# Heaps

## Goals of the Assignment

The goal of this assignment is to use the data structures we've discussed this week to solve a problem. As always, you are expected to demonstrate good software engineering practices including use of version control and testing. ***Read this document in its entirety*** before asking the course staff for help!

## Problem Statement

Shinji's is the most exclusive sushi shop in Rochester. It only serves premium quality unique sushi creations every day. To order, customers view the available items behind a glass display case. Each item is labeled with only a number. The customer picks between 3 and 8 items and hands their list to the cashier. The cashier then removes the items from the case and carefully places them into the to-go bag, heaviest item to lightest item. The customer then pays, takes the order home, and enjoys it.

## Activities

For this activity, you will write a simulation for the process in the above problem statement using the data structures discussed during Unit 06. You may use Java's implementation for `Set`, `Map`, `PriorityQueue` (Java's heap), and `Stack`. You may **not** use a `List`.

1. Start by creating a `Sushi` class. Each piece of sushi is identified by the following pieces of information:
    a. `number` - the number that is associated with the piece of sushi and used to order it.
    b. `weight` - how much the sushi weighs (random value from 20 - 100 grams).
    c. `price` - the cost of the piece of sushi (random value between $10 and $40).
    d. In addition to the state and any accessors, Sushi should have a toString which includes all the state information and be able to be sorted by weight.

2. Each `Order` consists of a `cost` and a `bag`, which contains the `Sushi`. When creating this class, consider what data structure would best represent the `bag`.

3. The main class will be the `Shop`. Only one customer will be simulated per run. The main flow for the shop should be:
    a. Create the display with 25 items in it.
    b. The customer writes down the number for 3 - 8 unique items that are in the display (you must use a `Set` to store the numbers).
    c. The cashier takes the wanted items list, removes the actual items from the display, and adds them to an order.

d. The customer pays for the order and takes the food home. Once home, they take each piece out and eat it.

## Sample Output:

```
Welcome to Shinji's Sushi Shop!
The cashier takes your order: 0 18 4 11
The cashier prepares your order.
That will be $99.
Here is your order, please come again.
Back at your apartment, you take out your sushi and eat it ...
Eating Sushi{number=11, weight=53 grams, $32} ... yum!
Eating Sushi{number=4, weight=60 grams, $31} ... yum!
Eating Sushi{number=0, weight=81 grams, $23} ... yum!
Eating Sushi{number=18, weight=99 grams, $13} ... yum!
```

## Submission Instructions

You must ensure that your solution to this assignment is pushed to GitHub *before* the start of the next lecture period.