

traverseForAverage for right subtree.

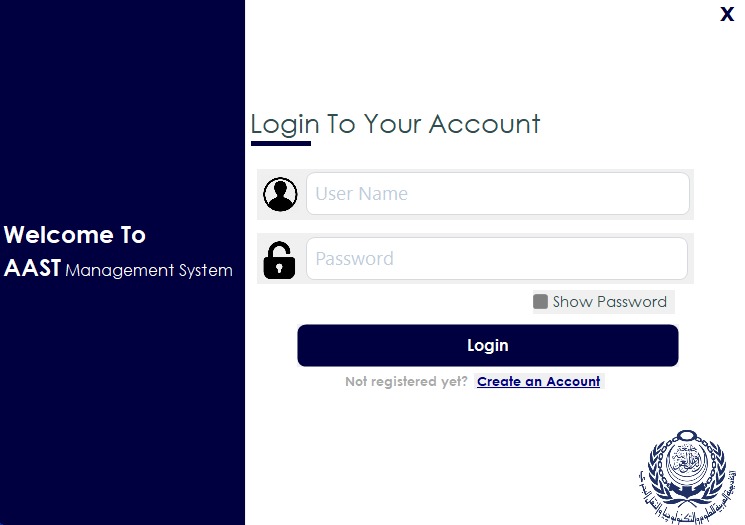
We have a main function that starts the program by calling runProgram()

# Continue until the user chooses to exit.**Flow Chart:**

****

***Figure 1****:flow chart*

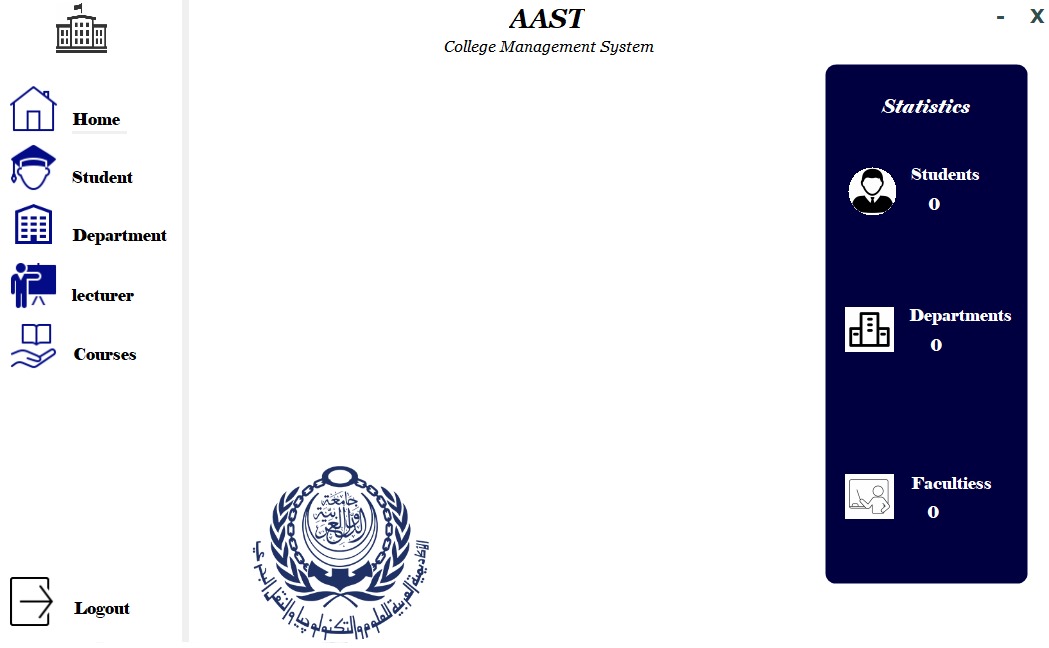
# GUI:





*Figure2***:***GUI login*

# 





*Figure3***:***GUI home*