**Looping Construct with Floating Point Numbers – Option 1 – CT4**

Logan LeBoeuf

Colorado State University Global

Course Code: CSC - 320

Dr. Vanessa Cooper

March 9, 2025

**Looping Construct with Floating Point Numbers**

So, approaching this program, I will do my usual of using main to call the method doing the heavy lifting, with another method to return the inputs. Because I also need to calculate things like the max, minimum, average, total, and interest at 20%, having each of those as a method I can call one at a time feels tidy. Since it must prevent an endless loop, the program should ask for input, see if it’s valid, and if the input is invalid three times, it will cancel everything and close.

|  |
| --- |
| **main return void method():**          printDataMethod(getInputMethod())      } |
| **Float[] getInputMethod ():**  //This should return an array of values it asks the user for. This is where a while loop will run, iterating on a counter that only goes up when a valid is valid, and breaking the loop if too many invalid values are entered. I also need to add semicolons and actually initialize things because this isn’t Python no matter how much my muscle memory insists on it.  //Since the Float object can parse floats and throws an error if a value isn’t one, I can use try-catch for this.  Scanner = new Scanner  Int counter = 0;  Int badInputCounter = 0;  Float[] outputArray = Float[5];  While (counter < 5) {  Try {  Print(“Begin entering values.” + current count, “ out of 5 total”);  Input = scanner.nextLine;  parsedInput = parseFloat(input);  //If there’s an error it’ll be thrown here. If not, add the value to the array and iterate.  outputArray[counter] = parsedInput;  counter++; }    catch (formatting error) {  badInputCounter++;  Print(“Bad input, “ + badInputCounter + “ of 3.”);  If (badInputCounter > 2){  Break;}  }  Close scanner  Return outputArray |
| //I need total, avg, max, min, and interest on 20%.  **Float getTotal(Float[] inputArray){**  Total = 0;  For each value in InputArray {  Total += value;  }  Return Total;  }  **Float getAverage(Float[] inputArray**){  Average = getTotal(inputArray) / 5;  Return Average;  }  **Float getMax(Float[] inputArray**){  //I can just compare a value to each value in the array and if it’s higher, assign that.  Max = Float.NEGATIVE\_INFINITY  For each value in InputArray {  if Value > Max {  Max = Value;  }  }  Return Max;  }  **Float getMin(Float[] inputArray**){  //Just like above but opposite?  Min = Float.POSITIVE\_INFINITY  For each value in InputArray {  if Value < Min {  Min = Value;  }  }  Return Min;  }  **Float getInterest(Float[] inputArray**){  //If there’s five values, then 20% interest on the total should be the same as the average  Return getTotal(inputArray) \* 0.20f;  }  **Void printDataMethod(Float[] inputArray){**  //The while loop is taken care of, so I just need to see if any value in the array it returned was null. If any value is null, I can just stop. If they’re all good to go, then it can commence.  Bool valid = true;  For each value in inputArray{  If value == null{  valid = false;  }  }  If valid{  Float total = getTotal(inputArray);  Float avg = getAverage(inputArray);  Float max = getMax(inputArray);  Float min = getMin(inputArray);  Float interest = getInterest(inputArray);  Print(“Total: “ + total);  Print(“Averagel: “ + avg);  Print(“Max: “ + max);  Print(“Min: “ + min);  Print(“Interest on total at 20%: “ + interest);  }  Else {  Print(“Program did not run successfully, please try again.”)  }  } |

**Source Code**

A screen shot of a computer program

AI-generated content may be incorrect.Trying to commit to brackets in the pseudocode may have been a mistake, but it helped with the resulting code.

A screenshot of a computer program

AI-generated content may be incorrect.

**Application Executing**

The application asks the user for values. If the value can’t parse as a float, it throws an error. After three errors, the program tells the user it didn’t run successfully. If the user manages to input five values that actually parse, then it continues on and prints the requested information (total, average, max, min, and interest on 20% of the total, which should be the same value as average.)

**A computer screen with white text

AI-generated content may be incorrect.**

**Git Repository**

Afterwards, this was pushed to my git repository.

A screenshot of a computer

AI-generated content may be incorrect.