

```

keylor@keylor-VirtualBox:~/Documents/github-fundamentals-KeylorSoto19/Examen_1$ gdb criba
GNU gdb (Ubuntu 9.2-0ubuntu1~20.04) 9.2
Copyright (C) 2020 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/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 criba...
(no debugging symbols found in criba)
(gdb) run
Starting program: /home/keylor/Documents/github-fundamentals-KeylorSoto19/Examen_1/criba

Program received signal SIGSEGV, Segmentation fault.
0x0000000000401068 in while ()
(gdb) disas conseguir_primos
Dump of assembler code for function conseguir_primos:
   0x0000000000401050 <+0>:    mov     (%ebx),%r8d
   0x0000000000401054 <+4>:    cmp     $0x0,%r8d
   0x0000000000401058 <+8>:    je      0x40107a <fin_while>
   0x000000000040105a <+10>:   mov     %r8d,0x402000(%rdx)
   0x0000000000401061 <+17>:   add     $0x4,%rdx
   0x0000000000401065 <+21>:   mov     %rax,%r9
End of assembler dump.
(gdb) x/13xd 0x402000
0x402000:    2      3      5      17
0x402010:   23     47    -983036  0
0x402020:    0      0      0      0
0x402030:    0
(gdb)

```

Ejecución del programa:

El programa ejecuta correctamente hasta el inicio, solo que conforme avanza el proceso, su contenido en rax (es el índice) se ensucia de forma que el programa se vuelve inaccesible. En la imagen podemos que ver la dirección 0x402000 corresponde al vector "resultado". En la imagen podemos observar como se va llenando correctamente el vector, sin embargo, luego se ensucia algún registro haciendo que no entren algunos números primos o poniendo números fuera de rango.

Explicación: El programa consta de una matriz, donde su base es copiada en rbx, y usando a rax como índice, se recorren dos ciclos, donde uno recorre la matriz y el while interno recorre los números que se tienen que tachar. Se escogió esta estrategia con la intención de minimizar el gasto de los operandos div y mul.

```

For help, type "help".
Type "apropos word" to search for commands related to "word"...
Reading symbols from criba...
(No debugging symbols found in criba)
(gdb) run
Starting program: /home/keylor/Documents/github-fundamentals-KeylorSoto19/Examen_1/criba
Inferior 1 (process 29440) exited normally]
(gdb) i r conseguir_primos
The program has no registers now.
(gdb) disas conseguir_primos
Dump of assembler code for function conseguir_primos:
   0x0000000000401050 <+0>:      mov     (%ebx),%r8d
   0x0000000000401054 <+4>:      cmp     $0x0,%r8d
   0x0000000000401058 <+8>:      je      0x40107a <fin_while>
   0x000000000040105a <+10>:     mov     %r8d,0x402000(%rdx)
   0x0000000000401061 <+17>:     add     $0x4,%rdx
   0x0000000000401065 <+21>:     mov     %rax,%r9
End of assembler dump.
(gdb) x/13xd 0x402000
0x402000:  0      0      0      0
0x402010:  0      0      0      0
0x402020:  0      0      0      0
0x402030:  0
(gdb) x/200xd 0x402000
0x402000:  0      0      0      0
0x402010:  0      0      0      0
0x402020:  0      0      0      0
0x402030:  0      0      0      0
0x402040:  0      0      0      0
0x402050:  0      0      0      0
0x402060:  0      0      0      2
0x402070:  3      4      5      6
0x402080:  7      8      9     10
0x402090: 11     12     13     14
0x4020a0: 15     16     17     18
0x4020b0: 19     20     21     22
0x4020c0: 23     24     25     26
0x4020d0: 27     28     29     30
0x4020e0: 31     32     33     34
0x4020f0: 35     36     37     38
0x402100: 39     40     41     42
0x402110: 43     44     45     46
0x402120: 47     48     49     50
0x402130: 51     52     53     54
0x402140: 55     56     57     58
0x402150: 59     60     61     62
0x402160: 63     64     65     66
0x402170: 67     68     69     70
0x402180: 71     72     73     74
0x402190: 75     76     77     78
0x4021a0: 79     80     81     82
0x4021b0: 83     84     85     86
0x4021c0: 87     88     89     90
0x4021d0: 91     92     93     94
0x4021e0: 95     96     97     98
0x4021f0: 99     100    101    Cannot access memory at address 0x4021fc
(gdb)

```

También se logró que el programa terminara sin errores, sin embargo, el vector resultante quedaba sin ningún número en él (lo que se mira más abajo es la misma matriz).