

PAIL

ASSIGNMENTS

Name: Vaibhav Rawat

Class: SY-06

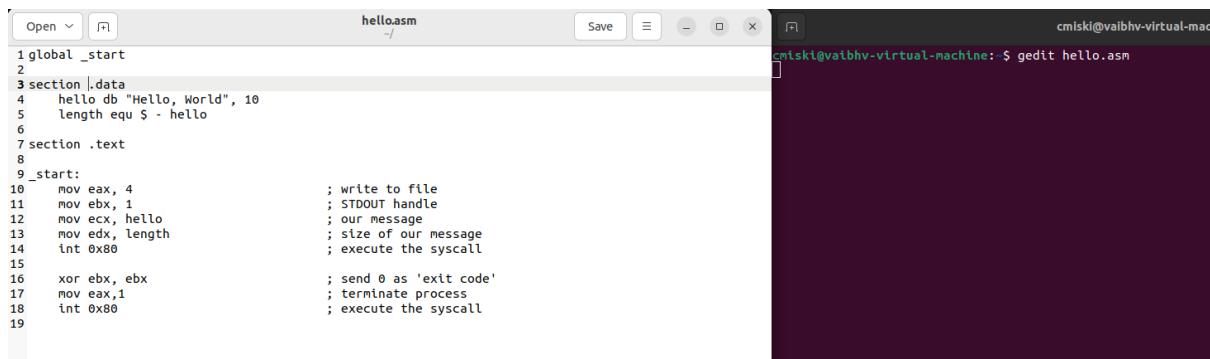
RollNo: 59

EnrollmentNo: ADT24SOCB1344

GitHub Repo: <https://github.com/cmiski/PAIL.git>

Lab-1

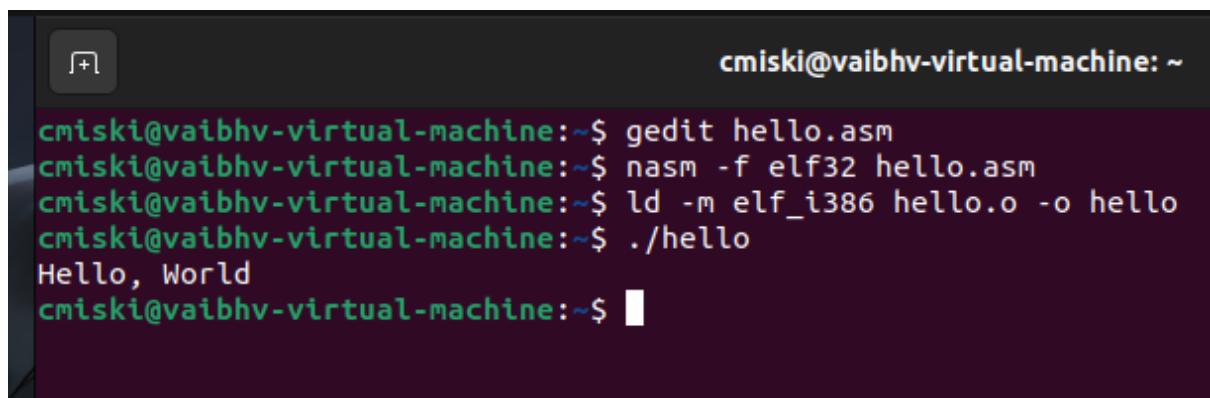
Hello World : A simple assembly program that prints “Hello, World!”.



The screenshot shows a terminal window with two panes. The left pane displays an assembly file named 'hello.asm' containing the following code:

```
1 global _start
2
3 section .data
4     hello db "Hello, World", 10
5     length equ $ - hello
6
7 section .text
8
9 _start:
10    mov eax, 4          ; write to file
11    mov ebx, 1          ; STDOUT handle
12    mov ecx, hello      ; our message
13    mov edx, length      ; size of our message
14    int 0x80            ; execute the syscall
15
16   xor ebx, ebx        ; send 0 as 'exit code'
17   mov eax,1            ; terminate process
18   int 0x80            ; execute the syscall
19
```

The right pane shows the terminal command: `cmiski@vaibhv-virtual-machine:~$ gedit hello.asm`. The assembly code is pasted into the terminal.



The screenshot shows a terminal window with the following session:

```
cmiski@vaibhv-virtual-machine:~$ gedit hello.asm
cmiski@vaibhv-virtual-machine:~$ nasm -f elf32 hello.asm
cmiski@vaibhv-virtual-machine:~$ ld -m elf_i386 hello.o -o hello
cmiski@vaibhv-virtual-machine:~$ ./hello
Hello, World
cmiski@vaibhv-virtual-machine:~$
```

```

cmiski@vaibhv-virtual-machine:~$ gedit hello.asm
cmiski@vaibhv-virtual-machine:~$ nasm -f elf32 -g hello.asm -o hello.o
cmiski@vaibhv-virtual-machine:~$ ld -m elf_i386 hello.o -o hello
cmiski@vaibhv-virtual-machine:~$ ./hello
Hello, World
cmiski@vaibhv-virtual-machine:~$ gdb ./hello
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04.2) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./hello...
(gdb)
(gdb) break _start
Breakpoint 1 at 0x8049000: file hello.asm, line 10.
(gdb) run
Starting program: /home/cmiski/hello

Breakpoint 1, _start () at hello.asm:10
10      mov    eax, 4                      ; write to file
(gdb) set disassembly-flavor intel
(gdb) disassemble _start
Dump of assembler code for function _start:
=> 0x08049000 <_start>:   mov    eax,0x4
  0x08049005 <_start+5>:  mov    ebx,0x1
  0x0804900a <_start+10>: mov    ecx,0x804a000
  0x0804900f <_start+15>: mov    edx,0xd
  0x08049014 <_start+20>: int    0x80
  0x08049016 <_start+22>: xor    ebx,ebx
  0x08049018 <_start+24>: mov    eax,0x1
  0x0804901d <_start+29>: int    0x80
End of assembler dump.
(gdb) layout asm

```

The screenshot shows the GDB graphical interface. At the top, there is a status bar with the text "[Register Values Unavailable]". Below the status bar, the assembly code for the _start function is displayed in a scrollable window. The code consists of several mov and int instructions. The assembly code window has a light blue border. At the bottom of the interface, there is a terminal-like window showing the command history and output.

```

B+> 0x08049000 <_start>:   mov    eax,0x4
  0x08049005 <_start+5>:  mov    ebx,0x1
  0x0804900a <_start+10>: mov    ecx,0x804a000
  0x0804900f <_start+15>: mov    edx,0xd
  0x08049014 <_start+20>: int    0x80
  0x08049016 <_start+22>: xor    ebx,ebx
  0x08049018 <_start+24>: mov    eax,0x1
  0x0804901d <_start+29>: int    0x80
  0x0804901f          add    BYTE PTR [eax],al
  0x08049021          add    BYTE PTR [eax],al
  0x08049023          add    BYTE PTR [eax],al
  0x08049025          add    BYTE PTR [eax],al
  0x08049027          add    BYTE PTR [eax],al
  0x08049029          add    BYTE PTR [eax],al
  0x0804902b          add    BYTE PTR [eax],al
  0x0804902d          add    BYTE PTR [eax],al
  0x0804902f          add    BYTE PTR [eax],al

native process 47625 In: _start
(gdb) layout regs
(gdb)

```

```

Registers group: general
eax    0x1      1          ecx    0x804a000  134520832
ebx    0x0      0          esp    0xfffffd2b0  0xfffffd2b0
est     0x0      0          edi    0x0      0
eflags 0x246   [ PF ZF IF ] cs     0x23      35
ds     0xb      43         es     0x2b      43
gs     0x0      0          edx    0xd      13
ebp    0x0      0          eip    0x804901d  0x804901d < start+29>
ss     0x2b     43         fs     0x0      0
fs     0x0      0          L18   PC: 0

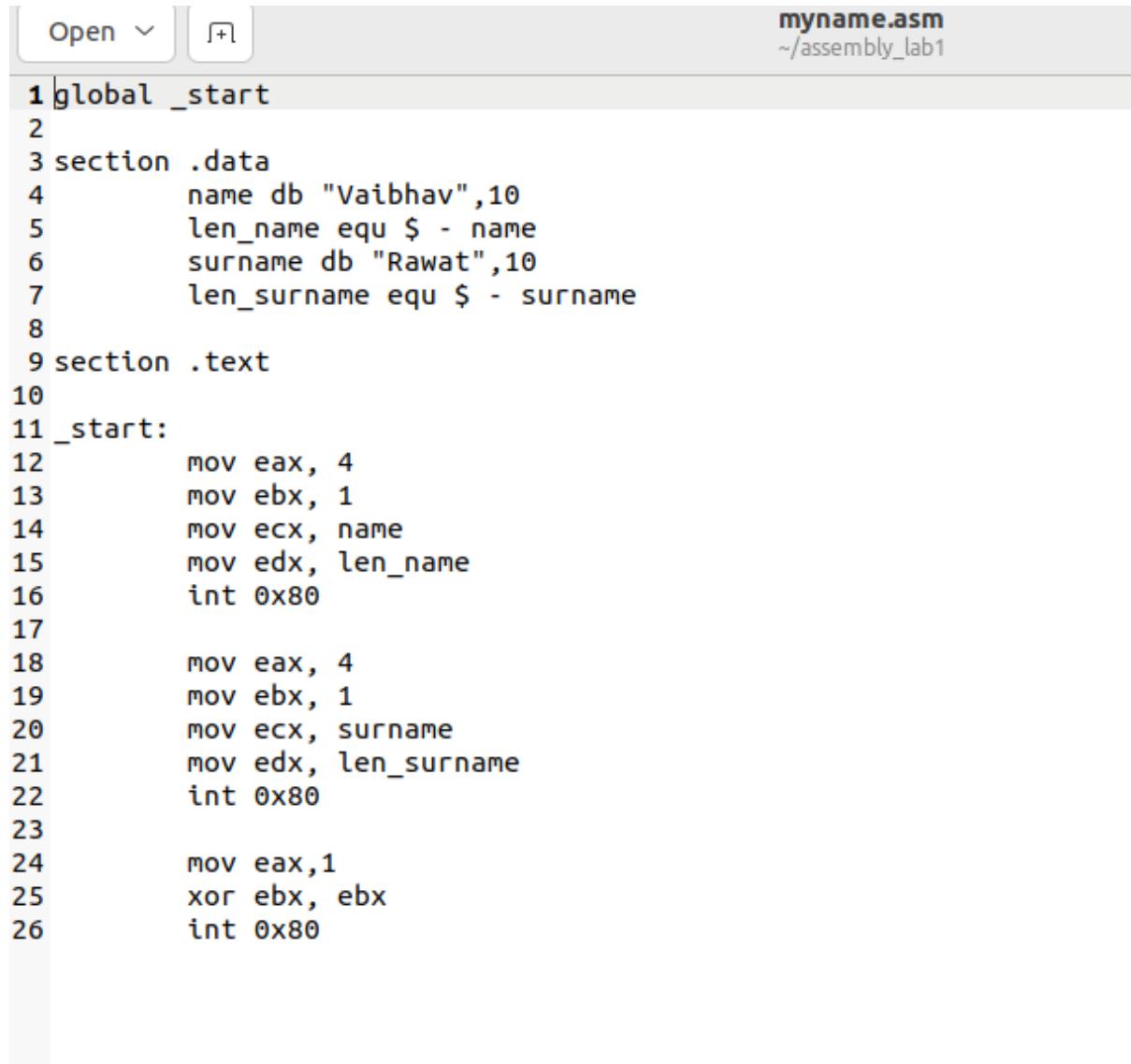
B+ 0x00040000 < start>    mov    eax,0x4
0x00040004 < _start+5>    mov    ebx,0x1
0x00040008 < _start+10>   mov    ecx,0x804a000
0x0004000c < _start+15>   mov    edx,0xd
0x00040010 < _start+20>   xor    ebx,ebx
0x00040014 < _start+24>   mov    eax,0x1
> 0x0004001d < _start+29> int    0x80
0x0004001f add    BYTE PTR [eax],al
0x00040021 add    BYTE PTR [eax],al
0x00040023 add    BYTE PTR [eax],al
0x00040025 add    BYTE PTR [eax],al
0x00040027 add    BYTE PTR [eax],al
0x00040029 add    BYTE PTR [eax],al
0x0004002b add    BYTE PTR [eax],al
0x0004002d add    BYTE PTR [eax],al
0x0004002f add    BYTE PTR [eax],al
native process 47625 In: start
(gdb) layout regs
(gdb) nexti
(gdb) ■
L18 PC: 0

Registers group: general
eax    0x1      1          ecx    0x804a000  134520832
ebx    0x0      0          esp    0xfffffd2b0  0xfffffd2b0
est     0x0      0          edi    0x0      0
eflags 0x246   [ PF ZF IF ] cs     0x23      35
ds     0xb      43         es     0x2b      43
gs     0x0      0          edx    0xd      13
ebp    0x0      0          eip    0x804901d  0x804901d < start+29>
ss     0x2b     43         fs     0x0      0
fs     0x0      0          L18   PC: 0

0x0004001d < start+29> int    0x80
native No process In:
(gdb) layout regs
(gdb) nexti
[Inferior 1 (process 47625) exited normally]
(gdb) quit■
L??
```

Lab-2

Print Name & Surname - Prints my name on the first line and my surname on the second line.



```
myname.asm
~/assembly_lab1

1 global _start
2
3 section .data
4     name db "Vaibhav",10
5     len_name equ $ - name
6     surname db "Rawat",10
7     len_surname equ $ - surname
8
9 section .text
10
11_start:
12     mov eax, 4
13     mov ebx, 1
14     mov ecx, name
15     mov edx, len_name
16     int 0x80
17
18     mov eax, 4
19     mov ebx, 1
20     mov ecx, surname
21     mov edx, len_surname
22     int 0x80
23
24     mov eax,1
25     xor ebx, ebx
26     int 0x80
```



```
cmiski@vaibhv-virtual-machine:~/assembly_lab1$ gedit myname.asm
cmiski@vaibhv-virtual-machine:~/assembly_lab1$ nasm -f elf32 myname.asm
cmiski@vaibhv-virtual-machine:~/assembly_lab1$ ld -m elf_i386 myname.o -o myname
cmiski@vaibhv-virtual-machine:~/assembly_lab1$ ./myname
Vaibhav
Rawat
cmiski@vaibhv-virtual-machine:~/assembly_lab1$
```

```
cmiski@vaibhv-virtual-machine:~/assembly_lab1$ nasm -f elf32 -g myname.asm -o myname.o
cmiski@vaibhv-virtual-machine:~/assembly_lab1$ ld -m elf_i386 myname.o -o myname
cmiski@vaibhv-virtual-machine:~/assembly_lab1$ ./myname
Vaibhav
Rawat
cmiski@vaibhv-virtual-machine:~/assembly_lab1$ gdb ./myname
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04.2) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
  <http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./myname...
(gdb)
(gdb) break _start
Breakpoint 1 at 0x8049000: file myname.asm, line 12.
(gdb) run
Starting program: /home/cmiski/assembly_lab1/myname

Breakpoint 1, _start () at myname.asm:12
12          mov    eax, 4
(gdb) set disassembly-flavor intel
(gdb) disassemble _start
Dump of assembler code for function _start:
=> 0x08049000 <+0>:   mov    eax,0x4
  0x08049005 <+5>:   mov    ebx,0x1
  0x0804900a <+10>:  mov    ecx,0x804a000
  0x0804900f <+15>:  mov    edx,0x8
  0x08049014 <+20>:  int    0x80
  0x08049016 <+22>:  mov    eax,0x4
  0x0804901b <+27>:  mov    ebx,0x1
  0x08049020 <+32>:  mov    ecx,0x804a008
  0x08049025 <+37>:  mov    edx,0x6
  0x0804902a <+42>:  int    0x80
  0x0804902c <+44>:  mov    eax,0x1
  0x08049031 <+49>:  xor    ebx,ebx
  0x08049033 <+51>:  int    0x80
End of assembler dump.
(gdb) layout asm
```

```
B+> 0x8049000 <_start>    mov    eax,0x4
0x8049005 <_start+5>    mov    ebx,0x1
0x804900a <_start+10>   mov    ecx,0x804a000
0x804900f <_start+15>   mov    edx,0x8
0x8049014 <_start+20>   int    0x80
0x8049016 <_start+22>   mov    eax,0x4
0x804901b <_start+27>   mov    ebx,0x1
0x8049020 <_start+32>   mov    ecx,0x804a008
0x8049025 <_start+37>   mov    edx,0x6
0x804902a <_start+42>   int    0x80
0x804902c <_start+44>   mov    eax,0x1
0x8049031 <_start+49>   xor    ebx,ebx
0x8049033 <_start+51>   int    0x80
0x8049035                   add    BYTE PTR [eax],al
0x8049037                   add    BYTE PTR [eax],al
0x8049039                   add    BYTE PTR [eax],al
0x804903b                   add    BYTE PTR [eax],al
0x804903d                   add    BYTE PTR [eax],al
0x804903f                   add    BYTE PTR [eax],al
0x8049041                   add    BYTE PTR [eax],al
0x8049043                   add    BYTE PTR [eax],al
0x8049045                   add    BYTE PTR [eax],al
0x8049047                   add    BYTE PTR [eax],al
0x8049049                   add    BYTE PTR [eax],al
0x804904b                   add    BYTE PTR [eax],al
0x804904d                   add    BYTE PTR [eax],al
0x804904f                   add    BYTE PTR [eax],al
0x8049051                   add    BYTE PTR [eax],al
0x8049053                   add    BYTE PTR [eax],al
0x8049055                   add    BYTE PTR [eax],al
0x8049057                   add    BYTE PTR [eax],al
0x8049059                   add    BYTE PTR [eax],al
0x804905b                   add    BYTE PTR [eax],al
0x804905d                   add    BYTE PTR [eax],al
```

```
native process 48713 In: _start
(gdb) █
```

[Register Values Unavailable]

```
B+> 0x8049000 <_start>    mov    eax,0x4
0x8049005 <_start+5>    mov    ebx,0x1
0x804900a <_start+10>   mov    ecx,0x804a000
0x804900f <_start+15>   mov    edx,0x8
0x8049014 <_start+20>   int    0x80
0x8049016 <_start+22>   mov    eax,0x4
0x804901b <_start+27>   mov    ebx,0x1
0x8049020 <_start+32>   mov    ecx,0x804a008
0x8049025 <_start+37>   mov    edx,0x6
0x804902a <_start+42>   int    0x80
0x804902c <_start+44>   mov    eax,0x1
0x8049031 <_start+49>   xor    ebx,ebx
0x8049033 <_start+51>   int    0x80
0x8049035                   add    BYTE PTR [eax],al
0x8049037                   add    BYTE PTR [eax],al
0x8049039                   add    BYTE PTR [eax],al
0x804903b                   add    BYTE PTR [eax],al
0x804903d                   add    BYTE PTR [eax],al
0x804903f                   add    BYTE PTR [eax],al
0x8049041                   add    BYTE PTR [eax],al
0x8049043                   add    BYTE PTR [eax],al
0x8049045                   add    BYTE PTR [eax],al
0x8049047                   add    BYTE PTR [eax],al
0x8049049                   add    BYTE PTR [eax],al
0x804904b                   add    BYTE PTR [eax],al
0x804904d                   add    BYTE PTR [eax],al
0x804904f                   add    BYTE PTR [eax],al
0x8049051                   add    BYTE PTR [eax],al
0x8049053                   add    BYTE PTR [eax],al
0x8049055                   add    BYTE PTR [eax],al
0x8049057                   add    BYTE PTR [eax],al
0x8049059                   add    BYTE PTR [eax],al
0x804905b                   add    BYTE PTR [eax],al
0x804905d                   add    BYTE PTR [eax],al
```

```
native process 48713 In: start
(gdb) layout regs
(gdb) █
```

```

Register group: general
eax          0x1          1          ccx          0x804a008      134520840
esi          0x0          0          esp          0xfffffd270      0xfffffd270
eflags       0x246        [ PF ZF IF ]      cs           0x0
ds           0x2b         43          es           0x23         35
gs           0x0           0          ss           0x2b         43
                                         es           0x2b         43
                                         fs           0x0           0

0x8049033 <_start+51>    int     0x80

native No process In:
(gdb) layout regs
(gdb) nexti
[Inferior 1 (process 48713) exited normally]
(gdb) quit

```

Lab-3

Addition

1) In registers:

Code:

```

section .text

global _start

_start:
    MOV eax,2
    MOV ebx,9
    ADD eax,ebx
    int 80h

```

Output:

```
cmiski@valbhv-virtual-machine:~/PAIL/lab3$ gedit addition.s
cmiski@valbhv-virtual-machine:~/PAIL/lab3$ nasm -f elf -o addition.o addition.s
cmiski@valbhv-virtual-machine:~/PAIL/lab3$ ld -m elf_i386 -o addition addition.o
cmiski@valbhv-virtual-machine:~/PAIL/lab3$ gdb addition
GNU gdb (Ubuntu 12.1-0ubuntu1-22.04.2) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bug-reporting/>
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from addition...
(No debugging symbols found in addition)
(gdb) layout asm
cmiski@valbhv-virtual-machine:~/PAIL/lab3$
```

```
cmiski@valbhv-virtual-machine:~/PAIL/lab3$ register dump general
eax 0xb 11 ecx 0x0 0 edx 0x0 0
ebx 0x9 9 esp 0xfffffd270 0xfffffd270 ebp 0x0 0
esi 0x0 0 edi 0x0 0 eip 0x804900c < start+12>
eflags 0x202 [ IF ] cs 0x23 35 ss 0x2b 43
ds 0x2b 43 es 0xb 43 fs 0x0 0
gs 0x0 0
```

```
B: 0x0049000 <_start>    mov    eax,0x2
0x0049005 <_start+5>    mov    ebx,0x9
0x0049010 <_start+10>    add    BYTE PTR [eax],al
> 0x0049012 <_start+12>    int    0x80
0x0049014 add    BYTE PTR [eax],al
0x0049016 add    BYTE PTR [eax],al
0x0049018 add    BYTE PTR [eax],al
0x004901a add    BYTE PTR [eax],al
0x004901c add    BYTE PTR [eax],al
0x004901e add    BYTE PTR [eax],al
0x0049020 add    WORD PTR [eax],ax
0x0049022 add    BYTE PTR [eax],al
0x0049024 add    BYTE PTR [eax],al

Disassembly for process 3820 In: start
(gdb) layout regs
(gdb) b _start
Breakpoint 1 at 0x0049000
(gdb) r
starting program: /home/cmiski/PAIL/lab3/addition

Breakpoint 1, 0x00049000 in _start ()
(gdb) s
0x00049005 in _start ()
0x00049006 in _start ()
0x0004900c in _start ()
(gdb)
```

2) Using Variables

Code:

```
section .data

num1 db 2
num2 db 3
result db 0
```

```
newline db 10

section .text
global _start

_start:
    MOV al,[num1]
    ADD al,[num2]

    ADD al,'0'
    MOV [result],al

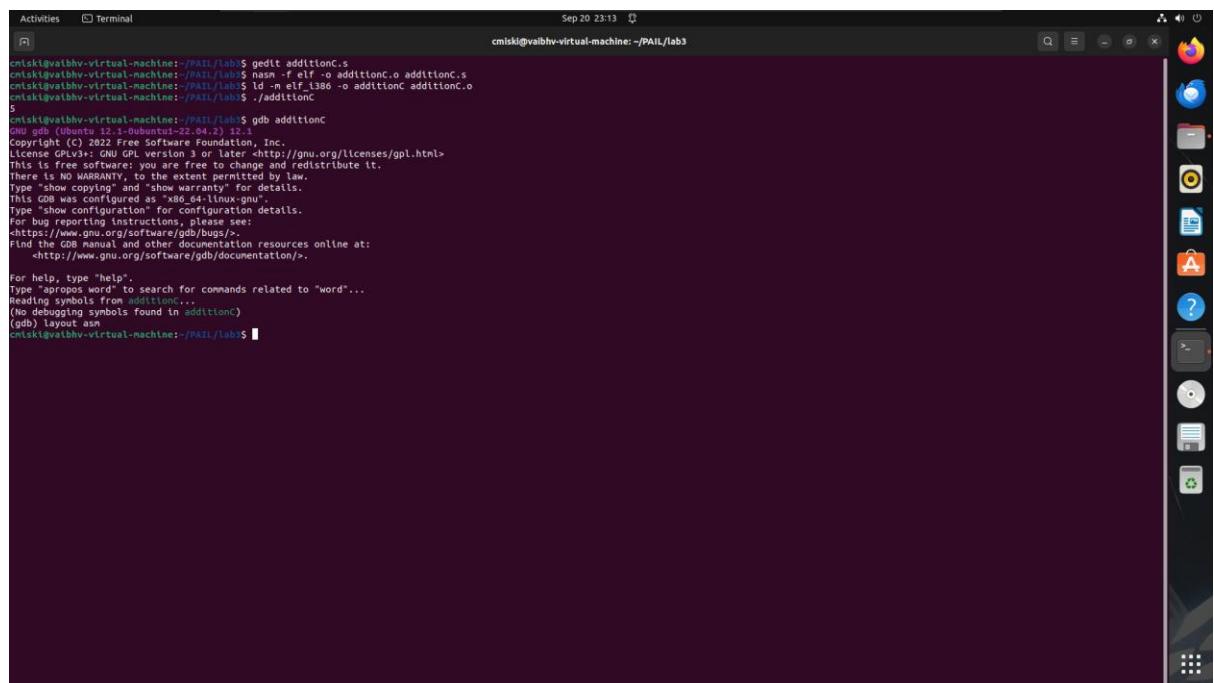
    MOV eax,5
    MOV ebx,1
    MOV ecx,result
    MOV edx,1
    INT 80h

    MOV eax,4
    MOV ebx,1
    MOV ecx,newline
    MOV edx,1
    INT 80h

    MOV eax,1
    XOR ebx,ebx
    INT 80h
```

Output:

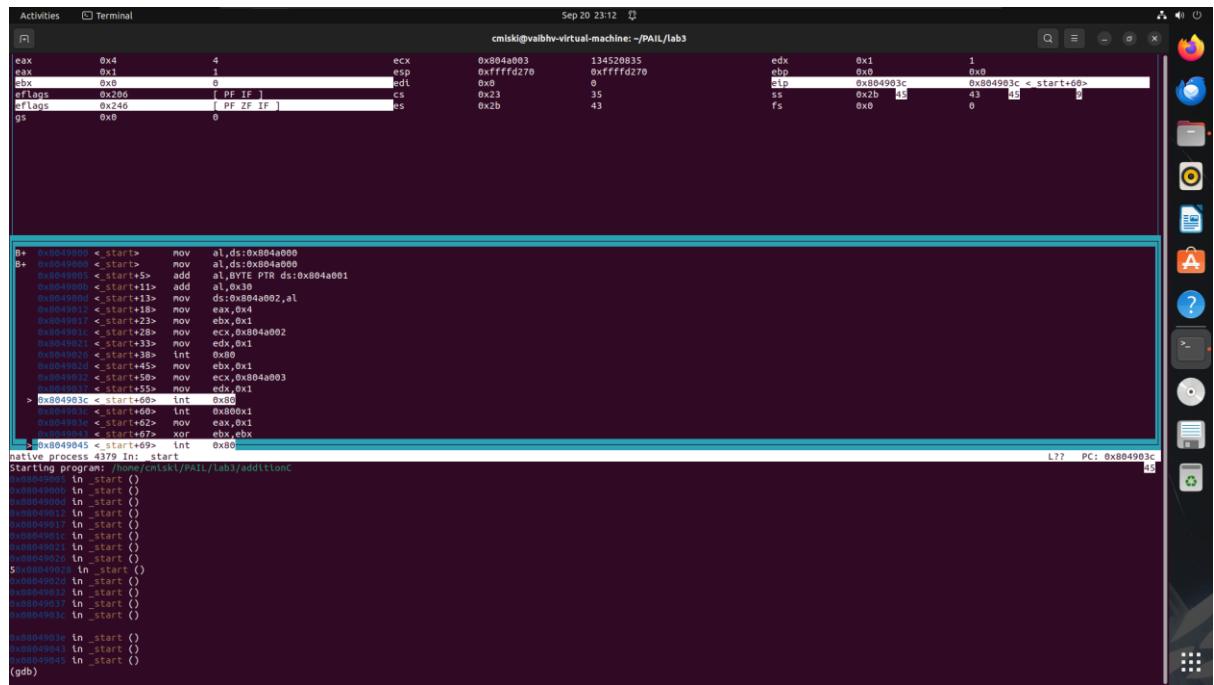
1) Terminal:



```
cmiski@valbhv-virtual-machine:~/PAIL/lab3$ gedit additionC.c
cmiski@valbhv-virtual-machine:~/PAIL/lab3$ gcc -f elf -o additionC.o additionC.c
cmiski@valbhv-virtual-machine:~/PAIL/lab3$ ld -m elf_i386 -o additionC additionC.o
cmiski@valbhv-virtual-machine:~/PAIL/lab3$ ./additionC
5
cmiski@valbhv-virtual-machine:~/PAIL/lab3$ gdb additionC
GNU gdb (Ubuntu 12.1-0ubuntu1-22.04.2) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>
Find the GDB Manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./additionC...
No debugging symbols found in ./additionC
(gdb) layout asm
cmiski@valbhv-virtual-machine:~/PAIL/lab3$
```

2) GDB:



```
cmiski@valbhv-virtual-machine:~/PAIL/lab3$ ./additionC
5
cmiski@valbhv-virtual-machine:~/PAIL/lab3$ gdb ./additionC
Reading symbols from ./additionC...
(gdb) register
Registers on frame 0x10049800:
eax    0x4          4
eax    0x1          1
eax    0x0          0
ebx    0x0          0
gs     0x0          0
eflags 0x246       [ PF IF ] 
eflags 0x146       [ PF IF IF ] 
gs     0x0          0
(gdb) info registers
eax    0x4          4
eax    0x1          1
eax    0x0          0
ebx    0x0          0
gs     0x0          0
eflags 0x246       [ PF IF ] 
eflags 0x146       [ PF IF IF ] 
gs     0x0          0
(gdb) info registers
eax    0x804a003    134520835
eax    0xfffffd270   0xfffffd270
eax    0x0          0
ecx    0x23         35
edx    0x1          1
ebp    0x8049803c   0x8049803c < start+60>
esp    0x8049803c   0x8049803c < start+60>
cs     0x2b         43
ss     0x2b         43
ds     0x0          0
es     0x0          0
fs     0x0          0
(gdb) info registers
eax    0x804a003    134520835
eax    0xfffffd270   0xfffffd270
eax    0x0          0
ecx    0x23         35
edx    0x1          1
ebp    0x8049803c   0x8049803c < start+60>
esp    0x8049803c   0x8049803c < start+60>
cs     0x2b         43
ss     0x2b         43
ds     0x0          0
es     0x0          0
fs     0x0          0
(gdb) disas
Dump of assembler code for function _start:
0x00049800 <_start>:    mov    al,ds:0x804a000
0x00049806 <_start>:    mov    al,ds:0x804a000
0x0004980c <_start+4>:  add    al,al
0x00049810 <_start+8>:  add    al,0x10
0x00049814 <_start+11>: add    al,0x10
0x00049818 <_start+13>: mov    ds:0x804a002,al
0x0004981c <_start+18>: mov    eax,0x4
0x00049820 <_start+23>: mov    ebx,0x1
0x00049824 <_start+27>: mov    edx,0x804a002
0x00049828 <_start+31>: mov    eax,0x1
0x0004982c <_start+35>: int    $0x0
0x00049830 <_start+45>: mov    ebx,0x1
0x00049834 <_start+50>: mov    ecx,0x804a003
0x00049838 <_start+54>: xor    ebx,0x1
0x0004983c <_start+58>: int    $0x80
0x00049840 <_start+60>: int    $0x80
0x00049844 <_start+64>: int    $0x8001
0x00049848 <_start+62>: mov    eax,0x1
0x0004984c <_start+67>: xor    ebx,ebx
0x00049850 <_start+71>: int    $0x80
(gdb) break *0x8049803c
Breakpoint 1 at 0x8049803c: int 0x80
(gdb) run
Starting program: /home/cmiski/PAIL/lab3/additionC
Breakpoint 1, 0x00049803 in _start ()
(gdb) info registers
eax    0x8049803    134520835
eax    0xfffffd270   0xfffffd270
eax    0x0          0
ecx    0x23         35
edx    0x1          1
ebp    0x8049803c   0x8049803c < start+60>
esp    0x8049803c   0x8049803c < start+60>
cs     0x2b         43
ss     0x2b         43
ds     0x0          0
es     0x0          0
fs     0x0          0
(gdb) info registers
eax    0x8049803    134520835
eax    0xfffffd270   0xfffffd270
eax    0x0          0
ecx    0x23         35
edx    0x1          1
ebp    0x8049803c   0x8049803c < start+60>
esp    0x8049803c   0x8049803c < start+60>
cs     0x2b         43
ss     0x2b         43
ds     0x0          0
es     0x0          0
fs     0x0          0
(gdb)
```

3) Taking User Input Code:

```
section .data
```

```
result db 0
newline db 10

section .bss
    input1 RESB 4
    input2 RESB 4

section .text
global _start

_start:
    MOV eax,3
    MOV ebx,0
    MOV ecx,input1
    MOV edx,4
    INT 80h

    MOV eax,3
    MOV ebx,0
    MOV ecx,input2
    MOV edx,4
    INT 80h

    MOV al,[input1]
    SUB al,'0'
    MOV bl,al

    MOV al,[input2]
    SUB al,'0'
    ADD al,bl
```

```
ADD al,'0'  
MOV [result],al
```

```
MOV eax,4  
MOV ebx,1  
MOV ecx,result  
MOV edx,1  
INT 80h
```

```
MOV eax,4  
MOV ebx,1  
MOV ecx,newline  
MOV edx,1  
INT 80h
```

```
MOV eax,1  
XOR ebx,ebx  
INT 80h
```

Output:

1) Terminal:

```
Activities Terminal Sep 21 00:29 cmiski@valbhv-virtual-machine: ~/PAIL/lab3  
cmiski@valbhv-virtual-machine:~/PAIL/lab3$ gedit additionU.s  
cmiski@valbhv-virtual-machine:~/PAIL/lab3$ gcc -f elf -o additionU.o additionU.s  
cmiski@valbhv-virtual-machine:~/PAIL/lab3$ ld -m elf_i386 -o additionU additionU.o  
cmiski@valbhv-virtual-machine:~/PAIL/lab3$ ./additionU  
2  
3  
4  
cmiski@valbhv-virtual-machine:~/PAIL/lab3$ gdb additionU  
GNU gdb (Ubuntu 12.1-0ubuntu1-22.04.2) 12.1  
Copyright (C) 2022 Free Software Foundation, Inc.  
License GPLv3.0+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>  
This software is copyrighted. You are free to copy it, redistribute it.  
There is NO WARRANTY. To the extent permitted by law,  
Type "show copying" and "show warranty" for details.  
This GDB was configured as "x86_64-linux-gnu".  
Type "show configuration" for configuration details.  
For bug reporting, use https://www.gnu.org/software/gdb/bugs/.  
Find the GDB manual and other documentation resources online at:  
<http://www.gnu.org/software/gdb/documentation/>.  
For help, type "help".  
Type "apropos word" to search for commands related to "word"....  
Reading symbols from additionU...  
(No debugging symbols found in additionU)  
(gdb) layout asm  
cmiski@valbhv-virtual-machine:~/PAIL/lab3$
```

2) GDB:

```
Activities Terminal Sep 21 00:28 cmiski@valbhv-virtual-machine: ~/PAIL/lab3  
cmiski@valbhv-virtual-machine:~/PAIL/lab3$  
eax 0x3 3 ecx 0x804a008 134520840 edx 0x4 4  
bx 0x7 7 esp 0xfffffd270 0xffffffffd270 rbp 0x0 0  
bx 0x2 2 ebx 0x1 1 eip 0x804902a <_start+42> r12 0x0 0  
eflags 0x202 [ IF ] cs 0x23 35 ss 0x2b 14 r13 0x0 0  
eflags 0x202 [ IF ] es 0xb 43 r14 0x0 0 r15 0x2 2  
gs 0x0 0  
  
B* 0x0049000 <_start> mov eax,0x3  
0x0049005 <_start+32> mov ecx,0x804a008  
0x004902a <_start+37> mov edx,0x404a004  
0x004902f <_start+42> int 0x80  
0x004902c <_start+44> int al,ds:0x804a004  
0x0049030 <_start+46> add al,bl  
0x0049033 <_start+51> mov bl,al  
0x0049035 <_start+53> mov al,ds:0x804a008  
0x0049037 <_start+58> sub al,0x30  
0x0049039 <_start+5a> add al,cl  
> Bx004903e <_start+62> add al,0x31\004a004  
0x0049045 <_start+64> mov ds:0x804a000,al  
0x0049045 <_start+69> mov eax,0x4  
0x0049045 <_start+74> mov ebx,0x1004a008  
0x0049045 <_start+79> mov ecx,0x804a000  
0x0049045 <_start+84> mov edx,0x404a004  
0x0049045 <_start+89> int 0x80 30  
0x0049045 <_start+91> mov eax,0x4  
native process 5452 In: _start L?? PC: 0x804902a  
(gdb) layout regs  
r0 0x00490010 In: _start ()  
r1 0x00490011 In: _start ()  
r2 0x00490012 In: _start ()  
r3 0x00490013 In: _start ()  
r4 0x00490014 In: _start ()  
r5 0x00490015 In: _start ()  
r6 0x00490016 In: _start ()  
r7 0x00490017 In: _start ()  
r8 0x00490018 In: _start ()  
r9 0x00490019 In: _start ()  
r10 0x0049001a In: _start ()  
r11 0x0049001b In: _start ()  
r12 0x0049001c In: _start ()  
r13 0x0049001d In: _start ()  
r14 0x0049001e In: _start ()  
r15 0x0049001f In: _start ()  
(gdb) [REDACTED]
```

```

Activities Terminal Sep 21 00:29 cmiski@valbhv-virtual-machine: ~/PAIL/lab3
eax 0x3 3 ecx 0x804a008 134520840 edx 0x4 4
eax 0x4 4 ecx 0x804a001 134520833 edx 0x4 4
ebx 0x1 1 edi 0 1
eflags 0x202 [ IF ] es 0x23 35 fs 0x2b 43
eflags 0x202 [ IF ] cs 0x2b 43 gs 0x0 0
gs 0x0 0

```

```

B+ 0x004902a <_start>: mov    eax,0x3
0x004902c <_start+1b>: sub    al,0x20
0x0049030 <_start+1f>: add    al,bl
0x0049034 <_start+60>: add    al,0x30x804a004
0x0049040 <_start+64>: mov    ds:0x804a000,al
0x0049045 <_start+69>: mov    eax,al
0x0049049 <_start+74>: mov    al,0x1004a008
0x004904f <_start+79>: mov    ecc,0x804a000
0x0049054 <_start+84>: mov    edx,0x1
0x0049059 <_start+89>: int    0x80
0x0049063 <_start+91>: mov    eax,0x4
0x0049067 <_start+95>: mov    al,0x1
0x0049065 <_start+101>: mov    ecc,0x804a001
0x004906a <_start+106>: mov    edx,0x1
> 0x004906f <_start+111>: int    0x80
0x0049071 <_start+113>: mov    eax,0x1
0x0049075 <_start+118>: mov    ebx,ebx
0x004907b <_start+120>: int    0x80

```

native process 5452 in: start

(gdb) layout regs

CODE:

A] on registers

CODE:

```

section .data

result db 0

newline db 10

```

```

section .text

```

```

global _start

```

```

_start:

MOV al,7
MOV bl,2
SUB al,bl
ADD al,'0'
MOV [result],al

```

Subtraction

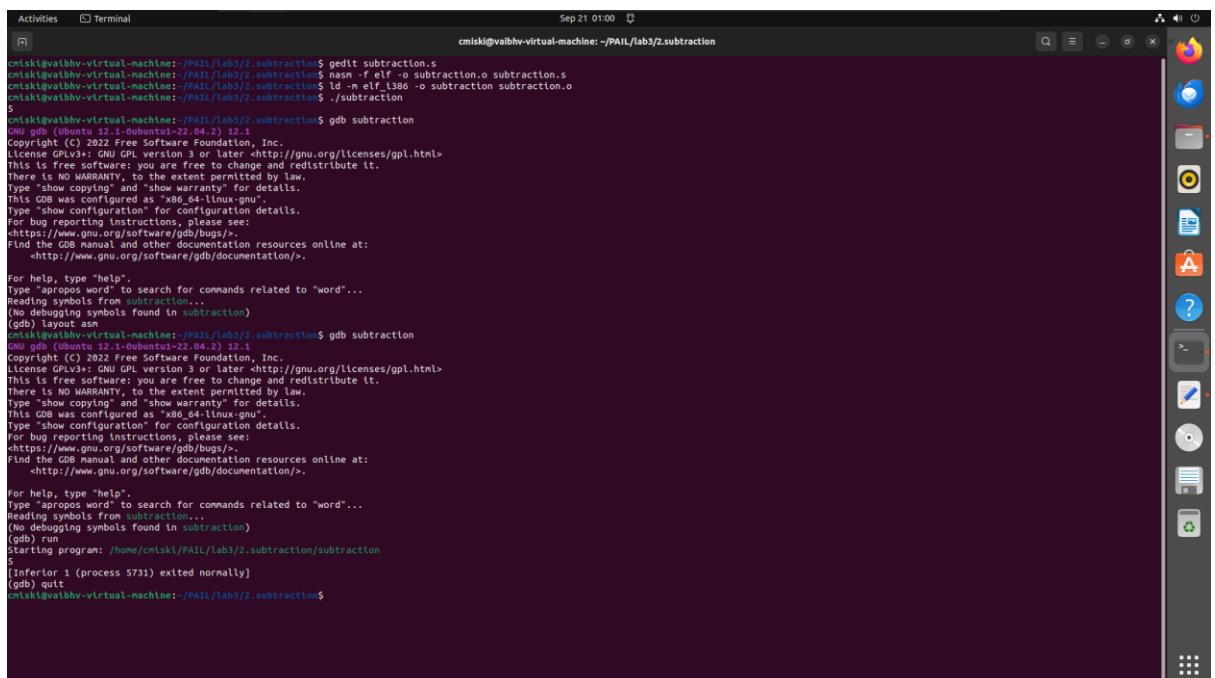
```
MOV eax,4  
MOV ebx,1  
MOV ecx,result  
MOV edx,1  
INT 80h
```

```
MOV eax,4  
MOV ebx,1  
MOV ecx,newline  
MOV edx,1  
INT 80h
```

```
MOV eax,1  
XOR ebx,ebx  
INT 80h
```

OUTPUT:

1. CONSOLE:



The screenshot shows a terminal window titled "Terminal" with the command line "cmiski@vaibhv-virtual-machine: ~/PAIL/lab3/2.subtraction". The terminal displays the assembly code for subtraction and its execution results. The assembly code includes instructions to move values into registers, perform subtraction, and output the result. The execution results show the program outputting the value 1, followed by a newline character.

```
Activities Terminal Sep 21 01:00 cmiski@vaibhv-virtual-machine: ~/PAIL/lab3/2.subtraction  
cmiski@vaibhv-virtual-machine:~/PAIL/lab3/2.subtraction$ gedit subtraction.s  
cmiski@vaibhv-virtual-machine:~/PAIL/lab3/2.subtraction$ nasm -f elf -o subtraction.o subtraction.s  
cmiski@vaibhv-virtual-machine:~/PAIL/lab3/2.subtraction$ ld -m elf_i386 -o subtraction subtraction.o  
cmiski@vaibhv-virtual-machine:~/PAIL/lab3/2.subtraction$ ./subtraction  
MOV eax,4  
MOV ebx,1  
MOV ecx,result  
MOV edx,1  
INT 80h  
  
MOV eax,4  
MOV ebx,1  
MOV ecx,newline  
MOV edx,1  
INT 80h  
  
MOV eax,1  
XOR ebx,ebx  
INT 80h
```

2. GDB:

```
Register group: general
eax 0x0 0 ecx 0x0 0 edx 0x0 0
ebx 0x0 0 esp 0xfffffd230 0xfffffd230
esi 0x0 0 edi 0x0 0 ebp 0x0 0
eflags 0x202 [ IF ] cs 0x23 35 rip 0x8049000 <_start>
ds 0xb 43 es 0xb 43 ss 0xb 43
gs 0xb 0 fs 0x0 0

0x8049000 < start>: mov al,0x7
0x8049002 < start+2>: mov bl,0x2
0x8049004 < start+4>: sub al,bl
0x8049006 < start+6>: add al,0x10
0x8049008 < start+8>: mov ds:0x804a000,al
0x804900d < start+13>: mov eax,0x4
0x8049012 < start+18>: mov ebx,0x1
0x8049017 < start+23>: mov ecx,0x804a000
0x804901c < start+28>: mov edx,0x1
0x8049021 < start+33>: int 0x0
0x8049023 < start+35>: mov eax,0x0
0x8049028 < start+40>: mov ebx,0x1
0x804902d < start+45>: mov ecx,0x804a001
0x8049032 < start+50>: mov edx,0x1
0x8049037 < start+55>: int 0x0
0x8049039 < start+57>: mov eax,0x1
0x804903e < start+62>: xor ebx,ebx

(gdb) layout regs
(gdb) b *start
Breakpoint 1 at 0x8049000
(gdb) r
Starting program: /home/cmiski/PAIL/lab3/2.subtraction/subtraction
Breakpoint 1, 0x00049000 in _start ()
(gdb)
```

```
Register group: general
eax 0x5 5 ecx 0x0 0 edx 0x0 0
ebx 0x2 2 esp 0xfffffd230 0xfffffd230
esi 0x0 0 edi 0x0 0 ebp 0x0 0
eflags 0x206 [ PF IF ] cs 0x23 35 rip 0x8049000 < start+6>
ds 0xb 43 es 0xb 43 ss 0xb 43
gs 0xb 0 fs 0x0 0

0x8049000 < start>: mov al,0x7
0x8049002 < start+2>: mov bl,0x2
0x8049004 < start+4>: sub al,bl
0x8049006 < start+6>: add al,0x10
0x8049008 < start+8>: mov ds:0x804a000,al
0x804900d < start+13>: mov eax,0x4
0x8049012 < start+18>: mov ebx,0x1
0x8049017 < start+23>: mov ecx,0x804a000
0x8049021 < start+28>: int 0x0
0x8049023 < start+33>: mov edx,0x1
0x8049028 < start+35>: mov eax,0x4
0x804902d < start+40>: mov ebx,0x1
0x8049032 < start+45>: mov ecx,0x804a001
0x8049037 < start+50>: int 0x0
0x8049039 < start+55>: mov eax,0x1
0x804903e < start+62>: xor ebx,ebx

(gdb) layout regs
(gdb) b *start
Breakpoint 1 at 0x8049000
(gdb) r
Starting program: /home/cmiski/PAIL/lab3/2.subtraction/subtraction
Breakpoint 1, 0x00049000 in _start ()
(gdb) s
x00049002 in _start()
x00049004 in _start()
x00049006 in _start()
(gdb)
```

```

Activities Terminal Sep 21 00:59 cmiski@valibhv-virtual-machine: ~/PAUL/lab3/2.subtraction
eax 0x4 4 ecx 0x804a001 134526833 edx 0x1 1
ebx 0x1 1 esp 0xfffffd230 eip 0x8049037 0xb
ecx 0x0 0 ebp 0x0 0
eflags 0x286 [ PF IF ] cs 0x23 35 ss 0x2b 13
eflags 0x246 [ PF ZF IF ] gs 0x2b 43 ds 0x2b 10
gs 0x0 0 fs 0x0 0 es 0x0 0

```

```

B+ 0x8049000 <_start>    MOV    al,0x7
                           0x4
                           04a001
>
                           0x1
[ No Assembly Available ]

```

```

native process 5719 In: _start
No process In:
0x8049006 ln _start()
0x804900c ln _start()
0x80490ed ln _start()
0x8049012 ln _start()
0x8049017 ln _start()
0x804901c ln _start()
0x8049021 ln _start()
0x8049026 ln _start()
0x804902b ln _start()
0x804902d ln _start()
0x8049032 ln _start()
0x8049037 ln _start()
0x8049039 ln _start()
0x804903e ln _start()
0x8049040 ln _start()
[Inferior 1 (process 5719) exited normally]
(gdb)

```

B] Using User Inputs

CODE:

```

section .data
    result db 0
    newline db 10

section .bss
    input1 RESB 4
    input2 RESB 4

section .text
global _start

_start:
    MOV eax,3
    MOV ebx,0
    MOV ecx,input1
    MOV edx,4
    INT 80h

```

```
MOV eax,3  
MOV ebx,0  
MOV ecx,input2  
MOV edx,4  
INT 80h
```

```
MOV al,[input1]  
SUB al,'0'  
MOV bl,al
```

```
MOV al,[input2]  
SUB al,'0'  
SUB bl,al
```

```
ADD bl,'0'  
MOV [result],bl
```

```
MOV eax,4  
MOV ebx,1  
MOV ecx,result  
MOV edx,1  
INT 80h
```

```
MOV eax,4  
MOV ebx,1  
MOV ecx,newline  
MOV edx,1  
INT 80h
```

```
MOV eax,1  
XOR ebx,ebx
```

INT 80h

OUTPUT

1. Terminal:

```
Activities Terminal Sep 21 01:29 cmiski@vibhv-virtual-machine: ~/PAIL/lab3/2.subtraction
cmiski@vibhv-virtual-machine:~/PAIL/lab3/2.subtraction$ gedit subtractionU.s
cmiski@vibhv-virtual-machine:~/PAIL/lab3/2.subtraction$ nasm -f elf -o subtractionU.o subtractionU.s
cmiski@vibhv-virtual-machine:~/PAIL/lab3/2.subtraction$ ld -m elf_i386 -o subtractionU subtractionU.o
cmiski@vibhv-virtual-machine:~/PAIL/lab3/2.subtraction$ ./subtractionU
2
5
cmiski@vibhv-virtual-machine:~/PAIL/lab3/2.subtraction$ gdb subtractionU
GNU gdb (Ubuntu 12.1-0ubuntu1-22.04.2) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from subtractionU...
(No debugging symbols found in subtractionU)
(gdb) layout asm
cmiski@vibhv-virtual-machine:~/PAIL/lab3/2.subtraction$ gdb subtractionU
GNU gdb (Ubuntu 12.1-0ubuntu1-22.04.2) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from subtractionU...
(No debugging symbols found in subtractionU)
(gdb) run
Starting program: /home/cmiski/PAIL/lab3/2.subtraction/subtractionU
7
2
5
Inferior 1 (process 5996) exited normally
(gdb) quit
cmiski@vibhv-virtual-machine:~/PAIL/lab3/2.subtraction$
```

2. GDB:

```
Activities Terminal Sep 21 01:27 cmiski@vibhv-virtual-machine: ~/PAIL/lab3/2.subtraction
Register group: general
eax      0x0          0          ecx      0x0          0          edx      0x0          0
ebx      0x0          0          esp      0xfffffd230  0xfffffd230  ebp      0x0          0
esi      0x0          0          edi      0x0          0          eip      0x8049000 <_start>
eflags   0x202       [ IF ]    cs      0x23          35          ss      0x2b          43
ds       0xb2          43         es      0x2b          43          fs      0x0          0
gs       0x0          0

0x8049000 < start>:  mov    eax,0x3
0x8049004 < start+4>:  mov    eax,0x0
0x8049008 < start+8>:  mov    ccx,0x80aa004
0x804900c < start+12>: mov    edx,0x4
0x8049010 < start+16>: int    0xb8
0x8049014 < start+20>: mov    eax,0x3
0x8049018 < start+22>: mov    eax,0x3
0x804901c < start+24>: mov    ebx,0x0
0x8049020 < start+28>: mov    ccx,0x80aa008
0x8049024 < start+32>: mov    edx,0x4
0x8049028 < start+36>: int    0xb8
0x8049032 < start+40>: mov    al,0x30
0x8049036 < start+44>: sub    al,0x30
0x804903a < start+48>: mov    bl,0x0
0x804903e < start+52>: mov    al,ds:0x804a008
0x8049042 < start+56>: sub    al,0x30
0x8049046 < start+60>: sub    bl,al
0x804904a < start+62>: add    bl,0x30

bative process 5996 Int: start
(gdb) layout regs
(gdb) b _start
Breakpoint 1 at 0x8049000
(gdb) r
Starting program: /home/cmiski/PAIL/lab3/2.subtraction/subtractionU
Breakpoint 1, 0x00049000 in _start ()
(gdb)
```

Activities Terminal Sep 21 01:27 cmiski@valbhv-virtual-machine: ~/PAIL/lab3/2.subtraction

eax	0x3	3	ecx	0x804a008	134520840	edx	0x4	4
eax	0x2	2	esp	0xffffd230	0xffffd230	ebp	0x0	0x0
ebx	0x5	5	edi	0x0	0	clp	0x04902a	0x804902a <_start+42>
eflags	0x202	[IF]	esi	0x23	35	st	0x2b	43
eflags	0x206	[PF IF]	es	0x2b	43	fs	0x0	0
gs	0x0	6						

```
B+ 0x0010000 <_start> mov eax,0x3
0x0010003 <_start+3> mov ecx,0x804a008
0x0010032 <_start+37> mov edx,0x404a004
0x0010072 <_start+42> int 0x80
0x0010072 <_start+44> int al,ds:0x804a004
0x0010093 <_start+49> sub al,0x30
0x0010093 <_start+51> mov bl,al
0x0010093 <_start+53> mov al,ds:0x804a008
0x0010093 <_start+58> sub al,0x30
0x0010093 <_start+60> sub bl,al
> 0x0010093 <_start+62> add bl,0x30<0x4a004
0x0010093 <_start+63> mov ds,0x804a000,bl
0x0010093 <_start+71> mov eax,0x4
0x0010094 <_start+76> mov ebx,0x10<0x804a008
0x0010095 <_start+81> mov ecx,0x804a000
0x0010095 <_start+86> mov edx,0x1
0x0010095 <_start+91> int 0x80 30
0x0010095 <_start+93> mov eax,0x4
```

native process 5968 In: start
(gdb) layout regs
0x0049005 tn _start ()
0x0049006 tn _start ()
0x004900f tn _start ()
0x0049014 tn _start ()
7
0x004901b tn _start ()
0x0049020 tn _start ()
0x0049025 tn _start ()
0x004902a tn _start ()
2
0x004902c tn _start ()
0x0049031 tn _start ()
0x0049033 tn _start ()
0x0049035 tn _start ()
0x004903a tn _start ()
0x004903e tn _start ()
(gdb)

Activities Terminal Sep 21 01:28 cmiski@valbhv-virtual-machine: ~/PAIL/lab3/2.subtraction

eax	0x4	4	ecx	0x804a001	134520833	edx	0x1	1
eax	0x1	1	esp	0x0	0	ebp	0x04902a	0x804902a <_start+42>
ebx	0x8	8	edi	0x23	35	clp	0x04902a	0x804902a <_start+42>
eflags	0x286	[PF IF]	esi	0x2b	43	st	0x2b	43
eflags	0x246	[PF ZF IF]	es	0x2b	43	fs	0x0	0

```
B+ 0x0010000 <_start> MOV eax,0x3
0x0010030 <_start+58> SUB al,0x30
0x0010093 <_start+91> INT 0x80 30
0x0010093 <_start+93> MOV eax,0x4
```

[No Assembly Available]
nat
L7? PC: 0x804902a

(gdb) layout regs
0x0049004 No process In:
0x0049005 tn _start ()
0x0049006 tn _start ()
0x0049007 tn _start ()
0x0049008 tn _start ()
0x0049009 tn _start ()
0x0049010 tn _start ()
0x0049011 tn _start ()
0x0049012 tn _start ()
0x0049013 tn _start ()
0x0049014 tn _start ()
0x0049015 tn _start ()
0x0049016 tn _start ()
0x0049017 tn _start ()
0x0049018 tn _start ()
0x0049019 tn _start ()
0x0049020 tn _start ()
0x0049021 tn _start ()
0x0049022 tn _start ()
0x0049023 tn _start ()
0x0049024 tn _start ()
0x0049025 tn _start ()
0x0049026 tn _start ()
0x0049027 tn _start ()
0x0049028 tn _start ()
0x0049029 tn _start ()
0x0049030 tn _start ()
0x0049031 tn _start ()
0x0049032 tn _start ()
0x0049033 tn _start ()
0x0049034 tn _start ()
0x0049035 tn _start ()
0x0049036 tn _start ()
0x0049037 tn _start ()
0x0049038 tn _start ()
0x0049039 tn _start ()
0x0049040 tn _start ()
0x0049041 tn _start ()
0x0049042 tn _start ()
0x0049043 tn _start ()
0x0049044 tn _start ()
0x0049045 tn _start ()
0x0049046 tn _start ()
0x0049047 tn _start ()
0x0049048 tn _start ()
0x0049049 tn _start ()
0x0049050 tn _start ()
0x0049051 tn _start ()
0x0049052 tn _start ()
0x0049053 tn _start ()
0x0049054 tn _start ()
0x0049055 tn _start ()
0x0049056 tn _start ()
0x0049057 tn _start ()
0x0049058 tn _start ()
0x0049059 tn _start ()
0x0049060 tn _start ()
0x0049061 tn _start ()
0x0049062 tn _start ()
0x0049063 tn _start ()
0x0049064 tn _start ()
0x0049065 tn _start ()
0x0049066 tn _start ()
0x0049067 tn _start ()
0x0049068 tn _start ()
0x0049069 tn _start ()
0x0049070 tn _start ()
0x0049071 tn _start ()
0x0049072 tn _start ()
0x0049073 tn _start ()
0x0049074 tn _start ()
0x0049075 tn _start ()
0x0049076 tn _start ()
0x0049077 tn _start ()
0x0049078 tn _start ()
0x0049079 tn _start ()
[Inferior 1 (process 5968) exited normally]
(gdb)

MULTIPLICATION

A] on registers

Code:

section .data

resmsg db "Result: ", 0

```
lenr equ $-resmsg  
newline db 0xA, 0
```

```
section .bss  
    result resb 12
```

```
section .text  
    global _start
```

```
_start:  
    mov eax, 29  
    mov ebx, 29  
    imul eax, ebx  
  
    mov edi, result + 11  
    mov byte [edi], 0
```

```
.convert_loop:  
    xor edx, edx  
    mov ecx, 10  
    div ecx  
    add dl, '0'  
    dec edi  
    mov [edi], dl  
    test eax, eax  
    jnz .convert_loop
```

```
    mov eax, 4  
    mov ebx, 1  
    mov ecx, resmsg  
    mov edx, lenr  
    int 0x80
```

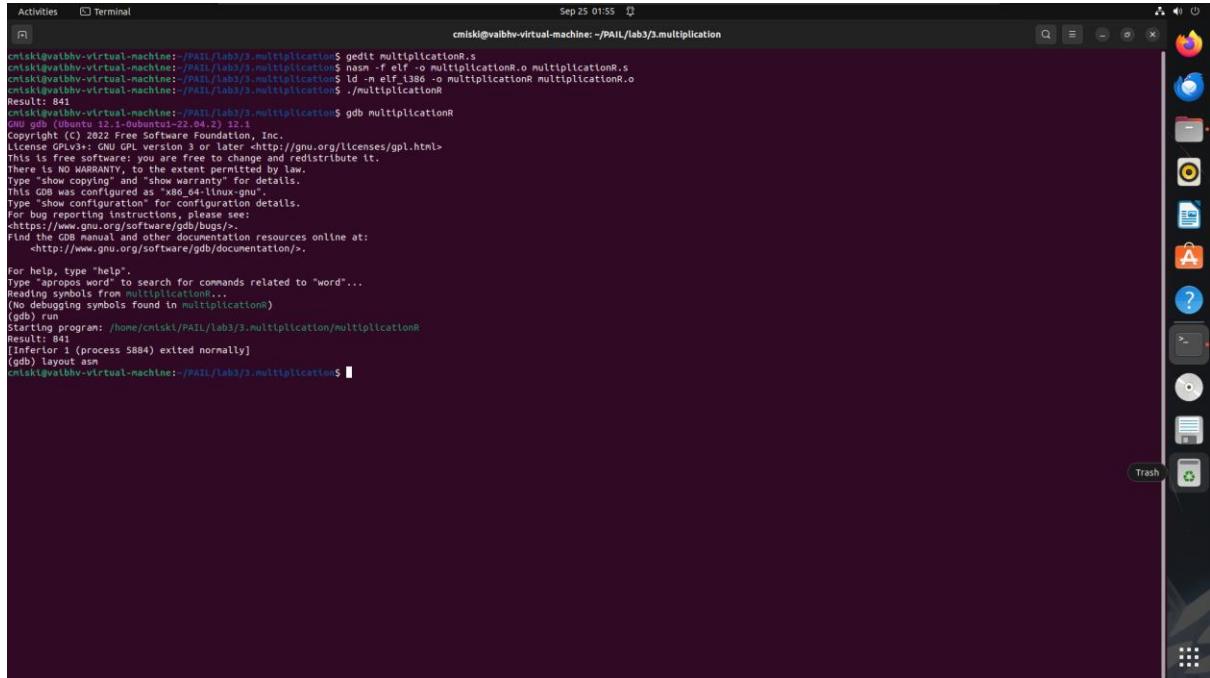
```
mov eax, 4
mov ebx, 1
mov ecx, edi
mov edx, result + 11
sub edx, edi
int 0x80
```

```
mov eax, 4
mov ebx, 1
mov ecx, newline
mov edx, 1
int 0x80
```

```
mov eax, 1
xor ebx, ebx
int 0x80
```

Output:

1]Console



The screenshot shows a terminal window titled "Terminal" with the command-line interface of a Linux system. The session starts with the user navigating to their home directory and cloning a repository named "PAIL". The user then moves into the "lab3" directory and runs the command "multiplicationR". This triggers a build process using "gcc" to compile "multiplication.c" into "multiplicationR", followed by "nasm" to assemble "multiplication.s" into "multiplication.o", and finally "ld" to link "multiplicationR" into "multiplicationR.o". The final output is "multiplicationR". The user then runs "gdb" on "multiplicationR" and attaches it to the running process with PID 841. The GDB prompt "(gdb)" is visible at the bottom of the terminal window.

```
Activities Terminal Sep 25 01:55 cmiski@vaibhv-virtual-machine: ~/PAIL/lab3/3.multiplication
cmiski@vaibhv-virtual-machine:~/PAIL/lab3/3.multiplication$ gedit multiplicationR.s
cmiski@vaibhv-virtual-machine:~/PAIL/lab3/3.multiplication$ nasm -f elf -o multiplicationR.o multiplicationR.s
cmiski@vaibhv-virtual-machine:~/PAIL/lab3/3.multiplication$ ld -m elf_i386 -o multiplicationR multiplicationR.o
multiplicationR:~/PAIL/lab3/3.multiplication$ ./multiplicationR
Result: 841
cmiski@vaibhv-virtual-machine:~/PAIL/lab3/3.multiplication$ gdb multiplicationR
GNU gdb (Ubuntu 12.1-0ubuntu1-22.04.2) 12.1
Copyright (C) 2021 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from multiplicationR...
(No debugging symbols found in multiplicationR)
(gdb) 
starting program: /home/cmiski/PAIL/lab3/3.multiplication/multiplicationR
Result: 841
[Inferior 1 (process 5884) exited normally]
(gdb) layout asm
cmiski@vaibhv-virtual-machine:~/PAIL/lab3/3.multiplication$
```

2]GDB

The image shows two vertically stacked GDB sessions running on a Linux desktop environment. Both sessions are connected to the same process (native process 5887) and show the assembly code for the multiplication program. The assembly code includes instructions for moving values between registers (eax, ebx, ecx, edx, esp, etc.), performing arithmetic operations like addition and multiplication, and jumping between loops. The registers pane at the top of each GDB window displays the current state of the CPU registers for each session. The bottom pane of each window shows the assembly code with line numbers and labels like '_start' and '_start+13'. The desktop background features a dark theme with various application icons visible on the right side.

```
B+ 0x0049000 <_start>      mov  eax,0x1d
0x0049005 <_start+5>      mov  ebx,0x1d
> 0x004900a <_start+10>    imul  eax,ebx
0x0049012 <_start+18>    mov  BYTE PTR [edi],0x0
0x0049014 <_start.convert_loop>    add  eax,0x10
0x0049017 <_start.convert_loop+2>  mov  ebx,0x1a
0x004901c <_start.convert_loop+7>  div  eax,ebx
0x0049021 <_start.convert_loop+12> dec  edx
0x0049022 <_start.convert_loop+15> test  eax,edx
0x0049024 <_start.convert_loop+17> jne  0x0049015 <_start.convert_loop>
0x0049026 <_start.convert_loop+19> mov  eax,0x4
0x004902d <_start.convert_loop+24> mov  ebx,0x1
0x0049032 <_start.convert_loop+29> mov  ecx,0x804a000
0x0049037 <_start.convert_loop+34> mov  edx,0x9

native process 5887 In: _start
(gdb) layout regs
(gdb) b _start
Breakpoint 1 at 0x0049000
(gdb) r
Starting program: /home/cmiski/PAIL/lab3/3.multiplication/multiplicationR

Breakpoint 1, 0x0049000 in _start ()
(gdb) s
x0x0049005 in _start ()
x0x004900a in _start ()
(gdb)

Activities Terminal Sep 25 01:54 cmiski@valbh-virtual-machine: ~/PAIL/lab3/3.multiplication L?? PC: 0x004900a
Register group: general
eax 0x1d 29 ecx 0x0 0 edx 0x0 0
ebx 0x1d 29 esp 0xfffffd220 ebp 0x0 0
est 0x0 0 edi 0x0 0 ss 0x2b 43
eflags 0x202 [ IF ] cs 0x23 35 fs 0x0 0
ds 0x2b 43 es 0x2b 43
gs 0x0 0 mov  eax,0x1d
mov  ebx,0x1d
imul  eax,ebx
mov  edi,0x804a017
mov  BYTE PTR [edi],0x0
add  eax,0x10
mov  ebx,0x1a
div  eax,ebx
dec  edx
test  eax,edx
jne  0x0049015 <_start.convert_loop>
mov  eax,0x4
mov  ebx,0x1
mov  ecx,0x804a000
mov  edx,0x9

native process 5887 In: _start
(gdb) layout regs
(gdb) b _start
Breakpoint 1 at 0x0049000
(gdb) r
Starting program: /home/cmiski/PAIL/lab3/3.multiplication/multiplicationR

Breakpoint 1, 0x0049000 in _start ()
(gdb) s
x0x0049005 in _start ()
x0x004900a in _start ()
x0x004900d in _start ()
(gdb)
```

B] taking user inputs

Code:

```
section .data
```

```
msg1 db "Enter first number: ", 0
len1 equ $-msg1
msg2 db "Enter second number: ", 0
len2 equ $-msg2
```

```
resmsg db "Result: ", 0
lenr equ $-resmsg
newline db 0xA, 0
```

```
section .bss
num1 resb 12
num2 resb 12
result resb 24
```

```
section .text
global _start
```

```
_start:
    mov eax, 4
    mov ebx, 1
    mov ecx, msg1
    mov edx, len1
    int 0x80
```

```
    mov eax, 3
    mov ebx, 0
    mov ecx, num1
    mov edx, 12
    int 0x80
```

```
    mov esi, num1
    xor eax, eax
```

```
.convert1:
```

```
    mov bl, [esi]
    cmp bl, 0xA
    je .done1
    cmp bl, 0
```

```
je .done1
sub bl, '0'
imul eax, eax, 10
add eax, ebx
inc esi
jmp .convert1

.done1:
    mov ebx, eax

    mov eax, 4
    mov ebx, 1
    mov ecx, msg2
    mov edx, len2
    int 0x80

    mov eax, 3
    mov ebx, 0
    mov ecx, num2
    mov edx, 12
    int 0x80

    mov esi, num2
    xor eax, eax

.convert2:
    mov bl, [esi]
    cmp bl, 0xA
    je .done2
    cmp bl, 0
    je .done2
    sub bl, '0'
    imul eax, eax, 10
    add eax, ebx
```

```
inc esi
jmp .convert2

.done2:
    mov ecx, eax

    mov eax, ebx
    imul eax, ecx

    mov edi, result + 23
    mov byte [edi], 0

.convert_loop:
    xor edx, edx
    mov ebx, 10
    div ebx
    add dl, '0'
    dec edi
    mov [edi], dl
    test eax, eax
    jnz .convert_loop

    mov eax, 4
    mov ebx, 1
    mov ecx, resmsg
    mov edx, lenr
    int 0x80

    mov eax, 4
    mov ebx, 1
    mov ecx, edi
    mov edx, result + 23
    sub edx, edi
```

```

int 0x80

mov eax, 4
mov ebx, 1
mov ecx, newline
mov edx, 1
int 0x80

```

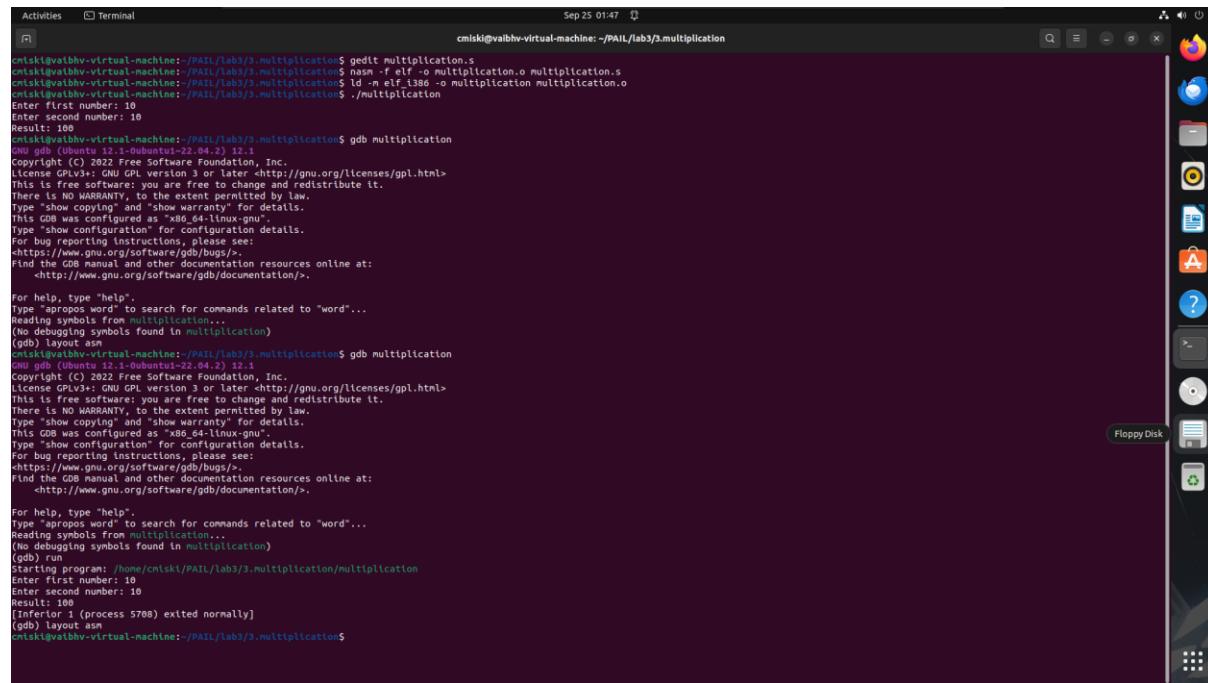
```

mov eax, 1
xor ebx, ebx
int 0x80

```

Output:

1]Console:



The screenshot shows a terminal window titled "Terminal" with the following session history:

```

Activities Terminal Sep 25 01:47
cmiski@vaibhv-virtual-machine:~/PAUL/lab3/multiplication$ gedit multiplication.s
cmiski@vaibhv-virtual-machine:~/PAUL/lab3/multiplication$ nasm -f elf -o multiplication.o multiplication.s
cmiski@vaibhv-virtual-machine:~/PAUL/lab3/multiplication$ ld -e elf_i386 -o multiplication multiplication.o
cmiski@vaibhv-virtual-machine:~/PAUL/lab3/multiplication$ ./multiplication
Enter first number: 10
Enter second number: 10
Result: 100
cmiski@vaibhv-virtual-machine:~/PAUL/lab3/multiplication$ gdb multiplication
GNU gdb (Ubuntu 12.1-0ubuntu1-22.04.2) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from multiplication...
(No debugging symbols found in multiplication)
(gdb) layout asm
cmiski@vaibhv-virtual-machine:~/PAUL/lab3/multiplication$ gdb multiplication
GNU gdb (Ubuntu 12.1-0ubuntu1-22.04.2) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from multiplication...
(No debugging symbols found in multiplication)
(gdb) starting program: /home/cmiski/PAUL/lab3/multiplication/multiplication
Enter first number: 10
Enter second number: 10
Result: 100
[Process 5708] exited normally
(gdb) layout asm
cmiski@vaibhv-virtual-machine:~/PAUL/lab3/multiplication$
```

2]GDB:

```

Activities Terminal Sep 25 01:43 cmiski@valbh-virtual-machine: ~/PAIL/lab3/3.multiplication
Registers
eax 0x4 4 ecx 0x0 0 edx 0x0 0
ebx 0x1 1 esp 0xfffffd230 0xfffffd230
esi 0x0 0 edi 0x0 0
eflags 0x202 [ IF ] cs 0x23 35 ss 0x2b 43
ds 0xb 43 es 0xb 43 fs 0x0 0
gs 0x0 0

B+ 0x049005 <_start> mov eax,0x4
> 0x04900a <_start+5> mov ebx,0x1
0x04900f <_start+10> mov edx,0x15
0x049014 <_start+15> int 0x80
0x049018 <_start+20> mov eax,0x3
0x04901b <_start+27> mov ebx,0x0
0x049020 <_start+32> mov ecx,0x804a038
0x049025 <_start+37> mov edx,0xc
0x04902a <_start+42> int 0x80
0x04902e <_start+47> mov eax,0x804a038
0x049031 <_start+49> xor eax, eax
0x049033 <_start.convert> mov bl,BYTE PTR [est]
0x049035 <_start.convert+2> cmp bl,0xa
0x049038 <_start.convert+5> je 0x049040 <_start.done1>
0x04903d <_start.convert+7> cmp bl,0xb
0x04904d <_start.convert+10> je 0x04904a <_start.done1>

native process 5689 In: _start
(gdb) layout regs
(gdb) _start
Breakpoint 1 at 0x049000
(gdb) r
Starting program: /home/cmiski/PAIL/lab3/3.multiplication/multiplication

Breakpoint 1, 0x049000 in _start ()
(gdb) s
0x049005 in _start ()
0x04900a in _start ()
(gdb)

Activities Terminal Sep 25 01:44 cmiski@valbh-virtual-machine: ~/PAIL/lab3/3.multiplication
Registers
eax 0x3 3 ecx 0x804a038 134520888 edx 0xc 12
eax 0x0 0 esp 0xfffffd230 0xfffffd230
ebx 0x1 1 edi 0x0 0
esi 0x804a038 134520888 cs 0x23 35 ss 0x2b 42
eflags 0x202 [ IF ] es 0xb 43 fs 0x0 6
gs 0x0 0

B+ 0x049005 <_start> mov eax,0x4
0x049013 <_start.convert> mov bl,BYTE PTR [est]
0x049018 <_start.convert+2> cmp bl,0xa000
0x04901c <_start.convert+4> je 0x049040 <_start.done1>
0x049031 <_start.convert+7> cmp bl,0x0
0x04903d <_start.convert+10> je 0x04904a <_start.done1>
0x04904d <_start.convert+12> sub bl,0x30
0x049050 <_start.convert+15> add eax,ebx
0x049054 <_start.convert+18> add eax,ebx
> 0x049058 <_start.convert+20> inc eax
0x049062 <_start.convert+21> jmp 0x049031 <_start.convert1>
0x049066 <_start.done1> mov ebx, eax
0x04906a <_start.done1+2> mov eax,0x0
0x04906c <_start.done1+4> mov ebx,0x1
0x04906e <_start.done1+6> mov eax,0x804a01_start.done1>
0x049070 <_start.done1+7> mov edx,0x16
0x049074 <_start.done1+9> int 0x80 0x4a <_start.done1>
0x049078 <_start.done1+24> mov eax,0x3

native process 5689 In: _start
(gdb) layout regs
(gdb) _convert1
0x04900f in _start ()
0x049014 in _start ()
Enter first number: 0x08849816 ln _start ()
0x049020 in _start ()
0x049025 in _start ()
0x049028 in _start ()
0x04902b in _start ()
0x04902c in _start ()
0x049031 in _start ()
0x049033 in _start.convert1 ()
0x049035 in _start.convert1 ()
0x049038 in _start.convert1 ()
0x049039 in _start.convert1 ()
0x04903d in _start.convert1 ()
0x049042 in _start.convert1 ()

10
0x04902c in _start ()
0x049031 in _start ()
0x049033 in _start.convert1 ()
0x049035 in _start.convert1 ()
0x049038 in _start.convert1 ()
0x049039 in _start.convert1 ()
0x04903d in _start.convert1 ()
0x049042 in _start.convert1 ()

(gdb)

```

The screenshot shows a terminal window with the following content:

```

Activities Terminal Sep 25 01:47 cmiski@vaibhv-virtual-machine: ~/PAUL/lab1/3.multiplication
eax 0x4 4 ecx 0x804a034 134520884 edx 0x1 1
eax 0x1 1 cs 0x23 35 edx 0x1 1
ebx 0x0 0 fs 0x0 0
eflags 0x286 [ PF ZF IF ] fs 0x0 0
eflags 0x246 [ PF ZF IF ] fs 0x0 0 convert loop

B+ 0x049000 <_start>      mov    eax,0x4
0x04904a <_start.done1>    mov    ebx, eax
0x0490d3 <_start.convert_loop+40> mov    ebx,0x1
0x0490e7

0x4
04a034
0x1 [ No Assembly Available ]
PTR [eax],al

(gdb) egs start.done1
0x049045 in _start.convert1 [convert_loop]
0x049045 in _start.convert1 [convert_loop]
0x0490cc in _start.convert_loop ()
0x0490d3 in _start.convert_loop ()
0x0490e3 in _start.convert_loop ()
0x0490d8 in _start.convert_loop ()
0x0490da in _start.convert_loop ()
0x0490df in _start.convert_loop ()
0x0490e0 in _start.convert_loop ()
0x0490e3 in _start.convert_loop ()
0x0490e8 in _start.convert_loop ()
0x0490ed in _start.convert_loop ()
0x0490f2 in _start.convert_loop ()
0x0490f7 in _start.convert_loop ()

0x80490f9 in _start.convert_loop ()
0x80490fe in _start.convert_loop ()
0x8049100 in _start.convert_loop ()
[Inferior 1 (process 5711) exited normally]
(gdb)

```

Division

A] on registers

Code:

```

section .data

qmsg db "Quotient: ",0
lenq equ $-qmsg

rmsg db " Remainder: ",0
lenr equ $-rmsg

newline db 0xA,0

```

```

section .bss

quotient resb 12
remainder resb 12

```

```

section .text

global _start

```

```
_start:
    mov eax,123      ; dividend
    mov ebx,10       ; divisor
    xor edx,edx     ; clear EDX before division
    div ebx         ; EAX = quotient, EDX = remainder

    mov esi,eax     ; store quotient
    mov edi,edx     ; store remainder

    ; Convert quotient to string
    mov eax,esi
    mov ecx,quotient+11
    mov byte [ecx],0

.q_loop:
    xor edx,edx
    mov ebp,10
    div ebp
    add dl,'0'
    dec ecx
    mov [ecx],dl
    test eax,eax
    jnz .q_loop

    ; Convert remainder to string
    mov eax,edi
    mov ecx,remainder+11
    mov byte [ecx],0

.r_loop:
    xor edx,edx
    mov ebp,10
    div ebp
    add dl,'0'
```

```
dec ecx
mov [ecx],dl
test eax, eax
jnz .r_loop

; Print quotient
mov eax,4
mov ebx,1
mov ecx,qmsg
mov edx,lenq
int 0x80
mov eax,4
mov ebx,1
mov ecx,ecx
mov ecx,ecx
mov ecx,ecx
mov ecx,quotient
mov edx,quotient+11
sub edx,ecx
int 0x80

; Print remainder
mov eax,4
mov ebx,1
mov ecx,rmsg
mov edx,lenr
int 0x80
mov eax,4
mov ebx,1
mov ecx,remainder
mov edx,remainder+11
sub edx,ecx
```

```

int 0x80

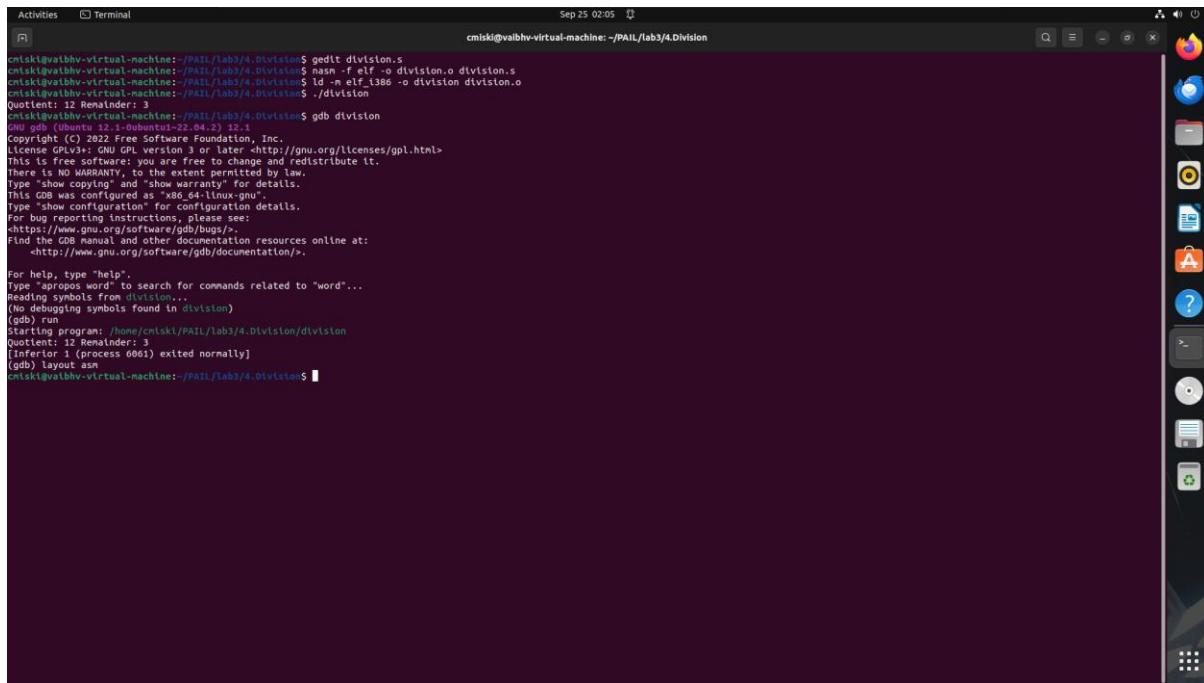
; Print newline
mov eax,4
mov ebx,1
mov ecx,newline
mov edx,1
int 0x80

; Exit
mov eax,1
xor ebx,ebx
int 0x80

```

Output:

1] Console



The screenshot shows a terminal window on a Linux desktop environment. The terminal output is as follows:

```

Activities Terminal Sep 25 02:05
cmislki@valbhv-virtual-machine:~/PAIL/labs/4.Division
cmislki@valbhv-virtual-machine:~/PAIL/labs/4.Division$ gedit division.s
cmislki@valbhv-virtual-machine:~/PAIL/labs/4.Division$ nasm -f elf -o division.o division.s
cmislki@valbhv-virtual-machine:~/PAIL/labs/4.Division$ ld -m elf_i386 -o division division.o
cmislki@valbhv-virtual-machine:~/PAIL/labs/4.Division$ ./division
Quotient: 12 Remainder: 3
cmislki@valbhv-virtual-machine:~/PAIL/labs/4.Division$ gdb division
GNU gdb (Ubuntu 12.1-0ubuntu1-22.04.2) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This software is free software: you can redistribute it and/or modify it
There is NO WARRANTY, to the extent permitted by law.
Type "show copying" and "show warranty" for details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<http://www.gnu.org/software/gdb/bugs/>
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from division... (No debugging symbols found in division)
(gdb) run
Starting program: /home/cmislki/PAIL/labs/4.Division/division
Quotient: 12 Remainder: 3
[Inferior 1 (process 6661) exited normally]
(gdb) layout asm
cmislki@valbhv-virtual-machine:~/PAIL/labs/4.Division$ 

```

2] GDB

Activities Terminal Sep 25 02:03 cmiski@valbhv-virtual-machine: ~/PAIL/lab3/4.Division

Register group: general		
eax 0x7b 123	ecx 0x0 0	edx 0x0 0
ebx 0xa 10	esp 0xfffffd240 0xfffffd240	ebp 0x0 0x0
esi 0x0 0	edi 0x0 0	esi 0x0 0x0
eflags 0x202 [IF]	cs 0x23 35	ss 0x2b 43
ds 0xb 43	es 0x2b 43	fs 0x0 0
gs 0x0 0		

```
B4 <start>: <start>: mov eax,0x7b
> 0x00490005 <start+5>    mov ebx,0xa
> 0x0049000a <start+10>   xor edx,edx
0x0049000c <start+12>   div ebx
0x0049000e <start+14>   mov eax,esi
0x00490010 <start+16>   mov edi,edx
0x00490012 <start+18>   mov eax,esi
0x00490014 <start+20>   mov ecx,0x804a027
0x00490019 <start+25>   mov BYTE PTR [ecx],0x0
0x0049001c <start+28>   xor edx,edx
0x0049001e <start-q_loop+2> mov eax,0x8
0x00490023 <start-q_loop+7> div ebp
0x00490025 <start-q_loop+9> add d1,0x30
0x00490028 <start-q_loop+12> dec ecx
0x00490029 <start-q_loop+13> mov BYTE PTR [ecx],dl
0x0049002c <start-q_loop+15> test eax,esi
0x0049002d <start-q_loop+17> jne <start>: <start.o_loops>
```

L7? PC: 0x0049000a

native process 6064 In: start
(gdb) layout regs
(gdb) b _start
Breakpoint 1 at 0x00490000
(gdb) r
Starting program: /home/cmiski/PAIL/lab3/4.Division/division
Breakpoint 1, 0x00490000 in _start ()
(gdb) s1
0x00490005 in _start ()
0x0049000a in _start ()
(gdb)

Activities Terminal Sep 25 02:05 cmiski@valbhv-virtual-machine: ~/PAIL/lab3/4.Division

Register group: general		
eax 0xc 12	ecx 0x0 0	edx 0x3 3
ebx 0xa 10	esp 0xfffffd240 0xfffffd240	ebp 0x8 0x8
esi 0x0 0	edi 0x0 0	esi 0x0 0x0
eflags 0x212 [AF IF]	cs 0x23 35	ss 0x2b 43
ds 0xb 43	es 0x2b 43	fs 0x0 0
gs 0x0 0		

```
B4 <start>: <start+12>   div ebx
> 0x0049000e <start+14>   mov eax,esi
0x00490010 <start+16>   mov edi,edx
0x00490012 <start+18>   mov eax,esi
0x00490014 <start+20>   mov ecx,0x804a027
0x00490019 <start+25>   mov BYTE PTR [ecx],0x0
0x0049001c <start-q_loop+2> xor edx,edx
0x0049001e <start-q_loop+7> mov eax,0x8
0x00490023 <start-q_loop+9> div ebp
0x00490025 <start-q_loop+9> add d1,0x30
0x00490028 <start-q_loop+12> dec ecx
0x00490029 <start-q_loop+13> mov BYTE PTR [ecx],dl
0x0049002c <start-q_loop+15> test eax,esi
0x0049002d <start-q_loop+17> jne <start>: <start.q_loop>
0x0049002f <start-q_loop+19> mov eax,edi
0x00490031 <start-q_loop+21> mov ecx,0x804a033
0x00490034 <start-q_loop+26> mov BYTE PTR [ecx],0x0
```

L7? PC: 0x0049000e

native process 6069 In: start
(gdb) layout regs
(gdb) b _start
Breakpoint 1 at 0x00490000
(gdb) r
Starting program: /home/cmiski/PAIL/lab3/4.Division/division
Breakpoint 1, 0x00490000 in _start ()
(gdb) s1
0x00490005 in _start ()
0x0049000a in _start ()
0x0049000c in _start ()
0x0049000e in _start ()
(gdb)

The terminal window shows the assembly code for a division program. The assembly code includes comments indicating loops for quotient and remainder calculation. The registers section shows various CPU registers with their current values.

```
Register group: general
eax      0x3          3
ebx      0xa          10
ecx      0xc          12
eflags   0x246        [ PF ZF IF ]
ds       0x2b         43
gs       0x0           0
eax      0x3          3
ebx      0xa          10
ecx      0xc          12
eflags   0x246        [ PF ZF IF ]
ds       0x2b         43
fs       0x0           0
0x049911 <start.q_loop>: xor    edx,edx
0x049913 <start.q_loop+2>: mov    ebp,0xa
0x049915 <start.q_loop+7>: dtv   ebp
0x049917 <start.q_loop+9>: add    dl,0x30
0x049919 <start.q_loop+11>: dec    ecx
0x04991b <start.q_loop+13>: mov    dl,BYTE PTR [ecx],dl
0x04992b <start.q_loop+15>: test   eax,eax
0x04992d <start.q_loop+17>: jne    eax,edi
0x049931 <start.q_loop+19>: mov    ecx,0x049933
0x049933 <start.r_loop>: xor    edx,edx
0x049935 <start.r_loop+2>: mov    ebp,0xa
0x04993b <start.r_loop+7>: dtv   ebp
0x049940 <start.r_loop+9>: add    dl,0x30
0x049942 <start.r_loop+11>: dec    ecx
0x049946 <start.r_loop+13>: mov    dl,BYTE PTR [ecx],dl
native process 6064 In: _start.q_loop
0x049911 ln _start.q_loop()
0x049913 ln _start.q_loop()
0x049915 ln _start.q_loop()
0x049917 ln _start.q_loop()
0x049919 ln _start.q_loop()
0x04992b ln _start.q_loop()
0x04992d ln _start.q_loop()
0x049931 ln _start.q_loop()
0x049933 ln _start.q_loop()
0x049935 ln _start.q_loop()
0x04993b ln _start.q_loop()
0x049940 ln _start.q_loop()
0x049942 ln _start.q_loop()
0x049946 ln _start.q_loop()
0x049948 ln _start.q_loop()
0x049950 ln _start.q_loop()
0x049952 ln _start.q_loop()
0x049954 ln _start.q_loop()
0x049956 ln _start.q_loop()
0x049958 ln _start.q_loop()
0x04995a ln _start.q_loop()
0x04995c ln _start.q_loop()
0x04995e ln _start.q_loop()
0x049960 ln _start.q_loop()
0x049962 ln _start.q_loop()
0x049964 ln _start.q_loop()
0x049966 ln _start.q_loop()
0x049968 ln _start.q_loop()
0x04996a ln _start.q_loop()
0x04996c ln _start.q_loop()
0x04996e ln _start.q_loop()
0x049970 ln _start.q_loop()
0x049972 ln _start.q_loop()
0x049974 ln _start.q_loop()
0x049976 ln _start.q_loop()
0x049978 ln _start.q_loop()
0x04997a ln _start.q_loop()
0x04997c ln _start.q_loop()
0x04997e ln _start.q_loop()
0x049980 ln _start.q_loop()
0x049982 ln _start.q_loop()
0x049984 ln _start.q_loop()
0x049986 ln _start.q_loop()
0x049988 ln _start.q_loop()
0x04998a ln _start.q_loop()
0x04998c ln _start.q_loop()
0x04998e ln _start.q_loop()
0x049990 ln _start.q_loop()
0x049992 ln _start.q_loop()
0x049994 ln _start.q_loop()
0x049996 ln _start.q_loop()
0x049998 ln _start.q_loop()
0x04999a ln _start.q_loop()
0x04999c ln _start.q_loop()
0x04999e ln _start.q_loop()
0x0499a0 ln _start.q_loop()
0x0499a2 ln _start.q_loop()
0x0499a4 ln _start.q_loop()
0x0499a6 ln _start.q_loop()
0x0499a8 ln _start.q_loop()
0x0499aa ln _start.q_loop()
0x0499ac ln _start.q_loop()
0x0499ae ln _start.q_loop()
0x0499b0 ln _start.q_loop()
0x0499b2 ln _start.q_loop()
0x0499b4 ln _start.q_loop()
0x0499b6 ln _start.q_loop()
0x0499b8 ln _start.q_loop()
0x0499ba ln _start.q_loop()
0x0499bc ln _start.q_loop()
0x0499be ln _start.q_loop()
0x0499c0 ln _start.q_loop()
0x0499c2 ln _start.q_loop()
0x0499c4 ln _start.q_loop()
0x0499c6 ln _start.q_loop()
0x0499c8 ln _start.q_loop()
0x0499ca ln _start.q_loop()
0x0499cc ln _start.q_loop()
0x0499cc lni _start.q_loop()
native process 6064 In: _start.r_loop
0x049911 ln _start.r_loop()
0x049913 ln _start.r_loop()
0x049915 ln _start.r_loop()
0x049917 ln _start.r_loop()
0x049919 ln _start.r_loop()
0x04992b ln _start.r_loop()
0x04992d ln _start.r_loop()
0x049931 ln _start.r_loop()
0x049933 ln _start.r_loop()
0x049935 ln _start.r_loop()
0x04993b ln _start.r_loop()
0x049940 ln _start.r_loop()
0x049942 ln _start.r_loop()
0x049946 ln _start.r_loop()
0x049948 ln _start.r_loop()
0x04994a ln _start.r_loop()
0x04994c ln _start.r_loop()
0x04994e ln _start.r_loop()
0x049950 ln _start.r_loop()
0x049952 ln _start.r_loop()
0x049954 ln _start.r_loop()
0x049956 ln _start.r_loop()
0x049958 ln _start.r_loop()
0x04995a ln _start.r_loop()
0x04995c ln _start.r_loop()
0x04995e ln _start.r_loop()
0x049960 ln _start.r_loop()
0x049962 ln _start.r_loop()
0x049964 ln _start.r_loop()
0x049966 ln _start.r_loop()
0x049968 ln _start.r_loop()
0x04996a ln _start.r_loop()
0x04996c ln _start.r_loop()
0x04996e ln _start.r_loop()
0x049970 ln _start.r_loop()
0x049972 ln _start.r_loop()
0x049974 ln _start.r_loop()
0x049976 ln _start.r_loop()
0x049978 ln _start.r_loop()
0x04997a ln _start.r_loop()
0x04997c ln _start.r_loop()
0x04997e ln _start.r_loop()
0x049980 ln _start.r_loop()
0x049982 ln _start.r_loop()
0x049984 ln _start.r_loop()
0x049986 ln _start.r_loop()
0x049988 ln _start.r_loop()
0x04998a ln _start.r_loop()
0x04998c ln _start.r_loop()
0x04998e ln _start.r_loop()
0x049990 ln _start.r_loop()
0x049992 ln _start.r_loop()
0x049994 ln _start.r_loop()
0x049996 ln _start.r_loop()
0x049998 ln _start.r_loop()
0x04999a ln _start.r_loop()
0x04999c ln _start.r_loop()
0x04999e ln _start.r_loop()
0x0499a0 ln _start.r_loop()
0x0499a2 ln _start.r_loop()
0x0499a4 ln _start.r_loop()
0x0499a6 ln _start.r_loop()
0x0499a8 ln _start.r_loop()
0x0499aa ln _start.r_loop()
0x0499bc ln _start.r_loop()
0x0499cc ln _start.r_loop()
0x0499cc lni _start.r_loop()
(gdb)
```

B] Taking user inputs

Code:

```
section .data
```

```
msg1 db "Enter dividend: ",0
```

```
len1 equ $-msg1
```

```
msg2 db "Enter divisor: ",0
```

```
len2 equ $-msg2
```

```
qmsg db "Quotient: ",0
```

```
lenq equ $-qmsg
```

```
rmsg db " Remainder: ",0
```

```
lenr equ $-rmsg
```

```
newline db 0xA,0
```

```
section .bss
```

```
num1 resb 12
```

```
num2 resb 12
```

```
quotient resb 12
```

```
remainder resb 12
```

```
section .text
```

```
global _start

_start:
; ---- Read dividend ----
mov eax,4
mov ebx,1
mov ecx,msg1
mov edx,len1
int 0x80

mov eax,3
mov ebx,0
mov ecx,num1
mov edx,12
int 0x80

; ---- Convert dividend to integer ----
mov esi,num1
xor eax,eax
.convert1:
    mov bl,[esi]
    cmp bl,0xA          ; stop at newline
    je .done1
    sub bl,'0'
    imul eax,eax,10
    add eax,ebx
    inc esi
    jmp .convert1
.done1:
    mov esi,eax          ; store dividend in ESI

; ---- Read divisor ----
```

```
    mov eax,4
    mov ebx,1
    mov ecx,msg2
    mov edx,len2
    int 0x80

    mov eax,3
    mov ebx,0
    mov ecx,num2
    mov edx,12
    int 0x80

; ----- Convert divisor to integer -----
    mov edi,num2
    xor eax,eax
.convert2:
    mov bl,[edi]
    cmp bl,0xA
    je .done2
    sub bl,'0'
    imul eax,eax,10
    add eax,ebx
    inc edi
    jmp .convert2
.done2:
    mov edi,eax

; ----- Perform division -----
    mov eax,esi
    xor edx,edx
    div edi
    mov esi,eax
```

```
    mov edi,edx

    ; ---- Convert quotient to string ----

    mov eax,esi
    mov ecx,quotient+11
    mov byte [ecx],0

.q_loop:
    xor edx,edx
    mov ebp,10
    div ebp
    add dl,'0'
    dec ecx
    mov [ecx],dl
    test eax,eax
    jnz .q_loop

    ; ---- Convert remainder to string ----

    mov eax,edi
    mov ecx,remainder+11
    mov byte [ecx],0

.r_loop:
    xor edx,edx
    mov ebp,10
    div ebp
    add dl,'0'
    dec ecx
    mov [ecx],dl
    test eax,eax
    jnz .r_loop

    ; ---- Print quotient ----

    mov eax,4
```

```
mov ebx,1
mov ecx,qmsg
mov edx,lenq
int 0x80
mov eax,4
mov ebx,1
mov ecx,quotient
mov edx,quotient+11
sub edx,ecx
int 0x80

; ---- Print remainder ----
mov eax,4
mov ebx,1
mov ecx,rmsg
mov edx,lenr
int 0x80
mov eax,4
mov ebx,1
mov ecx,remainder
mov edx,remainder+11
sub edx,ecx
int 0x80

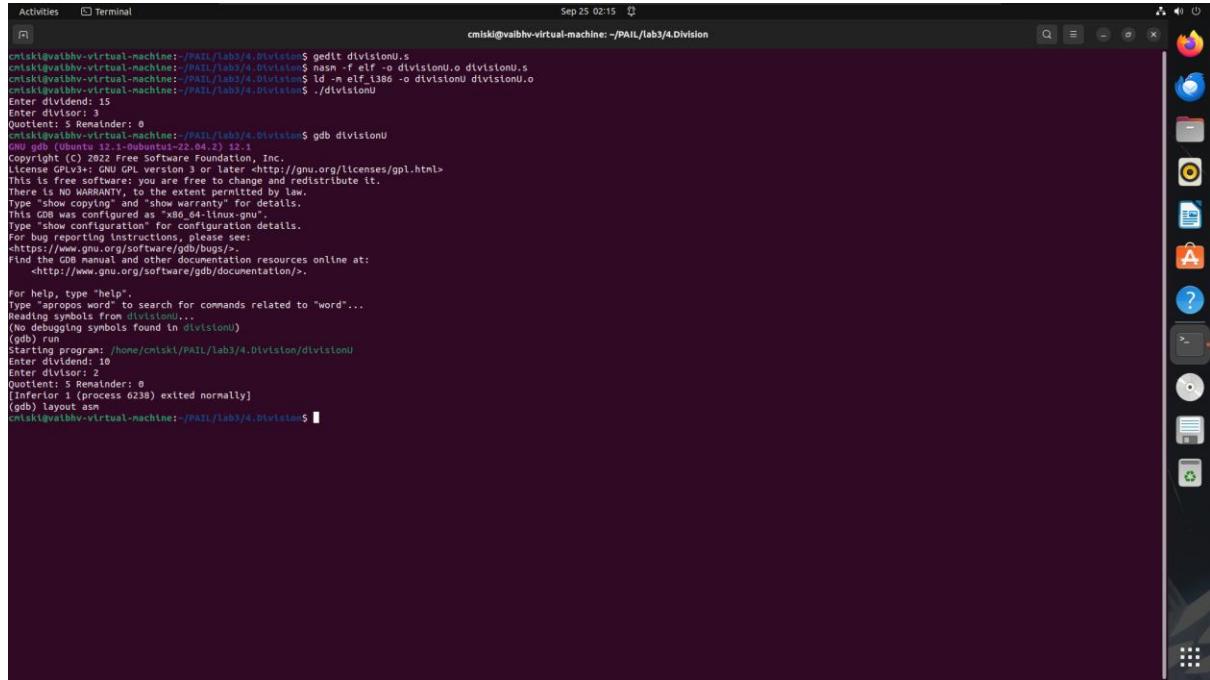
; ---- Print newline ----
mov eax,4
mov ebx,1
mov ecx,newline
mov edx,1
int 0x80

; ---- Exit ----
```

```
mov eax,1  
xor ebx,ebx  
int 0x80
```

Output:

1] Console



The screenshot shows a terminal window titled "Terminal" with the command line "cmiskt@valbhv-virtual-machine: ~/PAIL/lab3/4.Division". The terminal displays the assembly code and its execution in GDB:

```
cmiskt@valbhv-virtual-machine:~/PAIL/lab3/4.Division$ gedit divisionU.s  
cmiskt@valbhv-virtual-machine:~/PAIL/lab3/4.Division$ gcc -fPIC -o divisionU.o divisionU.s  
cmiskt@valbhv-virtual-machine:~/PAIL/lab3/4.Division$ ld -m elf_i386 -o divisionU divisionU.o  
cmiskt@valbhv-virtual-machine:~/PAIL/lab3/4.Division$ ./divisionU  
Enter dividend: 15  
Enter divisor: 3  
Quotient: 5 Remainder: 0  
cmiskt@valbhv-virtual-machine:~/PAIL/lab3/4.Division$ gdb divisionU  
GNU gdb (Ubuntu 12.1-0ubuntu1-22.04.2) 12.1  
Copyright (C) 2022 Free Software Foundation, Inc.  
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>  
This software is copyrighted by Free Software Foundation, Inc.  
There is NO WARRANTY, to the extent permitted by law.  
Type "show copying" and "show warranty" for details.  
This GDB was configured as "x86_64-linux-gnu".  
Type "show configuration" for configuration details.  
For bug reporting instructions, see:  
https://www.gnu.org/software/gdb/bugs/.  
Find the GDB manual and other documentation resources online at:  
http://www.gnu.org/software/gdb/documentation/.  
  
For help, type "help".  
Type "apropos word" to search for commands related to "word"....  
Reading symbols from divisionU...  
(No debugging symbols found in divisionU)  
(gdb) run  
Starting program: /home/cmiskt/PAIL/lab3/4.Division/divisionU  
Enter dividend: 10  
Enter divisor: 2  
Quotient: 5 Remainder: 0  
[Inferior 1 (process 6238) exited normally]  
(gdb) layout asm  
cmiskt@valbhv-virtual-machine:~/PAIL/lab3/4.Division$
```

2] GDB

Activities Terminal Sep 25 02:14 cmiski@valbh-virtual-machine: ~/PAIL/lab3/4.Division

```
B* <start>: mov eax,0x4
              mov ebx,0x1
              mov ecx,0x804a000
              mov edx,0x11
              int 0x80
              mov eax,0x3
              mov ebx,0x0
              mov cx,0x804a03c
              mov edx,0xc
              int 0x80
              mov esl,0x804a03c
              xor eax,eax
              mov bl,BYTE PTR [esl]
              cmp bl,0xa
              je 0x0049041 <_start.done>
              sub bl,0x30
              imul eax,eax,0xa
native process 6241 Int: _start
(gdb) layout regs
(gdb) b _start
Breakpoint 1 at 0x8049000
(gdb) r
starting program: /home/cmiski/PAIL/lab3/4.Division/divisionU
Breakpoint 1, 0x0049000 in _start ()
(gdb) s
```

Activities Terminal Sep 25 02:14 cmiski@valbh-virtual-machine: ~/PAIL/lab3/4.Division

```
eax 0x3 3 ecx 0x804a03c 134520892 edx 0x0 12
esp 0x0 0 esp 0xffffd240 0xffffd240 edi 0x0 12
ebx 0x0 0 cs 0x23 35 ss 0xb 10 0x804902a <_start+42> .convert1+5>
eflags 0x216 49 fs 0x2b 43
gs 0x0 0 fs 0x0 0
```

B* <start>: mov eax,0x4
 mov ebx,0x1
 mov ecx,0x804a000
 mov edx,0x11
 int 0x80
 mov eax,0x3
 mov ebx,0x0
 mov cx,0x804a03c
 mov edx,0xc
 int 0x80
 mov esl,0x804a03c
 xor eax,eax
 mov bl,BYTE PTR [est]
 cmp bl,0xa
 je 0x0049041 <_start.done>
 sub bl,0x30
 imul eax,eax,0xa
native process 6241 Int: _start
(gdb) layout regs
breakpoint 1, 0x00490000 in _start ()
(gdb) s
l0d049005 in _start ()
0x004900a in _start ()
0x004900f in _start ()
0x0049014 in _start ()
0x004901b in _start ()
0x0049020 in _start ()
0x0049025 in _start ()
0x0049031 in _start ()
0x0049035 in _start.convert1+2>
0x0049039 in _start.convert1+6>
Enter dividend: 0x0049031 in _start ()
0x004901b in _start ()
0x0049020 in _start ()
0x0049025 in _start ()
0x0049028 in _start ()
0x004902c in _start ()
0x0049031 in _start ()
0x0049033 in _start.convert1 ()
0x0049035 in _start.convert1 ()
0x0049038 in _start.convert1 ()
(gdb) l

```

Activities Terminal Sep 25 02:14 cmiski@vaibhav-virtual-machine: ~/PAIL/lab3/4.Division
eax 0x3 3 ecx 0x804a03c 134520892 edx 0xc 12
eax 0x4 4 edx 0x804a011 134520849 edx 0xd 16
ebx 0x1 1 edi 0 0 ss 0x2b 71 43 71 .done1+44>
ext 0xa 10 es 0x23 35 fs 0x0 0
eflags 0x246 [ PF ZF IF ] edi 0x2b 43 .done1+22>
gs 0x0 0

```

```

B: 0x00490000 <_start> mov eax,0x4
0x00490015 <_start.convert1> mov bl,BYTE PTR [esi]
0x0049001a <_start.convert1+2> cmp bl,0xa 0x4000
0x0049001d <_start.convert1+5> je 0x0049041 <_start.done1>
0x004901a <_start.convert1+7> sub bl,0x30
0x004901d <_start.convert1+10> imul eax, eax, 0xa
0x0049020 <_start.convert1+13> add eax, ebx
0x0049042 <_start.convert1+15> inc esl 0x4083c
0x0049041 <_start.convert1+16> jmp 0x0049031 <_start.convert1>
0x0049047 <_start.done1> mov est,eax
0x0049047 <_start.done1+2> mov eax,0x40a03c
0x0049050 <_start.done1+3> xor eax, eax
0x0049051 <_start.done1+4> mov ecx,0x804a011esi1]
0x0049056 <_start.done1+7> mov edx,0x10
> 0x004905b <_start.done1+22> int 0x80 0x41 <_start.done1>
0x004905d <_start.done1+24> mov eax,0x0
0x004905e <_start.done1+29> mov ebx,0x0,0xa
0x004905f <_start.done1+34> mov ecx,0x804a048
native process 6241 In: _start
(gdb) layout regs ,done1
0x00490033 ln _start.convert1 ()
0x00490036 ln _start.convert1 ()
0x00490039 ln _start.convert1 ()
0x0049003a ln _start.convert1 ()
0x0049003d ln _start.convert1 ()
0x00490040 ln _start.convert1 ()
0x00490043 ln _start.convert1 ()
0x00490047 ln _start.convert1 ()
0x0049004c ln _start.convert1 ()
0x00490051 ln _start.convert1 ()
0x00490056 ln _start.convert1 ()
0x0049005b ln _start.convert1 ()
(gdb) 
Activities Terminal Sep 25 02:15 cmiski@vaibhav-virtual-machine: ~/PAIL/lab3/4.Division
eax 0x3 3 ecx 0x804a048 134520904 edx 0xc 12
eax 0x2 2 edx 0x0 0 ss 0x2b 71 43 71 .done1+44>
ebx 0xa 10 cs 0x23 35 fs 0x0 0
eflags 0x246 [ PF ZF IF ] edi 0x804a049 134520905 .convert2+2>
eflags 0x262 [ IF ]

```

```

B: 0x00490000 <_start> mov eax,0x4
0x00490045 <_start.done1> mov esl,eax
0x00490045 <_start.convert2> mov bl,BYTE PTR [edi]
> 0x004907c <_start.convert2> cmp bl,0xa
0x004907f <_start.convert2+2> je 0x0049080 <_start.done2>
0x0049080 <_start.convert2+5> sub bl,0x30
0x0049084 <_start.convert2+10> imul eax, eax, 0xa
0x0049087 <_start.convert2+13> add eax, ebx0x403c
0x0049090 <_start.convert2+15> inc esl
0x0049093 <_start.convert2+16> jmp 0x0049097a <_start.convert2>
0x0049095 <_start.done2> mov edi,eax
0x0049095 <_start.done2+2> int 0x80,esi
0x0049095 <_start.done2+4> xor edx,edx0x4048esi1]
0x0049092 <_start.done2+6> div edi
0x0049092 <_start.done2+8> mov edx,0x4048 PTR [edi].done1>
0x0049095 <_start.done2+10> mov edi,edx
0x0049095 <_start.done2+12> mov eax,esl <_start.done2>
0x0049095 <_start.done2+14> mov ecx,0x804a05f
native process 6241 In: done1+14> mov BYTE PTR [ecx],0x0
(gdb) layout regs ,done1
0x00490062 ln _start.convert1 <convert2>
0x00490067 ln _start.done1 ()
0x00490067 ln _start.done1 ()
0x0049006c ln _start.done1 ()
0x00490071 ln _start.done1 ()
2
0x00490073 ln _start.done1 ()
0x00490078 ln _start.done1 ()
0x0049007a ln _start.convert2 ()
0x0049007c ln _start.convert2 ()
0x0049007f ln _start.convert2 ()
0x00490081 ln _start.convert2 ()
0x00490084 ln _start.convert2 ()
0x00490087 ln _start.convert2 ()
0x00490089 ln _start.convert2 ()
0x00490091 ln _start.convert2 ()
0x00490097a ln _start.convert2 ()
0x00490097c ln _start.convert2 ()
(gdb) 

```

Lab-4

A] Array Addition (result less than 10 and 2nd by using AAM)

Code:

```
section .text
```

```
global _start
```

```
_start:  
  
    mov eax, x  
    mov ebx, 0  
    mov ecx, 5
```

```
top:  
    add bl, [eax]  
    inc eax  
    loop top
```

```
done:  
    add bl, '0'  
    mov [sum], bl
```

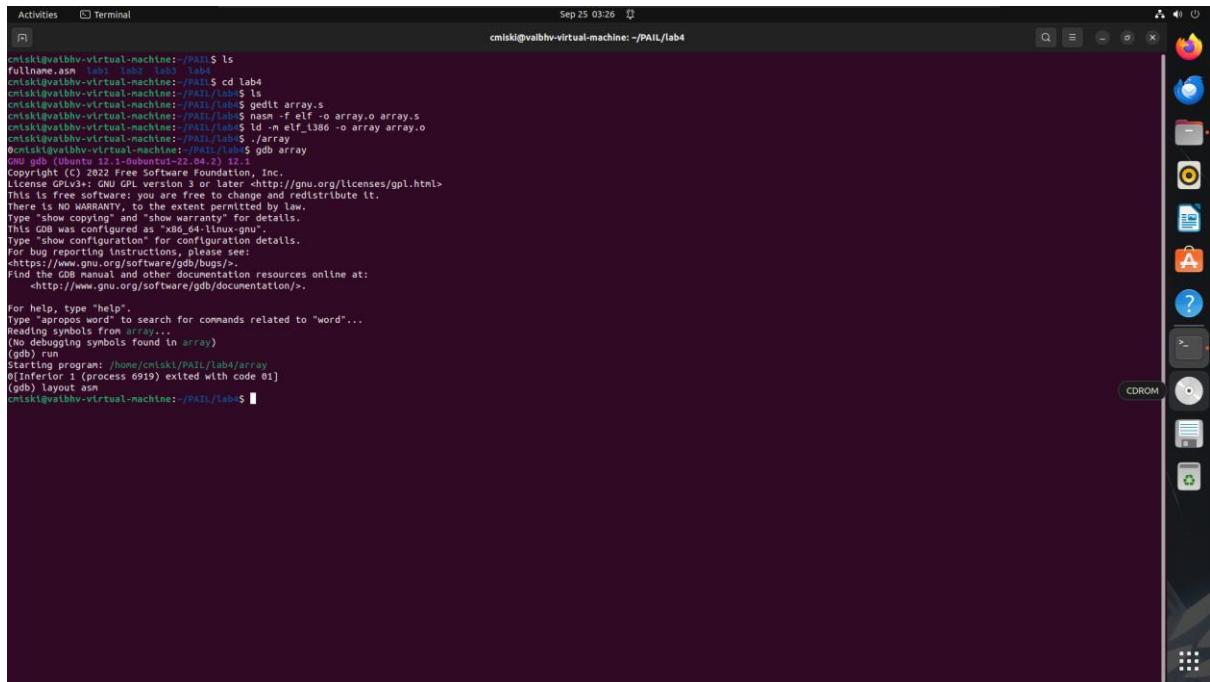
```
display:  
    mov edx, 1  
    mov ecx, sum  
    mov ebx, 1  
    mov eax, 4  
    int 0x80
```

```
    mov eax, 1  
    int 0x80
```

```
section .data  
x:  
    times 5 db 0
```

```
sum:  
    db 0, 0xa
```

Output:



The screenshot shows a terminal window titled "Terminal" with the command line "cmislk@vaibhv-virtual-machine:~/PAIL/lab4". The terminal displays the assembly code for a C program named "array". The assembly code includes instructions like "ld", "add", "sub", "mul", and "div". It also contains comments such as "/*", "*/", and "/* \$t1 = */". The assembly code is generated from the C source code provided in the question.

```
cmislk@vaibhv-virtual-machine:~/PAIL/lab4$ ls
fullname.asm  lab1  lab2  lab3  lab4
cmislk@vaibhv-virtual-machine:~/PAIL/lab4$ cd lab4
cmislk@vaibhv-virtual-machine:~/PAIL/lab4$ ls
array.s
cmislk@vaibhv-virtual-machine:~/PAIL/lab4$ nm -f elf array.o array.s
cmislk@vaibhv-virtual-machine:~/PAIL/lab4$ ld -o elf_i386 -o array array.o
cmislk@vaibhv-virtual-machine:~/PAIL/lab4$ ./array
0x0000000000400000
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is free software: you are free to change and redistribute it.
There is NO WARRANTY, to the extent permitted by law.
Type "show configuration" for configuration details.
This GDB was configured as "x86_64-linux-gnu".
Type "show configuration" for configuration details.
For bug reporting instructions, please see:
<https://www.gnu.org/software/gdb/bugs/>.
Find the GDB manual and other documentation resources online at:
<http://www.gnu.org/software/gdb/documentation/>.

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from array...
(No debugging symbols found in array)
(gdb) run
Starting program: /home/cmislk/PAIL/lab4/array
0[Inferior 1 (process 4919) exited with code 0]
(gdb) layout asm
cmislk@vaibhv-virtual-machine:~/PAIL/lab4$
```

```

Register group: general
eax          0x5          5
ecx          0x4          4
edx          0x0          0
ebx          0x0          0
esp    0xfffffd2c0 0xfffffd2c0

0x8049007 <sum_loop>    mov    al,BYTE PTR [esi+0x804a000]
0x804900d <sum_loop+6>   add    al,BYTE PTR [esi+0x804a004]
0x8049013 <sum_loop+12>  daa
> 0x8049014 <sum_loop+13> mov    BYTE PTR [esi+0x804a008],al
0x804901a <sum_loop+19>  inc    esi
0x804901b <sum_loop+20>  loop   0x8049007 <sum_loop>
0x804901d <sum_loop+22>  mov    eax,0x4

native process 6092 In: sum_loop          L??  PC: 0x8049014
0x08049007 in sum_loop ()
(gdb) nexti
0x0804900d in sum_loop ()
(gdb) nexti
0x08049013 in sum_loop ()
(gdb) nexti
0x08049014 in sum_loop ()
(gdb) ■

Activities Terminal Sep 25 03:26 cmiski@valbhv-virtual-machine: ~/PAIL/lab4
Register group: general
eax          0x1          1
ebx          0x1          1
ecx          0x0          0
esi          0x0          0
eflags        0x206      [ PF IF ]
ds           0x2b         43
es           0xb          43
gs           0x0          0

[ No Assembly Available ]

native No process In:
0x08049012 in top()
0x08049009 in top()
0x08049011 in top()
0x08049012 in top()
0x0804900f in top()
0x08049011 in top()
0x08049012 in top()
0x0804900e in top()
0x08049017 in done()
0x0804901d in display()
0x08049022 in display()
0x08049027 in display()
0x08049028 in display()
0x08049031 in display()
0x08049033 in display()
0x08049038 in display()
[Inferior 1 (process 6922) exited with code 01]
(gdb)

```

B] String Operation

Code:

```
global _start
```

```
section .text
```

```
_start:
    cld
    mov ecx, len
    mov esi, s1
    mov edi, s2

loop_here:
    lodsb
    or al, 20h
    stosb
    loop loop_here

    ; print result
    mov edx, len
    mov ecx, s2
    mov ebx, 1
    mov eax, 4
    int 0x80

    ; exit
    mov eax, 1
    xor ebx, ebx
    int 0x80

section .data
s1 db 'HELLO, WORLD', 0xa
len equ $-s1

section .bss
s2 resb len
```

Output:

The screenshot shows a terminal window on a Linux desktop environment. The terminal window title is "Terminal" and the date and time are "Sep 25 03:27". The command history in the terminal is as follows:

```
cmiski@valbhv-virtual-machine:~/PAIL/lab4$ gedit string.s
cmiski@valbhv-virtual-machine:~/PAIL/lab4$ gcc -c string.s -o string.o
cmiski@valbhv-virtual-machine:~/PAIL/lab4$ ld -m elf_i386 -o string string.o
cmiski@valbhv-virtual-machine:~/PAIL/lab4$ ./string
hello, world*
```

The desktop interface includes a dock with icons for various applications like the Dash, Home, and Dash to Dock, along with a vertical panel on the right containing icons for the Dash, Home, and other system functions.