- **Course:** CSC2200.03
- **Student:** *Japheth Wun*, SFSU ID: 921555593
- **Teammate:** <*First Name*> <*Last Name*>, SFSU ID: <SFSU ID>
- **Assignment Number:** ##
- **Assignment Due Date & Time:** Month-Day-Year at Hour:Minute AM/PM

Part A

Array = {9 2 8 4 7 5 6 2 4 3 1 7}
- **Question #1:** Show the contents of the array each time a selection sort
- **Answer/Summary/Discussion:** {7 2 8 4 7 5 6 2 4 3 1 9}
{7 2 1 4 7 5 6 2 4 3 8 9}
{3 2 1 4 7 5 6 2 4 7 8 9}
{3 2 1 4 4 5 6 2 7 7 8 9}
{3 2 1 4 4 5 6 2 7 7 8 9}
{3 2 1 4 4 5 2 6 7 7 8 9}
{3 2 1 4 4 2 5 6 7 7 8 9}
{3 2 1 2 4 4 5 6 7 7 8 9}
{2 2 1 3 4 4 5 6 7 7 8 9}
{1 2 2 3 4 4 5 6 7 7 8 9}

- **Question #2:** Show the contents of the array each time an insertion sort
- **Answer/Summary/Discussion:**

{2 9 8 4 7 5 6 2 4 3 1 7}	{2 8 9 4 7 5 6 2 4 3 1 7}	{2 8 4 9 7 5 6 2 4 3 1 7}
{2 4 8 9 7 5 6 2 4 3 1 7}	{2 4 8 7 9 5 6 2 4 3 1 7}	{2 4 7 8 9 5 6 2 4 3 1 7}
{2 4 7 8 5 9 6 2 4 3 1 7}	{2 4 7 5 8 9 6 2 4 3 1 7}	{2 4 5 7 8 9 6 2 4 3 1 7}
{2 4 5 7 8 6 9 2 4 3 1 7}	{2 4 5 7 6 8 9 2 4 3 1 7}	{2 4 5 6 7 8 9 2 4 3 1 7}
{2 4 5 6 7 8 2 9 4 3 1 7}	{2 4 5 6 7 2 8 9 4 3 1 7}	{2 4 5 6 2 7 8 9 4 3 1 7}
{2 4 5 2 6 7 8 9 4 3 1 7}	{2 4 2 5 6 7 8 9 4 3 1 7}	{2 2 4 5 6 7 8 9 4 3 1 7}
{2 2 4 5 6 7 8 4 9 3 1 7}	{2 2 4 5 6 7 4 8 9 3 1 7}	{2 2 4 5 6 4 7 8 9 3 1 7}
{2 2 4 5 4 6 7 8 9 3 1 7}	{2 2 4 4 5 6 7 8 9 3 1 7}	{2 2 4 4 5 6 7 8 3 9 1 7}
{2 2 4 4 5 6 7 3 8 9 1 7}	{2 2 4 4 5 6 3 7 8 9 1 7}	{2 2 4 4 5 3 6 7 8 9 1 7}
{2 2 4 4 3 5 6 7 8 9 1 7}	{2 2 4 3 4 5 6 7 8 9 1 7}	{2 2 3 4 4 5 6 7 8 9 1 7}
{2 2 3 4 4 5 6 7 8 1 9 7}	{2 2 3 4 4 5 6 7 1 8 9 7}	{2 2 3 4 4 5 6 1 7 8 9 7}
{2 2 3 4 4 5 1 6 7 8 9 7}	{2 2 3 4 4 1 5 6 7 8 9 7}	{2 2 3 4 1 4 5 6 7 8 9 7}
{2 2 3 1 4 4 5 6 7 8 9 7}	{2 2 1 3 4 4 5 6 7 8 9 7}	{2 1 2 3 4 4 5 6 7 8 9 7}
{1 2 2 3 4 4 5 6 7 8 9 7}	{1 2 2 3 4 4 5 6 7 8 7 9}	{1 2 2 3 4 4 5 6 7 7 8 9}

- **Question #3:** Show the contents of the array each time a Shell sort
- **Answer/Summary/Discussion:**

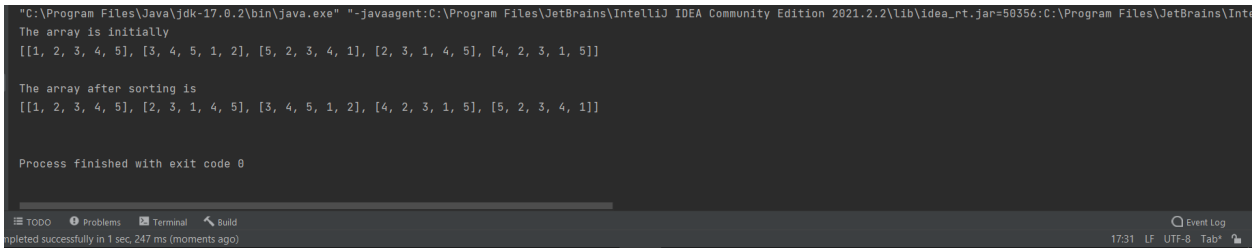
{7 2 8 4 9 5 6 2 4 3 1 7}	{7 2 8 4 9 5 6 2 4 3 1 7}	{7 2 8 4 4 5 6 2 9 3 1 7}
{4 2 8 4 7 5 6 2 9 3 1 7}	{4 2 8 4 7 3 6 2 9 5 1 7}	{4 2 6 4 7 3 8 2 9 5 1 7}
{4 2 6 4 7 3 1 2 9 5 8 7}	{4 2 1 4 7 3 6 2 9 5 8 7}	{4 2 1 2 7 3 6 4 9 5 8 7}

{1 2 4 2 7 3 6 4 9 5 8 7} {1 2 4 2 6 3 7 4 9 5 8 7} {1 2 4 2 6 3 7 4 8 5 9 7}
{1 2 4 2 6 3 7 4 8 5 9 7} {1 2 2 4 6 3 7 4 8 5 9 7} {1 2 2 4 3 6 7 4 8 5 9 7}
{1 2 2 3 4 6 7 4 8 5 9 7} {1 2 2 3 4 6 4 7 8 5 9 7} {1 2 2 3 4 4 6 7 8 5 9 7}
{1 2 2 3 4 4 6 7 5 8 9 7} {1 2 2 3 4 4 6 5 7 8 9 7} {1 2 2 3 4 4 5 6 7 8 9 7}
{1 2 2 3 4 4 5 6 7 8 7 9} {1 2 2 3 4 4 5 6 7 7 8 9}

Part B

- **Question #1:** - *Suppose we want to find the largest entry in an unsorted array of n entries. Algorithm A searches the entire array sequentially and records the largest entry seen so far. Algorithm B sorts the array into descending order and then reports the first entry as the largest. Compare the time efficiency of the two approaches. Coding is not required but highly recommended.*
- **Answer/Summary/Discussion:** Officially Algo A is better because its efficiency is immediate such that is it, O(n). For Algo B, you have to sort it, which gives it an $O(n^2)$.

- **Question #2:** - *Consider an n by n array of integer values. Write an algorithm to sort the rows of the array by their first value*
- **Answer/Summary/Discussion:**



Part C

After each of the following statements executes, what are the contents of the queue?
- **Question #1:** *Please explain*
- **Answer/Summary/Discussion:**

```
QueueInterface myQueue = new LinkedQueue<>();          //Queue ={}
myQueue.enqueue("Jane");
myQueue.enqueue("Jess");
myQueue.enqueue("Jon");
//Queue ={Jon, Jess, Jane}
myQueue.enqueue(myQueue.dequeue());
//Queue ={Jane, Jon, Jess}
myQueue.enqueue(myQueue.getFront());
//Queue ={Jess, Jane, Jon, Jess}
myQueue.enqueue("Jim");
//Queue ={Jim, Jane, Jon, Jess}
String name = myQueue.dequeue();
myQueue.enqueue(myQueue.getFront());
//Queue ={Jess, Jim, Jane, Jon, Jess}
```

- **Question #2:** *Please explain*
- **Answer/Summary/Discussion:**

```
DequeInterface myDeque = new LinkedDeque<>();          //Queue ={}
myDeque.addToFront("Jim");                             //Queue ={Jim}
myDeque.addToFront("Jess");                            //Queue ={Jim, Jess}
myDeque.addToBack("Jen");                              //Queue ={Jen, Jim, Jess}
myDeque.addToBack("Josh");                             //Queue ={Josh, Jen, Jim, Jess}
String name = myDeque.removeFront();
myDeque.addToBack(name);                  //Queue ={Jen, Jim, Jess, Jen, Jim, Jess}
myDeque.addToBack(myDeque.getFront());
                                          //Queue ={Jess, Jen, Jim, Jess, Jen, Jim, Jess}
myDeque.addToFront(myDeque.removeBack());
                                          //Queue ={Jim, Jess, Jen, Jim, Jess, Jen}
myDeque.addToFront(myDeque.getBack());
                                          //Queue ={Jess, Jen, Jim, Jess, Jen, Jim}
```

- **Question #3:** *Please explain*
- **Answer/Summary/Discussion:**

```
PriorityQueueInterface myPriorityQueue = new LinkedPriorityQueue<>();  //Queue = {}
myPriorityQueue.add("Jim");
myPriorityQueue.add("Josh");
myPriorityQueue.add("Jon");
myPriorityQueue.add("Jane");
//Queue = {Jane, Jon, Josh, Jim}
String name = myPriorityQueue.remove();
//Queue = {Jane,Jon, Josh}
myPriorityQueue.add(name);
//Queue = {Jane,Jon, Josh, Jane, Jon, Josh, Jim}
myPriorityQueue.add(myPriorityQueue.peek());
//Queue = {Jim, Jane,Jon, Josh, Jane, Jon, Josh, Jim}
myPriorityQueue.add("Jose");
//Queue = {Jose, Jim, Jane,Jon, Josh, Jane, Jon, Josh, Jim}
myPriorityQueue.remove();
//Queue = {Jose, Jim, Jane,Jon, Josh, Jane, Jon, Josh}
```

Part D

- **Question #:** *Make Queue*
- **Answer/Summary/Discussion:**

Part E

- **Question #:** Sort queue by first-name, last-name, student-id, gpa, number-of-small-questions, number-of-big-questions
- **Answer/Summary/Discussion:** *I tried using compareTo the names but it did not work, so I did it the normal way by comparing each letter. For the id, gpa, number of small and large questions and the combination of small and large questions, I compared their double/int values with inequality signs.*