Objectives:

- Implement a simple class with public and private members and multiple constructors.
- Gain a better understanding of searching and sorting algorithms.
- Practice problem solving using Object Oriented Programming.

Overview:

In this assignment, you are expected to *write two classes called School and Student* that will allow creation and handling of School and Student objects, as described below.

Detailed Structure of Assignment Task:

This assignment has two components.

Part I [InLab] : Designing the Classes "Student" and "School": We first discuss the design of the structure:

- The `Student` class should have two private data members of the `int` type, *student_id* and *marks*. It should have the following `public` functions
  - One empty constructor (`Student()`)
  - Constructor of the form `Student(int sid, int m)` to initialize the private fields *student_id* and *marks*
  - `int get_id()` which returns the `student_id`
  - `int get_marks()` which returns `marks`
- The `School` class should have a private data members *students* (which is an array of `Student` objects) and n (the size of the array)
- The `School` class should have the following `public` functions:
  - `School(int num)`: For this constructor,you need to allocate space for *num* Student objects in the *students* array.
  - A destructor to de-allocate the memory allocated to the *students* array
  - `void read_students()`: n lines of input should be read from standard input, the *students* array should be initialized.
  - There should be a member function called `linear_search` that will take a single `int` type parameter `id` and return the marks of student object with *student_id* equal to `id`. This function should perform a simple $O(n)$ search on the students array.

Part II [Take Home] : Sorting and Binary Search:

- The `School` class should also have the following public functions:
  - `void sort()`, which will not take any input parameters and will not return anything. The function must sort the array in place using `insertion sort` and then print student ids in `decreasing order` of marks as space separated integers in a single line.
  - `void binary_search(int marks)`,which will take a single int type parameter *marks*. This function should make use of the sort function written earlier which will sort and print the student ids in a single line and then perform a $O(log(n))$ binary search on the students array and return the *student_id* of the student object with marks equal to *marks*. If no student object with the given marks is found, return -1.
- After defining the class, you should write a driver program that reads in the input as specified below and generates the corresponding output as specified. The driver performs the basic IO in C++.

Input-Output:

Part I
Input format
Each test case has the following format:
First line is an integer N, the size of the students array. N lines follow each of which has two integers *id* and *marks*. Last line is the *student_id* to search for.

Output format
Output a single integer which is the *marks* of the student with the given *student_id*. Output -1 if there is no student having the given id.

Part II
Input format
Each test case has the following format:
First line is an integer N, the size of the students array. N lines follow each of which has two integers *id* and *marks*. Last line is the *marks* to search for.

Output format
Output will be 2 two lines

- $student\_id_i$ $student\_id_j$ $student\_id_k$...
- *student_id*

First line is student ids in decreasing order of marks as space separated integers.
Second line is a single integer which is the *student_id* of the student with the given marks. Output -1 if there is no student having the given marks.