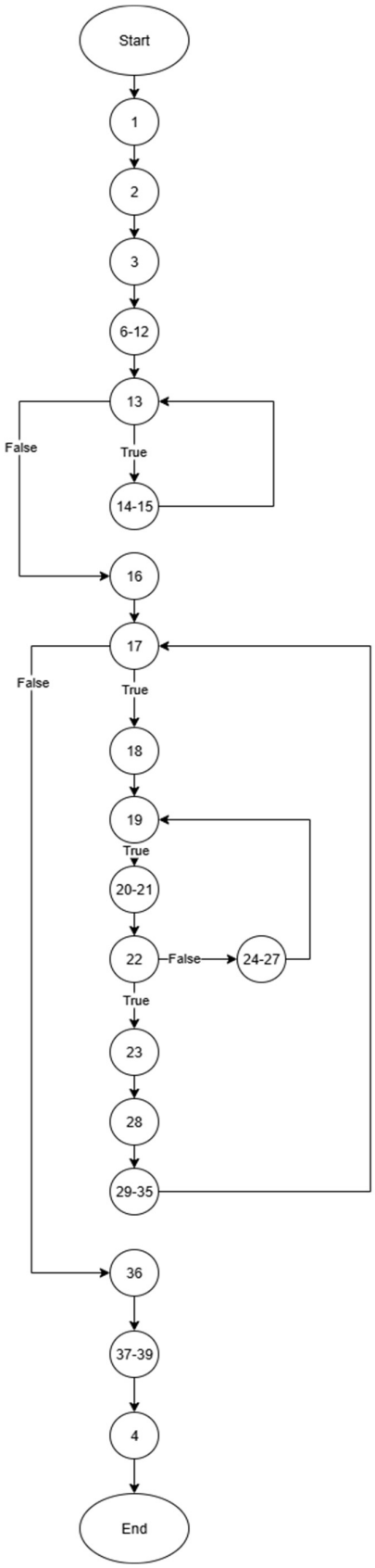
LAB 2: Problem Statement: Re-write and extend the minimum.c program (exercise 1.4) with the name Min_Max_Safe_Input for the following additional conditions:

1. Array size entered by the user will be whole number between 1 to 100
2. The number entered by the user in the Array cannot be a character but can be any real number.
3. The program shall also find minimum and maximum value entered in the array.

```c
#include <stdio.h>
#include <float.h>
void MinMax();
int main() {
    MinMax();
    return 0;
}
void MinMax() {
    double array[100];
    double min = DBL_MAX, max = -DBL_MAX;
    int Number;
    int i;
    double tmpData;
    printf("Enter the size of the array (1-100): ");
    while (scanf("%d", &Number) != 1 || Number < 1 || Number > 100) {
        printf("Invalid size! Enter a whole number between 1 and 100: ");
        while (getchar() != '\n');
    }
    for (i = 0; i < Number; i++) {
        printf("Enter A[%d]: ", i + 1);
        while(1){
            char term;
            int result = scanf("%lf%c", &tmpData, &term);
            if (result == 2 && term == '\n') {
                break;
            }
            else {
                printf("Invalid input! Enter a real number: ");
            }
        }
        array[i] = tmpData;
        if (tmpData < min){
            min = tmpData;
        }
        if (tmpData > max){
            max = tmpData;
        }
    }
    printf("Minimum value: %.6lf\n", min);
    printf("Maximum value: %.6lf\n", max);
}
```

2.1: **Prepare the CFG of the minimum.c Program**



2.2 **Write the test cases to find the errors in the Program**.

S. No	Input	Expected Output	Actual Input	Result	Comments
1.	Size=4 [-1.5, -2.5, 3, 4]	-2.5, 4	-2.5, 4	Pass	Handles negative real numbers
2.	Size=2 [11.50, 12]	11.5, 12	11.5, 12	Pass	Handles all real numbers
3.	Size=1 [c]	Error	Error	Pass	Input data type check
4.	Size=-1	Invalid	Invalid	Pass	Size input type check

1. Test Case1:

```
Enter the size of the array (1-100): 4
Enter A[1]: -1.5
Enter A[2]: -2.5
Enter A[3]: 3
Enter A[4]: 4
Minimum value: -2.500000
Maximum value: 4.000000
```

2. Test Case2:

```
afe_input.c -o min_max_safe_input } ; if
Enter the size of the array (1-100): 2
Enter A[1]: 11.50
Enter A[2]: 12
Minimum value: 11.500000
Maximum value: 12.000000
PS C:\Users\ahuja\OneDrive\Desktop\Softwa
```

3. Test Case3:

```
Enter any real number (int,float,double) A[1]: c
Invalid input. Please enter an integer for A[1]:
```

4. Test Case4:

```
Enter the size of the array (1-100): -1
Invalid size! Enter a whole number between 1 and 100:
```

2.3: **Summarize the errors to facilitate the user to debug the program**

There is no error in the program; it is able to handle all edge cases.

2.4: **Debug the Program for the identified error**

The code is correct, no debugging is needed.

2.5: **Prepare the set of Test Cases to re-test the Program**.

The above code is checked in exercise 2.2.