Pawan Chandra

Pawan Khatri

**Project 1 Write Up**

1. What helped us a lot was going to the University of Washington website [here](https://courses.cs.washington.edu/courses/cse332/13wi/projects/project1/). It was almost the same directions as the one provided but with slightly more details. It also had links on how to use software such as Sox and how to edit configuration in Eclipse.
2. I tested the stack by using Eclipses debugging feature and closely keeping track of the nodes. Most of the functions like push, isEmpty, etc. weren’t new and I had previous projects from my CSC20 that I used as a guide.
3. No I did not use any other package besides the java.util.EmptyStackException
4. For the array stacks all I did was set the size to a nice round even number like 10. After all the elements were filled I simply multiplied the size of the array(10) by 2, doubling it as they get filled. Then the previous information will be stored into a temp array using a forloop. If a file contains one million elements then the array size would be 10 \* where k = the number of times the array is filled up. Million would be 18 billion is 28.
5. To simulate a stack using queue I would create 2 additional queues. Once all the data is transferred to the main queue, I would pass those values to the second queue, all but the last value. The last value would be stored to the third queue. I would repeat the process, this time transferring second queue to main queue and taking the last value to third queue. This would eventually have the elements in order of stack when using remove.
6. void push (element d)

{

q.enqueue(d);

size++;

}

void pop ()

{

if (q.isEmpty())

return;

while(q.size()!= 1) // gets all the way to the second last element

{

q2.push(q.peek());

q.pop;

}

size--;

return q.pop(); //get the last element

}

1. Using queue for this project would be difficult for multiple reasons. Main reason being is that we want to reverse the original data and because queues are first in first out, it would take multiple steps to do what stacks do in a single step. Another reason is that it is going to take more memory and time to implement a queue compared to a stack. With Arrays it is easy to index from the last element to the first and vice versa. Arrays are also finite in size so if the size is known, arrays are more effective.