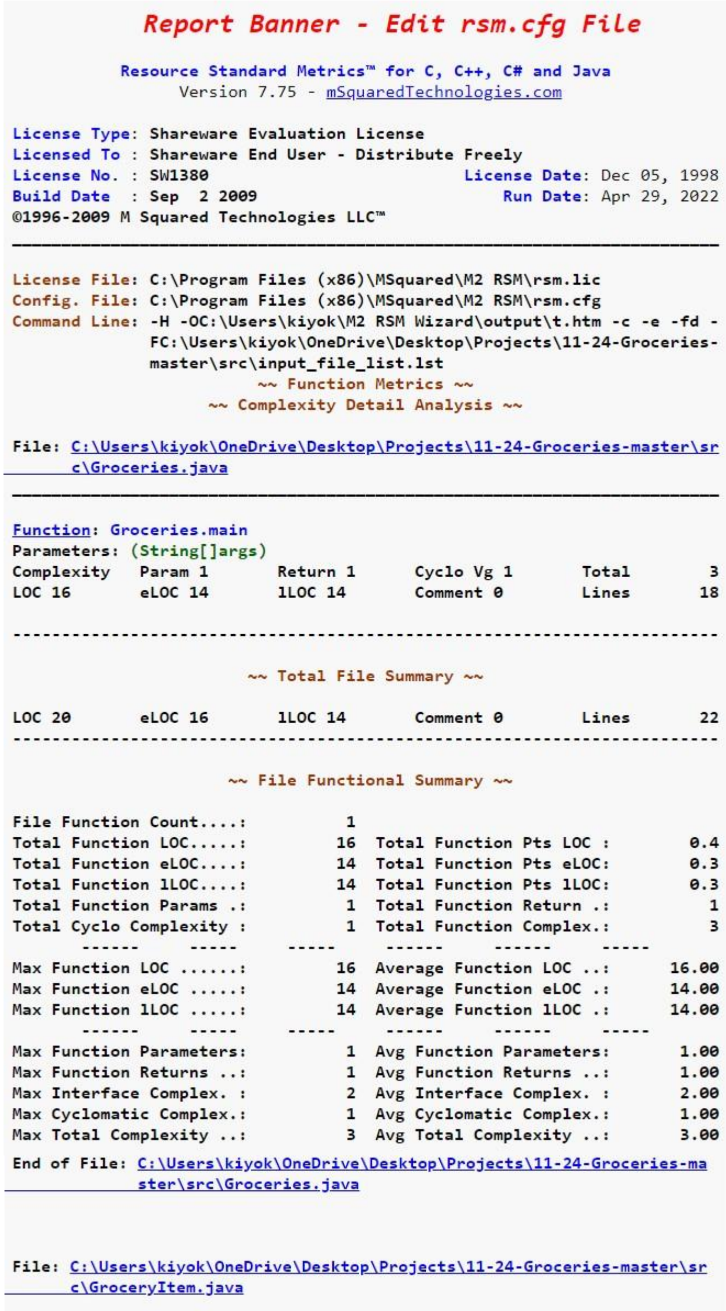
Software Maintainability

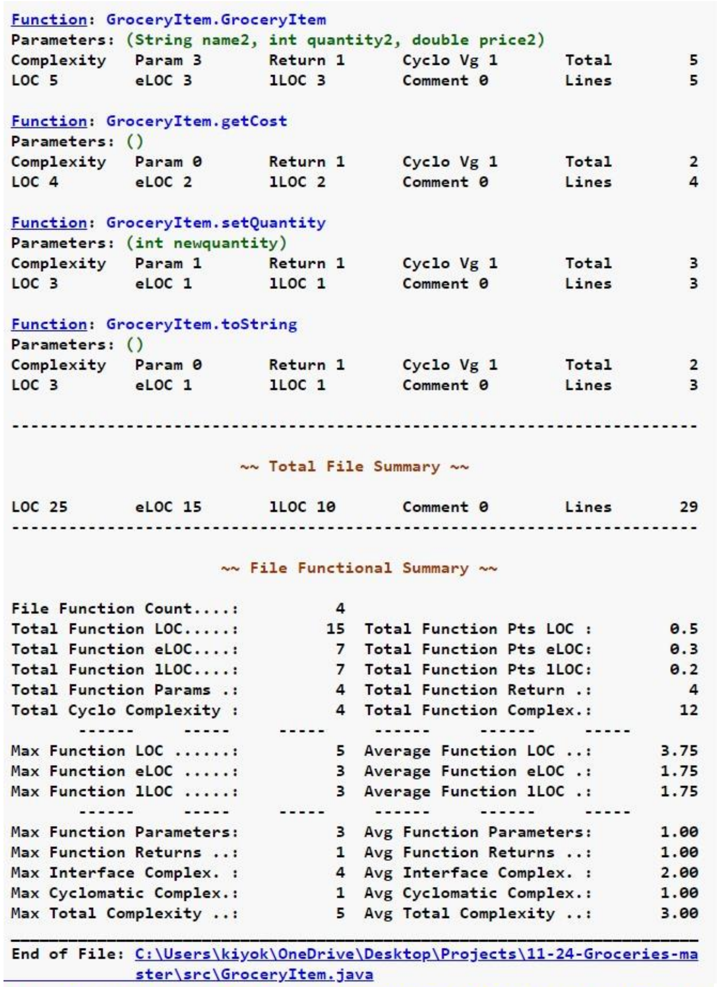
Part 1: Tool Demonstration

Tool Demonstration

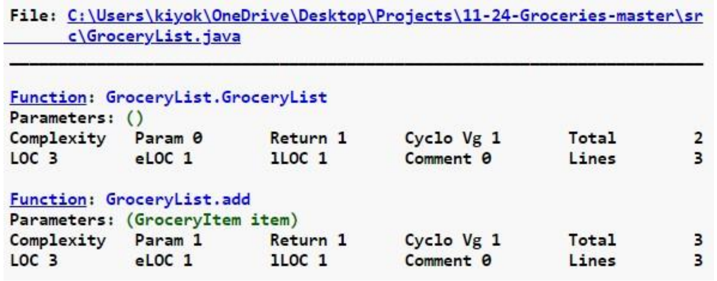
The program used for this demonstration is a grocery list creator written in java [1].



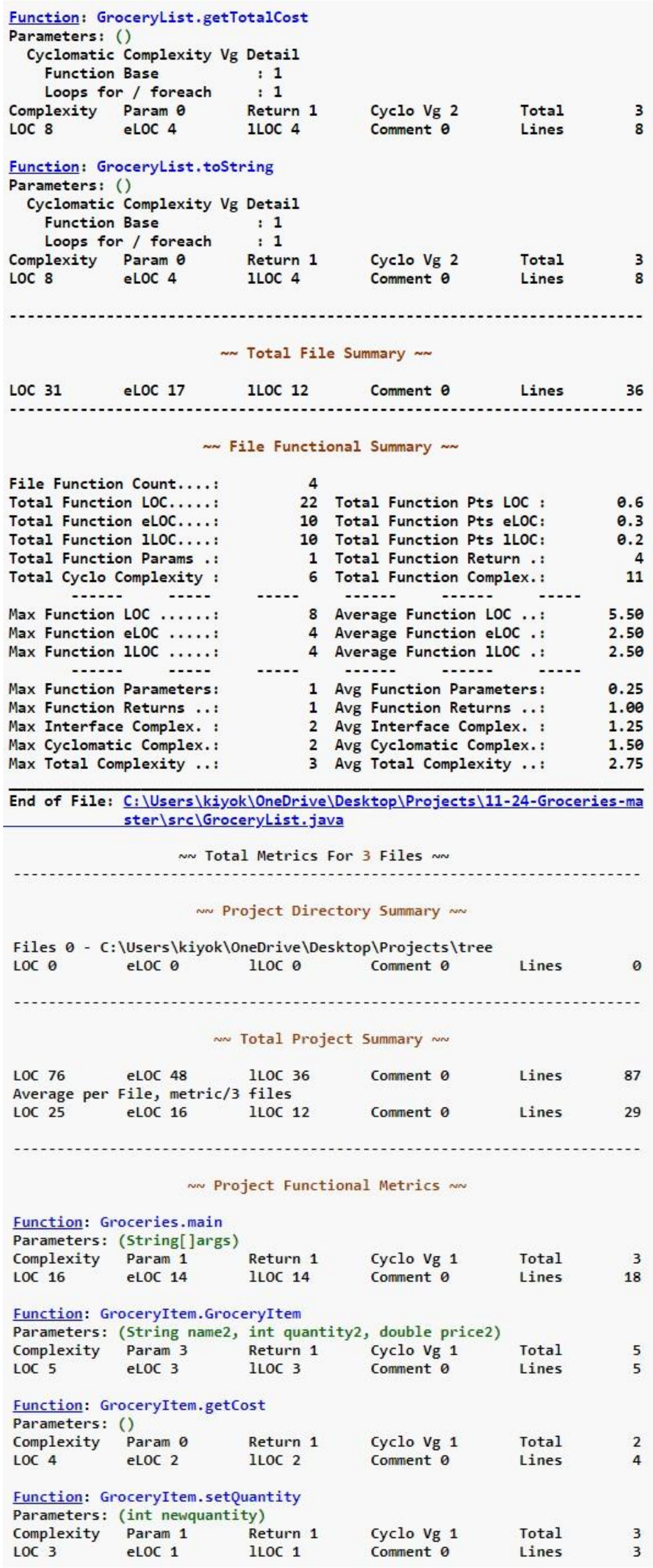
```
              Report Banner - Edit rsm.cfg File

        Resource Standard Metrics™ for C, C++, C# and Java
              Version 7.75 - mSquaredTechnologies.com

License Type: Shareware Evaluation License
Licensed To : Shareware End User - Distribute Freely
License No. : SW1380                       License Date: Dec 05, 1998
Build Date  : Sep  2 2009                      Run Date: Apr 29, 2022
©1996-2009 M Squared Technologies LLC™
______________________________________________________________________

License File: C:\Program Files (x86)\MSquared\M2 RSM\rsm.lic
Config. File: C:\Program Files (x86)\MSquared\M2 RSM\rsm.cfg
Command Line: -H -OC:\Users\kiyok\M2 RSM Wizard\output\t.htm -c -e -fd -
              FC:\Users\kiyok\OneDrive\Desktop\Projects\11-24-Groceries-
              master\src\input_file_list.lst
                         ~~ Function Metrics ~~
                    ~~ Complexity Detail Analysis ~~

File: C:\Users\kiyok\OneDrive\Desktop\Projects\11-24-Groceries-master\sr
      c\Groceries.java
______________________________________________________________________

Function: Groceries.main
Parameters: (String[]args)
Complexity   Param 1       Return 1      Cyclo Vg 1       Total       3
LOC 16       eLOC 14       1LOC 14       Comment 0        Lines      18

----------------------------------------------------------------------

                        ~~ Total File Summary ~~

LOC 20       eLOC 16       1LOC 14       Comment 0        Lines      22
----------------------------------------------------------------------

                     ~~ File Functional Summary ~~

File Function Count....:          1
Total Function LOC.....:         16  Total Function Pts LOC :      0.4
Total Function eLOC....:         14  Total Function Pts eLOC:      0.3
Total Function 1LOC....:         14  Total Function Pts 1LOC:      0.3
Total Function Params .:          1  Total Function Return .:        1
Total Cyclo Complexity :          1  Total Function Complex.:        3
       ------     -----     -----     ------     ------     -----
Max Function LOC ......:         16  Average Function LOC ..:    16.00
Max Function eLOC .....:         14  Average Function eLOC .:    14.00
Max Function 1LOC .....:         14  Average Function 1LOC .:    14.00
       ------     -----     -----     ------     ------     -----
Max Function Parameters:          1  Avg Function Parameters:     1.00
Max Function Returns ..:          1  Avg Function Returns ..:     1.00
Max Interface Complex. :          2  Avg Interface Complex. :     2.00
Max Cyclomatic Complex.:          1  Avg Cyclomatic Complex.:     1.00
Max Total Complexity ..:          3  Avg Total Complexity ..:     3.00
End of File: C:\Users\kiyok\OneDrive\Desktop\Projects\11-24-Groceries-ma
             ster\src\Groceries.java


File: C:\Users\kiyok\OneDrive\Desktop\Projects\11-24-Groceries-master\sr
      c\GroceryItem.java
```



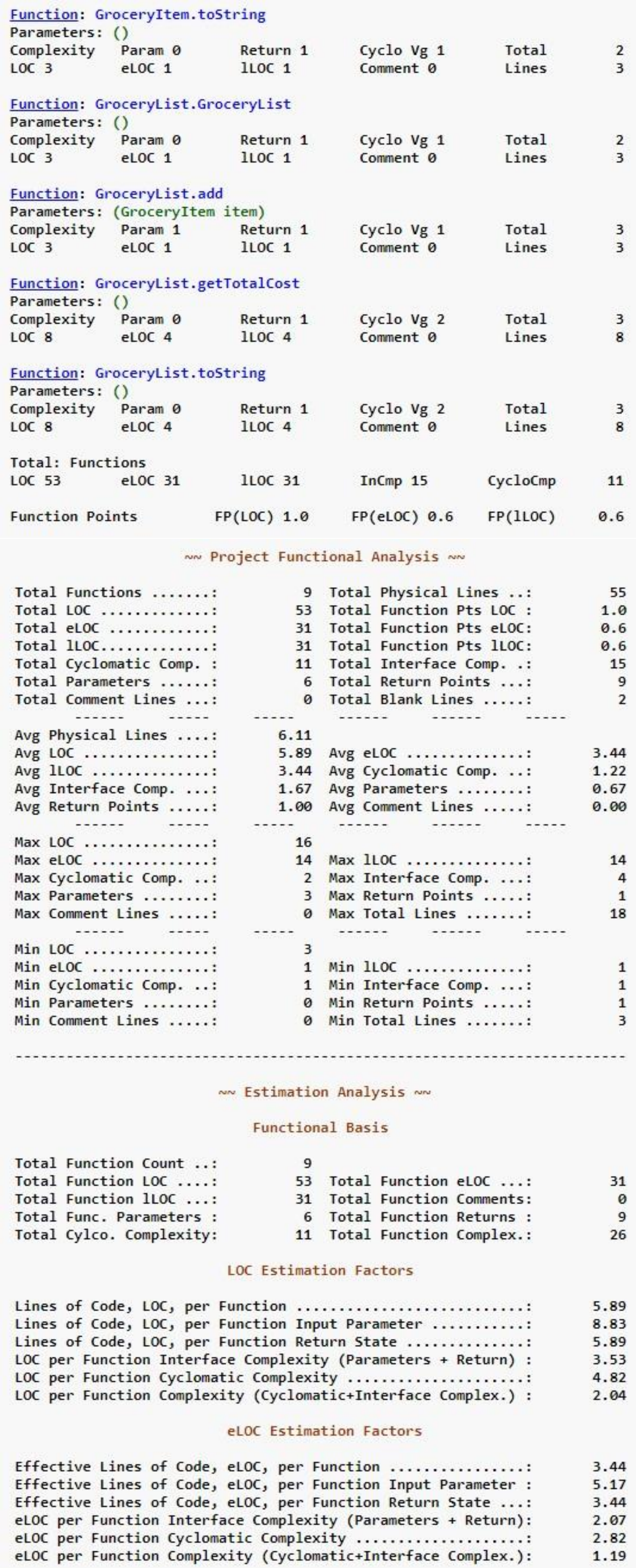
```
Function: GroceryItem.GroceryItem
Parameters: (String name2, int quantity2, double price2)
Complexity   Param 3       Return 1      Cyclo Vg 1       Total       5
LOC 5        eLOC 3        1LOC 3        Comment 0        Lines       5

Function: GroceryItem.getCost
Parameters: ()
Complexity   Param 0       Return 1      Cyclo Vg 1       Total       2
LOC 4        eLOC 2        1LOC 2        Comment 0        Lines       4

Function: GroceryItem.setQuantity
Parameters: (int newquantity)
Complexity   Param 1       Return 1      Cyclo Vg 1       Total       3
LOC 3        eLOC 1        1LOC 1        Comment 0        Lines       3

Function: GroceryItem.toString
Parameters: ()
Complexity   Param 0       Return 1      Cyclo Vg 1       Total       2
LOC 3        eLOC 1        1LOC 1        Comment 0        Lines       3

----------------------------------------------------------------------

                        ~~ Total File Summary ~~

LOC 25       eLOC 15       1LOC 10       Comment 0        Lines      29
----------------------------------------------------------------------

                     ~~ File Functional Summary ~~

File Function Count....:          4
Total Function LOC.....:         15  Total Function Pts LOC :      0.5
Total Function eLOC....:          7  Total Function Pts eLOC:      0.3
Total Function 1LOC....:          7  Total Function Pts 1LOC:      0.2
Total Function Params .:          4  Total Function Return .:        4
Total Cyclo Complexity :          4  Total Function Complex.:       12
       ------     -----     -----     ------     ------     -----
Max Function LOC ......:          5  Average Function LOC ..:     3.75
Max Function eLOC .....:          3  Average Function eLOC .:     1.75
Max Function 1LOC .....:          3  Average Function 1LOC .:     1.75
       ------     -----     -----     ------     ------     -----
Max Function Parameters:          3  Avg Function Parameters:     1.00
Max Function Returns ..:          1  Avg Function Returns ..:     1.00
Max Interface Complex. :          4  Avg Interface Complex. :     2.00
Max Cyclomatic Complex.:          1  Avg Cyclomatic Complex.:     1.00
Max Total Complexity ..:          5  Avg Total Complexity ..:     3.00
______________________________________________________________________
End of File: C:\Users\kiyok\OneDrive\Desktop\Projects\11-24-Groceries-ma
             ster\src\GroceryItem.java
```



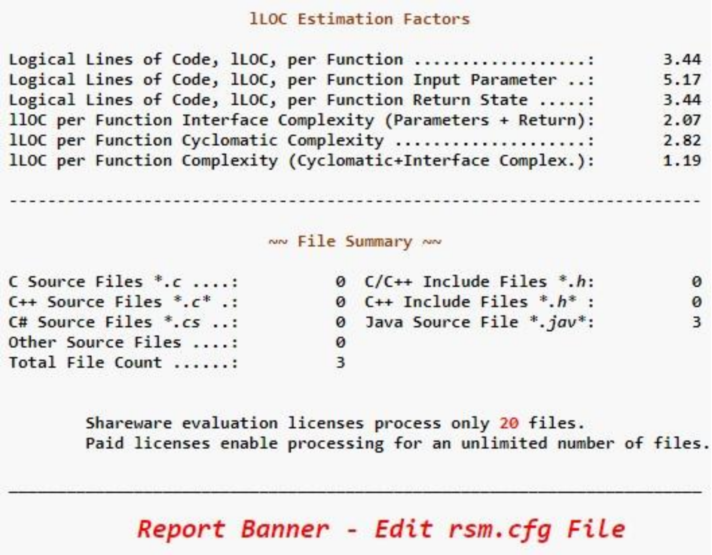
```
File: C:\Users\kiyok\OneDrive\Desktop\Projects\11-24-Groceries-master\sr
      c\GroceryList.java
______________________________________________________________________

Function: GroceryList.GroceryList
Parameters: ()
Complexity   Param 0       Return 1      Cyclo Vg 1       Total       2
LOC 3        eLOC 1        1LOC 1        Comment 0        Lines       3

Function: GroceryList.add
Parameters: (GroceryItem item)
Complexity   Param 1       Return 1      Cyclo Vg 1       Total       3
LOC 3        eLOC 1        1LOC 1        Comment 0        Lines       3
```



```
Function: GroceryList.getTotalCost
Parameters: ()
  Cyclomatic Complexity Vg Detail
    Function Base          : 1
    Loops for / foreach    : 1
Complexity   Param 0      Return 1      Cyclo Vg 2      Total        3
LOC 8        eLOC 4       lLOC 4        Comment 0       Lines        8

Function: GroceryList.toString
Parameters: ()
  Cyclomatic Complexity Vg Detail
    Function Base          : 1
    Loops for / foreach    : 1
Complexity   Param 0      Return 1      Cyclo Vg 2      Total        3
LOC 8        eLOC 4       lLOC 4        Comment 0       Lines        8

------------------------------------------------------------------------

                      ~~ Total File Summary ~~

LOC 31       eLOC 17      lLOC 12       Comment 0       Lines       36
------------------------------------------------------------------------

                    ~~ File Functional Summary ~~

File Function Count....:           4
Total Function LOC.....:          22  Total Function Pts LOC :     0.6
Total Function eLOC....:          10  Total Function Pts eLOC:     0.3
Total Function lLOC....:          10  Total Function Pts lLOC:     0.2
Total Function Params .:           1  Total Function Return .:       4
Total Cyclo Complexity :           6  Total Function Complex.:      11
       ------     -----     -----     ------     ------     -----
Max Function LOC ......:           8  Average Function LOC ..:    5.50
Max Function eLOC .....:           4  Average Function eLOC .:    2.50
Max Function lLOC .....:           4  Average Function lLOC .:    2.50
       ------     -----     -----     ------     ------     -----
Max Function Parameters:           1  Avg Function Parameters:    0.25
Max Function Returns ..:           1  Avg Function Returns ..:    1.00
Max Interface Complex. :           2  Avg Interface Complex. :    1.25
Max Cyclomatic Complex.:           2  Avg Cyclomatic Complex.:    1.50
Max Total Complexity ..:           3  Avg Total Complexity ..:    2.75
________________________________________________________________________
End of File: C:\Users\kiyok\OneDrive\Desktop\Projects\11-24-Groceries-ma
             ster\src\GroceryList.java

                    ~~ Total Metrics For 3 Files ~~
------------------------------------------------------------------------

                    ~~ Project Directory Summary ~~

Files 0 - C:\Users\kiyok\OneDrive\Desktop\Projects\tree
LOC 0        eLOC 0       lLOC 0        Comment 0       Lines        0

------------------------------------------------------------------------

                     ~~ Total Project Summary ~~

LOC 76       eLOC 48      lLOC 36       Comment 0       Lines       87
Average per File, metric/3 files
LOC 25       eLOC 16      lLOC 12       Comment 0       Lines       29

------------------------------------------------------------------------

                   ~~ Project Functional Metrics ~~

Function: Groceries.main
Parameters: (String[]args)
Complexity   Param 1      Return 1      Cyclo Vg 1      Total        3
LOC 16       eLOC 14      lLOC 14       Comment 0       Lines       18

Function: GroceryItem.GroceryItem
Parameters: (String name2, int quantity2, double price2)
Complexity   Param 3      Return 1      Cyclo Vg 1      Total        5
LOC 5        eLOC 3       lLOC 3        Comment 0       Lines        5

Function: GroceryItem.getCost
Parameters: ()
Complexity   Param 0      Return 1      Cyclo Vg 1      Total        2
LOC 4        eLOC 2       lLOC 2        Comment 0       Lines        4

Function: GroceryItem.setQuantity
Parameters: (int newquantity)
Complexity   Param 1      Return 1      Cyclo Vg 1      Total        3
LOC 3        eLOC 1       lLOC 1        Comment 0       Lines        3
```



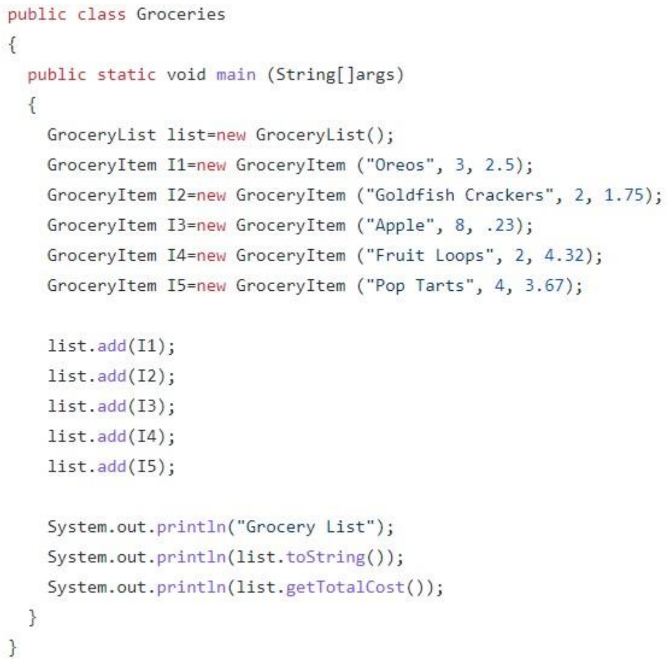
```
Function: GroceryItem.toString
Parameters: ()
Complexity   Param 0      Return 1      Cyclo Vg 1      Total        2
LOC 3        eLOC 1       lLOC 1        Comment 0       Lines        3

Function: GroceryList.GroceryList
Parameters: ()
Complexity   Param 0      Return 1      Cyclo Vg 1      Total        2
LOC 3        eLOC 1       lLOC 1        Comment 0       Lines        3

Function: GroceryList.add
Parameters: (GroceryItem item)
Complexity   Param 1      Return 1      Cyclo Vg 1      Total        3
LOC 3        eLOC 1       lLOC 1        Comment 0       Lines        3

Function: GroceryList.getTotalCost
Parameters: ()
Complexity   Param 0      Return 1      Cyclo Vg 2      Total        3
LOC 8        eLOC 4       lLOC 4        Comment 0       Lines        8

Function: GroceryList.toString
Parameters: ()
Complexity   Param 0      Return 1      Cyclo Vg 2      Total        3
LOC 8        eLOC 4       lLOC 4        Comment 0       Lines        8

Total: Functions
LOC 53       eLOC 31      lLOC 31       InCmp 15       CycloCmp     11

Function Points         FP(LOC) 1.0    FP(eLOC) 0.6   FP(lLOC)     0.6

                ~~ Project Functional Analysis ~~

Total Functions .......:          9  Total Physical Lines ..:        55
Total LOC .............:         53  Total Function Pts LOC :       1.0
Total eLOC ............:         31  Total Function Pts eLOC:       0.6
Total lLOC.............:         31  Total Function Pts lLOC:       0.6
Total Cyclomatic Comp. :         11  Total Interface Comp. .:        15
Total Parameters ......:          6  Total Return Points ...:         9
Total Comment Lines ...:          0  Total Blank Lines .....:         2
      ------     -----     -----     ------     ------     -----
Avg Physical Lines ....:       6.11
Avg LOC ...............:       5.89  Avg eLOC ..............:      3.44
Avg lLOC ..............:       3.44  Avg Cyclomatic Comp. ..:      1.22
Avg Interface Comp. ...:       1.67  Avg Parameters ........:      0.67
Avg Return Points .....:       1.00  Avg Comment Lines .....:      0.00
      ------     -----     -----     ------     ------     -----
Max LOC ...............:         16
Max eLOC ..............:         14  Max lLOC ..............:        14
Max Cyclomatic Comp. ..:          2  Max Interface Comp. ...:         4
Max Parameters ........:          3  Max Return Points .....:         1
Max Comment Lines .....:          0  Max Total Lines .......:        18
      ------     -----     -----     ------     ------     -----
Min LOC ...............:          3
Min eLOC ..............:          1  Min lLOC ..............:         1
Min Cyclomatic Comp. ..:          1  Min Interface Comp. ...:         1
Min Parameters ........:          0  Min Return Points .....:         1
Min Comment Lines .....:          0  Min Total Lines .......:         3

------------------------------------------------------------------------

                     ~~ Estimation Analysis ~~

                        Functional Basis

Total Function Count ..:          9
Total Function LOC ....:         53  Total Function eLOC ...:        31
Total Function lLOC ...:         31  Total Function Comments:         0
Total Func. Parameters :          6  Total Function Returns :         9
Total Cylco. Complexity:         11  Total Function Complex.:        26

                      LOC Estimation Factors

Lines of Code, LOC, per Function ...........................:      5.89
Lines of Code, LOC, per Function Input Parameter ...........:      8.83
Lines of Code, LOC, per Function Return State ..............:      5.89
LOC per Function Interface Complexity (Parameters + Return) :      3.53
LOC per Function Cyclomatic Complexity .....................:      4.82
LOC per Function Complexity (Cyclomatic+Interface Complex.) :      2.04

                      eLOC Estimation Factors

Effective Lines of Code, eLOC, per Function ................:      3.44
Effective Lines of Code, eLOC, per Function Input Parameter :      5.17
Effective Lines of Code, eLOC, per Function Return State ...:      3.44
eLOC per Function Interface Complexity (Parameters + Return):      2.07
eLOC per Function Cyclomatic Complexity ....................:      2.82
eLOC per Function Complexity (Cyclomatic+Interface Complex.):      1.19
```



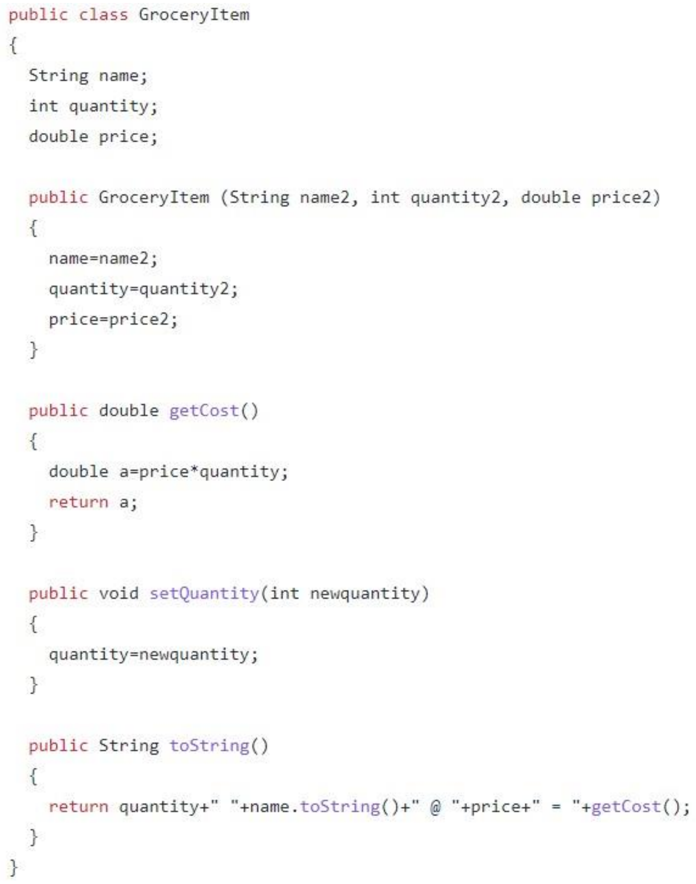
```
                        lLOC Estimation Factors

Logical Lines of Code, lLOC, per Function ..................:       3.44
Logical Lines of Code, lLOC, per Function Input Parameter ..:       5.17
Logical Lines of Code, lLOC, per Function Return State .....:       3.44
llOC per Function Interface Complexity (Parameters + Return):       2.07
lLOC per Function Cyclomatic Complexity ....................:       2.82
lLOC per Function Complexity (Cyclomatic+Interface Complex.):       1.19

------------------------------------------------------------------------

                          ~~ File Summary ~~

C Source Files *.c ....:          0  C/C++ Include Files *.h:          0
C++ Source Files *.c* .:          0  C++ Include Files *.h* :          0
C# Source Files *.cs ..:          0  Java Source File *.jav*:          3
Other Source Files ....:          0
Total File Count ......:          3


        Shareware evaluation licenses process only 20 files.
        Paid licenses enable processing for an unlimited number of files.

________________________________________________________________________

             Report Banner - Edit rsm.cfg File
```

Additional metrics I added within this report is estimate labor rate analysis based on LOC and complexity. This provided the total count of functions LOC and complexity to find the function averages of Loc and complexity. From this information an analyst can derive an estimation cost projection for labor based on function averages. The file functional summary provides an estimation of maximum LOC within each function and function complexity while maintaining the quality level of the project. This is pertinent for this program if the programmer decides on further developing the program.

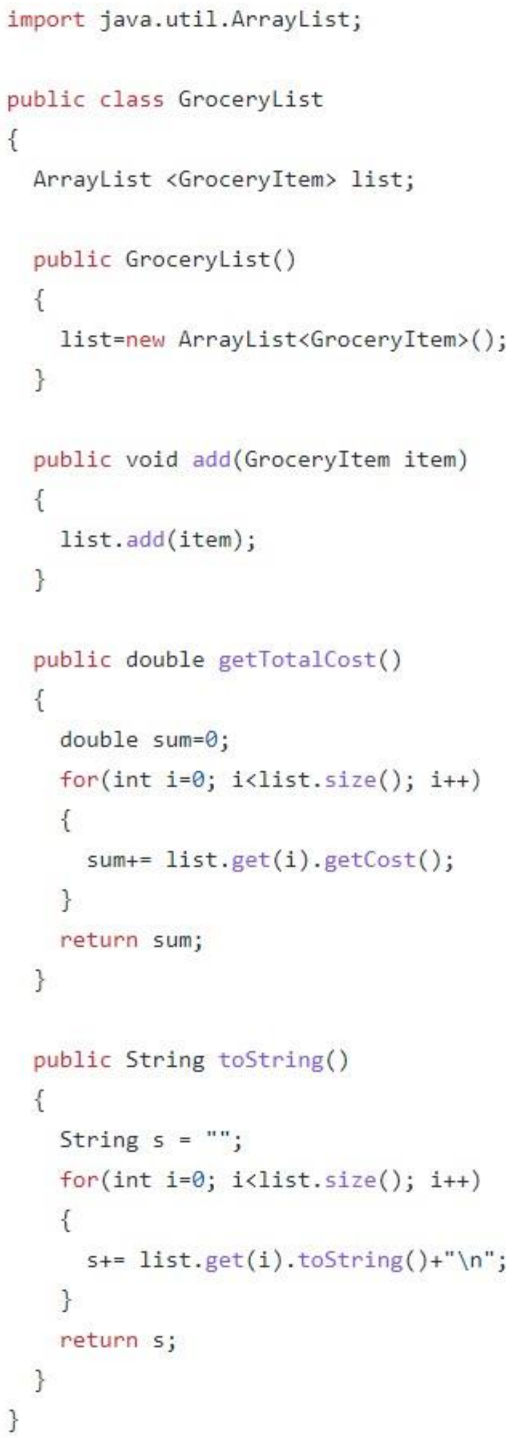
Metrics Report and Source Code



```
public class Groceries
{
  public static void main (String[]args)
  {
    GroceryList list=new GroceryList();
    GroceryItem I1=new GroceryItem ("Oreos", 3, 2.5);
    GroceryItem I2=new GroceryItem ("Goldfish Crackers", 2, 1.75);
    GroceryItem I3=new GroceryItem ("Apple", 8, .23);
    GroceryItem I4=new GroceryItem ("Fruit Loops", 2, 4.32);
    GroceryItem I5=new GroceryItem ("Pop Tarts", 4, 3.67);

    list.add(I1);
    list.add(I2);
    list.add(I3);
    list.add(I4);
    list.add(I5);

    System.out.println("Grocery List");
    System.out.println(list.toString());
    System.out.println(list.getTotalCost());
  }
}
```



```
public class GroceryItem
{
  String name;
  int quantity;
  double price;

  public GroceryItem (String name2, int quantity2, double price2)
  {
    name=name2;
    quantity=quantity2;
    price=price2;
  }

  public double getCost()
  {
    double a=price*quantity;
    return a;
  }

  public void setQuantity(int newquantity)
  {
    quantity=newquantity;
  }

  public String toString()
  {
    return quantity+" "+name.toString()+" @ "+price+" = "+getCost();
  }
}
```



```
import java.util.ArrayList;

public class GroceryList
{
  ArrayList <GroceryItem> list;

  public GroceryList()
  {
    list=new ArrayList<GroceryItem>();
  }

  public void add(GroceryItem item)
  {
    list.add(item);
  }

  public double getTotalCost()
  {
    double sum=0;
    for(int i=0; i<list.size(); i++)
    {
      sum+= list.get(i).getCost();
    }
    return sum;
  }

  public String toString()
  {
    String s = "";
    for(int i=0; i<list.size(); i++)
    {
      s+= list.get(i).toString()+"\n";
    }
    return s;
  }
}
```

Part 2: Software Maintainability Measure

Maintainability Measure Identification

The author describes maintainability as a set of attributes that impact the level of effort necessary to conduct specified modifications [2]. Software maintainability is measured through the readability of the software's source code (RSC), the quality of its documentation (DOQ) , and the understandability of software (UOS). These three aspects are significant for understanding an application overall, how the software is revised overtime, and the enhancement of the software [2]. This paper talks about complexity and maintainability measures on lexical levels. These measures include Lines of Code (LOC), total number of lines commented, Halstead length, Halstead volume, Halstead effort, number of blank lines, the number of executable semicolons or statements, the average number of statements between two references that are successive to the same variable, and etc. [2]. Some of these measures are estimated in the nondefault metrics I chose from the first part, estimate labor rate analysis based on LOC and complexity. These maintainability measures are calculated through the construction of formulas. Formulas such that approach maintainability with a function of directly measurable attributes between A_1 and A_n ; this function is written as so $M = \int(A_1, A_2, \dots, A_n)$ [2]. Approaching the formula may look simple on the surface, but many difficulties may arise when measuring such attributes against one another then combining these attributes in a function of $\int$ [2]. There are many resources allocated towards the maintenance of software overall, which sizably impacts costs [2]. Studies of how measures change overtime as the software is maintained were to verify the predictions of maintenance cost. Land describes how long-term software maintenance can deteriorate software overtime, thus impacting cost [2].

Relation to RSM Tool Metrics

The lexical level measurement method is related to the RSM tool metrics by using maintainability functions to calculate the total and averages of LOC and complexity within a project. With the RSM tool an analyst can effortlessly get function estimations for the average number of return points, maximum number of commented lines, or total number of blank lines. These calculations fall under the category of maintainability measures on lexical levels. Using the RSM tool metrics is beneficial in the ease of said calculations for large projects or in the developmental process of any software project.

References

1. Brooks, L. GitHub - lucybrooks/11-24-Groceries: Program to build a grocery list. *GitHub*, 2022. https://github.com/lucybrooks/11-24-Groceries.
2. Land, Rikard. 3.*Artes.uu.se*, 2022. Measurements of Software Maintainability. 2022. http://www.artes.uu.se/events/gsconf02/papers/Land_Maintainability.pdf.