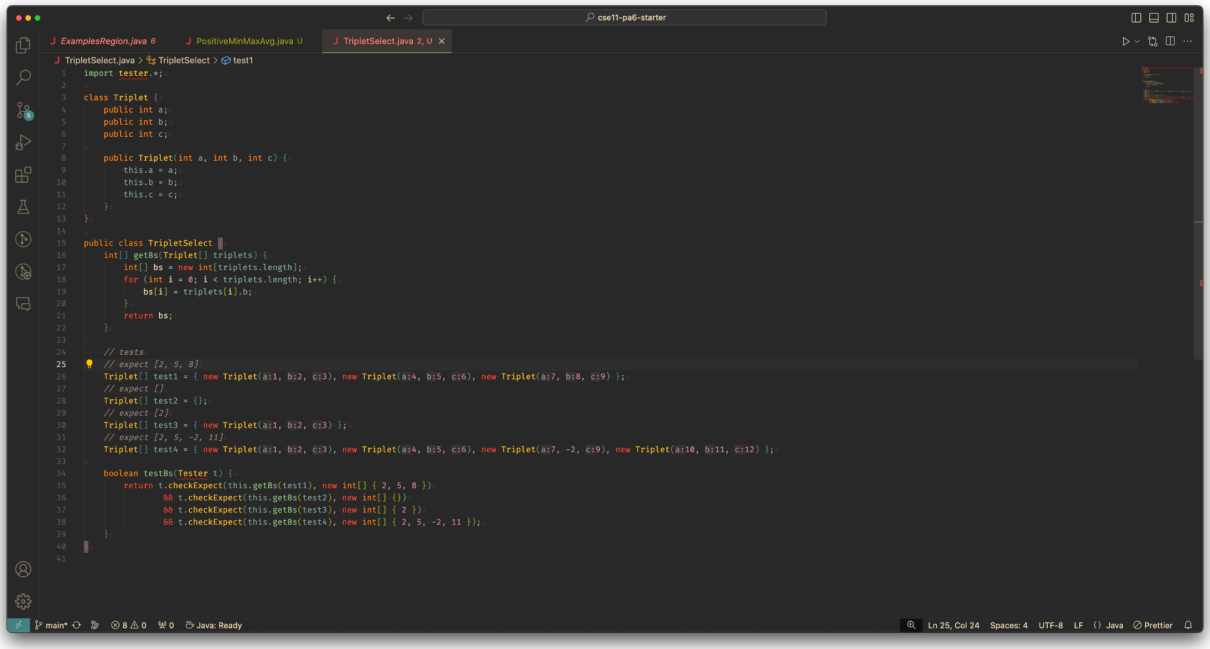
PA6 - Programming Workflow Template

Name: Jacob Hansen
UCSD email: j4hansen@ucsd.edu

First 30 Minutes

Screenshot or copy/paste of program:



```java
import tester.*;

class Triplet {
    public int a;
    public int b;
    public int c;

    public Triplet(int a, int b, int c) {
        this.a = a;
        this.b = b;
        this.c = c;
    }
}

public class TripletSelect {
    int[] getBs(Triplet[] triplets) {
        int[] bs = new int[triplets.length];
        for (int i = 0; i < triplets.length; i++) {
            bs[i] = triplets[i].b;
        }
        return bs;
    }

    // tests
    // expect [2, 5, 8]
    Triplet[] test1 = { new Triplet(a:1, b:2, c:3), new Triplet(a:4, b:5, c:6), new Triplet(a:7, b:8, c:9) };
    // expect []
    Triplet[] test2 = {};
    // expect [2]
    Triplet[] test3 = { new Triplet(a:1, b:2, c:3) };
    // expect [2, 5, -2, 11]
    Triplet[] test4 = { new Triplet(a:1, b:2, c:3), new Triplet(a:4, b:5, c:6), new Triplet(a:7, -2, c:9), new Triplet(a:10, b:11, c:12) };

    boolean testBs(Tester t) {
        return t.checkExpect(this.getBs(test1), new int[] { 2, 5, 8 })
                && t.checkExpect(this.getBs(test2), new int[] {})
                && t.checkExpect(this.getBs(test3), new int[] { 2 })
                && t.checkExpect(this.getBs(test4), new int[] { 2, 5, -2, 11 });
    }
}
```

```java
import java.util.ArrayList;

public class PositiveMinMaxAvg {
    Run | Debug
    public static void main(String[] args) {
        // catch less than 2 args
        if (args.length < 2) {
            System.out.println(x:0);
            System.exit(status:0);
        }
        // parse args to doubles
        ArrayList<Double> nums = new ArrayList<>();
        for (int i = 0; i < args.length; i++) {
            try {
                double num = Double.parseDouble(args[i]);
                // only add num if it's positive
                if (num >= 0) {
                    nums.add(num);
                }
                // if parseDouble fails it throws a NumberFormatException
            } catch (NumberFormatException e) {
                System.out.println(x:"Arguments must be valid doubles");
                // exit with failure
                System.exit(status:1);
            }
        }
        // after this nums could be empty if all negative in args
        if (nums.size() == 0) {
            System.out.println(x:0);
            System.exit(status:0);
        }

        // find min and max
        double min = nums.get(index:0);
        double max = nums.get(index:0);
        for (Double n : nums) {
            if (n < min) {
                min = n;
            }
            if (n > max) {
                max = n;
            }
        }
        System.out.println((min + max) / 2.0);
    }

    /**
     * Test cases (6):
     * From PA6:
     * -1.5 2.5 3.5 5 : expect 3.75
     * (nothing) : expect 0
     * 1 : expect 0
     * 1 2 : expect 1.5
     * 1 7.8 -21.4: expect 4.4
     * -1.5 -2.5 -3.5 -4.5 : expect 0
     */
```

At the end of the second screenshot is just ending curly braces

Screenshot or copy/paste of ./run (if any):

```Java
❯ ./run TripletSelect
Tester Library v.3.0
------------------------------------
Tests defined in the class: TripletSelect:
---------------------------
TripletSelect:
---------------
new TripletSelect:1(
 this.test1 = new Triplet[3]:2{
  [0] new Triplet:3(
   this.a = 1
   this.b = 2
   this.c = 3),
  [1] new Triplet:4(
   this.a = 4
   this.b = 5
   this.c = 6),
  [2] new Triplet:5(
   this.a = 7
   this.b = 8
   this.c = 9)
 }
 this.test2 = new Triplet[0]:6{
 }
 this.test3 = new Triplet[1]:7{
  [0] new Triplet:8(
   this.a = 1
   this.b = 2
   this.c = 3)
 }
 this.test4 = new Triplet[4]:9{
  [0] new Triplet:10(
   this.a = 1
   this.b = 2
   this.c = 3),
  [1] new Triplet:11(
   this.a = 4
   this.b = 5
   this.c = 6),
  [2] new Triplet:12(
   this.a = 7
```

```
    this.b = -2
    this.c = 9),
   [3] new Triplet:13(
    this.a = 10
    this.b = 11
    this.c = 12)
  })
---------------

Ran 4 tests.
All tests passed.

--- END OF TEST RESULTS ---
WARNING: A terminally deprecated method in java.lang.System has been called
WARNING: System::setSecurityManager has been called by tester.Main
(file:/Users/zoophere/Library/Mobile%20Documents/iCloud~md~obsidian/Documents/W
orkspace/CourseMaterials/WI24/CSE-11/psets-repo/cse11-pa6-starter/tester.jar)
WARNING: Please consider reporting this to the maintainers of tester.Main
WARNING: System::setSecurityManager will be removed in a future release
```

```
    * Test cases (6):
    * From PA6:
    * -1.5 2.5 3.5 5 : expect 3.75
    * (nothing) : expect 0
    * 1 : expect 0
    * 1 2 : expect 1.5
    * 1 7.8 -23.4: expect 4.4
    * -1.5 -2.5 -3.5 -4.5 : expect 0
    */
  }
```

PROBLEMS 8 OUTPUT DEBUG CONSOLE TERMINAL PORTS GITLENS

```
❯ javac PositiveMinMaxAvg.java
❯ java PositiveMinMaxAvg -1.5 2.5 3.5 5
3.75
❯ java PositiveMinMaxAvg
0
❯ java PositiveMinMaxAvg 1
0
❯ java PositiveMinMaxAvg 1 2
1.5
❯ java PositiveMinMaxAvg 1 7.8 -23.4
4.4
❯ java PositiveMinMaxAvg -1.5 -2.5 -3.5 -4.5
0
```

Thoughts on your progress:

I thought these exercises were pretty easy compared to the other programming assignments. One issue I ran into was initially I processed all arguments into an array of doubles and then did the min and max algorithm. What I ran into was that if `nums[0]` was negative, then assigning `double min = nums[0]` messed up the algorithm because even though I was only changing min or max in the for loop if `nums[i]` was positive, the negative would be stuck into the algorithm from the first assignment making the final average wrong. First, I tried to only assign `nums[0]` if it was positive, but quickly realized that to do this, I would have to loop over the entire array to find a positive value. Then, if there were only negative values, it would never work. So, I decided to use an `ArrayList` instead and push values only if they were positive while parsing `args,` which solved the problem.

I finished with 3 minutes left on my timer.

Distractions:

No

Second 30 Minutes

Screenshot or copy/paste of program:

Finished in the first 30 minutes.

Screenshot or copy/paste of ./run (if any):

Thoughts on your progress:

Distractions:

Final 30 Minutes

Screenshot or copy/paste of program:

Finished in the first 30 minutes.

Screenshot or copy/paste of ./run (if any):

Thoughts on your progress:

Distractions:

Overall Reflection

I spent the most time debugging the issue I outlined in the first reflection. I also thought for a while about if it would be ok to import `ArrayList` and decided to do it because it was the simplest solution I could think of to the error.

I also spent a good amount of time on getting the tests to run for `TripletSelect` because I defined the `Triplet` class after `TripletSelect` so when the tests tried to run it couldn't find the `Triplet` class.

In the future, I think I would think more about the order of operations before writing code. I am reasonably comfortable with basic algorithms because I took a data structures course at my old community college. So, figuring out which solution should come first in the intermediate steps would save time overall. If I had thought more about how to parse the doubles from `PositiveMinMaxAvg` from the beginning, I could have avoided the bug I ran into.

I think that the process for this PA was more straightforward because the outline for the issues was simple for the exercises I chose because there was no existing code to understand (I avoided the one with the Regions code) and no complex class structure and relationships to understand.

I think what I took away from this exercise was that a lot of programming is done in thinking, not actual writing. I liked the example of "programming" while walking the dog because it shows that its important to think about a problem's structure before sitting down to write.