CMSC133, Spring 2020, Quiz #4 (Memory Map)
Deadline: Wednesday, April 29, 2:00 pm (No late deadline)

Specifications

Draw a memory map for the code you see on the next page, until the execution reaches the point indicated by the comment /* HERE */. In your diagram:

- You mut have a stack, heap, and static memory sections as illustrated by the examples at:

  http://www.cs.umd.edu/~nelson/classes/resources/MemoryMapsInformation/MemoryMapsInformation.pdf

- Identify each frame as illustrated by the previous examples.
- Draw your variables as they are encountered during program execution.

```
public class ShoppingCart {
   private String customer;
   private int itemsCount;
   private StringBuffer items;
   private static final double MAX_ITEMS_NUMBER = 40;

   public ShoppingCart(String customer) {
      this.customer = customer;
      items = new StringBuffer();
   }

   public ShoppingCart addItem(String desc) {
      if (itemsCount < MAX_ITEMS_NUMBER) {
         items.append(desc);
         itemsCount++;
      }

      return this;
   }

   public ShoppingCart specialItemAdd(String desc) {
      String tai = "Spec:" + desc;

      /* HERE */
      addItem(tai);

      return this;
   }

   public String toString() {
      return "ShoppingCart [customer=" + customer + ", itemsCount=" + itemsCount + ", items=" + items + "]";
   }
}

public class Driver {
   public static void proc(ShoppingCart[] allc, int which, String item) {
      allc[which].specialItemAdd(item);

      System.out.println(allc[1]);
      System.out.println(allc[3]);
   }

   public static void main(String[] args) {
      ShoppingCart laura = new ShoppingCart("Laura");
      laura.addItem("milk");
      ShoppingCart tom = new ShoppingCart("Tom");
      tom.addItem("salt");

      ShoppingCart[] carts = new ShoppingCart[4];
      carts[1] = laura;
      carts[3] = tom;

      proc(carts, 1, "oil");
   }
}
```