Criterion C: Development

Initializing the product:

```
//ToolShop will initialize the overarching arrays to create an array that houses 4 arraylists.
public ToolShop(){
    generalPurpose = new ArrayList<>();
    bikeTools = new ArrayList<>();
    powerTools = new ArrayList<>();
    sprays = new ArrayList<>();
    tools[0] = generalPurpose;
    tools[1] = bikeTools;
    tools[2] = powerTools;
    tools[3] =  sprays;
}
```

Printing Inventory:

```
/*This will run through the main array and every arraylist in it so that all
the tools in the tool shop will be printed. This will alow the client to see
what tools are in stock and which ones he will need to purchase.
*/
public void printInventory(){
    for(ArrayList x: tools){
        for(Object t: x){
            Tool tool = ((Tool)(t));
            System.out.println(tool.getName() + ": " + tool.getQuantity());
        }
        System.out.println();
    }
}
```

This code was hard to create because it would need to loop through the main array, then through every arraylist calling each name and the quantity of each tool in the inventory.

Interface Class:

```
public interface Tool {
    public String getName();
    public int getQuantity();
    public String getType();
    public void changer(int n);
}
```

This interface class was used to link all the tools together so that it would be easier to access the different type of tools. This prevents repetive code in the program by making it refer to a certain class based on the object type rather than trying to run it on every project.

Remover:

```
/*This method allows the client to remove tools/products that have been
used. It will adjust the quantity or completly remove the item if the item
runs out. It also needs to warn the client when he is running low on a certain
product or tool.
*/
public void remover(Tool tool, int n){
    int index = legend(tool);
    String toolName = tool.getName();
    //Type cast it
    for(int i = 0; i<tools[index].size(); i++){
        if(((Tool)(tools[index].get(i))).getName().equals(toolName)){
            ((Tool)(tools[index].get(i))).changer(n);
            System.out.println("You have " + ((Tool)(tools[index].get(i))).getQuantity() + " " + tool.getName() + " left.");
            if(((Tool)(tools[index].get(i))).getQuantity() <= 2){
                System.out.println("You are running out. You should purchase more " + tool.getName());
            }
        }
        break;
    }
}
```

This was the hardest code to write since it needs to identify what tool is imputed, then go to the right section, and finally find the tool. Once it found the tool, the program would then take off *n*, the quantity that the client wants to remove. After remvoing the quantity, it would need to check to see how much of the item is left and will need to notify the client if the item is running low.

What array slot?

```
/*Without this method, no tool would be able to be sorted into the right
section since the tool does not containt a type. This method allows the
program to operate and correctly sort the tools, add, or even remove the
tools.
*/
public int legend(Tool tool){
    if(tool.getType().equals("General Purpose")){
        return 0;
    }else if(tool.getType().equals("Bike Tools")){
        return 1;
    }else if(tool.getType().equals("Power Tools")){
        return 2;
    }else{
        return 3;
    }
}
```

Adder:

```
/*It is not as complex as the remover tool below but it does require a bit
of work. It first has to determine where the tool is added using the legend
method. It then has to check if the tool is already in the inventory and if
it is, it adds to its quantity. If it is not, it will just add it to the end
of the arraylist.
*/
public void adder(Tool tool){
    int index = legend(tool);
    String toolName = tool.getName();
    int n = tool.getQuantity();
    for(int i = 0; i<tools[index].size(); i++){
        if(((Tool)(tools[index].get(i))).getName().equals(toolName)){
            ((Tool)(tools[index].get(i))).changer(n);
            System.out.println("You have " + ((Tool)(tools[index].get(i))).getQuantity() + " " + tool.getName() + " left.");
            return;
        }
    }
    tools[index].add(tool);
    System.out.println("You have successfully added " + n + " " + toolName + ".");
}
```