```
(gdb) disass
Dump of assembler code for function main:
   0x0000000000400577 <+0>:     push   %rbp
   0x0000000000400578 <+1>:     mov    %rsp,%rbp
   0x000000000040057b <+4>:     sub    $0x10,%rsp
=> 0x000000000040057f <+8>:     movl   $0x6,-0x4(%rbp)
   0x0000000000400586 <+15>:    mov    -0x4(%rbp),%eax
   0x0000000000400589 <+18>:    mov    %eax,%edi
   0x000000000040058b <+20>:    callq  0x40052d <factorial>
   0x0000000000400590 <+25>:    mov    $0x0,%eax
   0x0000000000400595 <+30>:    leaveq
   0x0000000000400596 <+31>:    retq
End of assembler dump.
```

This is the assembly code for main function.

The line with the arrow (movl) is to assign the number 6 to the register.

The 2 lines below (mov) is to pass the variable into the function, and make a copy of the variable.

The line below (callq) is to call the factorial function.

And finally, the line at last (retq) is to return and end the main function.

```
(gdb) disass
Dump of assembler code for function factorial:
   0x000000000040052d <+0>:     push   %rbp
   0x000000000040052e <+1>:     mov    %rsp,%rbp
   0x0000000000400531 <+4>:     sub    $0x20,%rsp
   0x0000000000400535 <+8>:     mov    %edi,-0x14(%rbp)
=> 0x0000000000400538 <+11>:    movl   $0x1,-0x4(%rbp)
   0x000000000040053f <+18>:    movl   $0x1,-0x8(%rbp)
   0x0000000000400546 <+25>:    jmp    0x400556 <factorial+41>
   0x0000000000400548 <+27>:    mov    -0x4(%rbp),%eax
   0x000000000040054b <+30>:    imul   -0x8(%rbp),%eax
   0x000000000040054f <+34>:    mov    %eax,-0x4(%rbp)
   0x0000000000400552 <+37>:    addl   $0x1,-0x8(%rbp)
   0x0000000000400556 <+41>:    mov    -0x8(%rbp),%eax
   0x0000000000400559 <+44>:    cmp    -0x14(%rbp),%eax
   0x000000000040055c <+47>:    jle    0x400548 <factorial+27>
   0x000000000040055e <+49>:    mov    -0x4(%rbp),%edx
   0x0000000000400561 <+52>:    mov    -0x14(%rbp),%eax
   0x0000000000400564 <+55>:    mov    %eax,%esi
   0x0000000000400566 <+57>:    mov    $0x400630,%edi
   0x000000000040056b <+62>:    mov    $0x0,%eax
   0x0000000000400570 <+67>:    callq  0x400410 <printf@plt>
   0x0000000000400575 <+72>:    leaveq
   0x0000000000400576 <+73>:    retq
End of assembler dump.
```

This is the assembly code for the factorial function.

The line above the arrow (mov) is to take in the parameter (%edi).

The line with the arrow (movl) is to assign the int 1 into the register.

Then, the lines after represent the for loop (movl, jmp, mov, imul, mov, addl, mov, cmp, jle). These are to assign the int 1 to i, then check if i fulfills the condition. And multiply fact by i, and assign the value to fact. And lastly, add 1 to i.

Afterwards, the code (mov) assign num and fact to the printf function, and (callq) print the message.

Finally, retq to return and leave the factotrial function.

```c
#include <stdio.h>

void factorial(int num){

    int fact = 1;
    int i;
    for(i = 1;i<=num;++i){
        fact = fact * i;
    }

    printf("The factorial of %d is %d \n",num,fact);

}

int main() {

    int factNumber = 6;
    factorial(factNumber);

    return 0;
}
```

The c code after debug looks like this. The main bug is about the for loop.

Different from other languages, c need to declare the loop counter before the loop, instead of inside the brackets.

The main login of the function is to take in a int, and initialize the factorial as 1. Then multiply the factorial by i, which will increment from1 to the input number.

And finally, print the message of the factorial result.