|  |
| --- |
| **Spring 2021**  **CS-240: Object-oriented Programming** |
| Lab-10 Manual |
| **Arrays of Objects, Sorting and Searching** |



# GIFT School of Engineering and Applied Sciences

|  |
| --- |
| v1.3  3/30/2020 |

# Task #1: Creating and Manipulating Arrays of Objects, Sorting and Searching

# In this task, you are being asked to practice creating arrays of objects. Also, you will be writing the selection sort and binary search as applied on arrays of objects.

**NOTE: Write your class and the *main* method in separate files.**

Write a class definition **Student** with three **private** data member **id (int)**, **firstName (String),** **lastName (String)** and **gpa (double)**. The class has the following methods as described below:

* Write the default constructor, you may assign default values in it.
* Write the four-argument overloaded constructor.
* Write the copy constructor, which assign the argument object’s state values to the caller object.
* This class also contains the following methods:
  + **Setter** and **Getter** methods  
    Create setter and getters methods for each instance variable.
  + **void print()**This method prints all the values of data members of a single object with appropriate messages.
  + **static void print(Student[] s)**This static method prints all the values of data members of all objects of the argument array with appropriate messages. The method will call the **print** methodto print the values of every single object.
  + **boolean isEqual(Student s)**This method returns *true* if the **id** value of the caller object is equal to the **id** value of the argument object,and *false* otherwise.
  + **boolean isGPALesser(Student s)**This method returns *true* if the **gpa** value of the caller object is less than the **gpa** value of the argument object, and *false* otherwise.
  + **static void sort(Student[] s)**This static method applies selection sort on the input array and sorts on the basis of **gpa** of each student in ascending order. The method should make use of **isGPALesser** methodto compare the **gpa** of any two students.
  + **static int linearSearch(Student[] s, int key)**This static method uses linear search and searches the array of students for the **id** (key), and returns its *index* if found, otherwise, returns -1.
  + **static void search(Student[] s, int key)**This static method uses linear search and searches the array of students for the **id** (key), and prints all student data by calling **print** method. If no **id** matches, it displays **"No Student Found".**
  + **static int search(Student[] s, double key)**This static method uses binary search and searches the array of students for the **gpa** (key), and returns its *index* if found, otherwise, returns -1.

1. Create a program called Student.java for the class, and RunStudent.java having the main method.

# Create an array for five students, all having different state values.

# Create a sixth object of Student with copy constructor (use the third Student object as argument).

# Next, assign the six student objects to each index of the array. For example,

**final int SIZE = 6;**

**Student[] s = new Student[SIZE];**

**s[0] = new Student();**

**s[1] = new Student(101, “Ali”, “Akbar”, 2.95);**

etc.

# Display the data of all students using the print(Student[] s) method.

# Apply isEqual method to the sixth object by passing Student # 3 as argument and show appropriate message according to the result of the method.

1. Without sorting the array, apply **linearSearch(Student[], int)** method, and display the message if the key is found or not.
2. Apply **sort** method, and then displays all student’s data.
3. Apply **search(Student[], int)** method, and display the message if the key is found or not.
4. Apply **search(Student[], double)** method and display the message if the key is found or not.

# Task #2: Creating and Manipulating Arrays of Objects, Sorting and Searching

# In this task, you are being asked to practice creating arrays of objects. Also, you will be writing the selection sort and binary search as applied on arrays of objects.

# NOTE: Write your class and the *main* method in separate files.

Write a class named **Course** that has the following **private** fields:

* **code.** The **code** field references a **String** object that holds the course’s code.
* **name.** The **name** field references a **String** object that holds the course’s name.
* **credits.** The **credit** is an **int** variable that holds the course’s credit hours.

Now, write the following code as described:

* Write the default constructor, a three-argument overloaded constructor and a copy constructor.
* **Setter** and **Getter** methods  
  Create setter and getters methods for each instance variable.
* **void print()**Displays all the values of data members of an object with appropriate messages.
* **static void print(Course[] c)**This static method prints all the values of data members of all objects of argument array with appropriate messages. The method will call the **print** methodto print the values of every single object.
* **boolean isEqual(Course c)**This method returns *true* if the *caller* object and the argument object have exactly the *same state values***,** and *false* otherwise.
* **static void sort(Course[] s)**This static method applies selection sort on the input array and sorts on the basis of **name** of each course in ascending order. The method should make use of **compareTo** methodof **String** to compare the **name** of any two courses.

**The String compareTo Method**

The String **compareTo** method takes another String value as an argument and compares both Strings (caller and argument) based on their Unicode values. The method does the following:

* + - Returns **0** if the values of both caller and argument String objects is *same*.
    - Returns **a positive value** if the value of caller String object is *greater* than the value of argument String object.
    - Returns **a negative value** if the value of caller String object is *less* than the value of argument String object.

**Examples:**

**String s1="java";**

**String s2="java";**

**String s3="nava";**

**String s4="jasa";**

**s1.compareTo(s2); //Returns 0, because both are //equal**

**s1.compareTo(s3); //Returns -4, because "j" is 4 //times lower than "n"**

**s1.compareTo(s4); //Returns 3 because "v" is 3 times //greater than "s"**

* **static int linearSearch(Course [] c, String key)**This static method uses linear search and searches the array of objects for the **code** (key), and returns its *index* if found, otherwise, returns -1.
* **static int search(Course[] s, String key)**This static method uses binary search and searches the array of courses for the **name** (key), and returns its *index* if found, otherwise, returns -1.

# Create a program called Course.java for the class, and RunCourse.java having the main method.

# Create an array of five objects having different states using all constructors, including the copy constructor, and assign the five Course objects to each index of the array.

# Display the data of all courses in the array of objects.

# Apply isEqual method to the fifth object by passing Course object # 2 as argument and show appropriate message according to the result of the method.

1. Without sorting the array, apply **linearSearch(Course[], String)** method, and display the message if the key is found or not.
2. Apply **sort** method, and then displays all course’s data.
3. Apply **search(Course[], String)** method, and display the message if the key is found or not.

# Task #3: Creating and Manipulating Arrays of Objects, Sorting and Searching

# In this task, you are being asked to practice creating arrays of objects. Also, you will be writing the selection sort and binary search as applied on arrays of objects.

# NOTE: Write your class and the *main* method in separate files.

Write a class named **Box3D** that has the following **private** fields:

* **length.** The **length** field is a **double** that holds the Box’s length.
* **width.** The **width** field is a **double** that holds the Box’s width.
* **height.** The **height** field is a **double** that holds the Box’s height.
* **color.** The **color** field references a **String** object that holds the color of the Box.

Now, write the following code as described:

* Write the default constructor, a four-argument overloaded constructor, and a copy constructor.
* **Setter** and **Getter** methods  
  Create setter and getters methods for each instance variable.
* **void print()**Displays all the values of data members of an object with appropriate messages.
* **static void print(Box3D[] b)**This static method prints all the values of data members of all objects of argument array with appropriate messages. The method will call the **print** to print the values of every single object.
* **boolean isEqual(Box3D b)**This method returns *true* if the *caller* object and the argument object **b** have exactly the *same state values***,** and *false* otherwise.
* **double area()**This method returns the area of the Box. (The area can be calculated by **length x width x height**).
* **static void sort(Box3D[] b)**This static method applies selection sort on the input array and sorts on the basis of *area* of each Box in ascending order. The method should make use of **area** methodto compare the *area* of any two Box3D objects.
* **static int linearSearch(Box3D [] b, String key)**This static method uses linear search and searches the array of objects for the **color** (key), and returns its *index* if found, otherwise, returns -1.
* **static void search(Box3D[] b, double key)**This static method uses linear search and searches the array of **Box3D**, and prints all **Box3D** objects which have area *greater than or equal* to the *key* argument. If no object matches the criteria, then prints the message **"No Object Found".** The method should use the **print** method to print the object.

# Create a program called Box3D.java for the class, and RunBox3D.java having the main method.

# Create an array for five Box3D objects having different states and assign the five Box3D objects to each index of the array.

# Display the data of all Box3D objects.

# Create a sixth object of Box3D with copy constructor (use Box3D object #4 as argument).

# Apply isEqual method to the sixth object by passing Box3D object #4 as argument and show appropriate message according to the result of the method.

1. Without sorting the array, apply **linearSearch(Box3D[], String)** method, and display the message if the key is found or not.
2. Apply **sort** method, and then display the data of all **Box3D** objects.
3. Apply **search(Box3D[], double)** method and display appropriate messages.