Assignment-1

Mohammed Sadiq - EE18BTECH11051

Github repository

https://github.com/Sadiq0123/C−and−DS/tree/main/Assignment−1

1 Problem

Consider the following C program:

```
#include <stdio.h>

int counter = 0;
int calc (int a, int b) {
    int c;

    counter++;
    if (b==3) return (a*a*a);
    else {
        c = calc(a, b/3);
        return (c*c*c);
    }
}

int main (){
    calc(4, 81);
    printf ("%d", counter);
}
```

The output of this program is _.

2 Solution

The answer to the question is 4. The variable `counter` is a global variable with initial value set to 0, and is incremented by 1 everytime the function `calc` is called.

The function `calc` returns a^b only if b is a power of 3; else the function returns nothing. The code behaves this way because eventually the function calls `calc(a, b/3)` where $b < 3$, which leads to an infinite recursive call of `calc(a, 0)`. So, the stack overflow causes Segmentation Fault error during runtime.

If b is valid, then `calc` recursively calls itself for $b/3$. So, the total number of times `counter` is incremented is $\log_3(b)$ times. The following table shows the value of `counter` at every step.

calc(a, b)	counter
calc(4, 81)	1
calc(4, 27)	2
calc(4, 9)	3
calc(4, 3)	4

As $\log_3(81) = 4$, the output of the code would be equal to 4.