**SHOPPING CART PROJECT PART 2**

**Table of Contents**

1. Cover Page
2. Table of Contents
3. Statement of Independent Effort
4. Analysis of Specification
5. Pseudocode
6. Flowchart
7. Test Cases
8. Code
9. Grade Sheet

# Statement of Independent Effort

I, JaKeyvan Jones, hereby certify that is my original work completed without the assistance of anyone or any outside resources.

JaKeyvan Jones

**Analysis of Specification**

**If/Elseif**

If memberLevel == "Diamond" “Begins statement on what happens if member has Diamond level account”

If totalSpend > 700.00 “Begins statement on what happens if user with Diamond level account spends at least $700”

Elseif totalSpend > 300.00 “Begins statement on what happens if user with Diamond level account spends at least $300”

Elseif totalSpend > 100.00 “Begins statement on what happens if user with Diamond level account spends at least $100”

Elseif memberLevel == "Gold" “Begins statement on what happens if member has Gold level account”

If totalSpend > 300.00 “Begins statement on what happens if user with Gold level account spends at least $300”

Elseif totalSpend > 100.00 “Begins statement on what happens if user with Gold level account spends at least $100”

Elseif memberLevel == "Blue" “Begins statement on what happens if member has Blue level account”

If totalSpend > 100.00 “Begins statement on what happens if user with Blue level account spends at least $100”

Else totalSpend < 100.00 “Begins statement on what happens if user spends less than a $100

**Assign**

Assign discountPercent = 0.12 “Assigns discount percent to 12%”

Assign discountPercent = 0.085 “Assigns discount percent to 8.5%”

Assign discountPercent = 0.06 “Assigns discount percent to 6%”

Assign discountPercent = 0 “Assigns discount percent to 0%”

**Display**

Display getEstimate("Diamond") “Outputs estimate for Diamond level”

Display getEstimate("Gold”) “Outputs estimate for Gold level”

Display getEstimate("Blue”) “Outputs estimate for Blue level”

**Flowchart**

Diagram, engineering drawing

Description automatically generated

**PSEUDOCODE**

BEGIN

Declare float getEstimate, pricePerUnit,

Declare int noUnits

Declare string memberLevel

Assign float priceWithoutTax = pricePerUnit \* noUnits

Assign float totalSpend = priceWithoutTax\*0.06 + priceWithoutTax

If memberLevel == "Diamond"

If totalSpend > 700.00

Assign discountPercent = 0.12

Elseif totalSpend > 300.0

Assign discountPercent = 0.085

Elseif totalSpend > 100.00

Assign discountPercent = 0.06

Elseif memberLevel == "Gold"

If totalSpend > 300.0

Assign discountPercent = 0.085

Elseif totalSpend > 100.00

Assign discountPercent = 0.06

Elseif memberLevel == "Blue"

If totalSpend > 100.00

Assign discountPercent = 0.06

Else totalSpend < 100.00

Assign discountPercent = 0

Assign float finalCost = totalSpend - totalSpend\*discountPercent

Return finalCost

Display getEstimate("Diamond", 20, 20)

Display getEstimate("Gold", 20, 20)

Display getEstimate("Blue", 20, 20)

Display getEstimate(" ", 20, 20)

Display getEstimate("Diamond", 20, 100)

Display getEstimate("Gold", 20, 100)

Display getEstimate("Blue", 20, 100)

Display getEstimate(" ", 20, 100)

END

**Test Cases**

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| SKU | Name | Items in Unit | Price Per Unit | Quantity Purchased | Total | Result | Account Type | Percent discount | Member Level Total | Store Credit |
| HF-342 | ½ inch bolt | 50 | 20.00 | 2 | $2120.00 | Pass | Diamond | 12% | $1865.60 | $1134.40 |
| LK-322 | ¼ inch nail | 25 | 5.75 | 1 | $152.38 | Pass | Blue | 6% | $143.75 | $2856.25 |
| HF-342 | ½ inch bolt | 50 | 20.00 | 3 | $3180.00 | Fail, inefficient funds |  | N/A | N/A |  |
| KF-231 | Hammer | 1 | 15.23 | 30 | $484.31 | Pass | Gold | 8.5% | $443.14 | $2556.86 |
| HF-342 | Hammer | 1 | 15.23 |  |  | Fail, product name and sku do not match |  | N/A | N/A |  |

**Code**

// declare variables

float getEstimate(string memberLevel, float pricePerUnit, int noUnits ) {

float priceWithoutTax = pricePerUnit \* noUnits; // assign price without tax equation

float totalSpend = priceWithoutTax\*0.06 + priceWithoutTax; // assign total spend equation

float discountPercent = 0;

if( memberLevel == "Diamond" ) {

if( totalSpend > 700.0 ) {

discountPercent = 0.12; // if user spends at least $700 they will get 12% discount

} else if( totalSpend > 300.0 ) {

discountPercent = 0.085; // if user spends at least $300 they will get 8.25% discount

} else if( totalSpend > 100.0 ) {

discountPercent = 0.06; // if user spends at least $100 they will get 6% discount

}

} else if( memberLevel == "Gold" ) {

if( totalSpend > 300.0 ) {

discountPercent = 0.085; // if user spends at least $300 they will get 8.25% discount

} else if( totalSpend > 100.0 ) {

discountPercent = 0.06; // if user spends at least $100 they will get 6% discount

}

} else if( memberLevel == "Blue" ) {

if( totalSpend > 100.0 ) {

discountPercent = 0.06; // if user spends at least $100 they will get 6% discount

}

} else { // if user doesn’t spend any of the above amounts then they get no discount

discountPercent = 0; // assign discount price to zero

}

float finalCost = totalSpend - totalSpend\*discountPercent; // assign final cost equation

return finalCost;

}

int main()

{

// display estimates for each member level

cout << getEstimate("Diamond", 20, 20) << endl;

cout << getEstimate("Gold", 20, 20) << endl;

cout << getEstimate("Blue", 20, 20) << endl ;

cout << getEstimate(" ", 20, 20) << endl ;

cout << getEstimate("Diamond", 20, 100) << endl;

cout << getEstimate("Gold", 20, 100) << endl;

cout << getEstimate("Blue", 20, 100) << endl ;

cout << getEstimate(" ", 20, 100) << endl ;

# Grade Sheet

*Fundamentals of Programming*

*Ms. Vanessa Coote*

*Before submitting the project package, the student should review each of the elements listed below and put a checkmark only in those checkboxes where the designated elements has been reviewed and meets specifications. After completing your document package, number your pages and write the designated page numbers onto the spaces provided on the grading sheet.*

**\_\_\_\_\_ Professionalism (10 points)**

* Following directions
* Neatly assembled 8 ½ by 11
* Cover page
* Page numbers
* Documentation

**\_\_\_\_\_ Source Program Listing and Proper Execution of Program (30 points)**

*It is expected that each student’s program will run correctly*

* Program source code listing matches code on submission and/or backups
* Inclusion of comment lines in source code
* Comments at the beginning of the program including programmer, project name and number, date written, and brief program description.
* Comments at key locations throughout the code
* Descriptive variable names (that follow naming convention)
* Logic is correct
* Logic is clear and easy to follow
* Proper formatting of statements
* Alignment, proper indentation, etc
* Proper use of data types and data conversions

**\_\_\_\_\_ Test Data (5 points)**

* Each test case properly calculated by hand and documented
* Suitable choice of you own test data case

**\_\_\_\_\_ Input Window (10 points)**

* Correct data type for each input section
* Analysis of data type (e.g. int, float, double etc.)
* Appropriate restrictions for each input section
* Data input value shown matches specified test data
* Appropriate display for each input section

**\_\_\_\_\_ Output (15 points)**

* Suitable layout of output (including required fields, easy to read layout, etc.)
* All data cases displayed
* Correct value displayed for each case
* Correct format of fields (e.g. use of integers and not float as appropriate, dollars and cents, display of $, etc)
* Required output format
* Aesthetics (User-friendliness, easy to understand output, alignments, etc)

**\_\_\_\_\_ Documentation (40 points)**

* Analysis of specifications
* Pseudocode
* Flowchart
* Hard copy of program

**\_\_\_\_\_ Fully Functioning Program (30 points)**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Possible points = 140

Points Earned =