
UBA FACULTAD DE INGENIERÍA

66.20 Organización de Computadoras

Trabajo Práctico 0

2^{do} Cuatrimestre 2017

Integrantes:

Rodriquez Longhi, Federico	93336
federico.rlonghi@gmail.com	
Nicolas	NNNNN
...@...	
Juampa	NNNNN
...@...	

Índice

1. Introducción	2
2. Documentación	2
3. Compilación	2
4. Pruebas	2
4.1. Corridas de Prueba	3
5. Código en C	4
6. Código en MIPS	7
7. Enunciado	17

1. Introducción

El trabajo práctico consistió en la elaboración de un programa escrito en lenguaje C, el cual consistía en el procesamiento de texto para determinar palabras palíndromas dentro del mismo. El código fue ejecutado sobre el sistema operativo linux y netbsd (provisto por el curso). Dentro del ambiente virtual se compilo el código para obtener la salida en código MIPS.

2. Documentación

El uso del programa se compone de las siguientes opciones que le son pasadas por parámetro:

- `-h` o `--help`: muestra la ayuda.
- `-V` o `--version`: muestra la versión.
- `-i` o `--input`: recibe como parámetro un archivo de texto como entrada. En caso de que no usar esta opción, se toma como entrada la entrada estándar.
- `-o` o `--output`: recibe como parámetro un archivo de texto como salida. En caso de que no usar esta opción, se toma como salida la salida estándar.

3. Compilación

El programa puede ser compilado ubicándose en la carpeta que contiene el código fuente `tp0.c` y correr el siguiente comando:

```
gcc -Wall -o "tp0tp0.c"
```

También se provee de un script `compilar` el cual nos compilará el código automáticamente:

```
./compilar
```

4. Pruebas

Para las pruebas se proveen de dos scripts que las ejecutan. El primer script `test.sh` ejecuta los ejemplos del enunciado.

El segundo script `test_p.sh` ejecuta las pruebas propias. Este archivo esta diseñado para poder agregar pruebas de forma sencilla, simplemente se debe agregar una linea en el sector de pruebas de la siguiente manera:

```
make_test <nombre><entrada de texto><salida esperada>
```

Este script crea los archivos correspondientes en la carpeta tests (dentro del directorio sobre el cual se ejecuta). Los archivos creados son de la forma:

- test-<nombre del test>_in: archivo de entrada
- test-<nombre del test>_out: archivo de salida generado por el programa
- test-<nombre del test>_expected: archivo de salida esperado

4.1. Corridas de Prueba

A continuación se muestran las corridas de prueba generadas por el script:

```
1
2 Compiling Source
3 Compilation Success
4 Starting Tests
5
6 Test: one_letter_a
7 Test passed
8
9 Test: empty_file
10 Test passed
11
12 Test: no_palindroms
13 Test passed
14
15 Test: todos_palindromos
16 Test passed
17
18 Test: varias_lineas
19 Test passed
20
21 Test: all_letters
22 Test passed
23
24 Test: case_sensitive
25 Test passed
26
27 Test: numbers_and_letters
28 Test passed
29
30 Test: text_with_dash
31 Test passed
32
33 -----
34 All 9 tests passed!!!
35 -----
```

5. Código en C

```

1  #include <stdio.h>
2  #include <stdlib.h>
3
4  #include <ctype.h>
5  #include <unistd.h>
6  #include <getopt.h>
7  #include <errno.h>
8  #include <string.h>
9
10 /* imprimir el uso de tp0 */
11 void print_usage() {
12     printf("Usage: tp0 -i [input_file] -o [output_file]\n");
13 }
14
15 /* imprimir la pagina de ayuda */
16 void print_help() {
17     printf("\tUsage:\n"
18         "\t\ttp0 -h\n"
19         "\t\ttp0 -V\n"
20         "\t\ttp0 [options]\n"
21         "\tOptions:\n"
22         "\t\t-V, --version\tPrint version and quit.\n"
23         "\t\t-h, --help\tPrint this information.\n"
24         "\t\t-i, --input\tLocation of the input file.\n"
25         "\t\t-o, --output\tLocation of the output file.\n"
26         "\tExamples:\n"
27         "\t\ttp0 -i ~/input -o ~/output\n");
28 }
29
30 /* imprimir la version del programa */
31 void print_version(){
32     printf("tp0 1.0\n");
33 }
34
35 /* funcion para determinar si una palabra es capicua o no */
36 int es_capicua(char *palabra){
37
38     size_t len = strlen(palabra); // hay que ver si se puede
39                                     usar strlen
40
41     int capicua = 1;
42     int i=0;
43     while (capicua && i < (len / 2)){
44         if (tolower(palabra[i]) != tolower(palabra[len - i -
45             1])){
46             return 0;
47         }
48         i++;
49     }
50     return 1;
51 }

```

```
50
51 int read_word (FILE *f, char *word) {
52     return fscanf(f, " %1023s", word);
53 }
54
55 int main(int argc, char *argv[]) {
56
57     int opt= 0;
58
59     int help = -1;
60     int version = -1;
61     int input = -1;
62     int output = -1;
63
64     char *input_filename = NULL;
65     char *output_filename = NULL;
66
67     // especificacion de las opciones
68     static struct option long_options[] = {
69         {"help",      no_argument,      0, 'h' },
70         {"version",   no_argument,      0, 'V' },
71         {"input",     required_argument, 0, 'i' },
72         {"output",    required_argument, 0, 'o' },
73         {0,           0,                0,  0  }
74     };
75
76     int long_index = 0;
77
78     // evaluacion de los parametros enviados al programa
79     while ((opt = getopt_long(argc, argv, "hVui:o:",
80                             long_options, &long_index)) != -1) {
81         switch (opt) {
82             case 'h' :
83                 help = 0;
84                 break;
85             case 'V' :
86                 version = 0;
87                 break;
88             case 'i' :
89                 input = 0;
90                 input_filename = optarg;
91                 break;
92             case 'o' :
93                 output = 0;
94                 output_filename = optarg;
95                 break;
96             case '?':
97                 exit(1);
98             default:
99                 print_usage();
100                 exit(EXIT_FAILURE);
101         }
102     }
103
```

```
104 // procesamiento de los parametros
105 if (help == 0) {
106     print_help();
107     exit(0);
108 }
109 else if (version == 0) {
110     print_version();
111     exit(0);
112 }
113
114 /* Si no se recibe parametro de ayuda o version se ejecuta
    el programa */
115
116 // estableciendo los archivos de entrada y salida
117 FILE *input_file = stdin;
118 FILE *output_file = stdout;
119
120 if (input == 0){
121     input_file = fopen(input_filename, "r");
122     if (input_file == NULL) {
123         printf ("can't open input file, errno = %d\n",
124             errno);
125         return 1;
126     }
127 }
128 if (output == 0){
129     output_file = fopen(output_filename, "w");
130     if (output_file == NULL) {
131         printf ("Can't open output file, errno = %d\n",
132             errno);
133         return 1;
134     }
135 }
136
137 /* ejecucion del programa */
138 char word[1024];
139 int i = read_word(input_file, word);
140 while (i == 1){
141     if (es_capicua(word)){
142         fprintf(output_file, "%s\n", word);
143     }
144     i = read_word(input_file, word);
145 }
146
147 // cierro los archivos
148 if (input == 0){
149     fclose(input_file);
150 }
151 if (output == 0){
152     fclose(output_file);
153 }
154
```



```
155     return 0;
156 }
```

6. Código en MIPS

```
1  .file 1 "tp0.c"
2  .section .mdebug.abi32
3  .previous
4  .abicalls
5  .rdata
6  .align 2
7  $LC0:
8  .ascii  "Usage: tp0 -i [input_file] -o [output_file]\n\000"
9  .text
10 .align 2
11 .globl print_usage
12 .ent print_usage
13 print_usage:
14 .frame $fp,40,$ra      # vars= 0, regs= 3/0, args= 16, extra=
15                          8
16 .mask 0xd0000000,-8
17 .fmask 0x00000000,0
18 .set noreorder
19 .cpload $t9
20 .set reorder
21 subu $sp,$sp,40
22 .cprestore 16
23 sw $ra,32($sp)
24 sw $fp,28($sp)
25 sw $gp,24($sp)
26 move $fp,$sp
27 la $a0,$LC0
28 la $t9,printf
29 jal $ra,$t9
30 move $sp,$fp
31 lw $ra,32($sp)
32 lw $fp,28($sp)
33 addu $sp,$sp,40
34 j $ra
35 .end print_usage
36 .size print_usage, .-print_usage
37 .rdata
38 .align 2
39 $LC1:
40 .ascii  "\tUsage:\n"
41 .ascii  "\t\ttp0 -h\n"
42 .ascii  "\t\ttp0 -V\n"
43 .ascii  "\t\ttp0 [options]\n"
44 .ascii  "\tOptions:\n"
45 .ascii  "\t\t-V, --version\tPrint version and quit.\n"
46 .ascii  "\t\t-h, --help\tPrint this information.\n"
```

```
46 .ascii "\t\t-i, --input\tLocation of the input file.\n"
47 .ascii "\t\t-o, --output\tLocation of the output file.\n"
48 .ascii "\tExamples:\n"
49 .ascii "\t\ttp0 -i ~/input -o ~/output\n\000"
50 .text
51 .align 2
52 .globl print_help
53 .ent print_help
54 print_help:
55 .frame $fp,40,$ra # vars= 0, regs= 3/0, args= 16, extra=
56 8
57 .mask 0xd0000000,-8
58 .fmask 0x00000000,0
59 .set noreorder
60 .cpload $t9
61 .set reorder
62 subu $sp,$sp,40
63 .cpstore 16
64 sw $ra,32($sp)
65 sw $fp,28($sp)
66 sw $gp,24($sp)
67 move $fp,$sp
68 la $a0,$LC1
69 la $t9,printf
70 jal $ra,$t9
71 move $sp,$fp
72 lw $ra,32($sp)
73 lw $fp,28($sp)
74 addu $sp,$sp,40
75 j $ra
76 .end print_help
77 .size print_help,.-print_help
78 .rdata
79 .align 2
80 $LC2:
81 .ascii "tp0 1.0\n\000"
82 .text
83 .align 2
84 .globl print_version
85 .ent print_version
86 print_version:
87 .frame $fp,40,$ra # vars= 0, regs= 3/0, args= 16, extra=
88 8
89 .mask 0xd0000000,-8
90 .fmask 0x00000000,0
91 .set noreorder
92 .cpload $t9
93 .set reorder
94 subu $sp,$sp,40
95 .cpstore 16
96 sw $ra,32($sp)
97 sw $fp,28($sp)
98 sw $gp,24($sp)
99 move $fp,$sp
```

```
98     la    $a0,$LC2
99     la    $t9,printf
100    jal   $ra,$t9
101    move   $sp,$fp
102    lw     $ra,32($sp)
103    lw     $fp,28($sp)
104    addu   $sp,$sp,40
105    j      $ra
106    .end   print_version
107    .size  print_version, .-print_version
108    .align 2
109    .globl es_capicua
110    .ent   es_capicua
111 es_capicua:
112     .frame $fp,56,$ra      # vars= 16, regs= 3/0, args= 16, extra=
                             8
113     .mask 0xd0000000,-8
114     .fmask 0x00000000,0
115     .set   noreorder
116     .cpld  $t9
117     .set   reorder
118     subu   $sp,$sp,56
119     .cprestore 16
120     sw     $ra,48($sp)
121     sw     $fp,44($sp)
122     sw     $gp,40($sp)
123     move   $fp,$sp
124     sw     $a0,56($fp)
125     lw     $a0,56($fp)
126     la     $t9,strlen
127     jal   $ra,$t9
128     sw     $v0,24($fp)
129     li     $v0,1          # 0x1
130     sw     $v0,28($fp)
131     sw     $zero,32($fp)
132 $L21:
133     lw     $v0,28($fp)
134     beq    $v0,$zero,$L22
135     lw     $v0,24($fp)
136     srl    $v1,$v0,1
137     lw     $v0,32($fp)
138     sltu   $v0,$v0,$v1
139     bne    $v0,$zero,$L23
140     b      $L22
141 $L23:
142     lw     $v1,56($fp)
143     lw     $v0,32($fp)
144     addu   $v0,$v1,$v0
145     lb     $v0,0($v0)
146     sll    $v1,$v0,1
147     lw     $v0,_tolower_tab_
148     addu   $v0,$v1,$v0
149     addu   $a0,$v0,2
150     lw     $v1,24($fp)
```

```
151     lw    $v0,32($fp)
152     subu   $v1,$v1,$v0
153     lw    $v0,56($fp)
154     addu   $v0,$v1,$v0
155     addu   $v0,$v0,-1
156     lb    $v0,0($v0)
157     sll    $v1,$v0,1
158     lw    $v0,_tolower_tab_
159     addu   $v0,$v1,$v0
160     addu   $v0,$v0,2
161     lh    $v1,0($a0)
162     lh    $v0,0($v0)
163     beq    $v1,$v0,$L25
164     sw    $zero,36($fp)
165     b      $L20
166 $L25:
167     lw    $v0,32($fp)
168     addu   $v0,$v0,1
169     sw    $v0,32($fp)
170     b      $L21
171 $L22:
172     li    $v0,1      # 0x1
173     sw    $v0,36($fp)
174 $L20:
175     lw    $v0,36($fp)
176     move   $sp,$fp
177     lw    $ra,48($sp)
178     lw    $fp,44($sp)
179     addu   $sp,$sp,56
180     j      $ra
181     .end   es_capicua
182     .size  es_capicua,.-es_capicua
183     .rdata
184     .align 2
185 $LC3:
186     .ascii "  %1023s\000"
187     .text
188     .align 2
189     .globl read_word
190     .ent   read_word
191 read_word:
192     .frame $fp,40,$ra      # vars= 0, regs= 3/0, args= 16, extra=
193                               8
194     .mask 0xd0000000,-8
195     .fmask 0x00000000,0
196     .set   noreorder
197     .cplod $t9
198     .set   reorder
199     subu   $sp,$sp,40
200     .cprestore 16
201     sw    $ra,32($sp)
202     sw    $fp,28($sp)
203     sw    $gp,24($sp)
204     move   $fp,$sp
```

```
204    sw    $a0,40($fp)
205    sw    $a1,44($fp)
206    lw    $a0,40($fp)
207    la    $a1,$LC3
208    lw    $a2,44($fp)
209    la    $t9,fscanf
210    jal   $ra,$t9
211    move   $sp,$fp
212    lw    $ra,32($sp)
213    lw    $fp,28($sp)
214    addu   $sp,$sp,40
215    j      $ra
216    .end   read_word
217    .size  read_word,.-read_word
218    .rdata
219    .align 2
220 $LC4:
221     .ascii "help\000"
222     .align 2
223 $LC5:
224     .ascii "version\000"
225     .align 2
226 $LC6:
227     .ascii "input\000"
228     .align 2
229 $LC7:
230     .ascii "output\000"
231     .data
232     .align 2
233     .type long_options.0, @object
234     .size long_options.0, 80
235 long_options.0:
236     .word $LC4
237     .word 0
238     .word 0
239     .word 104
240     .word $LC5
241     .word 0
242     .word 0
243     .word 86
244     .word $LC6
245     .word 1
246     .word 0
247     .word 105
248     .word $LC7
249     .word 1
250     .word 0
251     .word 111
252     .word 0
253     .word 0
254     .word 0
255     .word 0
256     .rdata
257     .align 2
```

```
258 $LC8:
259     .ascii    "hVui:o:\000"
260     .align    2
261 $LC9:
262     .ascii    "archivo entrada: %s\n\000"
263     .align    2
264 $LC10:
265     .ascii    "archivo salida: %s\n\000"
266     .align    2
267 $LC11:
268     .ascii    "r\000"
269     .align    2
270 $LC12:
271     .ascii    "can't open input file, errno = %d\n\000"
272     .align    2
273 $LC13:
274     .ascii    "w\000"
275     .align    2
276 $LC14:
277     .ascii    "Can't open output file, errno = %d\n\000"
278     .align    2
279 $LC15:
280     .ascii    "%s\n\000"
281     .text
282     .align    2
283     .globl    main
284     .ent      main
285 main:
286     .frame    $fp,1128,$ra      # vars= 1080, regs= 3/0, args= 24,
                                extra= 8
287     .mask     0xd0000000,-8
288     .fmask    0x00000000,0
289     .set      noreorder
290     .cpld     $t9
291     .set      reorder
292     subu      $sp,$sp,1128
293     .cprestore 24
294     sw        $ra,1120($sp)
295     sw        $fp,1116($sp)
296     sw        $gp,1112($sp)
297     move      $fp,$sp
298     sw        $a0,1128($fp)
299     sw        $a1,1132($fp)
300     sw        $zero,32($fp)
301     li        $v0,-1           # 0xffffffffffffffff
302     sw        $v0,36($fp)
303     li        $v0,-1           # 0xffffffffffffffff
304     sw        $v0,40($fp)
305     li        $v0,-1           # 0xffffffffffffffff
306     sw        $v0,44($fp)
307     li        $v0,-1           # 0xffffffffffffffff
308     sw        $v0,48($fp)
309     sw        $zero,52($fp)
310     sw        $zero,56($fp)
```

```
311     sw    $zero,60($fp)
312 $L28:
313     addu   $v0,$fp,60
314     sw     $v0,16($sp)
315     lw     $a0,1128($fp)
316     lw     $a1,1132($fp)
317     la     $a2,$LC8
318     la     $a3,long_options.0
319     la     $t9,getopt_long
320     jal    $ra,$t9
321     sw     $v0,32($fp)
322     lw     $v1,32($fp)
323     li     $v0,-1          # 0xffffffffffffffff
324     bne    $v1,$v0,$L30
325     b      $L29
326 $L30:
327     lw     $v0,32($fp)
328     addu   $v0,$v0,-63
329     sw     $v0,1108($fp)
330     lw     $v1,1108($fp)
331     sltu   $v0,$v1,49
332     beq    $v0,$zero,$L37
333     lw     $v0,1108($fp)
334     sll    $v1,$v0,2
335     la     $v0,$L38
336     addu   $v0,$v1,$v0
337     lw     $v0,0($v0)
338     .cpadd $v0
339     j      $v0
340     .rdata
341     .align 2
342 $L38:
343     .gpword $L36
344     .gpword $L37
345     .gpword $L37
346     .gpword $L37
347     .gpword $L37
348     .gpword $L37
349     .gpword $L37
350     .gpword $L37
351     .gpword $L37
352     .gpword $L37
353     .gpword $L37
354     .gpword $L37
355     .gpword $L37
356     .gpword $L37
357     .gpword $L37
358     .gpword $L37
359     .gpword $L37
360     .gpword $L37
361     .gpword $L37
362     .gpword $L37
363     .gpword $L37
364     .gpword $L37
```

```
365 .gpword $L37
366 .gpword $L33
367 .gpword $L37
368 .gpword $L37
369 .gpword $L37
370 .gpword $L37
371 .gpword $L37
372 .gpword $L37
373 .gpword $L37
374 .gpword $L37
375 .gpword $L37
376 .gpword $L37
377 .gpword $L37
378 .gpword $L37
379 .gpword $L37
380 .gpword $L37
381 .gpword $L37
382 .gpword $L37
383 .gpword $L37
384 .gpword $L32
385 .gpword $L34
386 .gpword $L37
387 .gpword $L37
388 .gpword $L37
389 .gpword $L37
390 .gpword $L37
391 .gpword $L35
392 .text
393 $L32:
394     sw    $zero,36($fp)
395     b     $L28
396 $L33:
397     sw    $zero,40($fp)
398     b     $L28
399 $L34:
400     sw    $zero,44($fp)
401     lw    $v0,optarg
402     sw    $v0,52($fp)
403     b     $L28
404 $L35:
405     sw    $zero,48($fp)
406     lw    $v0,optarg
407     sw    $v0,56($fp)
408     b     $L28
409 $L36:
410     li    $a0,1      # 0x1
411     la    $t9,exit
412     jal   $ra,$t9
413 $L37:
414     la    $t9,print_usage
415     jal   $ra,$t9
416     li    $a0,1      # 0x1
417     la    $t9,exit
418     jal   $ra,$t9
```



```
419 $L29:
420     lw  $v0,36($fp)
421     bne $v0,$zero,$L39
422     la  $t9,print_help
423     jal $ra,$t9
424     move $a0,$zero
425     la  $t9,exit
426     jal $ra,$t9
427 $L39:
428     lw  $v0,40($fp)
429     bne $v0,$zero,$L40
430     la  $t9,print_version
431     jal $ra,$t9
432     move $a0,$zero
433     la  $t9,exit
434     jal $ra,$t9
435 $L40:
436     la  $a0,$LC9
437     lw  $a1,52($fp)
438     la  $t9,printf
439     jal $ra,$t9
440     la  $a0,$LC10
441     lw  $a1,56($fp)
442     la  $t9,printf
443     jal $ra,$t9
444     la  $v0,__$sF
445     sw  $v0,64($fp)
446     la  $v0,__$sF+88
447     sw  $v0,68($fp)
448     lw  $v0,44($fp)
449     bne $v0,$zero,$L42
450     lw  $a0,52($fp)
451     la  $a1,$LC11
452     la  $t9,fopen
453     jal $ra,$t9
454     sw  $v0,64($fp)
455     lw  $v0,64($fp)
456     bne $v0,$zero,$L42
457     la  $t9,__$errno
458     jal $ra,$t9
459     la  $a0,$LC12
460     lw  $a1,0($v0)
461     la  $t9,printf
462     jal $ra,$t9
463     li  $v0,1      # 0x1
464     sw  $v0,1104($fp)
465     b  $L27
466 $L42:
467     lw  $v0,48($fp)
468     bne $v0,$zero,$L44
469     lw  $a0,56($fp)
470     la  $a1,$LC13
471     la  $t9,fopen
472     jal $ra,$t9
```

```
473     sw    $v0,68($fp)
474     lw    $v0,68($fp)
475     bne   $v0,$zero,$L44
476     la    $t9,__errno
477     jal   $ra,$t9
478     la    $a0,$LC14
479     lw    $a1,0($v0)
480     la    $t9,printf
481     jal   $ra,$t9
482     li    $v1,1      # 0x1
483     sw    $v1,1104($fp)
484     b     $L27
485 $L44:
486     li    $v0,1      # 0x1
487     sw    $v0,72($fp)
488 $L46:
489     lw    $v1,72($fp)
490     li    $v0,1      # 0x1
491     beq   $v1,$v0,$L48
492     b     $L47
493 $L48:
494     addu   $v0,$fp,80
495     lw    $a0,64($fp)
496     move   $a1,$v0
497     la    $t9,read_word
498     jal   $ra,$t9
499     sw    $v0,72($fp)
500     addu   $v0,$fp,80
501     move   $a0,$v0
502     la    $t9,es_capicua
503     jal   $ra,$t9
504     beq   $v0,$zero,$L46
505     addu   $v0,$fp,80
506     lw    $a0,68($fp)
507     la    $a1,$LC15
508     move   $a2,$v0
509     la    $t9,fprintf
510     jal   $ra,$t9
511     b     $L46
512 $L47:
513     lw    $v0,44($fp)
514     bne   $v0,$zero,$L50
515     lw    $a0,64($fp)
516     la    $t9,fclose
517     jal   $ra,$t9
518 $L50:
519     lw    $v0,48($fp)
520     bne   $v0,$zero,$L51
521     lw    $a0,68($fp)
522     la    $t9,fclose
523     jal   $ra,$t9
524 $L51:
525     sw    $zero,1104($fp)
526 $L27:
```

```
527     lw    $v0,1104($fp)
528     move  $sp,$fp
529     lw    $ra,1120($sp)
530     lw    $fp,1116($sp)
531     addu  $sp,$sp,1128
532     j     $ra
533     .end   main
534     .size  main,.-main
535     .ident "GCC: (GNU) 3.3.3 (NetBSD nb3 20040520)"
```

7. Enunciado

El enunciado se encuentra anexado al final de este documento.