Course outcomes for CSE 1310:

After having taken this course, students should be able to perform the following specific tasks, related to reading code and writing code:

***Part 1: Reading code***

Students should be able to read a program written in Java (that uses constructs covered in this class) and:

* understand how a program is executed line-by-line, from beginning to end.
* be able to specify the value of any variable at any point in the execution.
* be able to specify the output of the program.

Specific topics and examples for reading code:

**Topic 1.1: Code manipulating booleans.**

**Topic 1.2: Code using conditionals**

* be able to follow the program execution for specific inputs (taking them on a specific execution path)
* be able to identify all paths of execution in a program and the data/inputs needed to take each path. – Testing

**Topic 1.3: Code using for and while loops.**

**Topic 1.4: Code using multiple function calls, possibly including recursive calls.**

**Topic 1.5: Code manipulating array pointers.**

More specifically, students should be able to identify situations when array and array list objects point to the same location in memory, so that changing the contents in one of those objects automatically changes the contents in the other objects pointing to the same memory. Example: Given a function foo that has arrays a and c as input, students should be able to understand how the values of arrays a and c are affected by function foo.

**Topic 1.6: Variable scope and namespaces.**

More specifically, students should be able to identify, for any variable, the lines of code that constitute the scope of that variable. For function calls, students should understand that variables declared in different functions and in main are distinct, even if they have the same name.

E.g. 1: be able to recognize if a program uses a variable out of scope.

E.g. 2: be able to simulate by hand the execution of function calls (including the passing of arguments (position-wise correspondence of argument to parameter and NOT name-wise), the function execution and the return of data and control back to the caller).

***Part 2: Writing code***

Students should be able to write Java code that solves various problems involving conditionals, loops, functions, arrays, and files.

Specific topics and examples for writing code:

**Topic 2.1: Code performing simple numerical operations to compute a quantity of interest.**

**Topic 2.2: Code using conditionals.**

E.g. read a word and print if it starts with a vowel, consonant or none of these.

**Topic 2.3: Code using loops to count or accumulate values over elements of a set.**

E.g. a count\_vowels, sum\_squares function

**Topic 2.4: Code using loops to solve problems where multiple values must be searched to see if any of them satisfies a certain criterion.**

E.g. Given an array, returns true if all the numbers in it are even.

**Topic 2.5: Code using nested loops explicitly or implicitly (e.g., by writing a function that uses a for-loop, and calls at each iteration another function that also uses a for-loop).**

Example problem 1: read a number, N, and print a triangle of stars.

Example problem 2: given a string and a number, N, print each letter N times.

**Topic 2.6: Write functions that compute specific arithmetic quantities.**

Example problem 1: compute volume of a sphere given the radius.

**Topic 2.7: Write functions that manipulate strings to produce other strings.**

Example problem 1: given a string, text, build and return a string that is obtained by removing all vowels from text.

Example problem 1: print\_coded function, (e.g. Caesar’s code)

**Topic 2.8: Write multiple functions that are used together to solve a problem.**

**Topic 2.9: Write code that can loop through arrays to identify one or more elements of interest.**

E.g. 1: return the index of the shortest string from an array list of strings.

E.g. 2: Given an integer array, return a new array, with elements that <=100.

**Topic 2.10: Write code that can use and manipulate 2-dimensional arrays.**

Example problem 1: given two 2-D arrays, produce a 3-rd 2D array.

Example problem 2: given a 2D array, iterate over all elements and do something.

**Topic 2.11: Write functions that can process file data.**

Example problem 1: Write a function that can load the content of a text file into an array of strings (one string for each line).

Example problem 2: Write a function that can load the content of a CSV file into a 2D array of strings (one string for each value in the CSV file).

Example problem 2: reverse\_lines(String input\_name, String output\_name)

**Topic 2.12: Write programs of about 150 lines of code, processing data and producing results according to specifications provided in English.**

**Topic 2.13: Write a recursive function.**

***Part 3: Problem solving.***

Students should be able to take the following steps in solving a problem:

* understand it to the point where they can solve it ‘on paper’ (i.e. for different inputs, can produce the output),
* identify the algorithmic steps needed to solve the problem (i.e. the specific repetitive process that can be implemented as a program),
* understand what data is being used and how to represent it in the program,
* write the code,
* test their code.

***Part 4: Miscellaneous topics.***

**Topic 4.1: Binary numbers**

Students should be able to convert, on paper, between positive binary and decimal integers. Students should also be able to write Java code that does such conversions automatically.