Programming Project 1: Kirchoff Current Law Solver

EAS240 Introduction to Programming using C/C++
Fall 2020 Dr. Hammond

Your goal is to write a program that solves a basic electrical circuit using Kirchoff's current law. We will limit our circuits to include only resistors and current sources.

This project will be broken down into two sections: (1) implementing a Gauss-Jordan elimination solver and (2) using the solver to calculate the node voltages of a basic electric circuit. Each section will have multiple checkpoints where you can ensure that you are getting the right solution via auto grading in your textbook.

Required files:
- EAS240_PP1_functions.h – contains all of your structs and function declarations
- EAS240_PP1_functions.c – contains all of your function definitions
- EAS240_PP1_main.c – contains the remaining required code
- EAS240_PP1_finalReport.pdf – contains a write up of your programming process

Part A: Gauss-Jordan elimination solver

Checkpoint A1: Write a program that solves the specific problem outlined in Appendix A. In your program, access the elements in your matrix as often as possible paying attention to the provided variables. Your program should print out the final solution in the following manner:

```
V1 = 0.7000 V
V2 = -2.8000 V
V3 = -0.7000 V
V4 = -2.3000 V
```

Checkpoint A2: Convert your program from Checkpoint A1 to work for any 4 x 5 array by reading in an array row by row from the terminal/user.

Checkpoint A3: Convert your program from Checkpoint A2 to work for any size array by reading in the number of rows from the user, constraining the number of columns to be the number of rows plus 1, using dynamic memory allocation, and filling the array with random integers between -5 and 5. Read in the seed from the user.

Checkpoint A4: Convert your program from Checkpoint A3 so that it performs Gauss-Jordan elimination in a function. Your function must pass in the array as a pointer, the number of rows, and the number of columns. All calculations should be performed within the array so the function should return void.

```
void PerformGaussElimination( double * arr, int ROWS, int COLS );
```

NOTE: It might be useful to create a function that also prints out the elements of a 2D array.

Part A Due date: Friday, October 23rd
- *Submission process: For each checkpoint, submit your code to the corresponding zyLabs in Chapter 12. Nothing is due on UBlearns.*
- *Your submission will be graded based on the rubric in Appendix D and your code submitted on zyBooks.*

Part B: KCL Solver

Struct: `Branch`
1. `type` – stores a 1 if the branch contains a current source and stores a 0 if the branch contains a resistor
2. `value` – stores the value of the current source or resistor element
3. `startNode` – stores the node the branch starts at, defined as the smaller, non-ground node
4. `endNode` – store the node the branch end as, defined as the larger node or ground

Checkpoint B1: Write a program that implements the `Branch` structure with 4 members (described above) and stores the circuit from Appendix B in an array of branches. Each element can be stored as described in the ”Circuit in a text file” description. Write out a function that prints out the circuit in a function where the node voltage is printed out first followed by the elements that start at that node. All values are printed out with a width of 10. An example is seen in Appendix B: Example output.

```
void PrintCircuit( const Branch * circuit, double * voltages, int
numBranches, int numNodes );
```

- `circuit` – an array of branches containing the circuit
- `voltages` – an array of voltages (you can manually enter in the solution below)
- `numBranches` – the number of branches
- `numNodes` – the number of nodes

Checkpoint B2: Add to the program from Checkpoint B1 to solve the circuit from Appendix B by using the branch values to fill in the matrix. Call your Gauss-Jordan solver function (`PerformGaussElimination`) from Part A4 to solve the system. In your program, access the elements in your circuit array as often as possible paying attention to the provided variables. Store the solution in your voltages array.

Checkpoint B3: Convert your program from Checkpoint B2 so that it can work for a generic circuit by reading in the circuit from the terminal/user. You should prompt the user for the number of branches, the number of nodes, and each branch element reading in all 4 values in one line. You will need to use dynamic memory allocation to create your circuit array.

Part B Checkpoint B3 Due date: Friday, October 30th

- *Submission process: For each checkpoint, submit your code to the corresponding zyLabs in Chapter 12. Nothing is due on UBlearns.*
- *Your submission will be graded based on the rubric in Appendix D and your code submitted on zyBooks.*

Checkpoint B4: Convert your program from Checkpoint B3 to a function that solves a generic circuit. Your function must pass in the circuit array of elements, an array to store the voltage solution, the number of branches and the number of nodes. Your function will “return” the solution in the voltages array so the return type is void.

```
void SolveCircuit( const Branch * circuit, double * voltages, int
                       numBranches, int numNodes );
```

Checkpoint B5: Convert your program from Checkpoint B4 so that it reads in a circuit from a file and prompts the user for the filename. The file is formatted as described in Appendix B. Convert your PrintCircuit function so that it prints the results to a file by appending the results to the input file containing the circuit.

```
void PrintCircuit( char filename[], const Branch * circuit,
                       double * voltages, int numBranches, int numNodes );
```

Part B Due date: Friday, November 6th

- *Submission process: For each remaining checkpoint, submit your code to the corresponding zyLabs in Chapter 12. Nothing is due on UBlearns*
- *Your submission will be graded based on the rubric in Appendix D and your code submitted on zyBooks.*
- *NOTE: This is due at the same time as your final submission*

NOTE: Checkpoint B5 should result in your final submission with the following components:

1. **EAS240_PP1_functions.h** – contains your Branch struct implementation and 3 function declarations (`PerformGaussElimination`, `PrintCircuit`, `SolveCircuit`). You may include any other functions in this file as well that you used to aid in your programming process.
2. **EAS240_PP1_functions.c** – contains your function definitions.

3. **EAS240_PP1_main.c** – prompts the user for a file containing a circuit, inputs the circuit into an array, creates an array to store the node voltages, solve the circuit by calling your `SolveCircuit` function, and prints the results to end of the same file by calling your `PrintCircuit` function. Lastly, do not forget to free any dynamically allocated memory and close out any opened files.

Part C: Final Report

You must submit a final report containing the following components:

1. A title page stating your name, the class and semester, and the name of the project
2. A description of your code for part A and part B in plain English. You may include any self-written pseudocode or flowcharts; however, no C code should appear in this document.
3. Your results in a tabular format (accompanied by a description) showing the output of your code for the following test cases:
    a. Part A

Number of Rows	Seed
2	48237
4	8207
7	58486
10	2154614
22	1264

    b. Part B: provided files circuit1.txt, circuit2.txt, and circuit3.txt
4. Your reflections containing the comments on the following:
    a. How you developed/approached writing your code.
    b. The good programming practices that you used while writing your code and how they improved your coding process.
    c. The most challenging aspects of writing your code.
    d. The least challenging aspects of writing your code.
    e. How you would improve your code given more time.

Final submission Due date: Friday, November 6th:

- *Final Submission process: zip your three required files into a zip file and upload (with your final report) to UBlearns*
- *Your submission will be graded based on the rubric in Appendix D and the files submitted to UBlearns*

Appendix A: Gauss-Jordan Elimination example

Given a system of equations written in matrix format, perform row operations until a matrix is in reduced row-echelon form.

Definitions:

System of equations:

$$\begin{gathered} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = b_m \end{gathered}$$

Matrix form:

$$\left[\begin{array}{cccc|c} a_{11} & a_{12} & \cdots & a_{1n} & b_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} & b_m \end{array}\right]$$

Row operations:

1. Replace an equation/row by adding it to any other equation/row (elimination step), i.e. $R2 = R2 - (A)R1$ where row 2 is replaced by taking row 2 and subtracting row 1 times A.
2. Multiply an equation/row by a nonzero constant

Reduced row-echelon form:

$$\left[\begin{array}{cccc|c} 1 & 0 & \cdots & 0 & x_1 \\ 0 & 1 & \cdots & 0 & x_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ 0 & 0 & \cdots & 1 & x_n \end{array}\right]$$

Example:

The matrix, $\left[\begin{array}{cccc|c} -4 & 5 & 1 & -5 & -6 \\ -3 & -3 & 3 & 4 & -5 \\ -5 & 4 & 4 & -5 & -6 \\ -3 & 1 & 6 & -7 & 7 \end{array}\right]$, corresponds to this system of equations, $\begin{gathered} -4x_1 + 5x_2 + x_3 - 5x_4 = -6 \\ -3x_1 - 3x_2 + 3x_3 + 4x_4 = -5 \\ -5x_1 + 4x_2 + 4x_3 - 5x_4 = -6 \\ -3x_1 + x_2 + 6x_3 - 7x_4 = 7 \end{gathered}$.

It is reduced in the following manner:

1. Perform row operations to make the **first** column zeros (except element a_{11}).
   a. $R2 = R2 - \left(\frac{a_{21}}{a_{11}}\right)R1 \rightarrow R2 = R2 - \left(\frac{-3}{-4}\right)R1$
   b. $R3 = R3 - \left(\frac{a_{31}}{a_{11}}\right)R1 \rightarrow R3 = R3 - \left(\frac{-5}{-4}\right)R1$
   c. $R4 = R4 - \left(\frac{a_{41}}{a_{11}}\right)R1 \rightarrow R4 = R4 - \left(\frac{-3}{-4}\right)R1$

$$\left[\begin{array}{cccc|c} -4 & 5 & 1 & -5 & -6 \\ -3-\left(\frac{-3}{-4}\right)(-4) & -3-\left(\frac{-3}{-4}\right)(5) & 3-\left(\frac{-3}{-4}\right)(1) & 4-\left(\frac{-3}{-4}\right)(-5) & -5-\left(\frac{-3}{-4}\right)(-6) \\ -5-\left(\frac{-5}{-4}\right)(-4) & 4-\left(\frac{-5}{-4}\right)(5) & 4-\left(\frac{-5}{-4}\right)(1) & -5-\left(\frac{-5}{-4}\right)(-5) & -6-\left(\frac{-5}{-4}\right)(-6) \\ -3-\left(\frac{-3}{-4}\right)(-4) & 1-\left(\frac{-3}{-4}\right)(5) & 6-\left(\frac{-3}{-4}\right)(1) & -7-\left(\frac{-3}{-4}\right)(-5) & 7-\left(\frac{-3}{-4}\right)(-6) \end{array}\right] = \left[\begin{array}{cccc|c} -4 & 5 & 1 & -5 & -6 \\ 0 & -6.75 & 2.25 & 7.75 & -0.5 \\ 0 & -2.25 & 2.75 & 1.25 & 1.5 \\ 0 & -2.75 & 5.25 & -3.25 & 11.5 \end{array}\right]$$

2. Perform row operations to make the **second** column zeros (except element a_{22}).
   a. $R1 = R1 - \left(\frac{a_{12}}{a_{22}}\right)R2 \rightarrow R1 = R1 - \left(\frac{5}{-6.75}\right)R2$

b. $R3 = R3 - \left(\frac{a_{32}}{a_{22}}\right) R2 \rightarrow R3 = R3 - \left(\frac{-2.25}{-6.75}\right) R2$

c. $R4 = R4 - \left(\frac{a_{42}}{a_{22}}\right) R2 \rightarrow R4 = R4 - \left(\frac{-2.75}{-6.75}\right) R2$

$$\left[\begin{array}{cccc|c} -4-\left(\frac{5}{-6.75}\right)(0) & 5-\left(\frac{5}{-6.75}\right)(-6.75) & 1-\left(\frac{5}{-6.75}\right)(2.25) & -5-\left(\frac{5}{-6.75}\right)(7.75) & -6-\left(\frac{5}{-6.75}\right)(-0.5) \\ 0 & -6.75 & 2.25 & 7.75 & -0.5 \\ 0-\left(\frac{-2.25}{-6.75}\right)(0) & -2.25-\left(\frac{-2.25}{-6.75}\right)(-6.75) & 2.75-\left(\frac{-2.25}{-6.75}\right)(2.25) & 1.25-\left(\frac{-2.25}{-6.75}\right)(7.75) & 1.5-\left(\frac{-2.25}{-6.75}\right)(-0.5) \\ 0-\left(\frac{-2.75}{-6.75}\right)(0) & -2.75-\left(\frac{-2.75}{-6.75}\right)(-6.75) & 5.25-\left(\frac{-2.75}{-6.75}\right)(2.25) & -3.25-\left(\frac{-2.75}{-6.75}\right)(7.75) & 11.5-\left(\frac{-2.75}{-6.75}\right)(-0.5) \end{array}\right] = \left[\begin{array}{cccc|c} -4 & 0 & 2.67 & 0.74 & -6.37 \\ 0 & -6.75 & 2.25 & 7.75 & -0.5 \\ 0 & 0 & 2 & -1.33 & 1.67 \\ 0 & 0 & 4.33 & -6.41 & 11.7 \end{array}\right]$$

3. Perform row operations to make the **third** column zeros (except element a_{33}).

a. $R1 = R1 - \left(\frac{a_{13}}{a_{33}}\right) R3 \rightarrow R1 = R1 - \left(\frac{2.67}{2}\right) R3$

b. $R2 = R2 - \left(\frac{a_{23}}{a_{33}}\right) R3 \rightarrow R2 = R2 - \left(\frac{2.25}{2}\right) R3$

c. $R4 = R4 - \left(\frac{a_{43}}{a_{33}}\right) R3 \rightarrow R4 = R4 - \left(\frac{4.33}{2}\right) R3$

$$\left[\begin{array}{cccc|c} -4-\left(\frac{2.67}{2}\right)(0) & 0-\left(\frac{2.67}{2}\right)(0) & 2.67-\left(\frac{2.67}{2}\right)(2) & 0.74-\left(\frac{2.67}{2}\right)(-1.33) & -6.37-\left(\frac{2.67}{2}\right)(1.67) \\ 0-\left(\frac{2.25}{2}\right)(0) & -6.75-\left(\frac{2.25}{2}\right)(0) & 2.25-\left(\frac{2.25}{2}\right)(2) & 7.75-\left(\frac{2.25}{2}\right)(-1.33) & -0.5-\left(\frac{2.25}{2}\right)(1.67) \\ 0 & 0 & 2 & -1.33 & 1.67 \\ 0-\left(\frac{4.33}{2}\right)(0) & 0-\left(\frac{4.33}{2}\right)(0) & 4.33-\left(\frac{4.33}{2}\right)(2) & -6.41-\left(\frac{4.33}{2}\right)(-1.33) & 11.7-\left(\frac{4.33}{2}\right)(1.67) \end{array}\right] = \left[\begin{array}{cccc|c} -4 & 0 & 0 & 2.52 & -8.59 \\ 0 & -6.75 & 0 & 9.25 & -2.375 \\ 0 & 0 & 2 & -1.33 & 1.67 \\ 0 & 0 & 0 & -3.52 & 8.09 \end{array}\right]$$

4. Perform row operations to make the **fourth** column zeros (except element a_{44}).

a. $R1 = R1 - \left(\frac{a_{14}}{a_{44}}\right) R4 \rightarrow R1 = R1 - \left(\frac{2.52}{-3.52}\right) R4$

b. $R2 = R2 - \left(\frac{a_{24}}{a_{44}}\right) R4 \rightarrow R2 = R2 - \left(\frac{9.25}{-3.52}\right) R4$

c. $R3 = R3 - \left(\frac{a_{34}}{a_{44}}\right) R4 \rightarrow R3 = R3 - \left(\frac{-1.33}{-3.52}\right) R4$

$$\left[\begin{array}{cccc|c} -4-\left(\frac{2.52}{-3.52}\right)(0) & 0-\left(\frac{2.52}{-3.52}\right)(0) & 0-\left(\frac{2.52}{-3.52}\right)(0) & 2.52-\left(\frac{2.52}{-3.52}\right)(-3.52) & -8.59-\left(\frac{2.52}{-3.52}\right)(8.09) \\ 0-\left(\frac{9.25}{-3.52}\right)(0) & -6.75-\left(\frac{9.25}{-3.52}\right)(0) & 0-\left(\frac{9.25}{-3.52}\right)(0) & 9.25-\left(\frac{9.25}{-3.52}\right)(-3.52) & -2.375-\left(\frac{9.25}{-3.52}\right)(8.09) \\ 0-\left(\frac{-1.33}{-3.52}\right)(0) & 0-\left(\frac{-1.33}{-3.52}\right)(0) & 2-\left(\frac{-1.33}{-3.52}\right)(0) & -1.33-\left(\frac{-1.33}{-3.52}\right)(-3.52) & 1.67-\left(\frac{-1.33}{-3.52}\right)(8.09) \\ 0 & 0 & 0 & -3.52 & 8.09 \end{array}\right] = \left[\begin{array}{cccc|c} -4 & 0 & 0 & 0 & -2.8 \\ 0 & -6.75 & 0 & 0 & 18.9 \\ 0 & 0 & 2 & 0 & -1.4 \\ 0 & 0 & 0 & -3.52 & 8.09 \end{array}\right]$$

5. Divide (or multiply by the inverse) each row by the value along the diagonal.

a. $R1 = \left(\frac{1}{a_{11}}\right) R1 = \left(\frac{1}{-4}\right) R1$

b. $R2 = \left(\frac{1}{a_{22}}\right) R2 = \left(\frac{1}{-6.75}\right) R2$

c. $R3 = \left(\frac{1}{a_{33}}\right) R3 = \left(\frac{1}{2}\right) R3$

d. $R4 = \left(\frac{1}{a_{44}}\right) R4 = \left(\frac{1}{-3.52}\right) R4$

$$\left[\begin{array}{cccc|c} -4/-4 & 0 & 0 & 0 & -2.8/-4 \\ 0 & -6.75/-6.75 & 0 & 0 & 18.9/-6.75 \\ 0 & 0 & 2/2 & 0 & -1.4/2 \\ 0 & 0 & 0 & -3.52/-3.52 & 8.09/-3.52 \end{array}\right] = \left[\begin{array}{cccc|c} 1 & 0 & 0 & 0 & 0.7 \\ 0 & 1 & 0 & 0 & -2.8 \\ 0 & 0 & 1 & 0 & -0.7 \\ 0 & 0 & 0 & 1 & -2.3 \end{array}\right]$$

SOLUTION: $\boldsymbol{x_1 = 0.7, x_2 = -2.8, x_3 = -0.7, x_4 = -2.3}$

Appendix B: Kirchoff Current Law example

Given a circuit composed only of current sources and resistors, solve for the voltage at each node using Kirchoff's current law (KCL).

Definitions:

Kirchoff Current Law:

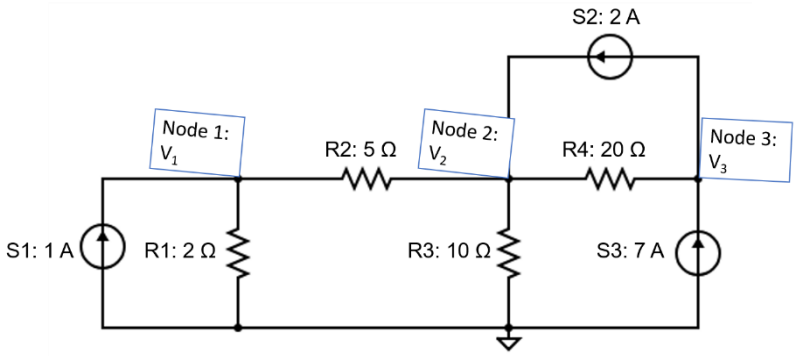
- The net current entering a node is zero
- The sum of the currents going into a node equals the sum of the current leaving a node
- The net current leaving a node is zero

Ohm's Law: $Current = \frac{Voltage}{Resistance}$

Circuit assumptions:

1. Current sources are defined such that current is entering into the node
2. Current through a resistor is defined as exiting a node

Example:



1. Sum all the currents going into and out of node **1**:

$$S1 = \frac{V_1 - 0}{R1} + \frac{V_1 - V_2}{R2} \rightarrow S1 = V_1\left(\frac{1}{R1} + \frac{1}{R2}\right) + V_2\left(-\frac{1}{R2}\right)$$

$$1 = V_1\left(\frac{1}{2} + \frac{1}{5}\right) + V_2\left(\frac{1}{5}\right)$$

2. Sum all the currents going into and out of node **2**:

$$S2 = \frac{V_2 - V_1}{R2} + \frac{V_2 - 0}{R3} + \frac{V_2 - V_3}{R4} \rightarrow S2 = V_1\left(-\frac{1}{R2}\right) + V_2\left(\frac{1}{R2} + \frac{1}{R3} + \frac{1}{R4}\right) + V_3\left(-\frac{1}{R4}\right)$$

$$2 = V_1\left(-\frac{1}{5}\right) + V_2\left(\frac{1}{5} + \frac{1}{10} + \frac{1}{20}\right) + V_3\left(-\frac{1}{20}\right)$$

3. Sum all the currents going into and out of node **3**:

$$-S2 + S3 = \frac{V_3 - V_2}{R4} \rightarrow -S2 + S3 = V_2\left(-\frac{1}{R4}\right) + V_3\left(\frac{1}{R4}\right)$$

$$-2 + 7 = V_2\left(-\frac{1}{R4}\right) + V_3\left(\frac{1}{R4}\right)$$

4. Write system of equations in matrix form

$$\begin{bmatrix} \frac{1}{R1}+\frac{1}{R2} & -\frac{1}{R2} & 0 & S1 \\ -\frac{1}{R2} & \frac{1}{R2}+\frac{1}{R3}+\frac{1}{R4} & -\frac{1}{R4} & S2 \\ 0 & -\frac{1}{R4} & \frac{1}{R4} & -S2+S3 \end{bmatrix} \rightarrow \begin{bmatrix} \frac{1}{2}+\frac{1}{5} & \frac{1}{5} & 0 & 1 \\ -\frac{1}{5} & \frac{1}{5}+\frac{1}{10}+\frac{1}{20} & -\frac{1}{20} & 2 \\ 0 & -\frac{1}{20} & \frac{1}{20} & -2+7 \end{bmatrix}$$

5. Solve system of equation using Gauss-Jordan elimination

$$\begin{bmatrix} \frac{1}{2}+\frac{1}{5} & \frac{1}{5} & 0 & 1 \\ -\frac{1}{5} & \frac{1}{5}+\frac{1}{10}+\frac{1}{20} & -\frac{1}{20} & 2 \\ 0 & -\frac{1}{20} & \frac{1}{20} & -2+7 \end{bmatrix} \rightarrow \left[\begin{array}{ccc|c} 1 & 0 & 0 & 10 \\ 0 & 1 & 0 & 30 \\ 0 & 0 & 1 & 130 \end{array}\right]$$

SOLUTION: $\boldsymbol{V_1 = 10, V_2 = 30, V_3 = 130}$

Circuit in a text file:

File contents	*Description of line*
7	*Number of branches*
3	*Number of nodes*
1 1 1 0	*S1: source, 1A, node 1, ground*
0 2 1 0	*R1: resistor, 2 ohm, node 1, ground*
0 5 1 2	*R2: resistor, 5 ohm, node 1, node 2*
0 10 2 0	*R3: resistor, 10 ohm, node 2, ground*
1 2 2 3	*S2: source, 2A, node 2, node 3*
0 20 2 3	*R4: resistor, 20 ohm, node 2, node 3*
1 7 3 0	*S3: source, 7 A, node 3, ground*

Example output:

```
V1 =    10.0000
Node 1 -       1.00 Amps - Node 0
Node 1 -       2.00 Ohms - Node 0
Node 1 -       5.00 Ohms - Node 2
V2 =    30.0000
Node 2 -      10.00 Ohms - Node 0
Node 2 -       2.00 Amps - Node 3
Node 2 -      20.00 Ohms - Node 3
V3 =   130.0000
Node 3 -       7.00 Amps - Node 0
```

Appendix C: Style guide

Comments: Comments must be used sufficiently throughout your program to document the purpose of the code and each function and provide insight into your developed algorithm.

Indentation: Your program is expected to follow standard indentation styles. The contents of the outermost code block should not be indented. The contents of any code within a code block should be indented.

Braces: All ending braces should be on their own line. Starting braces may be on their own line or following the declaration of the structure.

Example:

```
// CORRECT
int main( void )
{
    for( int i = 0; i < N; i++ ) {
        if( i < x ) {
            doSomething();
        }
        else {
            doSomethingElse();
        }
    }

    return 0;
}

// INCORRECT
int main( void )
{
    for( int i = 0; i < N; i++ ) {
    if( i < x ) {
    doSomething();
    }
    else {
doSomethingElse();}}

    return 0;
}
```

Appendix D: Rubrics

Category	Possible Points
Part A: due 10/23 in zyBooks	**50**
Checkpoint A1: completed/submitted on zyBooks	5
Checkpoint A2: completed/submitted on zyBooks	5
Checkpoint A3: completed/submitted on zyBooks	10
Checkpoint A4: Performs dynamic memory allocation, with no memory leaks, depending on the user defined matrix size	5
Checkpoint A4: Performs random number generation with a user defined seed to fill the matrix with arbitrary integers between -5 and 5	5
Checkpoint A4: Utilizes a function (PerformGaussElimination) to perform Gauss-Jordan elimination given an arbitrarily sized matrix	15
Checkpoint A4: Prints out the solution according to the specified format	5
Part B	**50**
Checkpoint B1: completed/submitted on zyBooks	10
Checkpoint B2: completed/submitted on zyBooks	10
Checkpoint B3: completed/submitted on zyBooks **due 10/30 in zyBooks**	10
Checkpoint B4: completed/submitted on zyBooks	10
Checkpoint B5: completed/submitted on zyBooks **due 11/6 in zyBooks**	10
Final code submission due 11/6 in UBlearns	**75**
Code is properly submitted with the required files/types, adheres to the style guide, and is sufficiently commented	10
Defines a Branch structure to store each element defined in the circuit	10
Performs file input/output to read in the circuit values from a file and write out the results to the same file	10
Performs dynamic memory allocation with, no memory leaks, to store an arbitrarily sized circuit and the solution	10
Utilizes a function (SolveCircuit) to translate the circuit into the system of equations in an array and utilizes the Gauss-Jordan elimination solver to solve for the node voltages	20
Utilizes a function (PrintCircuit) to print out the circuit according to the specified format	15
Final Report due 11/6 in UBlearns	**45**
The report is professionally presented with clear organization and thought	5
Description of code is sufficient and meaningful	10
Part A: Results are presented in an organized and easy to understand manner showing the output for each required set of inputs	10
Part B: Results are presented in an organized and easy to understand manner showing the output for each circuit	10
Reflections include answers to all questions with sufficient detail	10
Total	**220**