

75:42 - Taller de Programación I

Ejercicio N° _____ Padrón _____

Alumno _____ Firma _____

Nota:		Corrige:		Entrega #1
				Fecha de entrega
				Fecha de devolución

Nota:		Corrige:		Entrega #2
				Fecha de entrega
				Fecha de devolución

El presente trabajo, así como la entrega electrónica correspondiente al mismo, constituyen una obra de creación completamente personal, no habiendo sido inspirada ni siendo copia completa o parcial de ninguna fuente pública, privada, de otra persona o naturaleza.

Índice general

1	Introduccion	4
2	Paso 0: Entorno de Trabajo	5
2.1	Sistema operativo y herramientas instaladas	5
2.2	Aplicacion 'Hola Mundo'	6
2.3	¿Que es Valgrind?	7
2.4	¿Que representa sizeof()?¿Cual seria el valor de salida de sizeof(char) y sizeof(int)?	9
2.5	Sizeof() de un struct vs sizeof() de cada uno de sus elementos	9
2.6	STDIN, STDOUT, STDERR	10
3	Paso 1: SERCOM - Errores de generacion y normas de programacion	12
3.1	Entrega rechazada en SERCOM	12
3.2	Errores de estilo	12
3.3	Errores de generacion del ejecutable	14
3.4	Warning	15
4	Paso 2: Errores de generacion 2	16
4.1	Correcciones realizadas respecto al Paso 1	16
4.2	Entrega del paso 2	16
4.3	Errores de generacion del ejecutable	17
5	Paso 3: Errores de generacion 3	20
5.1	Correcciones realizadas respecto al Paso 2	20
5.2	Errores de generacion del ejecutable	20
6	Paso 4: Memory Leaks y Buffer Overflows	22
6.1	Correcciones realizadas respecto al Paso 3	22
6.2	Resultado de ejecucion con Valgrind de la prueba 'TDA'	22
6.3	Resultado de ejecucion con Valgrind de la prueba 'Long Filename'	24
6.4	Funcion strncpy	25
6.5	Segmentation fault y un buffer overflow	25
7	Paso 5:Codigo de retorno y salida estandar	26
7.1	Correcciones realizadas respecto al Paso 4	26
7.2	Fallo de las pruebas 'Invalid File' y 'Single Word'	26
7.3	Comando hexdump	27
7.4	Ejecucion con GDB	28

Índice general

8 Paso 6: Entrega exitosa	30
8.1 Correcciones realizadas respecto al Paso 5	30
8.2 Entregas realizadas	30
8.3 Ejecucion de la prueba 'Single Word' de forma local	31
9 Paso 7	32

1 Introduccion

El trabajo practico tenia distintos objetivos los cuales se fueron cumpliendo a medida que se completo una serie de pasos. Se dara una breve explicacion de cada uno de los objetivos alcanzados.

- Familiarizarse con el sistema de entregas SERCOM: Consistio en aprender a realizar adecuadamente una entrega y comprender que respuesta se obtenia de SerCom, ya sea una entrega exitosa o fallida.
- Preparacion del ambiente de trabajo propio: Consistio en preparar localmente un ambiente de trabajo con las herramientas necesarias para el trabajo practico realizado y futuros trabajos. Entre estas herramientas estaban incluidas Valgrind, GDB y un compilador adecuado como GCC o G++.
- Nivelar conocimientos basicos en C: Se tuvo que repasar temas aprendidos en materias previas e investigar algunos nuevos conceptos basicos.
- Preparacion de informe tecnico: Realizar un informe adecuado para la presentacion del trabajo.

En los siguientes capitulos se explicara en detalle cuales fueron los pasos a seguir para lograr alcanzar los objetivos del trabajo.

2 Paso 0: Entorno de Trabajo

2.1. Sistema operativo y herramientas instaladas

Este paso consistio en tener un ambiente de trabajo adecuado. En las siguientes imagenes se puede ver el sistema operativo utilizado y las herramientas con sus respectivas versiones.

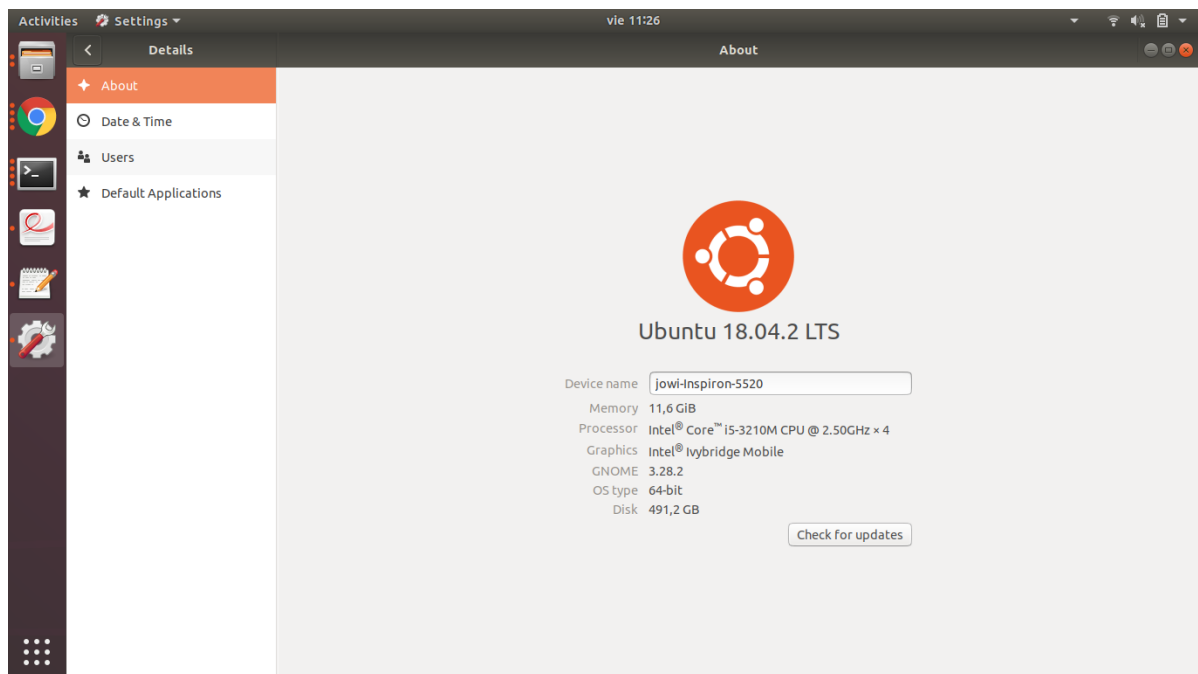


Figura 2.1: Sistema operativo utilizado

2 Paso 0: Entorno de Trabajo

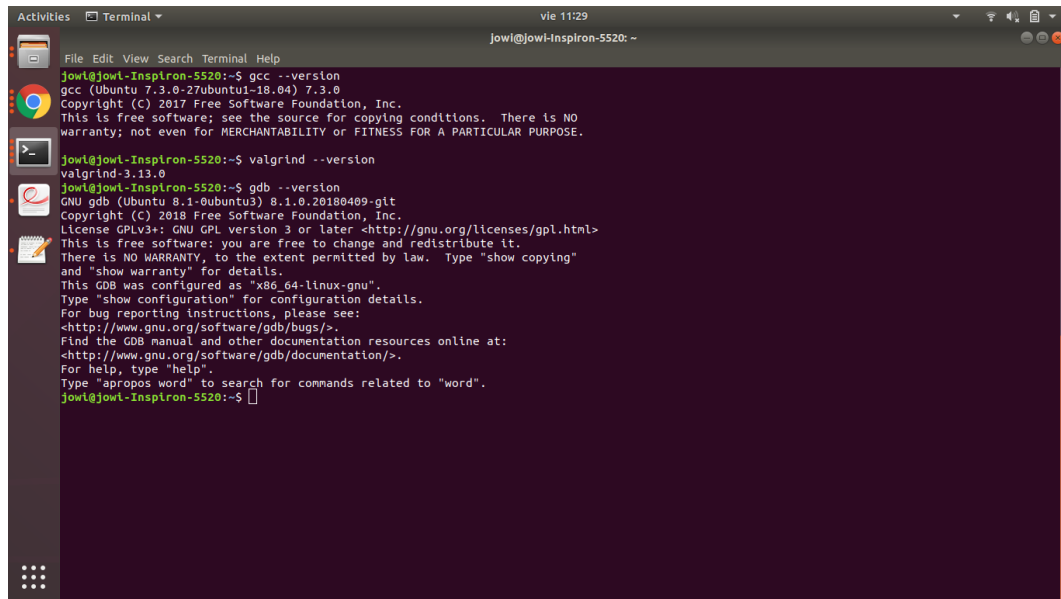


Figura 2.2: Herramientas utilizadas con sus respectivas versiones

2.2. Aplicacion 'Hola Mundo'

Se compilo y ejecuto una aplicacion simple ISO C que imprime por pantalla el mensaje 'Hola Mundo' y retorna un 0 (cero). Las siguientes imagenes muestran la ejecucion de dicha aplicacion (con y sin Valgrind).

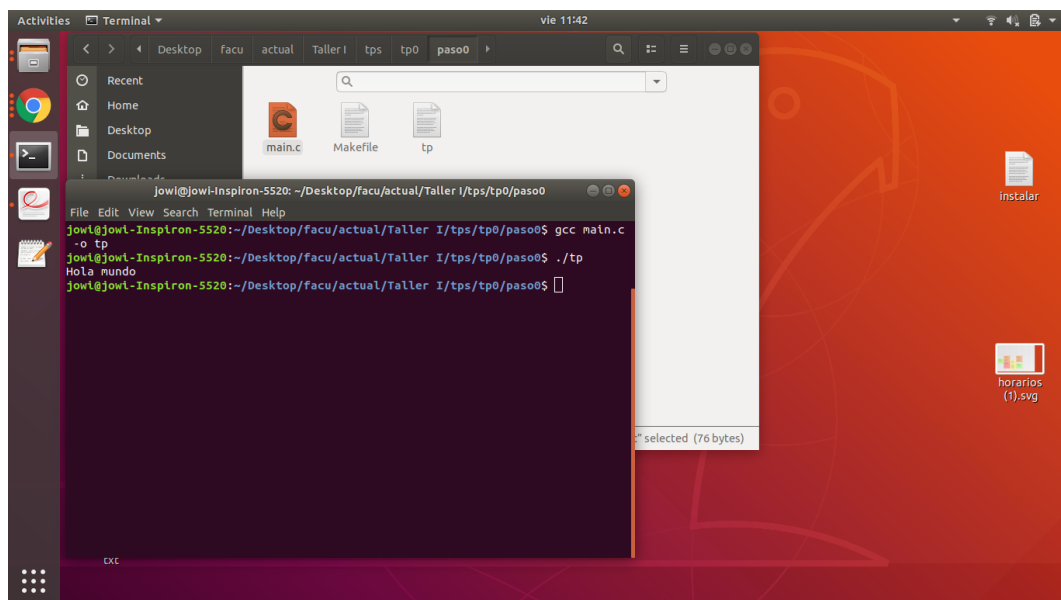


Figura 2.3: Ejecucion de la aplicacion sin Valgrind

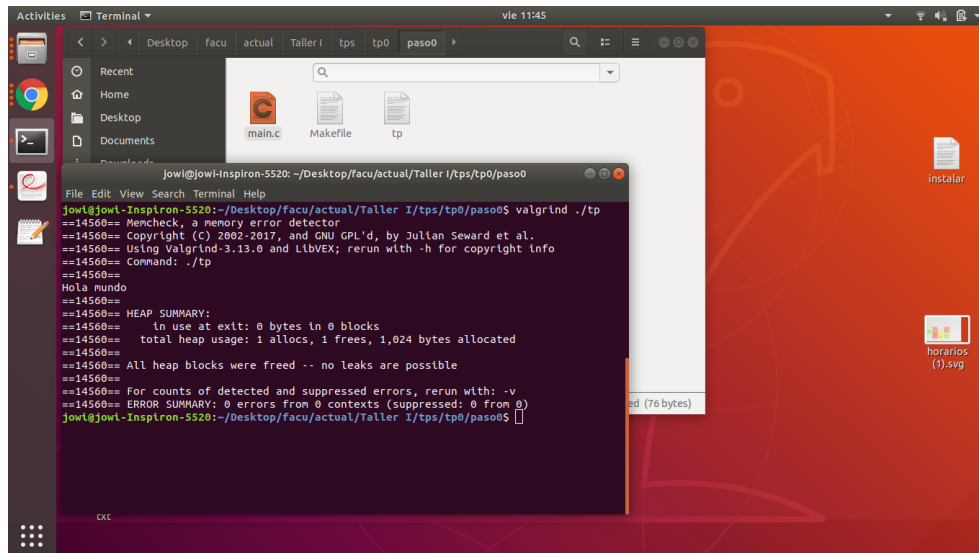
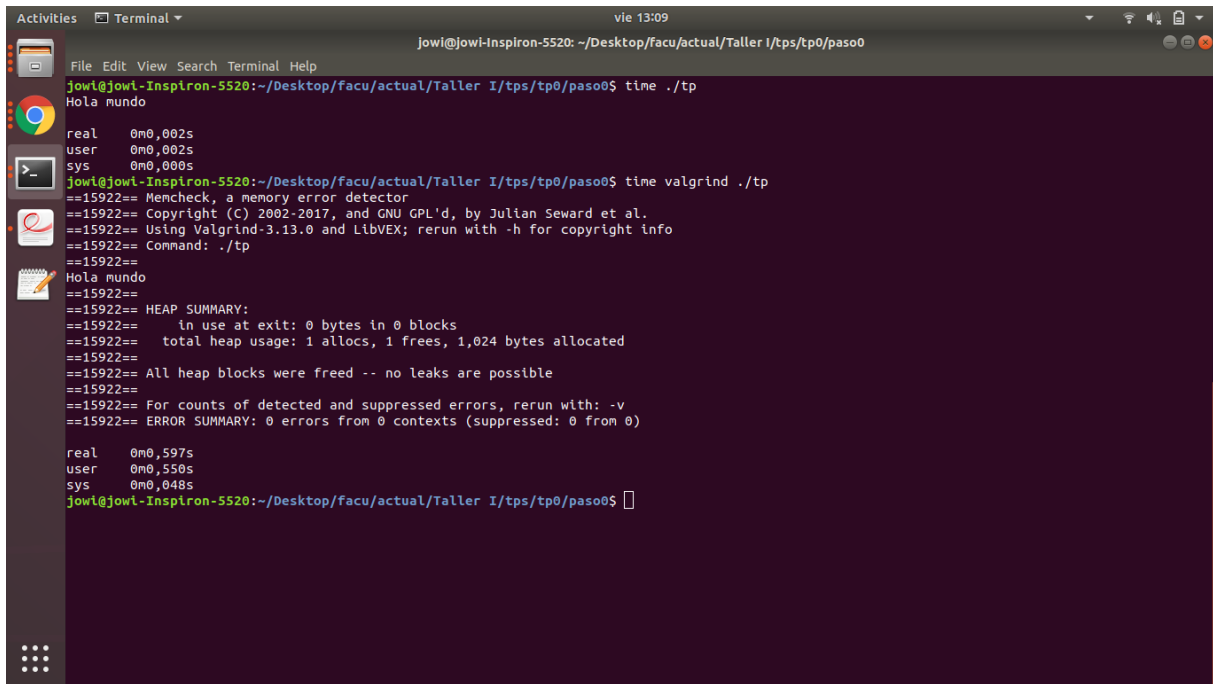


Figura 2.4: Ejecucion de la aplicacion con Valgrind

2.3. ¿Que es Valgrind?

Valgrind es un software open source, de licencia gratuita, que permite detectar problemas de memoria y analizar el rendimiento de programas. Valgrind se puede utilizar para crear nuevas herramientas de trabajo o utilizar las que ya existen. Actualmente posee seis herramientas para la depuracion de programas (ayudan a identificar y corregir errores de programacion en los programas) y tres herramientas experimentales. Por el momento es posible utilizar Valgrind en distintas plataformas de Linux, Solaris, Android y Darwin. Su herramienta de uso mas comun es Memcheck. Dicha herramienta permite detectar problemas como por ejemplo el uso de memoria no inicializada, fugas de memoria, lectura/escritura de memoria que ya fue liberada entre otros errores. La principal desventaja de valgrind es que realentiza la ejecucion del programa, consultando con distintas fuentes de internet se obtuvo informacion de que realentiza el programa en el orden de 10 a 30 veces mas lento. Se realizo la prueba para el programa anterior y el orden fue superior, esto se debe a que valgrind tarda un tiempo fijo en inicializar que para programas cortos el orden de veces que lo realentiza es mas notorio. Para comprobar esto se realizo un programa similar donde la cantidad de veces que imprimia por pantalla 'Hola Mundo' fue 50.000 y se puede ver que la cantidad de veces que se realentizo el programa esta dentro del rango investigado en internet. [1] [2]

2 Paso 0: Entorno de Trabajo



The terminal window shows the execution of a program named 'tp' in the directory ~/Desktop/facu/actual/Taller I/tps/tp0/paso0. The first run uses the 'time' command, showing a real time of 0m0.002s. The second run uses 'time valgrind ./tp', which shows a real time of 0m0.597s. The Valgrind output includes a heap summary indicating 1,024 bytes allocated and no leaks.

```
jowi@jowi-Inspiron-5520: ~/Desktop/facu/actual/Taller I/tps/tp0/paso0
jowi@jowi-Inspiron-5520:~/Desktop/facu/actual/Taller I/tps/tp0/paso0$ time ./tp
Hola mundo
real    0m0.002s
user    0m0.002s
sys     0m0.000s
jowi@jowi-Inspiron-5520:~/Desktop/facu/actual/Taller I/tps/tp0/paso0$ time valgrind ./tp
==15922== Memcheck, a memory error detector
==15922== Copyright (C) 2002-2017, and GNU GPL'd, by Julian Seward et al.
==15922== Using Valgrind-3.13.0 and LibVEX; rerun with -h for copyright info
==15922== Command: ./tp
==15922==
Hola mundo
==15922==
==15922== HEAP SUMMARY:
==15922==   in use at exit: 0 bytes in 0 blocks
==15922==   total heap usage: 1 allocs, 1 frees, 1,024 bytes allocated
==15922==
==15922== All heap blocks were freed -- no leaks are possible
==15922==
==15922== For counts of detected and suppressed errors, rerun with: -v
==15922== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)

real    0m0.597s
user    0m0.550s
sys     0m0.048s
jowi@jowi-Inspiron-5520:~/Desktop/facu/actual/Taller I/tps/tp0/paso0$
```

Figura 2.5: Tiempo de ejecucion del programa 'Hola Mundo' con y sin Valgrind

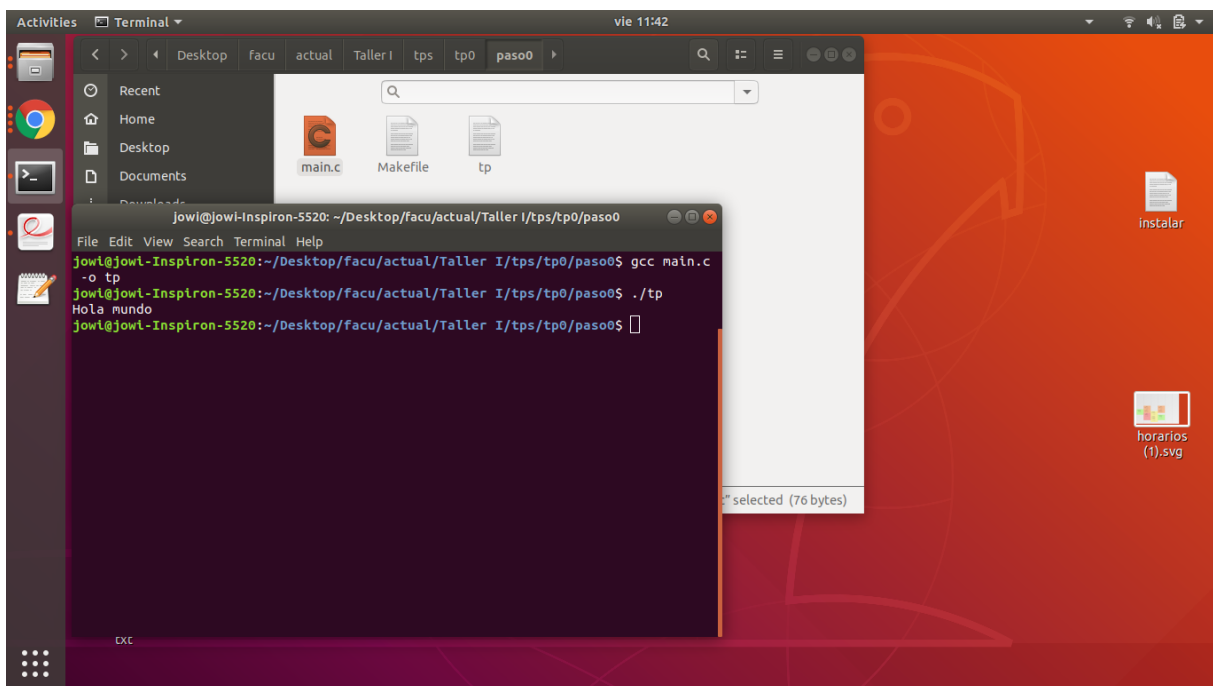


Figura 2.6: Tiempo de ejecucion del programa 'Hola Mundo' (x50.000) sin Valgrind

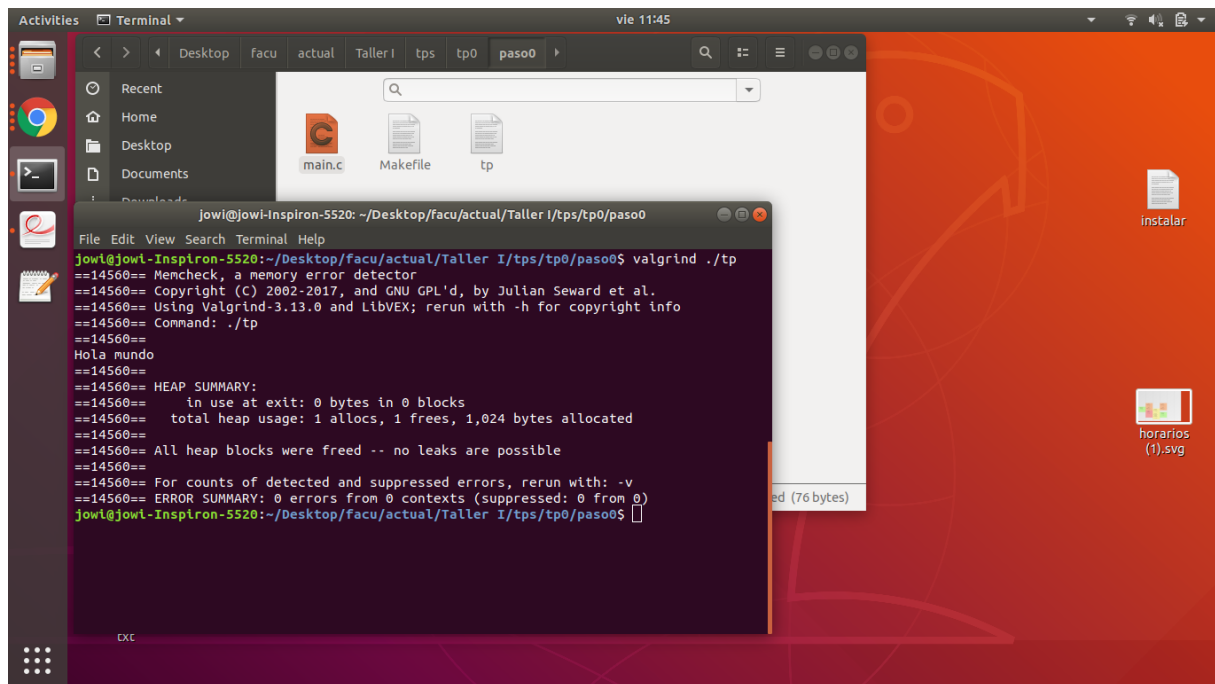


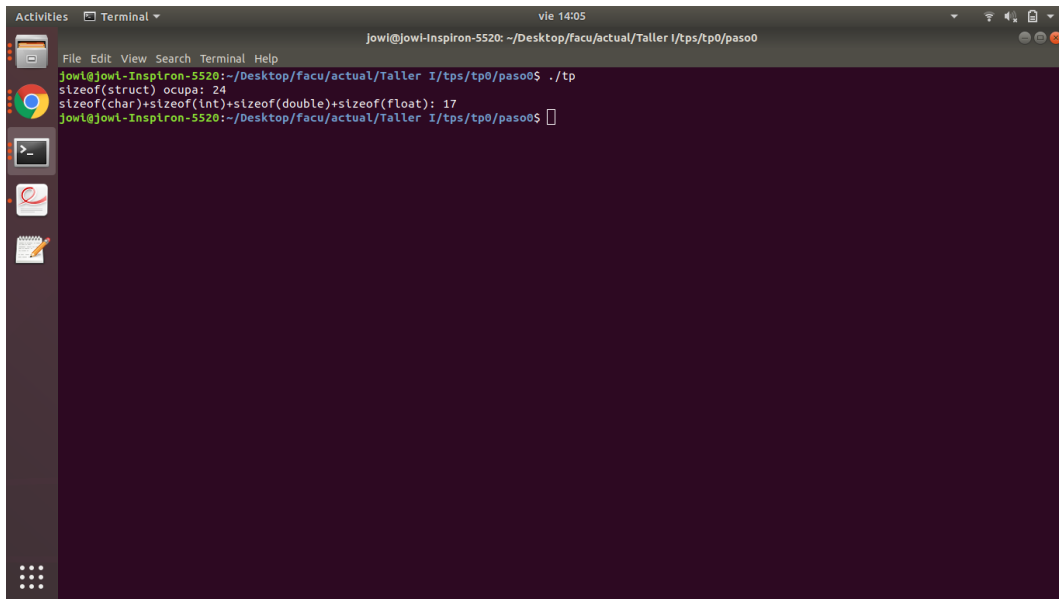
Figura 2.7: Tiempo de ejecución del programa 'Hola Mundo' (x50.000) con Valgrind

2.4. ¿Que representa sizeof()? ¿Cual seria el valor de salida de sizeof(char) y sizeof(int)?

La función `sizeof()` representa la cantidad de bytes ocupados por un tipo de dato. El valor de salida de `sizeof(char)` va ser *siempre* 1 byte, no depende del compilador en el que se este trabajando mientras que para `sizeof(int)` podra ser 2 bytes o 4 bytes dependiendo de si se esta trabajando con un compilador de 16 bits, 32 bits o 64 bits.

2.5. `sizeof()` de un struct vs `sizeof()` de cada uno de sus elementos

El `sizeof()` de una struct de C no es igual a la suma del `sizeof()` de cada uno sus elementos. Para corroborar esto se realizo un contra ejemplo donde se genero un programa que imprimia por pantalla el `sizeof()` de un struct que contenia un int, un char, un float y un double y el resultado era distinto a la suma de cada uno de los `sizeof()`. Esto se debe a que el compilador agrega padding con el objetivo de ganar velocidad al momento de acceder a los datos del struct a cambio de perder memoria (la memoria que se pierde suele ser poca, aunque en caso de necesitar esta memoria hay opciones de compilacion que evita que el compilador realice padding, una de ellas es 'package').



```
Activities Terminal vie 14:05
jowi@jowi-Inspiron-5520: ~/Desktop/facu/actual/Taller I/tps/tp0/paso0
File Edit View Search Terminal Help
jowi@jowi-Inspiron-5520:~/Desktop/facu/actual/Taller I/tps/tp0/paso0$ ./tp
sizeof(struct) ocupa: 24
sizeof(char)+sizeof(int)+sizeof(double)+sizeof(float): 17
jowi@jowi-Inspiron-5520:~/Desktop/facu/actual/Taller I/tps/tp0/paso0$
```

Figura 2.8: Sizeof() de un struct vs la suma de sizeof() de cada elemento

2.6. STDIN, STDOUT, STDERR

El archivo stdin es el archivo que entra con informacion al programa, el stdout es el que el sale con informacion comun del programa y el stderr es el archivo que contiene los errores que hubo en el programa. Se puede redireccionar mediante los caracteres '`i`' y '`j`' el stdin, stdout y stderr. Se puede ver en la siguiente imagen (2.9) de como la salida del programa hecho en items anteriores es copiada en un txt utilizando el caracter '`j`'. El comando pipe '`—`' es utilizado para crear tuberias, se puede conectar el stdout de un programa con el stdin de otro. Se utiliza de la forma: comando1 `—` comando2

2 Paso 0: Entorno de Trabajo

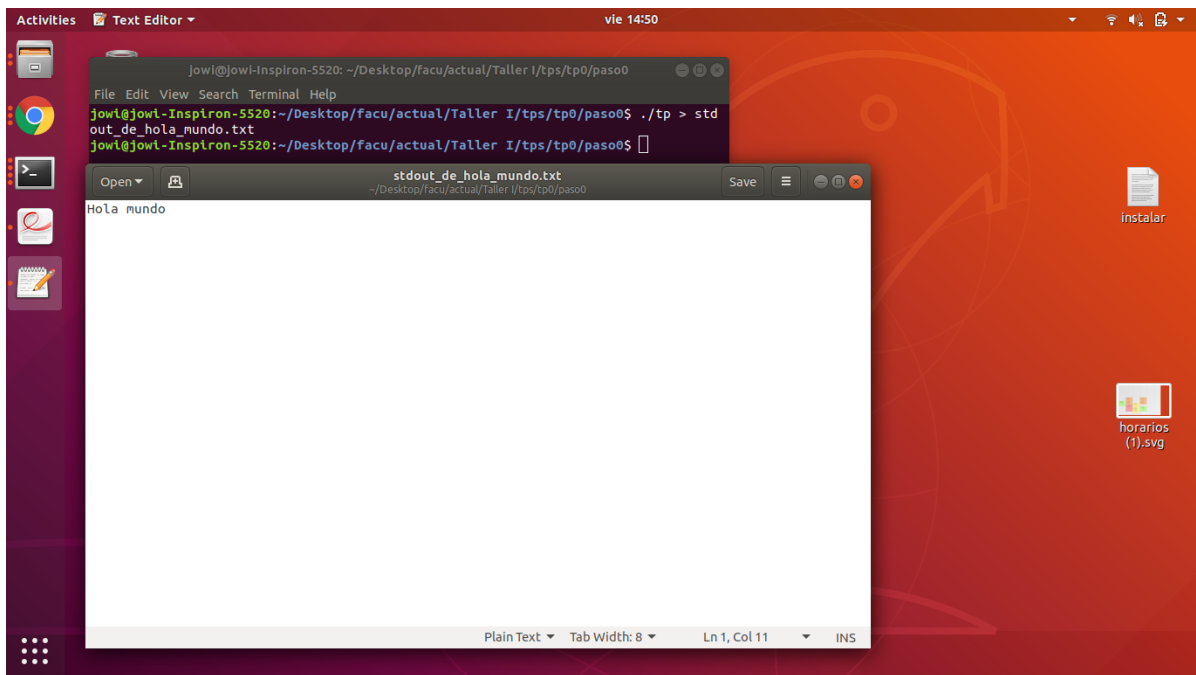


Figura 2.9: Salida de un programa es utilizada en otro

3 Paso 1: SERCOM - Errores de generacion y normas de programacion

- ./paso1_wordscouter.c:41: Mismatching spaces inside () in if [whitespace/parens] [5] : Habia mas espacios de un lado que de otro dentro de un if en la linea 41.
- ./paso1_wordscouter.c:41: Should have zero or one spaces inside (and) in if [whitespace/parens] [5] : Dentro de un if se debe dejar uno o cero espacios de cada lado (estos deben coincidir) en la linea 41.
- ./paso1_wordscouter.c:47: An else should appear on the same line as the preceding } [whitespace/newline] [4] : Se habia hecho un salto de linea antes del utilizar un else y este debe estar en la misma linea en la que se cierra la llave.
- ./paso1_wordscouter.c:47: Missing space before (in if([whitespace/parens] [5] : Se debe a que los parentesis que contienen la condicion del if deben estar separados por un espacio del if en la linea 47.
- ./paso1_wordscouter.c:52: Extra space before last semicolon. If this should be an empty statement, use instead. [whitespace/semicolon] [5]: Se dejo un espacio en blanco y luego se puso el ';', no debe haber espacios entre el anteultimo caracter de la linea y el ';' final.
- extra: ./paso1_wordscouter.h:5: Lines should be j= 80 characters long [whitespace/line_length] [2]: La catedra no lo concidero y google lo toma como un error leve (ya que tiene una puntuacion de 2) pero la linea supero los 80 caracteres.

Para el archivo paso1_main.c:

- ./paso1_main.c:12: Almost always, snprintf is better than strcpy [runtime/printf] [4] : Por cuestiones de eficiencia recomienda utilizar snprintf antes que strcpy.
- ./paso1_main.c:15: An else should appear on the same line as the preceding } [whitespace/newline] [4] : Se habia hecho un salto de linea antes del utilizar un else y este debe estar en la misma linea en la que se cierra la llave.
- ./paso1_main.c:15: If an else has a brace on one side, it should have it on both [readability/braces] [5] : Si un else tiene una llave en un lado debe tenerla en el otro tambien. Como el else debe ir en la misma linea que se cierra la llave del if, el mismo debe tener su codigo dentro de llaves por mas que ocupe una unica linea.

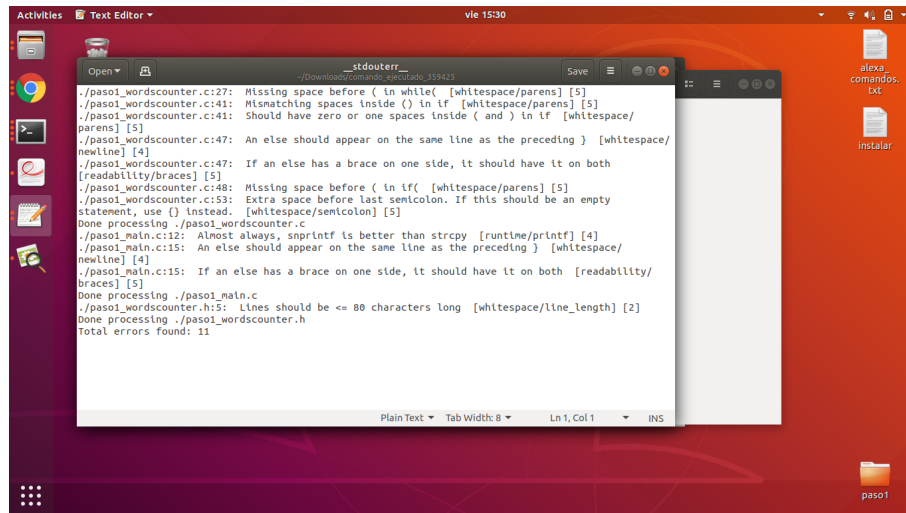


Figura 3.2: Errores de estilo

3.3. Errores de generacion del ejecutable

Los errores de generacion del ejecutable se deben a que en el main se utiliza la libreria "paso1_wordscouter.h". entonces hay que incluirla ya que se esta tratando de utilizar un tipo de dato el compilador nosabe cuanto espacio ocupa y funciones no sabe que son (no estan declaradas en ningun lado). Los errores obtenidos:

- unknown type name
- implicit declaration of function

Ambos son causados por el mismo problema (falta incluir en el main "paso1_wordscouter.h") y ambos errores son previos a la etapa de linkeo, durante la etapa de compilacion.

3 Paso 1: SERCOM - Errores de generacion y normas de programacion

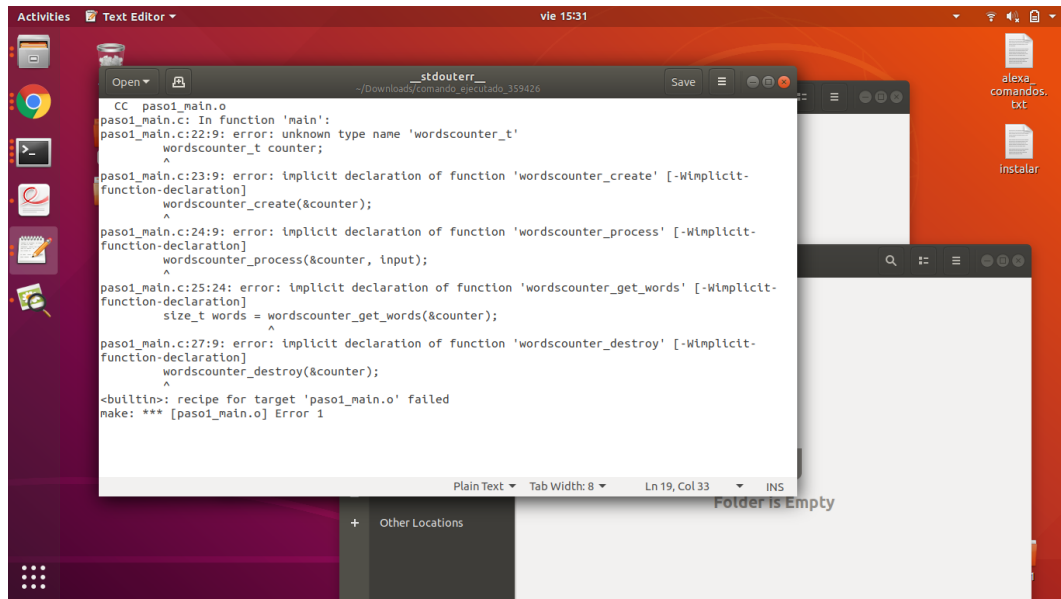


Figura 3.3: Errores de generacion del ejecutable

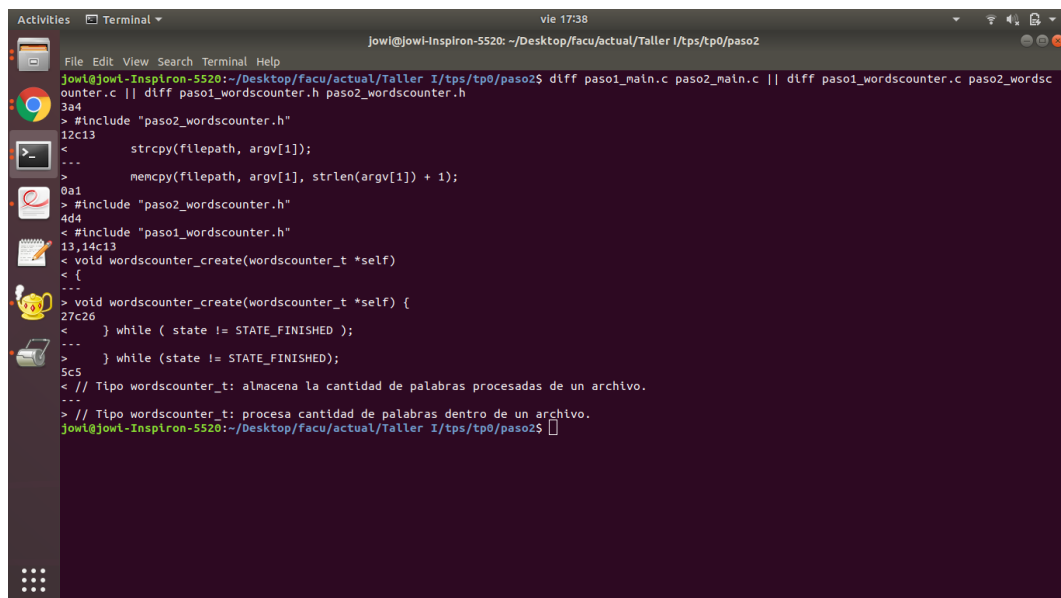
3.4. Warning

El sistema no reporto ningun WARNING dado que en el make estamos utilizando el flag Werror el cual hace que todos los warning sean considerados como errores. El flag Wall hacer que se muestren todos los warning que halla en nuestro programa pero al estar siendo usado al mismo tiempo el flag -Werror estos se toman como errores.

4 Paso 2: Errores de generacion 2

4.1. Correcciones realizadas respecto al Paso 1

Respecto de la version anterior se corrigieron todos los errores de estilo que habia en los archivos del paso 1. Se cambio strcpy por memcpy el cual tiene algunas ventajas, una de ellas es que strcpy copia solo cadenas mientras que memcpy copia n bytes sin importar que hay en ellos. Ademas se incluyo en el main la libreria "paso2_wordscouter.h" la cual era necesaria para no tener los errores de compilacion del paso1. El ultimo cambio es que en el archivo "paso2_wordscouter.h" se aclara que el contador ya no almacenara las palabras sino que las va procesar, esto alcanza para poder contar la cantidad de palabras.



```
Activities Terminal vie 17:38
jowi@jowi-Inspiron-5520: ~/Desktop/facu/actual/Taller I/tps/tp0/paso2
jowi@jowi-Inspiron-5520:~/Desktop/facu/actual/Taller I/tps/tp0/paso2$ diff paso1_main.c paso2_main.c || diff paso1_wordscouter.c paso2_wordscouter.c || diff paso1_wordscouter.h paso2_wordscouter.h
3a4
> #include "paso2_wordscouter.h"
12c13
< strcpy(fllepath, argv[1]);
---
> memcpy(fllepath, argv[1], strlen(argv[1]) + 1);
0a1
> #include "paso2_wordscouter.h"
4d4
< #include "paso1_wordscouter.h"
13,14c13
< void wordscounter_create(wordscounter_t *self)
< {
---
> void wordscounter_create(wordscounter_t *self) {
27c26
< } while ( state != STATE_FINISHED );
---
> } while (state != STATE_FINISHED);
5c5
< // Tipo wordscounter_t: almacena la cantidad de palabras procesadas de un archivo.
---
> // Tipo wordscounter_t: procesa cantidad de palabras dentro de un archivo.
jowi@jowi-Inspiron-5520:~/Desktop/facu/actual/Taller I/tps/tp0/paso2$
```

Figura 4.1: Diferencias entre paso 1 y paso 2

4.2. Entrega del paso 2

En las siguientes imagenes podemos ver en SERCOM que no hubo errores de estilo pero el programa no tuvo una entrega exitosa.

4 Paso 2: Errores de generacion 2

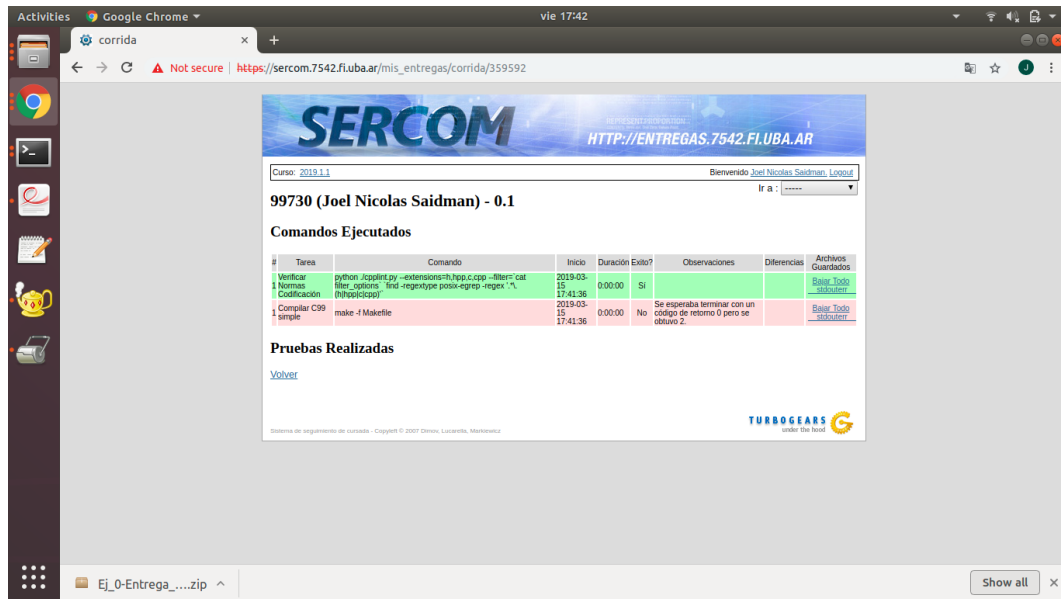


Figura 4.2: Entrega rechazada

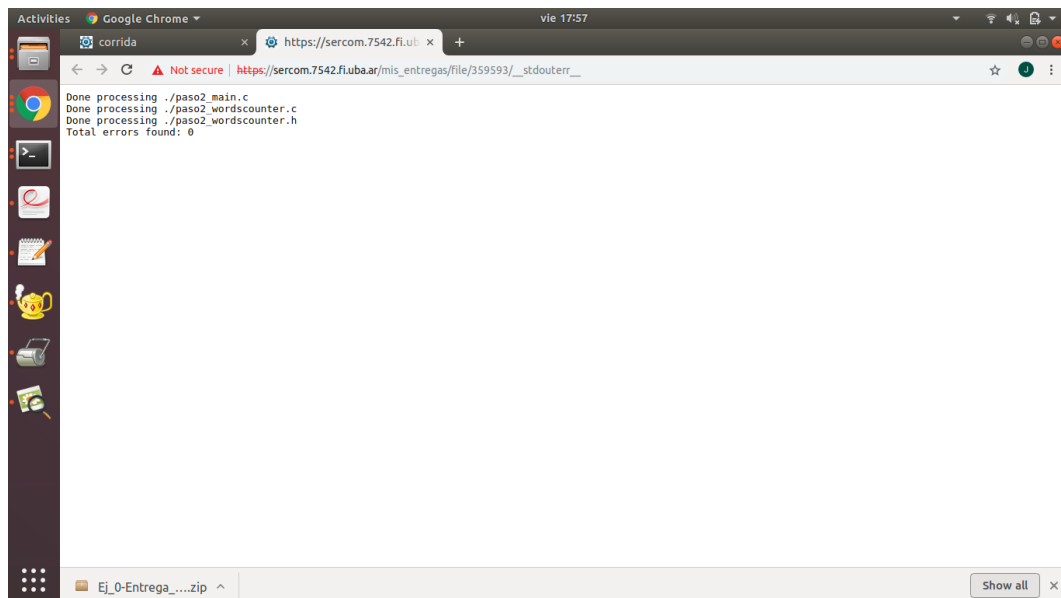


Figura 4.3: Entrega sin errores de estilo

4.3. Errores de generacion del ejecutable

Se explicara cada uno de los errores obtenidos, comenzare mostrando los errores que existen por el mismo problema, estos son:

- In file included from paso2_wordscouter.c:1:0: paso2_wordscouter.h:6:5: error: unknown type name 'size_t' size_t words;

4 Paso 2: Errores de generacion 2

- `paso_wordscouter.h:19:1: error: unknown type name 'size_t' size_t wordscounter_get_words(wordscounter_t *self);`
- `paso2_wordscouter.h:24:49: error: unknown type name 'FILE' void wordscounter_process(wordscounter_t *self, FILE *text_file);`

Estos errores se deben a que se estan tratando de usar tipos de datos del cual el compilador no tiene ninguna informacion, entonces no puede reservar espacio para el tipo de dato. La solucion a este problema es incluir dentro del archivo la libreria correspondiente (`stdio.h`).

Otro error que se debe al mismo problema de no incluir la libreria correspondiente es el siguiente:

- `paso2_wordscouter.c:18:8: error: conflicting types for 'wordscounter_get_words' size_t wordscounter_get_words(wordscounter_t *self) In file included from paso2_wordscouter.c:1:0: paso2_wordscouter.h:19:8: note: previous declaration of 'wordscounter_get_words' was here size_t wordscounter_get_words(wordscounter_t *self);`

El problema aca es que en el archivo `paso2_wordscouter.c` estaba incluida la libreria `stdio.h` entonces reconoce el tipo de dato que devuelve la funcion, en cambio en el archivo `paso2_wordscouter.h` no estaba incluida entonces el compilador da error a que la funcion ya fue declarada con un tipo en el archivo `.c` y esta declarada con otro tipo distinto en el archivo `.h`

Finalmente el error:

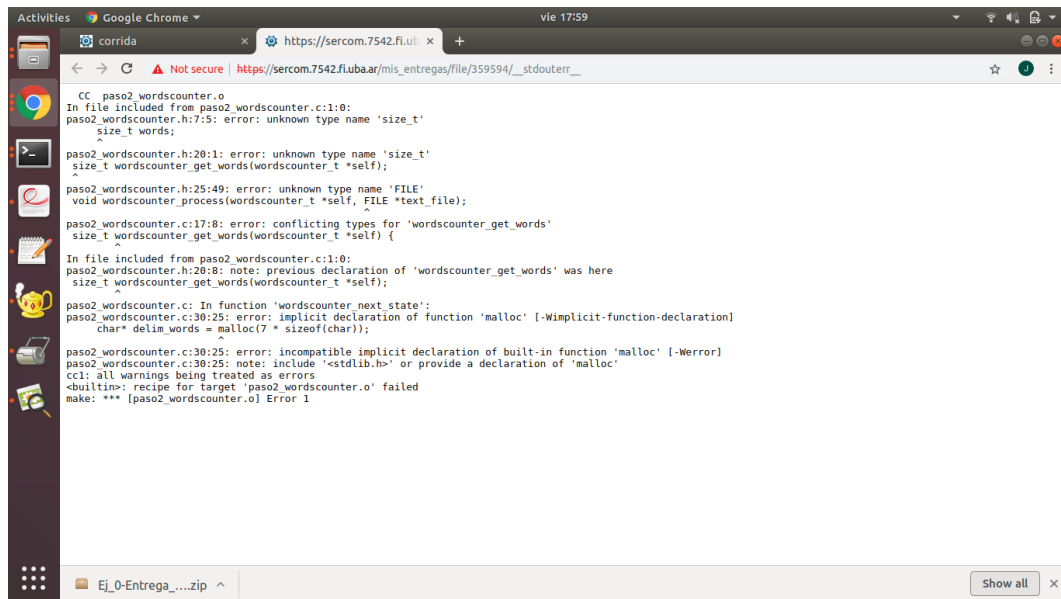
- `paso2_wordscouter.c: In function 'wordscounter_next_state': paso2_wordscouter.c:30:25: error: implicit declaration of function 'malloc' [-Wimplicit-function-declaration] char* delim_words = malloc(7 * sizeof(char)); paso2_wordscouter.c:30:25: error: incompatible implicit declaration of built-in function 'malloc' [-Werror] paso2_wordscouter.c: 30:25: note: include 'stdlib.h' or provide a declaration of 'malloc'`

El mismo compilador nos da una ayuda aca, diciendonos que estamos utilizando una funcion de la libreria `stdlib.h` sin haberla incluido.

Los errores de `'unknown type name'` y `'implicit declaration of function'` son al igual que en el paso1 errores previos a la etapa de compilacion.

El error `'conflicting types'` tambien ocurre antes de la etapa de linkeo ya que estoy incluyendo el archivo `paso2_wordscouter.h` dentro del `paso2_wordscouter.c` y este se incluye previo a la etapa de linkeo entonces el error ocurre dentro de la etapa de compilacion.

4 Paso 2: Errores de generacion 2



The screenshot shows a web browser window with the address bar displaying `https://sercom.7542.fi.uba.ar/mis_entregas/file/359594/_stdouterr_`. The main content area displays the output of a compilation process, showing several errors and warnings. The errors include: unknown type name 'size_t', conflicting types for 'wordscouter_get_words', and incompatible implicit declaration of built-in function 'malloc'. The process failed to create the target 'paso2_wordscouter.o'.

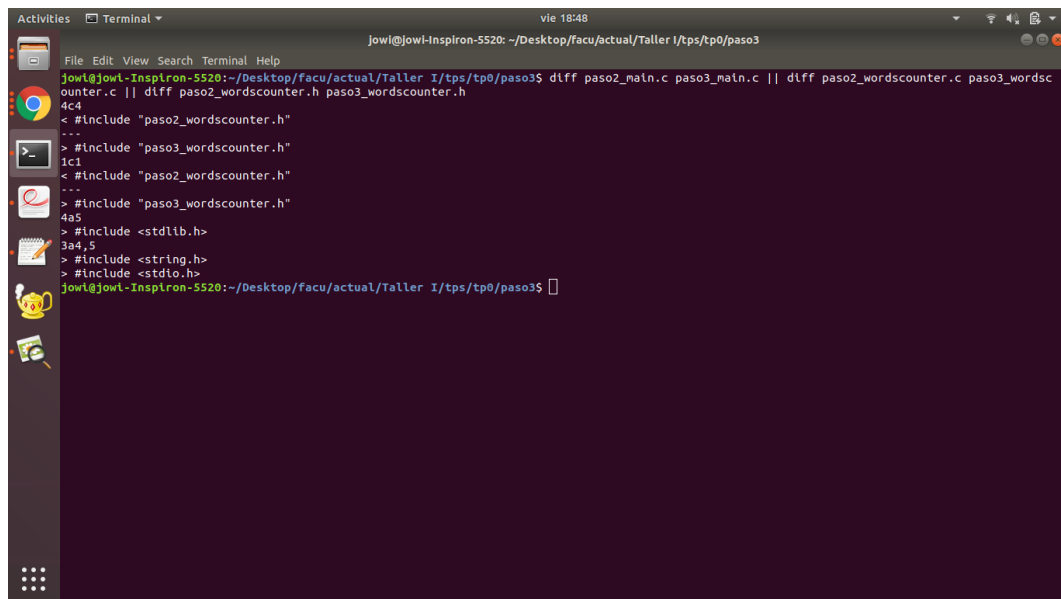
```
CC paso2_wordscouter.o
In file included from paso2_wordscouter.c:1:0:
paso2_wordscouter.h:7:5: error: unknown type name 'size_t'
    size_t words;
    ^
paso2_wordscouter.h:20:1: error: unknown type name 'size_t'
    size_t wordscounter_get_words(wordscounter_t *self);
    ^
paso2_wordscouter.h:25:49: error: unknown type name 'FILE'
    void wordscounter_process(wordscounter_t *self, FILE *text_file);
                                                    ^
paso2_wordscouter.c:17:8: error: conflicting types for 'wordscouter_get_words'
    size_t wordscounter_get_words(wordscounter_t *self) {
    ^
In file included from paso2_wordscouter.c:1:0:
paso2_wordscouter.h:20:1: note: previous declaration of 'wordscouter_get_words' was here
    size_t wordscounter_get_words(wordscounter_t *self);
    ^
paso2_wordscouter.c: In function 'wordscouter_next_state':
paso2_wordscouter.c:30:25: error: implicit declaration of function 'malloc' [-Wimplicit-function-declaration]
    char* delim_words = malloc(7 * sizeof(char));
                        ^
paso2_wordscouter.c:30:25: error: incompatible implicit declaration of built-in function 'malloc' [-Werror]
paso2_wordscouter.c:30:25: note: include '<stdlib.h>' or provide a declaration of 'malloc'
cc1: all warnings being treated as errors
-builtin: recipe for target 'paso2_wordscouter.o' failed
make: *** [paso2_wordscouter.o] Error 1
```

Figura 4.4: Errores de generacion del ejecutable

5 Paso 3: Errores de generacion 3

5.1. Correcciones realizadas respecto al Paso 2

Como se puede ver en la imagen 5.1 los cambios realizados respecto al paso2 fui incluir las librerias necesarias en cada archivo para evitar que ocurran los errores mencionados en el paso2.



```
Activities  Terminal  vie 18:48
jowl@jowl-Inspiron-5520: ~/Desktop/facu/actual/Taller I/tps/tp0/paso3
jowl@jowl-Inspiron-5520:~/Desktop/facu/actual/Taller I/tps/tp0/paso3$ diff paso2_main.c paso3_main.c || diff paso2_wordscoun
ter.c || diff paso2_wordscounter.h paso3_wordscounter.h
4c4
< #include "paso2_wordscounter.h"
...
> #include "paso3_wordscounter.h"
1c1
< #include "paso2_wordscounter.h"
...
> #include "paso3_wordscounter.h"
4a5
> #include <stdlib.h>
3a4,5
> #include <string.h>
> #include <stdio.h>
jowl@jowl-Inspiron-5520:~/Desktop/facu/actual/Taller I/tps/tp0/paso3$
```

Figura 5.1: Diferencias respecto al paso 2

5.2. Errores de generacion del ejecutable

El error obtenido se puede ver en la imagen 5.2, este quiere decir que hay una referencia al nombre 'wordscounter_destroy' dentro del main la cual no fue encontrada su definicion en ninguno de los archivos objeto o las librerias incluidas. Este error ocurre en la etapa de linkeo ya que se busca la definicion en archivos externos al main.

5 Paso 3: Errores de generacion 3

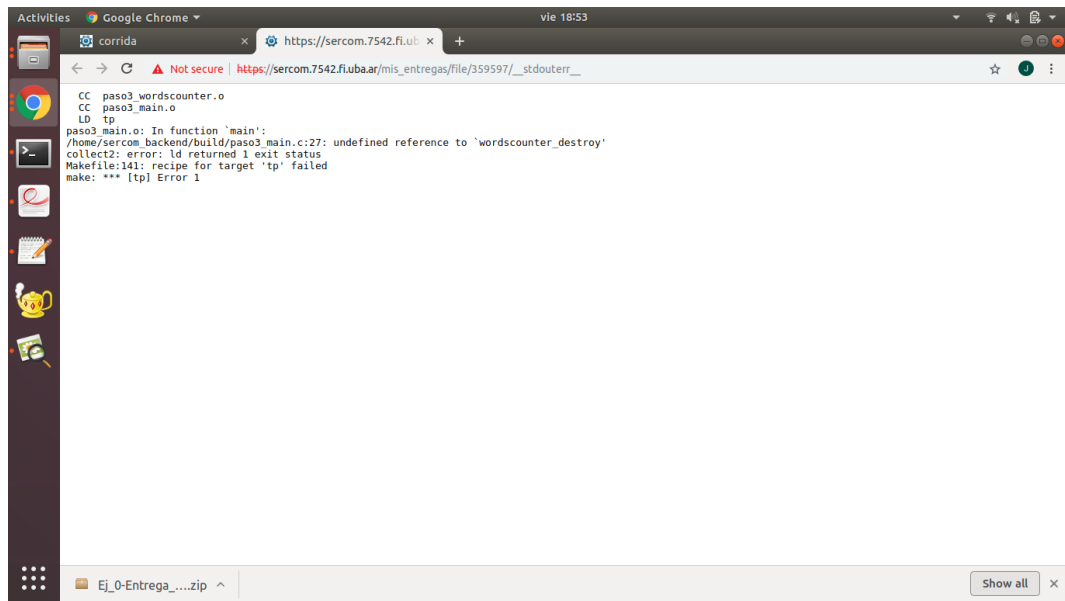


Figura 5.2: Errores de generacion del ejecutable

6 Paso 4: Memory Leaks y Buffer Overflows

6.1. Correcciones realizadas respecto al Paso 3

El principal cambio fue hecho en el archivo 'paso4_wordscouter.c' el cual se incluyo la definicion de la funcion 'wordscouter_destroy(wordscouter_t *self)'. Ademas se cambio el nombre de la libreria 'paso4_wordscouter.h', lo que causo que se vea en la imagen el cambio de los include (es un cambio irrelevante).



```
File Edit View Search Terminal Help
jowi@jowi-Inspiron-5520: ~/Desktop/facu/actual/Taller I/tps/tp0/paso4
jowi@jowi-Inspiron-5520:~/Desktop/facu/actual/Taller I/tps/tp0/paso4$ diff paso3_main.c paso4_main.c || diff paso3_wordscouter.c paso4_wordscouter.c || diff paso3_wordscouter.h paso4_wordscouter.h
4c4
< #include "paso3_wordscouter.h"
...
> #include "paso4_wordscouter.h"
1c1
< #include "paso3_wordscouter.h"
...
> #include "paso4_wordscouter.h"
15a16,19
> }
>
> void wordscouter_destroy(wordscouter_t *self) {
>     //do nothing
jowi@jowi-Inspiron-5520:~/Desktop/facu/actual/Taller I/tps/tp0/paso4$
```

Figura 6.1: Diferencias respecto al paso 3

6.2. Resultado de ejecucion con Valgrind de la prueba 'TDA'

Valgrind nos esta reportando que pedimos memoria para aloca bytes y nunca la liberamos. Podemos ver un resumen de la cantidad de bytes utilizados, cuantas veces pedimos memoria, cuantas veces la liberamos y saber que cantidad de bytes dejamos en uso al

momento del cierre del programa en el Heap Summary:

```
'==00:00:00:00.577 2806== HEAP SUMMARY:
==00:00:00:00.577 2806== in use at exit: 1,849 bytes in 216 blocks
==00:00:00:00.577 2806== total heap usage: 218 allocs, 2 frees, 10,041 bytes allocated'
```

Luego podemos ver mas en detalle porque perdimos memoria, en este caso fueron dos errores distintos. El primero, que dice:

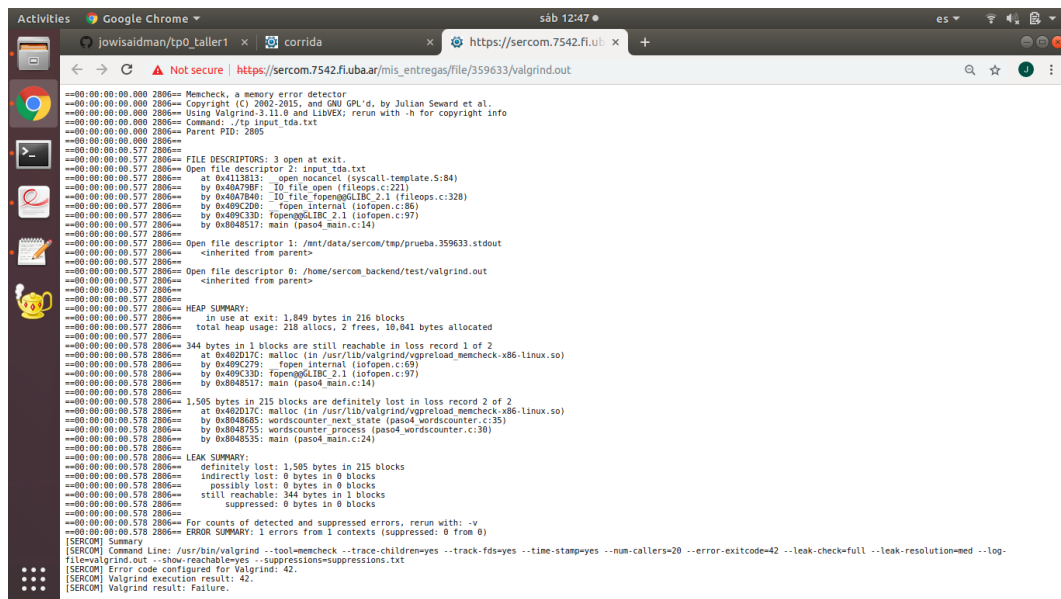
```
'==00:00:00:00.578 2806== 344 bytes in 1 blocks are still reachable in loss record 1 of
2
==00:00:00:00.578 2806== at 0x402D17C: malloc (in /usr/lib/valgrind/vgpreload_memcheck-
x86-linux.so)
==00:00:00:00.578 2806== by 0x409C279: _fopen_internal (iofopen.c:69)
==00:00:00:00.578 2806== by 0x409C33D: fopen@@GLIBC_2.1 (iofopen.c:97)
==00:00:00:00.578 2806== by 0x8048517: main (paso4_main.c:14)'
```

Este error indica que en el main se abrio un archivo y luego nunca fue cerrado, esto causo una perdida de memoria de 344 bytes. El segundo error que obtuve fue:

```
'==00:00:00:00.578 2806== 1,505 bytes in 215 blocks are definitely lost in loss record 2
of 2
==00:00:00:00.578 2806== at 0x402D17C: malloc (in /usr/lib/valgrind/vgpreload_memcheck-
x86-linux.so)
==00:00:00:00.578 2806== by 0x8048685: wordscounter_next_state (paso4_wordscounter.c:35)
==00:00:00:00.578 2806== by 0x8048755: wordscounter_process (paso4_wordscounter.c:30)
==00:00:00:00.578 2806== by 0x8048535: main (paso4_main.c:24)'
```

Esto quiere decir que hubo un llamado encadenado, primero el main llamo a wordscounter_process, luego wordscounter_process llamo a wordscounter_next_state y dentro de esta ultima fue llamada la funcion malloc. Los malloc no fueron liberados y fueron ejecutados 215 veces perdiendo un total de 1505 bytes.

6 Paso 4: Memory Leaks y Buffer Overflows



```
==00:00:00.000 2806== Memcheck, a memory error detector
==00:00:00.000 2806== Copyright (C) 2002-2015, and GNU GPL'd, by Julian Seward et al.
==00:00:00.000 2806== Using Valgrind-3.11.0 and LibVEX; rerun with -h for copyright info
==00:00:00.000 2806== Command: ./tp input_tda.txt
==00:00:00.000 2806== Parent PID: 2805

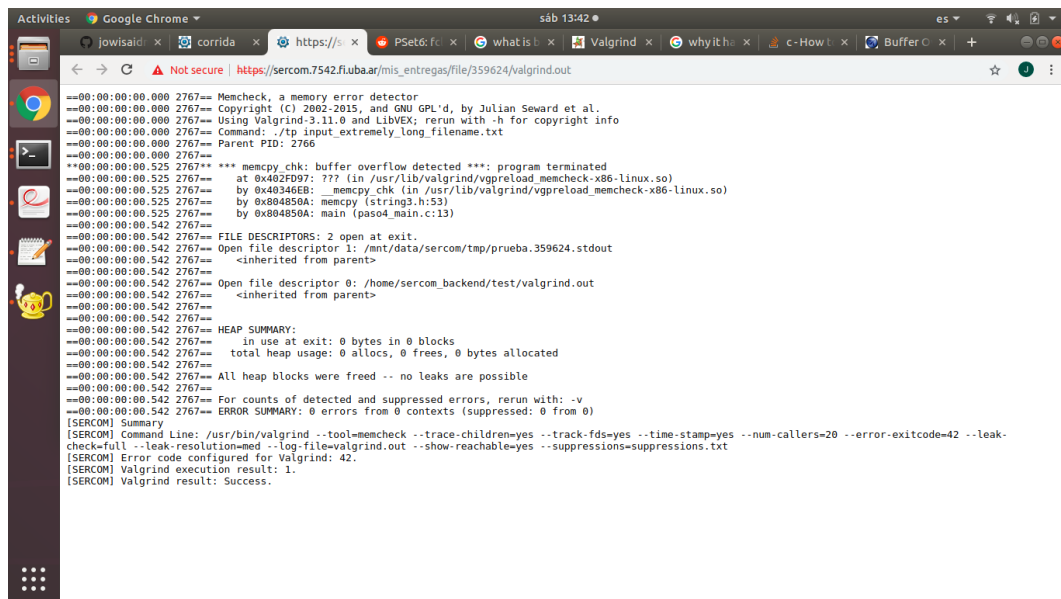
==00:00:00.577 2806==
==00:00:00.577 2806== FILE DESCRIPTORS: 3 open at exit.
==00:00:00.577 2806== Open file descriptor 2: input_tda.txt
==00:00:00.577 2806==   at 0x4113813: open_nocancel (syscall-template.S:84)
==00:00:00.577 2806==   by 0x40A789F: IO file open (fileops.c:221)
==00:00:00.577 2806==   by 0x40A7840: IO file fopen@Glibc 2.1 (fileops.c:328)
==00:00:00.577 2806==   by 0x409C200: fopen_internal (iofopen.c:86)
==00:00:00.577 2806==   by 0x409C33D: fopen@Glibc 2.1 (iofopen.c:97)
==00:00:00.577 2806==   by 0x08048517: main (pasos4_main.c:14)
==00:00:00.577 2806== Open file descriptor 1: /mnt/data/sercom/tp/prueba.359633.stdout
==00:00:00.577 2806==   inherited from parent+
==00:00:00.577 2806== Open file descriptor 0: /home/sercom/backend/test/valgrind.out
==00:00:00.577 2806==   inherited from parent+
==00:00:00.577 2806==
==00:00:00.577 2806== HEAP SUMMARY:
==00:00:00.577 2806==   in use at exit: 1,849 bytes in 216 blocks
==00:00:00.577 2806==   total heap usage: 218 allocs, 2 frees, 10,041 bytes allocated
==00:00:00.577 2806==
==00:00:00.578 2806== 344 bytes in 1 blocks are still reachable in loss record 1 of 2
==00:00:00.578 2806==   at 0x402D17C: malloc (in /usr/lib/valgrind/vgpreload_memcheck-x86-linux.so)
==00:00:00.578 2806==   by 0x409C798: fopen_internal (iofopen.c:69)
==00:00:00.578 2806==   by 0x409C33D: fopen@Glibc 2.1 (iofopen.c:97)
==00:00:00.578 2806==   by 0x08048517: main (pasos4_main.c:14)
==00:00:00.578 2806==
==00:00:00.578 2806== 1,505 bytes in 215 blocks are definitely lost in loss record 2 of 2
==00:00:00.578 2806==   at 0x402D17C: malloc (in /usr/lib/valgrind/vgpreload_memcheck-x86-linux.so)
==00:00:00.578 2806==   by 0x08048685: wordscounter_next_state (pasos4_wordscounter.c:35)
==00:00:00.578 2806==   by 0x08048755: wordscounter_process (pasos4_wordscounter.c:39)
==00:00:00.578 2806==   by 0x08048535: main (pasos4_main.c:24)
==00:00:00.578 2806==
==00:00:00.578 2806== LEAK SUMMARY:
==00:00:00.578 2806==   definitely lost: 1,505 bytes in 215 blocks
==00:00:00.578 2806==   indirectly lost: 0 bytes in 0 blocks
==00:00:00.578 2806==   possibly lost: 0 bytes in 0 blocks
==00:00:00.578 2806==   still reachable: 344 bytes in 1 blocks
==00:00:00.578 2806==   suppressed: 0 bytes in 0 blocks
==00:00:00.578 2806==
==00:00:00.578 2806== For counts of detected and suppressed errors, rerun with: -v
==00:00:00.578 2806== ERROR SUMMARY: 1 errors from 1 contexts (suppressed: 0 from 0)
[SERCOM] Summary
[SERCOM] Command Line: /usr/bin/valgrind --tool=memcheck --trace-children=yes --track-fds=yes --time-stamp=yes --num-callers=20 --error-exitcode=42 --leak-check=full --leak-resolution=med --log-file=valgrind.out --show-reachable=yes --suppressions=suppressions.txt
[SERCOM] Error code configured for Valgrind: 42.
[SERCOM] Valgrind execution result: 42.
[SERCOM] Valgrind result: Failure.
```

Figura 6.2: Errores de la prueba 'tda'

6.3. Resultado de ejecucion con Valgrind de la prueba 'Long Filename'

En este caso el error reportado por valgrind fue que hubo un buffer overflow dentro de la funcion main. Como se puede ver en la imagen 6.3, lo que ocurrio fue que la funcion memcpy quiso poner una cantidad de datos mayor a la que el buffer puede soportar. La prueba que se hacer es si el contador de palabras puede ejecutarse con archivos grandes y como la funcion memcpy esta siendo utilizada de forma que copie todo el archivo junto no entraron todos los datos en el buffer. [3]

6 Paso 4: Memory Leaks y Buffer Overflows



```
==00:00:00.000 2767== Memcheck, a memory error detector
==00:00:00.000 2767== Copyright (C) 2002-2015, and GNU GPL'd, by Julian Seward et al.
==00:00:00.000 2767== Using Valgrind-3.11.0 and LibVEX; rerun with -h for copyright info
==00:00:00.000 2767== Command: ./tp_input_extremely_long_filename.txt
==00:00:00.000 2767== Parent PID: 2766
==00:00:00.000 2767==
*** memcopy_chk: buffer overflow detected ***: program terminated
==00:00:00.525 2767== at 0x402f097: ??? (in /usr/lib/valgrind/vgpreload_memcheck-x86-linux.so)
==00:00:00.525 2767== by 0x40346E8: memcopy_chk (in /usr/lib/valgrind/vgpreload_memcheck-x86-linux.so)
==00:00:00.525 2767== by 0x804850A: memcopy (string3.h:53)
==00:00:00.525 2767== by 0x804850A: main (paso4_main.c:13)
==00:00:00.542 2767==
==00:00:00.542 2767== FILE DESCRIPTORS: 2 open at exit.
==00:00:00.542 2767== Open file descriptor 1: /mnt/data/sercom/tmp/prueba.359624.stdout
==00:00:00.542 2767== <inherited from parent>
==00:00:00.542 2767== Open file descriptor 0: /home/sercom_backend/test/valgrind.out
==00:00:00.542 2767== <inherited from parent>
==00:00:00.542 2767==
==00:00:00.542 2767== HEAP SUMMARY:
==00:00:00.542 2767== in use at exit: 0 bytes in 0 blocks
==00:00:00.542 2767== total heap usage: 0 allocs, 0 frees, 0 bytes allocated
==00:00:00.542 2767==
==00:00:00.542 2767== All heap blocks were freed -- no leaks are possible
==00:00:00.542 2767==
==00:00:00.542 2767== For counts of detected and suppressed errors, rerun with: -v
==00:00:00.542 2767== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
[SERCOM] Summary
[SERCOM] Command Line: /usr/bin/valgrind --tool=memcheck --trace-children=yes --track-fds=yes --time-stamp=yes --num-callers=20 --error-exitcode=42 --leak-check=full --leak-resolution=med --log-file=valgrind.out --show-reachable=yes --suppressions=suppressions.txt
[SERCOM] Error code configured for Valgrind: 42.
[SERCOM] Valgrind execution result: 1.
[SERCOM] Valgrind result: Success.
```

Figura 6.3: Errores de la prueba 'long file'

6.4. Funcion strncpy

Utilizando la funcion strncpy ocurriria el mismo error ya que se trataria de copiar el archivo entero y no se podria.[4]

6.5. Segmentation fault y un buffer overflow

El error de 'segmentation fault' indica que hubo un acceso a memoria que no estaba permitido, ya sea leer o escribir sobre memoria la cual no fue pedida o que esta mas alla de los limites que el sistema operativo guardo para la ejecucion de nuestro programa.

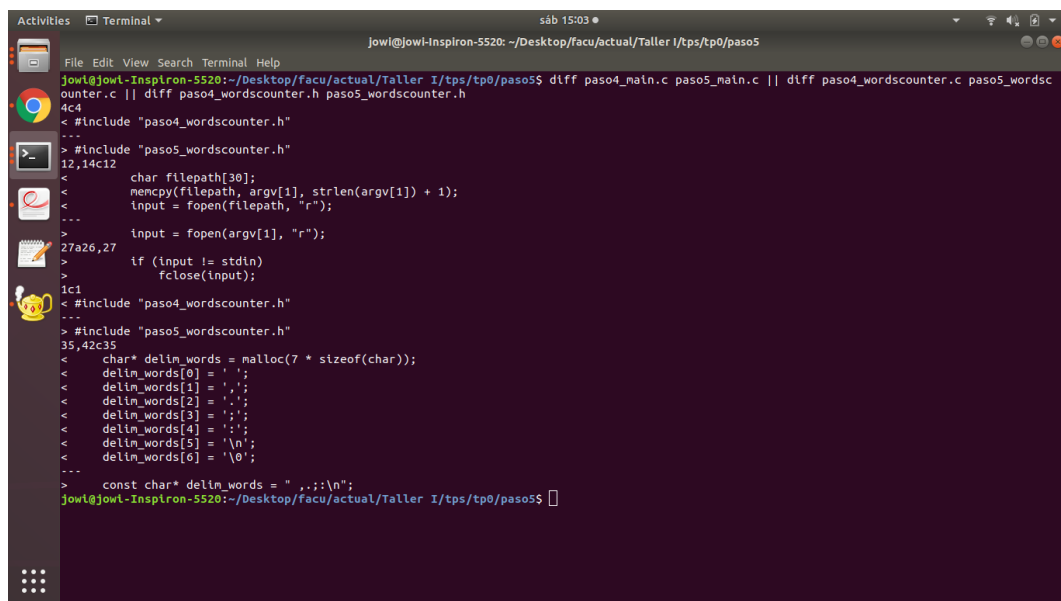
El error 'buffer overflow' ocurre cuando un programa intenta poner mas datos en un bufer de los que puede tener o quiere colocar datos en un area de memoria mas alla del bufer utilizado, un ejemplo de este error fue el visto en el punto c de este paso.

Un error de segmentation fault puede estar causado por un buffer overflow ya que se esta tratando de acceder a memoria que el programa no tiene los derechos para acceder. [3] [5] [6]

7 Paso 5: Codigo de retorno y salida estandar

7.1. Correcciones realizadas respecto al Paso 4

Uno de los cambios que se puede ver en la imagen 7.1 es que `delim_words` era un puntero a un string mutable que se guardaba en memoria dinamica y ahora paso a ser un puntero a un string inmutable que no ocupa memoria dinamica (D1). Otro cambio es que ya el main no utiliza `memcpy` entonces el archivo no se esta nunca todo el archivo junto en memoria (ya pasa el test de `longfile`). El ultimo cambio es que ahora dentro del main se cierra el archivo de entrada entonces no ocurre perdida de memoria por dejar el archivo abierto.



```
jowli@jowli-Inspiron-5520: ~/Desktop/facu/actual/Taller I/tps/tp0/paso5
jowli@jowli-Inspiron-5520:~/Desktop/facu/actual/Taller I/tps/tp0/paso5$ diff paso4_main.c paso5_main.c || diff paso4_wordscouter.c paso5_wordscouter.c
4c4
< #include "paso4_wordscouter.h"
...
> #include "paso5_wordscouter.h"
12,14c12
<     char filepath[30];
<     memcpy(filepath, argv[1], strlen(argv[1]) + 1);
<     input = fopen(filepath, "r");
...
>     input = fopen(argv[1], "r");
27a26,27
>     if (input != stdin)
>         fclose(input);
1c1
< #include "paso4_wordscouter.h"
...
> #include "paso5_wordscouter.h"
35,42c35
<     char* delim_words = malloc(7 * sizeof(char));
<     delim_words[0] = ' ';
<     delim_words[1] = ' ';
<     delim_words[2] = ' ';
<     delim_words[3] = ' ';
<     delim_words[4] = ' ';
<     delim_words[5] = '\n';
<     delim_words[6] = '\0';
...
>     const char* delim_words = " .,:;\n";
jowli@jowli-Inspiron-5520:~/Desktop/facu/actual/Taller I/tps/tp0/paso5$
```

Figura 7.1: Diferencias respecto al paso 4

7.2. Fallo de las pruebas 'Invalid File' y 'Single Word'

La prueba de 'Invalid File' La ayuda que da sercom para esta falla es que esperaba que devuelva 1 y 255. La prueba de 'Single Word' falla por que la salida que da el programa

no es la que debería ser, indica que hay 0 palabras cuando en realidad el archivo contiene 1 palabra. Esto ocurre por que el archivo no tiene ninguno de los caracteres que se utilizan como separadores de palabras entonces el programa no cuenta la palabra que contiene el archivo y el resultado da 0 cuando debería ser 1. Para este caso sercom dice que: 'La salida estándar no coincide con lo esperado (archivo '___stdout___diff').' y se puede ver que el stdout es un 0 cuando debería ser un 1.

En la imagen 7.2 se puede ver que sercom indica claramente que pruebas fallan en color rojo, da observaciones sobre porque fallo y se pueden descargar los archivos utilizados en la prueba para poder correrlo localmente hasta poder corregir el test.

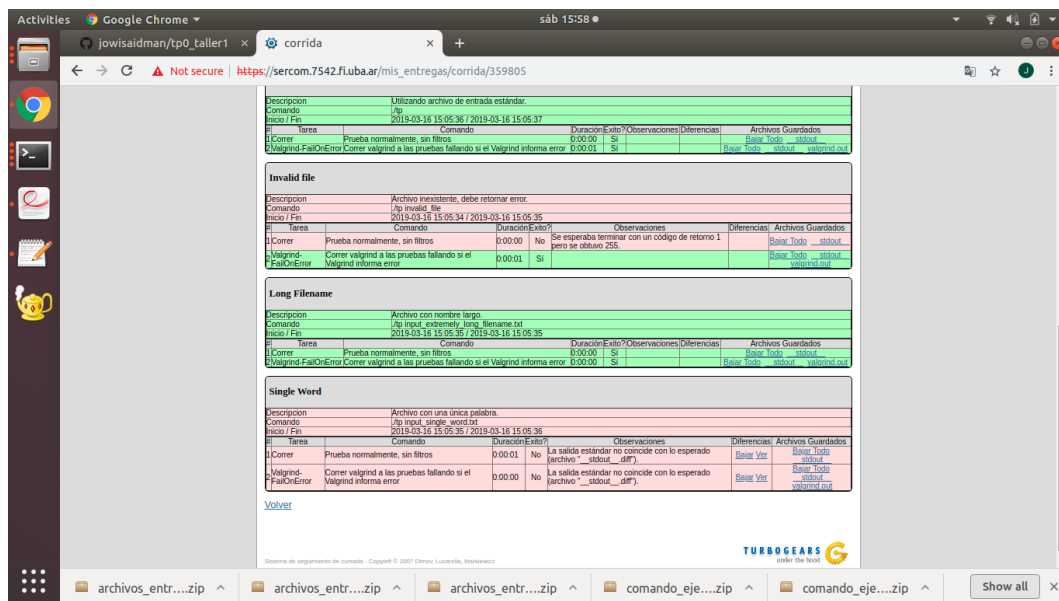


Figura 7.2: Fallo de las pruebas 'single word' y 'invalid file'

7.3. Comando hexdump

En la imagen 7.3 se puede ver que el ultimo caracter es un 64, que esta en base hexadecimal. Este numero al pasarlo a decimal es un 100 y viendo su valor en la tabla ASCII es una 'd', que es la ultima letra de la palabra que contiene el archivo.

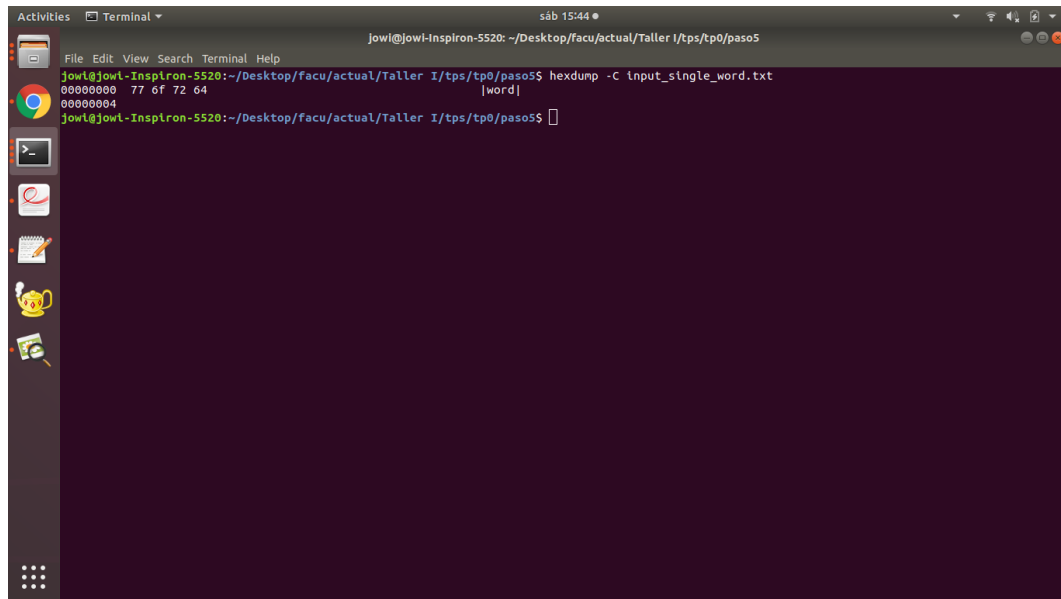


Figura 7.3: Fallo de las pruebas 'single word' y 'invalid file'

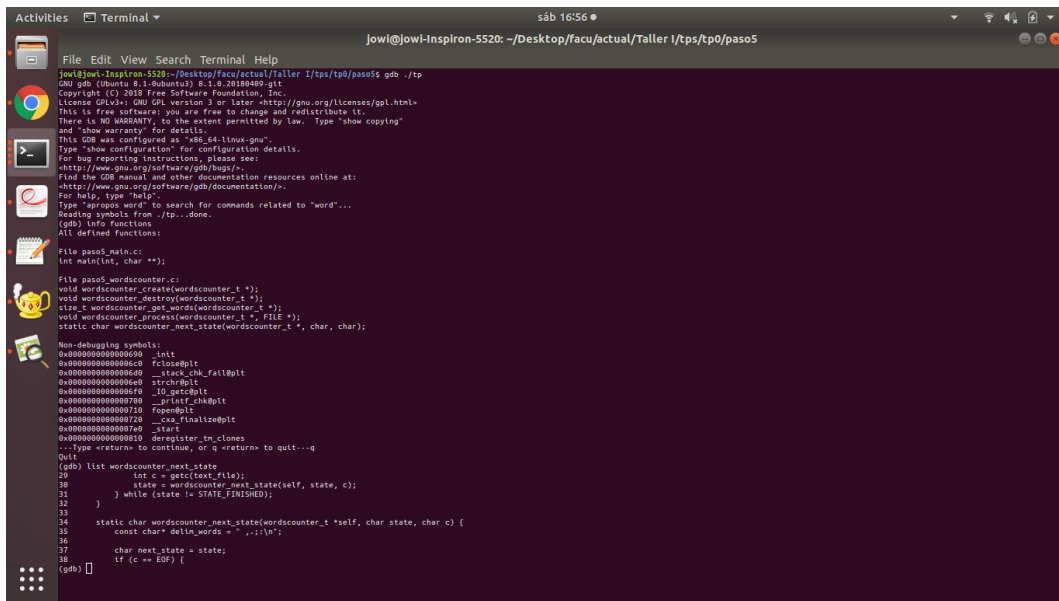
7.4. Ejecucion con GDB

A continuacion explicare cada uno de los comandos utilizados con gdb.

- info functions: Da informacion sobre las funciones en el programa.
- list wordscounter_next_state : Imprime las lineas que estan alrededor de la funcion wordscounter_next_state
- list: Imprime las siguientes 10 lineas de la ultima que se vio.
- break 45: El comando break trata ejecutar el programa hasta llegar a la linea indicada, en este caso 45.
- run input_single_word.txt : Corre el programa utilizando de input el archivo input_single_word.txt

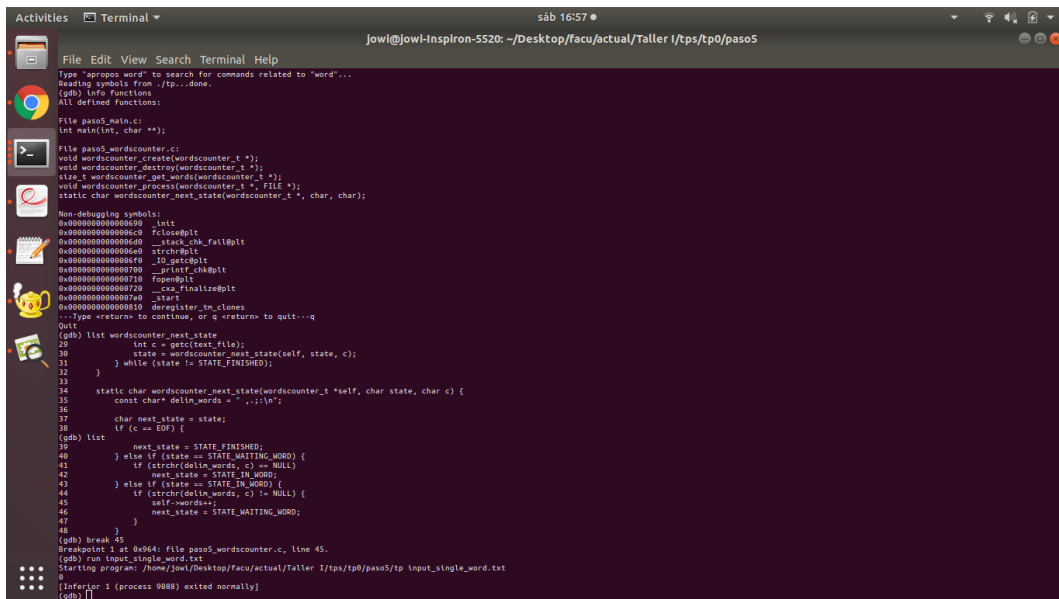
El debugger no se detuvo en el breakpoint de la linea 45 porque no ejecuto dicha linea, el breakpoint hubiera ocurrido si el programa pasaba por la linea indicada.

7 Paso 5: Código de retorno y salida estándar



```
Activities Terminal sáb 16:56 ● jowi@jowi-Inspiron-5520: ~/Desktop/facu/actual/Taller I/tps/tp0/paso5
File Edit View Search Terminal Help
jowi@jowi-Inspiron-5520:~/Desktop/facu/actual/Taller I/tps/tp0/paso5$ gdb ./tp
GNU gdb (Ubuntu 8.1-0ubuntu3) 8.1.0.20180409-git
Copyright (C) 2018 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 ./tp...done.
(gdb) info functions
All defined functions:
File paso5_main.c:
int main(int, char **);
File paso5_wordcounter.c:
void wordcounter_create(wordcounter_t *);
void wordcounter_destroy(wordcounter_t *);
size_t wordcounter_get_words(wordcounter_t *);
void wordcounter_process(wordcounter_t *, FILE *);
static char wordcounter_next_state(wordcounter_t *, char, char);
Non-debugging symbols:
0x0000000000000000 _init
0x0000000000000000 _fini
0x0000000000000000 __stack_chk_fail@plt
0x0000000000000000 __strchr@plt
0x0000000000000000 __IO_getc@plt
0x0000000000000000 __printf_chk@plt
0x0000000000000000 __fopen@plt
0x0000000000000000 __cxa_finalize@plt
0x0000000000000000 _start
0x0000000000000010 deregister_tm_clones
...Type <return> to continue, or q <return> to quit...q
Quit
(gdb) list wordcounter_next_state
29     int c = getc(stdin);
30     state = wordcounter_next_state(self, state, c);
31     } while (state != STATE_FINISHED);
32 }
33
34 static char wordcounter_next_state(wordcounter_t *self, char state, char c) {
35     const char* delin_words = " .,:!\"'";
36
37     char next_state = state;
38     if (c == EOF) {
39
40         next_state = STATE_FINISHED;
41     } else if (state == STATE_WAITING_WORD) {
42         if (strchr(delin_words, c) == NULL) {
43             next_state = STATE_IN_WORD;
44         } else if (state == STATE_IN_WORD) {
45             if (strchr(delin_words, c) != NULL) {
46                 self->words++;
47                 next_state = STATE_WAITING_WORD;
48             }
49         }
50     }
51 }
52
53 (gdb) break 45
Breakpoint 1 at 0x064: file paso5_wordcounter.c, line 45.
(gdb) run input_single_word.txt
Starting program: /home/jowi/Desktop/facu/actual/Taller I/tps/tp0/paso5/tp input_single_word.txt
0
[Inferior 1 (process 9080) exited normally]
(gdb) |
```

Figura 7.4: Ejecucion de comandos con gdb 1



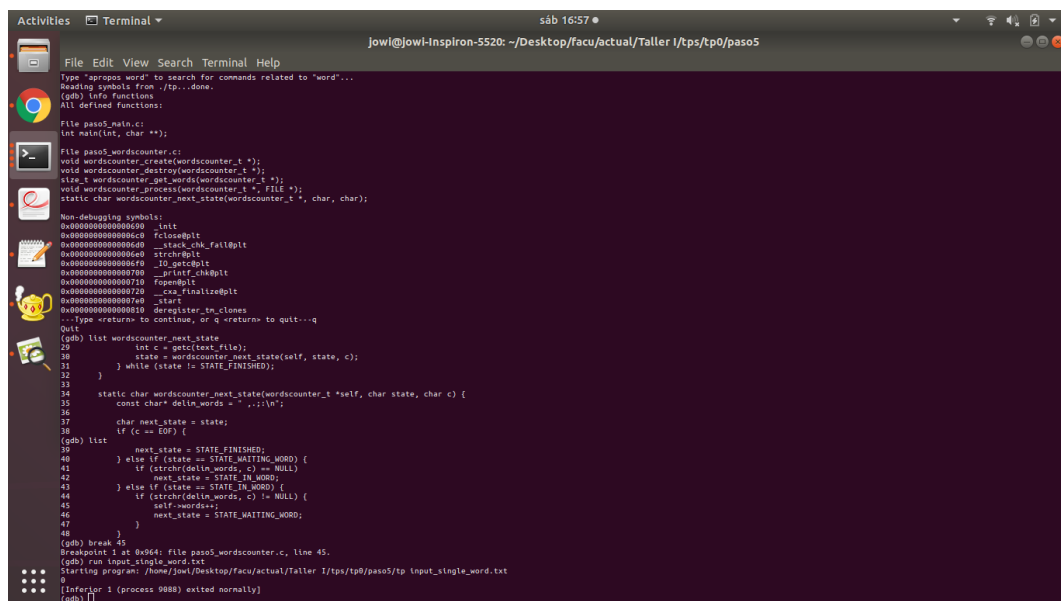
```
Activities Terminal sáb 16:57 ● jowi@jowi-Inspiron-5520: ~/Desktop/facu/actual/Taller I/tps/tp0/paso5
File Edit View Search Terminal Help
Type "apropos word" to search for commands related to "word"...
Reading symbols from ./tp...done.
(gdb) info functions
All defined functions:
File paso5_main.c:
int main(int, char **);
File paso5_wordcounter.c:
void wordcounter_create(wordcounter_t *);
void wordcounter_destroy(wordcounter_t *);
size_t wordcounter_get_words(wordcounter_t *);
void wordcounter_process(wordcounter_t *, FILE *);
static char wordcounter_next_state(wordcounter_t *, char, char);
Non-debugging symbols:
0x0000000000000000 _init
0x0000000000000000 _fini
0x0000000000000000 __stack_chk_fail@plt
0x0000000000000000 __strchr@plt
0x0000000000000000 __IO_getc@plt
0x0000000000000000 __printf_chk@plt
0x0000000000000000 __fopen@plt
0x0000000000000000 __cxa_finalize@plt
0x0000000000000000 _start
0x0000000000000010 deregister_tm_clones
...Type <return> to continue, or q <return> to quit...q
Quit
(gdb) list wordcounter_next_state
29     int c = getc(stdin);
30     state = wordcounter_next_state(self, state, c);
31     } while (state != STATE_FINISHED);
32 }
33
34 static char wordcounter_next_state(wordcounter_t *self, char state, char c) {
35     const char* delin_words = " .,:!\"'";
36
37     char next_state = state;
38     if (c == EOF) {
39
40         next_state = STATE_FINISHED;
41     } else if (state == STATE_WAITING_WORD) {
42         if (strchr(delin_words, c) == NULL) {
43             next_state = STATE_IN_WORD;
44         } else if (state == STATE_IN_WORD) {
45             if (strchr(delin_words, c) != NULL) {
46                 self->words++;
47                 next_state = STATE_WAITING_WORD;
48             }
49         }
50     }
51 }
52
53 (gdb) break 45
Breakpoint 1 at 0x064: file paso5_wordcounter.c, line 45.
(gdb) run input_single_word.txt
Starting program: /home/jowi/Desktop/facu/actual/Taller I/tps/tp0/paso5/tp input_single_word.txt
0
[Inferior 1 (process 9080) exited normally]
(gdb) |
```

Figura 7.5: Ejecucion de comandos con gdb 2

8 Paso 6: Entrega exitosa

8.1. Correcciones realizadas respecto al Paso 5

Se definio a DELIM_WORDS como una constante global, ademas se corrigio la funcion wordscouter_next_state de forma que si el proximo caracter encontrado es el EOF y esta en una palabra el contador suma, de esta forma se corrigio el caso de un archivo de una sola palabra.



```
Activities Terminal
Jowi@Jowi-Inspiron-5520: ~/Desktop/facu/actual/Taller I/tps/tp0/paso5

Type "apropos word" to search for commands related to 'word'...
Reading symbols from ./tps...done.
(gdb) info functions
All defined functions:
File paso5_main.c:
int main(int, char **);

File paso5_wordscouter.c:
void wordscouter_create(wordscouter_t *);
void wordscouter_destroy(wordscouter_t *);
size_t wordscouter_get_word(wordscouter_t *);
void wordscouter_process(wordscouter_t *, FILE *);
static char wordscouter_next_state(wordscouter_t *, char, char);

Non-debugging symbols:
0x0000000000000000 int
0x0000000000000000 _fclose@plt
0x0000000000000000 _stack_chk_fail@plt
0x0000000000000000 _strchr@plt
0x0000000000000000 _IO_getc@plt
0x0000000000000000 _printf_chk@plt
0x0000000000000000 _fopen@plt
0x0000000000000000 _cxa_finalize@plt
0x0000000000000000 _start
0x0000000000000000 deregister_tm_clones
...Type <return> to continue, or q <return> to quit...q
Quit
(gdb) list wordscouter_next_state
29 int c = getc(text_file);
30 state = wordscouter_next_state(self, state, c);
31 } while (state != STATE_FINISHED);
32
33
34 static char wordscouter_next_state(wordscouter_t *self, char state, char c) {
35     const char* delim_words = " \t\n";
36     char next_state = state;
37     if (c == EOF) {
38         (gdb) list
39         next_state = STATE_FINISHED;
40     } else if (state == STATE_WAITING_WORD) {
41         if (strchr(delim_words, c) == NULL)
42             next_state = STATE_IN_WORD;
43     } else if (state == STATE_IN_WORD) {
44         if (strchr(delim_words, c) != NULL) {
45             self->word++;
46             next_state = STATE_WAITING_WORD;
47         }
48     }
49 }
(gdb) break 45
Breakpoint 1 at 0x004: file paso5_wordscouter.c, line 45.
(gdb) run input_single_word.txt
Starting program: /home/jowi/Desktop/facu/actual/Taller I/tps/tp0/paso5/tp_input_single_word.txt
[Inferior 1 (process 9088) exited normally]
(gdb) ]
```

Figura 8.1: Ejecucion de comandos con gdb 2

8.2. Entregas realizadas

A continuacion podemos ver una foto con todas las entregas realizadas hasta el paso 6.

8 Paso 6: Entrega exitosa



Figura 8.2: Ejecucion de comandos con gdb 2

8.3. Ejecucion de la prueba 'Single Word' de forma local

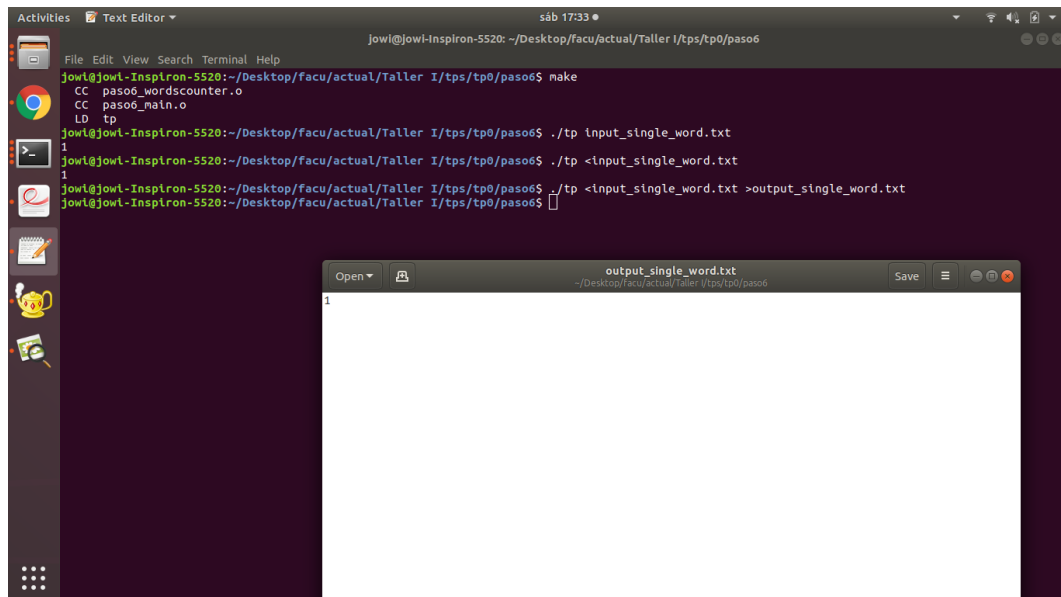


Figura 8.3: Output de la ejecucion del programa con el archivo single word

9 Paso 7

Bibliografía

- [1] <http://valgrind.org/docs/manual/quick-start.html>
- [2] <https://stackoverflow.com/questions/29365611/does-running-valgrind-slow-down-my-application>
- [3] https://www.owasp.org/index.php/Buffer_Overflow
- [4] <https://randomascii.wordpress.com/2013/04/03/stop-using-strncpy-already/>
- [5] <https://smallbusiness.chron.com/segmentation-fault-linux-27699.html>
- [6] http://web.mit.edu/10.001/Web/Tips/tips_on_segmentation.html