

Paso 0:

a) ScreenShot

```
mati@Mati ~/Documents/FIUBA/taller/tp$ gcc main.c -o tp
mati@Mati ~/Documents/FIUBA/taller/tp$ ./tp
Hola Mundo
mati@Mati ~/Documents/FIUBA/taller/tp$ valgrind ./tp
==11344== Memcheck, a memory error detector
==11344== Copyright (C) 2002-2015, and GNU GPL'd, by Julian Seward et al.
==11344== Using Valgrind-3.11.0 and LibVEX; rerun with -h for copyright info
==11344== Command: ./tp
==11344==
Hola Mundo==11344==
==11344== HEAP SUMMARY:
==11344==    in use at exit: 0 bytes in 0 blocks
==11344==   total heap usage: 1 allocs, 1 frees, 1,024 bytes allocated
==11344==
==11344== All heap blocks were freed -- no leaks are possible
==11344==
==11344== For counts of detected and suppressed errors, rerun with: -v
==11344== ERROR SUMMARY: 0 errors from 0 contexts (suppressed: 0 from 0)
mati@Mati ~/Documents/FIUBA/taller/tp$
```

b) Valgrind

Sirve como herramienta para verificar la correcta ejecución de los programas. Su herramienta más común y popular se llama Memcheck. La misma permite detectar (en la mayoría de los casos) errores relacionados con el uso de la memoria (comunes en los lenguajes C/C++).

Tiene muchas formas de uso, las más comunes son `--leak-check` detecta pérdida de memoria de forma detallada.

Otra opción interesante para nuestra materia seguramente sea `--trace-children=<yes|no>`, donde la misma rastreará los subprocesos en las ejecuciones.

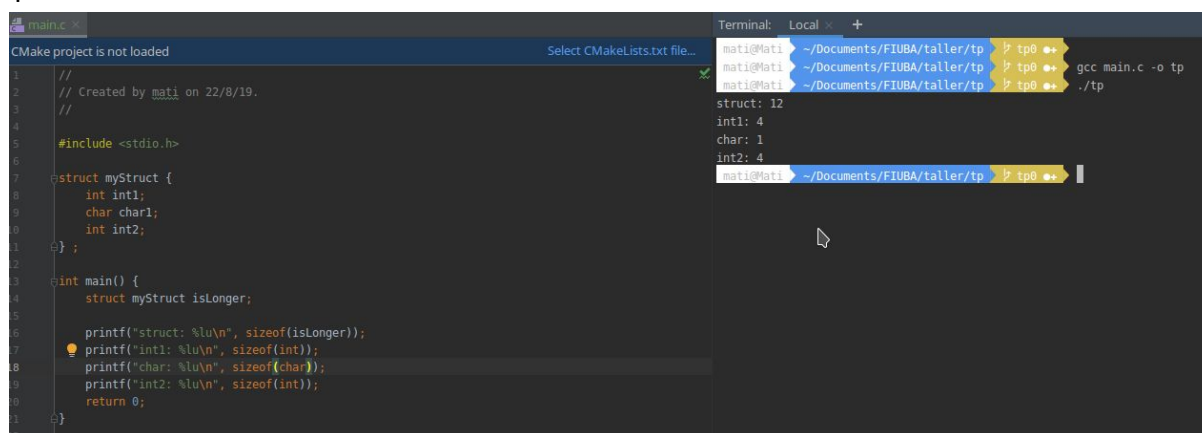
Entre otras también se destaca la `--vgdb=<no|yes|full>`, que seguramente será de gran utilidad para poder hacer un debug más exhaustivo viendo si hay pérdida de memoria hasta cierto punto.

c) sizeof()

Es una función que acepta como único parámetro una variable o un tipo y devuelve su tamaño en bytes. El mismo puede variar dependiendo de la arquitectura, por eso el valor de salida `sizeof(char)` o `sizeof(int)` dependerá de la arquitectura donde corra el programa.

d) sizeof() struct

El sizeof de un struct sin tener la “propiedad” `__packed__` siempre va a ser mayor o igual a la suma de cada uno de sus elementos. Para ejemplificar esto, adjunto una imagen que lo demuestra:



The screenshot shows a CMake IDE with a C program on the left and a terminal on the right. The C program defines a struct `myStruct` with two `int` elements (4 bytes each), one `char` element (1 byte), and another `int` element (4 bytes). The `main` function prints the size of the struct and its components. The terminal output shows the program executed, with the struct size being 12, which is greater than the sum of its individual elements (9).

```
// Created by mati on 22/8/19.
//
#include <stdio.h>

struct myStruct {
    int int1;
    char char1;
    int int2;
};

int main() {
    struct myStruct isLonger;

    printf("struct: %lu\n", sizeof(isLonger));
    printf("int1: %lu\n", sizeof(int));
    printf("char: %lu\n", sizeof(char));
    printf("int2: %lu\n", sizeof(int));
    return 0;
}
```

```
mat@Mati: ~/Documents/FIUBA/taller/tp
mat@Mati: ~/Documents/FIUBA/taller/tp$ gcc main.c -o tp
mat@Mati: ~/Documents/FIUBA/taller/tp$ ./tp
struct: 12
int1: 4
char: 1
int2: 4
```

Donde la suma de cada elemento es 9, mientras que la del struct es 12.

e) STDIN, STDOUT y STDERR

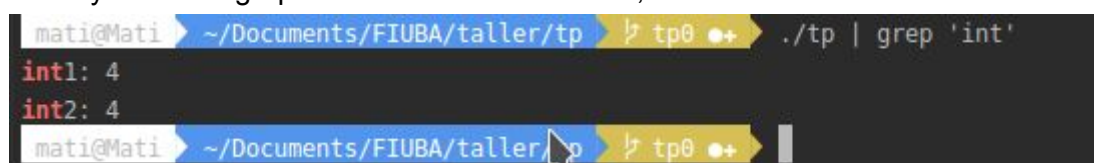
Son archivos que sirven como canales de entrada y salida de información, en nuestro caso de un programa.

En cuanto a cada uno de ellos, el STDIN sirve como canal de entrada (generalmente usado por el usuario). Mientras que el STDOUT y STDERR sirven como canales de salida. Donde STDOUT es la salida estándar del programa donde podemos informar algún mensaje o resultado al usuario u otro programa. Mientras que el STDERR es la salida estándar de error del programa. Es decir ante un fallo, el error (mensaje) se va a imprimir en tal archivo.

Para redirigir los canales de entrada y salida estándar, basta como utilizar “>” o “<” en la línea de ejecución. Por ejemplo `./tp > salida.txt`, generará un archivo donde aparecerá toda la salida estándar que antes salía por pantalla en el archivo `salida.txt`. Si en vez de poner un solo “>” ponemos dos, es decir `./tp >> salida.txt`, agregará al fin de archivo la salida del programa.

Lo mismo sucede con los demás, en el caso del input el signo es el inverso “<” y tomará el archivo como standard input, es decir `./tp < input.txt`.

En cuanto al carácter **pipe** “|”, el mismo permite tomar la salida estándar de uno con la entrada estándar del otro. En el ejemplo realizado anteriormente puedo tomar la salida y hacer un grep del resultado de la misma, seleccionando solo el tamaño del tipo `int`.



The screenshot shows a terminal window where the command `./tp | grep 'int'` is executed. The output shows the lines `int1: 4` and `int2: 4` from the program's output, filtered by the `grep` command.

```
mat@Mati: ~/Documents/FIUBA/taller/tp$ ./tp | grep 'int'
int1: 4
int2: 4
```

Paso 1:

a) Problemas de estilos

./paso1_wordscouter.c:27: Missing space before (in while([whitespace/parens] [5]

falta un espacio entre while y el inicio del paréntesis

./paso1_wordscouter.c:41: Mismatching spaces inside () in if [whitespace/parens] [5]

No coinciden los espacios dentro del paréntesis

./paso1_wordscouter.c:41: Should have zero or one spaces inside (and) in if [whitespace/parens] [5]

dentro del paréntesis del if debería haber uno o ningún espacio, en este caso tiene 2

./paso1_wordscouter.c:47: An else should appear on the same line as the preceding } [whitespace/newline] [4]

El else if debería estar en la línea de arriba, separado por un espacio después de }

./paso1_wordscouter.c:47: If an else has a brace on one side, it should have it on both [readability/braces] [5]

Interpreta de que solo tiene el } de la derecha, al no estar arriba el else, piensa que solo tiene una llave. Ambos puntos se soluciona subiendo el else arriba.

./paso1_wordscouter.c:48: Missing space before (in if([whitespace/parens] [5]

Falta un espacio entre el if y el inicio del paréntesis

./paso1_wordscouter.c:53: Extra space before last semicolon. If this should be an empty statement, use {} instead. [whitespace/semicolon] [5]

Tiene un espacio demás entre next state y el “;” . Deberían ir pegados

Done processing ./paso1_wordscouter.c

./paso1_main.c:12: Almost always, snprintf is better than strcpy [runtime/printf] [4]

Nos dice que utilicemos snprintf en vez de strcpy. Pero también es recomendable usar strncpy, donde podemos poner la longitud de copiado, evitando problemas de overflow.

./paso1_main.c:15: An else should appear on the same line as the preceding } [whitespace/newline] [4]

Idem ./paso1_wordscouter.c:47 , donde el else debería ir en la línea de arriba.

./paso1_main.c:15: If an else has a brace on one side, it should have it on both [readability/braces] [5]

Idem ./paso1_wordscouter.c:47 , donde interpreta que le falta una {

Done processing ./paso1_main.c

./paso1_wordscouter.h:5: Lines should be <= 80 characters long [whitespace/line_length] [2]

La línea 5 tiene 3 caracteres demás por línea. Debería pasar la palabra archivo abajo a una nueva línea.

Done processing ./paso1_wordscouter.h

Total errors found: 11

b) Errores en la generación del ejecutable

```
CC paso1_main.o
paso1_main.c: In function 'main':
paso1_main.c:22:9: error: unknown type name 'wordscounter_t'
    wordscounter_t counter;
    ^
paso1_main.c:23:9: error: implicit declaration of function 'wordscounter_create'
[-Wimplicit-function-declaration]
    wordscounter_create(&counter);
    ^
paso1_main.c:24:9: error: implicit declaration of function 'wordscounter_process'
[-Wimplicit-function-declaration]
    wordscounter_process(&counter, input);
    ^
paso1_main.c:25:24: error: implicit declaration of function 'wordscounter_get_words'
[-Wimplicit-function-declaration]
    size_t words = wordscounter_get_words(&counter);
                   ^
paso1_main.c:27:9: error: implicit declaration of function 'wordscounter_destroy'
[-Wimplicit-function-declaration]
    wordscounter_destroy(&counter);
    ^
<builtin>: recipe for target 'paso1_main.o' failed
make: *** [paso1_main.o] Error 1
```

Los errores se deben a que se están llamando funciones de `paso1_wordscounter` y no se está incluyendo dicho archivo en `main.c`. Para eso, basta solo realizar un `#include "paso1_wordscounter.h"`

Este error se da en el linker. Una forma de corroborarlo es compilar el código sin linkear (donde no genera error) y compilar el código linkeando (donde se produce el error).

c) Warning

El sistema no generó ningún warning ya que el Makefile proporcionado por la cátedra (entendiendo que el mismo que tiene SERCOM) corre con la opción de **-Werror**, donde todos los warnings son considerados errores, deteniendo además el proceso.

Paso 2:

a) Correcciones

- Main.c :
 - Se incluyó `paso2_wordscounter.h`
 - Se cambio la función `strcpy` por `memcpy`
 - También se alineó el código `} else {` en una sola línea
- `paso2_wordcounter.h`:
 - Se cambió el comentario para que abarque menos de 80 caracteres.
- `paso2_wordcounter.c`:
 - Se realizó primero el `#include paso2_wordscounter.h` antes que el resto de las librerías.
 - Se alineó el texto en la línea 13, 45 y 51.

b) Screenshot

96277 (Matías Lahore) - 0.1

Comandos Ejecutados

#	Tarea	Comando	Inicio	Duración	Exito?	Observaciones	Diferencias	Archivos Guardados
1	Verificar Normas Codificación	<code>python ./cpplint.py --extensions=h,hpp,c,cpp --filter='cat filter_options' `find -regextype posix-egrep -regex '.*\.(h hpp c cpp)'</code>	2019-08-23 11:40:51	0:00:00	Sí			Bajar Todo stdouterr
1	Compilar C99 simple	<code>make -f Makefile</code>	2019-08-23 11:40:51	0:00:00	No	Se esperaba terminar con un código de retorno 0 pero se obtuvo 2.		Bajar Todo stdouterr

c) Errores de ejecución

CC `paso2_wordscounter.o`

In file included from `paso2_wordscounter.c`:1:0:

`paso2_wordscounter.h`:7:5: error: unknown type name 'size_t'

```
size_t words;
^
```

`paso2_wordscounter.h`:20:1: error: unknown type name 'size_t'

```
size_t wordscounter_get_words(wordscounter_t *self);
```

Error de compilación: Ambos errores de `size_t` se dan porque primero quiere compilar lo que hay en el `.h` sin antes incluir la librería de `string`

```
^
```

`paso2_wordscounter.h`:25:49: error: unknown type name 'FILE'

```
void wordscounter_process(wordscounter_t *self, FILE *text_file);
```

Error de compilación: Idem que el anterior, pero le falta primero la librería `stdio`

```
^
```

```

paso2_wordscouter.c:17:8: error: conflicting types for 'wordscouter_get_words'
size_t wordscouter_get_words(wordscouter_t *self) {
    ^
Error de compilación: Idem que el primero

In file included from paso2_wordscouter.c:1:0:
paso2_wordscouter.h:20:8: note: previous declaration of 'wordscouter_get_words' was
here
size_t wordscouter_get_words(wordscouter_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));
                        ^
Error de compilación: Falto agregar la libreria stdlib, donde está definida la función
malloc

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
<built-in>: recipe for target 'paso2_wordscouter.o' failed
make: *** [paso2_wordscouter.o] Error 1
Error de compilación: Idem que el anterior, falto incluir la libreria stdlib.

```

Paso 3:

a) Correcciones

- Main.c :
 - Se cambio el `paso2_wordscouter.h` por `paso3_wordscouter.h`
- `paso2_wordcounter.h`:
 - Se incluye las librerías `string` y `stdio`
 -
- `paso2_wordcounter.c`:
 - Se incluye la librería `stdlib`.

b) Errores de ejecución

```

CC paso3_wordscouter.o
CC paso3_main.o
LD tp

```

```
paso3_main.o: In function `main':
/home/sercom_backend/build/paso3_main.c:27: undefined reference to
`wordscounter_destroy'
collect2: error: ld returned 1 exit status
Makefile:141: recipe for target 'tp' failed
make: *** [tp] Error 1
```

Error del linker: Si bien la función está definida en `paso3_wordscounter.h`, no está implementada. Por lo que al compilar no hay problema, pero falla al momento de realizar la link edición.

Paso 4:

a) Correcciones

- Main.c :
 - No se realizaron cambios
- `paso2_wordcounter.h`:
 - No se realizaron cambios
- `paso2_wordcounter.c`:
 - Se implementó (vacía) la función `wordscounter_destroy`

b) Errores de Valgrind TDA

```
==00:00:00.000 3582== Memcheck, a memory error detector
==00:00:00.000 3582== Copyright (C) 2002-2015, and GNU GPL'd, by Julian Seward et al.
==00:00:00.000 3582== Using Valgrind-3.11.0 and LibVEX; rerun with -h for copyright info
==00:00:00.000 3582== Command: ./tp input_tda.txt
==00:00:00.000 3582== Parent PID: 3581
==00:00:00.000 3582==
==00:00:00.553 3582==
==00:00:00.553 3582== FILE DESCRIPTORS: 3 open at exit.
==00:00:00.553 3582== Open file descriptor 2: input_tda.txt
==00:00:00.553 3582==   at 0x4113813: __open_nocancel (syscall-template.S:84)
==00:00:00.553 3582==   by 0x40A79BF: _IO_file_open (fileops.c:221)
==00:00:00.553 3582==   by 0x40A7B40: _IO_file_fopen@@GLIBC_2.1 (fileops.c:328)
==00:00:00.553 3582==   by 0x409C2D0: __fopen_internal (iofopen.c:86)
==00:00:00.553 3582==   by 0x409C33D: fopen@@GLIBC_2.1 (iofopen.c:97)
==00:00:00.553 3582==   by 0x8048517: main (paso4_main.c:14)
==00:00:00.553 3582==
==00:00:00.553 3582== Open file descriptor 1: /mnt/data/sercom/tmp/prueba.392942.stdout
==00:00:00.553 3582==   <inherited from parent>
==00:00:00.553 3582==
==00:00:00.553 3582== Open file descriptor 0: /home/sercom_backend/test/valgrind.out
==00:00:00.553 3582==   <inherited from parent>
==00:00:00.553 3582==
==00:00:00.553 3582==
==00:00:00.553 3582== HEAP SUMMARY:
==00:00:00.553 3582==   in use at exit: 1,849 bytes in 216 blocks
==00:00:00.553 3582==   total heap usage: 218 allocs, 2 frees, 10,041 bytes allocated
==00:00:00.553 3582==
```

```

==00:00:00:00.554 3582== 344 bytes in 1 blocks are still reachable in loss record 1 of 2
==00:00:00:00.554 3582== at 0x402D17C: malloc (in /usr/lib/valgrind/vgpreload_memcheck-x86-linux.so)
==00:00:00:00.554 3582== by 0x409C279: __fopen_internal (iofopen.c:69)
==00:00:00:00.554 3582== by 0x409C33D: fopen@@GLIBC_2.1 (iofopen.c:97)
==00:00:00:00.554 3582== by 0x8048517: main (paso4_main.c:14)
==00:00:00:00.554 3582==
==00:00:00:00.554 3582== 1,505 bytes in 215 blocks are definitely lost in loss record 2 of 2
==00:00:00:00.554 3582== at 0x402D17C: malloc (in /usr/lib/valgrind/vgpreload_memcheck-x86-linux.so)
==00:00:00:00.554 3582== by 0x8048685: wordscounter_next_state (paso4_wordscounter.c:35)
==00:00:00:00.554 3582== by 0x8048755: wordscounter_process (paso4_wordscounter.c:30)
==00:00:00:00.554 3582== by 0x8048535: main (paso4_main.c:24)
==00:00:00:00.554 3582==
==00:00:00:00.554 3582== LEAK SUMMARY:
==00:00:00:00.554 3582== definitely lost: 1,505 bytes in 215 blocks
==00:00:00:00.554 3582== indirectly lost: 0 bytes in 0 blocks
==00:00:00:00.554 3582== possibly lost: 0 bytes in 0 blocks
==00:00:00:00.554 3582== still reachable: 344 bytes in 1 blocks
==00:00:00:00.554 3582== suppressed: 0 bytes in 0 blocks
==00:00:00:00.554 3582==
==00:00:00:00.554 3582== For counts of detected and suppressed errors, rerun with: -v
==00:00:00:00.554 3582== 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.

```

En resumidas cuentas, sucede de que por un lado se está dejando abierto el archivo (que se abrió en el main línea 14). Por otro lado no se está liberando la memoria reservada en wordscounter, línea 35. Bastaría con hacer un `fclose(input)` antes de terminar el programa y un `free(delim_words)` antes de retornar el valor en la función `wordscounter_next_state`.

c) Errores de Valgrind con Long Filename

```

==00:00:00:00.000 3543== Memcheck, a memory error detector
==00:00:00:00.000 3543== Copyright (C) 2002-2015, and GNU GPL'd, by Julian Seward et al.
==00:00:00:00.000 3543== Using Valgrind-3.11.0 and LibVEX; rerun with -h for copyright info
==00:00:00:00.000 3543== Command: ./tp input_extremely_long_filename.txt
==00:00:00:00.000 3543== Parent PID: 3542
==00:00:00:00.000 3543==
**00:00:00:00.516 3543** *** memcpy_chk: buffer overflow detected ***: program terminated
==00:00:00:00.516 3543== at 0x402FD97: ??? (in /usr/lib/valgrind/vgpreload_memcheck-x86-linux.so)
==00:00:00:00.516 3543== by 0x40346EB: __memcpy_chk (in /usr/lib/valgrind/vgpreload_memcheck-x86-linux.so)
==00:00:00:00.516 3543== by 0x804850A: memcpy (string3.h:53)
==00:00:00:00.516 3543== by 0x804850A: main (paso4_main.c:13)
==00:00:00:00.532 3543==
==00:00:00:00.532 3543== FILE DESCRIPTORS: 2 open at exit.
==00:00:00:00.532 3543== Open file descriptor 1: /mnt/data/sercom/tmp/prueba.392933.stdout
==00:00:00:00.532 3543== <inherited from parent>
==00:00:00:00.532 3543==
==00:00:00:00.532 3543== Open file descriptor 0: /home/sercom_backend/test/valgrind.out
==00:00:00:00.532 3543== <inherited from parent>
==00:00:00:00.532 3543==

```



```

==00:00:00:00.532 3543==
==00:00:00:00.532 3543== HEAP SUMMARY:
==00:00:00:00.532 3543==    in use at exit: 0 bytes in 0 blocks
==00:00:00:00.532 3543==   total heap usage: 0 allocs, 0 frees, 0 bytes allocated
==00:00:00:00.532 3543==
==00:00:00:00.532 3543== All heap blocks were freed -- no leaks are possible
==00:00:00:00.532 3543==
==00:00:00:00.532 3543== For counts of detected and suppressed errors, rerun with: -v
==00:00:00:00.532 3543== 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.

```

Es un problema visto en clase (parecido). Lo que sucede es que memcpy quiere copiar en la cadena de caracteres "filepath" la ruta del archivo que ocupa más de 30. Por consiguiente lo que sucede es un overflow, como indica valgrind. Por otro lado además se esta tomando el largo de la cadena del origen y no del destino. Esto no es problema siempre y cuando sepamos que el destino es igual o más largo que el origen.

d) Strncpy

Otra forma posible es usar strncpy. Donde también se le puede indicar la cantidad a copiar. Pero de todas formas como menciono anteriormente, hay que tomar el largo del "filepath" (destino). Ya que si no sucede de que se pisa memoria del programa, generando así algun bug/crasheo del mismo, como sucede en la ejecución de esta prueba.

e) Segmentation fault y buffer overflow

Un segmentation fault o violación de acceso es un problema que sucede en tiempo de ejecución. El mismo se produce cuando se quiere acceder/escribir una porción de memoria que no le pertenece al segmento del programa. En el caso anterior se está escribiendo parte de la ruta en "filepath" y el resto en alguna parte de la memoria que no le pertenece, produciendo así un segmentation fault.

En cuanto al buffer overflow, es algo parecido, sin ser necesariamente un segmentation fault. Sucede cuando a una variable se le escribe más contenido del que puede albergar, pero el resto del contenido se termina de escribir en alguna otra variable del programa. Este caso se estudió en la clase, tanto como un error del programador, pero sobretudo un problema de seguridad del mismo, ya que se podía manipular el comportamiento del programa a través de un buffer overflow.

Paso 5:

a) Correcciones

- Main.c :
 - Se eliminó strncpy y se le pasa argv[1] directamente al fopen
- paso2_wordcounter.h:
 - No se realizaron cambios
- paso2_wordcounter.c:
 - delim_words ahora tiene definido todos los separadores en una misma línea.

b) 'Invalid file' y 'Single Word'

Invalid file							
Descripción		Archivo inexistente, debe retornar error.					
Comando		./tp invalid_file					
Inicio / Fin		2019-08-26 13:10:18 / 2019-08-26 13:10:18					
#	Tarea	Comando	Duración	Exito?	Observaciones	Diferencias	Archivos Guardados
1	Correr	Prueba normalmente, sin filtros	0:00:00	No	Se esperaba terminar con un código de retorno 1 pero se obtuvo 255.		Bajar Todo stdout
2	Valgrind-FailOnError	Correr valgrind a las pruebas fallando si el Valgrind informa error	0:00:00	Sí			Bajar Todo stdout valgrind.out

El problema está en el código de error esperado. La prueba espera que al no poder abrir el archivo (ya que no existe) le devuelva un código de error 1. Pero el programa devuelve otro número. Con cambiar el -1 por 1 debería pasar la prueba.

Single Word							
Descripción		Archivo con una única palabra.					
Comando		./tp input_single_word.txt					
Inicio / Fin		2019-08-26 13:10:19 / 2019-08-26 13:10:20					
#	Tarea	Comando	Duración	Exito?	Observaciones	Diferencias	Archivos Guardados
1	Correr	Prueba normalmente, sin filtros	0:00:00	No	La salida estándar no coincide con lo esperado (archivo "___stdout__.diff").	Bajar Ver	Bajar Todo stdout
2	Valgrind-FailOnError	Correr valgrind a las pruebas fallando si el Valgrind informa error	0:00:00	No	La salida estándar no coincide con lo esperado (archivo "___stdout__.diff").	Bajar Ver	Bajar Todo stdout valgrind.out

El problema está en que la función 'wordscouter_next_state' no contempla el caso de que exista una sola palabra sin limitador. Por lo tanto cuenta solo cuando hay un delimitador. Con agregar que si llego a fin de archivo cuente una vez más antes de retornar la cantidad debería pasar la prueba.

c) Hexdump

```
mati@Mati ~/Documents/FIUBA/taller/tp $ tp0 + hexdump -C input_single_word.txt
00000000  77 6f 72 64                                     |word|
00000004
```

El último carácter es la 'd' en hexadecimal.

d) Gdb

```
mati@Mati ~/Documents/FIUBA/taller/tp $ tp0 + gdb ./tp
GNU gdb (Ubuntu 7.11.1-0ubuntu1~16.5) 7.11.1
Copyright (C) 2016 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_wordscounter.c:
void wordscounter_create(wordscounter_t *);
void wordscounter_destroy(wordscounter_t *);
size_t wordscounter_get_words(wordscounter_t *);
void wordscounter_process(wordscounter_t *, FILE *);
static char wordscounter_next_state(wordscounter_t *, char, char);

Non-debugging symbols:
0x0000000000400570 _init
0x00000000004005a0 _fclose@plt
0x00000000004005b0 __stack_chk_fail@plt
0x00000000004005c0 strchr@plt
0x00000000004005d0 __libc_start_main@plt
0x00000000004005e0 _IO_getc@plt
0x00000000004005f0 __printf_chk@plt
0x0000000000400600 fopen@plt
0x00000000004006d0 _start
0x0000000000400700 deregister_tm_clones
0x0000000000400740 register_tm_clones
0x0000000000400780 __do_global_ctors_aux
0x00000000004007a0 frame_dummy
0x00000000004008a0 __libc_csu_init
0x0000000000400910 __libc_csu_fini
0x0000000000400914 _fini
(gdb) █
```

Inicia gdb y le pasa el programa a debuggear.

Con info functions lista todas las funciones definidas en nuestro programa 'tp'

```
(gdb) list wordscounter_next_state
29         int c = getc(text_file);
30         state = wordscounter_next_state(self, state, c);
31     } while (state != STATE_FINISHED);
32 }
33
34 static char wordscounter_next_state(wordscounter_t *self, char state, char c) {
35     const char* delim_words = " ,.;:\n";
36
37     char next_state = state;
38     if (c == EOF) {
(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->words++;
46             next_state = STATE_WAITING_WORD;
47         }
48     }
(gdb)
```

El comando 'list' lista 10 líneas del programa. Si no se le pasa algún argumento lista las primeras 10 líneas.

En este caso le pasamos el nombre de una función. Para eso listo +-5 líneas de donde estaba declarada. Con el siguiente comando 'list' volvió a imprimir las siguientes 10 líneas. También se le pueden pasar otro tipo de índices, como dirección de memoria o líneas de un archivo.

```
(gdb) break 45
Breakpoint 1 at 0x400822: file paso5_wordscounter.c, line 45.
(gdb) run input_single_word.txt
Starting program: /home/mati/Documents/FIUBA/taller/tp/tp input_single_word.txt
0
[Inferior 1 (process 16227) exited normally]
(gdb) quit
mat@Mati ~/Documents/FIUBA/taller/tp > ? tp0 +
```

Agregar un breakpoint en la línea 45 y después corrió el programa pasándole el archivo 'input_single_word.txt' para que lo procese. Por último con 'quit' sale del entorno de gdb.

El programa no se detuvo, ya que como mencione anteriormente tiene un bug de que no va a contar ninguna palabra debido a que el último carácter no es un '.' o ';' y por lo tanto llega a un EOF, sin entrar en el if. Por último retorna 0.

Paso 6:

a) Correcciones

- Main.c :
 - Se cambio el código de retorno de error de '-1' por '1'
- paso2_wordcounter.h:
 - No se realizaron cambios
- paso2_wordcounter.c:
 - Se definió `delim_words` en vez de usar `const char*`
 - Se mejoró el caso de pruebas 'single_word' donde toma en cuenta un archivo solo de una palabra sin limitador.

b) Screenshot entregas realizadas

Ejercicio	Resultado	Fecha	Duración	Observaciones	Operaciones
0.1 (Contador de Palabras)	Aceptado	2019-08-26 14:46:59	0:00:06		Corrida Bajar Navegar PDF
0.1 (Contador de Palabras)	Rechazado	2019-08-26 13:10:13	0:00:06		Corrida Bajar Navegar PDF
0.1 (Contador de Palabras)	Rechazado	2019-08-24 09:06:55	0:00:12		Corrida Bajar Navegar PDF
0.1 (Contador de Palabras)	Rechazado	2019-08-23 17:26:11	0:00:01		Corrida Bajar Navegar PDF
0.1 (Contador de Palabras)	Rechazado	2019-08-23 11:40:41	0:00:00		Corrida Bajar Navegar PDF
0.1 (Contador de Palabras)	Rechazado	2019-08-23 09:35:29	0:00:01		Corrida Bajar Navegar PDF

c) Screenshot single_word local

```
mati@Mati ~/Documents/FIUBA/taller/tp % tp0 +
mati@Mati ~/Documents/FIUBA/taller/tp % tp0 + make
CC paso6_wordcounter.o
CC paso6_main.o
LD tp
mati@Mati ~/Documents/FIUBA/taller/tp % tp0 + ./tp input_single_word.txt
1
mati@Mati ~/Documents/FIUBA/taller/tp % tp0 + ./tp <input_single_word.txt
1
mati@Mati ~/Documents/FIUBA/taller/tp % tp0 + ./tp <input_single_word.txt >output_single_word.txt
mati@Mati ~/Documents/FIUBA/taller/tp % tp0 +
```