Module 4 – Assessment

**Code Debugging Tools (GBD and Valgrind)**

1. Using Valgind identify memleaks in the given program. Explore optional flags in Valgrind.

Answer:

**Valgrind Commands**

> sudo bash // become su

>apt-get install valgrind

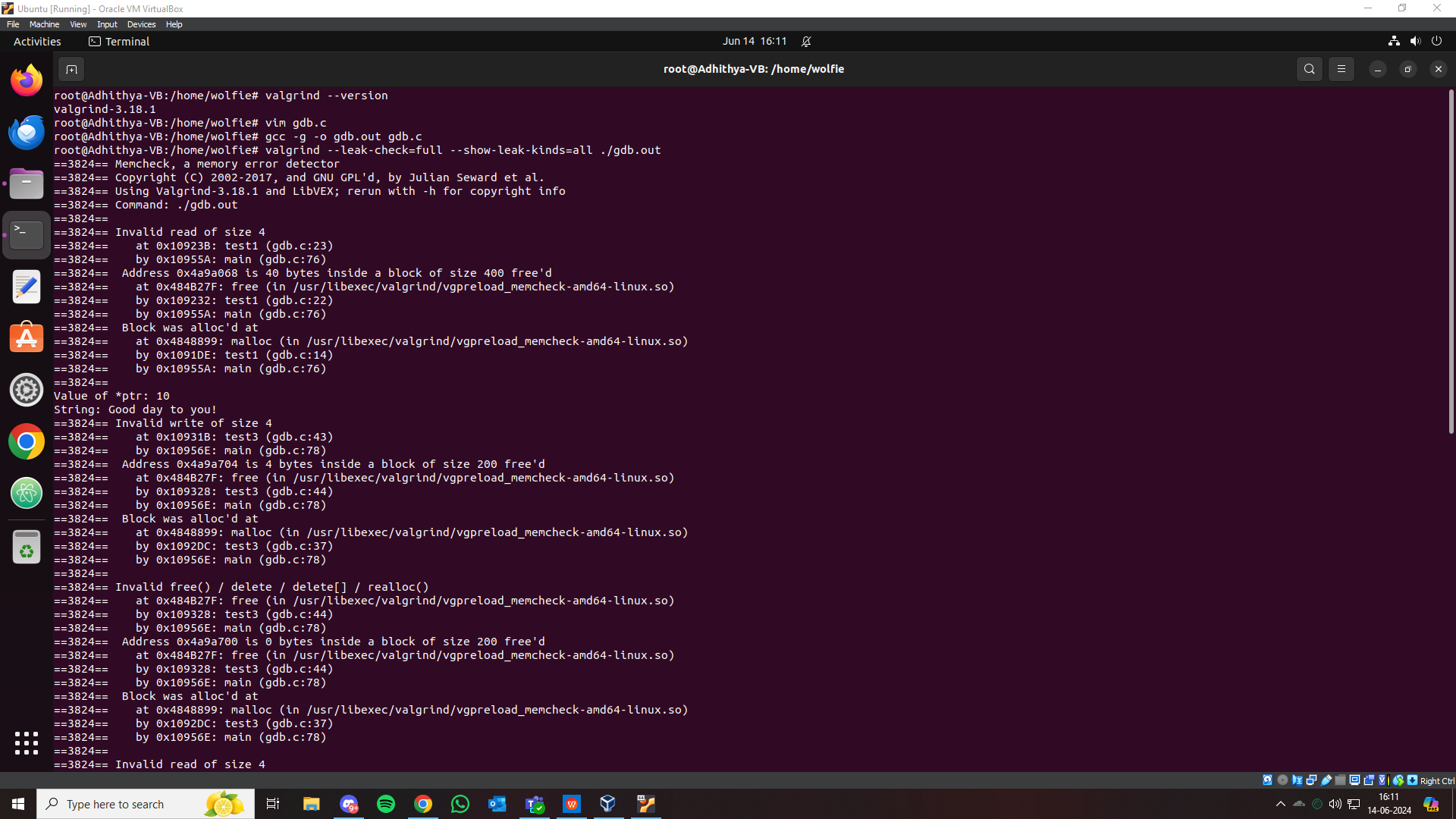
>vim <filename>.c

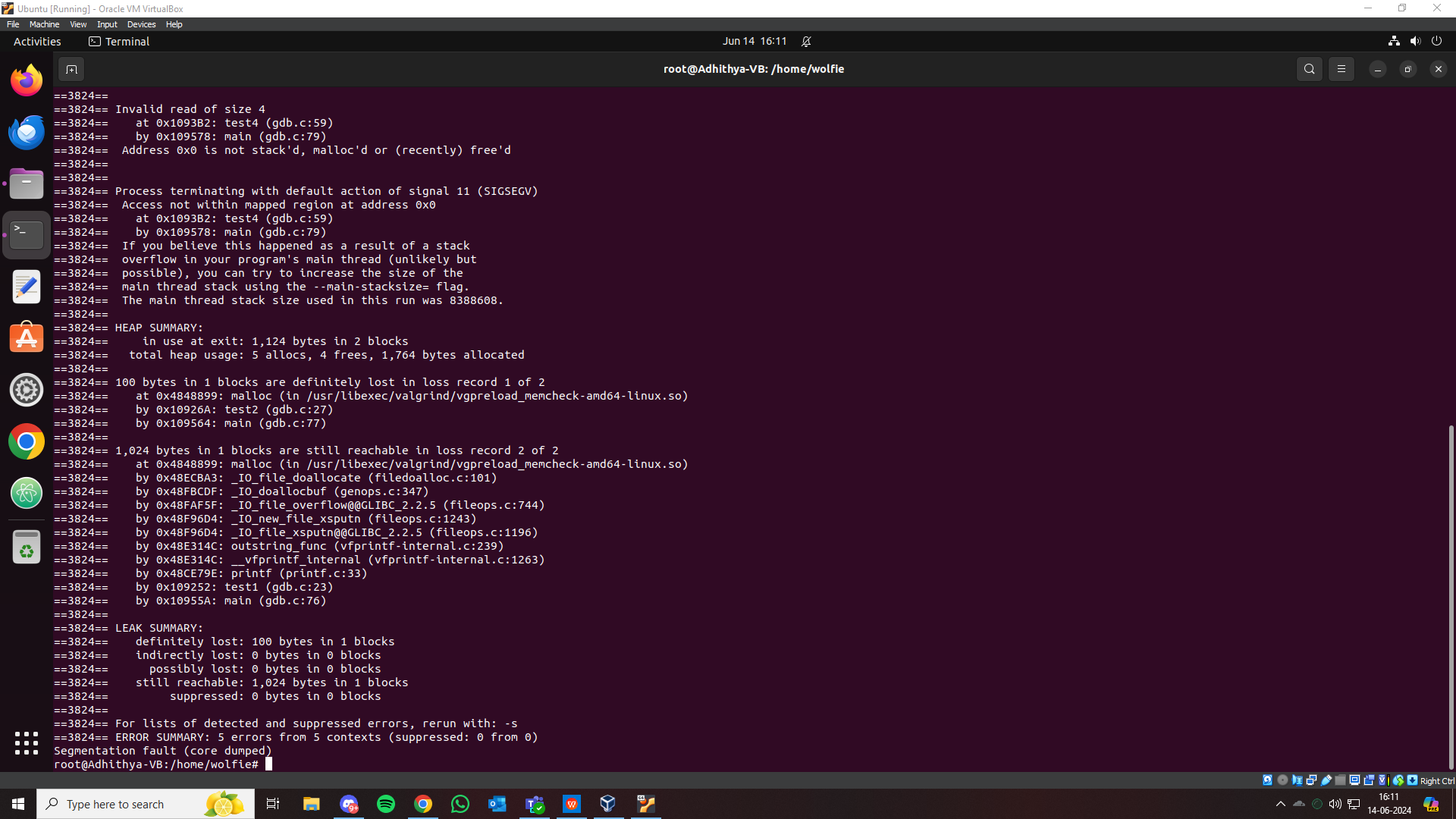
>gcc -g -o <filename>.out <filename>.c

>valgrind --leak-check=full --show-leak-kinds=all ./<filename>.out

--leak-check=full // provides information about memory leaks.

--show-leak-kinds=all //list all memory leak types in end.





**Some optional flags in valgrind include**

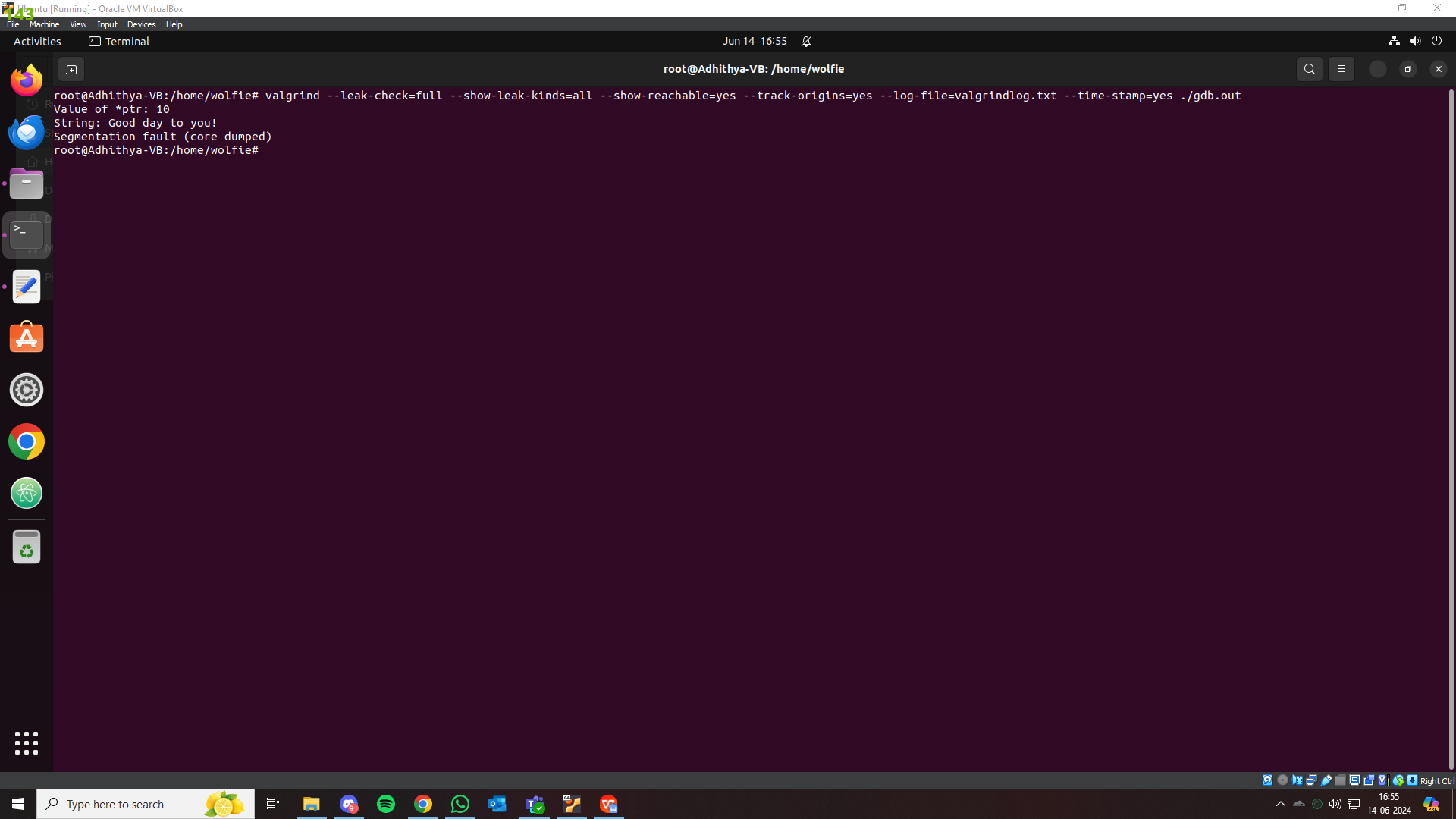
--leak-check= yes/no/summary/full // checks for memory leaks.

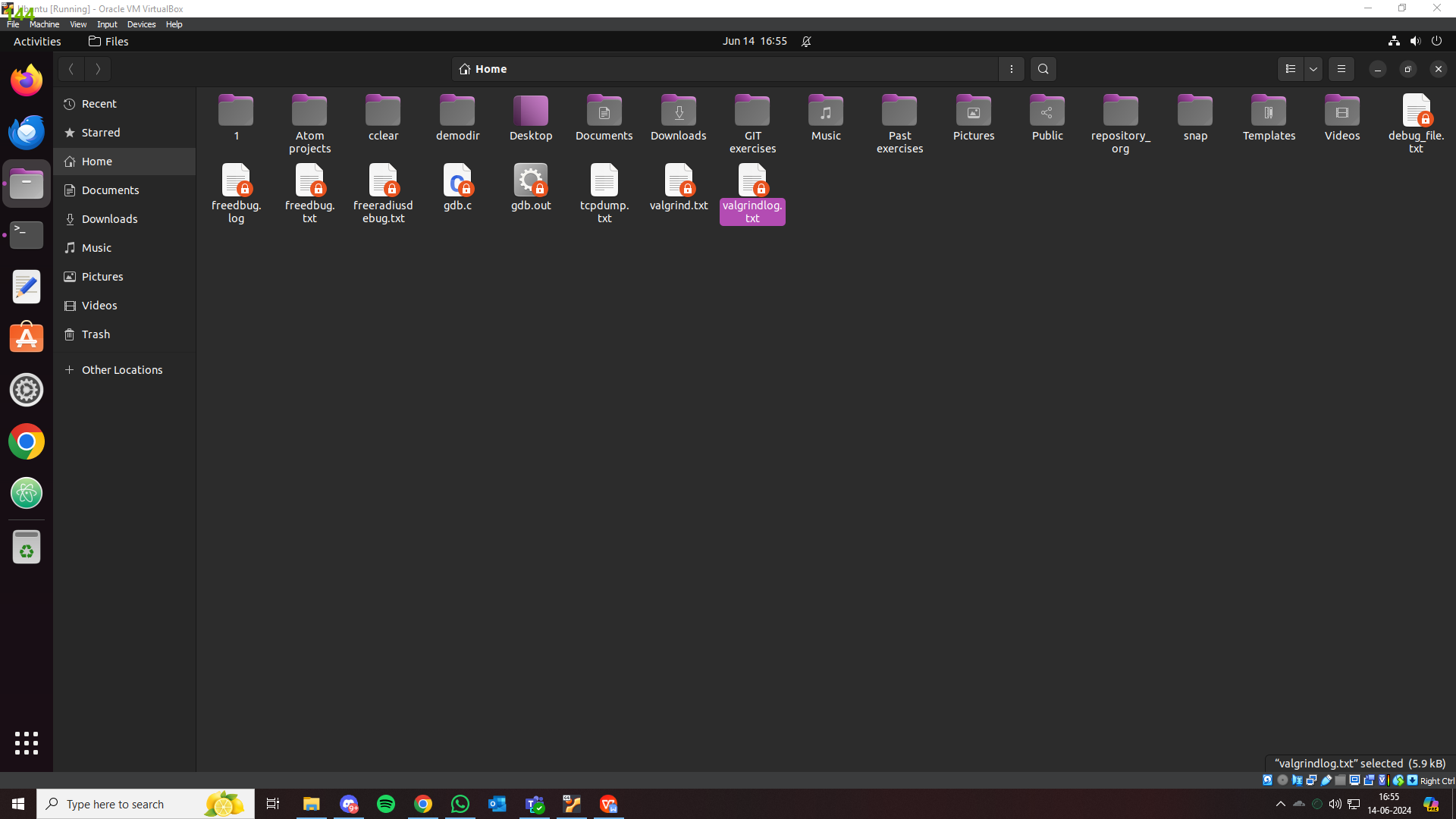
--show-reachable= yes/no // reports whether memory leaks are reachable.

--track-origins= yes/no //track uninitialized variables.

--log-file= <file> //create a log file.

--time-stamp= yes/no // add time stamp to log messages.





1. With the same program, using GDB, set breakpoints, run the program, list the code, run from one breakpoint to another, print the value of variables while execution, check assemble code, disable breakpoints, check registers info, explore optional flags.

Answer:

**GDB COMMANDS**

> sudo bash // become su

>apt-get install gdb

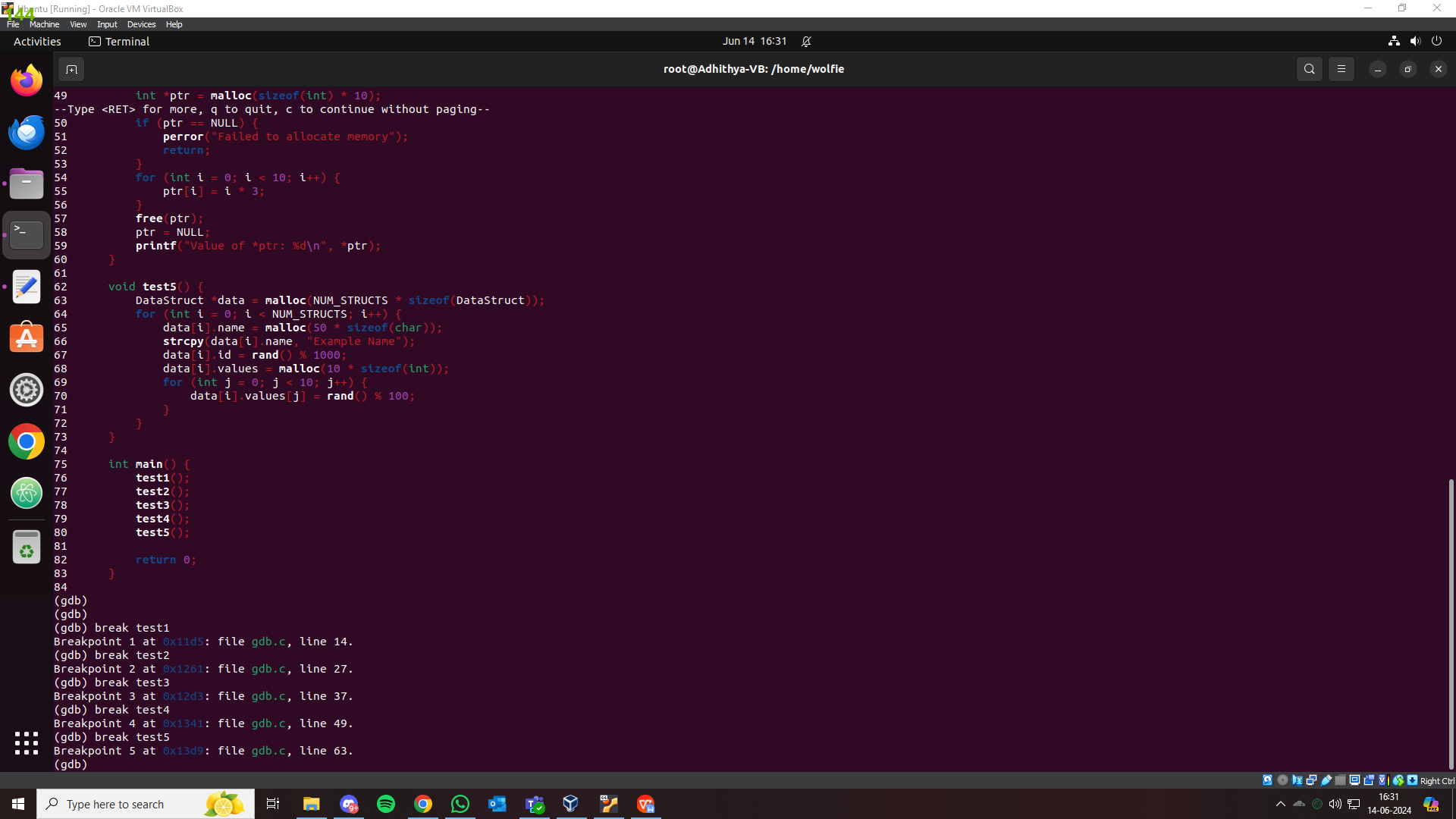
> gdb ./<filename>.out

1. **set breakpoints**

Answer:

> break <line no.>

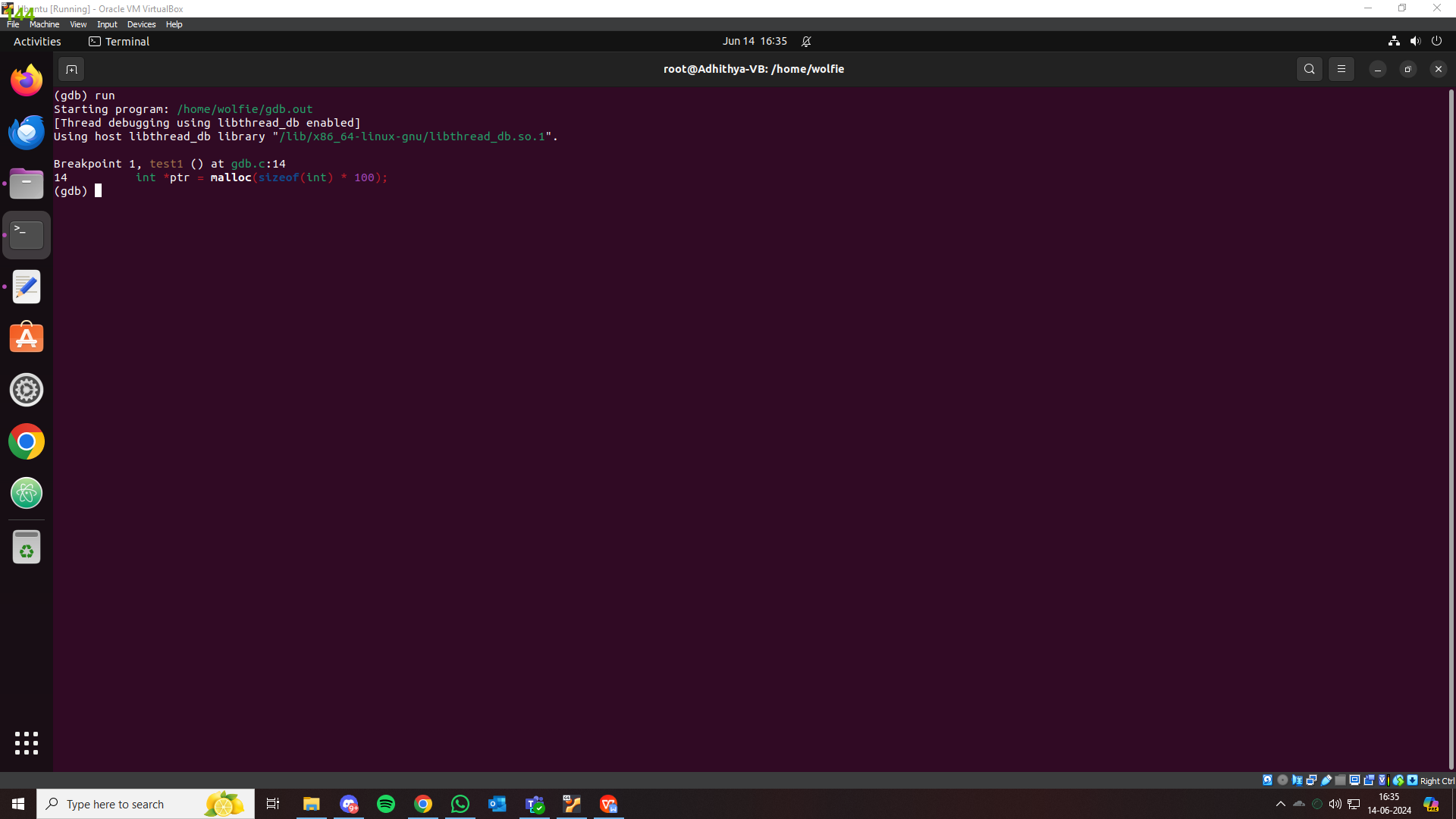
> break <function name>



1. **run the program**

Answer:

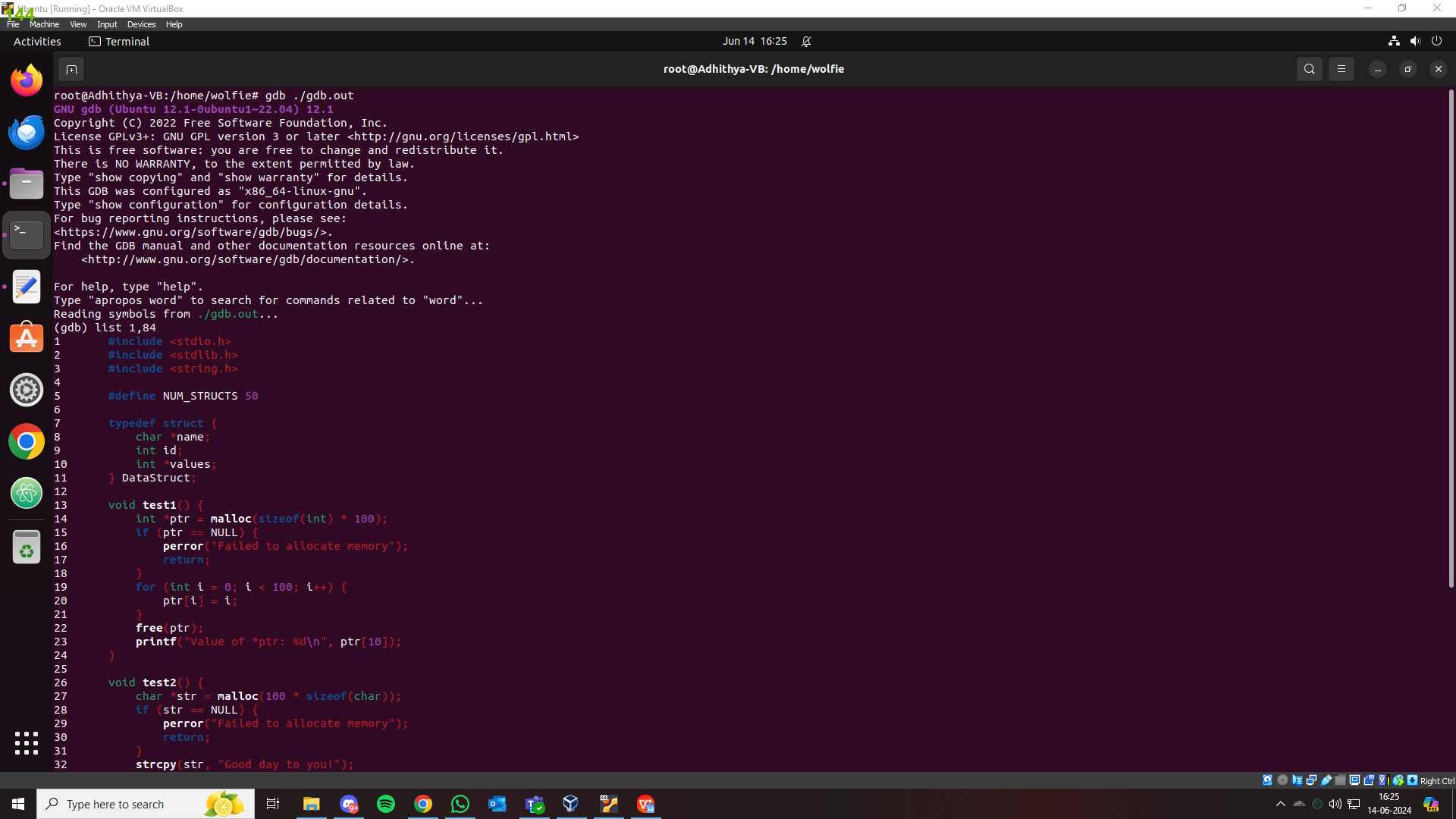
> run



1. **list the code**

Answer:

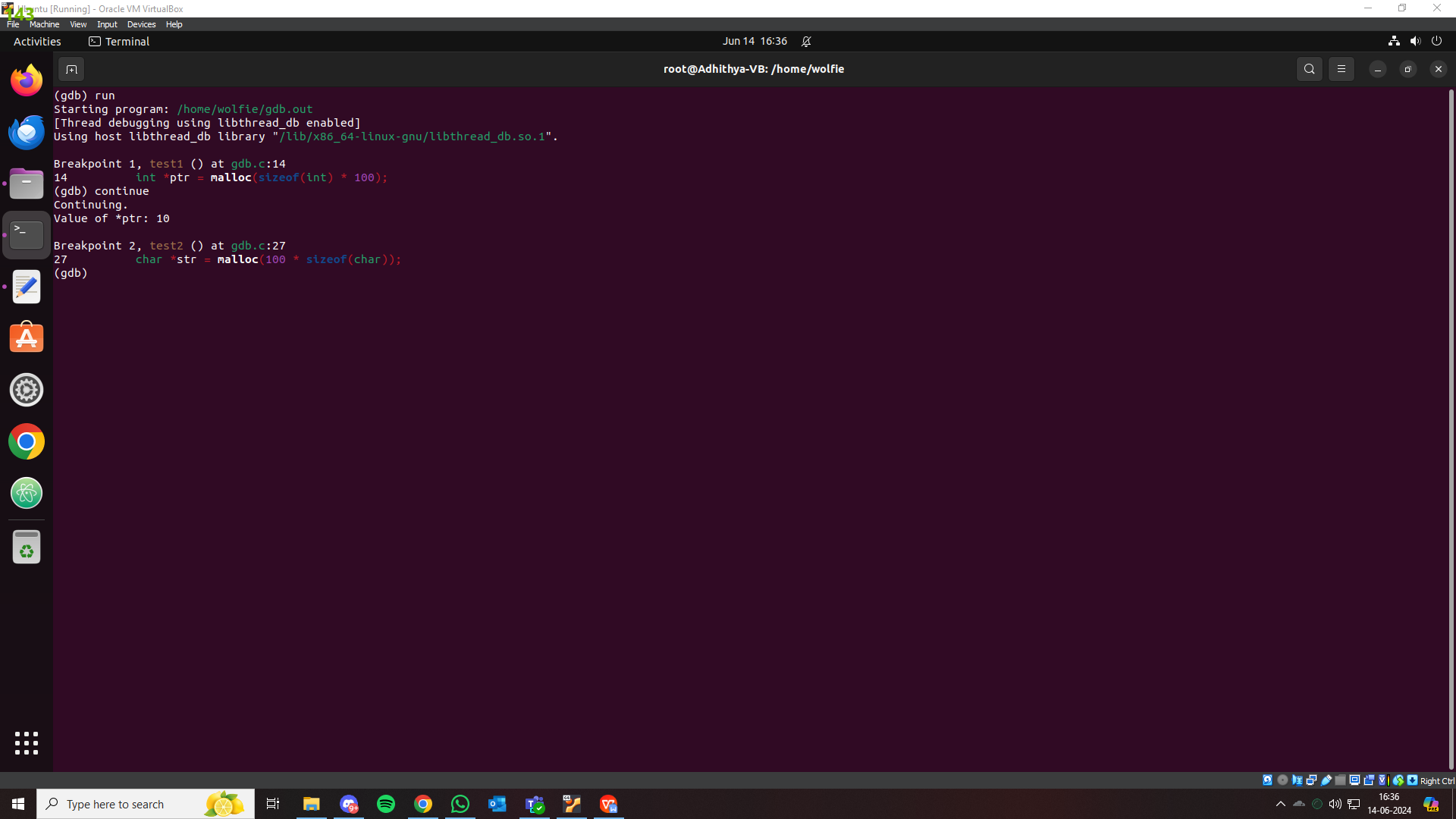
> list a,b //a starting line ,b last line



1. **run from one breakpoint to another**

Answer:

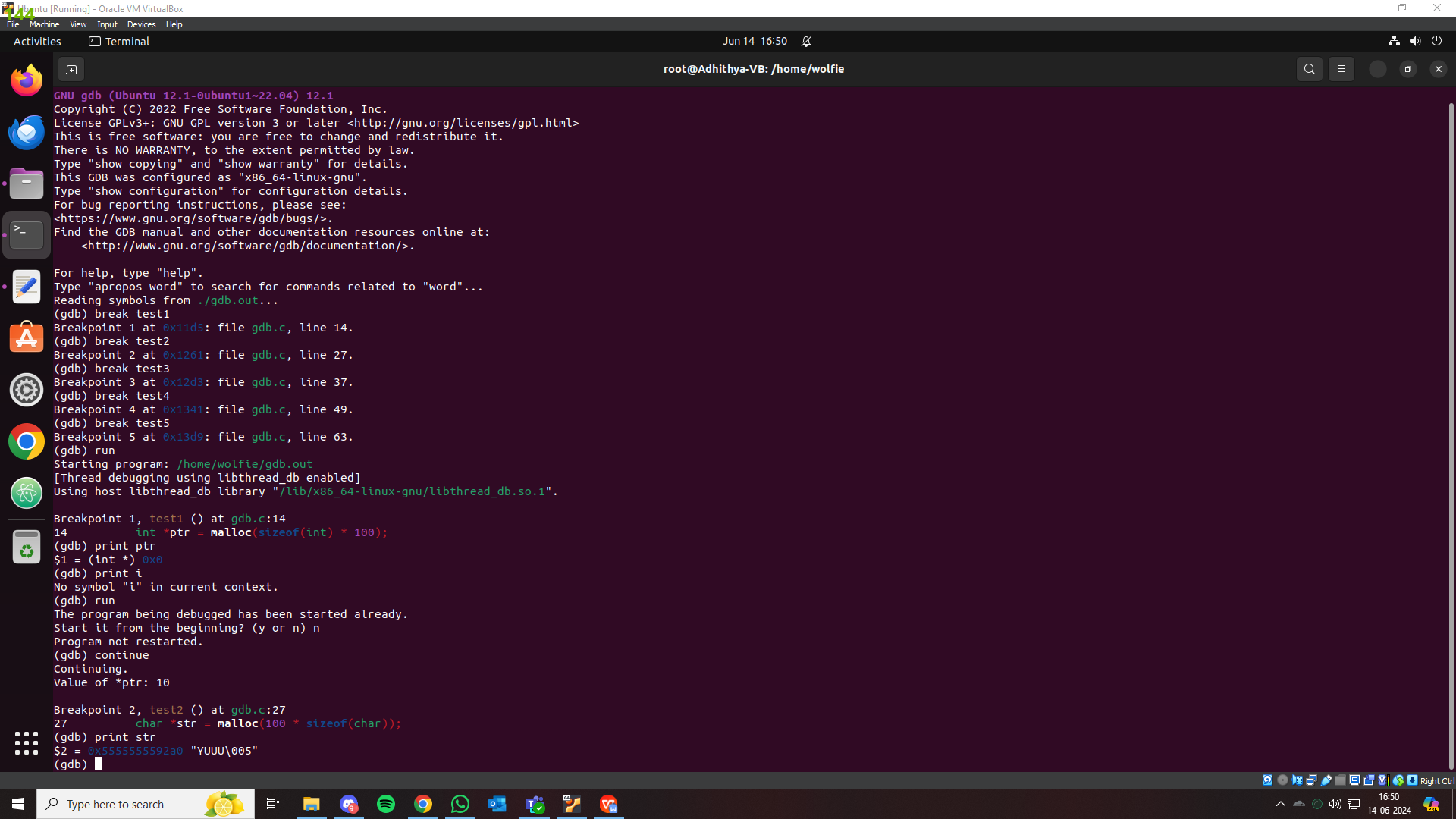
>continue



1. **print the value of variables while execution**

Answer:

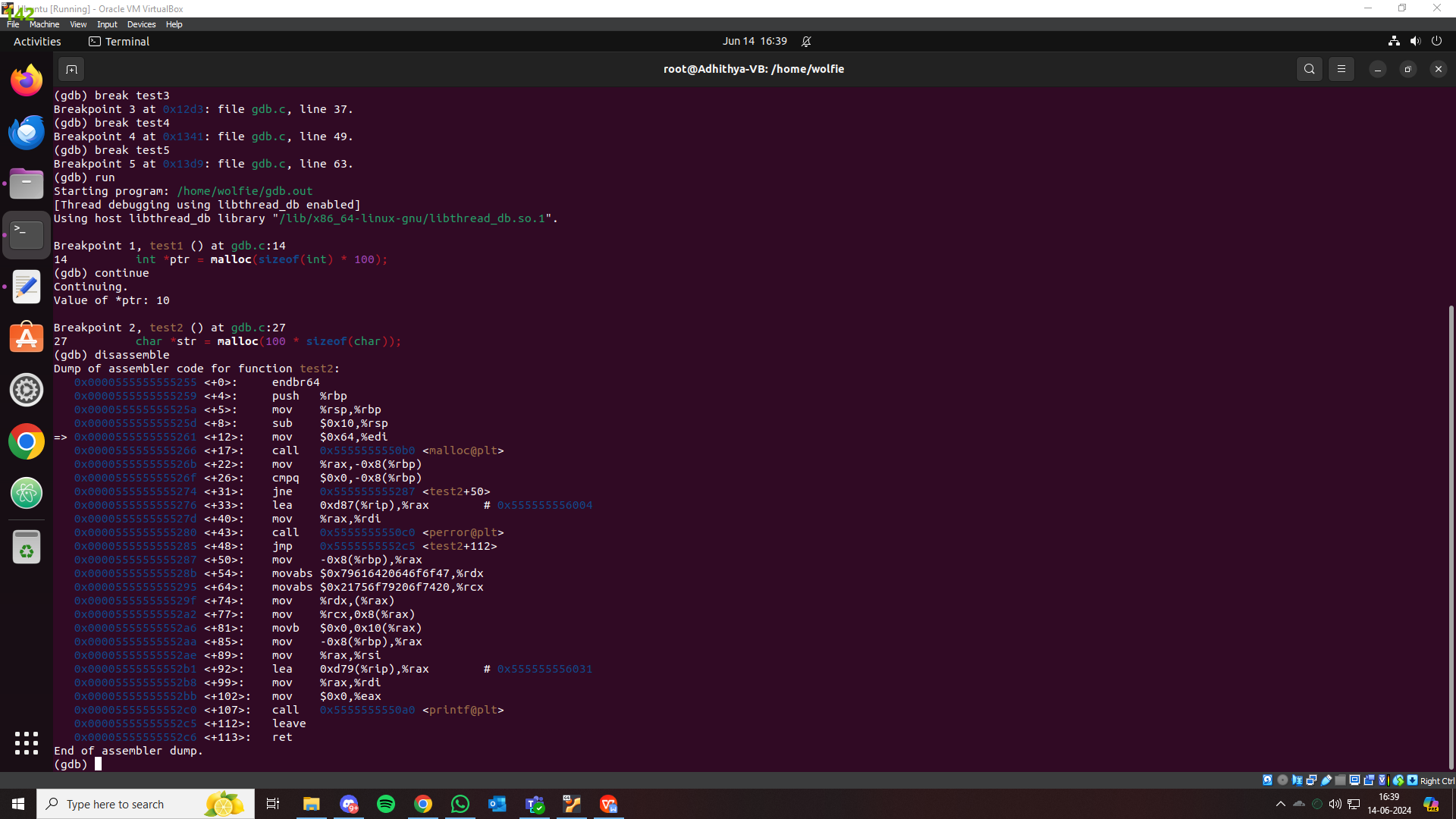
> print <var\_name>



1. **check assemble code**

Answer:

>disassemble

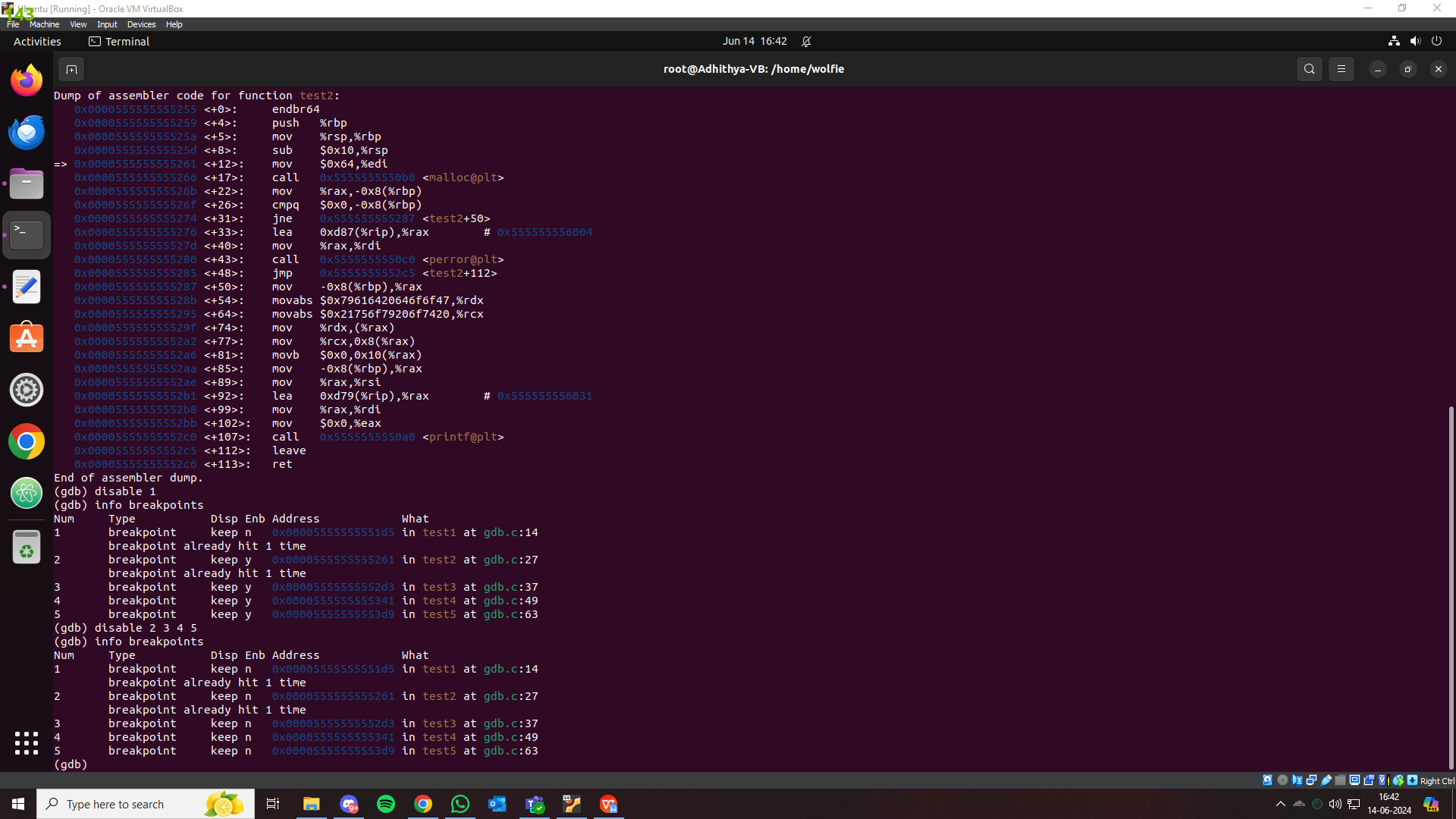


1. **disable breakpoints**

Answer:

> disable <breakpoint no.>

>info breakpoints //check whether disabled



1. **check registers info**

Answer:

> info registers

