<u>CODE:</u>

```cpp
#define _CRT_SECURE_NO_WARNINGS // To use strtok instead of strtok_s
#include <iostream>
#include <vector>
#include <cstring> // To use `strtok` function
#include <string>

using namespace std;

double calculate_resistance(vector<double>& resistances, char type) {
    double total_resistance = 0.0;

    if (type == 'S') {
        for (double resistance : resistances) {
            total_resistance += resistance;
        }
    }
    else {  // Parallel circuit
        for (double resistance : resistances) {
            total_resistance += 1.0 / resistance;
        }
        total_resistance = 1.0 / total_resistance;
    }

    return total_resistance;
}

bool is_number(const string& s) {
    char* p;
    (void)strtod(s.c_str(), &p);
    /*
  Ignoring the return value is fine in this case, because the function is only used to
check if the string is a number.
    In the is_number function, strtod(s.c_str(), &p) is used to check if the entire
string s can be interpreted as a number:
    - s.c_str() is the string to check.
    - &p is a pointer to a char that will be set to point to any part of the string
that couldn't be converted.
    */
    return *p == 0;
}

int main() {
    string circuit_description;
    cout << "Enter circuit description: ";
    getline(cin, circuit_description);
```

```cpp
    vector<vector<double>> resistances_stack;
    vector<char> types;
    char* token = strtok(&circuit_description[0], " ");

    while (token != NULL) {
        string token_str(token);
        if (token_str == "E" || token_str == "e") {
            if (!types.empty()) {
                char type = types.back();
                types.pop_back();
                vector<double> current_resistances = resistances_stack.back();
                resistances_stack.pop_back();
                double equivalent_resistance =
calculate_resistance(current_resistances, type);
                if (!resistances_stack.empty()) {
                    resistances_stack.back().push_back(equivalent_resistance);
                }
                else {
                    resistances_stack.push_back(vector<double>{equivalent_resistance});
                }
            }
        }
        else if (token_str == "s" || token_str == "p" || token_str == "S" || token_str
== "P") {
            token_str = toupper(token_str[0]); // here we convert s & p to uppercase to
handle the calculate_resistance function, we assume a user might be too lazy to
uppercase the letters (example: me XD)
            types.push_back(token_str[0]);
            resistances_stack.push_back(vector<double>{});
        }
        else {
            if (is_number(token_str)) {
                resistances_stack.back().push_back(stod(token_str));
            }
            else {
                cout << "Wrong Circuit Description" << endl;
                return 1;
            }
        }

        token = strtok(NULL, " ");
    }

    cout << "Equivalent resistance: " << resistances_stack.back()[0] << " ohms" <<
endl;

    cout << "Enter voltage: ";
    double voltage;
    cin >> voltage;
    cout << "Current: " << voltage / resistances_stack.back()[0] << " A" << endl;

    return 0;
}
```
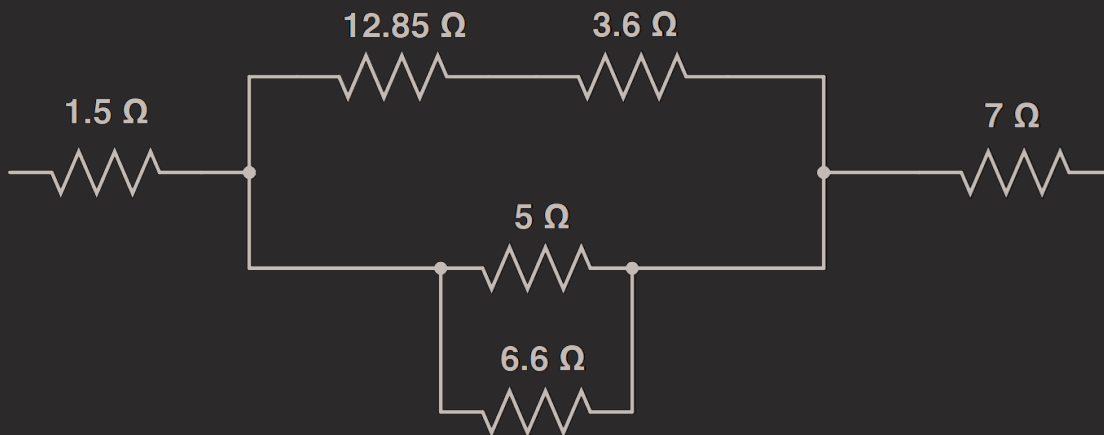
TEST CASES:

1 Circuit description: **S 1.5 P S 12.85 3.6 e 5 6.6 e 7 E**
Voltage applied: **3.8**

```
Enter circuit description: s 1.5 p s 12.85 3.6 e 5 6.6 e 7 E
Equivalent resistance: 10.9254 ohms
Enter voltage: 3.8
Current: 0.347814 A
```

Theoretical Solution:

$1.5\ \Omega + (\,(12.85\ \Omega + 3.6\ \Omega)\ \|\ (\,5\ \Omega\ \|\ 6.6\ \Omega)\,) + 7\ \Omega = 10.92\ \Omega$ **#PASS**

$\because V = IR,\ V = 3.8\ V \therefore I = 3.8/10.92 = 0.348\ A$ **#PASS**

2 Circuit description: **S L 2.5 5.2 e 4.7 8 E**
Voltage applied: **9**

```
Enter circuit description: S L 2.5 5.2 e 4.7 8 E
Wrong Circuit Description
```
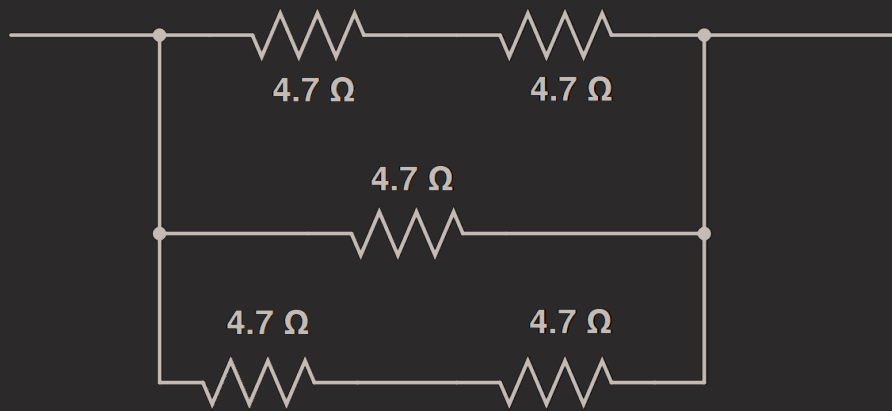
Theoretical Solution:
(NOT APPLICABLE)

Wrong Circuit Description   **#PASS**

3 Circuit description: **P S 4.7 4.7 e 4.7 S 4.7 4.7 e E**
Voltage applied: **7**

```
Enter circuit description: P S 4.7 4.7 e 4.7 S 4.7 4.7 e E
Equivalent resistance: 2.35 ohms
Enter voltage: 7
Current: 2.97872 A
```

Theoretical Solution:

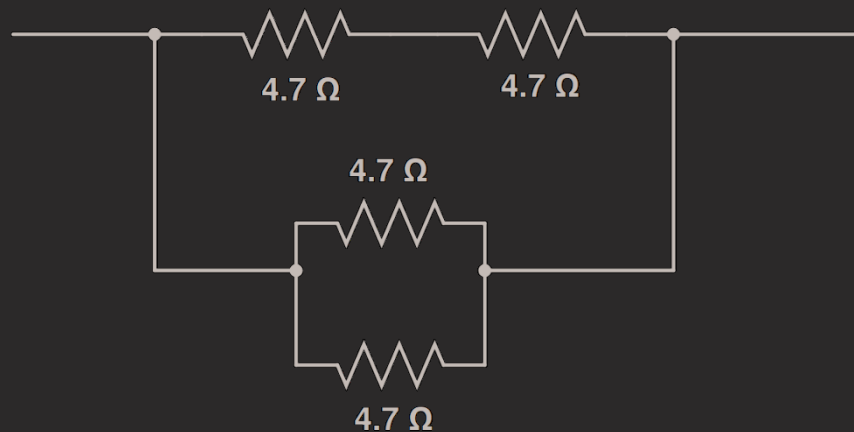$$(4.7\ \Omega + 4.7\ \Omega)\ \|\ (4.7\ \Omega)\ \|\ (4.7\ \Omega + 4.7\ \Omega) = 2.35\ \Omega$$ #PASS

$$\because V = IR,\ V = 7\ V \therefore I = 7/2.35 = 2.98\ A$$ #PASS

④ Circuit description: **P S 4.7 4.7 e 4.7 4.7 E**
  Voltage applied: **9**

```
Enter circuit description: P S 4.7 4.7 e 4.7 4.7 E
Equivalent resistance: 1.88 ohms
Enter voltage: 9
Current: 4.78723 A
```

Theoretical Solution:

$$(4.7\ \Omega + 4.7\ \Omega)\ \|\ (4.7\ \Omega\ \|\ 4.7\ \Omega) = 1.88\ \Omega$$ #PASS

$$\because V = IR,\ V = 9\ V \therefore I = 9/1.88 = 4.79\ A$$ #PASS

5 Circuit description: **Z S 8.2 3.1 e 1.3 7.8 E**
   Voltage applied: **5**

```
Enter circuit description: Z S 8.2 3.1 e 1.3 7.8 E
Wrong Circuit Description
```

Theoretical Solution:

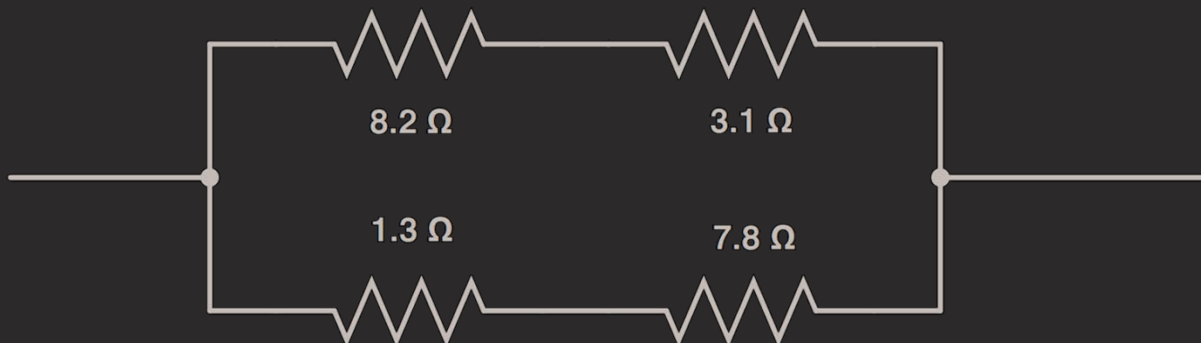 (NOT APPLICABLE)

   Wrong Circuit Description   #PASS

6 Circuit description: **P S 8.2 3.1 e S 1.3 7.8 e E**
   Voltage applied: **5**

```
Enter circuit description: P S 8.2 3.1 e S 1.3 7.8 e E
Equivalent resistance: 5.04069 ohms
Enter voltage: 5
Current: 0.991928 A
```

Theoretical Solution:

$(8.2 \, \Omega + 3.1 \, \Omega) \parallel (1.3 \, \Omega + 7.8 \, \Omega) = 5.04 \, \Omega$ #PASS

$\because V = IR , V = 5 \, V \therefore I = 5/5.04 = 0.992 \, A$ #PASS
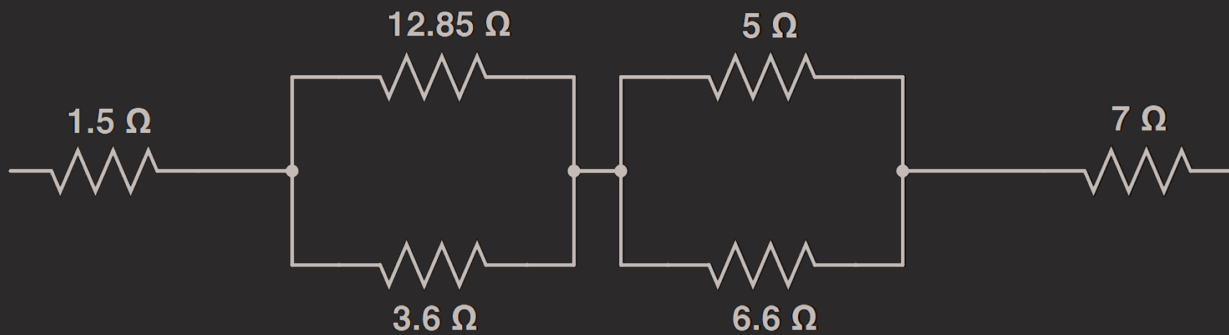
7 (THIS IS CASE **1** MODIFIED)

Circuit description: **S 1.5 P 12.85 3.6 e P 5 6.6 e 7 E**
Voltage applied: **3.8**

```
Enter circuit description: S 1.5 P 12.85 3.6 e P 5 6.6 e 7 E
Equivalent resistance: 14.157 ohms
Enter voltage: 3.8
Current: 0.268419 A
```

Theoretical Solution:

$$1.5\,\Omega + (12.85\,\Omega \parallel 3.6\,\Omega) + (5\,\Omega \parallel 6.6\,\Omega) + 7\,\Omega = 14.16\,\Omega \qquad \text{\#PASS}$$
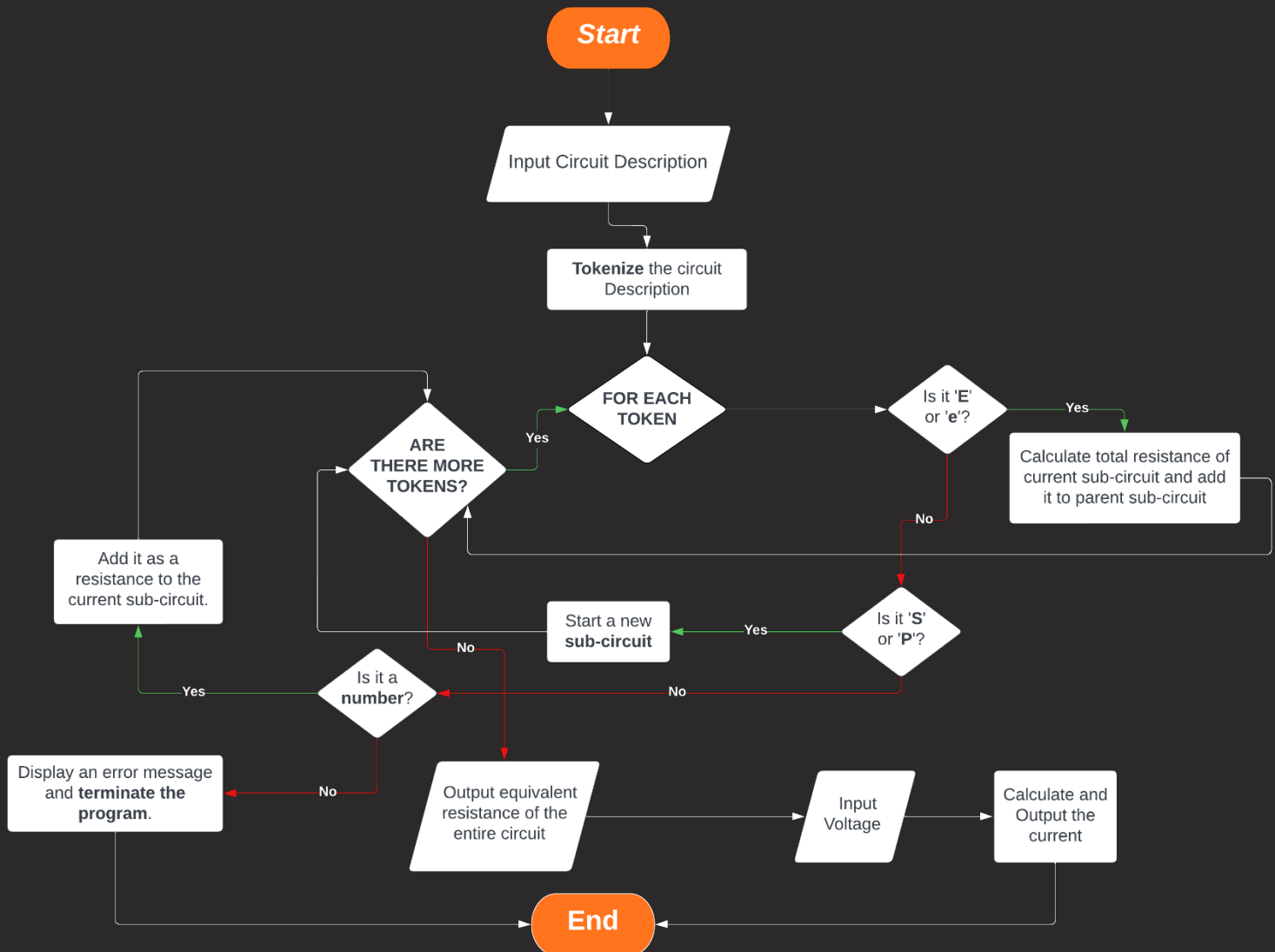
$$\because V = IR,\ V = 3.8\,\text{V} \therefore I = 3.8/14.16 = 0.268\,\text{A} \qquad \text{\#PASS}$$

FLOWCHART:

Basic Flowchart:

This flowchart is simplified, doesn't include **all** the code details, but makes it easier to follow the code logic, which is the basic purpose of a flowchart.

The flowchart accurately depicts the main purpose of the program, the flow of the code and the logic behind it.

It does not include every step taken in a code-ish manner, but it includes the major steps and their flow to be easily understood by someone who is relatively unrelated to coding.

A detailed representation of the code is shown in the following page…

Detailed Process:

This is the detailed flow of the process; I did not represent it in a flowchart because doing so would contradict the main purpose of flowcharts; simplicity.

1. Start
2. Output: "Enter circuit description: "
3. Input circuit description
4. Tokenize circuit description using whitespace as delimiter
5. Initialize empty vectors: resistances, types
6. Start loop: For each token
    - Is the token 'E' or 'e'?
        - Yes:
            - If types vector is not empty:
                - Pop last element from types vector, assign it to 'type'
                - Calculate the total resistance of the current sub-circuit based on 'type' (using calculate_resistance function)
                - Clear resistances vector
                - Add the calculated resistance to resistances vector
            - Go to the next iteration (continue to next token)
        - No: Go to the next step
    - Is the token 'S' or 'P'?
        - Yes:
            - Push token to types vector
            - Go to the next iteration (continue to next token)
        - No: Go to the next step
    - Is the token a number (using is_number function)?
        - Yes:
            - Convert token to double, add it to resistances vector
            - Go to the next iteration (continue to next token)
        - No:
            - Output: "Wrong Circuit Description"
            - End program (return 1)
7. End loop
8. Output: "Equivalent resistance: " + first element of resistances vector + " ohms"
9. Output: "Enter voltage: "
10. Input voltage
11. Calculate current = voltage / first element of resistances vector
12. Output: "Current: " + current + " A"
13. End program (return 0)
14. End

GitHub

SCAN THE QR CODE FOR:

- Complete project progress
- Current and previous versions of code
- Source files
- First access to any edits and project extras
- PDFs including:
    - Given tasks.
    - My documentation.
      (LIGHT & DARK themed)

OR VISIT THE LINK:

https://bit.ly/CSE131Proj

[or you can also just click on the **GitHub** icon above :) **]**