Trabajo Práctico #0: Infraestructura Básica

Fernández Cynthia, *Padrón Nro. 91.487*
cm.fernandez.28@gmail.com

Quispe Gaston, *Padrón Nro. 86.398*
gaston.quispe@gmail.com

Nombre y Apellido de Autor, *Padrón Nro. 90.596*
valeria.mrb@gmail.com

1er. Cuatrimestre de 2017
66.20 Organización de Computadoras – Práctica Miércoles
Facultad de Ingeniería, Universidad de Buenos Aires

Resumen

El objetivo de este trabajo práctico es la adquisición de práctica en la utilización de herramientas de software, necesarias para el curso de la materia, a través de la implementación de un programa simple en lenguaje C para la detección de palíndromos. Finalmente se obtendrá el código MIPS32 de dicho programa.

Índice

1. Documentación relevante

Se decidió implementar el código C dentro de un main(). No implementaremos clases ni librerías propias porque la complejidad lógica del programa no lo amerita.
Para el control de la versión utilizamos GitHub. Se trabajo con una rama 'dev' local para realizar cambios y una rama 'master' local para hacer push contra el 'master remoto'.

1.1. Algunos detalles de implementación

Con respecto a la implementación de la lógica de detección de palabras palíndromas, provista por la función int es_capicua(char* palabra), se utiliza la función char tolower(char caracter), provista por la librería ctype, para cumplir con la condición de ser case insensitive.

Las formas posibles de interactuar con el programa son:

- **./a.out -i <nombreArchivoEntrada> -o <nombreArchivoSalida>**: Leer el contenido de archivo de entrada y lo escribe en el de salida
- **./a.out -i <nombreArchivoEntrada>**: Escribir en salida stdout, (por default la consola).
- **./a.out -o <nombreArchivoSalida>**: Lee entrada stdin. Dos opciones.
  - Entrada interacitiva: La finalización del ingreso de una linea se identifica al presionar enter. La finalización de la ejecución de la escritura del archivo se identifica con ctr+d.
  - Redirección mediante pipe: El archivo de entrada se recibe a traves de un comando previo.
    Ex: echo "soy un archivo sin palindromos" | -o <nombreArchivoSalida>
- **./a.output**: Lee entrada stdin y stdout.

Los nombres de archivos de **entrada** y **salida** pueden reemplazarse con con el caracter **-**. Lo cual indica que dichos archivos son **stdin** y **stdout** respectivamente.

1.2. Asunciones

De la interpretación del enunciado y las dudas resueltas en grupo, cabe destacar lo siguiente:

- Consideramos palíndromos palabras de **un solo caracter válido**.
- Dado que los espacios no son caracteres incluidos, entendemos que se sólo se evaluarán palabras palíndromas y no frases. Ergo, el programa no está diseñado para detectar frases palíndromas.

- El programa no está diseñado para detectar repeticiones, es decir, si se ingresa dos veces la misma palabra a la entrada y ésta cumple la condición de ser palíndroma, se devolverá **dos veces** esa palabra a la salida.
- Utilizamos los caracteres inválidos, es decir, aquellos que no están contemplados en los componentes léxicos del stream de entrada, **como separadores**. De modo que si una palabra incluye un carácter inválida quedará separada en dos palabras automáticamente. Por ejemplo, si ingresamos AA@BB, el sistema tomará como que se ingresaron las palabras AA y BB.
- Cuando existe un error de lectura o escritura, se continúa con la ejecución y se muestra un mensaje de error por **stderr**. Siendo necesario abortar dicha ejecución con el comando ctrl + d.

2. Comandos para compilar

Para compilar el programa utilizamos GCC (GNU Compiler Collection) incluido en el sistema operativo NetBSD. Dicho sistema es emulado mediante el programa **gxemul** y accedido mediante un tunel ssh reverso desde un sistema linux (en nuestro caso, **Ubuntu 16.04**).

1. Movemos los archivos del proyecto a compilar desde linux al home en NetBSD:
   scp -P2222 -r /home/tporga root@127.0.0.1: ~
2. El comando que corremos en la terminal de NetBSD para compilar el código fuente en lenguaje C es:
   gcc main.c -Wall
   Si no indicamos parámetros de salida, se genera un archivo ejecutable llamado **a.out**, dentro del directorio donde se encuentra el código fuente. Para ejecutarlo utilizamos el comando **./a.out**.
3. Para obtener el código en ensamblador generado por gcc corremos la siguiente linea en el shell.
   gcc -Wall -O0 -S -mrnames main.c
   Donde:
   - Wall: Activa todos los mensajes de warning del compilador.
   - S: Detiene al compilador luego de generar el assembly.
   - mrnames (solo para MIPS): Indica al compilador que genere la salida utilizando nombre de registro en lugar de número de registro.
   - O0: No aplica optimizaciones.

   Este comando genera un archivo llamado **main.s** que contiene el código ensamblador que gcc genera para **main.c**.
4. Para traer los documentos compilados de nuevo a linux:
   scp -P2222 -r root@127.0.0.1:/home/tporga ~

3. Corridas de pruebas

Se realizan ejecutando el script de pruebas automatizadas **tests.sh** con el comando:
.\tests.sh.
Este script espera que:

- Los archivos de entrada y salida esperadas de encuentren en la carpeta **Tests**.
- El archivo ejecutable se llame **a.out** y se ecuentre en la misma carpeta que **tests.sh**.

Las pruebas leen el ejecutable a.out. Se corre el archivo tests.sh para ejecutar todas las pruebas, con el comando bash test.sh.

Las 10 pruebas realizadas son:

1. PRUEBA: Entrada vacía.
   RESULTADO: Salida vacía.

2. PRUEBA: Entrada de dos frases, con caracteres válidos, tres palíndromos.

   - ENTRADA:

   ```
   1 Somos los primeros en completar el TP 0.
   2 Ojo que La fecha de entrega del TP0 es el martes
     12 de septiembre.
   ```

   - RESULTADO:

   ```
   1 Somos
   2 0
   3 Ojo
   ```

3. PRUEBA: Entrada de tres líneas, con caracteres válidos, incluyendo guión medio.

   - ENTRADA:

   ```
   1 Ana lava lana.
   2 amor-Roma
   3 Siempre viajo a Neuquen.
   ```

   - RESULTADO:

   ```
   1 Ana
   2 amor-Roma
   3 a
   4 Neuquen
   ```

4. PRUEBA: Entrada de mil caracteres con un único palíndromo.

- ENTRADA:

```
ab abc abcd abcde abcdef abcdefg abcdefgh abcdefghi abcdefghij ab abc abcd abcde abcdef abcdefg abcdefgh abcdefghi abcdefghij ab abc abcd abcde abcdef abcdefg abcdefgh abcdefghi abcdefghij ab abc abcd abcde abcdef abcdefg abcdefgh abcdefghi abcdefghij ab abc abcd abcde abcdef abcdefg abcdefgh abcdefghi abcdefghij ab abc abcd abcde abcdef abcdefg abcdefgh abcdefghi abcdefghij ab abc abcd abcde abcdef abcdefg abcdefgh abcdefghi abcdefghij ojo abc abcd abcde abcdef abcdefg abcdefgh abcdefghi abcdefghij ab abc abcd abcde abcdef abcdefg abcdefgh abcdefghi abcdefghij ab abc abcd abcde abcdef abcdefgh abcdefghi abcdefghij ab abc abcd abcde abcdef abcdefg abcdefgh abcdefghi abcdefghij ab abc abcd abcde abcdef abcdefg abcdefgh abcdefghi abcdefghij ab abc abcd abcde abcdef abcdefg abcdefgh abcdefghi abcdefghij
```

- RESULTADO:

```
ojo
```

5. PRUEBA: Entrada de varios palíndromos con caracteres inválidos.

- ENTRADA:

```
#### 11 @@@@
a—a
?????? ojo_ojo
———
~somos~
ab33ba
```

- RESULTADO:

```
11
a—a
ojo_ojo
———
somos
ab33ba
```

6. PRUEBA: Entrada de una palabra palíndroma dentro de caracteres inválidos.

   - ENTRADA:

   ```
   1 ~$#@%&!radar[]*{+.}{
   ```

   - RESULTADO:

   ```
   1 radar
   ```

7. PRUEBA:Entrada de palíndromos con guiones y números.

   - ENTRADA:

   ```
   1 ——
   2 _ _
   3 99899
   4 99@99
   ```

   - RESULTADO:

   ```
   1 ——
   2 _ _
   3 99899
   4 99
   5 99
   ```

8. PRUEBA: Entrada de varios palíndromos entre caracteres inválidos en una sola línea.

   - ENTRADA:

   ```
   1 !!!!Neuquen,$$$
     oro,@@oso?ojo**radar&&&&&&&&&&&&&&&&&&&&reconocer,rotor#####salas
   ```

   - RESULTADO:

   ```
   1 Neuquen
   2 oro
   3 oso
   4 ojo
   5 radar
   6 reconocer
   7 rotor
   8 salas
   ```

9. PRUEBA: Entrada de palabras no palindromas entre caracteres inválidos.

   - ENTRADA:

```
!!!! hola , $$$
oros , @@osos? ojos ** radares&&&&&&&&&&&&&&&&&&&&chau , roto#####sala
```

- RESULTADO: Vacía.

10. PRUEBA: Entrada de caracteres inválidos.

- ENTRADA:

```
%%%%%%%%%#%”%”%”%”%”””””
””””””””;:[:!#%&#”!#$”
”#@@@@@@@@@@@@@@@@@@@@@@@[]#*[#
%&&&&&&&&&&&&&&&&&&&&&&
()()()())))))))))))))))))))))))))))))))))
```

- RESULTADO: Vacía.

4. Problemas e inconvenientes que se presentaron durante el desarrollo del trabajo práctico

Los principales inconvenientes surgieron de la inexperiencia en la utilización de las herramientas propuestas para el desarrollo del trabajo práctico. Las enumeramos a continuación:

- Necesidad de instalar Open SSH para generar el túnel.
- Necesidad de borrar el archivo know_host para establecer conexión a la ip.
- Decidimos no utilizar un IDE ya que la extensión del código no iba a superar las 200 líneas. Simplemente utilizamos un editor de texto. En la experiencia nos hubiese resultado útil tener un herramienta para debuguear.
- Los test observados en la sección 5.1 del enunciado no se adaptan a la definición palíndromo que encontramos y, además, es ambigua. Por lo tanto, y en base a respuestas del grupo yahoo, establecimos que el ejemplo del test donde el 0 no es palindromo **es incorrecto**.
  Los inconvenientes que surgieron a partir de la interpretación del enunciado fueron resueltas en la subsección '**Asunciones**'.

f

5. Código Fuente C

```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <ctype.h>

#define VERSION "0.1.0 beta"
typedef enum {
  STDIN_STDOUT,
  ARCHIVO_STDOUT,
  STDIN_ARCHIVO,
  ARCHIVO_ARCHIVO
} modo_entrada_salida;

int es_capicua(char* palabra)
{
  size_t izq = 0, longitud;
  longitud = strlen(palabra);
  if (longitud == 0)
    return 0;

  for (izq = 0; izq < longitud / 2; izq++) {
    int der = longitud - izq - 1;
    if (tolower((int)palabra[izq]) !=
tolower((int)palabra[der]))
      return 0;
  }
  return 1;
}

int caracter_valido(int caracter)
{
  int minuscula = tolower((int)caracter);
  if (
    (minuscula > 96 && minuscula < 123) ||
    (minuscula > 47 && minuscula < 58)  ||
      minuscula == 45 || minuscula == 95)
  {
    return 1;
  }
  return 0;
}

int leer_palabra_valida(char** str, FILE* fp) {
  char* buffer = NULL;
  size_t buffer_size = 128;
  int c;
  size_t p_comienzo = 0;
```

```
47   size_t p_longitud = 0;
48   buffer = malloc(buffer_size);
49   if (!buffer) {
50     fprintf(stderr, "Error al reservar memoria");
51     exit(1);
52   }
53   // Avanzar puntero hasta encontrar el comienzo de una
   palabra valida
54   while (!feof(fp) && !caracter_valido(c = fgetc(fp))) {
55     if (ferror(fp))
56       fprintf(stderr, "Error al leer archivo de
   entrada");
57     p_comienzo++;
58   }
59
60   // Avanzar puntero hasta que finalice la palabra valida
61   while (!feof(fp) && caracter_valido(c)) {
62     if (p_comienzo + p_longitud + 1 == buffer_size) {
63       buffer_size *= 2;
64       buffer = realloc(buffer, buffer_size);
65       if (!buffer) {
66         fprintf(stderr, "Error al reservar memoria");
67         exit(1);
68       }
69     }
70     buffer[p_longitud++] = c;
71     c = fgetc(fp);
72     if (ferror(fp))
73       fprintf(stderr, "Error al leer archivo de
   entrada");
74   }
75   buffer[p_longitud] = '\0';
76   *str = buffer;
77
78   if (feof(fp) || p_longitud == 0)
79     return 0; //No se encontro palabra valida
80   return 1; //Se encontro palabra valida
81 }
82
83 int procesar_archivo(FILE* archivo_entrada, FILE*
   archivo_salida)
84 {
85   char* palabra_valida;
86
87   if (!archivo_entrada || !archivo_salida) {
88     if (archivo_entrada) fclose(archivo_entrada);
89     if (archivo_salida)  fclose(archivo_salida);
90     return 0;
91   }
92
```

```
93   while (leer_palabra_valida(&palabra_valida,
   archivo_entrada)) {
94     if (es_capicua(palabra_valida)) {
95       fprintf(archivo_salida, "%s\n", palabra_valida);
96       if (ferror(archivo_salida))
97           fprintf(stderr, "Error al escribir en archivo
   de salida\n");
98     }
99     free(palabra_valida);
100  }
101
102  return 1;
103 }
104
105 void mostrar_usage()
106 {
107   printf("Usage:\n");
108   printf("\ttp0 -h\n");
109   printf("\ttp0 -V\n");
110   printf("\ttp0 [options]\n");
111   printf("\n");
112   printf("Options:\n");
113   printf("\t-v --version\tPrint version and quit.\n");
114   printf("\t-h --help\tPrint this information.\n");
115   printf("\t-i --input\tLocation of the input file.\n");
116   printf("\t-o --output\tLocation of the output
   file.\n");
117   printf("\n");
118   printf("Examples:\n");
119   printf("\ttp0 -i ~/input ~/output\n");
120 }
121
122 void mostrar_version()
123 {
124   printf("tp0 version: %s\n", VERSION);
125 }
126
127 int error_parametros_incorrectos()
128 {
129   fprintf(stderr, "erro fatal: Los parametros son
   incorrectos!\n");
130   mostrar_usage();
131   exit(1);
132 }
133
134 int main(int argc, char **argv)
135 {
136   size_t i;
137   int version = 0;
138   int help = 0;
```

```
139   int input = 0;
140   int output = 0;
141   char* input_path;
142   char* output_path;
143   FILE* input_handler = NULL;
144   FILE* output_handler = NULL;
145   modo_entrada_salida entrada_salida = 0;
146
147   for (i = 1; i < argc; i++) {
148     if (strcmp(argv[i], "-v") == 0 || strcmp(argv[i],
    "--version") == 0) {
149       version++;
150     } else if (strcmp(argv[i], "-h") == 0 ||
    strcmp(argv[i], "--help") == 0) {
151       help++;
152     } else if (strcmp(argv[i], "-i") == 0 ||
    strcmp(argv[i], "--input") == 0) {
153       input++;
154       if (i + 1 <= argc - 1) {
155         i++;
156         input_path = argv[i];
157       } else {
158         error_parametros_incorrectos();
159       }
160     } else if (strcmp(argv[i], "-o") == 0 ||
    strcmp(argv[i], "--output") == 0) {
161       output++;
162       if (i + 1 <= argc - 1) {
163         i++;
164         output_path = argv[i];
165       } else {
166         error_parametros_incorrectos();
167       }
168     } else {
169       error_parametros_incorrectos();
170     }
171   }
172
173   if (version > 1 || help > 1 || input > 1 || output >
    1) {
174     error_parametros_incorrectos();
175   }
176
177   if (help == 1) {
178     if (argc == 2) {
179       mostrar_usage();
180       exit(0);
181     } else {
182       error_parametros_incorrectos();
183     }
```

```
184   }
185
186   if (version == 1) {
187     if (argc == 2) {
188       mostrar_version();
189       exit(0);
190     } else {
191       error_parametros_incorrectos();
192     }
193   }
194
195   if      ((input == 0 && output == 0) ||
196     (input == 0 && output == 1 && strcmp(output_path,
    "-") == 0) ||
197     (input == 1 && output == 0 && strcmp(input_path,
    "-") == 0) ||
198     (input == 1 && output == 1 && strcmp(input_path,
    "-") == 0 &&
199                 strcmp(output_path, "-") == 0))
200       entrada_salida = STDIN_STDOUT;
201   else
202   if      ((input == 0 && output == 1 &&
    strcmp(output_path, "-") != 0) ||
203     (input == 1 && output == 1 && strcmp(input_path,
    "-") == 0 &&
204                       strcmp(output_path, "-") != 0))
205       entrada_salida = STDIN_ARCHIVO;
206   else
207   if      ((input == 1 && output == 0 &&
    strcmp(input_path,  "-") != 0) ||
208     (input == 1 && output == 1 && strcmp(input_path,
    "-") != 0 &&
209                 strcmp(output_path, "-") == 0))
210       entrada_salida = ARCHIVO_STDOUT;
211   else
212   if  (input == 1 && output == 1 &&
213     strcmp(input_path,  "-") != 0 &&
214     strcmp(output_path,  "-") != 0)
215       entrada_salida = ARCHIVO_ARCHIVO;
216   else
217     error_parametros_incorrectos();
218
219   switch (entrada_salida) {
220     case STDIN_STDOUT:
221       input_handler = stdin;
222       output_handler = stdout;
223       break;
224     case ARCHIVO_STDOUT:
225       input_handler = fopen(input_path, "r");
226       output_handler = stdout;
```

```
227         if (!input_handler) {
228           fprintf(stderr, "Error en apertura de archivo");
229           return 1;
230         }
231         break;
232       case STDIN_ARCHIVO:
233         input_handler = stdin;
234         output_handler = fopen(output_path, "w");
235         if (!output_handler) {
236           fprintf(stderr, "Error en apertura de archivo");
237           return 1;
238         }
239         break;
240       case ARCHIVO_ARCHIVO:
241         input_handler = fopen(input_path, "r");
242         output_handler = fopen(output_path, "w");
243 
244         if (!input_handler || !output_handler) {
245           if (input_handler) fclose(input_handler);
246           if (output_handler) fclose(output_handler);
247           fprintf(stderr, "Error en apertura de archivo");
248           return 1;
249         }
250         break;
251     }
252 
253     procesar_archivo(input_handler, output_handler);
254 
255     switch (entrada_salida) {
256       case STDIN_STDOUT:
257         break;
258       case ARCHIVO_STDOUT:
259         fclose(input_handler);
260         break;
261       case STDIN_ARCHIVO:
262         fclose(output_handler);
263         break;
264       case ARCHIVO_ARCHIVO:
265         fclose(input_handler);
266         fclose(output_handler);
267         break;
268     }
269 
270     return 0;
271 }
```

6. Código MIPS32

```
1   .file 1 "main.c"
2   .section .mdebug.abi32
3   .previous
4   .abicalls
5   .text
6   .align 2
7   .globl es_capicua
8   .ent es_capicua
9 es_capicua:
10  .frame $fp,56,$ra    # vars= 16, regs= 3/0, args= 16,
  extra= 8
11  .mask 0xd0000000,-8
12  .fmask 0x00000000,0
13  .set noreorder
14  .cpload $t9
15  .set reorder
16  subu $sp,$sp,56
17  .cprestore 16
18  sw $ra,48($sp)
19  sw $fp,44($sp)
20  sw $gp,40($sp)
21  move $fp,$sp
22  sw $a0,56($fp)
23  sw $zero,24($fp)
24  lw $a0,56($fp)
25  la $t9,strlen
26  jal $ra,$t9
27  sw $v0,28($fp)
28  lw $v0,28($fp)
29  bne $v0,$zero,$L18
30  sw $zero,36($fp)
31  b $L17
32 $L18:
33  sw $zero,24($fp)
34 $L19:
35  lw $v0,28($fp)
36  srl $v1,$v0,1
37  lw $v0,24($fp)
38  sltu $v0,$v0,$v1
39  bne $v0,$zero,$L22
40  b $L20
41 $L22:
42  lw $v1,28($fp)
43  lw $v0,24($fp)
44  subu $v0,$v1,$v0
45  addu $v0,$v0,-1
46  sw $v0,32($fp)
```

```
47    lw   $v1,56($fp)
48    lw   $v0,24($fp)
49    addu   $v0,$v1,$v0
50    lb   $v0,0($v0)
51    sll $v1,$v0,1
52    lw   $v0,_tolower_tab_
53    addu   $v0,$v1,$v0
54    addu   $a0,$v0,2
55    lw   $v1,56($fp)
56    lw   $v0,32($fp)
57    addu   $v0,$v1,$v0
58    lb   $v0,0($v0)
59    sll $v1,$v0,1
60    lw   $v0,_tolower_tab_
61    addu   $v0,$v1,$v0
62    addu   $v0,$v0,2
63    lh   $v1,0($a0)
64    lh   $v0,0($v0)
65    beq $v1,$v0,$L21
66    sw   $zero,36($fp)
67    b $L17
68 $L21:
69    lw   $v0,24($fp)
70    addu   $v0,$v0,1
71    sw   $v0,24($fp)
72    b $L19
73 $L20:
74    li   $v0,1      # 0x1
75    sw   $v0,36($fp)
76 $L17:
77    lw   $v0,36($fp)
78    move   $sp,$fp
79    lw   $ra,48($sp)
80    lw   $fp,44($sp)
81    addu   $sp,$sp,56
82    j $ra
83    .end   es_capicua
84    .size es_capicua, .-es_capicua
85    .align   2
86    .globl   caracter_valido
87    .ent   caracter_valido
88 caracter_valido:
89    .frame   $fp,24,$ra     # vars= 8, regs= 2/0, args= 0,
   extra= 8
90    .mask 0x50000000,-4
91    .fmask   0x00000000,0
92    .set   noreorder
93    .cpload $t9
94    .set   reorder
95    subu   $sp,$sp,24
```

```
 96    .cprestore 0
 97    sw   $fp,20($sp)
 98    sw   $gp,16($sp)
 99    move   $fp,$sp
100    sw   $a0,24($fp)
101    lw   $v0,24($fp)
102    sll $v1,$v0,1
103    lw   $v0,_tolower_tab_
104    addu   $v0,$v1,$v0
105    addu   $v0,$v0,2
106    lh   $v0,0($v0)
107    sw   $v0,8($fp)
108    lw   $v0,8($fp)
109    slt $v0,$v0,97
110    bne $v0,$zero,$L27
111    lw   $v0,8($fp)
112    slt $v0,$v0,123
113    bne $v0,$zero,$L26
114 $L27:
115    lw   $v0,8($fp)
116    slt $v0,$v0,48
117    bne $v0,$zero,$L28
118    lw   $v0,8($fp)
119    slt $v0,$v0,58
120    bne $v0,$zero,$L26
121 $L28:
122    lw   $v1,8($fp)
123    li   $v0,45       # 0x2d
124    beq $v1,$v0,$L26
125    lw   $v1,8($fp)
126    li   $v0,95       # 0x5f
127    beq $v1,$v0,$L26
128    b $L25
129 $L26:
130    li   $v0,1      # 0x1
131    sw   $v0,12($fp)
132    b $L24
133 $L25:
134    sw   $zero,12($fp)
135 $L24:
136    lw   $v0,12($fp)
137    move   $sp,$fp
138    lw   $fp,20($sp)
139    addu   $sp,$sp,24
140    j $ra
141    .end   caracter_valido
142    .size caracter_valido, .-caracter_valido
143    .rdata
144    .align   2
145 $LC0:
```

```
146   .ascii  "Error al reservar memoria\000"
147   .align  2
148 $LC1:
149   .ascii  "Error al leer archivo de entrada\000"
150   .text
151   .align  2
152   .globl  leer_palabra_valida
153   .ent  leer_palabra_valida
154 leer_palabra_valida:
155   .frame  $fp,64,$ra    # vars= 24, regs= 3/0, args= 16,
    extra= 8
156   .mask 0xd0000000,-8
157   .fmask  0x00000000,0
158   .set  noreorder
159   .cpload $t9
160   .set  reorder
161   subu  $sp,$sp,64
162   .cprestore 16
163   sw  $ra,56($sp)
164   sw  $fp,52($sp)
165   sw  $gp,48($sp)
166   move  $fp,$sp
167   sw  $a0,64($fp)
168   sw  $a1,68($fp)
169   sw  $zero,24($fp)
170   li  $v0,128     # 0x80
171   sw  $v0,28($fp)
172   sw  $zero,36($fp)
173   sw  $zero,40($fp)
174   lw  $a0,28($fp)
175   la  $t9,malloc
176   jal $ra,$t9
177   sw  $v0,24($fp)
178   lw  $v0,24($fp)
179   bne $v0,$zero,$L30
180   la  $a0,__sF+176
181   la  $a1,$LC0
182   la  $t9,fprintf
183   jal $ra,$t9
184   li  $a0,1     # 0x1
185   la  $t9,exit
186   jal $ra,$t9
187 $L30:
188   .set  noreorder
189   nop
190   .set  reorder
191 $L31:
192   lw  $v0,68($fp)
193   lhu $v0,12($v0)
194   srl $v0,$v0,5
```

```
195   andi  $v0,$v0,0x1
196   bne $v0,$zero,$L32
197   lw  $a0,68($fp)
198   la  $t9,fgetc
199   jal $ra,$t9
200   sw  $v0,32($fp)
201   lw  $a0,32($fp)
202   la  $t9,caracter_valido
203   jal $ra,$t9
204   bne $v0,$zero,$L32
205   lw  $v0,68($fp)
206   lhu $v0,12($v0)
207   srl $v0,$v0,6
208   andi  $v0,$v0,0x1
209   beq $v0,$zero,$L35
210   la  $a0,__sF+176
211   la  $a1,$LC1
212   la  $t9,fprintf
213   jal $ra,$t9
214 $L35:
215   lw  $v0,36($fp)
216   addu  $v0,$v0,1
217   sw  $v0,36($fp)
218   b $L31
219 $L32:
220   .set  noreorder
221   nop
222   .set  reorder
223 $L36:
224   lw  $v0,68($fp)
225   lhu $v0,12($v0)
226   srl $v0,$v0,5
227   andi  $v0,$v0,0x1
228   bne $v0,$zero,$L37
229   lw  $a0,32($fp)
230   la  $t9,caracter_valido
231   jal $ra,$t9
232   bne $v0,$zero,$L38
233   b $L37
234 $L38:
235   lw  $v1,36($fp)
236   lw  $v0,40($fp)
237   addu  $v0,$v1,$v0
238   addu  $v1,$v0,1
239   lw  $v0,28($fp)
240   bne $v1,$v0,$L40
241   lw  $v0,28($fp)
242   sll $v0,$v0,1
243   sw  $v0,28($fp)
244   lw  $a0,24($fp)
```

```
245   lw   $a1,28($fp)
246   la   $t9,realloc
247   jal  $ra,$t9
248   sw   $v0,24($fp)
249   lw   $v0,24($fp)
250   bne  $v0,$zero,$L40
251   la   $a0,__sF+176
252   la   $a1,$LC0
253   la   $t9,fprintf
254   jal  $ra,$t9
255   li   $a0,1      # 0x1
256   la   $t9,exit
257   jal  $ra,$t9
258 $L40:
259   addu   $a1,$fp,40
260   lw   $v1,0($a1)
261   move   $a0,$v1
262   lw   $v0,24($fp)
263   addu   $a0,$a0,$v0
264   lbu  $v0,32($fp)
265   sb   $v0,0($a0)
266   addu   $v1,$v1,1
267   sw   $v1,0($a1)
268   lw   $a0,68($fp)
269   la   $t9,fgetc
270   jal  $ra,$t9
271   sw   $v0,32($fp)
272   lw   $v0,68($fp)
273   lhu  $v0,12($v0)
274   srl  $v0,$v0,6
275   andi   $v0,$v0,0x1
276   beq  $v0,$zero,$L36
277   la   $a0,__sF+176
278   la   $a1,$LC1
279   la   $t9,fprintf
280   jal  $ra,$t9
281   b $L36
282 $L37:
283   lw   $v1,24($fp)
284   lw   $v0,40($fp)
285   addu   $v0,$v1,$v0
286   sb   $zero,0($v0)
287   lw   $v1,64($fp)
288   lw   $v0,24($fp)
289   sw   $v0,0($v1)
290   lw   $v0,68($fp)
291   lhu  $v0,12($v0)
292   srl  $v0,$v0,5
293   andi   $v0,$v0,0x1
294   bne  $v0,$zero,$L44
```

```
295   lw   $v0,40($fp)
296   bne $v0,$zero,$L43
297 $L44:
298   sw   $zero,44($fp)
299   b $L29
300 $L43:
301   li   $v0,1      # 0x1
302   sw   $v0,44($fp)
303 $L29:
304   lw   $v0,44($fp)
305   move   $sp,$fp
306   lw   $ra,56($sp)
307   lw   $fp,52($sp)
308   addu   $sp,$sp,64
309   j $ra
310   .end   leer_palabra_valida
311   .size leer_palabra_valida, .-leer_palabra_valida
312   .rdata
313   .align   2
314 $LC2:
315   .ascii   "%s\n\000"
316   .align   2
317 $LC3:
318   .ascii   "Error al escribir en archivo de salida\n\000"
319   .text
320   .align   2
321   .globl   procesar_archivo
322   .ent   procesar_archivo
323 procesar_archivo:
324   .frame   $fp,48,$ra     # vars= 8, regs= 3/0, args= 16,
    extra= 8
325   .mask 0xd0000000,-8
326   .fmask   0x00000000,0
327   .set   noreorder
328   .cpload $t9
329   .set   reorder
330   subu   $sp,$sp,48
331   .cprestore 16
332   sw   $ra,40($sp)
333   sw   $fp,36($sp)
334   sw   $gp,32($sp)
335   move   $fp,$sp
336   sw   $a0,48($fp)
337   sw   $a1,52($fp)
338   lw   $v0,48($fp)
339   beq $v0,$zero,$L47
340   lw   $v0,52($fp)
341   bne $v0,$zero,$L46
342 $L47:
343   lw   $v0,48($fp)
```

```
344   beq $v0,$zero,$L48
345   lw  $a0,48($fp)
346   la  $t9,fclose
347   jal $ra,$t9
348 $L48:
349   lw  $v0,52($fp)
350   beq $v0,$zero,$L49
351   lw  $a0,52($fp)
352   la  $t9,fclose
353   jal $ra,$t9
354 $L49:
355   sw  $zero,28($fp)
356   b $L45
357 $L46:
358   .set  noreorder
359   nop
360   .set  reorder
361 $L50:
362   addu  $a0,$fp,24
363   lw  $a1,48($fp)
364   la  $t9,leer_palabra_valida
365   jal $ra,$t9
366   bne $v0,$zero,$L52
367   b $L51
368 $L52:
369   lw  $a0,24($fp)
370   la  $t9,es_capicua
371   jal $ra,$t9
372   beq $v0,$zero,$L53
373   lw  $a0,52($fp)
374   la  $a1,$LC2
375   lw  $a2,24($fp)
376   la  $t9,fprintf
377   jal $ra,$t9
378   lw  $v0,52($fp)
379   lhu $v0,12($v0)
380   srl $v0,$v0,6
381   andi  $v0,$v0,0x1
382   beq $v0,$zero,$L53
383   la  $a0,__sF+176
384   la  $a1,$LC3
385   la  $t9,fprintf
386   jal $ra,$t9
387 $L53:
388   lw  $a0,24($fp)
389   la  $t9,free
390   jal $ra,$t9
391   b $L50
392 $L51:
393   li  $v0,1       # 0x1
```

```
394   sw   $v0,28($fp)
395 $L45:
396   lw   $v0,28($fp)
397   move   $sp,$fp
398   lw   $ra,40($sp)
399   lw   $fp,36($sp)
400   addu   $sp,$sp,48
401   j $ra
402   .end   procesar_archivo
403   .size procesar_archivo, .-procesar_archivo
404   .rdata
405   .align   2
406 $LC4:
407   .ascii   "Usage:\n\000"
408   .align   2
409 $LC5:
410   .ascii   "\ttp0 -h\n\000"
411   .align   2
412 $LC6:
413   .ascii   "\ttp0 -V\n\000"
414   .align   2
415 $LC7:
416   .ascii   "\ttp0 [options]\n\000"
417   .align   2
418 $LC8:
419   .ascii   "\n\000"
420   .align   2
421 $LC9:
422   .ascii   "Options:\n\000"
423   .align   2
424 $LC10:
425   .ascii   "\t-v --version\tPrint version and quit.\n\000"
426   .align   2
427 $LC11:
428   .ascii   "\t-h --help\tPrint this information.\n\000"
429   .align   2
430 $LC12:
431   .ascii   "\t-i --input\tLocation of the input
    file.\n\000"
432   .align   2
433 $LC13:
434   .ascii   "\t-o --output\tLocation of the output
    file.\n\000"
435   .align   2
436 $LC14:
437   .ascii   "Examples:\n\000"
438   .align   2
439 $LC15:
440   .ascii   "\ttp0 -i ~/input ~/output\n\000"
441   .text
```

```
442   .align  2
443   .globl  mostrar_usage
444   .ent  mostrar_usage
445 mostrar_usage:
446   .frame  $fp,40,$ra     # vars= 0, regs= 3/0, args= 16,
    extra= 8
447   .mask 0xd0000000,-8
448   .fmask  0x00000000,0
449   .set  noreorder
450   .cpload $t9
451   .set  reorder
452   subu  $sp,$sp,40
453   .cprestore 16
454   sw  $ra,32($sp)
455   sw  $fp,28($sp)
456   sw  $gp,24($sp)
457   move  $fp,$sp
458   la  $a0,$LC4
459   la  $t9,printf
460   jal $ra,$t9
461   la  $a0,$LC5
462   la  $t9,printf
463   jal $ra,$t9
464   la  $a0,$LC6
465   la  $t9,printf
466   jal $ra,$t9
467   la  $a0,$LC7
468   la  $t9,printf
469   jal $ra,$t9
470   la  $a0,$LC8
471   la  $t9,printf
472   jal $ra,$t9
473   la  $a0,$LC9
474   la  $t9,printf
475   jal $ra,$t9
476   la  $a0,$LC10
477   la  $t9,printf
478   jal $ra,$t9
479   la  $a0,$LC11
480   la  $t9,printf
481   jal $ra,$t9
482   la  $a0,$LC12
483   la  $t9,printf
484   jal $ra,$t9
485   la  $a0,$LC13
486   la  $t9,printf
487   jal $ra,$t9
488   la  $a0,$LC8
489   la  $t9,printf
490   jal $ra,$t9
```

```
491   la   $a0,$LC14
492   la   $t9,printf
493   jal  $ra,$t9
494   la   $a0,$LC15
495   la   $t9,printf
496   jal  $ra,$t9
497   move  $sp,$fp
498   lw   $ra,32($sp)
499   lw   $fp,28($sp)
500   addu  $sp,$sp,40
501   j $ra
502   .end  mostrar_usage
503   .size mostrar_usage, .-mostrar_usage
504   .rdata
505   .align  2
506 $LC16:
507   .ascii  "tp0 version: %s\n\000"
508   .align  2
509 $LC17:
510   .ascii  "0.1.0beta\000"
511   .text
512   .align  2
513   .globl  mostrar_version
514   .ent  mostrar_version
515 mostrar_version:
516   .frame  $fp,40,$ra     # vars= 0, regs= 3/0, args= 16,
    extra= 8
517   .mask 0xd0000000,-8
518   .fmask  0x00000000,0
519   .set  noreorder
520   .cpload $t9
521   .set  reorder
522   subu  $sp,$sp,40
523   .cprestore 16
524   sw   $ra,32($sp)
525   sw   $fp,28($sp)
526   sw   $gp,24($sp)
527   move  $fp,$sp
528   la   $a0,$LC16
529   la   $a1,$LC17
530   la   $t9,printf
531   jal  $ra,$t9
532   move  $sp,$fp
533   lw   $ra,32($sp)
534   lw   $fp,28($sp)
535   addu  $sp,$sp,40
536   j $ra
537   .end  mostrar_version
538   .size mostrar_version, .-mostrar_version
539   .rdata
```

```
540   .align  2
541 $LC18:
542   .ascii  "erro fatal: Los parametros son
    incorrectos!\n\000"
543   .text
544   .align  2
545   .globl  error_parametros_incorrectos
546   .ent  error_parametros_incorrectos
547 error_parametros_incorrectos:
548   .frame  $fp,40,$ra     # vars= 0, regs= 3/0, args= 16,
    extra= 8
549   .mask 0xd0000000,-8
550   .fmask  0x00000000,0
551   .set  noreorder
552   .cpload $t9
553   .set  reorder
554   subu  $sp,$sp,40
555   .cprestore 16
556   sw  $ra,32($sp)
557   sw  $fp,28($sp)
558   sw  $gp,24($sp)
559   move  $fp,$sp
560   la  $a0,__sF+176
561   la  $a1,$LC18
562   la  $t9,fprintf
563   jal $ra,$t9
564   la  $t9,mostrar_usage
565   jal $ra,$t9
566   li  $a0,1     # 0x1
567   la  $t9,exit
568   jal $ra,$t9
569   .end  error_parametros_incorrectos
570   .size error_parametros_incorrectos,
    .-error_parametros_incorrectos
571   .rdata
572   .align  2
573 $LC19:
574   .ascii  "-v\000"
575   .align  2
576 $LC20:
577   .ascii  "--version\000"
578   .align  2
579 $LC21:
580   .ascii  "-h\000"
581   .align  2
582 $LC22:
583   .ascii  "--help\000"
584   .align  2
585 $LC23:
586   .ascii  "-i\000"
```

```
587   .align  2
588 $LC24:
589   .ascii  "--input\000"
590   .align  2
591 $LC25:
592   .ascii  "-o\000"
593   .align  2
594 $LC26:
595   .ascii  "--output\000"
596   .align  2
597 $LC27:
598   .ascii  "-\000"
599   .align  2
600 $LC28:
601   .ascii  "r\000"
602   .align  2
603 $LC29:
604   .ascii  "Error en apertura de archivo\000"
605   .align  2
606 $LC30:
607   .ascii  "w\000"
608   .text
609   .align  2
610   .globl  main
611   .ent  main
612 main:
613   .frame  $fp,96,$ra    # vars= 56, regs= 3/0, args= 16,
    extra= 8
614   .mask 0xd0000000,-8
615   .fmask  0x00000000,0
616   .set  noreorder
617   .cpload $t9
618   .set  reorder
619   subu  $sp,$sp,96
620   .cprestore 16
621   sw  $ra,88($sp)
622   sw  $fp,84($sp)
623   sw  $gp,80($sp)
624   move  $fp,$sp
625   sw  $a0,96($fp)
626   sw  $a1,100($fp)
627   sw  $zero,28($fp)
628   sw  $zero,32($fp)
629   sw  $zero,36($fp)
630   sw  $zero,40($fp)
631   sw  $zero,52($fp)
632   sw  $zero,56($fp)
633   sw  $zero,60($fp)
634   li  $v0,1     # 0x1
635   sw  $v0,24($fp)
```

```
636 $L59:
637     lw   $v0,24($fp)
638     lw   $v1,96($fp)
639     sltu    $v0,$v0,$v1
640     bne $v0,$zero,$L62
641     b $L60
642 $L62:
643     lw   $v0,24($fp)
644     sll $v1,$v0,2
645     lw   $v0,100($fp)
646     addu    $v0,$v1,$v0
647     lw   $a0,0($v0)
648     la   $a1,$LC19
649     la   $t9,strcmp
650     jal $ra,$t9
651     beq $v0,$zero,$L64
652     lw   $v0,24($fp)
653     sll $v1,$v0,2
654     lw   $v0,100($fp)
655     addu    $v0,$v1,$v0
656     lw   $a0,0($v0)
657     la   $a1,$LC20
658     la   $t9,strcmp
659     jal $ra,$t9
660     bne $v0,$zero,$L63
661 $L64:
662     lw   $v0,28($fp)
663     addu    $v0,$v0,1
664     sw   $v0,28($fp)
665     b $L61
666 $L63:
667     lw   $v0,24($fp)
668     sll $v1,$v0,2
669     lw   $v0,100($fp)
670     addu    $v0,$v1,$v0
671     lw   $a0,0($v0)
672     la   $a1,$LC21
673     la   $t9,strcmp
674     jal $ra,$t9
675     beq $v0,$zero,$L67
676     lw   $v0,24($fp)
677     sll $v1,$v0,2
678     lw   $v0,100($fp)
679     addu    $v0,$v1,$v0
680     lw   $a0,0($v0)
681     la   $a1,$LC22
682     la   $t9,strcmp
683     jal $ra,$t9
684     bne $v0,$zero,$L66
685 $L67:
```

```
686   lw   $v0,32($fp)
687   addu   $v0,$v0,1
688   sw   $v0,32($fp)
689   b $L61
690 $L66:
691   lw   $v0,24($fp)
692   sll $v1,$v0,2
693   lw   $v0,100($fp)
694   addu   $v0,$v1,$v0
695   lw   $a0,0($v0)
696   la   $a1,$LC23
697   la   $t9,strcmp
698   jal $ra,$t9
699   beq $v0,$zero,$L70
700   lw   $v0,24($fp)
701   sll $v1,$v0,2
702   lw   $v0,100($fp)
703   addu   $v0,$v1,$v0
704   lw   $a0,0($v0)
705   la   $a1,$LC24
706   la   $t9,strcmp
707   jal $ra,$t9
708   bne $v0,$zero,$L69
709 $L70:
710   lw   $v0,36($fp)
711   addu   $v0,$v0,1
712   sw   $v0,36($fp)
713   lw   $v0,24($fp)
714   addu   $v1,$v0,1
715   lw   $v0,96($fp)
716   addu   $v0,$v0,-1
717   sltu   $v0,$v0,$v1
718   bne $v0,$zero,$L71
719   lw   $v0,24($fp)
720   addu   $v0,$v0,1
721   sw   $v0,24($fp)
722   lw   $v0,24($fp)
723   sll $v1,$v0,2
724   lw   $v0,100($fp)
725   addu   $v0,$v1,$v0
726   lw   $v0,0($v0)
727   sw   $v0,44($fp)
728   b $L61
729 $L71:
730   la   $t9,error_parametros_incorrectos
731   jal $ra,$t9
732   b $L61
733 $L69:
734   lw   $v0,24($fp)
735   sll $v1,$v0,2
```

```
736   lw   $v0,100($fp)
737   addu   $v0,$v1,$v0
738   lw   $a0,0($v0)
739   la   $a1,$LC25
740   la   $t9,strcmp
741   jal  $ra,$t9
742   beq  $v0,$zero,$L75
743   lw   $v0,24($fp)
744   sll  $v1,$v0,2
745   lw   $v0,100($fp)
746   addu   $v0,$v1,$v0
747   lw   $a0,0($v0)
748   la   $a1,$LC26
749   la   $t9,strcmp
750   jal  $ra,$t9
751   bne  $v0,$zero,$L74
752 $L75:
753   lw   $v0,40($fp)
754   addu   $v0,$v0,1
755   sw   $v0,40($fp)
756   lw   $v0,24($fp)
757   addu   $v1,$v0,1
758   lw   $v0,96($fp)
759   addu   $v0,$v0,-1
760   sltu   $v0,$v0,$v1
761   bne  $v0,$zero,$L76
762   lw   $v0,24($fp)
763   addu   $v0,$v0,1
764   sw   $v0,24($fp)
765   lw   $v0,24($fp)
766   sll  $v1,$v0,2
767   lw   $v0,100($fp)
768   addu   $v0,$v1,$v0
769   lw   $v0,0($v0)
770   sw   $v0,48($fp)
771   b  $L61
772 $L76:
773   la   $t9,error_parametros_incorrectos
774   jal  $ra,$t9
775   b  $L61
776 $L74:
777   la   $t9,error_parametros_incorrectos
778   jal  $ra,$t9
779 $L61:
780   lw   $v0,24($fp)
781   addu   $v0,$v0,1
782   sw   $v0,24($fp)
783   b  $L59
784 $L60:
785   lw   $v0,28($fp)
```

```
786     slt $v0,$v0,2
787     beq $v0,$zero,$L80
788     lw  $v0,32($fp)
789     slt $v0,$v0,2
790     beq $v0,$zero,$L80
791     lw  $v0,36($fp)
792     slt $v0,$v0,2
793     beq $v0,$zero,$L80
794     lw  $v0,40($fp)
795     slt $v0,$v0,2
796     beq $v0,$zero,$L80
797     b $L79
798 $L80:
799     la  $t9,error_parametros_incorrectos
800     jal $ra,$t9
801 $L79:
802     lw  $v1,32($fp)
803     li  $v0,1       # 0x1
804     bne $v1,$v0,$L81
805     lw  $v1,96($fp)
806     li  $v0,2       # 0x2
807     bne $v1,$v0,$L82
808     la  $t9,mostrar_usage
809     jal $ra,$t9
810     move  $a0,$zero
811     la  $t9,exit
812     jal $ra,$t9
813 $L82:
814     la  $t9,error_parametros_incorrectos
815     jal $ra,$t9
816 $L81:
817     lw  $v1,28($fp)
818     li  $v0,1       # 0x1
819     bne $v1,$v0,$L84
820     lw  $v1,96($fp)
821     li  $v0,2       # 0x2
822     bne $v1,$v0,$L85
823     la  $t9,mostrar_version
824     jal $ra,$t9
825     move  $a0,$zero
826     la  $t9,exit
827     jal $ra,$t9
828 $L85:
829     la  $t9,error_parametros_incorrectos
830     jal $ra,$t9
831 $L84:
832     lw  $v0,36($fp)
833     bne $v0,$zero,$L89
834     lw  $v0,40($fp)
835     bne $v0,$zero,$L89
```

```
836   b $L88
837 $L89:
838   lw  $v0,36($fp)
839   bne $v0,$zero,$L90
840   lw  $v1,40($fp)
841   li  $v0,1       # 0x1
842   bne $v1,$v0,$L90
843   lw  $a0,48($fp)
844   la  $a1,$LC27
845   la  $t9,strcmp
846   jal $ra,$t9
847   bne $v0,$zero,$L90
848   b $L88
849 $L90:
850   lw  $v1,36($fp)
851   li  $v0,1       # 0x1
852   bne $v1,$v0,$L91
853   lw  $v0,40($fp)
854   bne $v0,$zero,$L91
855   lw  $a0,44($fp)
856   la  $a1,$LC27
857   la  $t9,strcmp
858   jal $ra,$t9
859   bne $v0,$zero,$L91
860   b $L88
861 $L91:
862   lw  $v1,36($fp)
863   li  $v0,1       # 0x1
864   bne $v1,$v0,$L87
865   lw  $v1,40($fp)
866   li  $v0,1       # 0x1
867   bne $v1,$v0,$L87
868   lw  $a0,44($fp)
869   la  $a1,$LC27
870   la  $t9,strcmp
871   jal $ra,$t9
872   bne $v0,$zero,$L87
873   lw  $a0,48($fp)
874   la  $a1,$LC27
875   la  $t9,strcmp
876   jal $ra,$t9
877   bne $v0,$zero,$L87
878 $L88:
879   sw  $zero,60($fp)
880   b $L92
881 $L87:
882   lw  $v0,36($fp)
883   bne $v0,$zero,$L95
884   lw  $v1,40($fp)
885   li  $v0,1       # 0x1
```

```
886     bne $v1,$v0,$L95
887     lw  $a0,48($fp)
888     la  $a1,$LC27
889     la  $t9,strcmp
890     jal $ra,$t9
891     bne $v0,$zero,$L94
892 $L95:
893     lw  $v1,36($fp)
894     li  $v0,1       # 0x1
895     bne $v1,$v0,$L93
896     lw  $v1,40($fp)
897     li  $v0,1       # 0x1
898     bne $v1,$v0,$L93
899     lw  $a0,44($fp)
900     la  $a1,$LC27
901     la  $t9,strcmp
902     jal $ra,$t9
903     bne $v0,$zero,$L93
904     lw  $a0,48($fp)
905     la  $a1,$LC27
906     la  $t9,strcmp
907     jal $ra,$t9
908     bne $v0,$zero,$L94
909     b $L93
910 $L94:
911     li  $v0,2       # 0x2
912     sw  $v0,60($fp)
913     b $L92
914 $L93:
915     lw  $v1,36($fp)
916     li  $v0,1       # 0x1
917     bne $v1,$v0,$L99
918     lw  $v0,40($fp)
919     bne $v0,$zero,$L99
920     lw  $a0,44($fp)
921     la  $a1,$LC27
922     la  $t9,strcmp
923     jal $ra,$t9
924     bne $v0,$zero,$L98
925 $L99:
926     lw  $v1,36($fp)
927     li  $v0,1       # 0x1
928     bne $v1,$v0,$L97
929     lw  $v1,40($fp)
930     li  $v0,1       # 0x1
931     bne $v1,$v0,$L97
932     lw  $a0,44($fp)
933     la  $a1,$LC27
934     la  $t9,strcmp
935     jal $ra,$t9
```

```
936   beq $v0,$zero,$L97
937   lw  $a0,48($fp)
938   la  $a1,$LC27
939   la  $t9,strcmp
940   jal $ra,$t9
941   bne $v0,$zero,$L97
942 $L98:
943   li  $v0,1       # 0x1
944   sw  $v0,60($fp)
945   b $L92
946 $L97:
947   lw  $v1,36($fp)
948   li  $v0,1       # 0x1
949   bne $v1,$v0,$L101
950   lw  $v1,40($fp)
951   li  $v0,1       # 0x1
952   bne $v1,$v0,$L101
953   lw  $a0,44($fp)
954   la  $a1,$LC27
955   la  $t9,strcmp
956   jal $ra,$t9
957   beq $v0,$zero,$L101
958   lw  $a0,48($fp)
959   la  $a1,$LC27
960   la  $t9,strcmp
961   jal $ra,$t9
962   beq $v0,$zero,$L101
963   li  $v0,3       # 0x3
964   sw  $v0,60($fp)
965   b $L92
966 $L101:
967   la  $t9,error_parametros_incorrectos
968   jal $ra,$t9
969 $L92:
970   lw  $v0,60($fp)
971   sw  $v0,68($fp)
972   li  $v0,1       # 0x1
973   lw  $v1,68($fp)
974   beq $v1,$v0,$L105
975   lw  $v1,68($fp)
976   sltu  $v0,$v1,1
977   bne $v0,$zero,$L104
978   li  $v0,2       # 0x2
979   lw  $v1,68($fp)
980   beq $v1,$v0,$L107
981   li  $v0,3       # 0x3
982   lw  $v1,68($fp)
983   beq $v1,$v0,$L109
984   b $L103
985 $L104:
```

```
 986    la   $v0,__sF
 987    sw   $v0,52($fp)
 988    la   $v0,__sF+88
 989    sw   $v0,56($fp)
 990    b $L103
 991 $L105:
 992    lw   $a0,44($fp)
 993    la   $a1,$LC28
 994    la   $t9,fopen
 995    jal $ra,$t9
 996    sw   $v0,52($fp)
 997    la   $v0,__sF+88
 998    sw   $v0,56($fp)
 999    lw   $v0,52($fp)
1000    bne $v0,$zero,$L103
1001    la   $a0,__sF+176
1002    la   $a1,$LC29
1003    la   $t9,fprintf
1004    jal $ra,$t9
1005    li   $v0,1      # 0x1
1006    sw   $v0,64($fp)
1007    b $L58
1008 $L107:
1009    la   $v0,__sF
1010    sw   $v0,52($fp)
1011    lw   $a0,48($fp)
1012    la   $a1,$LC30
1013    la   $t9,fopen
1014    jal $ra,$t9
1015    sw   $v0,56($fp)
1016    lw   $v0,56($fp)
1017    bne $v0,$zero,$L103
1018    la   $a0,__sF+176
1019    la   $a1,$LC29
1020    la   $t9,fprintf
1021    jal $ra,$t9
1022    li   $v1,1      # 0x1
1023    sw   $v1,64($fp)
1024    b $L58
1025 $L109:
1026    lw   $a0,44($fp)
1027    la   $a1,$LC28
1028    la   $t9,fopen
1029    jal $ra,$t9
1030    sw   $v0,52($fp)
1031    lw   $a0,48($fp)
1032    la   $a1,$LC30
1033    la   $t9,fopen
1034    jal $ra,$t9
1035    sw   $v0,56($fp)
```

```
1036    lw   $v0,52($fp)
1037    beq  $v0,$zero,$L111
1038    lw   $v0,56($fp)
1039    bne  $v0,$zero,$L103
1040 $L111:
1041    lw   $v0,52($fp)
1042    beq  $v0,$zero,$L112
1043    lw   $a0,52($fp)
1044    la   $t9,fclose
1045    jal  $ra,$t9
1046 $L112:
1047    lw   $v0,56($fp)
1048    beq  $v0,$zero,$L113
1049    lw   $a0,56($fp)
1050    la   $t9,fclose
1051    jal  $ra,$t9
1052 $L113:
1053    la   $a0,__sF+176
1054    la   $a1,$LC29
1055    la   $t9,fprintf
1056    jal  $ra,$t9
1057    li   $v0,1      # 0x1
1058    sw   $v0,64($fp)
1059    b $L58
1060 $L103:
1061    lw   $a0,52($fp)
1062    lw   $a1,56($fp)
1063    la   $t9,procesar_archivo
1064    jal  $ra,$t9
1065    lw   $v1,60($fp)
1066    sw   $v1,72($fp)
1067    li   $v0,1      # 0x1
1068    lw   $v1,72($fp)
1069    beq  $v1,$v0,$L118
1070    lw   $v1,72($fp)
1071    sltu   $v0,$v1,1
1072    bne  $v0,$zero,$L116
1073    li   $v0,2      # 0x2
1074    lw   $v1,72($fp)
1075    beq  $v1,$v0,$L119
1076    li   $v0,3      # 0x3
1077    lw   $v1,72($fp)
1078    beq  $v1,$v0,$L120
1079    b $L116
1080 $L118:
1081    lw   $a0,52($fp)
1082    la   $t9,fclose
1083    jal  $ra,$t9
1084    b $L116
1085 $L119:
```

```
1086    lw  $a0,56($fp)
1087    la  $t9,fclose
1088    jal $ra,$t9
1089    b $L116
1090 $L120:
1091    lw  $a0,52($fp)
1092    la  $t9,fclose
1093    jal $ra,$t9
1094    lw  $a0,56($fp)
1095    la  $t9,fclose
1096    jal $ra,$t9
1097 $L116:
1098    sw  $zero,64($fp)
1099 $L58:
1100    lw  $v0,64($fp)
1101    move  $sp,$fp
1102    lw  $ra,88($sp)
1103    lw  $fp,84($sp)
1104    addu  $sp,$sp,96
1105    j $ra
1106    .end  main
1107    .size main, .-main
1108    .ident  "GCC: (GNU) 3.3.3 (NetBSD nb3 20040520)"
```

7. Código de pruebas automatizadas

```
1 cantidad_archivos_entrada=$(ls -1q Tests/Entrada*.txt |
  wc -l)
2 test_fallidos=0
3 test_pasados=0
4
5 for i in `seq 1 $cantidad_archivos_entrada`
6 do
7         path_entrada='Tests/Entrada'$i'.txt'
8         path_salida='Tests/SalidaEsperada'$i'.txt'
9
10        if [[ $(./a.out -i "$path_entrada" | diff
  "$path_salida" -) ]];
11        then
12                echo "Test $i: ERROR"
13                ./a.out -i $path_entrada | diff
  $path_salida -
14                test_fallidos=$((test_fallidos+1))
15        else
16                echo "Test $i:OK"
17                test_pasados=$((test_pasados+1))
18        fi
19 done
20
21 echo 'Pasados' $test_pasados 'tests de'
  $cantidad_archivos_entrada
```

8. Enunciado

66:20 Organización de Computadoras
Trabajo práctico #0: Infraestructura básica
1^{er} cuatrimestre de 2017

`$Date: 2017/08/22 09:15:02 $`

1. Objetivos

Familiarizarse con las herramientas de software que usaremos en los siguientes trabajos, implementando un programa (y su correspondiente documentación) que resuelva el problema piloto que presentaremos más abajo.

2. Alcance

Este trabajo práctico es de elaboración grupal, evaluación individual, y de carácter obligatorio para todos alumnos del curso.

3. Requisitos

El trabajo deberá ser entregado personalmente, en la fecha estipulada, con una carátula que contenga los datos completos de todos los integrantes.

Además, es necesario que el trabajo práctico incluya (entre otras cosas, ver sección 6), la presentación de los resultados obtenidos explicando, cuando corresponda, con fundamentos reales, las causas o razones de cada resultado obtenido.

El informe deberá respetar el modelo de referencia que se encuentra en el grupo[1], y se valorarán aquellos escritos usando la herramienta TEX / LATEX.

4. Recursos

Usaremos el programa GXemul [1] para simular el entorno de desarrollo que utilizaremos en este y otros trabajos prácticos, una máquina MIPS corriendo una versión reciente del sistema operativo NetBSD [2].

En la clase del 15/8 hemos repasado los pasos necesarios para la instalación y configuración del entorno de desarrollo.

[1] `http://groups.yahoo.com/group/orga-comp`

5. Programa

Se trata de escribir, en lenguaje C, un programa para procesar archivos de texto por línea de comando: el programa recibirá los archivos o *streams* de entrada y salida, y deberá imprimir aquellas palabras del archivo de entrada (componentes léxicos) que sean palíndromos.

A fin de facilitar el proceso de desarrollo y corrección del TP, definiremos como *palabra* a aquellos componentes léxicos del *stream* de entrada computestos exclusivamente por combinaciones de caracteres `a-z`, `0-9`, "`-`" (signo menos) y "`_`" (guión bajo). El juego de caracteres utilizado en un stream de entrada válido es ASCII.

A los efectos de la salida, el comportamiento del programa deberá ser *case insensitive*; es decir, la salida permanece alterada ante permutaciones de mayúsculas y minúsculas.

De no recibir los nombres de los archivos (o en caso de recibir `-` como nombre de archivo) usaremos los *streams* estándar, `stdin` y `stdout`, según corresponda. A continuación, el programa deberá ir leyendo los datos de la entrada, generando la salida correspondiente. De ocurrir errores usaremos `stderr`. Una vez agotados los datos de entrada, el programa debe finalizar adecuadamente, retornando al sistema operativo con un código de finalización adecuado (de tal forma de retornar `0` siempre y cuando el programa finalice normalmente y no hayan ocurrido errores).

5.1. Ejemplos

Primero, usamos la opción `-h` para ver el mensaje de ayuda:

```
$ tp0 -h
Usage:
  tp0 -h
  tp0 -V
  tp0 [options]
Options:
  -V, --version     Print version and quit.
  -h, --help        Print this information.
  -i, --input       Location of the input file.
  -o, --output      Location of the output file.
Examples:
  tp0 -i ~/input -o ~/output
```

Codificamos un archivo vacío (cantidad de bytes nula):

```
$ touch /tmp/zero.txt
$ tp0 -i /tmp/zero.txt -o /tmp/out.txt
$ ls -l /tmp/out.txt
-rw-r--r--  1 user group 0 2017-03-19 15:14 /tmp/out.txt
```

Leemos un *stream* cuyo único contenido es el caracter ASCII `M`,

```
$ echo Hola M | tp0
M
```

Observar que la salida del programa contiene aquellas palabras de la entrada que sean palíndromos (`M` en este caso).

Veamos que sucede al procesar archivo de mayor complejidad:

```
$ cat entrada.txt
Somos los primeros en completar el TP 0.

Ojo que La fecha de entrega del TP0 es el martes 12 de septiembre.
$ tp0 -i entrada.txt -o -
Somos
Ojo
```

6. Informe

El informe deberá incluir al menos las siguientes secciones:

- Documentación relevante al diseño e implementación del programa;
- Comando(s) para compilar el programa;
- Las corridas de prueba, con los comentarios pertinentes;
- El código fuente, en lenguaje C, el cual también deberá entregarse en formato digital compilable (incluyendo archivos de entrada y salida de pruebas);
- El código MIPS32 generado por el compilador;
- Este enunciado.

El informe deberá entregarse en formato impreso y digital.

7. Fechas

- Entrega: 29/8/2017.
- Vencimiento: 12/9/2017.

Referencias

[1] GXemul, `http://gavare.se/gxemul/`.

[2] The NetBSD project, `http://www.netbsd.org/`.