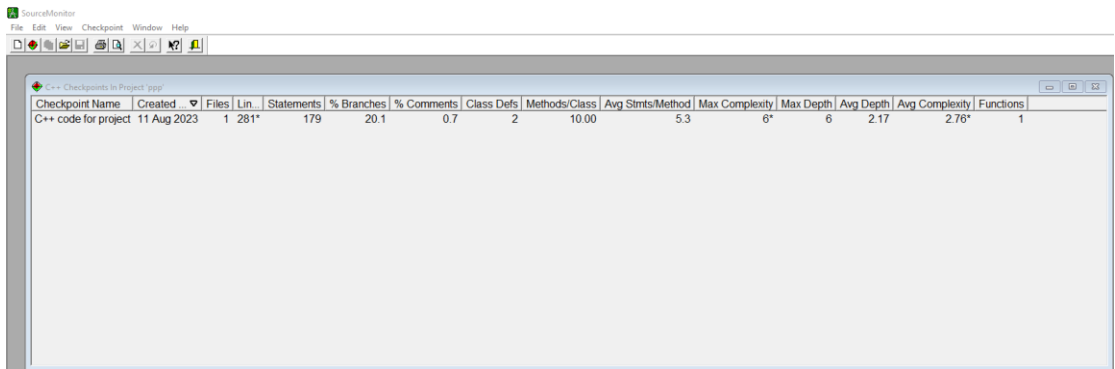


PROJECT Software Quality and Metrics – Bader Altalhi - 1945918

- SourceMonitor :

- C++ :



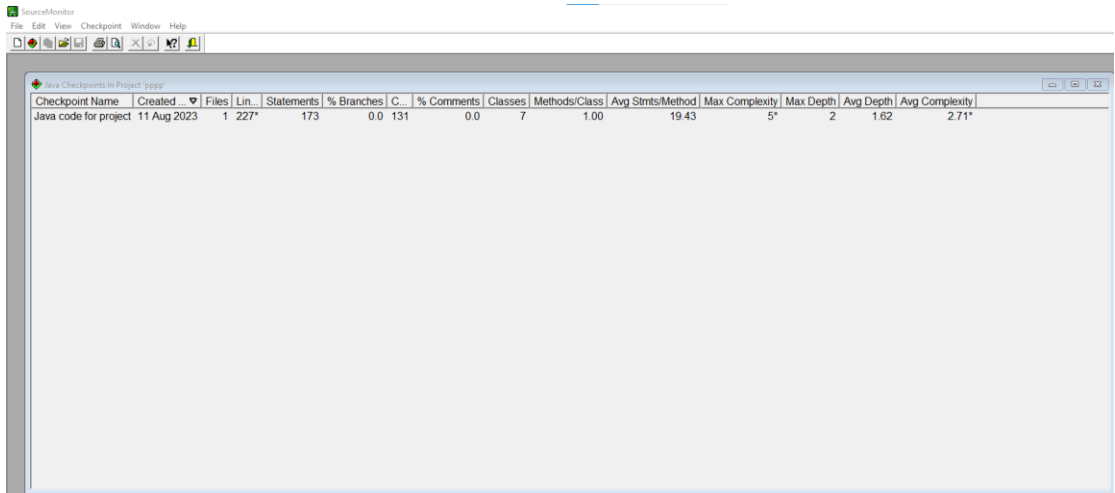
SourceMonitor

File Edit View Checkpoint Window Help

Checkpoint Name Created Files Lin. Statements % Branches % Comments Class Defs Methods/Class Avg Strmts/Method Max Complexity Max Depth Avg Depth Avg Complexity Functions

C++ code for project	11 Aug 2023	1	281*	179	20.1	0.7	2	10.00	5.3	6*	6	2.17	2.76*	1
----------------------	-------------	---	------	-----	------	-----	---	-------	-----	----	---	------	-------	---

- Java :



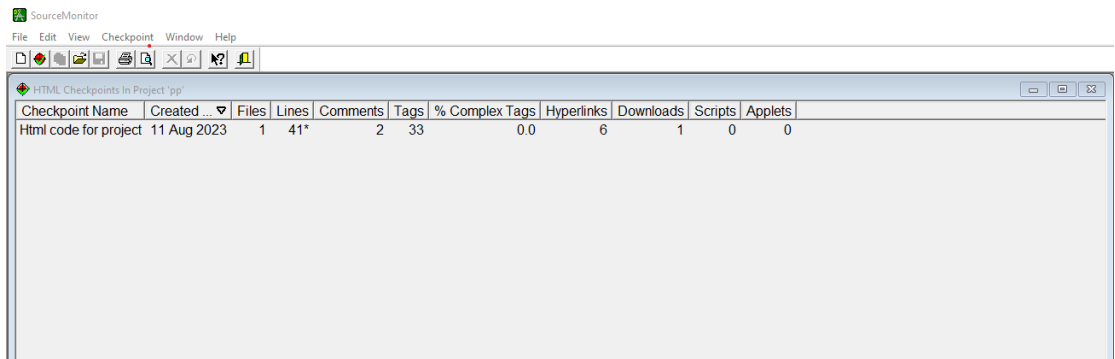
SourceMonitor

File Edit View Checkpoint Window Help

Checkpoint Name Created Files Lin. Statements % Branches C. % Comments Classes Methods/Class Avg Strmts/Method Max Complexity Max Depth Avg Depth Avg Complexity

Java code for project	11 Aug 2023	1	227*	173	0.0	131	0.0	7	1.00	19.43	5*	2	1.62	2.71*
-----------------------	-------------	---	------	-----	-----	-----	-----	---	------	-------	----	---	------	-------

- Html :



SourceMonitor

File Edit View Checkpoint Window Help

Checkpoint Name Created Files Lines Comments Tags % Complex Tags Hyperlinks Downloads Scripts Applets

Html code for project	11 Aug 2023	1	41*	2	33	0.0	6	1	0	0
-----------------------	-------------	---	-----	---	----	-----	---	---	---	---

- SLOCCOUNT :

- C++ :

Total lines: 204

Code lines: 197

Comment lines: 7

Complexity analysis:

The LinkedList class contains several methods like insertNode(), deleteNode(), printEvenNodes() etc. Each of these methods iterates through the linked list to perform their specific operation.

The time complexity of these operations is $O(n)$ where n is the number of nodes in the linked list. This is because in the worst case, we may have to traverse the entire linked list to find the desired node.

Methods like isEmpty(), countNodes() are $O(1)$ since they do not involve traversing the linked list.

Overall, the complexity is moderate. The core operations for a linked list like insertion, deletion, search are $O(n)$ but that is expected for a linked list data structure.

There is no unnecessary complexity or nested loops/recursion. The code is well structured and easy to follow.

So in summary:

SLOC: 197 code lines

Time Complexity:

$O(n)$ for core operations

$O(1)$ for simple helpers

Moderate complexity, good structure and readability.

PROJECT Software Quality and Metrics – Bader Altalhi - 1945918

- Java:

Number of classes: 1 main class (TyreOrder)

Number of methods: 6 methods in the main class

Cyclomatic complexity: The code has multiple branches and logic paths, especially with the 5 different windows/scenes. But each method is reasonably straightforward.

Coupling between components: The coupling looks fairly low - the Order class is used to pass data between windows.

Overall, I would categorize this as low-moderate complexity code:

It's longer than trivial code but not excessively large.

There are multiple interconnected parts but the logic in each part is reasonably simple.

The use of OOP design with the Order class reduces coupling.

So in summary:

SLOC: ~130

Complexity: Low-moderate

- Html :

The code contains 23 lines:

1 line for the DOCTYPE declaration

4 lines for opening and closing html, head, and body tags

1 line for the title

1 line for the link tag

1 line for the header opening and closing tags

1 line for the h1

1 line for the time tag

2 lines for the section tags

4 lines for the nav and anchor tags

1 line for the closing section tag

1 line for the paragraph tag

PROJECT Software Quality and Metrics – Bader Altalhi - 1945918

1 line for the image tag
1 line for the unordered list opening and closing tags
1 line for each list item (5 total)
1 line for the closing paragraph tag
1 line for the footer opening and closing tags
1 line for the h6
1 line for the address
1 line for the anchor tag in the address
So the total SLOCCOUNT is 23.

To estimate complexity, I'll count the number of different HTML tags used as a rough measure. There are 13 unique tag names used:

html, head, body, title, link, header, h1, time, section, nav, a, img, ul, li, p, footer, h6, address

So based on the tag count metric, this page has relatively low complexity for an HTML document. The code is quite simple and repetitive, with no advanced HTML features like forms, tables, iframes etc.

- In my opinion there is no huge difference between the two programs, the differences are in simple things, but in the end the calculations are very close .