Write a simplified shell program to receive user commands for performing the following operations on positive integers – a) Addition b) Subtraction c) Multiplications and d) Division. If the user enters wrong input, then the program should display an error message and proceed to receive the next command. The program stops when you enter the command “bye”. Handle error input using setjmp and longjmp functions.

Code:

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <setjmp.h>
jmp_buf buf;

int calculate(char operation[],int n1,int n2)
{
if (n1>=0 && n2>=0) //check numbers are positive or not
    {
        if ((strcasecmp(operation,"add") == 0)) // compares 1st token is equal to
 "add"
            {
                return n1+n2; // returns sum value
            }
        else if ((strcasecmp(operation,"sub") == 0) && n1>n2) // compares 1st tok
en is equal to "sub"
            {
                return n1-n2; // returns difference value
            }
        else if ((strcasecmp(operation,"mul") == 0)) // compares 1st token is equ
al to "mul"
            {
                return n1*n2; // returns multilpicaion value
            }
        else if ((strcasecmp(operation,"div") == 0) && n2!=0) // compares 1st tok
en is equal to "div"
            {
            return n1/n2; // returns division value
            }
        else
        {
            printf("Invalid");
             longjmp(buf,1); // Jump to the point located by setjmp
        }
    }
```

```
    else
        {
            printf("Numbers should be positive");
            longjmp(buf,1); // Jump to the point located by setjmp
        }

}


int main()
 {
    char input[100];        //input string
    char array[20][20];     //output token list

while(1)                    //To run the program continously
{
    gets(input);            //scan input string
    char* T;
    T=strtok(input," ");    //calling strtok function with delimiter ""
    int num=0;              //Declaring token index
    while (T!= NULL)        //runs strtok til token set to NULL
     {
        strcpy(array[num],T);   //copy token to array list
        num++;                  //Increment token index
        T = strtok(NULL," ");   //To get mext token in input
     }

    if (strcasecmp(array[0],"bye") == 0) // compares 1st token is equal to "bye"
    {
        exit(0);    // Terminates the program
    }
    int n1,n2,result;
    n1=atoi(array[1]);  //converts 2nd token to integer
    n2=atoi(array[2]);  //converts 3rd token to integer
    result=calculate(array[0],n1,n2); //calls calculate function
    printf("%d",result); // print returns value
    setjmp(buf);        //set the jump position using buf
    printf("\n");

}
    return 0;
}
```

Code description:

- First initialize header files and define jmp buf buf.
- In main function first initialize variables for input
- Using **strtok()** function, break the string into smaller tokens(split the sentence into words)
- Since array[1] and array[2] are declare as characters convert them to integers using **atoi()** function
- After conversion, declare the function calculate(), pass array[0], n,1,n2 variables using this function to perform operations
- In calculate function, array[0] is compared with add, sub, mul and div.
  ***strncasecmp*()** function compares two strings, while ignoring differences in case
- If array[0] is equal to "add", it performs addition and returns the value. Subtraction, multiplication and division operation takes place and returns their results similar to add operation
- If the numbers are not positive it displays the output as Numbers should be positive.
- If the result is negative and infinite it displays output as Invalid i.e. sub 6 9, div 4 0
- We use **setjmp** and **longjmp** to handle errors
- This program will execute continuously (used while(1) loop). To terminate the program, give "bye" as input

Output: