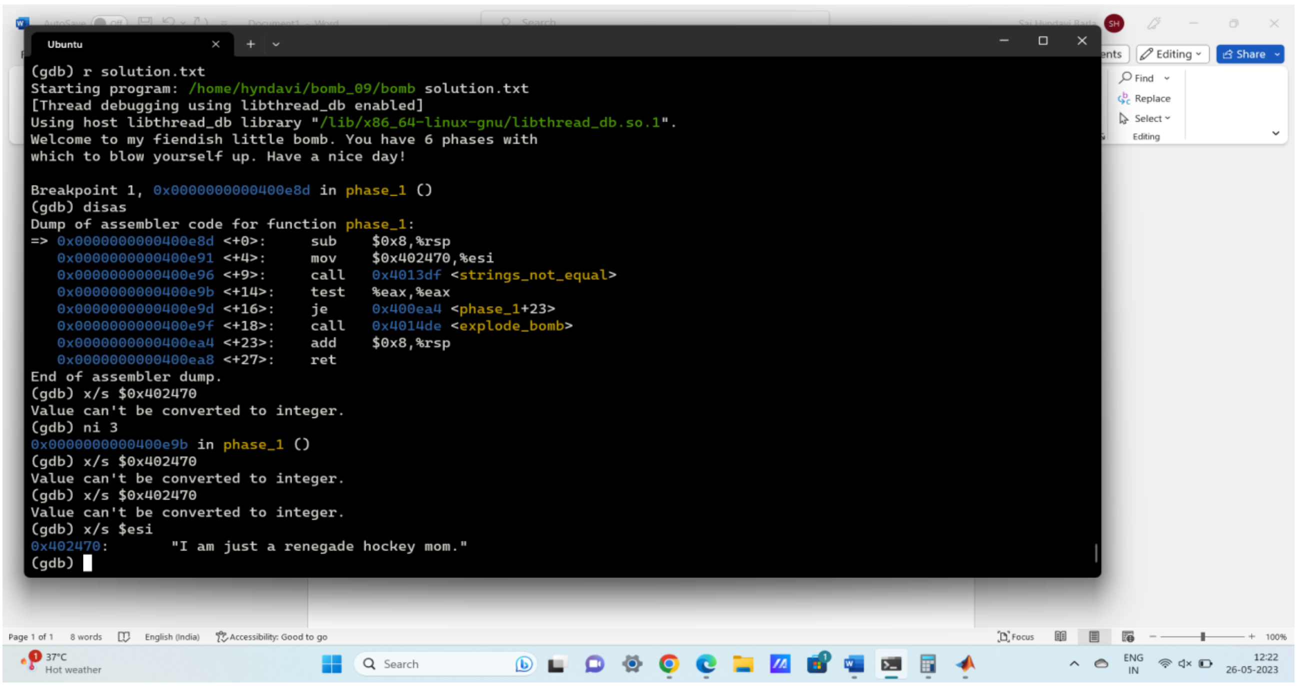
BOMB LAB SECTION – 1

26-05-2023

S20220010034 (bomb-09)

Phase-1

Use GDB bomb command to start the GDB debugger in the terminal

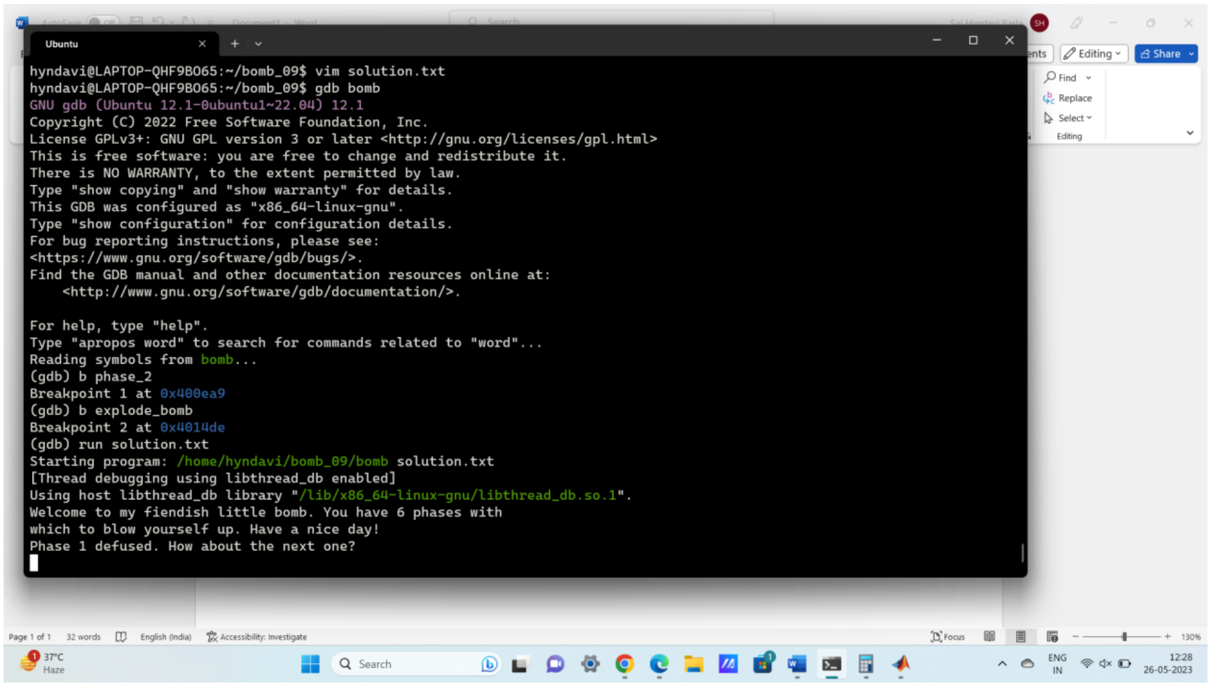


```
(gdb) r solution.txt
Starting program: /home/hyndavi/bomb_09/bomb solution.txt
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!

Breakpoint 1, 0x0000000000400e8d in phase_1 ()
(gdb) disas
Dump of assembler code for function phase_1:
=> 0x0000000000400e8d <+0>:     sub    $0x8,%rsp
   0x0000000000400e91 <+4>:     mov    $0x402470,%esi
   0x0000000000400e96 <+9>:     call   0x4013df <strings_not_equal>
   0x0000000000400e9b <+14>:    test   %eax,%eax
   0x0000000000400e9d <+16>:    je     0x400ea4 <phase_1+23>
   0x0000000000400e9f <+18>:    call   0x4014de <explode_bomb>
   0x0000000000400ea4 <+23>:    add    $0x8,%rsp
   0x0000000000400ea8 <+27>:    ret
End of assembler dump.
(gdb) x/s $0x402470
Value can't be converted to integer.
(gdb) ni 3
0x0000000000400e9b in phase_1 ()
(gdb) x/s $0x402470
Value can't be converted to integer.
(gdb) x/s $0x402470
Value can't be converted to integer.
(gdb) x/s $esi
0x402470:       "I am just a renegade hockey mom."
(gdb)
```

Check the location 0x402470

This string is compared in functions <string_not _equal> .if it is equal with given input ,then bomb wil defuse.

Save this string in solution.txt file and then we run it again.



```
hyndavi@LAPTOP-QHF9B065:~/bomb_09$ vim solution.txt
hyndavi@LAPTOP-QHF9B065:~/bomb_09$ gdb bomb
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04) 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 bomb...
(gdb) b phase_2
Breakpoint 1 at 0x400ea9
(gdb) b explode_bomb
Breakpoint 2 at 0x4014de
(gdb) run solution.txt
Starting program: /home/hyndavi/bomb_09/bomb solution.txt
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
Phase 1 defused. How about the next one?
```

Phase – 2

Here this function is taking six inputs ,and first input is stored in (%rsp)

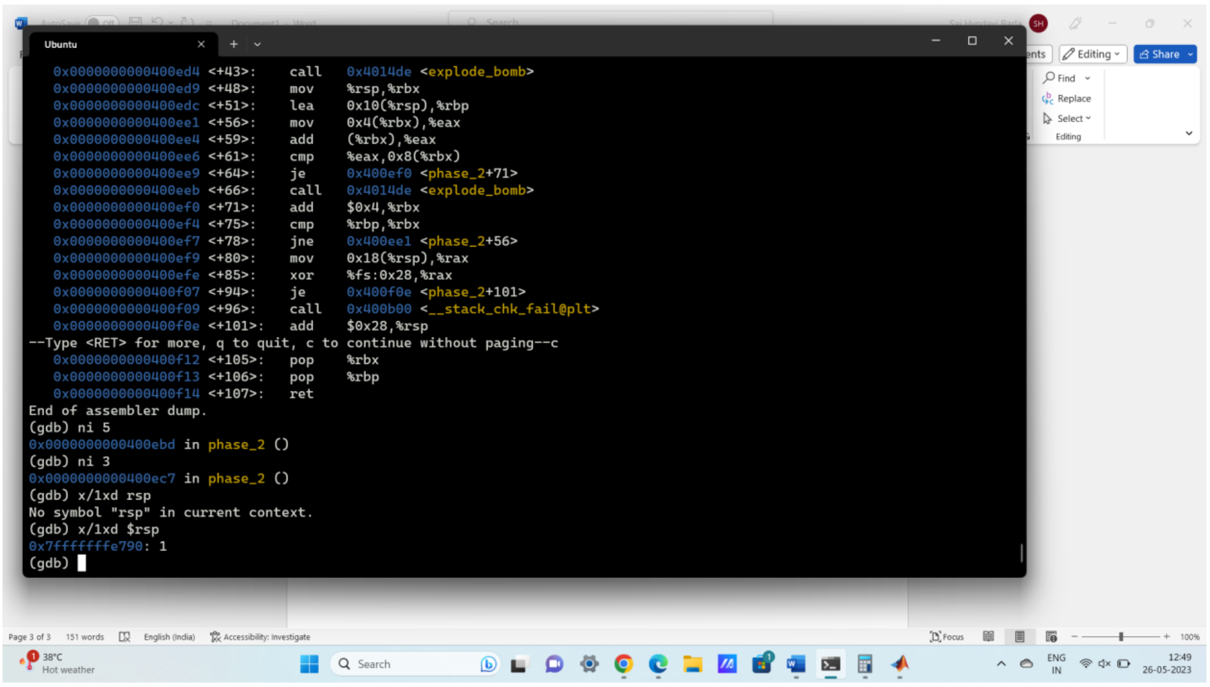
Here first input being compared with 0 ,if its not equal bomb will explode,so first input is 0.

Here second input is being compared with 1,if it is also not equal bomb will explode,so second input is 1.

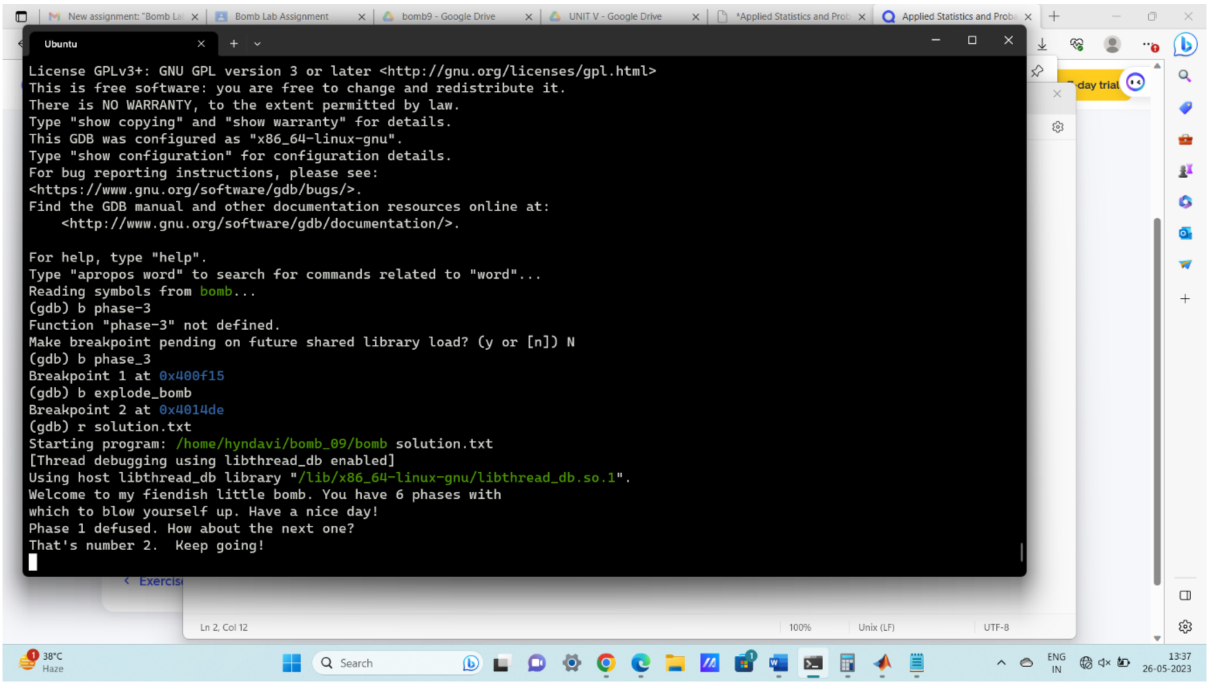
After ,if we carefully observe the first 2 inputs being added and compared with 3 input ,if its not equal bomb will explode so third input is 1.

Like this key for the phase 2 is 0 1 1 2 3 5

Save this solution.txt then we run it.



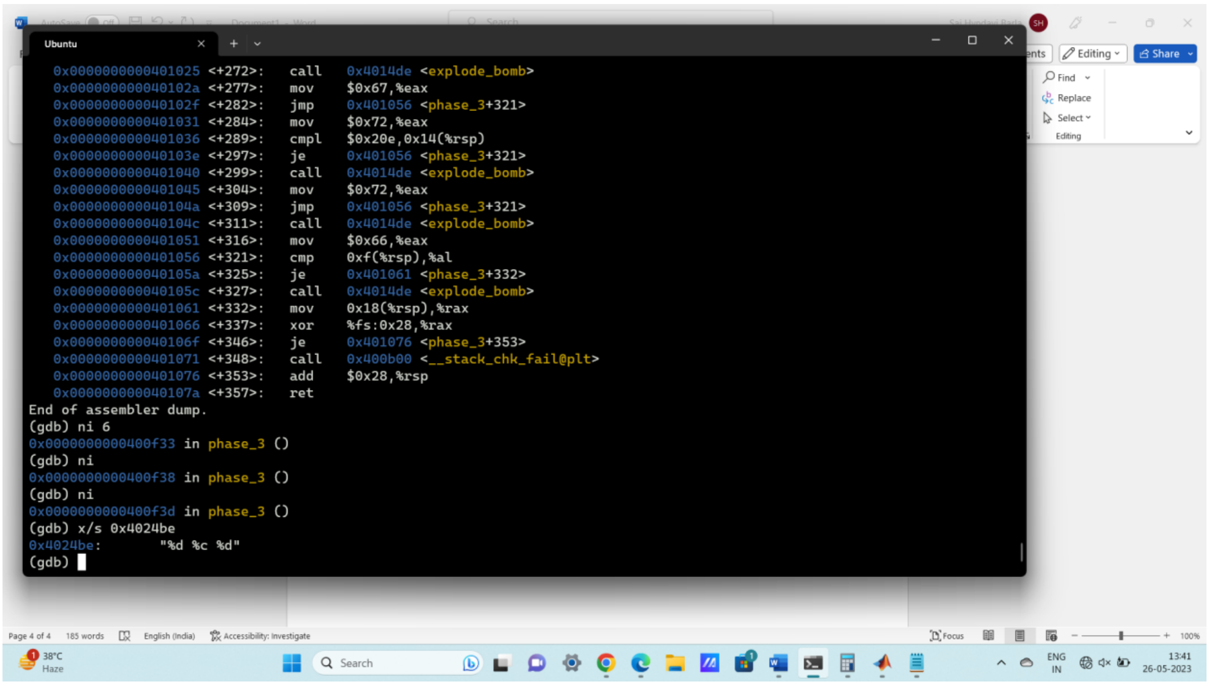
```
   0x0000000000400ed4 <+43>:    call   0x4014de <explode_bomb>
   0x0000000000400ed9 <+48>:    mov    %rsp,%rbx
   0x0000000000400edc <+51>:    lea    0x10(%rsp),%rbp
   0x0000000000400ee1 <+56>:    mov    0x4(%rbx),%eax
   0x0000000000400ee4 <+59>:    add    (%rbx),%eax
   0x0000000000400ee6 <+61>:    cmp    %eax,0x8(%rbx)
   0x0000000000400ee9 <+64>:    je     0x400ef0 <phase_2+71>
   0x0000000000400eeb <+66>:    call   0x4014de <explode_bomb>
   0x0000000000400ef0 <+71>:    add    $0x4,%rbx
   0x0000000000400ef4 <+75>:    cmp    %rbp,%rbx
   0x0000000000400ef7 <+78>:    jne    0x400ee1 <phase_2+56>
   0x0000000000400ef9 <+80>:    mov    0x18(%rsp),%rax
   0x0000000000400efe <+85>:    xor    %fs:0x28,%rax
   0x0000000000400f07 <+94>:    je     0x400f0e <phase_2+101>
   0x0000000000400f09 <+96>:    call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000400f0e <+101>:   add    $0x28,%rsp
--Type <RET> for more, q to quit, c to continue without paging--c
   0x0000000000400f12 <+105>:   pop    %rbx
   0x0000000000400f13 <+106>:   pop    %rbp
   0x0000000000400f14 <+107>:   ret
End of assembler dump.
(gdb) ni 5
0x0000000000400ebd in phase_2 ()
(gdb) ni 3
0x0000000000400ec7 in phase_2 ()
(gdb) x/1xd rsp
No symbol "rsp" in current context.
(gdb) x/1xd $rsp
0x7fffffffe790: 1
(gdb) 
```



```
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 bomb...
(gdb) b phase-3
Function "phase-3" not defined.
Make breakpoint pending on future shared library load? (y or [n]) N
(gdb) b phase_3
Breakpoint 1 at 0x400f15
(gdb) b explode_bomb
Breakpoint 2 at 0x4014de
(gdb) r solution.txt
Starting program: /home/hyndavi/bomb_09/bomb solution.txt
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
Phase 1 defused. How about the next one?
That's number 2.  Keep going!
```

Phase-3

If wee look at the assembly code,the scanf function return value is 3 ,so it is taking 3 inputs as format %d %c %d.



```
   0x0000000000401025 <+272>:   call   0x4014de <explode_bomb>
   0x000000000040102a <+277>:   mov    $0x67,%eax
   0x000000000040102f <+282>:   jmp    0x401056 <phase_3+321>
   0x0000000000401031 <+284>:   mov    $0x72,%eax
   0x0000000000401036 <+289>:   cmpl   $0x20e,0x14(%rsp)
   0x000000000040103e <+297>:   je     0x401056 <phase_3+321>
   0x0000000000401040 <+299>:   call   0x4014de <explode_bomb>
   0x0000000000401045 <+304>:   mov    $0x72,%eax
   0x000000000040104a <+309>:   jmp    0x401056 <phase_3+321>
   0x000000000040104c <+311>:   call   0x4014de <explode_bomb>
   0x0000000000401051 <+316>:   mov    $0x66,%eax
   0x0000000000401056 <+321>:   cmp    0xf(%rsp),%al
   0x000000000040105a <+325>:   je     0x401061 <phase_3+332>
   0x000000000040105c <+327>:   call   0x4014de <explode_bomb>
   0x0000000000401061 <+332>:   mov    0x18(%rsp),%rax
   0x0000000000401066 <+337>:   xor    %fs:0x28,%rax
   0x000000000040106f <+346>:   je     0x401076 <phase_3+353>
   0x0000000000401071 <+348>:   call   0x400b00 <__stack_chk_fail@plt>
   0x0000000000401076 <+353>:   add    $0x28,%rsp
   0x000000000040107a <+357>:   ret
End of assembler dump.
(gdb) ni 6
0x0000000000400f33 in phase_3 ()
(gdb) ni
0x0000000000400f38 in phase_3 ()
(gdb) ni
0x0000000000400f3d in phase_3 ()
(gdb) x/s 0x4024be
0x4024be:       "%d %c %d"
(gdb) 
```

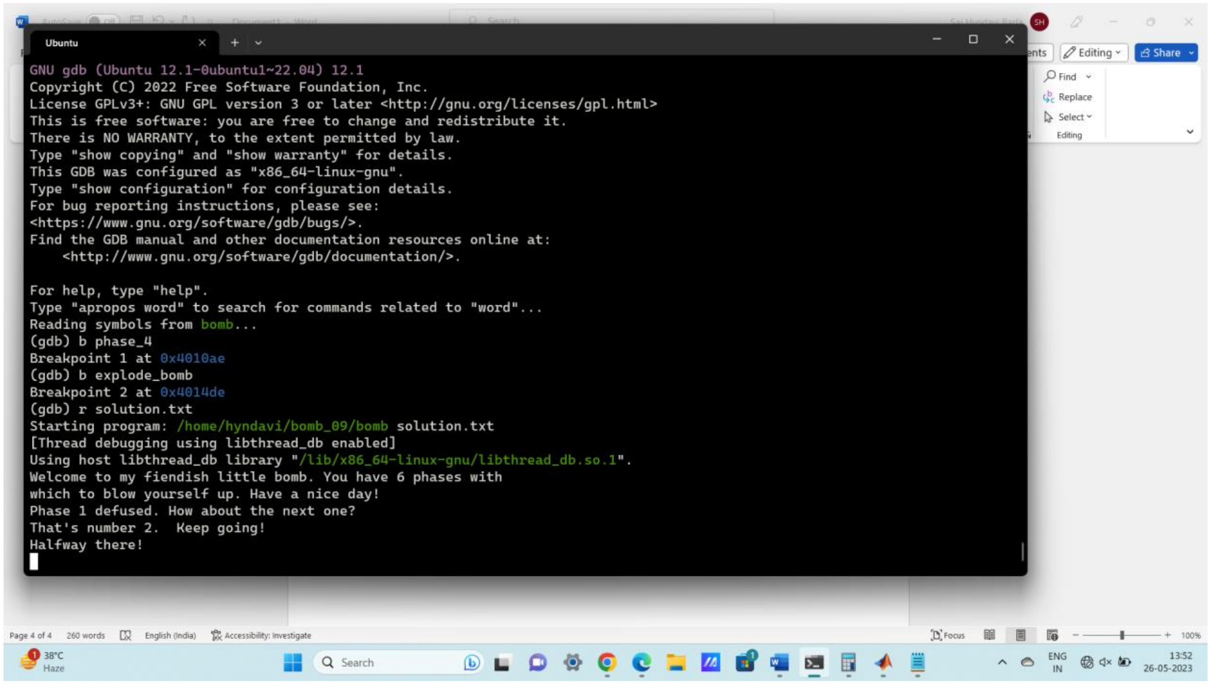
If we further look into the assembly language ,the first input is less then or equal to 7

Here jump switch cases atre there ,so based on our first input we can determine second input and third input.

In my case I will give first input as 1 then mysecond input wil be ‘u’

And third input will be 0X390 i.e 912

So solution for this phase is 1 u 912,save this then we run it again.



```
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04) 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 bomb...
(gdb) b phase_4
Breakpoint 1 at 0x4010ae
(gdb) b explode_bomb
Breakpoint 2 at 0x4014de
(gdb) r solution.txt
Starting program: /home/hyndavi/bomb_09/bomb solution.txt
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
Phase 1 defused. How about the next one?
That's number 2.  Keep going!
Halfway there!
```

Phase-4

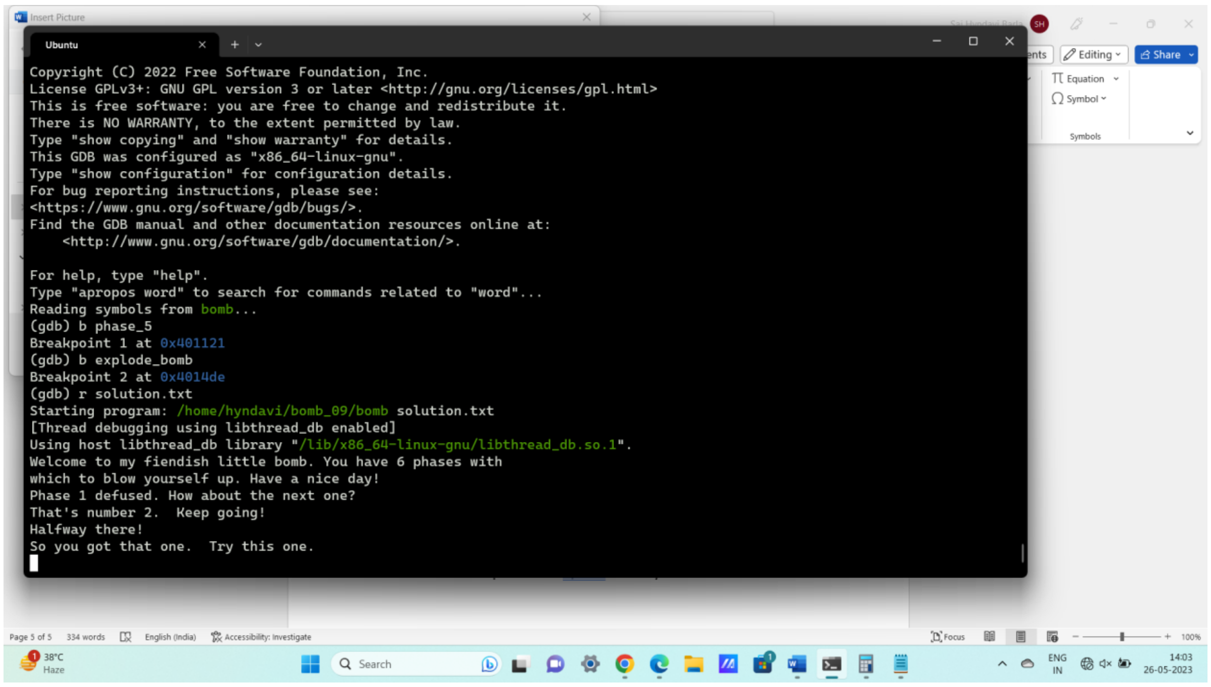
if we look into assembly, the location 0x0x40266f taking two inputs integers

we can observe that (%rsp) hold our first input and (%rsp+4) hold our second input

from the code, the line <phase+72> tells that our second input must be 7

for input 1 if we examine function 4 and based on return value, my input number one is 7

so solution for this phase is 7 7,save this keys and then we run it



```
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 bomb...
(gdb) b phase_5
Breakpoint 1 at 0x401121
(gdb) b explode_bomb
Breakpoint 2 at 0x4014de
(gdb) r solution.txt
Starting program: /home/hyndavi/bomb_09/bomb solution.txt
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
Phase 1 defused. How about the next one?
That's number 2.  Keep going!
Halfway there!
So you got that one.  Try this one.
```

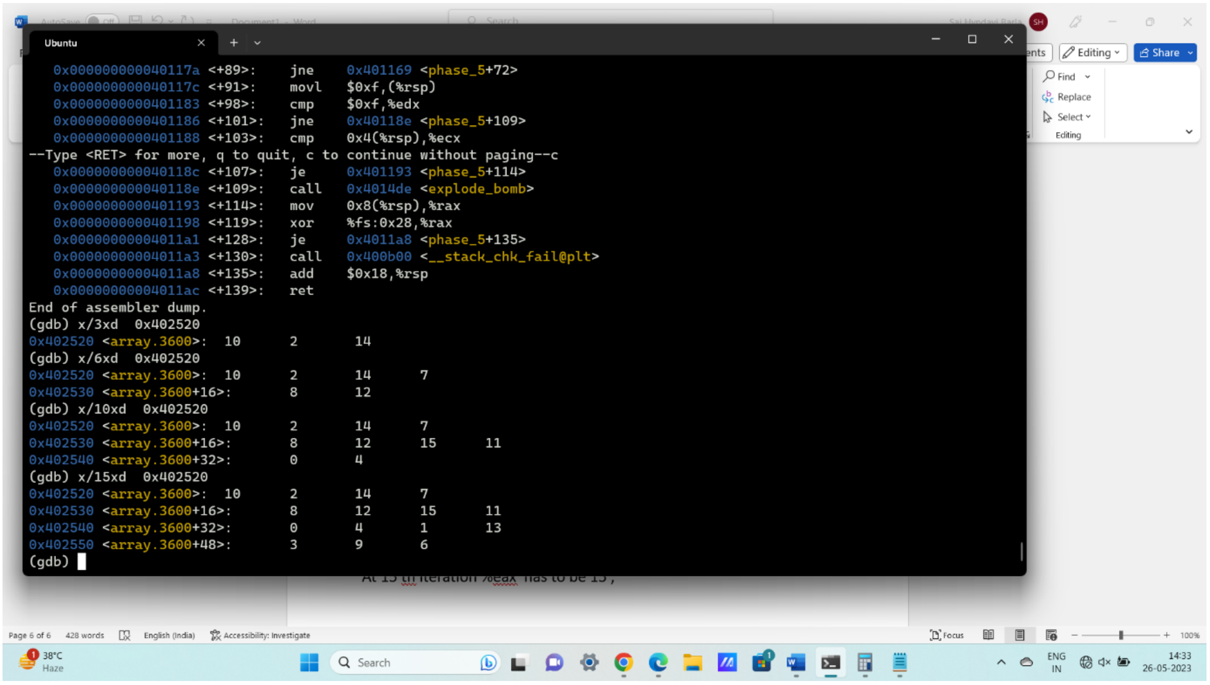
Phase -5

If we examine location $0x40266f ,our inputs must be 2 integers

If we look into code, our fist input is saved in %eax,and it is anded operation with 0xf and again the result is checking with 0xf ,if its equal, bomb will explode so our first input is not 15,31,63……

If we fast forward, there is a loop which,%ecx and %edx is initialised to zero,this loop will run 15 times as stoping condition for this loop is %edx is 15.

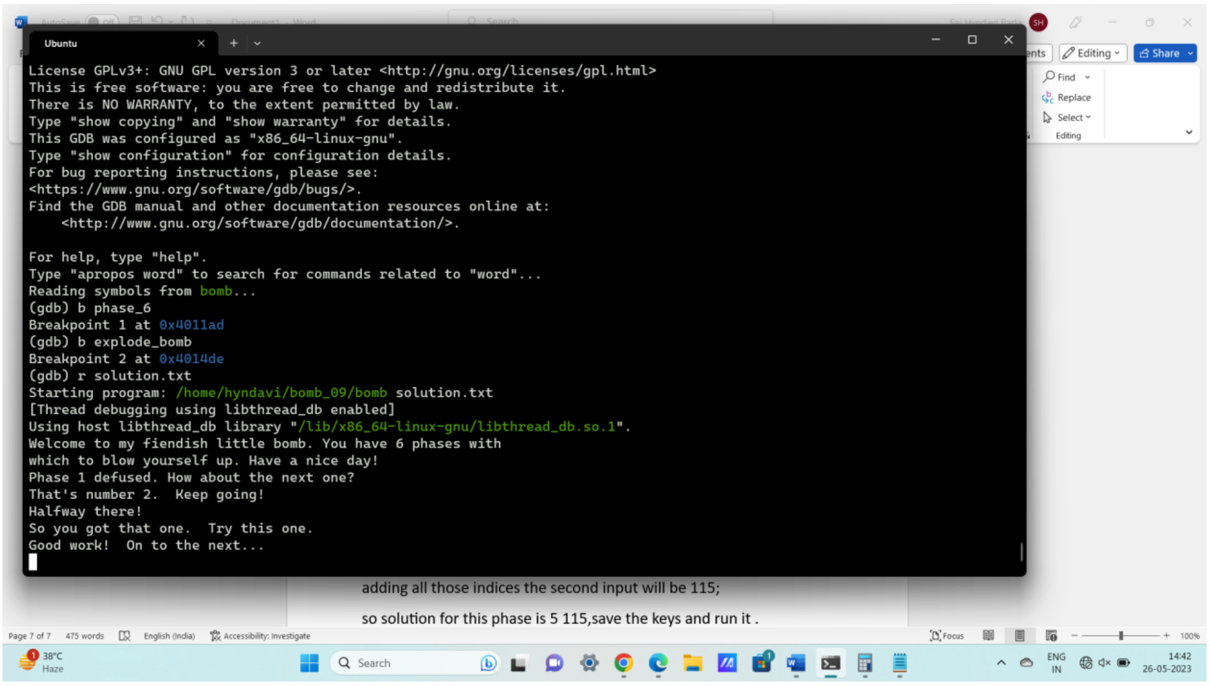
At 15 th iteration %eax has to be 15 ,this is array value of *(0x402520+((4)*(6))),if we go back to previous iteration based on eax = (x402520 +4*eax);our starting input will be 5.



```
   0x000000000040117a <+89>:    jne    0x401169 <phase_5+72>
   0x000000000040117c <+91>:    movl   $0xf,(%rsp)
   0x0000000000401183 <+98>:    cmp    $0xf,%edx
   0x0000000000401186 <+101>:   jne    0x40118e <phase_5+109>
   0x0000000000401188 <+103>:   cmp    0x4(%rsp),%ecx
--Type <RET> for more, q to quit, c to continue without paging--c
   0x000000000040118c <+107>:   je     0x401193 <phase_5+114>
   0x000000000040118e <+109>:   call   0x4014de <explode_bomb>
   0x0000000000401193 <+114>:   mov    0x8(%rsp),%rax
   0x0000000000401198 <+119>:   xor    %fs:0x28,%rax
   0x00000000004011a1 <+128>:   je     0x4011a8 <phase_5+135>
   0x00000000004011a3 <+130>:   call   0x400b00 <__stack_chk_fail@plt>
   0x00000000004011a8 <+135>:   add    $0x18,%rsp
   0x00000000004011ac <+139>:   ret
End of assembler dump.
(gdb) x/3xd  0x402520
0x402520 <array.3600>:  10      2       14
(gdb) x/6xd  0x402520
0x402520 <array.3600>:  10      2       14      7
0x402530 <array.3600+16>:       8       12
(gdb) x/10xd  0x402520
0x402520 <array.3600>:  10      2       14      7
0x402530 <array.3600+16>:       8       12      15      11
0x402540 <array.3600+32>:       0       4
(gdb) x/15xd  0x402520
0x402520 <array.3600>:  10      2       14      7
0x402530 <array.3600+16>:       8       12      15      11
0x402540 <array.3600+32>:       0       4       1       13
0x402550 <array.3600+48>:       3       9       6
(gdb)
```

adding all those indices the second input will be 115;

so solution for this phase is 5 115,save the keys and run it .



```
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 bomb...
(gdb) b phase_6
Breakpoint 1 at 0x4011ad
(gdb) b explode_bomb
Breakpoint 2 at 0x4014de
(gdb) r solution.txt
Starting program: /home/hyndavi/bomb_09/bomb solution.txt
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
Phase 1 defused. How about the next one?
That's number 2.  Keep going!
Halfway there!
So you got that one.  Try this one.
Good work!  On to the next...
```

adding all those indices the second input will be 115;

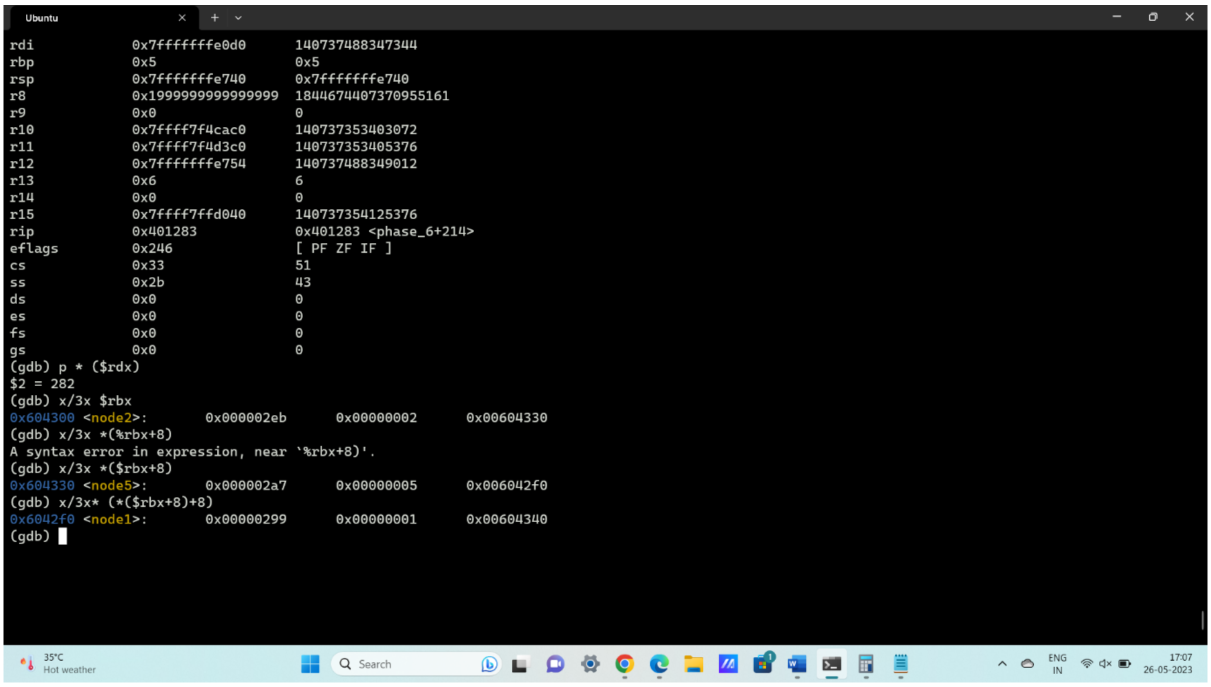
so solution for this phase is 5 115,save the keys and run it .

Phase – 6

If wee examine the assembly language ,we can see read_six numbers function,so function taking six inputs.

If wee look care fully there is 2 for loops in upper half code ,based on that our inputs must be different and range of inputs between 1 to 6

If we further go into code we will crash on into nodes

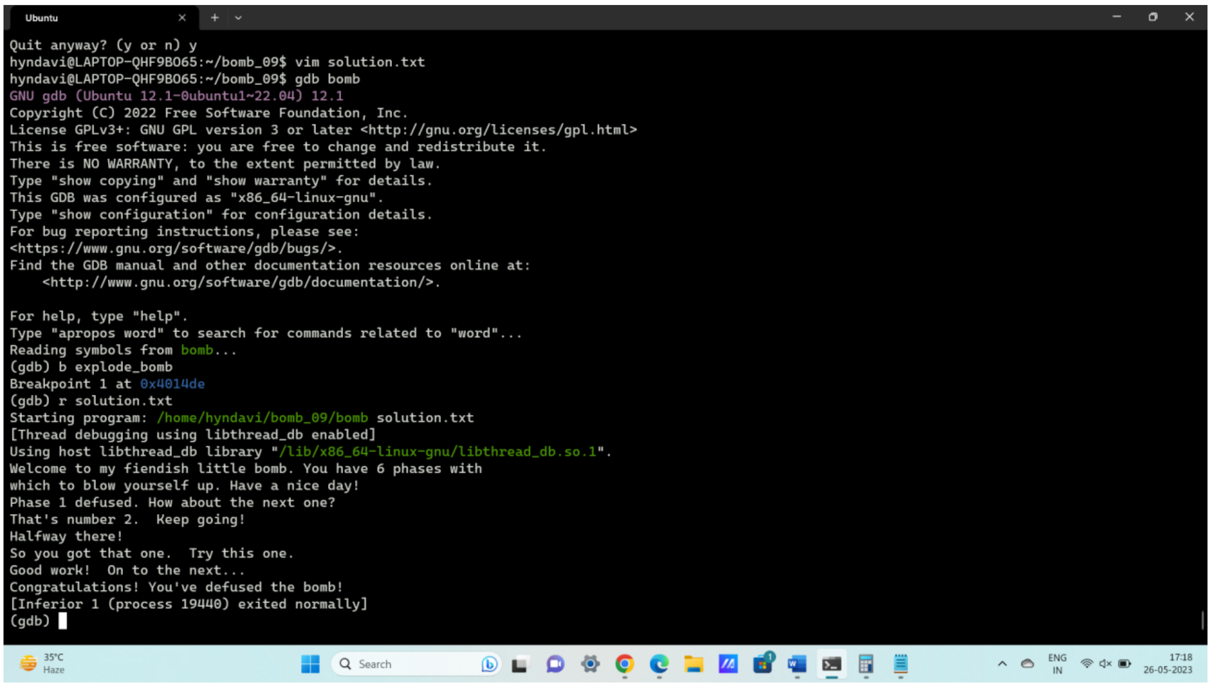


```
rdi            0x7fffffffe0d0      140737488347344
rbp            0x5                 0x5
rsp            0x7fffffffe740      0x7fffffffe740
r8             0x1999999999999999  1844674407370955161
r9             0x0                 0
r10            0x7ffff7f4cac0      140737353403072
r11            0x7ffff7f4d3c0      140737353405376
r12            0x7fffffffe754      140737488349012
r13            0x6                 6
r14            0x0                 0
r15            0x7ffff7ffd040      140737354125376
rip            0x401283            0x401283 <phase_6+214>
eflags         0x246               [ PF ZF IF ]
cs             0x33                51
ss             0x2b                43
ds             0x0                 0
es             0x0                 0
fs             0x0                 0
gs             0x0                 0
(gdb) p * ($rdx)
$2 = 282
(gdb) x/3x $rbx
0x604300 <node2>:       0x000002eb      0x00000002      0x00604330
(gdb) x/3x *(%rbx+8)
A syntax error in expression, near `%rbx+8)'.
(gdb) x/3x *($rbx+8)
0x604330 <node5>:       0x000002a7      0x00000005      0x006042f0
(gdb) x/3x* (*($rbx+8)+8)
0x6042f0 <node1>:       0x00000299      0x00000001      0x00604340
(gdb) 
```

If we look into last part of assembly code,the code is checking whether the first node in linked list having higher value then second node ,if not it will explode.

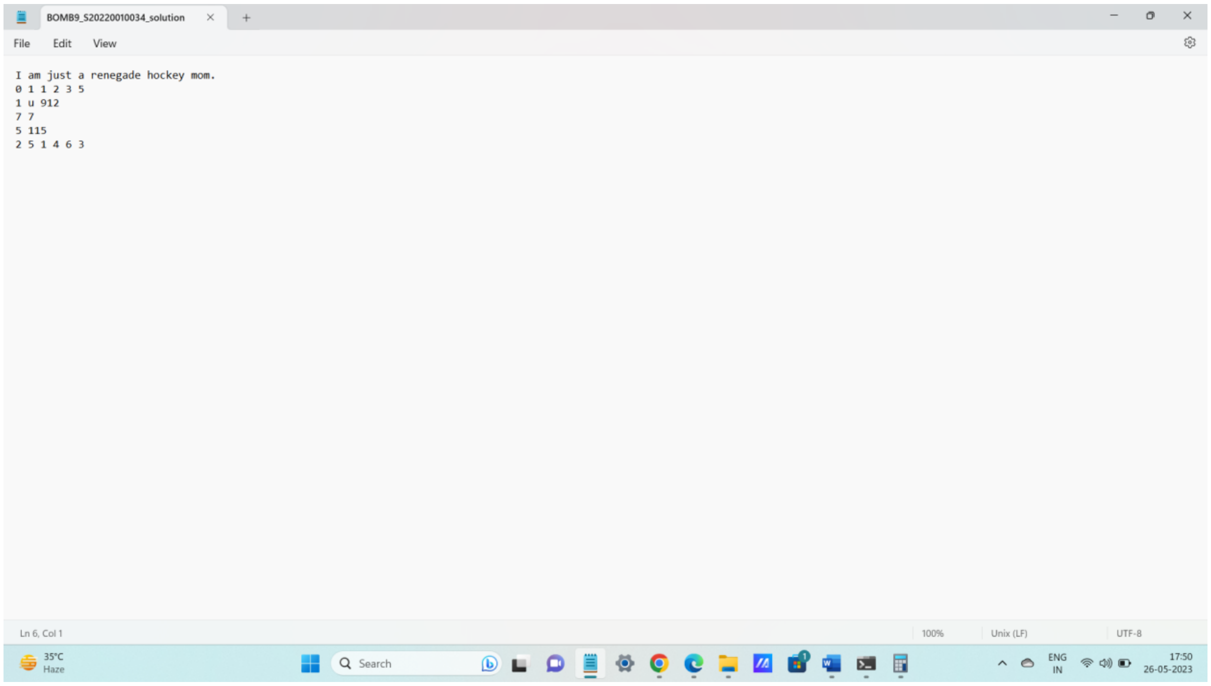
So based on that, our input should be will be 2 5 1 4 6 3

Save this values in txt file and then we run it.



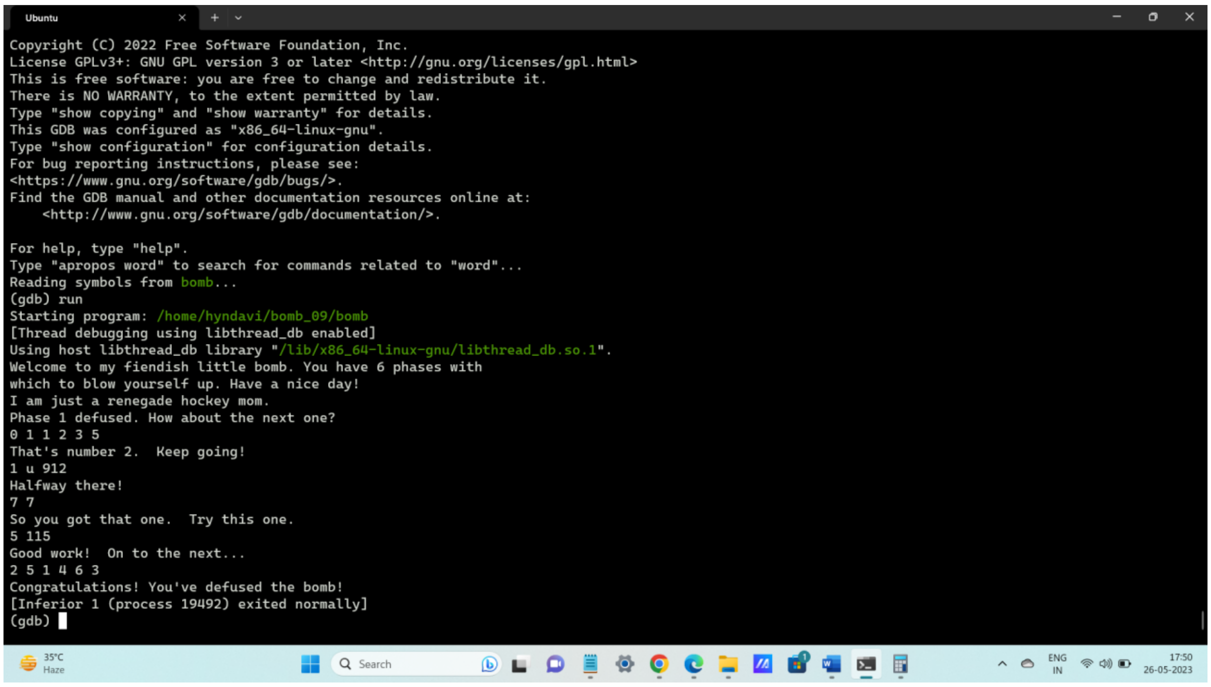
```
Quit anyway? (y or n) y
hyndavi@LAPTOP-QHF9B065:~/bomb_09$ vim solution.txt
hyndavi@LAPTOP-QHF9B065:~/bomb_09$ gdb bomb
GNU gdb (Ubuntu 12.1-0ubuntu1~22.04) 12.1
Copyright (C) 2022 Free Software Foundation, Inc.
License GPLv3+: GNU GPL version 3 or later <http://gnu.org/licenses/gpl.html>
This is 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 bomb...
(gdb) b explode_bomb
Breakpoint 1 at 0x4014de
(gdb) r solution.txt
Starting program: /home/hyndavi/bomb_09/bomb solution.txt
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
Phase 1 defused. How about the next one?
That's number 2.  Keep going!
Halfway there!
So you got that one.  Try this one.
Good work!  On to the next...
Congratulations! You've defused the bomb!
[Inferior 1 (process 19440) exited normally]
(gdb)
```

Here’s the screenshot of the solution.txt file



```
I am just a renegade hockey mom.
0 1 1 2 3 5
1 u 912
7 7
5 115
2 5 1 4 6 3
```

Inputs given



```
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 bomb...
(gdb) run
Starting program: /home/hyndavi/bomb_09/bomb
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
Welcome to my fiendish little bomb. You have 6 phases with
which to blow yourself up. Have a nice day!
I am just a renegade hockey mom.
Phase 1 defused. How about the next one?
0 1 1 2 3 5
That's number 2.  Keep going!
1 u 912
Halfway there!
7 7
So you got that one.  Try this one.
5 115
Good work!  On to the next...
2 5 1 4 6 3
Congratulations! You've defused the bomb!
[Inferior 1 (process 19492) exited normally]
(gdb)
```