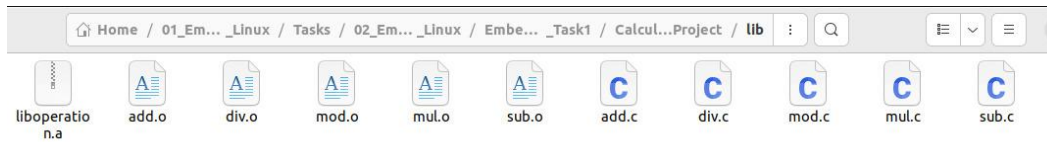
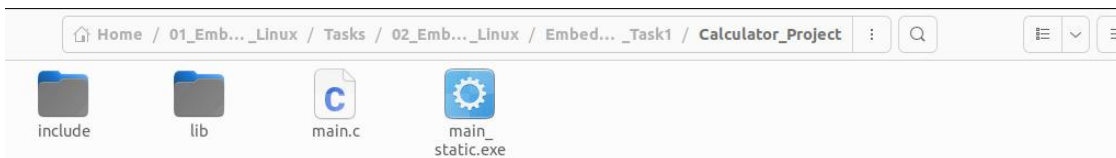


Creating a static library:

First, I created (.o) files for the five functions, then I created “liboperation.a” library statically through “ar - rcs” command



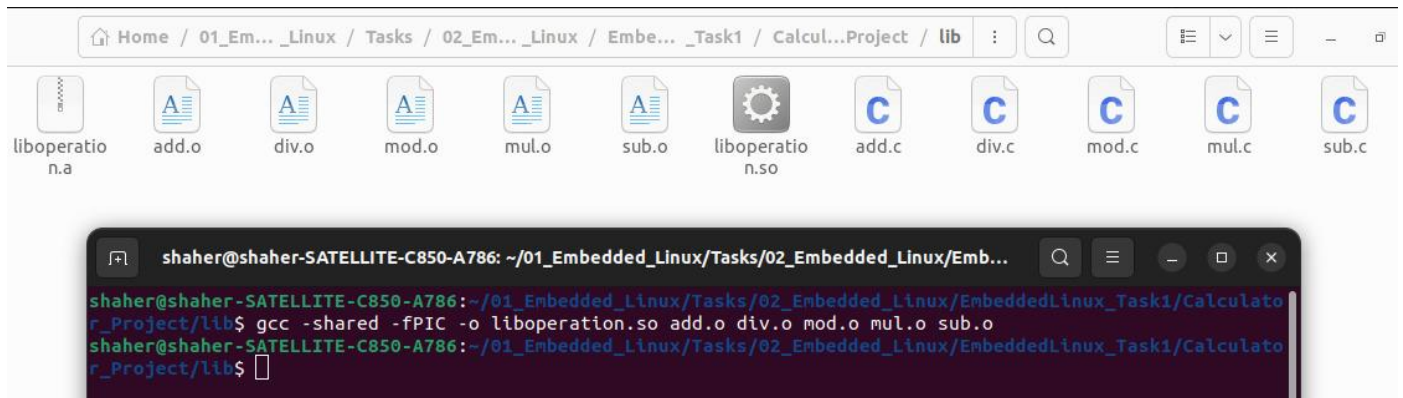
Then I compiled the (main.c) file linking that static library to it. And the I execute the “main_static.exe” file to test and make sure that everything is fine.



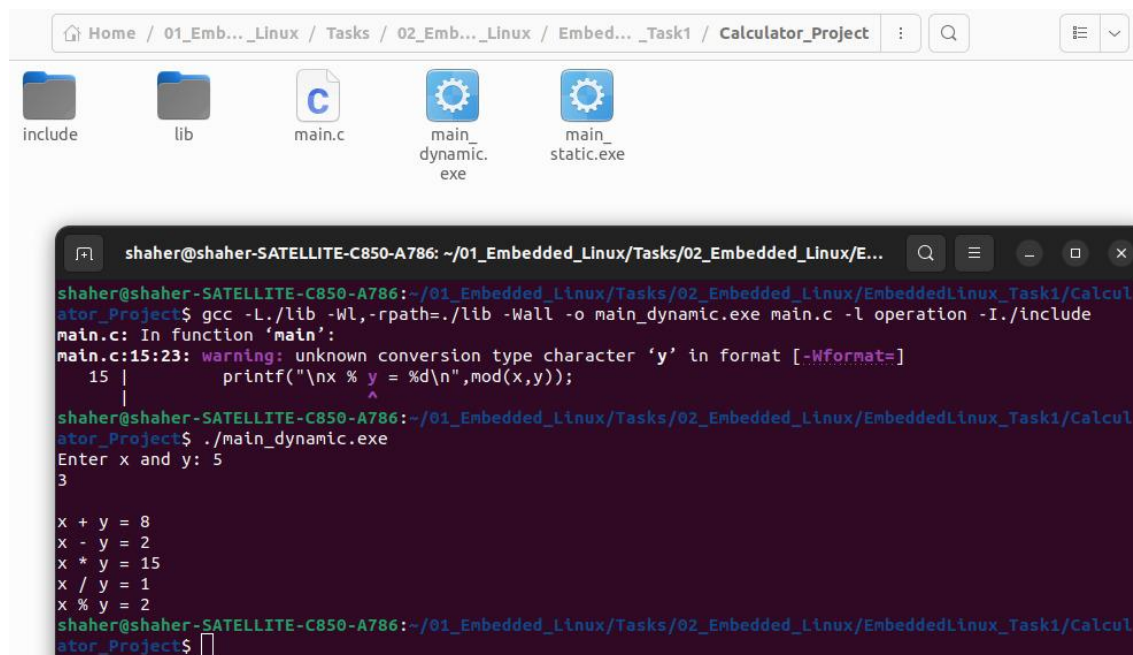
```
shaher@shaher-SATELLITE-C850-A786: ~/01_Embedded_Linux/Tasks/02_Embedded_Linux/Embedded...
shaher@shaher-SATELLITE-C850-A786:~/01_Embedded_Linux/Tasks/02_Embedded_Linux/EmbeddedLinux_Task1/Calculator_Pro
ject$ gcc -o main_static.exe main.c -L ./lib -l operation -I ./include -static
main.c: In function 'main':
main.c:15:23: warning: unknown conversion type character 'y' in format [-Wformat=]
   15 |         printf("\nx % y = %d\n",mod(x,y));
       |                   ^
shaher@shaher-SATELLITE-C850-A786:~/01_Embedded_Linux/Tasks/02_Embedded_Linux/EmbeddedLinux_Task1/Calculator_Pro
ject$ ./main_static.exe
Enter x and y: 5
3
x + y = 8
x - y = 2
x * y = 15
x / y = 1
x % y = 2
shaher@shaher-SATELLITE-C850-A786:~/01_Embedded_Linux/Tasks/02_Embedded_Linux/EmbeddedLinux_Task1/Calculator_Pro
ject$
```

Creating a dynamic library:

First, I created (.o) files for the five functions using -fPIC, then I created “liboperation.so” library dynamically



Then I compiled the (main.c) file linking this dynamic library to it. And then I execute the “main_dynamic.exe” file to test and make sure that everything is fine.



Comparison Steps:

1- Comparing through “ldd” command:

```
shaher@shaher-SATELLITE-C850-A786: ~/01_Embedded_Linux...
shaher@shaher-SATELLITE-C850-A786:~/01_Embedded_Linux/Tasks/02_Embedded_Linux/Em
beddedLinux_Task1/Calculator_Project$ ldd main_dynamic.exe
linux-vdso.so.1 (0x00007ffc7636c000)
liboperation.so => ./lib/liboperation.so (0x00007f55270f6000)
libc.so.6 => /lib/x86_64-linux-gnu/libc.so.6 (0x00007f5526e00000)
/lib64/ld-linux-x86-64.so.2 (0x00007f5527102000)
shaher@shaher-SATELLITE-C850-A786:~/01_Embedded_Linux/Tasks/02_Embedded_Linux/Em
beddedLinux_Task1/Calculator_Project$ ldd main_static.exe
not a dynamic executable
shaher@shaher-SATELLITE-C850-A786:~/01_Embedded_Linux/Tasks/02_Embedded_Linux/Em
beddedLinux_Task1/Calculator_Project$
```

As you see, in case of the dynamically linked file “main_dynamic.exe” the “ldd” command presents the library that this file is using or needing it. But in case of the statically linked file “main_static.exe” it says that it is not dynamically executable, as the “ldd” command is related to the dynamically linked files.

2- Comparing through “file” command:

```
shaher@shaher-SATELLITE-C850-A786: ~/01_Embedded_Linux/Tasks/02_Embedded_Linux/EmbeddedLinux_Task1/Calculator_Project
shaher@shaher-SATELLITE-C850-A786:~/01_Embedded_Linux/Tasks/02_Embedded_Linux/EmbeddedLinux_Task1/Calculator_Project$ file main_dynamic.exe
main_dynamic.exe: ELF 64-bit LSB pie executable, x86-64, version 1 (SYSV), dynamically linked, interpreter /lib64/ld-linux-x86-64.so.2, BuildID[sha1]=
8cfbdf6ac79b171f009d4be949f4e381afd5b754, for GNU/Linux 3.2.0, not stripped
shaher@shaher-SATELLITE-C850-A786:~/01_Embedded_Linux/Tasks/02_Embedded_Linux/EmbeddedLinux_Task1/Calculator_Project$ file main_static.exe
main_static.exe: ELF 64-bit LSB executable, x86-64, version 1 (GNU/Linux), statically linked, BuildID[sha1]=96cab6a435e5471f5fdb355d0eea4bed7cf068fb,
for GNU/Linux 3.2.0, not stripped
shaher@shaher-SATELLITE-C850-A786:~/01_Embedded_Linux/Tasks/02_Embedded_Linux/EmbeddedLinux_Task1/Calculator_Project$
```

As you see, for the dynamically linked file it states that it is dynamically linked, and for the statically linked file it states that it is statically linked 😊.

3- Comparing through “du” command:

```
shaher@shaher-SATELLITE-C850-A786: ~/01_Embedded_Linux/Tasks/02_Embedded_Linux/E...
shaher@shaher-SATELLITE-C850-A786:~/01_Embedded_Linux/Tasks/02_Embedded_Linux/EmbeddedLinux_Task1/Calcul
ator_Project$ du -sh main_static.exe
1012K  main_static.exe
shaher@shaher-SATELLITE-C850-A786:~/01_Embedded_Linux/Tasks/02_Embedded_Linux/EmbeddedLinux_Task1/Calcul
ator_Project$ du -sh main_dynamic.exe
16K    main_dynamic.exe
shaher@shaher-SATELLITE-C850-A786:~/01_Embedded_Linux/Tasks/02_Embedded_Linux/EmbeddedLinux_Task1/Calcul
ator_Project$
```

As we know the benefit from the dynamic linking is in the small size.

4- Comparing using “objdump” command:

```
ator_Project$ objdump -t main_static.exe | grep add.c
0000000000000000 l    df *ABS* 0000000000000000 add.c
shaher@shaher-SATELLITE-C850-A786:~/01_Embedded_Linux/Tasks/02_Embedded_Linux/EmbeddedLinux_Task1/Calcul
ator_Project$ objdump -t main_dynamic.exe | grep add
0000000000000000 F *UND* 0000000000000000 add
```

For the “add” function (as an example), in case of the statically linked file functions or files containing the implementation of the needed function already has “ABS” in front of them.

In contrary, in case of the statically linked file functions or files containing the implementation of the needed function already has “UND” in front of them because these files are linked to the main file during the run-time with the help of the **system loader**.

From the previous comparisons we conclude the following:

- Static libraries work in a way that it is added or “linked” to the executable file during the compiling time, so when we execute the application file it finds it quickly without the need for an aid from the system loader.
- Dynamic libraries work in a way that it is added during the runtime by a help from the system loader.