FirstOrder.C

```
// File used to deal with 1st order systems
#include <stdio.h> // Including stdio.h header file

int askUserFirst(struct transfer_function_first_order *t1, int i, float
step); // Declaration of the askUserFirst function to ask the user for
the transfer functions values
void DisplaySystemFirst(struct transfer_function_first_order *t1, int i);
// Declaration of the Display function to display the transfer functions
values for the system
void remove(void);
// Declartion of the clear function to clear the input buffer

struct transfer_function_first_order // Defining the transfer function
structure for the first order system
{
    float a; // Defining the numerator
    float b; // Defining the s term in the denominator
    float c; // Defining the sole pole in the denominator
};

int askUserFirst(struct transfer_function_first_order *t1, int i, float
step) // Deinition of the askUserFirst function to ask the user for the
transfer functions values
{
    float alph;
// Initialising the temporary alpha variable, used to calculate the
numerator of the open loop system transfer function
    printf("\nPlease enter the transfer function of the system you wish
to design the controller for:\n"); // Tells the user this is now the
transfer function values they're entering
    printf("     a\n --------\n (bs + c)\n");
// Shows the user the format of their entry
    printf("Note all values must be between -10000 and 10000\n");
// Lets the userknow the range of valid inputs

// Begins the propmting of the required values
    do
// Loops through at least once, as the do while is initiated
    {
        printf("a:\t");                                                 //
Asks the user to enter the value of the numerator
        if (scanf("%f", &alph) != 1 || alph > 10000 || alph < -10000) //
Prompts the user for an answer and checks the answer is numerical
        {
            printf("Invalid input, please enter a number between -10000
and 10000\n"); // Error message thrown if the input is invalid
        };
        remove(); // Clears the buffer
    } while (alph > 10000 || alph < -10000); // Keeps looping through
until a valid answer is entered
    t1[i].a = (alph * step); // Calculates the value of the numerator
using the step input and alph vairable, and stores it in the transfer
function structure's a value
    do                      // Loops through at least once, as the do
while is initiated
    {
```

```
        printf("b:\t");
// Asks the user to enter the value of the s term in the denominator
        if (scanf("%f", &t1[i].b) != 1 || (t1[i].b > 10000 || t1[i].b < -
10000)) // Prompts the user for an answer and checks the answer is
numerical
        {
            printf("Invalid input, please enter a number between -10000
and 10000\n"); // Error message thrown if the user enters an invalid
input
        };
        remove(); // Clears the buffer
    } while (t1[i].b > 10000 || t1[i].b < -10000); // Keeps looping
through until a valid answer is entered
    do // Loops through at least once, as the do while is initiated
    {
        printf("c:\t");
// Asks the user to enter the pole
        if (scanf("%f", &t1[i].c) != 1 || (t1[i].c > 10000 || t1[i].c < -
10000)) // Prompts the user for an answer and checks the answer is
numerical
        {
            printf("Invalid input, please enter a number between -10000
and 10000\n"); // Error message thrown if the user enters an invalid
input
        };
        remove(); // Clears the buffer
    } while (t1[i].c > 10000 || t1[i].c < -10000); // Keeps looping
through until a valid answer is entered

    return 0; // Returns zero to signify the end of the function
};

void DisplaySystemFirst(struct transfer_function_first_order *t1, int i)
// Definition of the function to display the first order system
{
    printf("You have entered the folloing transfer function:\n");
// Tells the user what will be shown to them
    printf("\t%f\n ----------------------\n (%.5f * S + %.5f)\n",
t1[i].a, t1[i].b, t1[i].c); // Displays teh actual values to the user in
the required format
};

void remove(void) // remove function definition
{
    int c = 0; // Defining the input variable used to clear the buffer
    do         // Runs at least once
    {
        c = getchar(); // Gets the next character in the buffer
    } while (c != '\n'); // loops through until it reaches a new line
};
```