Instructions

* Download and extract the .zip file with program source code and directions.
* Complete the *Test* functions in the *Submission.cpp* file.
* While working, Debug/Run the lab to see how many test runs pass or fail.
* Ideally, you want all test runs to pass to get the maximum amount of points.
* Your grade is shown at the bottom when you run the program.
* **You cannot use auto in place of known data types in PG1.**

**You cannot use multiple return statements in any method submitted for this project.** Additionally, the use of auto is not permitted. You must write the necessary programming statements/logic to accomplish each task. No credit will be given for using existing methods (thereby avoiding the need to write the programming statements) to accomplish the assigned tasks.

|  |
| --- |
| **Note:**This program will not compile/build until the Student class (Tests 1 and 2) is completed. |

Test 1 – Member fields and Constructors

Student Test1(string last, string first, int idNo)

For this test you will add member fields and constructors to the Student class.

The private member fields to add are a string to represent a last name, a string to represent a first name and an int to represent an identification number.

You will provide the default (no parameter) constructor and one overloaded constructor.

* The default constructor will set last and first names to empty strings ("") and set the id number to 1000000 (1 million).
* The overloaded constructor that will accept two strings (last name, first name) and one int (id number) and update the member fields the received values.

After making the specified additions in the Student class, create and then return a new Student object using the provided parameters.

Test 2 – Getters and Setters

Student Test2()

For this test, you must add public getters and setters for the Student class member fields. The methods must be named:

| **Getters** | **Setters** |
| --- | --- |
| GetFirstName | SetFirstName |
| GetLastName | SetLastName |
| GetIDNumber | SetIDNumber |

After making the specified additions in the Student class, create and then return a new Student object **using the default constructor**.

|  |
| --- |
| **Note:**Identifiers (method names and variables) are case-sensitive. For example, SetIdNumber is not the same as SetIDNumber. If a specific name is provided, you must use the exact name provided. |

|  |
| --- |
| **Note:**For Tests 3-5, you will use the enrollment array. It is a static member of the Submission class. |

Test 3 – Insert in collection

bool Test3(Student enrolled)

Given a string, enrolled, search the enrollment array to find an empty element (null). If an empty element is located, place enrolled in the empty element and return true. If there are no available slots, return false.

Test 4 – Remove from collection

bool Test4(int idNumber)

Given an int, idNumber, search the array to find idNumber. Examine each element in the enrollment array to find idNumber. If it is found, mark the array location as empty (null) and return true. If idNumber is not found in the array, return false.

Test 5 – Retrieve from collection

Student Test5(int idNumber)

Given an int, idNumber, search the array to find idNumber. Examine each element in the enrollment array to find idNumber. If it is found, return the first Student with the ID number idNumber. If idNumber is not found in the array, return null.

Rubric

1. Test 1 - Return the exact requested result - 20%
2. Test 2 - Return the exact requested result - 20%
3. Test 3 - Return the exact requested result - 20%
4. Test 4 - Return the exact requested result - 20%
5. Test 5 - Return the exact requested result - 20%