RISCV_VSD_SquadronMini Workshop

Instructor : Kunal Ghosh

PRE-REQUISITE: Installing the required applications for the workshop - Virtual Box, Ubuntu on VBBOX and VDI files

TASK-1:

A: Write a simple C code and compile it with gcc comipler.

B: Compile the same code with RISCV comipler to generate the assembly code for the same. Further Evaluate RISCV assembly code for the sample C code with two different options of comiplation.

NECESSARY INSTALLATIONS

Step 1: Setting up the virtual environment to work on

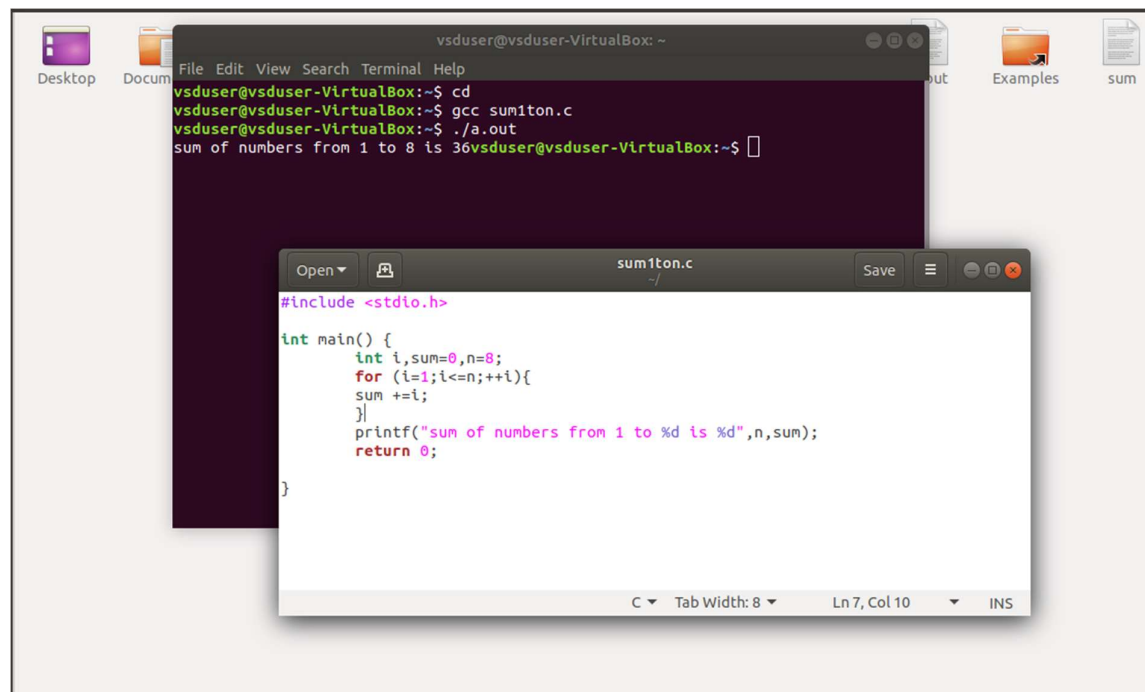Install Oracle Virtual Box, VMBox

Launch Virtual Machine on VMBox

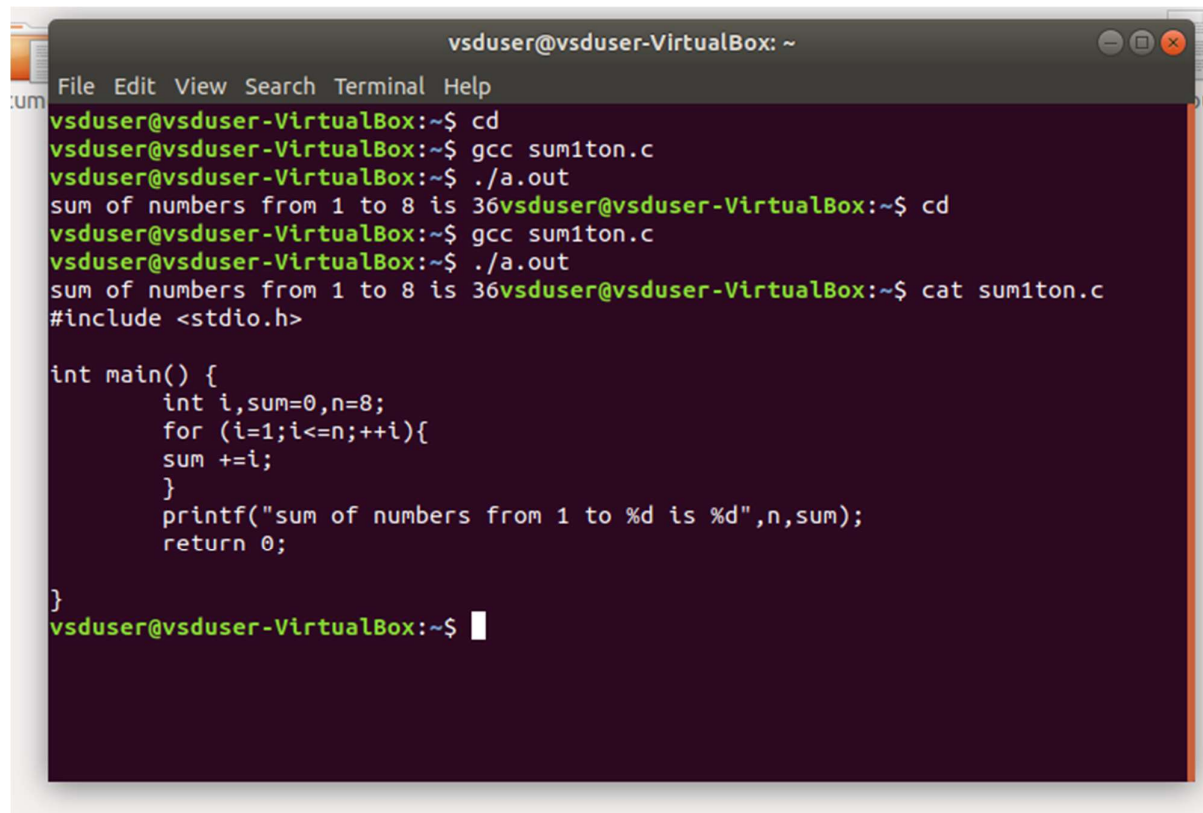Attach the VDI file to the Virtual Machine instance in VMBox

TASK 1A - COMPILE AND EXECUTE A SIMPLE C CODE USING GCC COMPILER

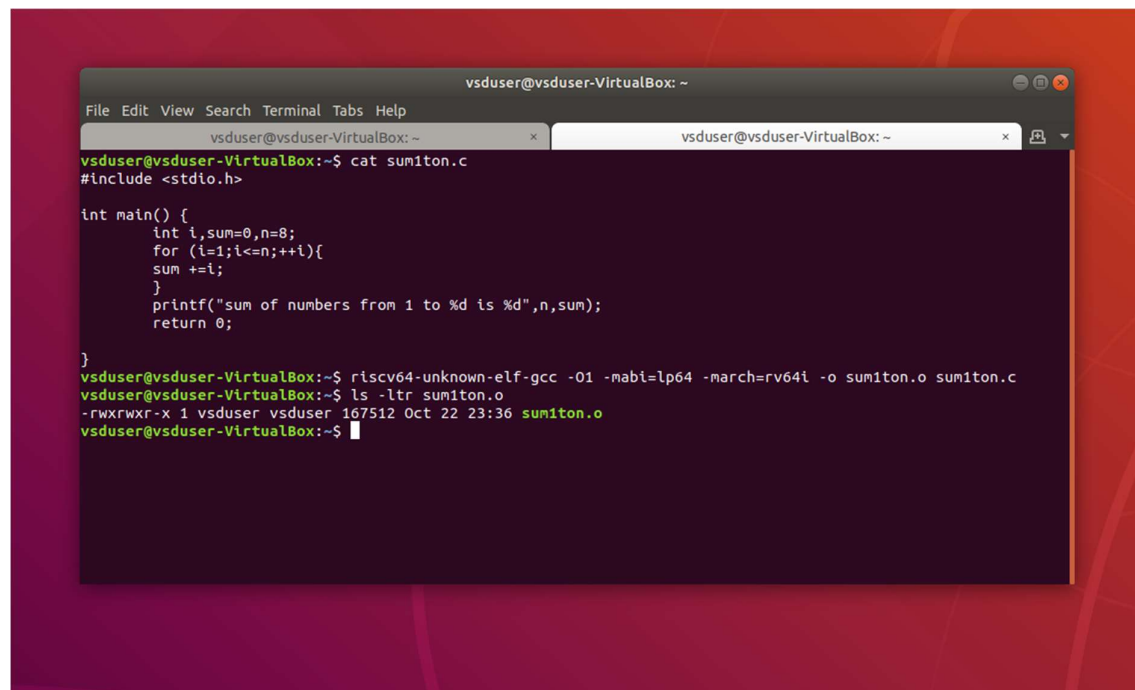$ cd
                Navigate to home directory:

$ gedit filename.c &
       This opens a blank file with filename.c, type the c

Save the file


```
vsduser@vsduser-VirtualBox:~$ cd
vsduser@vsduser-VirtualBox:~$ gcc sum1ton.c
vsduser@vsduser-VirtualBox:~$ ./a.out
sum of numbers from 1 to 8 is 36vsduser@vsduser-VirtualBox:~$ cd
vsduser@vsduser-VirtualBox:~$ gcc sum1ton.c
vsduser@vsduser-VirtualBox:~$ ./a.out
sum of numbers from 1 to 8 is 36vsduser@vsduser-VirtualBox:~$ cat sum1ton.c
#include <stdio.h>

int main() {
        int i,sum=0,n=8;
        for (i=1;i<=n;++i){
        sum +=i;
        }
        printf("sum of numbers from 1 to %d is %d",n,sum);
        return 0;

}
vsduser@vsduser-VirtualBox:~$
```



```
vsduser@vsduser-VirtualBox:~$ cat sum1ton.c
#include <stdio.h>

int main() {
        int i,sum=0,n=8;
        for (i=1;i<=n;++i){
        sum +=i;
        }
        printf("sum of numbers from 1 to %d is %d",n,sum);
        return 0;

}
vsduser@vsduser-VirtualBox:~$ riscv64-unknown-elf-gcc -O1 -mabi=lp64 -march=rv64i -o sum1ton.o sum1ton.c
vsduser@vsduser-VirtualBox:~$ ls -ltr sum1ton.o
-rwxrwxr-x 1 vsduser vsduser 167512 Oct 22 23:36 sum1ton.o
vsduser@vsduser-VirtualBox:~$
```

```
0000000000010184 <main>:
   10184:    ff010113          addi     sp,sp,-16
   10188:    00113423          sd       ra,8(sp)
   1018c:    02400613          li       a2,36
   10190:    00800593          li       a1,8
   10194:    00021537          lui      a0,0x21
   10198:    18050513          addi     a0,a0,384 # 21180 <__clzdi2+0x48>
   1019c:    26c000ef          jal      ra,10408 <printf>
   101a0:    00000513          li       a0,0
   101a4:    00813083          ld       ra,8(sp)
   101a8:    01010113          addi     sp,sp,16
   101ac:    00008067          ret

00000000000101b0 <atexit>:
   101b0:    00050593          mv       a1,a0
   101b4:    00000693          li       a3,0
   101b8:    00000613          li       a2,0
   101bc:    00000513          li       a0,0
   101c0:    4390206f          j        12df8 <__register_exitproc>

00000000000101c4 <exit>:
   101c4:    ff010113          addi     sp,sp,-16
   101c8:    00000593          li       a1,0
:
```