Assemble and link the provided assembly language file so that it contains debugging information:

- **(1) What command did you use to assemble the assembly language file?**

Start debugging this program using GDB - be sure it is displaying information in the Intel format

student@CIS204-208Ubn22:~/cis206/week5$ nasm -f elf64 -F dwarf homework3.asm

student@CIS204-208Ubn22:~/cis206/week5$ ld homework3.o -o homework3

student@CIS204-208Ubn22:~/cis206/week5$ ls -l

total 20

-rwxrwxr-x 1 student student 10032 Feb 24 20:33 homework3

-rw-rw-r-- 1 student student 930 Feb 24 20:30 homework3.asm

-rw-rw-r-- 1 student student 2368 Feb 24 20:32 homework3.o

- **(2) What command did you use to start the debugging?**

gdb homework3

set disassembly-flavor intel

Set breakpoints at the _start and line 30 of the program

- **(3) Display all existing breakpoints and take a screen snip of the command and the results**

```
(gdb) break _start
Breakpoint 1 at 0x401000: file homework3.asm, line 18.
(gdb) break 30
Breakpoint 2 at 0x401025: file homework3.asm, line 30.
(gdb) info breakpoints
Num     Type           Disp Enb Address            What
1       breakpoint     keep y   0x0000000000401000 homework3.asm:18
2       breakpoint     keep y   0x0000000000401025 homework3.asm:30
(gdb)
```

- **(4) Run the program so it stops at your first breakpoint - what command did you use?**

run

```
(gdb) run
Starting program: /home/student/cis206/week5/homework3

Breakpoint 1, _start () at homework3.asm:18
18          mov rax, 1                  ; Value for A stored in rax
```

- **(5) Step through the next three (3) line one at a time, and take a screen snip of the commands and the results**

```
(gdb) s
19          mov rbx, 2                  ; Value for B stored in rbx
(gdb) s
20          mov rcx, 3                  ; Value for C stored in rcx
(gdb) s
21          mov rdx, 4                  ; Value for D stored in rdx
```

- **(6) Run the command so that your program goes right to the next breakpoint, and take a screen snip of the command, and the results**

```
(gdb) c
Continuing.

Breakpoint 2, _start () at homework3.asm:30
30          mov rax, 60                 ; System call number to exit
```

- **(7) What is the current value of the rax register in hexadecimal?**

```
(gdb) print /x $rax
$1 = 0xfffffffffffffffc
```

The value is 0xfffffffffffffffc

- **(8) Display all the registers using a single command, and take a screen snip of the command and the results**

```
(gdb) info registers
rax            0xfffffffffffffffc  -4
rbx            0x2                 2
rcx            0x7                 7
rdx            0x4                 4
rsi            0x0                 0
rdi            0x0                 0
rbp            0x0                 0x0
rsp            0x7fffffffe080      0x7fffffffe080
r8             0x0                 0
r9             0x0                 0
r10            0x0                 0
r11            0x0                 0
r12            0x0                 0
r13            0x0                 0
r14            0x0                 0
r15            0x0                 0
rip            0x401025            0x401025 <_start+37>
eflags         0x297               [ CF PF AF SF IF ]
cs             0x33                51
ss             0x2b                43
ds             0x0                 0
es             0x0                 0
--Type <RET> for more, q to quit, c to continue without paging--
```