Universidade Federal de Sergipe

Interface Hardware Software Turma 03

Gabriel Teixeira Silveira

Relatório sobre depuração de código em C

Professor:
Calebe Micael de Oliveira Conceição

São Cristóvão
Julho de 2024

Abaixo, segue o código utilizado para compilação e depuração.

```
#include <stdio.h>

void trocar(int *a, int *b) {
    int temp = *a;
    *a = *b;
    *b = temp;
}

int lomutoPartition(int inicio, int fim, int *vetor) {
    int pivot = vetor[inicio];
    int mediana = inicio;// indice do ultimo elemento do primeiro segmento
    for(int i = inicio + 1; i < fim; i++) {
        if(vetor[i] < pivot) {
            mediana++;
            trocar(&vetor[mediana],&vetor[i]);
        }
    }
    trocar(&vetor[inicio],&vetor[mediana]);
    return mediana;
}

void QuickSort(int inicio, int fim, int *vetor) {
    int particao;
    if(inicio < fim) {
        particao = lomutoPartition(inicio, fim, vetor);
        QuickSort(inicio, particao, vetor);
        QuickSort(particao+1, fim, vetor);
    }
}

int main() {
    int vetor[12] = {4, 5, 6, 9, 2, 3, 1, 8, 7, 10};

    for(int i = 0 ; i < 10 ;i++) {
        printf("%d ", vetor[i]);
    }
    printf("\n");
    QuickSort(0, 10, vetor);
    for(int i = 0 ; i < 10 ;i++) {
        printf("%d ", vetor[i]);
    }
    printf("\n");
    return 0;
}
```

Primeiramente, compilei o código e testei seu funcionamento

```
gabriel-ubuntu-wsl-2@DESKTOP-V76KM6R:/mnt/c/Users/User/Programacao/IHS$ gcc
QuickSort.c -o QuickSort
gabriel-ubuntu-wsl-2@DESKTOP-V76KM6R:/mnt/c/Users/User/Programacao/IHS$ ls
 QuickSort   QuickSort.c
gabriel-ubuntu-wsl-2@DESKTOP-V76KM6R:/mnt/c/Users/User/Programacao/IHS$
./QuickSort
4 5 6 9 2 3 1 8 7 10
1 2 3 4 5 6 7 8 9 10
```

Em seguida, gerei a Árvore Sintática Abstrata (LLVM)

```
gabriel-ubuntu-wsl-2@DESKTOP-V76KM6R:/mnt/c/Users/User/Programacao/IHS$ clang -S -
Xclang -ast-dump QuickSort.c
TranslationUnitDecl 0x131aa28 <<invalid sloc>> <invalid sloc>
|-TypedefDecl 0x131b250 <<invalid sloc>> <invalid sloc> implicit __int128_t
'__int128'
| `-BuiltinType 0x131aff0 '__int128'
|-TypedefDecl 0x131b2c0 <<invalid sloc>> <invalid sloc> implicit __uint128_t
'unsigned __int128'
| `-BuiltinType 0x131b010 'unsigned __int128'
|-TypedefDecl 0x131b5c8 <<invalid sloc>> <invalid sloc> implicit
__NSConstantString 'struct __NSConstantString_tag'
| `-RecordType 0x131b3a0 'struct __NSConstantString_tag'
|   `-Record 0x131b318 '__NSConstantString_tag'
|-TypedefDecl 0x131b660 <<invalid sloc>> <invalid sloc> implicit
__builtin_ms_va_list 'char *'
| `-PointerType 0x131b620 'char *'
|   `-BuiltinType 0x131aad0 'char'
|-TypedefDecl 0x131b958 <<invalid sloc>> <invalid sloc> implicit referenced
```

```
__builtin_va_list 'struct __va_list_tag[1]'
| `-ConstantArrayType 0x131b900 'struct __va_list_tag[1]' 1
|   `-RecordType 0x131b740 'struct __va_list_tag'
|     `-Record 0x131b6b8 '__va_list_tag'
```

Em seguida, gerei a representação intermediária (LLVM)

```
gabriel-ubuntu-wsl-2@DESKTOP-V76KM6R:/mnt/c/Users/User/Programacao/IHS$ clang -S -
emit-llvm QuickSort.c
gabriel-ubuntu-wsl-2@DESKTOP-V76KM6R:/mnt/c/Users/User/Programacao/IHS$ ls
 QuickSort   QuickSort.c   QuickSort.ll   exemplo.c  'logo ufs.png'   relatorio-
depuracao.md
gabriel-ubuntu-wsl-2@DESKTOP-V76KM6R:/mnt/c/Users/User/Programacao/IHS$ cat
QuickSort.ll
; ModuleID = 'QuickSort.c'
source_filename = "QuickSort.c"
target datalayout = "e-m:e-p270:32:32-p271:32:32-p272:64:64-i64:64-f80:128-
n8:16:32:64-S128"
target triple = "x86_64-pc-linux-gnu"

@__const.main.vetor = private unnamed_addr constant [12 x i32] [i32 4, i32 5, i32
6, i32 9, i32 2, i32 3, i32 1, i32 8, i32
7, i32 10, i32 0, i32 0], align 16
@.str = private unnamed_addr constant [4 x i8] c"%d \00", align 1
@.str.1 = private unnamed_addr constant [2 x i8] c"\0A\00", align 1

; Function Attrs: noinline nounwind optnone uwtable
define dso_local void @trocar(i32* noundef %0, i32* noundef %1) #0 {
  %3 = alloca i32*, align 8
  %4 = alloca i32*, align 8
  %5 = alloca i32, align 4
  store i32* %0, i32** %3, align 8
  store i32* %1, i32** %4, align 8
  %6 = load i32*, i32** %3, align 8
  %7 = load i32, i32* %6, align 4
  store i32 %7, i32* %5, align 4
  %8 = load i32*, i32** %4, align 8
  %9 = load i32, i32* %8, align 4
  %10 = load i32*, i32** %3, align 8
  store i32 %9, i32* %10, align 4
  %11 = load i32, i32* %5, align 4
  %12 = load i32*, i32** %4, align 8
  store i32 %11, i32* %12, align 4
  ret void
}
```

Em seguida, compilei o código para um arquivo .elf e usei o comando objdump

```
gabriel-ubuntu-wsl-2@DESKTOP-V76KM6R:/mnt/c/Users/User/Programacao/IHS$ gcc -Wall
QuickSort.c -o QuickSort.elf
gabriel-ubuntu-wsl-2@DESKTOP-V76KM6R:/mnt/c/Users/User/Programacao/IHS$ ls
 QuickSort   QuickSort.c   QuickSort.elf   QuickSort.ll
gabriel-ubuntu-wsl-2@DESKTOP-V76KM6R:/mnt/c/Users/User/Programacao/IHS$ objdump --
disassemble -M amd QuickSort.elf

QuickSort.elf:     file format elf64-x86-64


Disassembly of section .init:

0000000000001000 <_init>:
    1000:       f3 0f 1e fa             endbr64
    1004:       48 83 ec 08             sub    $0x8,%rsp
    1008:       48 8b 05 d9 2f 00 00    mov    0x2fd9(%rip),%rax        #
    100f:       48 85 c0                test   %rax,%rax
    1012:       74 02                   je     1016 <_init+0x16>
    1014:       ff d0                   call   *%rax
    1016:       48 83 c4 08             add    $0x8,%rsp
    101a:       c3                      ret

0000000000001189 <trocar>:
    1189:       f3 0f 1e fa             endbr64
    118d:       55                      push   %rbp
    118e:       48 89 e5                mov    %rsp,%rbp
    1191:       48 89 7d e8             mov    %rdi,-0x18(%rbp)
    1195:       48 89 75 e0             mov    %rsi,-0x20(%rbp)
    1199:       48 8b 45 e8             mov    -0x18(%rbp),%rax
    119d:       8b 00                   mov    (%rax),%eax
    119f:       89 45 fc                mov    %eax,-0x4(%rbp)
    11a2:       48 8b 45 e0             mov    -0x20(%rbp),%rax
    11a6:       8b 10                   mov    (%rax),%edx
    11a8:       48 8b 45 e8             mov    -0x18(%rbp),%rax
    11ac:       89 10                   mov    %edx,(%rax)
    11ae:       48 8b 45 e0             mov    -0x20(%rbp),%rax
    11b2:       8b 55 fc                mov    -0x4(%rbp),%edx
    11b5:       89 10                   mov    %edx,(%rax)
    11b7:       90                      nop
    11b8:       5d                      pop    %rbp
    11b9:       c3                      ret

00000000000011ba <lomutoPartition>:
    11ba:       f3 0f 1e fa             endbr64
    11be:       55                      push   %rbp
    11bf:       48 89 e5                mov    %rsp,%rbp
    11c2:       48 83 ec 20             sub    $0x20,%rsp
    11c6:       89 7d ec                mov    %edi,-0x14(%rbp)
    11c9:       89 75 e8                mov    %esi,-0x18(%rbp)
    11cc:       48 89 55 e0             mov    %rdx,-0x20(%rbp)
    11d0:       8b 45 ec                mov    -0x14(%rbp),%eax
    11d3:       48 98                   cltq
    11d5:       48 8d 14 85 00 00 00    lea    0x0(,%rax,4),%rdx
    11dc:       00
    11dd:       48 8b 45 e0             mov    -0x20(%rbp),%rax
    11e1:       48 01 d0                add    %rdx,%rax
    11e4:       8b 00                   mov    (%rax),%eax
    11e6:       89 45 fc                mov    %eax,-0x4(%rbp)
    11e9:       8b 45 ec                mov    -0x14(%rbp),%eax
    11ec:       89 45 f4                mov    %eax,-0xc(%rbp)
    11ef:       8b 45 ec                mov    -0x14(%rbp),%eax
    11f2:       83 c0 01                add    $0x1,%eax
    11f5:       89 45 f8                mov    %eax,-0x8(%rbp)
    11f8:       eb 56                   jmp    1250 <lomutoPartition+0x96
    11fa:       8b 45 f8                mov    -0x8(%rbp),%eax
    11fd:       48 98                   cltq
    11ff:       48 8d 14 85 00 00 00    lea    0x0(,%rax,4),%rdx
    1206:       00
    1207:       48 8b 45 e0             mov    -0x20(%rbp),%rax
    120b:       48 01 d0                add    %rdx,%rax
    120e:       8b 00                   mov    (%rax),%eax
    1210:       39 45 fc                cmp    %eax,-0x4(%rbp)
    1213:       7e 37                   jle    124c <lomutoPartition+0x92
    1215:       83 45 f4 01             addl   $0x1,-0xc(%rbp)
    1219:       8b 45 f8                mov    -0x8(%rbp),%eax
    121c:       48 98                   cltq
    121e:       48 8d 14 85 00 00 00    lea    0x0(,%rax,4),%rdx
    1225:       00
    1226:       48 8b 45 e0             mov    -0x20(%rbp),%rax
    122a:       48 01 c2                add    %rax,%rdx
    122d:       8b 45 f4                mov    -0xc(%rbp),%eax
    1230:       48 98                   cltq
    1232:       48 8d 0c 85 00 00 00    lea    0x0(,%rax,4),%rcx
    1239:       00
    123a:       48 8b 45 e0             mov    -0x20(%rbp),%rax
    123e:       48 01 c8                add    %rcx,%rax
    1241:       48 89 d6                mov    %rdx,%rsi
    1244:       48 89 c7                mov    %rax,%rdi
    1247:       e8 3d ff ff ff          call   1189 <trocar>
    124c:       83 45 f8 01             addl   $0x1,-0x8(%rbp)
    1250:       8b 45 f8                mov    -0x8(%rbp),%eax
    1253:       3b 45 e8                cmp    -0x18(%rbp),%eax
    1256:       7c a2                   jl     11fa <lomutoPartition+0x40
    1258:       8b 45 f4                mov    -0xc(%rbp),%eax
    125b:       48 98                   cltq
    125d:       48 8d 14 85 00 00 00    lea    0x0(,%rax,4),%rdx
    1264:       00
    1265:       48 8b 45 e0             mov    -0x20(%rbp),%rax
    1269:       48 01 c2                add    %rax,%rdx
    126c:       8b 45 ec                mov    -0x14(%rbp),%eax
    126f:       48 98                   cltq
    1271:       48 8d 0c 85 00 00 00    lea    0x0(,%rax,4),%rcx
    1278:       00
    1279:       48 8b 45 e0             mov    -0x20(%rbp),%rax
    127d:       48 01 c8                add    %rcx,%rax
    1280:       48 89 d6                mov    %rdx,%rsi
    1283:       48 89 c7                mov    %rax,%rdi
    1286:	e8 fe fe ff ff       	call   1189 <trocar>
    128b:	8b 45 f4             	mov    -0xc(%rbp),%eax
    128e:	c9                   	leave
    128f:	c3                   	ret

0000000000001290 <QuickSort>:
    1290:	f3 0f 1e fa          	endbr64
    1294:	55                   	push   %rbp
    1295:	48 89 e5             	mov    %rsp,%rbp
    1298:	48 83 ec 20          	sub    $0x20,%rsp
    129c:	89 7d ec             	mov    %edi,-0x14(%rbp)
    129f:	89 75 e8             	mov    %esi,-0x18(%rbp)
    12a2:	48 89 55 e0          	mov    %rdx,-0x20(%rbp)
    12a6:	8b 45 ec             	mov    -0x14(%rbp),%eax
    12a9:	3b 45 e8             	cmp    -0x18(%rbp),%eax
    12ac:	7d 3f                	jge    12ed <QuickSort+0x5d>
    12ae:	48 8b 55 e0          	mov    -0x20(%rbp),%rdx
    12b2:	8b 4d e8             	mov    -0x18(%rbp),%ecx
    12b5:	8b 45 ec             	mov    -0x14(%rbp),%eax
    12b8:	89 ce                	mov    %ecx,%esi
    12ba:	89 c7                	mov    %eax,%edi
    12bc:	e8 f9 fe ff ff       	call   11ba <lomutoPartition>
    12c1:	89 45 fc             	mov    %eax,-0x4(%rbp)
    12c4:	48 8b 55 e0          	mov    -0x20(%rbp),%rdx
    12c8:	8b 4d fc             	mov    -0x4(%rbp),%ecx
    12cb:	8b 45 ec             	mov    -0x14(%rbp),%eax
    12ce:	89 ce                	mov    %ecx,%esi
    12d0:	89 c7                	mov    %eax,%edi
    12d2:	e8 b9 ff ff ff       	call   1290 <QuickSort>
    12d7:	8b 45 fc             	mov    -0x4(%rbp),%eax
    12da:	8d 48 01             	lea    0x1(%rax),%ecx
    12dd:	48 8b 55 e0          	mov    -0x20(%rbp),%rdx
    12e1:	8b 45 e8             	mov    -0x18(%rbp),%eax
    12e4:	89 c6                	mov    %eax,%esi
    12e6:	89 cf                	mov    %ecx,%edi
    12e8:	e8 a3 ff ff ff       	call   1290 <QuickSort>
    12ed:	90                   	nop
    12ee:	c9                   	leave
    12ef:	c3                   	ret

00000000000012f0 <main>:
    12f0:	f3 0f 1e fa          	endbr64
    12f4:	55                   	push   %rbp
    12f5:	48 89 e5             	mov    %rsp,%rbp
    12f8:	48 83 ec 50          	sub    $0x50,%rsp
    12fc:	64 48 8b 04 25 28 00 	mov    %fs:0x28,%rax
    1303:	00 00
    1305:	48 89 45 f8          	mov    %rax,-0x8(%rbp)
    1309:	31 c0                	xor    %eax,%eax
    130b:	48 c7 45 c0 00 00 00 	movq   $0x0,-0x40(%rbp)
    1312:	00
    1313:	48 c7 45 c8 00 00 00 	movq   $0x0,-0x38(%rbp)
    131a:	00
    131b:	48 c7 45 d0 00 00 00 	movq   $0x0,-0x30(%rbp)
    1322:       00
    1323:       48 c7 45 d8 00 00 00    movq   $0x0,-0x28(%rbp)
    132a:       00
    132b:       48 c7 45 e0 00 00 00    movq   $0x0,-0x20(%rbp)
    1332:       00
    1333:       48 c7 45 e8 00 00 00    movq   $0x0,-0x18(%rbp)
    133a:       00
    133b:       c7 45 c0 04 00 00 00    movl   $0x4,-0x40(%rbp)
    1342:       c7 45 c4 05 00 00 00    movl   $0x5,-0x3c(%rbp)
    1349:       c7 45 c8 06 00 00 00    movl   $0x6,-0x38(%rbp)
    1350:       c7 45 cc 09 00 00 00    movl   $0x9,-0x34(%rbp)
    1357:       c7 45 d0 02 00 00 00    movl   $0x2,-0x30(%rbp)
    135e:       c7 45 d4 03 00 00 00    movl   $0x3,-0x2c(%rbp)
    1365:       c7 45 d8 01 00 00 00    movl   $0x1,-0x28(%rbp)
    136c:       c7 45 dc 08 00 00 00    movl   $0x8,-0x24(%rbp)
    1373:       c7 45 e0 07 00 00 00    movl   $0x7,-0x20(%rbp)
    137a:       c7 45 e4 0a 00 00 00    movl   $0xa,-0x1c(%rbp)
    1381:       c7 45 b8 00 00 00 00    movl   $0x0,-0x48(%rbp)
    1388:       eb 23                   jmp    13ad <main+0xbd>
    138a:       8b 45 b8                mov    -0x48(%rbp),%eax
    138d:       48 98                   cltq
    138f:       8b 44 85 c0             mov    -0x40(%rbp,%rax,4),%eax
    1393:       89 c6                   mov    %eax,%esi
    1395:       48 8d 05 68 0c 00 00    lea    0xc68(%rip),%rax        #
    139c:       48 89 c7                mov    %rax,%rdi
    139f:       b8 00 00 00 00          mov    $0x0,%eax
    13a4:       e8 e7 fc ff ff          call   1090 <printf@plt>
    13a9:       83 45 b8 01             addl   $0x1,-0x48(%rbp)
    13ad:       83 7d b8 09             cmpl   $0x9,-0x48(%rbp)
    13b1:       7e d7                   jle    138a <main+0x9a>
    13b3:       bf 0a 00 00 00          mov    $0xa,%edi
0,%eax                                     0 <putchar@plt>
    13f6:       e8 95 fc ff ff          call   1090 <printf@plt>
    13fb:       83 45 bc 01             addl   $0x1,-0x44(%rbp)
    13ff:       83 7d bc 09             cmpl   $0x9,-0x44(%rbp)
    1403:       7e d7                   jle    13dc <main+0xec>
    1405:       bf 0a 00 00 00          mov    $0xa,%edi
    140a:       e8 61 fc ff ff          call   1070 <putchar@plt>
    140f:       b8 00 00 00 00          mov    $0x0,%eax
    1414:       48 8b 55 f8             mov    -0x8(%rbp),%rdx
    1418:       64 48 2b 14 25 28 00    sub    %fs:0x28,%rdx
    141f:       00 00
    1421:       74 05                   je     1428 <main+0x138>
    1423:       e8 58 fc ff ff          call   1080 <__stack_chk_fail@plt>
    1428:       c9                      leave
    1429:       c3                      ret
```

Em seguida, realizei a compilação com suporte para depuração através do gdb

```
gabriel-ubuntu-wsl-2@DESKTOP-V76KM6R:/mnt/c/Users/User/Programacao/IHS$ gdb
QuickSort.elf
```

```
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 QuickSort.elf...
(gdb) run
Starting program: /mnt/c/Users/User/Programacao/IHS/QuickSort.elf
[Thread debugging using libthread_db enabled]
Using host libthread_db library "/lib/x86_64-linux-gnu/libthread_db.so.1".
4 5 6 9 2 3 1 8 7 10
1 2 3 4 5 6 7 8 9 10
[Inferior 1 (process 19245) exited normally]
```

Em seguida, realizei a compilação com suporte para depuração através do lldb

```
gabriel-ubuntu-wsl-2@DESKTOP-V76KM6R:/mnt/c/Users/User/Programacao/IHS$ lldb
QuickSort.elf
Traceback (most recent call last):
  File "<string>", line 1, in <module>
ModuleNotFoundError: No module named 'lldb.embedded_interpreter'
(lldb) target create "QuickSort.elf"
Current executable set to '/mnt/c/Users/User/Programacao/IHS/QuickSort.elf'
(x86_64).
(lldb) run
Process 20322 launched: '/mnt/c/Users/User/Programacao/IHS/QuickSort.elf' (x86_64)
4 5 6 9 2 3 1 8 7 10
1 2 3 4 5 6 7 8 9 10
Process 20322 exited with status = 0 (0x00000000)
(lldb)
```

Após isso, alterei os algoritmos de ordenação, para testar o número de comparações e o desempenho entre eles

```
#include <stdio.h>

int SelectionSort(int tamanho, int *vetordeint) {
    int menor, temp, comparacoes = 0;
    for(int i = 0; i < tamanho-1; i++) {
        menor = i;
        for(int j = i+1; j < tamanho; j++) {
            if(vetordeint[j] < vetordeint[menor]) {
                menor = j;
                comparacoes++;
            }
        }
        temp = vetordeint[i];
        vetordeint[i] = vetordeint[menor];
        vetordeint[menor] = temp;
    }
    return comparacoes;
}

int bubbleSort(int tamanhoLista, int *vetordeinteiros) {
    int temporario, comparacoes = 0;
    for(int i = 0; i < tamanhoLista-1; i++) {
        for(int j = 0; j < (tamanhoLista-1)-i; j++) {
            if(vetordeinteiros[j+1] < vetordeinteiros[j]) {
                temporario = vetordeinteiros[j];
                vetordeinteiros[j] = vetordeinteiros[j+1];
                vetordeinteiros[j+1] = temporario;
                comparacoes++;
            }
        }
    }
    return comparacoes;
}

int main() {
    int vetor[12] = {4, 5, 6, 9, 2, 3, 1, 8, 7, 10};
    int originalVetor[12] = {4, 5, 6, 9, 2, 3, 1, 8, 7, 10};
    int ss = 0, bs = 0;

    for (int i = 0; i < 10000000; i++) {
        for (int j = 0; j < 10; j++) {
            vetor[j] = originalVetor[j];
        }

        ss += SelectionSort(10, vetor);

        for (int j = 0; j < 10; j++) {
            vetor[j] = originalVetor[j];
        }

        bs += bubbleSort(10, vetor);
    }
    printf("SelectionSort: %d BubbleSort: %d\n", ss, bs);
    return 0;
}
```

Em seguida, analisei o desempenho dos algoritmos através do comando gprof

```
gabriel-ubuntu-wsl-2@DESKTOP-V76KM6R:/mnt/c/Users/User/Programacao/IHS$ gcc -Wall
-g -pg QuickSort.c -o QuickSort.elf
gabriel-ubuntu-wsl-2@DESKTOP-V76KM6R:/mnt/c/Users/User/Programacao/IHS$
./QuickSort.elf
SelectionSort: 90000000 BubbleSort: 170000000
gabriel-ubuntu-wsl-2@DESKTOP-V76KM6R:/mnt/c/Users/User/Programacao/IHS$ gprof
QuickSort.elf gmon.out > analysis.txt
```

Esses foram os resultados da análise de desempenho e estão mais detalhados dentro do arquivo analysis.txt

```
  %   cumulative   self              self     total
 time   seconds   seconds    calls  ns/call  ns/call  name
 44.64      1.04     1.04 10000000   104.00   104.00  bubbleSort
 37.77      1.92     0.88 10000000    88.00    88.00  SelectionSort
 17.60      2.33     0.41                             main
```

Infelizmente, não foi possível comparar os resultados do gprof com o do perf, pois me deparei com diversos erros que me impossibilitaram.

```
gabriel-ubuntu-wsl-2@DESKTOP-V76KM6R:/mnt/c/Users/User/Programacao/IHS$ perf
record ./QuickSort.elf
Command 'perf' not found, but can be installed with:
sudo apt install linux-intel-iotg-tools-common        # version 5.15.0-1043.49, or
sudo apt install linux-nvidia-6.2-tools-common        # version 6.2.0-
1003.3~22.04.1
sudo apt install linux-nvidia-tools-common            # version 5.15.0-1040.40
sudo apt install linux-tools-common                   # version 5.15.0-88.98
sudo apt install linux-nvidia-5.19-tools-common       # version 5.19.0-1014.14
sudo apt install linux-nvidia-tegra-igx-tools-common  # version 5.15.0-1005.5
sudo apt install linux-nvidia-tegra-tools-common      # version 5.15.0-1018.18
sudo apt install linux-xilinx-zynqmp-tools-common     # version 5.15.0-1023.27
gabriel-ubuntu-wsl-2@DESKTOP-V76KM6R:/mnt/c/Users/User/Programacao/IHS$ sudo apt
install linux-tools-common
gabriel-ubuntu-wsl-2@DESKTOP-V76KM6R:/mnt/c/Users/User/Programacao/IHS$ perf
record ./QuickSort.elf
WARNING: perf not found for kernel 5.15.133.1-microsoft

  You may need to install the following packages for this specific kernel:
    linux-tools-5.15.133.1-microsoft-standard-WSL2
    linux-cloud-tools-5.15.133.1-microsoft-standard-WSL2

  You may also want to install one of the following packages to keep up to date:
    linux-tools-standard-WSL2
    linux-cloud-tools-standard-WSL2
```