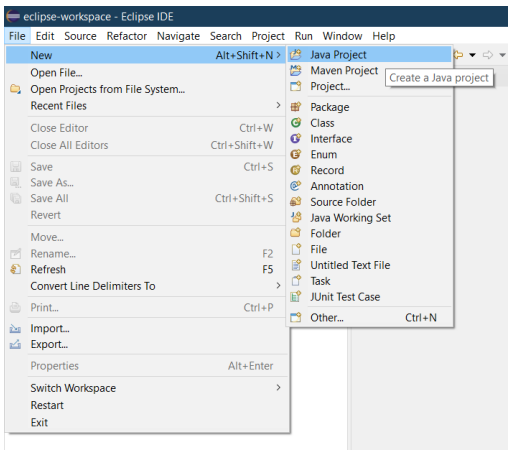
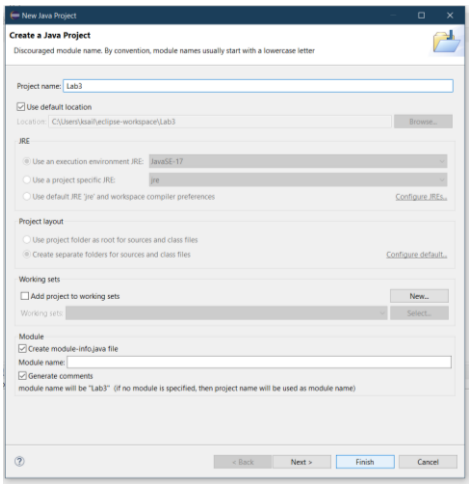
Department of Electrical and Software Engineering
Schulich School of Engineering

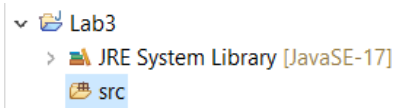
ENSF 694 - Principles of Software Development II
Summer 2023

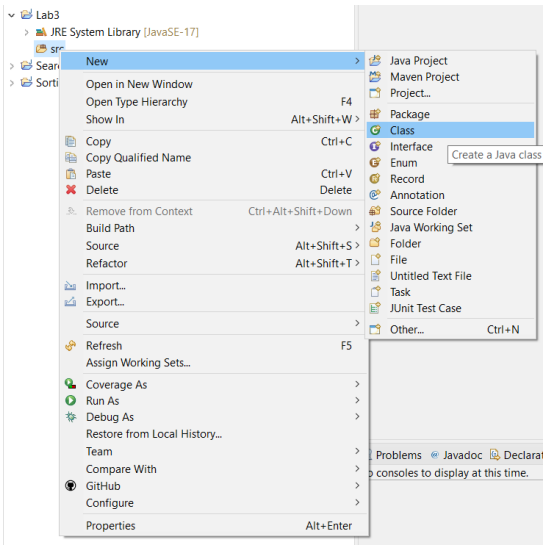
Lab 3 – July 05, 2023
Topic: Sorting Algorithm

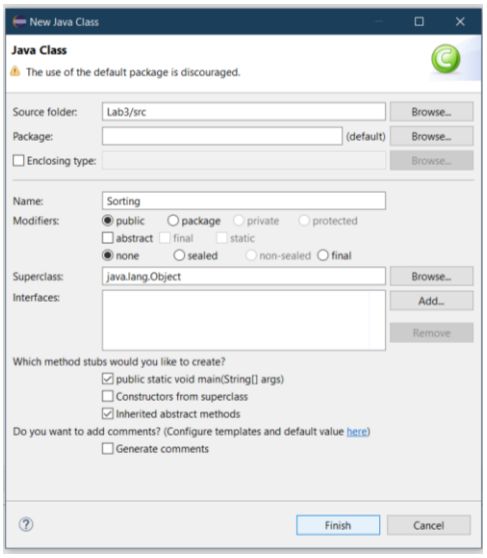
- All codes must be complete and compile without any errors.
- The codes should work for not only the given sample inputs but also any inputs of the same data types.
- **Submission: github link of the codes in the d2l dropbox 'Lab#03_Jul05' and push your code in the github classroom repository**
    - Go to this link - https://classroom.github.com/a/6WuMdXIQ
    - Refresh and accept the Lab3 link
    - Clone the repository and then push your code
    - Then submit the github link to the d2l dropbox 'Lab#03_Jul05'
- Steps to create the code folder (https://www.javatpoint.com/how-to-run-java-program-in-eclipse) -
    - In Eclipse, create a new Java Project named "Lab3"
    - In its' 'src' folder, create a new class named 'Sorting'
    - You may use the same format format mentioned in the question or your own
    - Submit the Lab3 project folder in github
    - You can add any additional condition checking or comments if you want to, but only after implementing what is mentioned in the question











Sample structure of the Sorting.java file (you may use this format or your own)-

*Sorting.java

```
1  import java.util.Arrays;
2  import java.util.Scanner;
3
4  public class Sorting {
5
6      void selectionSort(int [] a)
7      {
8          // Selection sort code here
9          // for descending order sorting
10     }
11
12     int partition(int[] a, int first, int last)
13     {
14         // Quicksort partition code here
15     }
16
17     void quickSort(int[] a, int first, int last)
18     {
19
20         // Quicksort code here
21         // for ascending order
22     }
23
24     public static void main(String[] args) {
25
26         // Input
27
28         // Call Sorting Methods
29
30         // Output
31
32     }
33 }
```

Lab Tasks

15 marks -
05 for Correct Input-Output Format +
05 for Selection Sort +
05 for Quick Sort or Merge Sort

- Implement the **selection sort** algorithm to sort the integers in **'descending order'**
- Then apply the **quick sort or merge sort** algorithm to sort the previously sorted array (i.e., the descending ordered sorted array you got from selection sort) in **'ascending order'**
- Prompt the user for all inputs. Take Inputs from users using the Scanner method.
- Show the input array first, then show both sorted arrays.

Sample Run of the Code:

```
Enter the size of the array:
10

Enter the elements of the array:
23
45
12
5
3
67
90
33
25
1

The elements of the Array are:
23 45 12 5 3 67 90 33 25 1

Sorted Array (Selection Sort - Descending Order): [90, 67, 45, 33, 25, 23, 12, 5, 3, 1]


Sorted Array (Quick Sort - Ascending Order): [1, 3, 5, 12, 23, 25, 33, 45, 67, 90]
```