

mar 19, 19 16:28

paso6_wordscouter.h

Page 1/1

```

1  #ifndef __WORDSCOUNTER_H__
2  #define __WORDSCOUNTER_H__
3
4  #include <string.h>
5  #include <stdio.h>
6
7  // Tipo wordscounter_t: procesa cantidad de palabras dentro de un archivo.
8  typedef struct {
9      size_t words;
10 } wordscounter_t;
11
12 // Inicializa la instancia self para ser utilizada
13 // Pre: self apunta un sector válido de memoria
14 void wordscounter_create(wordscounter_t *self);
15
16 // Destruye la instancia self liberando sus recursos.
17 // Pre: self fue inicializado mediante wordscounter_create
18 void wordscounter_destroy(wordscounter_t *self);
19
20 // Retorna la cantidad de palabras procesadas
21 // Pre: self fue inicializado mediante wordscounter_create
22 size_t wordscounter_get_words(wordscounter_t *self);
23
24 // Procesa el contenido de text_file, contando sus palabras.
25 // Pre: self fue inicializado mediante wordscounter_create.
26 //      text_file es un archivo válido, abierto para lectura.
27 void wordscounter_process(wordscounter_t *self, FILE *text_file);
28
29 #endif
30

```

mar 19, 19 16:28

paso6_wordscouter.c

Page 1/1

```

1  #include "paso6_wordscouter.h"
2  #include <string.h>
3  #include <stdio.h>
4  #include <stdbool.h>
5  #include <stdlib.h>
6
7  #define STATE_WAITING_WORD 0
8  #define STATE_IN_WORD 1
9  #define STATE_FINISHED 2
10 #define DELIM_WORDS " ,;:\n"
11
12 // Compara el caracter leÃ-do c y define el nuevo estado.
13 static char wordscounter_next_state(wordscounter_t *self, char state, char c);
14
15 void wordscounter_create(wordscounter_t *self) {
16     self->words = 0;
17 }
18
19 void wordscounter_destroy(wordscounter_t *self) {
20     //do nothing
21 }
22
23 size_t wordscounter_get_words(wordscounter_t *self) {
24     return self->words;
25 }
26
27 void wordscounter_process(wordscounter_t *self, FILE *text_file) {
28     char state = STATE_WAITING_WORD;
29     do {
30         int c = getc(text_file);
31         state = wordscounter_next_state(self, state, c);
32     } while (state != STATE_FINISHED);
33 }
34
35 static char wordscounter_next_state(wordscounter_t *self, char state, char c) {
36     char next_state = state;
37
38     if (state == STATE_WAITING_WORD) {
39         if (c == EOF) {
40             next_state = STATE_FINISHED;
41         } else if (strchr(DELIM_WORDS, c) == NULL) {
42             next_state = STATE_IN_WORD;
43         }
44     } else if (state == STATE_IN_WORD) {
45         if (c == EOF) {
46             next_state = STATE_FINISHED;
47             self->words++;
48         } else if (strchr(DELIM_WORDS, c) != NULL) {
49             self->words++;
50             next_state = STATE_WAITING_WORD;
51         }
52     }
53
54     return next_state;
55 }
56

```

mar 19, 19 16:28

pas06_main.c

Page 1/1

```
1 #include <string.h>
2 #include <stdio.h>
3 #include <stdbool.h>
4 #include "pas06_wordscounter.h"
5
6 #define SUCCESS 0
7 #define ERROR 1
8
9 int main(int argc, char* argv[]) {
10     FILE* input;
11     if (argc > 1) {
12         input = fopen(argv[1], "r");
13     } else {
14         input = stdin;
15     }
16
17     if (!input) {
18         return ERROR;
19     } else {
20         wordscounter_t counter;
21         wordscounter_create(&counter);
22         wordscounter_process(&counter, input);
23         size_t words = wordscounter_get_words(&counter);
24         printf("%zu\n", words);
25         wordscounter_destroy(&counter);
26         if (input != stdin)
27             fclose(input);
28         return SUCCESS;
29     }
30 }
31
```

mar 19, 19 16:28

Table of Content

Page 1/1

1	Table of Contents					
2	1 pas06_wordscounter.h sheets	1 to	1 (1) pages	1-	1	31 lines
3	2 pas06_wordscounter.c sheets	1 to	1 (1) pages	2-	2	57 lines
4	3 pas06_main.c..... sheets	2 to	2 (1) pages	3-	3	32 lines