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

Assignment #3 Java Structure: Algorithm

**Carol Ann Wilson**

Student ID: 040672794

Instructor: Piyush Jangam

Date: February 10, 2022

# Assignment #3 Java Structure: Algorithm

## Describe the following, in your own words:

### What is an algorithm?

An algorithm is a written process or procedure used to define the steps needed to resolve a problem. In the case of computer programming the algorithm is a step-by-step description, based on the problem, of the steps required to reach the resolution of the problem.

### Why do we use algorithms?

Being able to clearly define the problem and all of the steps or actions required to achieve the desired outcome is the best way to start out the process of developing code. If you don’t have any idea of the goal or how to get there (for example, finding the sum of two numbers) then you will be unlikely to end up with what you actually need or want.

## List and briefly explain all testing techniques.

1. Unit testing

Usually written by the developer, unit tests are done against smaller pieces of code to make sure a method, a loop, an equation, etc., is correct and functional. Usually done with sample data. For example, I might test that my formula for the area of a circle is accurate.

1. Integration Testing

Integration tests are used to make sure that the individual units from the Unit testing will now work together. Even if every unit test passes, when you merge all the different pieces unexpected things can, and usually do, happen. Integration tests help root these out and make for better software.

1. Regression Testing

Regression testing is used to make sure that old functionality in software still works correctly when new functionality is added as well as making sure that the new functionality works as expected.

1. Smoke Testing

Smoke testing is usually used to make sure that the software is ready for future testing. Basically, if nothing blows up when you build the software, then you can pass it on to the next rounds of testing. Unstable software is not worth doing system, stress, or performance tests on because you won’t know if the problem is the instability, or the actual thing being tested.

1. Alpha Testing

Alpha testing is a type of acceptance testing done before you release software to end-users or customers. Often done by QA staff, it validates functionality. Some large companies will use their employees as second stage alpha testers, especially when the employee might have a company issued device.

1. Beta Testing

Beta testing is usually done by a selected group of customers. In some cases, it might be a customer who asked for a specific feature to be developed. Beta testing lets software be used in a real-time environment with the understanding that there may still be issues and bugs that show up under more functional usage.

1. System Testing

System testing verifies that software can be deployed and works on various operating systems. If you release software that is meant to run on Windows, Linux, and macOS, but never actually test it on all three system types, you’re almost guaranteed to have one or more of them fail at the most basic install level on up.

When doing system testing you will also quite likely perform stress and performance tests and do security tests at the same time.

1. Stress Testing

Stress testing is used to determine what happens with your software when poor system conditions occur. Examples could include low disk space, too little memory or memory that is at full usage, or high/full CPU usage.

1. Performance Testing

Performance testing determines the speed and effectiveness of software under higher loads. Many software users in corporate environments need to know how many users can be on a system, how many functions can process simultaneously, and so on to plan for growth and redundancy.

1. Object-Oriented Testing

Object-oriented testing is the process of using various testing techniques to validate object-oriented software. This includes:

* Testing of Requirements,
* Design and Analysis of Testing,
* Testing of Code,
* Integration testing,
* System testing,
* User Testing.

### Which technique is the most efficient?

I honestly couldn’t answer this question. In my experience the most efficient testing is the tests that catch problems before they go too far into the code. So I would think that Unit and Integration tests would be the most efficient for minimizing bugs, but you can’t really do without the others.

## Write an algorithm to find the area of a Circle of radius r.

A = πr2

pi = 3.14, double r, double area (might contain decimals so requires a double not an int)

|  |  |
| --- | --- |
| **Step** |  |
| 1 | **Start** |
| 2 | Declarations: double r (radius of the circle), double area (calculated area of the circle), double pi =3.14 |
| 3 | Print to output: “Please enter the radius of your circle” |
| 4 | Take input of radius to *r* |
| 5 | area = pi\*r\*r |
| 6 | Print to output “The area of a circle with the radius of “ + *I on* + “is “ + *area* |
| 7 | **End** |

## Write an algorithm to read two numbers and find their sum.

num1 + num2 = sum, int num1, int num2, int sum

|  |  |
| --- | --- |
| **Step** |  |
| 1 | **Start** |
| 2 | Declarations: int num1, int num2, int sum |
| 3 | print “Please enter number 1” |
| 4 | Take input from user and store in *num1* |
| 5 | print “Please enter number 2” |
| 6 | Take input from user and store in *num2* |
| 7 | sum = num1 + num2 |
| 8 | print “The sum of your two numbers is “ + sum |
| 9 | **End** |

## Write an algorithm to convert temperature from Fahrenheit to Celsius.

(F – 32) \* 5/9, 5/9 = .556, double fahrenheit, double celsius

|  |  |
| --- | --- |
| **Step** |  |
| 1 | **Start** |
| 2 | Declarations: double fahrenheit, double celsius |
| 3 | print “Please enter your temperature in Fahrenheit” |
| 4 | Take input from user and store in fahrenheit |
| 5 | celsius = (fahrenheit -32) \* .556 |
| 6 | print “Your temperature in Celsius is “ + celsius |
| 7 | **End** |

## Write an algorithm to find the greater number between two numbers.

Int num1, int num2

|  |  |
| --- | --- |
| **Step** |  |
| 1 | **Start** |
| 2 | Declarations: int num1, intnum2 |
| 3 | if num1 > num2 |
| 4 | then print num1 + “is greater” |
| 5 | else print num2 + “is greater” |
| 6 | **End** |

## Write an algorithm to find the largest value of any three numbers.

Int num1, int num2, int num3

|  |  |
| --- | --- |
| **Step** |  |
| 1 | **Start** |
| 2 | Declarations: int num1, int num2, int num3 |
| 3 | if num1 > num2 |
| 4 | then if num1 > num3, |
| 5 | print num1 + “is largest” |
| 6 | else |
| 7 | if num2 > num3 |
| 8 | print num2 + “is largest” |
| 9 | else print num3 + “is largest” |
| 10 | **End** |

## References

*Types of software testing*. GeeksforGeeks. (2020, December 23). Retrieved February 10, 2022, from <https://www.geeksforgeeks.org/types-software-testing/>