**SENG2200 Week 2 Workshop Solutions**

**Part 1**

1. Draw UML

Student

Number

Name etc.

List<Courses>

Various getters and setters

Course

Course Number, Credit points Course name

Etc.

Various getters and setters

1 \*

1. Write an implementation

Public class Student

{

Private int number;

Private int name;

…

Public Student()

{

…

}

gettersAndSetters()

{

…

}

}

1. Complete the constructor

Set the data inside the constructor?

1. Write java code that will instantiate a Student object

Student myStudent = new Student(params);

1. Write java code for an array of students

Student[] studArray = new Student[2];

1. Populate the array

For(int I = 0; I < studArray.**length**; i++)

{

studArray[i] = new Student( params );

}

1. Compare this to C++

Sorry if this part isn’t perfect, its been a while since ive had to write this stuff down. Especially since I’ve been doing a lot of C# its all blurred into one a bit.

**I have to re-learn this every year too.**

**In Java**, q5 gives

|  |
| --- |
| Header Details  (size 2) |
| Null |
| null |

In Java, q6 gives

|  |
| --- |
| Header Details  (size 2) |
|  |
|  |

Default Student

Default Student

**In C++** it would vary. If we declared:

Student myArray[2]; // Two students init with default constrctor

We would get // or a compile error if no def constr.

|  |
| --- |
| Default Student |
| Default Student |

After putting myArray[i] = Student(); // Init with def constr (or error id none)

|  |
| --- |
| Default Student |
| Default  Student |

If instead we declared using **pointers**

Student \*myArray[2];

We would get

|  |
| --- |
| Garbage |
| Garbage |

After:

myArray[i] = new Student();

Default Student

|  |
| --- |
|  |
|  |

Default Student

Here the LHS (the array) is Stack, and RHS is heap.

1. How do we make it a Stack?

Throw an interface around it.

Stack

private ArrayStack

int top\_of\_stack;

pop()

push()

isEmpty()

**Part 2 We don’t cover generics until topic 6 – we’ll do this all with “Objects” until then**

1. Write Java Node – Objects and casting

public Class Node

{

private Object data;

private Node nextNode

Constructor()

gettersAndSetters()

}

1. Write a Queue

public class Queue

{

private int size;

private Node head;

private Node tail;

public Queue(){ tail = head = null };

public add(Object object)

{

if(head == null)

{

head = tail = new Node(object);

}

else

{

tail.next = new Node (object);

tail = object’s node;

}

size++;

}

Similar for pop()

}

1. Expand node for Deque – this involves making the List doubly linked.

Add ‘private Node previous’

1. Same as 10, but considering updating tail, and adding in the new methods.
2. Likely class discussion around here about a few ideas. Actual answer is two sentinel nodes. Explain, probably by drawing on the board, what they are. Advantage being that you can ignore all the ‘if head == null’ and if(head == tail) checks.
3. Explain how a single node works if they need help, then let them write it themselves.