Challenge 1

Lets first read the src file

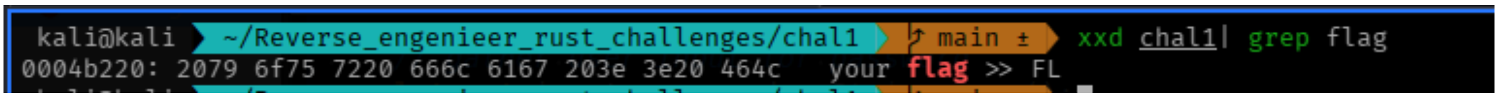
```
use std::io::{Write, stdin, stdout};

fn main() {
    // static username
    let username = "Test123";

    // static password
    let password = "Test123";

    // gather user input for username
    let mut input_username = String::new();
    print!("Please enter your username: ");
    let _ = stdout().flush();
    stdin()
        .read_line(&mut input_username)
        .expect("Did not enter string");

    // gather user input for password
    let mut input_password = String::new();
    print!("Please enter your password: ");
    let _ = stdout().flush();
    stdin()
        .read_line(&mut input_password)
        .expect("Did not enter string");

    // checking if username and password match (trim input)
    if input_username.trim() == username && input_password.trim() == password {
        println!("You did it well done!");
        println!("heres your flag >> FLAG{{Place_holder}");
    } else {
        println!("Invalid username or password.");
    }
}
```

so It has two hardcoded varbiables ‘username’ and ‘password’ and then it takes two user inputs for variables 'input_username' and 'input_password'
It then checks if the hard coded values match the input values if it does we get the flag if not it prints " Invalid username or password"

last thing to mention is it seems the flag is hardcoded

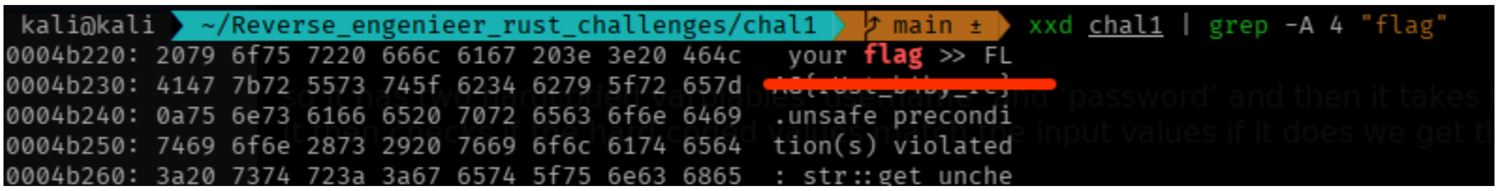
LEAKING THE FLAG

This is pretty straight forward we will leak the hardcoded flag using xxd and grep to search the hex data of the binary then greping for the string FLAG



```
kali@kali  ~/Reverse_engenieer_rust_challenges/chal1  main ±  xxd chal1| grep flag
0004b220: 2079 6f75 7220 666c 6167 203e 3e20 464c   your flag >> FL
```

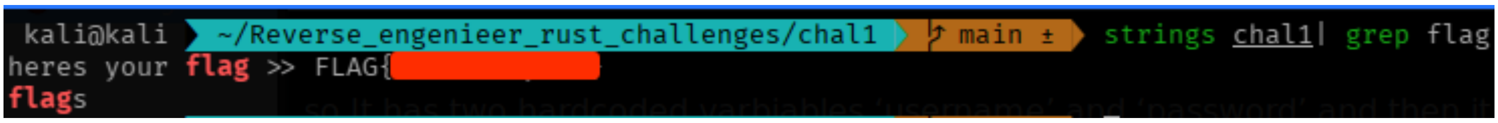
we found the string 'your flag >> FL' which means we are on the write track but just need a few more hex lines for the full flag. lucky for us grep allows to view x amount of lines with the -A flag



```
kali@kali  ~/Reverse_engenieer_rust_challenges/chal1  main ±  xxd chal1 | grep -A 4 "flag"
0004b220: 2079 6f75 7220 666c 6167 203e 3e20 464c   your flag >> FL
0004b230: 4147 7b72 5573 745f 6234 6279 5f72 657d  [redacted]
0004b240: 0a75 6e73 6166 6520 7072 6563 6f6e 6469  .unsafe precondi
0004b250: 7469 6f6e 2873 2920 7669 6f6c 6174 6564  tion(s) violated
0004b260: 3a20 7374 723a 3a67 6574 5f75 6e63 6865  : str::get_unche
```

and we have the flag

you may also use strings for a much easier approach



```
kali@kali  ~/Reverse_engenieer_rust_challenges/chal1  main ±  strings chal1| grep flag
heres your flag >> FLAG{[redacted]
flags
```