**22W\_CST8116\_451 Intro to Computer Programming**

Assignment #2 Java Programming Structure: Flowchart, Pseudo code

**Carol Ann Wilson**

Student ID: 040672794

Instructor: Piyush Jangam

Date: January 25, 2022

# Assignment #2 Java Programming Structure: Flowchart, Pseudo code

## What steps must be followed when writing a complete algorithm?

When writing a complete algorithm, you must:

1. Determine the problem you’re trying to solve, the purpose of the algorithm.
2. Decide on a starting point. Where would you start to resolve this problem. For example, where would you start if you were trying to determine the value of x + y, when those values are input by a user.
3. Determine the end point of the algorithm. How do you want the situation to end? What would the last step be?
4. List the steps from start to finish. What are all the steps needed to get from your starting point to your end point? For example, if the goal is to calculate the value of x + y when those values are provided by a user, what steps do you need to accomplish this.
5. Determine how to perform each step. Once you’ve determined where you’re starting and where you’re ending and all the steps needed to get there, you need to determine how to perform each step in greater detail.
6. Review the algorithm. Once you’ve completely written your algorithm, you should go back and make sure you haven’t missed anything. Are there steps that need to be added or do you have redundant steps that you can remove. If you modify the order does that make it more efficient?

## Which symbols are used for Flowchart?

There are six basic symbols used for flowcharting:

1. Terminal
2. Process
3. Input/output
4. Decision
5. Connector
6. Predefined Process

Flow lines may not be considered one of the six basic symbols; they are, however, required to show the direction to follow the process in and should always have an arrow that points you in the correct direction.

### Draw and explain each symbol.

|  |  |
| --- | --- |
|  | **Terminal**  This is the starting or ending point of the program, process, or an interrupt program. |
|  | **Process**  Processes are a type of internal operation that takes place inside the processor or memory of the system. |
|  | **Input/output**  Any input or output point required in the algorithm. Examples would include a user entering data or a printout of data. |
|  | **Decision**  A decision point is used to ask a binary format such as a yes/no or true/false style question. |
|  | **Connector**  Connectors are used to draw a flowchart without interesting lines or without requiring a reverse flow. It’s generally used to reduce the amount of confusion that could be caused if you continually overlap or “go back” in a diagram. |
|  | **Predefined Process**  The predefined process symbol is used to invoke subroutines or create an interrupt in a process. I might use it in Java programming to call a pre-created method, for example. |
|  | **Flow Lines**  Shows the direction to follow in the process flow |

## What is Pseudocode?

Pseudocode is the writing of an algorithm in what is, effectively, plain text. It is more readable for the average person. However, pseudocode doesn’t require programming language syntax so it cannot be compiled.

Advantages of includes being able to act as a basis for the documentation of your code, it’s easily readable, and provides a more detailed explanation of what each line of the program will do, making it easier to write the actual code.

To make things even easier on the coder, pseudocode usually uses naming conventions, sentence casing, and if/then and for/while structures of the language you’re planning on writing in. For example, using CamelCase for methods, etc.

## List the differences between flowchart and algorithms.

|  |  |
| --- | --- |
| **Algorithm** | **Flowchart** |
| Step by step instructions to solve a specific problem or achieve a specific goal | Diagram that shows the algorithm in a standard flow chart format |
| Harder to understand at a quick glance. An algorithm requires the observer to read each line to make sure they understand exactly what is being achieved. | Easier to grasp at a quick glance, pictures are often easier to follow that text and the symbols provide quick guidance as to function since they are standardized. |
| Harder to create, since detailed written information is required | Much easier to construct, since the symbols used fill in much of the needed knowledge such as whether it’s an input, a decision, etc. |
| Easier to debug errors. The increased level of written detail makes it easier to figure out where things might (or have) gone wrong. | Harder to debug. Pictorial representations can create assumptions that aren’t necessarily accurate when trying to figure out if there’s something missing or something wrong. |
| More difficult to express complex branching/looping in a written algorithm | Flowcharts are easier to show complex branching and looping, since it’s simple to follow the directions of arrows and pathways |
| An algorithm can be more easily converted to pseudocode. You’ve already written the directions in long text, converting it to the more streamlined pseudocode is simpler. | Since a flowchart is only a graphical representation with a minimal amount of text, it is not as useful for converting to pseudocode |

## References

### Websites

MKS075. (2020, May 21). *Difference between algorithm and Flowchart*. GeeksforGeeks. Retrieved January 25, 2022, from <https://www.geeksforgeeks.org/difference-between-algorithm-and-flowchart/>

*Difference Between Algorithm And Flowchart - Explained!* Dare2Compete. (2021, September 29). Retrieved January 25, 2022, from <https://dare2compete.com/blog/difference-between-algorithm-and-flowchart>