Simple_Loader Implementation

Launcher.c

- Takes the fib executable as input
- Checks if fib is of the ELF format
- Calls the functions `load_and_run_elf()`(provides it the input file fib) and `loader_cleanup()`, which have their function prototypes in `loader.h` and actual code in `loader.c`

Loader.c

- initialises pointer to struct ehdr and phdr
- Contains the code for functions `load_and_run_elf()` and `loader_cleanup()`

`load_and_run_elf()` :

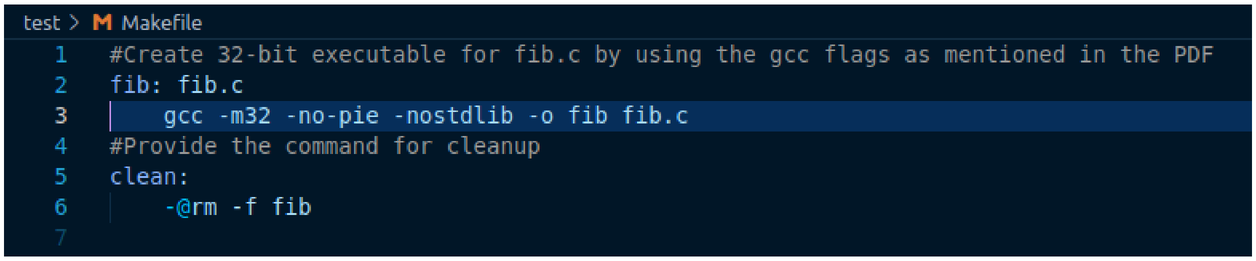
- Opens the fib executable in `O_RDONLY` mode
- Allocates memory to ehdr and phdr corresponding to the sizes of ELF header table and Program Header Table
- Loads the contents of ELF header table and PHT into ehdr and phdr
- Iterates through the contents of the PHT and for each segments of the type `PT_LOAD`, creates a mapping in the virtual memory of the process(with the size of the segment corresponding to the memory image)
- Loads the contents of these segments into the virtual memory mapping
- Creates a function pointer start which points to entry point of the executable(typecasted to `void*`) i.e. `e_entry`.
- result calls the `_start` function and prints the result

`loader_cleanup()`:

- frees up memory allocated to the ehdr and phdr pointers

- closes the previously opened executable

Makefile(test dir)



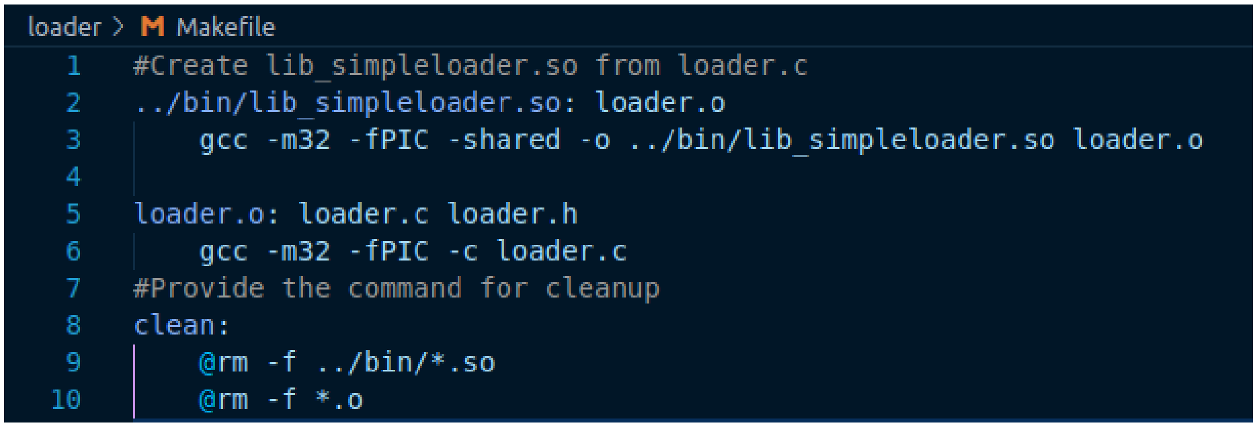
test > Makefile

```
#Create 32-bit executable for fib.c by using the gcc flags as mentioned in the PDF
fib: fib.c
	gcc -m32 -no-pie -nostdlib -o fib fib.c
#Provide the command for cleanup
clean:
	-@rm -f fib

```

- Compiles and links the `fib.c` file to create an executable fib only if there are any changes in fib.c or fib does not exist
- Removes fib executable upon invocation

Makefile(loader dir)



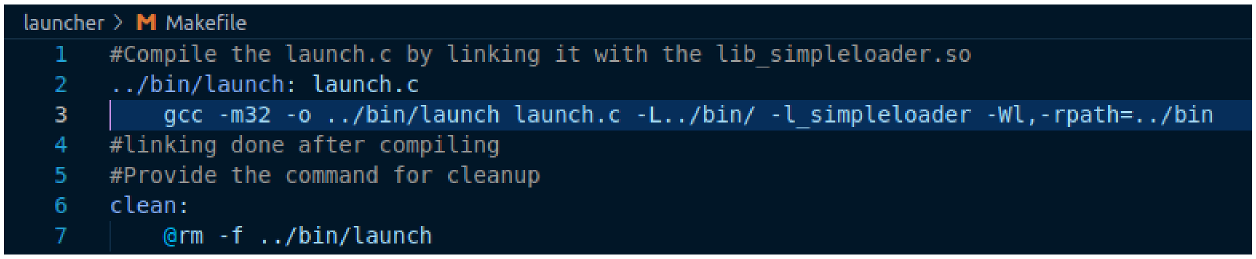
loader > Makefile

```
#Create lib_simpleloader.so from loader.c
../bin/lib_simpleloader.so: loader.o
	gcc -m32 -fPIC -shared -o ../bin/lib_simpleloader.so loader.o

loader.o: loader.c loader.h
	gcc -m32 -fPIC -c loader.c
#Provide the command for cleanup
clean:
	@rm -f ../bin/*.so
	@rm -f *.o
```

- Creates the shared library lib_simpleloader.so in the bin directory if it already doesn't exist or there are any changes in `loader.o`
- Compiles `loader.c` to create `loader.o` with dependencies set as `loader.c` and `loader.h`
- Upon invocation of clean, removes filenames ending with `.o` and `.so`

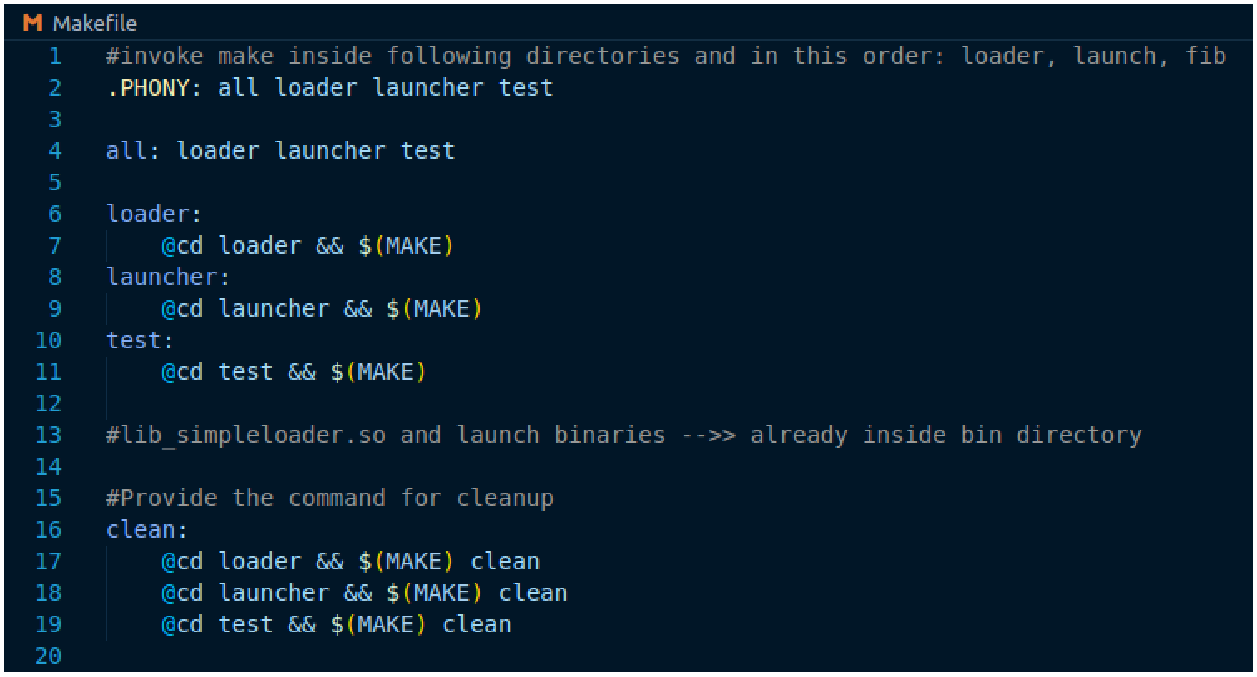
Makefile(launcher dir)



```
#Compile the launch.c by linking it with the lib_simpleloader.so
../bin/launch: launch.c
	gcc -m32 -o ../bin/launch launch.c -L../bin/ -l_simpleloader -Wl,-rpath=../bin
#linking done after compiling
#Provide the command for cleanup
clean:
	@rm -f ../bin/launch
```

- Compiles launch.c and links it with the shared library lib_simpleloader.so present in ../bin/ to create the executable launch in the bin directory
- Removes the launch file on invocation of clean

Overall Makefile



```
#invoke make inside following directories and in this order: loader, launch, fib
.PHONY: all loader launcher test

all: loader launcher test

loader:
	@cd loader && $(MAKE)
launcher:
	@cd launcher && $(MAKE)
test:
	@cd test && $(MAKE)

#lib_simpleloader.so and launch binaries -->> already inside bin directory

#Provide the command for cleanup
clean:
	@cd loader && $(MAKE) clean
	@cd launcher && $(MAKE) clean
	@cd test && $(MAKE) clean

```

- Sets loader, launcher and test as phony targets as they are not filenames
- invokes make command in each of these folders
- upon invocation of clean, it invokes make clean in all of these files

Contributions

Collaborative work: `loader.c` → Function Building, Memory Mapping, Documentation

Raghav : File manipulation, Error handling

Snehil : Makefile , Launcher.c creation

Github Link

https://github.com/SnehilK3372/Group_97_Loader_WithBonus