**Informe**

El tp lo comenzamos realizando un makefile bastante simple, con algunos flags, para poder compilarlo y obtener el código assembler cuando ya estuviese terminado.

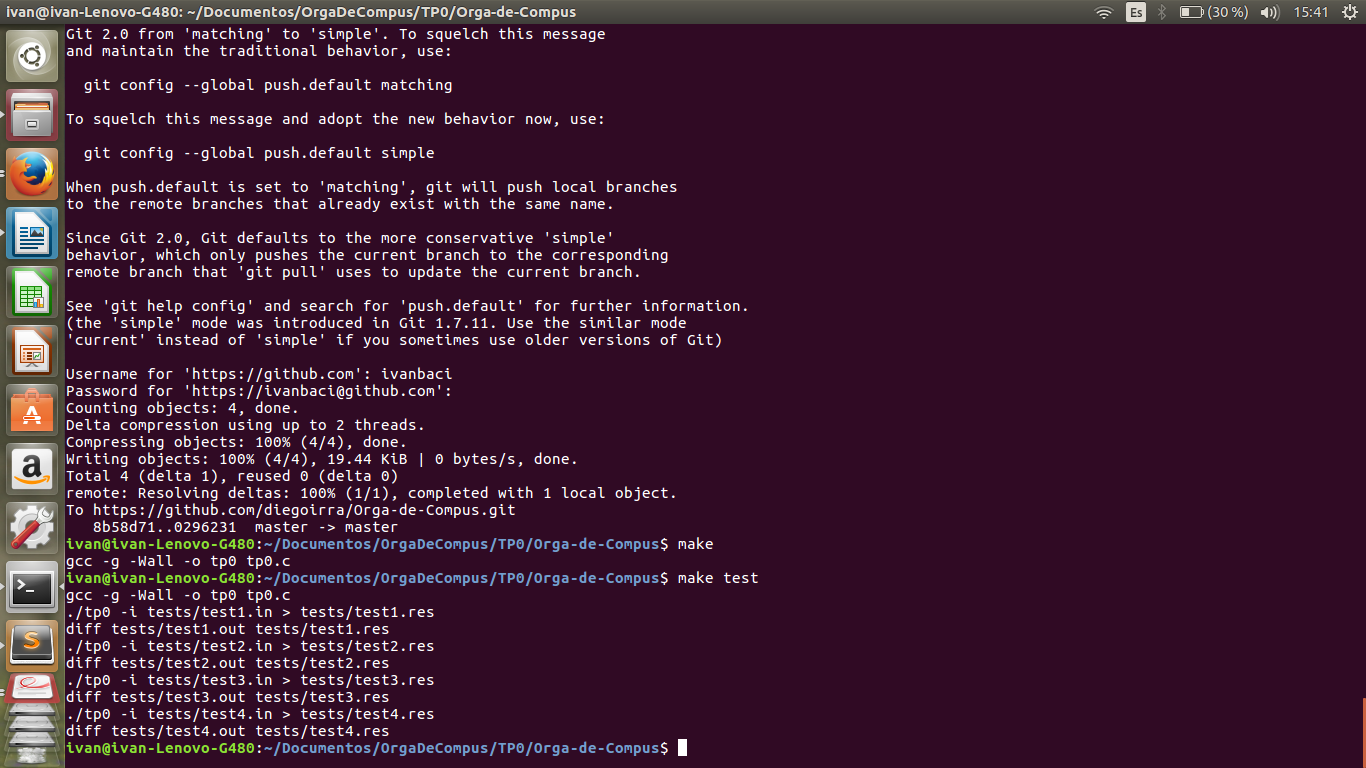
Luego comenzamos con el código ya propio de lo pedido en la consigna. En primer lugar, armamos el main de manera que distinguiese que opción se había ingresado para ejecutar el programa. Al verificar esto, armamos una función para el comando -h, la cual imprime lo pedido; con el comando -V directamente la versión y, por último, hay una función encargada de verificar cual es el archivo de entrada y cual el de salida del programa. En los primeros dos casos el programa termina inmediatamente después de la impresión.

Esta función, llamada handle, para comenzar a trabajar hicimos que obligatoriamente tenga que recibir un archivo de entrada y uno de salida. Ya con esto andando, le agregamos funcionalidad para llegar a lo pedido. Para esto realizamos chequeos de los argumentos recibidos a la hora de ejecutar el programa.

Primero se hacen las verificaciones del archivo de entrada. Si lo recibido para este mismo es distinto de NULL, lo abre para lectura; en caso contrario, toma como entrada por defecto la terminal (stdin). Luego, de manera similar, verifica el archivo de salida. Si es distinto de NULL, lo abre pero para escritura en este caso y, de manera contraria, si no se le pasó por comando de entrada, utiliza por defecto la terminal como salida (stdout). Cualquier error en la apertura de los archivos se informa mediante la salida estándar stderr.

Una vez terminados los chequeos relacionados a los archivos, lee del archivo de entrada y llama a la función es\_palindromo, la cual de manera muy simple chequea si la palabra recibida por parámetro es o no un palíndromo. Finalmente se vuelve a la función handle, en donde se cierran los archivos y finaliza la ejecución del programa.

Finalizado el código del programa, realizamos unas pruebas, las cuales se ejecutan desde el Makefile. Funcionan de la siguiente manera: agarran un archivo test.in y generan el resultado del programa en un archivo de salida test.res. Luego, comparan eso con un test.out, el cual tiene el resultado esperado. Para esta comparación se utiliza el comando diff, el cual devuelve las diferencias entre los dos archivos que recibe. Esto quiere decir que, en caso de haber diferencias entre lo obtenido y lo esperado, se generaría una salida, lo cual se puede ver que no ocurre.



**Makefile**

CC=gcc

CFLAGS := -g -Wall

INPUT=tp0.c

OUTPUT=tp0

TESTS\_DIR=tests

compile:

$(CC) $(CFLAGS) -o $(OUTPUT) $(INPUT)

asm:

$(CC) $(CFLAGS) -O0 -S $(INPUT)

test: compile

./$(OUTPUT) -i $(TESTS\_DIR)/test1.in > $(TESTS\_DIR)/test1.res

diff $(TESTS\_DIR)/test1.out $(TESTS\_DIR)/test1.res

./$(OUTPUT) -i $(TESTS\_DIR)/test2.in > $(TESTS\_DIR)/test2.res

diff $(TESTS\_DIR)/test2.out $(TESTS\_DIR)/test2.res

./$(OUTPUT) -i $(TESTS\_DIR)/test3.in > $(TESTS\_DIR)/test3.res

diff $(TESTS\_DIR)/test3.out $(TESTS\_DIR)/test3.res

./$(OUTPUT) -i $(TESTS\_DIR)/test4.in > $(TESTS\_DIR)/test4.res

diff $(TESTS\_DIR)/test4.out $(TESTS\_DIR)/test4.res

clean:

rm -f $(OUTPUT)

rm -f $(OUTPUT).s

rm -f $(TESTS\_DIR)/\*.res

**Código en C**

#include <stdio.h>

#include <stdlib.h>

#include <string.h>

#include <ctype.h>

#define SUCCESS 0

#define ERROR\_INPUT\_FILE 1

#define ERROR\_OUTPUT\_FILE 2

#define VERSION "1.0"

void print\_version() {

printf("%s\n", VERSION);

}

void print\_usage() {

printf("Usage:\n");

printf("\ttp0 -h\n");

printf("\ttp0 -V\n");

printf("\ttp0 [options]\n");

printf("Options:\n");

printf("\t-V, --version\tPrint version and quit.\n");

printf("\t-h, --help\tPrint this information.\n");

printf("\t-i, --input\tLocation of the input file.\n");

printf("\t-o, --output\tLocation of the output file.\n");

printf("Examples:\n");

printf("\ttp0 -i ~/input -o ~/output\n");

}

int es\_palindromo(char\* word){

unsigned int i, j;

int es\_palindromo = 1;

j = strlen(word)-1;

for(i=0; i< (strlen(word)/2); i++, j--) {

if (tolower(word[i]) != tolower(word[j])) {

es\_palindromo = 0;

break;

}

}

return es\_palindromo;

}

void handle(char\* input\_file\_name, char\* output\_file\_name) {

FILE \*in\_f;

FILE \*out\_f;

if (input\_file\_name != NULL && strcmp(input\_file\_name, "-") != 0) {

in\_f = fopen( input\_file\_name, "r" );

if (in\_f == NULL) {

fprintf(stderr, "No se pudo abrir el archivo de entrada: %s\n", input\_file\_name);

exit(ERROR\_INPUT\_FILE);

}

} else {

in\_f = stdin;

}

if (output\_file\_name!=NULL && strcmp(output\_file\_name, "-") != 0) {

out\_f = fopen( output\_file\_name, "w" );

if (out\_f == NULL) {

fprintf(stderr, "No se pudo abrir el archivo de salida: %s\n", output\_file\_name);

exit(ERROR\_OUTPUT\_FILE);

}

}

else {

out\_f = stdout;

}

char word[30];

while (fscanf(in\_f, " %30s", word) == 1) {

if (es\_palindromo(word)==1){

fputs(word, out\_f);

fputs("\n", out\_f);

}

}

if(in\_f != NULL && in\_f != stdin) {

fclose(in\_f);

}

if(out\_f != NULL && out\_f != stdout) {

fclose(out\_f);

}

exit(SUCCESS);

}

int main(int argc, char\*\* argv) {

if(argc == 1) {

handle(NULL, NULL);

} else if (argc == 2) {

if(strcmp(argv[1], "-h") == 0 || strcmp(argv[1], "--help")==0) {

print\_usage();

}

if(strcmp(argv[1], "-V") == 0 || strcmp(argv[1], "--version")==0) {

print\_version();

}

} else if(argc == 3) {

if(strcmp(argv[1], "-i") == 0) {

handle(argv[2], NULL);

}

if(strcmp(argv[1], "-o") == 0) {

handle(NULL, argv[2]);

}

} else if(argc == 5) {

if(strcmp(argv[1], "-i") == 0 && strcmp(argv[3], "-o") == 0) {

handle(argv[2], argv[4]);

}

if(strcmp(argv[1], "-o") == 0 && strcmp(argv[3], "-i") == 0) {

handle(argv[4], argv[2]);

}

}

print\_usage();

return SUCCESS;

}

**Código assembler**

.section .mdebug.abi32

.previous

.abicalls

.file 1 "tp0.c"

.section .debug\_abbrev,"",@progbits

$Ldebug\_abbrev0:

.section .debug\_info,"",@progbits

$Ldebug\_info0:

.section .debug\_line,"",@progbits

$Ldebug\_line0:

.text

$Ltext0:

.file 2 "/usr/include/mips/int\_types.h"

.file 3 "/usr/include/sys/ansi.h"

.file 4 "/usr/include/mips/ansi.h"

.file 5 "/usr/include/stdio.h"

.file 6 "/usr/include/mips/types.h"

.file 7 "/usr/include/sys/types.h"

.file 8 "/usr/include/sys/endian.h"

.file 9 "/usr/include/pthread\_types.h"

.file 10 "/usr/include/stdlib.h"

.rdata

.align 2

$LC0:

.ascii "%s\n\000"

.align 2

$LC1:

.ascii "1.0\000"

.text

.align 2

.globl print\_version

$LFB29:

.loc 1 11 0

.ent print\_version

print\_version:

.frame $fp,40,$31 # vars= 0, regs= 3/0, args= 16, extra= 8

.mask 0xd0000000,-8

.fmask 0x00000000,0

.set noreorder

.cpload $25

.set reorder

subu $sp,$sp,40

.cprestore 16

$LCFI0:

sw $31,32($sp)

$LCFI1:

sw $fp,28($sp)

$LCFI2:

sw $28,24($sp)

$LCFI3:

move $fp,$sp

$LCFI4:

.loc 1 12 0

la $4,$LC0

la $5,$LC1

la $25,printf

jal $31,$25

.loc 1 13 0

move $sp,$fp

lw $31,32($sp)

lw $fp,28($sp)

addu $sp,$sp,40

j $31

.end print\_version

$LFE29:

.size print\_version, .-print\_version

.rdata

.align 2

$LC2:

.ascii "Usage:\n\000"

.align 2

$LC3:

.ascii "\ttp0 -h\n\000"

.align 2

$LC4:

.ascii "\ttp0 -V\n\000"

.align 2

$LC5:

.ascii "\ttp0 [options]\n\000"

.align 2

$LC6:

.ascii "Options:\n\000"

.align 2

$LC7:

.ascii "\t-V, --version\tPrint version and quit.\n\000"

.align 2

$LC8:

.ascii "\t-h, --help\tPrint this information.\n\000"

.align 2

$LC9:

.ascii "\t-i, --input\tLocation of the input file.\n\000"

.align 2

$LC10:

.ascii "\t-o, --output\tLocation of the output file.\n\000"

.align 2

$LC11:

.ascii "Examples:\n\000"

.align 2

$LC12:

.ascii "\ttp0 -i ~/input -o ~/output\n\000"

.text

.align 2

.globl print\_usage

$LFB31:

.loc 1 15 0

.ent print\_usage

print\_usage:

.frame $fp,40,$31 # vars= 0, regs= 3/0, args= 16, extra= 8

.mask 0xd0000000,-8

.fmask 0x00000000,0

.set noreorder

.cpload $25

.set reorder

subu $sp,$sp,40

.cprestore 16

$LCFI5:

sw $31,32($sp)

$LCFI6:

sw $fp,28($sp)

$LCFI7:

sw $28,24($sp)

$LCFI8:

move $fp,$sp

$LCFI9:

.loc 1 16 0

la $4,$LC2

la $25,printf

jal $31,$25

.loc 1 17 0

la $4,$LC3

la $25,printf

jal $31,$25

.loc 1 18 0

la $4,$LC4

la $25,printf

jal $31,$25

.loc 1 19 0

la $4,$LC5

la $25,printf

jal $31,$25

.loc 1 20 0

la $4,$LC6

la $25,printf

jal $31,$25

.loc 1 21 0

la $4,$LC7

la $25,printf

jal $31,$25

.loc 1 22 0

la $4,$LC8

la $25,printf

jal $31,$25

.loc 1 23 0

la $4,$LC9

la $25,printf

jal $31,$25

.loc 1 24 0

la $4,$LC10

la $25,printf

jal $31,$25

.loc 1 25 0

la $4,$LC11

la $25,printf

jal $31,$25

.loc 1 26 0

la $4,$LC12

la $25,printf

jal $31,$25

.loc 1 27 0

move $sp,$fp

lw $31,32($sp)

lw $fp,28($sp)

addu $sp,$sp,40

j $31

.end print\_usage

$LFE31:

.size print\_usage, .-print\_usage

.align 2

.globl es\_palindromo

$LFB33:

.loc 1 29 0

.ent es\_palindromo

es\_palindromo:

.frame $fp,56,$31 # vars= 16, regs= 3/0, args= 16, extra= 8

.mask 0xd0000000,-8

.fmask 0x00000000,0

.set noreorder

.cpload $25

.set reorder

subu $sp,$sp,56

.cprestore 16

$LCFI10:

sw $31,48($sp)

$LCFI11:

sw $fp,44($sp)

$LCFI12:

sw $28,40($sp)

$LCFI13:

move $fp,$sp

$LCFI14:

sw $4,56($fp)

.loc 1 31 0

$LBB2:

li $2,1 # 0x1

sw $2,32($fp)

.loc 1 32 0

lw $4,56($fp)

la $25,strlen

jal $31,$25

addu $2,$2,-1

sw $2,28($fp)

.loc 1 33 0

sw $0,24($fp)

$L20:

lw $4,56($fp)

la $25,strlen

jal $31,$25

srl $3,$2,1

lw $2,24($fp)

sltu $2,$2,$3

bne $2,$0,$L23

b $L21

$L23:

.loc 1 34 0

lw $3,56($fp)

lw $2,24($fp)

addu $2,$3,$2

lb $2,0($2)

sll $3,$2,1

lw $2,\_tolower\_tab\_

addu $2,$3,$2

addu $4,$2,2

lw $3,56($fp)

lw $2,28($fp)

addu $2,$3,$2

lb $2,0($2)

sll $3,$2,1

lw $2,\_tolower\_tab\_

addu $2,$3,$2

addu $2,$2,2

lh $3,0($4)

lh $2,0($2)

beq $3,$2,$L22

.loc 1 35 0

sw $0,32($fp)

.loc 1 36 0

b $L21

.loc 1 33 0

$L22:

lw $2,24($fp)

addu $2,$2,1

sw $2,24($fp)

lw $2,28($fp)

addu $2,$2,-1

sw $2,28($fp)

b $L20

$L21:

.loc 1 39 0

lw $2,32($fp)

.loc 1 40 0

move $sp,$fp

lw $31,48($sp)

lw $fp,44($sp)

addu $sp,$sp,56

j $31

$LBE2:

.end es\_palindromo

$LFE33:

.size es\_palindromo, .-es\_palindromo

.rdata

.align 2

$LC13:

.ascii "-\000"

.align 2

$LC14:

.ascii "r\000"

.align 2

$LC15:

.ascii "No se pudo abrir el archivo de entrada: %s\n\000"

.align 2

$LC16:

.ascii "w\000"

.align 2

$LC17:

.ascii "No se pudo abrir el archivo de salida: %s\n\000"

.align 2

$LC18:

.ascii " %30s\000"

.align 2

$LC19:

.ascii "\n\000"

.text

.align 2

.globl handle

$LFB35:

.loc 1 42 0

.ent handle

handle:

.frame $fp,80,$31 # vars= 40, regs= 3/0, args= 16, extra= 8

.mask 0xd0000000,-8

.fmask 0x00000000,0

.set noreorder

.cpload $25

.set reorder

subu $sp,$sp,80

.cprestore 16

$LCFI15:

sw $31,72($sp)

$LCFI16:

sw $fp,68($sp)

$LCFI17:

sw $28,64($sp)

$LCFI18:

move $fp,$sp

$LCFI19:

sw $4,80($fp)

sw $5,84($fp)

.loc 1 47 0

$LBB3:

lw $2,80($fp)

beq $2,$0,$L26

lw $4,80($fp)

la $5,$LC13

la $25,strcmp

jal $31,$25

beq $2,$0,$L26

.loc 1 48 0

lw $4,80($fp)

la $5,$LC14

la $25,fopen

jal $31,$25

sw $2,24($fp)

.loc 1 50 0

lw $2,24($fp)

bne $2,$0,$L28

.loc 1 51 0

la $4,\_\_sF+176

la $5,$LC15

lw $6,80($fp)

la $25,fprintf

jal $31,$25

.loc 1 52 0

li $4,1 # 0x1

la $25,exit

jal $31,$25

$L26:

.loc 1 55 0

la $2,\_\_sF

sw $2,24($fp)

$L28:

.loc 1 58 0

lw $2,84($fp)

beq $2,$0,$L29

lw $4,84($fp)

la $5,$LC13

la $25,strcmp

jal $31,$25

beq $2,$0,$L29

.loc 1 59 0

lw $4,84($fp)

la $5,$LC16

la $25,fopen

jal $31,$25

sw $2,28($fp)

.loc 1 61 0

lw $2,28($fp)

bne $2,$0,$L32

.loc 1 62 0

la $4,\_\_sF+176

la $5,$LC17

lw $6,84($fp)

la $25,fprintf

jal $31,$25

.loc 1 63 0

li $4,2 # 0x2

la $25,exit

jal $31,$25

$L29:

.loc 1 67 0

la $2,\_\_sF+88

sw $2,28($fp)

.loc 1 72 0

$L32:

addu $2,$fp,32

lw $4,24($fp)

la $5,$LC18

move $6,$2

la $25,fscanf

jal $31,$25

move $3,$2

li $2,1 # 0x1

beq $3,$2,$L34

b $L33

$L34:

.loc 1 73 0

addu $2,$fp,32

move $4,$2

la $25,es\_palindromo

jal $31,$25

move $3,$2

li $2,1 # 0x1

bne $3,$2,$L32

.loc 1 74 0

addu $2,$fp,32

move $4,$2

lw $5,28($fp)

la $25,fputs

jal $31,$25

.loc 1 75 0

la $4,$LC19

lw $5,28($fp)

la $25,fputs

jal $31,$25

b $L32

$L33:

.loc 1 79 0

lw $2,24($fp)

beq $2,$0,$L36

lw $3,24($fp)

la $2,\_\_sF

beq $3,$2,$L36

.loc 1 80 0

lw $4,24($fp)

la $25,fclose

jal $31,$25

$L36:

.loc 1 83 0

lw $2,28($fp)

beq $2,$0,$L37

lw $3,28($fp)

la $2,\_\_sF+88

beq $3,$2,$L37

.loc 1 84 0

lw $4,28($fp)

la $25,fclose

jal $31,$25

$L37:

.loc 1 87 0

move $4,$0

la $25,exit

jal $31,$25

.loc 1 88 0

$LBE3:

.end handle

$LFE35:

.size handle, .-handle

.rdata

.align 2

$LC20:

.ascii "-h\000"

.align 2

$LC21:

.ascii "--help\000"

.align 2

$LC22:

.ascii "-V\000"

.align 2

$LC23:

.ascii "--version\000"

.align 2

$LC24:

.ascii "-i\000"

.align 2

$LC25:

.ascii "-o\000"

.text

.align 2

.globl main

$LFB37:

.loc 1 91 0

.ent main

main:

.frame $fp,40,$31 # vars= 0, regs= 3/0, args= 16, extra= 8

.mask 0xd0000000,-8

.fmask 0x00000000,0

.set noreorder

.cpload $25

.set reorder

subu $sp,$sp,40

.cprestore 16

$LCFI20:

sw $31,32($sp)

$LCFI21:

sw $fp,28($sp)

$LCFI22:

sw $28,24($sp)

$LCFI23:

move $fp,$sp

$LCFI24:

sw $4,40($fp)

sw $5,44($fp)

.loc 1 93 0

lw $3,40($fp)

li $2,1 # 0x1

bne $3,$2,$L39

.loc 1 94 0

move $4,$0

move $5,$0

la $25,handle

jal $31,$25

b $L40

$L39:

.loc 1 96 0

lw $3,40($fp)

li $2,2 # 0x2

bne $3,$2,$L41

.loc 1 97 0

lw $2,44($fp)

addu $2,$2,4

lw $4,0($2)

la $5,$LC20

la $25,strcmp

jal $31,$25

beq $2,$0,$L43

lw $2,44($fp)

addu $2,$2,4

lw $4,0($2)

la $5,$LC21

la $25,strcmp

jal $31,$25

bne $2,$0,$L42

$L43:

.loc 1 98 0

la $25,print\_usage

jal $31,$25

$L42:

.loc 1 101 0

lw $2,44($fp)

addu $2,$2,4

lw $4,0($2)

la $5,$LC22

la $25,strcmp

jal $31,$25

beq $2,$0,$L45

lw $2,44($fp)

addu $2,$2,4

lw $4,0($2)

la $5,$LC23

la $25,strcmp

jal $31,$25

bne $2,$0,$L40

$L45:

.loc 1 102 0

la $25,print\_version

jal $31,$25

b $L40

$L41:

.loc 1 105 0

lw $3,40($fp)

li $2,3 # 0x3

bne $3,$2,$L47

.loc 1 106 0

lw $2,44($fp)

addu $2,$2,4

lw $4,0($2)

la $5,$LC24

la $25,strcmp

jal $31,$25

bne $2,$0,$L48

.loc 1 107 0

lw $2,44($fp)

addu $2,$2,8

lw $4,0($2)

move $5,$0

la $25,handle

jal $31,$25

$L48:

.loc 1 110 0

lw $2,44($fp)

addu $2,$2,4

lw $4,0($2)

la $5,$LC25

la $25,strcmp

jal $31,$25

bne $2,$0,$L40

.loc 1 111 0

lw $2,44($fp)

addu $2,$2,8

move $4,$0

lw $5,0($2)

la $25,handle

jal $31,$25

b $L40

$L47:

.loc 1 114 0

lw $3,40($fp)

li $2,5 # 0x5

bne $3,$2,$L40

.loc 1 115 0

lw $2,44($fp)

addu $2,$2,4

lw $4,0($2)

la $5,$LC24

la $25,strcmp

jal $31,$25

bne $2,$0,$L52

lw $2,44($fp)

addu $2,$2,12

lw $4,0($2)

la $5,$LC25

la $25,strcmp

jal $31,$25

bne $2,$0,$L52

.loc 1 116 0

lw $2,44($fp)

addu $3,$2,8

lw $2,44($fp)

addu $2,$2,16

lw $4,0($3)

lw $5,0($2)

la $25,handle

jal $31,$25

$L52:

.loc 1 119 0

lw $2,44($fp)

addu $2,$2,4

lw $4,0($2)

la $5,$LC25

la $25,strcmp

jal $31,$25

bne $2,$0,$L40

lw $2,44($fp)

addu $2,$2,12

lw $4,0($2)

la $5,$LC24

la $25,strcmp

jal $31,$25

bne $2,$0,$L40

.loc 1 120 0

lw $2,44($fp)

addu $3,$2,16

lw $2,44($fp)

addu $2,$2,8

lw $4,0($3)

lw $5,0($2)

la $25,handle

jal $31,$25

$L40:

.loc 1 124 0

la $25,print\_usage

jal $31,$25

.loc 1 126 0

move $2,$0

.loc 1 127 0

move $sp,$fp

lw $31,32($sp)

lw $fp,28($sp)

addu $sp,$sp,40

j $31

.end main