Don’t Skip The Pseudocode

Benefits of using pseudocode to write computer programs

Felix Cabrera Follow
Sep 2 · 4 min read ★

Photo by Danial RiCaRoS on Unsplash

Stop! Don’t skip writing pseudocode until you read this …

When I started to learn how to code, I began to learn to write pseudocode. At first, I began to think that it was a tedious process, and I started to feel impatient and wanted to get to writing the code right away.

As I started to ignore writing pseudocode, I began to notice that I would not have a clear plan of the computer program I had to write. So, I would invest a lot of time struggling to write the code, but not achieving the desired outcome. Hence, appreciating the value of writing the pseudocode and then implementing the solution using a programming language.

So, today, I will define what is pseudocode and share my experience of how I learned that it is an essential practice to implement when writing code. Also, I will share some things I learned through my journey, which you can use too in your coding journey.

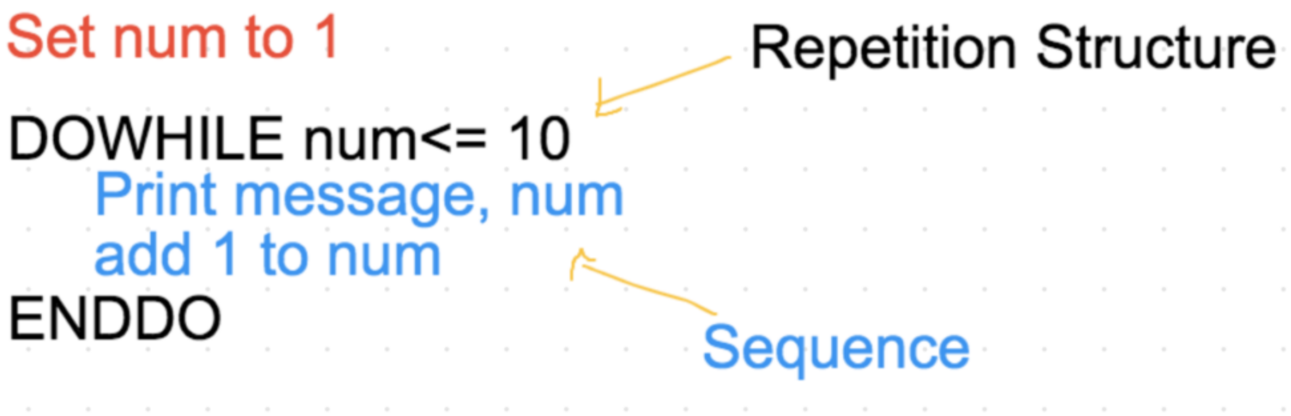
Thus, without further ado, let’s get started.

. . .

What Is Pseudocode?

Pseudocode is a detailed written description of what your code does. It does not have to use any syntax related to any programming language. Just words of the English language to convey the message of what your computer program’s function is.

Here’s a simple example:



```
Set num to 1
DOWHILE num<= 10
    Print message, num
    add 1 to num
ENDDO
```

On the first line, Set num to 1, the variable num is initialized to 1. On the second line, DOWHILE num<=10, it conveys the concept of a **repetition structure**. Moreover, it denotes the idea of a do-while loop, just like in programming languages such as Java and C++.

The **sequence of statements** in blue, the print message, num and add 1 to num, are going to be repeated until the condition is false. Again, just like we use a do while in a programming language, however, we describe the lines using the English language.

. . .

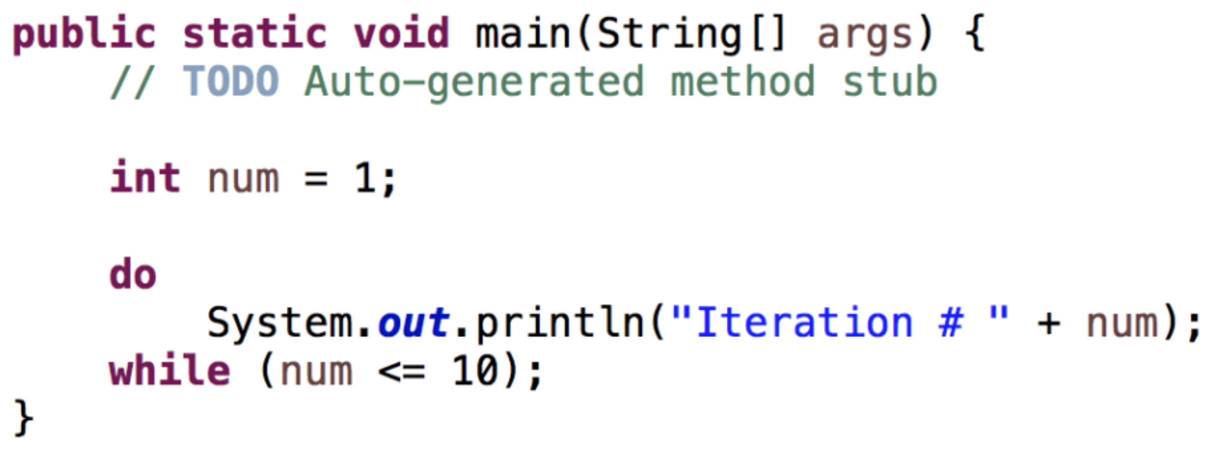
Why Use Pseudocode?

When I was learning how to code, I used to jump right to write the code. I did not organize my thoughts and not establish a plan to write the solution to a coding problem. I noticed that **I would have bugs in my code** and invest a lot of time to find, also, fix the error in my code. Furthermore, I realized that these errors **were in the logic** and **reasoning behind the approach** I used to solve the coding problem.

However, after I started to take the time and write the pseudocode, I **would save time** in fixing bugs from my code. Also, as I repeatedly used the approach of writing the pseudocode before writing my code, I got better at using this skill. So, the process was more straightforward as time went by.

. . .

You Can Use It As A Blueprint



```
public static void main(String[] args) {
    // TODO Auto-generated method stub

    int num = 1;

    do
        System.out.println("Iteration # " + num);
    while (num <= 10);
}
```

Through the years, I see pseudocode as a **blueprint to a solution to a coding problem** or an algorithm. Moreover, I have come to view it like a map you can use to write your code. Also, to **communicate your solution** with others despite the programming language, you may decide to use in the future. You can use pseudocode to express your thoughts behind your solution even with people who don’t know much about coding.

Another benefit of using pseudocode as a blueprint is that **you can use the same pseudocode** to translate your solution to different programming languages.

For instance, using our previous example, here’s how the solution would look like in Java. We have the initialization of the num variable; the do-while loop and the message displayed alongside the value of the num variable.

I understand that this is a simple solution; however, you can use the same approach to write more complex algorithms.

Final Thoughts

At the start of my coding journey, I just wanted to write the code. I wanted to skip writing the pseudocode because I would get impatient and translate my thoughts into code directly without proper organization. Nonetheless, after a lot of hassle, I started to appreciate using pseudocode. It can help you organize your thoughts, then, translate them into code.

In my journey, after repeatedly using pseudocode, it began to get more comfortable for me. Which can also happen to you.

There you have it, things I have learned when using pseudocode, so you can use these insights in your coding journey.

I hope this helps!

If you liked this article, I think you might be interested in these articles as well:

What Should I Add to My Resume Without Experience?

Find creative ways to stand out and create your opportunities

medium.com

How Can I Learn To Code At Home Without Losing Motivation?

Losing motivation when learning to code at home by yourself? Read this …

medium.com

The Most Valuable Skill Every Software Developer Should Have

There’s one skill that trumps the rest …

medium.com

***Disclaimer:** Results may vary. These tips and advice are based on my experience and opinion as a former undergraduate Computer Science student, tutor, teacher, and software developer. Everyone is different, so, the advice shared in this article may or may not work for you.*

. . .

Originally published at https://yadielcabrera.com.

Programming Self Improvement Life Lessons Productivity Education