1 Calling C functions

Assembly program:

```
extern test
extern exit

section .text
        global main

        main:
                PUSH 1
                PUSH 2

                CALL test

                PUSH eax
                CALL exit
```

Also, the **test()** function is defined in **test.c** as:

```
int test(int a, int b){
        printf("This is test()\n");
        return (a+b);
}
```

We **PUSH** 1 and 2 into the stack(2 is the first argument of **test()** and 1 is the second argument). After this we **CALL** the **test()** function with the arguments(from the stack). Now, **test()** returns the sum of the arguments passed to it. **The return value is stored in the register:** ***eax***. So we can **PUSH** the value of **eax** onto the stack and **CALL exit()**.

Now comes the compilation part, so first we use the nasm command to assemble our program:

```
$ nasm -f elf -o asm16.o asm16.asm
```

Next, compile using **gcc**:

```
$ gcc -no-pie -m32 asm16.o test.c -o asm16
```

Here we have to provide the **C** file where we have defined our **test()** function.

We pushed **eax** into the stack and called **exit**. Now **exit** will execute as **exit(the value in eax)** and that value in **eax** will get stored in **$?** variable which we can see from the image below:

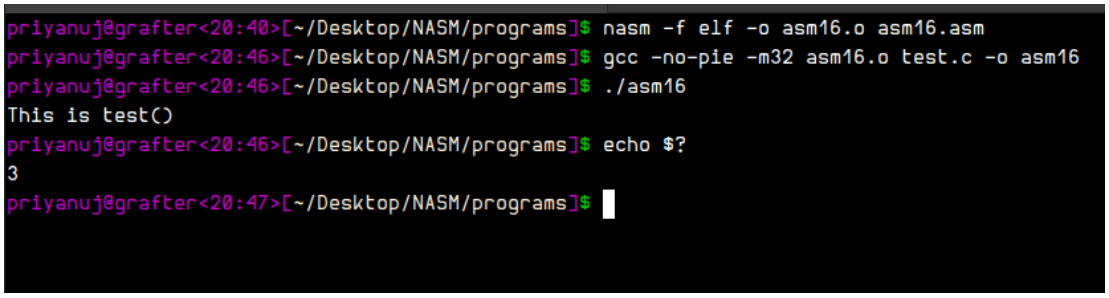

Figure 1: Output