Code Structure Measurement of SPL I

for

Shop Assistant

Submitted To

Dipok Chandra Das

Assistant Professor

Institute of Information Technology (IIT)

Noakhali Science and Technology University

Submitted by

Prosanto Deb (ASH1925005M)

Sultana Marjan (BKH1925010F)

Shahriar Ahmed (ASH1925015M)

Md. Alamgir Hossain (ASH1925016M)

Ratno Kumer Tripura (ASH1825042M)

Course Title: Software Metrics Lab

Course Code: SE 3204

Date of Submission: 05/03/2023

Table of Contents

Table of Figures

1. Project Information

Project Name	**Shop Assistant**
Supervised By	**Dipanita Saha** Assistant Professor Institute of Information Technology(IIT) Noakhali Science and Technology University
Team Members	**Prosanto Deb** (ASH1925005M)
	Sultana Marjan (BKH1925010F)
	Shahriar Ahmed (ASH1925015M)
	Md Alamgir Hossain (ASH1925016M)
	Ratno Kumer Tripura (ASH1825042M)
GitHub Link	Shop Assistant
Project Report	Shop Assistant Report

2. Definition

The control flow measures are usually modeled with directed graphs, where each node (or point) corresponds to a program statement or basic block (code that always executes sequentially), and each arc (or directed edge) indicates the flow of control from one statement or basic block to another. We call these directed graphs control flowgraphs or flowgraphs.

2.1 Structural Complexity Properties

1. **Nonnegativity**: System complexity cannot be negative.
2. **Null value**: The complexity of a system with no links is zero.
3. **Symmetry**: The complexity of a system does not depend on how links are represented.

4. **Module monotonicity**: System complexity "is no less than the sum of the complexities of any two of its modules with no relationships in common."
5. **Disjoint module additivity**: The complexity of a system of disjoint modules is the sum of the complexities of the modules.

2.2 Length Properties

1. **Nonnegativity**: System length cannot be negative.
2. **Null value**: A system with no links has zero length.
3. **Nonincreasing monotonicity for connected components**: Length does not increase when adding links between connected elements.
4. **Nondecreasing monotonicity for nonconnected components**: Length does not decrease when adding links between nonconnected elements.
5. **Disjoint modules**: The length of a system of disjoint modules is equal to the length of the module with the greatest length.

2.3 Coupling Properties

1. **Nonnegativity**: Module coupling cannot be negative.
2. **Null value**: A module without links to elements that are external to the module has zero coupling.
3. **Monotonicity**: Adding intermodule relationships does not decrease coupling.
4. **Merging modules**: Merging two modules creates a new module that has coupling that is at most the sum of the coupling of the two modules.
5. **Disjoint module additivity**: Merging disjoint modules without links between them creates a new module with coupling that is the sum of the coupling of the original modules.

2.4 Cohesion Properties

1. **Nonnegativity and normalization**: Module cohesion is normalized so that it is between zero and one.
2. **Null value**: A module whose elements have no links between them has zero cohesion.
3. **Monotonicity**: Adding links between elements in a module cannot decrease the cohesion of the module.
4. **Merging modules**: Merging two unrelated modules creates a new module with a maximum cohesion no greater than that of the original module with the greatest cohesion.

3. Process

The in-degree of a node is the number of arcs arriving at the node, and the out-degree is the number of arcs that leave the node.

3.1 Cyclomatic Complexity Measure

McCabe proposed the cyclomatic number of a program's flowgraph as a measure of program complexity. For a program with flowgraph F, the cyclomatic number is calculated as,

$$\nu(F) = e - n + 2$$

Cyclomatic complexity can also be calculated by the following equation,

$$\nu(F) = 1 + d$$

3.2 Tree Impurity

It defines how far a given graph deviates from being a tree. m(G) = 0 if and only if G is a tree.

$$m(G) = \frac{2(e - n + 1)}{(n - 1)(n - 2)}$$

4. Control Flow graphs

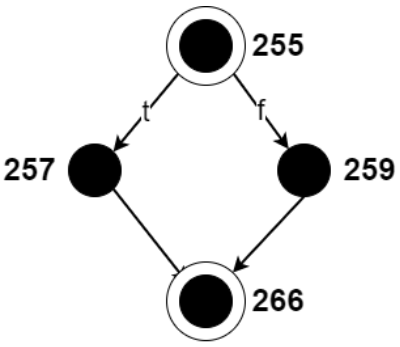

Figure: actionPerformed(ActionEvent e)

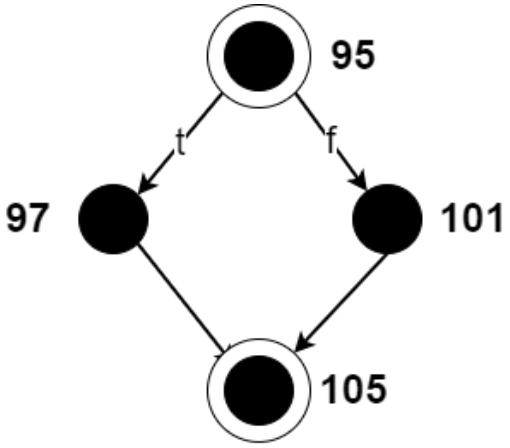

Figure: actionPerformed(ActionEvent e)

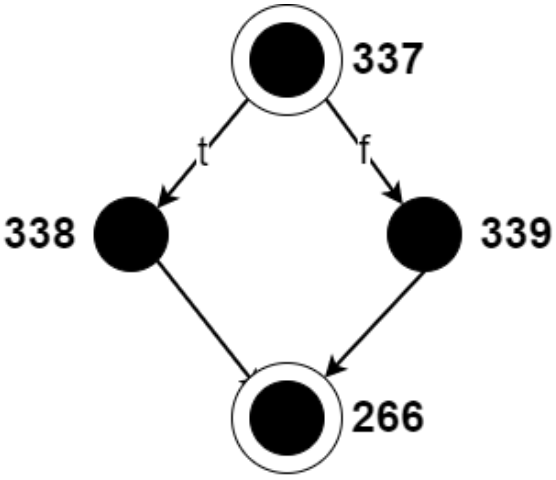

Figure: checkMoneyDigit(char c)

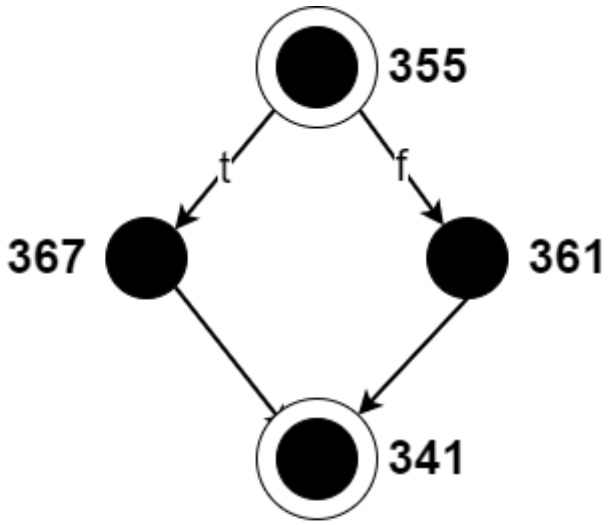

Figure: setDueCheckFeatures()

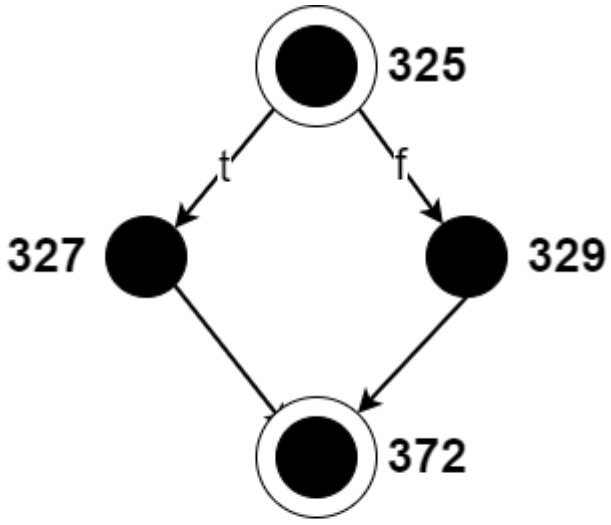

Figure: printButton()

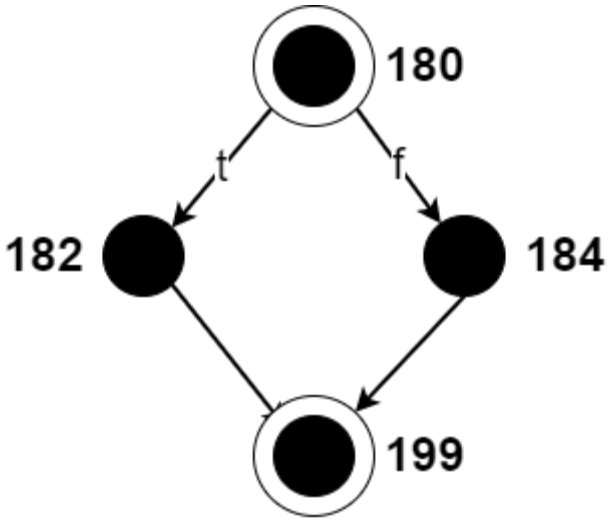

Figure: costSubmitButton()

Till now cyclometic complexity is 2 for the flow graphs described above. These are if and else scenarios.

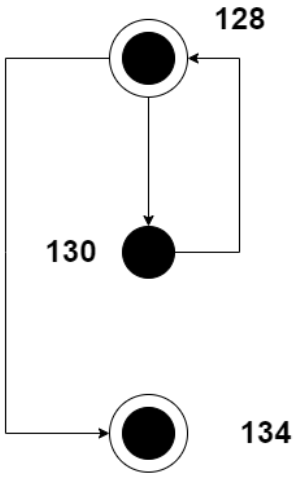

Figure: setStockCheckfeatures()

This loop contains a cyclometic complexity is 2 for a single condition.