Informe TP0

Cotarelo Rodrigo, *Padrón Nro. 98577*
cotarelorodrigo@gmail.com

Etchegaray Rodrigo, *Padrón Nro. 96856*
rorroeche@gmail.com

Longo Nicolás, *Padrón Nro. 98271*
longo.gnr@hotmail.com

1er. Cuatrimestre de 2019
66.20 Organización de Computadoras – Práctica Martes
Facultad de Ingeniería, Universidad de Buenos Aires

poner la fecha correspondiente

1. Enunciado

1. Objetivos
   Familiarizarse con las herramientas de software que usaremos en los siguien- tes trabajos, implementando un programa (y su correspondiente documenta- ción) 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 7), la presentación de los resultados obtenidos, explicando, cuando co- rresponda, con fundamentos reales, las razones de cada resultado obtenido.

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 12/3 hemos repasado, brevemente, los pasos necesarios para la instalación y configuración del entorno de desarrollo.

5. Programa
   Se trata de escribir en lenguaje C dos programas que permitan convertir archivos de texto desde y hacia plataformas UNIX como las que usamos en los trabajos prácticos.

   En particular, buscamos generar dos programas: unix2dos para transformar archivs de texto de UNIX a Windows y dos2unix para hacer la operación inversa.

   Por defecto, en ambos casos, tomaremos la entrada estándar stdin y salida por stdout. Las opciones -i y -o nos permiten además indicar explícitamente los archivos de entrada y salida usando - para indicar los flujos estándar que ya mencionmos.

   A continuación se presenta un ejemplo de codificación:

   $ (echo Uno; echo Dos; echo Tres) |unix2dos |od -t c

   0000000 U n o \r \n D o s \r \n T r e s \r \n

   0000020

Podemos ver, entonces, que por defecto unix2dos opera usando los streams estandar de entrada/salida, generando los newlines con la convencion adoptada por Windows. Notar ademas el uso del programa od, que nos permite ver caracteres que de otra forma no podrían ser observados.

A continuación, usamos dos2unix para hacer una conversión en el sentido inverso:

$ od -t c /tmp/dos.txt

0000000 U n o \r \n D o s \r \n T r e s \r \n

0000020

$ dos2unix -i /tmp/dos.txt -o - |od -t c

0000000 U n o \n D o s \n T r e s \n

0000015

Aqui /tmp/dos.txt contiene un archivo de texto usando la convención CR+LF para los caracteres newline, y hemos usado el programa dos2unix para transformarlo a la convensión LF adoptada en sistemas UNIX.

6. Casos de prueba
   Se deberá verificar el correcto comportamiento del programa, por lo que los alumnos deberán proponer casos de prueba que crean convenientes, indicando el motivo de la elección de cada caso, indicando el método utilizado para verificar que el programa responde correctamente en cada caso.

7. Portabilidad
   Como es usual, es necesario que la implementación desarrollada provea un grado minímo de portabilidad. Para satisfacer esto, el programa deberá funcio- nar al menos en NetBSD/pmax (usando el simulador GXemul [1]) y la versión de Linux usada para correr el simulador.

8. Informe
   El informe deberá incluir:

   - Documentación relevante al diseño, implementación, validación y utiliza- ción del programa.
   - La documentación necesaria para generar los binarios a partir del código fuente suministrado.
   - Las corridas de prueba, con los comentarios pertinentes.
   - El código fuente, el cual también deberá entregarse en formato digital compilable (incluyendo archivos de entrada y salida de pruebas).

- Este enunciado.

9. Fechas
   Fecha de vencimiento: martes 16/4 de 2019.

2. Implementacion

Se implementaron dos funciones para poder convertir un archivo de DOS a UNIX y viceversa. La implementación se basó en un ciclo while que recorre el archivo caracter por caracter mientras consulta si está parado sobre un '\r' o '\n'. En caso de ser alguno de estos caracteres, se efectua la modificación correspondiente en el archivo resultado o salida y sino, simplemente se escribe el mismo caracter en el documento de salida.

3. Errores que surgieron

Una de las complicaciones que surgieron durante la implementación fue la aparación de un caracter de más en la conversión del archivo. Esto sucedía porque nuestra implementación verificaba si estábamos en el End Of File (EOF) como condición de corte del while para una vez dentro del mismo efectuar la lectura de un nuevo caracter. Lo que sucedía, era que en C al leer el último caracter se seguía apuntando a este y no al EOF. Por lo tanto, esto implicaba hacer una lectura adicional para llegar al EOF y esto producía que se escribiese un caracter vacío en el documento de salida. Esto se solucionó utilizando para la lectura la función fgetc que lee caracer por caracter. fgetc lee el final del archivo y devuelve el EOF a través del código -1. Por lo tanto, cambiamos nuestra condición de corte del while a esperar que la función fgetc devolviese el código -1.

4. Validación

Las primeras pruebas básicas que se hicieron, fueron convertir archivos de texto de un formato a otro chequeando que el archivo de salida tuviese el mismo aspecto que el archivo de entrada. También se hicieron comparaciones entre los archivos de entrada y salida usando el programa *object dump (od)*, para tener una vista un poco más detallada de los documentos y sus caracteres.

Como esto no era suficiente, y las pruebas que deseábamos hacer debían ser más exhaustivas se decidió generar archivos de salida esperados. Es decir, para cada archivo con saltos de línea del tipo LF se generó uno exactamente igual pero con saltos de línea del tipo CR+LF y viceversa. De ésta manera, pudimos comprobar que los archivos generados por el programa son exactamente iguales a aquellos esperados utilizando la directiva *cmp* de la siguiente manera:

```
cmp archivoDePrueba01_salidaReal.txt archivoDePrueba02_salidaEsperada.txt
```

El resultado es el esperado si al ejecutar esa sentencia en la shell la salida es, literalmente, ninguna. De lo contrario, el programa *cmp* habrá encontrado

una diferencia entre los caracteres y eso evidenciaría un problema en la salida de nuestro programa conversor.

5. Utilización

Para ejecutar los programas se utiliza la consola o shell (convencionalmente), donde se corren ambos ejecutables. Dependiendo de qué tipo de conversión de archivo se desee, se usará alguno de los dos programas. La ejecución de cualquiera de estos puede recibir dos parámetros: el archivo de entrada y el archivo de salida. Si no recibe el primero, entonces se setea por defecto la entrada estándar (*stdin*); si no recibe el segundo, se setea por defecto la salida estándar (*stdout*).

Antes del nombre del archivo de entrada se debe especificar el flag '-i' y antes del nombre del archivo de salida, el flag '-o'.

Por ejemplo, para convertir un archivo de DOS a UNIX se ejecuta:

```
./dos2unix -i archivoDeEntradaWindows.txt -o archivoDeSalidaUnix.txt
```

6. Generación de binarios

Para la generación de los binarios a partir de los archivos con el código fuente suministrado se deben seguir los siguientes pasos (usaremos de ejemplo el programa unix2dos, pero aplica para ambos programas):

- Posicionarse en el directorio donde se encuentren los archivos con el código fuente.
- `gcc -Wall -O0 -o unix2dos unix2dos.c`

Para ejecutar los binarios creados, se deben seguir los pasos de la sección de Utilización.

7. Código fuente

7.1. unix2dos

```
/* Se explica el razonamiento a tratar para los parámetros:
en argv[0] está el path del programa
en argv[1] se espera que esté si el archivo de entrada es de windows o linux
en argv[2] se espera que esté el archivo de entrada
*/

#include <stdio.h>
#include <stdlib.h>
```

```
#include <string.h>

#define MAX_NAME 100

const char CR = '\r';
const char LF = '\n';

int mostrarMensajeErrorParametrosInvalidos()
{
  fprintf(stderr, "Los parámetros ingresados no son válidos.\n");
  return -1;
}

void linuxToWindows(FILE* input_file, FILE* output_file)
{
char c;
   while((c = fgetc(input_file)) != -1)
   {
   if ( c == LF )
   {
     fputc(CR, output_file);
     fputc(LF, output_file);
     }
     else
     {
     fputc(c, output_file);
     }
   }
}


int main(int argc, char** argv)
{
  // Inicializo los valores por defecto. No se ingresaron archivos y estos son stdin
  int seIngresoArchivoDeEntrada = 0;
  int seIngresoArchivoDeSalida = 0;
  char *input_fileName = NULL;
  char *output_fileName = NULL;

/* No puedo recibir más de 5 parámetros. Este es el máximo esperado. Por otro lado,
   recibir 1 parámetro (el nombre del programa), 3 parámetros (se especifica archivo
   entrada o de salida) y 5 parámetros (se especifican ambos). Además argc siempre e
   o igual a 1          */
```

```
if (argc > 5 || argc == 4 || argc == 2) {
  return mostrarMensajeErrorParametrosInvalidos();
}

// Caso de recepción de al menos un archivo por parametro
if (argc >= 3) {
  if (strcmp(argv[1], "-i") == 0) {
    input_fileName = argv[2];
    seIngresoArchivoDeEntrada = 1;
  }
  else if (strcmp(argv[1], "-o") == 0) {
    output_fileName = argv[2];
    seIngresoArchivoDeSalida = 1;
  }
  else {
    // Parámetro invalido
    return mostrarMensajeErrorParametrosInvalidos();
  }
  // Caso de recepción de dos archivos por parámetro
  if (argc > 4) {
    if (seIngresoArchivoDeEntrada) {
      if (strcmp(argv[3], "-o") == 0) {
        output_fileName = argv[4];
        seIngresoArchivoDeSalida = 1;
      }
      else {
        // Si ya se ingreso el parametro "-i" el único válido que queda es "-o"
        return mostrarMensajeErrorParametrosInvalidos();
      }
    }
    else if (seIngresoArchivoDeSalida) {
      if (strcmp(argv[3], "-i") == 0) {
        input_fileName = argv[4];
        seIngresoArchivoDeEntrada = 1;
      }
      else {
        // Si ya se ingreso el parametro "-o" el único válido que queda es "-i"
        return mostrarMensajeErrorParametrosInvalidos();
      }
    }
  }
}

FILE* input_file;
```

```
  if (input_fileName != NULL)
  {
    if( (input_file = fopen(input_fileName, "rt")) == NULL)
    {
        fprintf(stderr,"No se pudo abrir el archivo de entrada\n");
        return -1;
    }
  }
  else
    input_file = stdin;

  FILE* output_file;
  if(output_fileName != NULL)
  {
    if( (output_file = fopen(output_fileName, "wt")) == NULL)
    {
        fprintf(stderr,"No se pudo abrir el archivo de salida\n");
        return -1;
    }
  }
  else
    output_file = stdout;

  linuxToWindows(input_file, output_file);

  fclose(input_file);
  fclose(output_file);
  return 0;
}
```

7.2. dos2unix

```
/* Se explica el razonamiento a tratar para los parámetros:
en argv[0] está el path del programa
en argv[1] se espera que esté si el archivo de entrada es de windows o linux
en argv[2] se espera que esté el archivo de entrada
*/

#include <stdio.h>
#include <stdlib.h>
#include <string.h>

#define MAX_NAME 100
```

```
const char CR = '\r';
const char LF = '\n';

int mostrarMensajeErrorParametrosInvalidos()
{
  fprintf(stderr, "Los parámetros ingresados no son válidos.\n");
  return -1;
}

void windowsToLinux(FILE* input_file, FILE* output_file)
{
char c, d;

while((c = fgetc(input_file)) != -1)
{
if ( c == CR )
{
       if((d = fgetc(input_file)) == -1)
       {
         fputc(CR, output_file);
         break;
       }
       else
       {
         if( d == LF )
         {
           fputc(LF, output_file);
         }
        else
        {
           fputc(CR, output_file);
           fputc(d, output_file);
        }
       }
     }
     else
     {
       fputc(c, output_file);
     }

}
}
```

```
int main(int argc, char** argv)
{
  // Inicializo los valores por defecto. No se ingresaron archivos y estos son stdir
  int seIngresoArchivoDeEntrada = 0;
  int seIngresoArchivoDeSalida = 0;
  char *input_fileName = NULL;
  char *output_fileName = NULL;

/* No puedo recibir más de 5 parámetros. Este es el máximo esperado. Por otro lado,
   recibir 1 parámetro (el nombre del programa), 3 parámetros (se especifica archivo
   entrada o de salida) y 5 parámetros (se especifican ambos). Además argc siempre e
   o igual a 1          */

  if (argc > 5 || argc == 4 || argc == 2) {
    return mostrarMensajeErrorParametrosInvalidos();
  }

  // Caso de recepción de al menos un archivo por parametro
  if (argc >= 3) {
    if (strcmp(argv[1], "-i") == 0) {
      input_fileName = argv[2];
      seIngresoArchivoDeEntrada = 1;
    }
    else if (strcmp(argv[1], "-o") == 0) {
      output_fileName = argv[2];
      seIngresoArchivoDeSalida = 1;
    }
    else {
      // Parámetro invalido
      return mostrarMensajeErrorParametrosInvalidos();
    }
    // Caso de recepción de dos archivos por parámetro
    if (argc > 4) {
      if (seIngresoArchivoDeEntrada) {
        if (strcmp(argv[3], "-o") == 0) {
          output_fileName = argv[4];
          seIngresoArchivoDeSalida = 1;
        }
        else {
          // Si ya se ingreso el parametro "-i" el único válido que queda es "-o"
          return mostrarMensajeErrorParametrosInvalidos();
        }
      }
      else if (seIngresoArchivoDeSalida) {
```

```
        if (strcmp(argv[3], "-i") == 0) {
          input_fileName = argv[4];
          seIngresoArchivoDeEntrada = 1;
        }
        else {
          // Si ya se ingreso el parametro "-o" el único válido que queda es "-i"
          return mostrarMensajeErrorParametrosInvalidos();
        }
      }
    }
  }

  FILE* input_file;
  if (input_fileName != NULL)
  {
    if( (input_file = fopen(input_fileName, "rt")) == NULL)
    {
        fprintf(stderr,"No se pudo abrir el archivo de entrada\n");
        return -1;
    }
  }
  else
    input_file = stdin;

  FILE* output_file;
  if(output_fileName != NULL)
  {
    if( (output_file = fopen(output_fileName, "wt")) == NULL)
    {
        fprintf(stderr,"No se pudo abrir el archivo de salida\n");
        return -1;
    }
  }
  else
    output_file = stdout;

  windowsToLinux(input_file, output_file);

  fclose(input_file);
  fclose(output_file);
  return 0;
}
```

8. Código MIPS

8.1. unix2dos

```
.file 1 "unix2dos.c"
.section .mdebug.abi32
.previous
.abicalls
.globl CR
.rdata
.type CR, @object
.size CR, 1
CR:
.byte 13
.globl LF
.type LF, @object
.size LF, 1
LF:
.byte 10
.align 2
$LC0:
.ascii "Los par\303\241metros ingresados no son v\303\241lidos.\n"
.ascii "\000"
.text
.align 2
.globl mostrarMensajeErrorParametrosInvalidos
.ent mostrarMensajeErrorParametrosInvalidos
mostrarMensajeErrorParametrosInvalidos:
.frame $fp,40,$ra # vars= 0, regs= 3/0, args= 16, extra= 8
.mask 0xd0000000,-8
.fmask 0x00000000,0
.set noreorder
.cpload $t9
.set reorder
subu $sp,$sp,40
.cprestore 16
sw $ra,32($sp)
sw $fp,28($sp)
sw $gp,24($sp)
move $fp,$sp
la $a0,__sF+176
la $a1,$LC0
la $t9,fprintf
jal $ra,$t9
```

```
li $v0,-1 # 0xffffffffffffffff
move $sp,$fp
lw $ra,32($sp)
lw $fp,28($sp)
addu $sp,$sp,40
j $ra
.end mostrarMensajeErrorParametrosInvalidos
.size mostrarMensajeErrorParametrosInvalidos, .-mostrarMensajeErrorParametrosInvalid
.align 2
.globl linuxToWindows
.ent linuxToWindows
linuxToWindows:
.frame $fp,48,$ra # vars= 8, regs= 3/0, args= 16, extra= 8
.mask 0xd0000000,-8
.fmask 0x00000000,0
.set noreorder
.cpload $t9
.set reorder
subu $sp,$sp,48
.cprestore 16
sw $ra,40($sp)
sw $fp,36($sp)
sw $gp,32($sp)
move $fp,$sp
sw $a0,48($fp)
sw $a1,52($fp)
$L19:
lw $a0,48($fp)
la $t9,fgetc
jal $ra,$t9
sb $v0,24($fp)
lbu $v0,24($fp)
sll $v0,$v0,24
sra $v1,$v0,24
li $v0,-1 # 0xffffffffffffffff
bne $v1,$v0,$L21
b $L18
$L21:
lb $v1,24($fp)
lb $v0,LF
bne $v1,$v0,$L22
lb $v0,CR
move $a0,$v0
lw $a1,52($fp)
```

```
la $t9,fputc
jal $ra,$t9
lb $v0,LF
move $a0,$v0
lw $a1,52($fp)
la $t9,fputc
jal $ra,$t9
b $L19
$L22:
lb $v0,24($fp)
move $a0,$v0
lw $a1,52($fp)
la $t9,fputc
jal $ra,$t9
b $L19
$L18:
move $sp,$fp
lw $ra,40($sp)
lw $fp,36($sp)
addu $sp,$sp,48
j $ra
.end linuxToWindows
.size linuxToWindows, .-linuxToWindows
.rdata
.align 2
$LC1:
.ascii "-i\000"
.align 2
$LC2:
.ascii "-o\000"
.align 2
$LC3:
.ascii "rt\000"
.align 2
$LC4:
.ascii "No se pudo abrir el archivo de entrada\n\000"
.align 2
$LC5:
.ascii "wt\000"
.align 2
$LC6:
.ascii "No se pudo abrir el archivo de salida\n\000"
.text
.align 2
```

```
.globl main
.ent main
main:
.frame $fp,72,$ra # vars= 32, regs= 3/0, args= 16, extra= 8
.mask 0xd0000000,-8
.fmask 0x00000000,0
.set noreorder
.cpload $t9
.set reorder
subu $sp,$sp,72
.cprestore 16
sw $ra,64($sp)
sw $fp,60($sp)
sw $gp,56($sp)
move $fp,$sp
sw $a0,72($fp)
sw $a1,76($fp)
sw $zero,24($fp)
sw $zero,28($fp)
sw $zero,32($fp)
sw $zero,36($fp)
lw $v0,72($fp)
slt $v0,$v0,6
beq $v0,$zero,$L26
lw $v1,72($fp)
li $v0,4 # 0x4
beq $v1,$v0,$L26
lw $v1,72($fp)
li $v0,2 # 0x2
beq $v1,$v0,$L26
b $L25
$L26:
la $t9,mostrarMensajeErrorParametrosInvalidos
jal $ra,$t9
sw $v0,48($fp)
b $L24
$L25:
lw $v0,72($fp)
slt $v0,$v0,3
bne $v0,$zero,$L27
lw $v0,76($fp)
addu $v0,$v0,4
lw $a0,0($v0)
la $a1,$LC1
```

```
la $t9,strcmp
jal $ra,$t9
bne $v0,$zero,$L28
lw $v0,76($fp)
addu $v0,$v0,8
lw $v0,0($v0)
sw $v0,32($fp)
li $v0,1 # 0x1
sw $v0,24($fp)
b $L29
$L28:
lw $v0,76($fp)
addu $v0,$v0,4
lw $a0,0($v0)
la $a1,$LC2
la $t9,strcmp
jal $ra,$t9
bne $v0,$zero,$L30
lw $v0,76($fp)
addu $v0,$v0,8
lw $v0,0($v0)
sw $v0,36($fp)
li $v0,1 # 0x1
sw $v0,28($fp)
b $L29
$L30:
la $t9,mostrarMensajeErrorParametrosInvalidos
jal $ra,$t9
sw $v0,48($fp)
b $L24
$L29:
lw $v0,72($fp)
slt $v0,$v0,5
bne $v0,$zero,$L27
lw $v0,24($fp)
beq $v0,$zero,$L33
lw $v0,76($fp)
addu $v0,$v0,12
lw $a0,0($v0)
la $a1,$LC2
la $t9,strcmp
jal $ra,$t9
bne $v0,$zero,$L34
lw $v0,76($fp)
```

```
addu $v0,$v0,16
lw $v0,0($v0)
sw $v0,36($fp)
li $v0,1 # 0x1
sw $v0,28($fp)
b $L27
$L34:
la $t9,mostrarMensajeErrorParametrosInvalidos
jal $ra,$t9
sw $v0,48($fp)
b $L24
$L33:
lw $v0,28($fp)
beq $v0,$zero,$L27
lw $v0,76($fp)
addu $v0,$v0,12
lw $a0,0($v0)
la $a1,$LC1
la $t9,strcmp
jal $ra,$t9
bne $v0,$zero,$L38
lw $v0,76($fp)
addu $v0,$v0,16
lw $v0,0($v0)
sw $v0,32($fp)
li $v0,1 # 0x1
sw $v0,24($fp)
b $L27
$L38:
la $t9,mostrarMensajeErrorParametrosInvalidos
jal $ra,$t9
sw $v0,48($fp)
b $L24
$L27:
lw $v0,32($fp)
beq $v0,$zero,$L40
lw $a0,32($fp)
la $a1,$LC3
la $t9,fopen
jal $ra,$t9
sw $v0,40($fp)
lw $v0,40($fp)
bne $v0,$zero,$L42
la $a0,__sF+176
```

```
la $a1,$LC4
la $t9,fprintf
jal $ra,$t9
li $v0,-1 # 0xffffffffffffffff
sw $v0,48($fp)
b $L24
$L40:
la $v0,__sF
sw $v0,40($fp)
$L42:
lw $v0,36($fp)
beq $v0,$zero,$L43
lw $a0,36($fp)
la $a1,$LC5
la $t9,fopen
jal $ra,$t9
sw $v0,44($fp)
lw $v0,44($fp)
bne $v0,$zero,$L45
la $a0,__sF+176
la $a1,$LC6
la $t9,fprintf
jal $ra,$t9
li $v0,-1 # 0xffffffffffffffff
sw $v0,48($fp)
b $L24
$L43:
la $v0,__sF+88
sw $v0,44($fp)
$L45:
lw $a0,40($fp)
lw $a1,44($fp)
la $t9,linuxToWindows
jal $ra,$t9
lw $a0,40($fp)
la $t9,fclose
jal $ra,$t9
lw $a0,44($fp)
la $t9,fclose
jal $ra,$t9
sw $zero,48($fp)
$L24:
lw $v0,48($fp)
move $sp,$fp
```

```
lw $ra,64($sp)
lw $fp,60($sp)
addu $sp,$sp,72
j $ra
.end main
.size main, .-main
.ident "GCC: (GNU) 3.3.3 (NetBSD nb3 20040520)"
```

8.2. dos2unix

```
.file 1 "dos2unix.c"
.section .mdebug.abi32
.previous
.abicalls
.globl CR
.rdata
.type CR, @object
.size CR, 1
CR:
.byte 13
.globl LF
.type LF, @object
.size LF, 1
LF:
.byte 10
.align 2
$LC0:
.ascii "Los par\303\241metros ingresados no son v\303\241lidos.\n"
.ascii "\000"
.text
.align 2
.globl mostrarMensajeErrorParametrosInvalidos
.ent mostrarMensajeErrorParametrosInvalidos
mostrarMensajeErrorParametrosInvalidos:
.frame $fp,40,$ra # vars= 0, regs= 3/0, args= 16, extra= 8
.mask 0xd0000000,-8
.fmask 0x00000000,0
.set noreorder
.cpload $t9
.set reorder
subu $sp,$sp,40
.cprestore 16
sw $ra,32($sp)
sw $fp,28($sp)
```

```
sw $gp,24($sp)
move $fp,$sp
la $a0,__sF+176
la $a1,$LC0
la $t9,fprintf
jal $ra,$t9
li $v0,-1 # 0xffffffffffffffff
move $sp,$fp
lw $ra,32($sp)
lw $fp,28($sp)
addu $sp,$sp,40
j $ra
.end mostrarMensajeErrorParametrosInvalidos
.size mostrarMensajeErrorParametrosInvalidos, .-mostrarMensajeErrorParametrosInvalid
.align 2
.globl windowsToLinux
.ent windowsToLinux
windowsToLinux:
.frame $fp,48,$ra # vars= 8, regs= 3/0, args= 16, extra= 8
.mask 0xd0000000,-8
.fmask 0x00000000,0
.set noreorder
.cpload $t9
.set reorder
subu $sp,$sp,48
.cprestore 16
sw $ra,40($sp)
sw $fp,36($sp)
sw $gp,32($sp)
move $fp,$sp
sw $a0,48($fp)
sw $a1,52($fp)
$L19:
lw $a0,48($fp)
la $t9,fgetc
jal $ra,$t9
sb $v0,24($fp)
lbu $v0,24($fp)
sll $v0,$v0,24
sra $v1,$v0,24
li $v0,-1 # 0xffffffffffffffff
bne $v1,$v0,$L21
b $L18
$L21:
```

```
lb $v1,24($fp)
lb $v0,CR
bne $v1,$v0,$L22
lw $a0,48($fp)
la $t9,fgetc
jal $ra,$t9
sb $v0,25($fp)
lbu $v0,25($fp)
sll $v0,$v0,24
sra $v1,$v0,24
li $v0,-1 # 0xffffffffffffffff
bne $v1,$v0,$L23
lb $v0,CR
move $a0,$v0
lw $a1,52($fp)
la $t9,fputc
jal $ra,$t9
b $L18
$L23:
lb $v1,25($fp)
lb $v0,LF
bne $v1,$v0,$L25
lb $v0,LF
move $a0,$v0
lw $a1,52($fp)
la $t9,fputc
jal $ra,$t9
b $L19
$L25:
lb $v0,CR
move $a0,$v0
lw $a1,52($fp)
la $t9,fputc
jal $ra,$t9
lb $v0,25($fp)
move $a0,$v0
lw $a1,52($fp)
la $t9,fputc
jal $ra,$t9
b $L19
$L22:
lb $v0,24($fp)
move $a0,$v0
lw $a1,52($fp)
```

```
la $t9,fputc
jal $ra,$t9
b $L19
$L18:
move $sp,$fp
lw $ra,40($sp)
lw $fp,36($sp)
addu $sp,$sp,48
j $ra
.end windowsToLinux
.size windowsToLinux, .-windowsToLinux
.rdata
.align 2
$LC1:
.ascii "-i\000"
.align 2
$LC2:
.ascii "-o\000"
.align 2
$LC3:
.ascii "rt\000"
.align 2
$LC4:
.ascii "No se pudo abrir el archivo de entrada\n\000"
.align 2
$LC5:
.ascii "wt\000"
.align 2
$LC6:
.ascii "No se pudo abrir el archivo de salida\n\000"
.text
.align 2
.globl main
.ent main
main:
.frame $fp,72,$ra # vars= 32, regs= 3/0, args= 16, extra= 8
.mask 0xd0000000,-8
.fmask 0x00000000,0
.set noreorder
.cpload $t9
.set reorder
subu $sp,$sp,72
.cprestore 16
sw $ra,64($sp)
```

```
sw $fp,60($sp)
sw $gp,56($sp)
move $fp,$sp
sw $a0,72($fp)
sw $a1,76($fp)
sw $zero,24($fp)
sw $zero,28($fp)
sw $zero,32($fp)
sw $zero,36($fp)
lw $v0,72($fp)
slt $v0,$v0,6
beq $v0,$zero,$L30
lw $v1,72($fp)
li $v0,4 # 0x4
beq $v1,$v0,$L30
lw $v1,72($fp)
li $v0,2 # 0x2
beq $v1,$v0,$L30
b $L29
$L30:
la $t9,mostrarMensajeErrorParametrosInvalidos
jal $ra,$t9
sw $v0,48($fp)
b $L28
$L29:
lw $v0,72($fp)
slt $v0,$v0,3
bne $v0,$zero,$L31
lw $v0,76($fp)
addu $v0,$v0,4
lw $a0,0($v0)
la $a1,$LC1
la $t9,strcmp
jal $ra,$t9
bne $v0,$zero,$L32
lw $v0,76($fp)
addu $v0,$v0,8
lw $v0,0($v0)
sw $v0,32($fp)
li $v0,1 # 0x1
sw $v0,24($fp)
b $L33
$L32:
lw $v0,76($fp)
```

```
addu $v0,$v0,4
lw $a0,0($v0)
la $a1,$LC2
la $t9,strcmp
jal $ra,$t9
bne $v0,$zero,$L34
lw $v0,76($fp)
addu $v0,$v0,8
lw $v0,0($v0)
sw $v0,36($fp)
li $v0,1 # 0x1
sw $v0,28($fp)
b $L33
$L34:
la $t9,mostrarMensajeErrorParametrosInvalidos
jal $ra,$t9
sw $v0,48($fp)
b $L28
$L33:
lw $v0,72($fp)
slt $v0,$v0,5
bne $v0,$zero,$L31
lw $v0,24($fp)
beq $v0,$zero,$L37
lw $v0,76($fp)
addu $v0,$v0,12
lw $a0,0($v0)
la $a1,$LC2
la $t9,strcmp
jal $ra,$t9
bne $v0,$zero,$L38
lw $v0,76($fp)
addu $v0,$v0,16
lw $v0,0($v0)
sw $v0,36($fp)
li $v0,1 # 0x1
sw $v0,28($fp)
b $L31
$L38:
la $t9,mostrarMensajeErrorParametrosInvalidos
jal $ra,$t9
sw $v0,48($fp)
b $L28
$L37:
```

```
lw $v0,28($fp)
beq $v0,$zero,$L31
lw $v0,76($fp)
addu $v0,$v0,12
lw $a0,0($v0)
la $a1,$LC1
la $t9,strcmp
jal $ra,$t9
bne $v0,$zero,$L42
lw $v0,76($fp)
addu $v0,$v0,16
lw $v0,0($v0)
sw $v0,32($fp)
li $v0,1 # 0x1
sw $v0,24($fp)
b $L31
$L42:
la $t9,mostrarMensajeErrorParametrosInvalidos
jal $ra,$t9
sw $v0,48($fp)
b $L28
$L31:
lw $v0,32($fp)
beq $v0,$zero,$L44
lw $a0,32($fp)
la $a1,$LC3
la $t9,fopen
jal $ra,$t9
sw $v0,40($fp)
lw $v0,40($fp)
bne $v0,$zero,$L46
la $a0,__sF+176
la $a1,$LC4
la $t9,fprintf
jal $ra,$t9
li $v0,-1 # 0xffffffffffffffff
sw $v0,48($fp)
b $L28
$L44:
la $v0,__sF
sw $v0,40($fp)
$L46:
lw $v0,36($fp)
beq $v0,$zero,$L47
```

```
lw $a0,36($fp)
la $a1,$LC5
la $t9,fopen
jal $ra,$t9
sw $v0,44($fp)
lw $v0,44($fp)
bne $v0,$zero,$L49
la $a0,__sF+176
la $a1,$LC6
la $t9,fprintf
jal $ra,$t9
li $v0,-1 # 0xffffffffffffffff
sw $v0,48($fp)
b $L28
$L47:
la $v0,__sF+88
sw $v0,44($fp)
$L49:
lw $a0,40($fp)
lw $a1,44($fp)
la $t9,windowsToLinux
jal $ra,$t9
lw $a0,40($fp)
la $t9,fclose
jal $ra,$t9
lw $a0,44($fp)
la $t9,fclose
jal $ra,$t9
sw $zero,48($fp)
$L28:
lw $v0,48($fp)
move $sp,$fp
lw $ra,64($sp)
lw $fp,60($sp)
addu $sp,$sp,72
j $ra
.end main
.size main, .-main
.ident "GCC: (GNU) 3.3.3 (NetBSD nb3 20040520)"
```