

## Report on the basic Math Calculator created by C program.

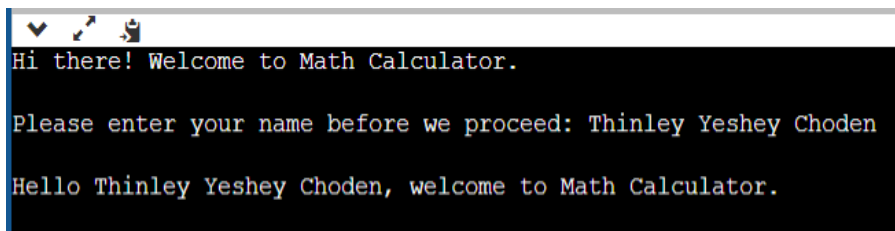
### Introduction

The program is a basic math calculator that receives two numbers and an operation from the user as inputs which gets calculated and displayed as result. The program is written in C and uses standard input output, as well as string header file functions for interaction with user.

### Program flow

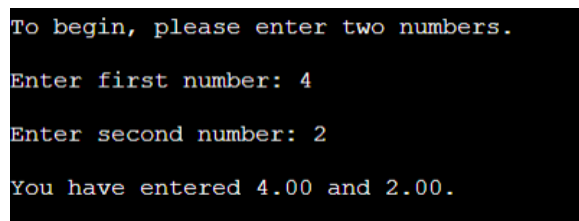
#### User input

- 1) The program starts by asking the user to input their name using printf() function.
- 2) scanf() was going to be used to store the name input but fgets() was used instead to enable the user to input their full name (including the blank space).



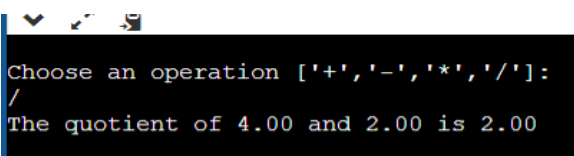
```
Hi there! Welcome to Math Calculator.  
  
Please enter your name before we proceed: Thinley Yeshey Choden  
  
Hello Thinley Yeshey Choden, welcome to Math Calculator.
```

- 3) It then reads in the user's name using fgets(), which limits the input to 30 characters and prevents buffer overflow.
- 4) The newline character at the end of the input is removed using strcspn() and replaced with a null terminator.
- 5) Next, the program asks the user to input two numbers using printf() and reads the value using scanf() with %.2lf as a format specifier indicating a double floating-point number.



```
To begin, please enter two numbers.  
  
Enter first number: 4  
  
Enter second number: 2  
  
You have entered 4.00 and 2.00.
```

- 6) Finally, the program prompts the user to select and enter an operator from '+', '-', '\*', or '/' using printf() and reads the operator using scanf().



```
Choose an operation ['+', '-', '*', '/']:  
/  
The quotient of 4.00 and 2.00 is 2.00
```

## Calculation

- 1) The selected program operation has been performed using the 'do...while' loop structure as well as the 'if else' and 'switch' loop structure.
- 2) Firstly, the program declares three variables: 'operator' of type 'char' to store the arithmetic operation selected by the user, 'valid\_operator' of type 'int' to ensure that the user enters a valid operator, and 'result' of type 'double' to store the result of the operation.

```
char operator;  
int valid_operator = 0;  
double result;
```

- 3) The program enters a do while loop that will repeatedly ask the user to enter an operator until a valid one has been entered.
- 4) A switch statement is used inside the 'do...while' loop to determine the mathematical operation that needs to be performed, using the 4 cases and 1 default statement.
- 5) If a valid operation is entered by the user, the 'valid\_operator' variable is set to 1 causing the loop to terminate and the program to move on to the next line of code after the loop. The corresponding mathematical operation is performed on the two inputs and the result gets stored in the 'result' variable

```
Enter first number: 4  
Enter second number: 2  
You have entered 4.00 and 2.00.  
Choose an operation ['+', '-', '*', '/']:  
+  
The sum of 4.00 and 2.00 is 6.00
```

```
Enter first number: 4  
Enter second number: 2  
You have entered 4.00 and 2.00.  
Choose an operation ['+', '-', '*', '/']:  
*  
The product of 4.00 and 2.00 is 8.00
```

```
Enter first number: 4  
Enter second number: 2  
You have entered 4.00 and 2.00.  
Choose an operation ['+', '-', '*', '/']:  
-  
The difference of 4.00 and 2.00 is 2.00
```

- 6) If the user enters a division(/) operator, the statement will enter an 'if...else' loop to check if the second input [num2] is 0, if so; the program prints out an error statement. And since the 'variable\_operator' is still 0, the program asks user to input another operator.

```
Enter first number: 4
Enter second number: 0
You have entered 4.00 and 0.00.
Choose an operation ['+', '-', '*', '/']:
/
Error: division by zero!
Choose an operation ['+', '-', '*', '/']:
█
```

- 7) If [num2] is not equal to 0, the division operation is performed as usual.

```
Enter first number: 4
Enter second number: 2
You have entered 4.00 and 2.00.
Choose an operation ['+', '-', '*', '/']:
/
The quotient of 4.00 and 2.00 is 2.00
```

- 8) Finally, if the user inputs an invalid operator, the default switch statement gets executed printing out an invalid statement and asking the user to enter a valid one.

```
Enter first number: 4
Enter second number: 2
You have entered 4.00 and 2.00.
Choose an operation ['+', '-', '*', '/']:
6
Invalid operator!!!
Choose an operation ['+', '-', '*', '/']:
```

## Output

Finally, the program thanks the user for using the calculator using printf() also requesting a high review for user experience.

```
Thank you for using Math Calculator.
If you enjoyed my service, please dont forget to give a high review!
...Program finished with exit code 0
Press ENTER to exit console.█
```

## Conclusion

To sum up, the program functions as a math calculator that prompts the user to input two numbers and an arithmetic operator, performs the requested operation, and displays the result. The program relies on standard C input/output functions to interact with the user and carry out the calculation.

Overall, this code is a direct implementation of a basic calculator, with appropriate error handling for division by zero and invalid operators.