

66.20 Organización de Computadoras

Trabajo Práctico 0:

Infraestructura básica

Burdet Rodrigo, *Padrón Nro. 93440*
rodrigoburdet@gmail.com

Colangelo Federico, *Padrón Nro. 89869*
federico.colangelo@semperti.com

Manzano Matias, *Padrón Nro. 83425*
matsebman@gmail.com

2do. Cuatrimestre de 2014
66.20 Organización de Computadoras
Facultad de Ingeniería, Universidad de Buenos Aires

7 de octubre de 2014

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 mas abajo.

2. Resumen

En el presente trabajo, se implementó un algoritmo que permite graficar los conjuntos de Mandelbrot dados ciertos parámetros para permitir centrarnos en una región en particular de dicho conjunto. El programa fue realizado en c 99 en un entorno de desarrollo linux. Compilado en Linux y en un ambiente emulado NetBSD.

3. Desarrollo

3.1. Paso 1: Configuración de Entorno de Desarrollo

El primer paso fue configurar el entorno de desarrollo, de acuerdo a la guía facilitada por la cátedra. Trabajamos con distribuciones Linux y con el GxEmul proporcionado por la cátedra, emulando un sistema NetBSD.

3.2. Paso 2: Implementación del programa

El programa debe ejecutarse por línea de comando y la salida del mismo dependerá del valor de los argumentos con los que se lo haya invocado.

3.2.1. Ingreso de parámetros

El formato para invocar al programa es el siguiente:

```
./tp0 [OPTIONS]
```

Los parámetros válidos que puede recibir el programa son los siguientes:

-o,	-output	(Parámetro obligatorio. Especifica archivo de salida, - para stdout)
-r,	-resolution	(Resolución de la imagen de salida).
-c,	-center	(Centro de la imagen).
-w,	-width	(Ancho del rectángulo a dibujar).
-H,	-height	(Alto del rectángulo a dibujar).
-v,	-version	(Muestra la versión).
-h,	-help	(Muestra la ayuda).

3.2.2. Interpretación de parámetros

Para parsear los parámetros se usó la librería de GNU getopt, en particular se usó getopt_long para permitir el pasaje de parámetros largos.

4. Compilación del programa

Para poder compilar el proyecto, se debe abrir una terminal Linux dentro del directorio donde se encuentra el código fuente escrito en C, y ejecutar el siguiente comando:

```
gcc -Wall -std=c99 main.c -o tp01
```

Esto generara un archivo ejecutable, llamado *tp0*.

¹Requiere tener instalado el compilador *GCC*

5. Compilación del programa en NetBSD

Para poder compilar el proyecto en NetBSD, se debe ejecutar el comando:

```
gcc -Wall -std=c99 -S -O0 main.c
```

6. Corridas de prueba y Mediciones

En las figuras que siguen a continuación se muestran los comandos utilizados para ejecutar el programa y se puede apreciar los resultados de las diferentes pruebas que realizamos.

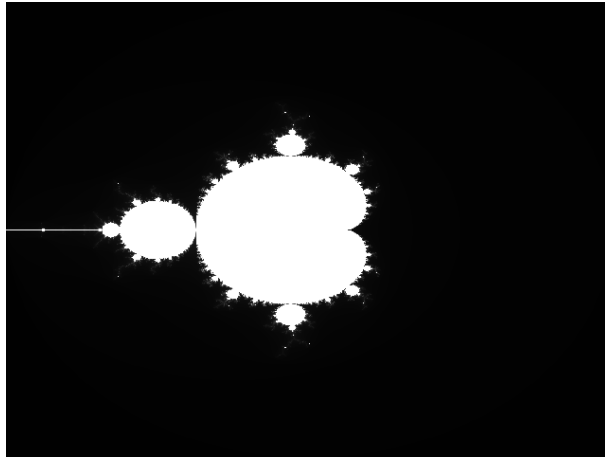


Figura 1: Llamada por defecto ./tp0 -o uno.pgm

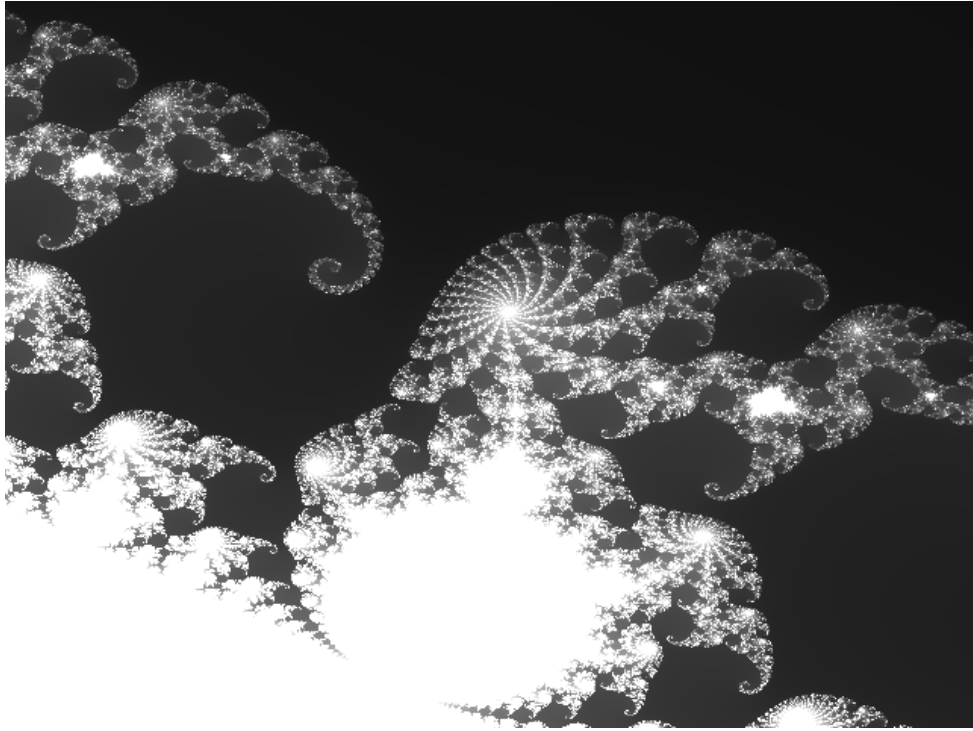


Figura 2: Llamada haciendo zoom sobre la región centrada en $(0.282, -0.01)$ con cuadrado de 0.005 de lado. `./tp0 -c +0.282-0.01i -w 0.005 -H 0.005 -o dos.pgm`

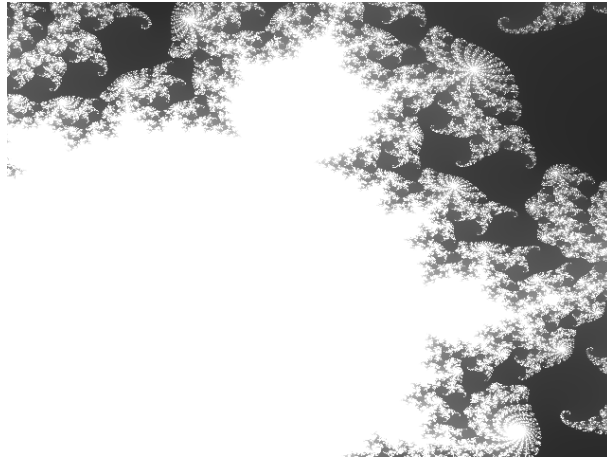


Figura 3: Llamada haciendo zoom sobre la región centrada en $(0.296, -0.02)$ con cuadrado de 0.003 de lado. `./tp0 -c +0.296-0.02i -w 0.003 -H 0.003 -o tres.pgm`

7. Conclusiones

Como se enuncia en el objetivo de este trabajo práctico, aprendimos a instalar y manejar el GxEmul, a realizar transferencias de archivos en Linux, así como también compilar y ejecutar programas en el NetBSD. Por otro lado, aprendimos a manejar y escribir informes en L^AT_EX. De este modo, estamos preparados para que en los próximos trabajos prácticos, nos aboquemos directamente al desarrollo de los mismos.

8. Código

8.1. main.c

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <getopt.h>
4 #include <string.h>
5
6 #define MAX_VAL 255
7 #define USAGE_ERROR -1
8 #define IO_ERROR -2
9 static const char VERSION[] = "1.0.1";
10
11 void usage(char* nombre);
12 void version(char* nombre);
13
14 /** Dada una resolucioin en formato WIDTHxHEIGHT devuelve un vector [WIDTH,HEIGHT] por ↵
15     defecto es [640,480]
16     * @param char* String con resolucioin del tipo WIDTHxHEIGHT
17     * @param int* vector de ints de resolucioin [WIDTH,HEIGHT]
18     * @return int 0 si se pudo parsear la resolucioin, -1 en otro caso
19     */
20 int parseResolution(char* res, int* resArray);
21
22 /** Dada una coordenada compleja en representacion binomica ( a + bi ) devuelve un vector ↵
23     [posX, posY], por defecto es [0,0]
24     * @param char* String con posicioin del centro de pantalla en forma binomica a+bi
25     * @param double* vector de ints de centro [posX,posY]
26     * @return int 0 si se pudo parsear la resolucioin, -1 en otro caso
27     */
28 int parseCenter(char* centerString, double* center);
29
30 /** Dadas las partes real e imaginara de un numero complejo devuelve 1 si el modulo es ↵
31     mayor que 2, 0 en caso contrario
32     * @param double parte real de un numero complejo
33     * @param double parte imaginaria de un numero complejo
34     * return int 1 si es el modulo es mayor que 2, 0 si es menor o igual
35     */
36 int stop(double re, double im);
37
38 /** Dados los parametros de interes calcula los conjuntos de mandelbrot
39     * @param int* vector de 2 enteros para especificar la resolucioin
40     * @param double* vector de 2 doubles para especificar la coordenada de origen
41     * @param double ancho del rectangulo a graficar
42     * @param double alto del rectangulo a graficar
43     * @param FILE* archivo donde escribe los conjuntos en formato pgm
44     * return int 0 si no hubo error
45     */
46 int print(int* res, double* center, double width, double height, FILE* file);
47
48
49 int main(int argc, char* argv[]){
50     int opt;
51     int res[2] = {640,480};
52     double width = 4;
53     double height = 4;
54     double center[2] = {0,0};
55     char* outDir = NULL;
56     FILE* file = stdout;
57     int parseCenterResult = 0;
58     int parseResolutionResult = 0;
```

```

59  int noArguments = 0;
60
61  static struct option long_options[] ={
62      {"version", no_argument, 0, 'V'},
63      {"help", no_argument, 0, 'h'},
64      {"resolution", required_argument, 0, 'r'},
65      {"center", required_argument, 0, 'c'},
66      {"width", required_argument, 0, 'w'},
67      {"height", required_argument, 0, 'H'},
68      {"output", required_argument, 0, 'o'},
69      {0, 0, 0, 0}
70  };
71
72  while ((opt = getopt_long (argc, argv, "r:c:w:H:o:Vh", long_options, NULL)) != -1){
73      switch(opt){
74          case 'r':
75              noArguments++;
76              parseResolutionResult = parseResolution(optarg, res);
77              break;
78          case 'w':
79              noArguments++;
80              width = atof(optarg);
81              break;
82          case 'H':
83              noArguments++;
84              height = atof(optarg);
85              break;
86          case 'o':
87              noArguments++;
88              outDir = optarg;
89              break;
90          case 'c':
91              noArguments++;
92              parseCenterResult = parseCenter(optarg, center);
93              break;
94          case 'h':
95              usage(argv[0]);
96              return 0;
97          case 'V':
98              version(argv[0]);
99              return 0;
100         case '?':
101             printf("Error\n");
102             return USAGE_ERROR;
103     }
104 }
105
106 if (noArguments == 0) {
107     usage(argv[0]);
108     return 0;
109 }
110
111 if (parseCenterResult == -1){
112     usage(argv[0]);
113     if (fprintf(stderr, "fatal: invalid center specification.\n") < 0)
114         return IO_ERROR;
115     return USAGE_ERROR;
116 }
117
118 if (parseResolutionResult == -1){
119     usage(argv[0]);
120     if (fprintf(stderr, "fatal: invalid resolution specification.\n") < 0)
121         return IO_ERROR;
122     return USAGE_ERROR;
123 }
124
125 if (outDir == NULL){
126     return USAGE_ERROR;
127 }

```

```

128
129 if (strcmp(outDir, "-") != 0){
130     file = fopen(outDir, "w");
131     if (!file){
132         usage(argv[0]);
133         if (fprintf(stderr, "fatal: cannot open output file.\n") < 1)
134             return IO_ERROR;
135         return USAGE_ERROR;
136     }
137 }
138
139 if (res[0] <= 0 || res[1] <= 0)
140     usage(argv[0]);
141 else{
142     if (fprintf(file, "P2\n%d\n%d\n%d\n", res[0], res[1], MAX_VAL) < 0)
143         return IO_ERROR;
144     print(res, center, width, height, file);
145 }
146
147 if (file != stdout)
148     if (fclose(file) < 0)
149         return IO_ERROR;
150
151 return 0;
152 }
153
154 int print(int* res, double* center, double width, double height, FILE* file){
155     double stepX = width / res[0];
156     double stepY = height / res[1];
157     int i, j, k;
158     double x, y;
159     double zx, zy;
160     double zx2, zy2;
161     x = center[0] + (-width + stepX) / 2;
162     //y = center[1] + (-height + stepY) / 2;
163     y = center[1] + (height - stepY) / 2;
164     for ( i = 1 ; i <= res[1] ; ++i ) {
165         //y = center[1] + height / 2 - i * stepY;
166         //printf("punto : %f , %f \n", x, y);
167         for ( j = 1 ; j <= res[0] ; ++j ) {
168             //x = center[0] - width / 2 + j * stepX;
169
170
171             zx = 0;
172             zy = 0;
173             zx2 = zx * zx;
174             zy2 = zy * zy;
175             for ( k = -1 ; k < MAX_VAL && !stop(zx, zy); k++ ) {
176                 zy = 2 * zx * zy + y;
177                 zx = zx2 - zy2 + x;
178                 zx2 = zx * zx;
179                 zy2 = zy * zy;
180                 //printf("calculo : %f , %f \n", zx, zy);
181             }
182             if (fprintf(file, "%d", k) < 0)
183                 return IO_ERROR;
184             if (fputc(' ', file) < 0)
185                 return IO_ERROR;
186             x = center[0] + (-width + 2 * j * stepX) / 2;
187         }
188         y = center[1] + (height - 2 * i * stepY) / 2;
189         if (fputc('\n', file) < 0)
190             return IO_ERROR;
191     }
192     return 0;
193 }
194
195 int parseResolution(char* str, int* res){
196     if (strlen(str) < 3)

```



```

197     return USAGE_ERROR;
198     char* aux ;
199     if ( ( aux = strtok(str,"x") ) != NULL){
200         res[0] = atoi(aux);
201         aux = strtok(NULL,"x");
202         res[1] = atoi(aux);
203     }else
204         return USAGE_ERROR;
205     return 0;
206 }
207
208 int parseCenter(char* centerString, double* center){
209     /*
210     if (str[strlen(str)-1] != 'i'){
211         return USAGE_ERROR;
212     }
213     char* aux = calloc(strlen(str)+1,sizeof(char));
214     unsigned int i = 0;
215     int pos;
216     int real = 1;
217     int signR;
218     int signI = 1;
219     if(str[0] == '+' || str[0] != '-'){
220         signR = 1 ;
221         i++;
222     }
223     if(str[0]=='-')
224         signR = -1;
225     for ( ; i< strlen(str) ; i++){
226         if (real){
227             aux[i-1] = str[i];
228             if (str[i] == '-' || str[i] == '+'){
229                 pos = i;
230                 center[0]= signR * atof(aux);
231                 real = 0;
232                 memset(aux,'\0',strlen(str));
233             }
234         }else{
235             if (str[pos] == '-')
236                 signI = -1;
237             aux[i-pos-1] = str[i];
238         }
239     }
240     aux[strlen(aux)-1] = '\0';
241     center[1] = signI *atof(aux);
242     return 0;
243
244     ///////////////////////////////////
245
246     if (centerString[strlen(centerString)-1] != 'i'){
247         return USAGE_ERROR;
248     }
249     center[0] = atof(strtok(centerString, "+i"));
250     center[1] = atof(strtok(NULL, "+i"));
251     return 0;
252     */
253     int realSign = 1;
254     int complexSign = 1;
255     unsigned int i = 1;
256     unsigned int found = 0;
257     if (centerString[strlen(centerString)-1] != 'i') {
258         return USAGE_ERROR;
259     }
260     if (centerString[0] == '-') {
261         realSign = -1;
262     }
263     while (found == 0 && i < strlen(centerString)) {
264         if (centerString[i] == '+') {
265             found = 1;

```

```

266     }
267     else if (centerString[i] == '-') {
268         found = 1;
269         complexSign = -1;
270     }
271     else {
272         i++;
273     }
274 }
275 if (found == 0) {
276     return USAGE_ERROR;
277 }
278 center[0] = atof(strtok(centerString, "+-i")) * realSign;
279 center[1] = atof(strtok(NULL, "+-i")) * complexSign;
280 return 0;
281 }
282
283
284 void version(char* name){
285     printf("%s version: %s: \n",name, VERSION);
286 }
287
288 int stop(double re, double im){
289     if ((re*re)+(im*im)>4)
290         return 1;
291     return 0;
292 }
293
294 void usage(char* name){
295     printf("Usage: \n");
296     printf("\t %s -h --help \n",name);
297     printf("\t %s -V --version \n",name);
298     printf("\t %s -r --resolution, permite cambiar la resolucio de la imagen generada. El ↵
        valor por defecto sera de 640x480 puntos \n",name);
299     printf("\t %s -c --center, para especificar el centro de la imagen, el punto central de ↵
        la porcion del plano complejo dibujada, expresado en forma binomica. El valor por ↵
        defecto sera 0 + 0i \n",name);
300     printf("\t %s -w --width, especifica el ancho del rectangulo que contiene la region del ↵
        plano complejo que estamos por dibujar. El valor por defecto sera 4 \n",name);
301     printf("\t %s -H --height, sirve, en forma similar, para especificar el alto del ↵
        rectangulo a dibujar. El valor por defecto sera 4 \n",name);
302     printf("\t %s -o --output, permite colocar la imagen de salida, (en formato PGM) en el ↵
        archivo pasado como argumento; o por salida estandar -cout- si el argumento es \"-\" ↵
        \n",name);
303     printf("Examples: \n");
304     printf("\t %s -o uno.pgm \n",name);
305     printf("\t %s -c +0.282-0.01i -w 0.005 -H 0.005 -o dos.pgm \n",name);
306     printf("\t %s -r 1x1 -o - \n",name);
307 }

```

8.2. main.s

```
1  .file 1 "main.c"
2  .section .mdebug.abi32
3  .previous
4  .abicalls
5  .rdata
6  .align 2
7  .type VERSION, @object
8  .size VERSION, 6
9  VERSION:
10  .ascii "1.0.1\000"
11  .align 2
12  $LC0:
13  .word 640
14  .word 480
15  .align 2
16  $LC2:
17  .ascii "version\000"
18  .align 2
19  $LC3:
20  .ascii "help\000"
21  .align 2
22  $LC4:
23  .ascii "resolution\000"
24  .align 2
25  $LC5:
26  .ascii "center\000"
27  .align 2
28  $LC6:
29  .ascii "width\000"
30  .align 2
31  $LC7:
32  .ascii "height\000"
33  .align 2
34  $LC8:
35  .ascii "output\000"
36  .data
37  .align 2
38  .type long_options.0, @object
39  .size long_options.0, 128
40  long_options.0:
41  .word $LC2
42  .word 0
43  .word 0
44  .word 86
45  .word $LC3
46  .word 0
47  .word 0
48  .word 104
49  .word $LC4
50  .word 1
51  .word 0
52  .word 114
53  .word $LC5
54  .word 1
55  .word 0
56  .word 99
57  .word $LC6
58  .word 1
59  .word 0
60  .word 119
61  .word $LC7
62  .word 1
63  .word 0
64  .word 72
```

```

65 .word $LC8
66 .word 1
67 .word 0
68 .word 111
69 .word 0
70 .word 0
71 .word 0
72 .word 0
73 .rdata
74 .align 2
75 $LC9:
76 .ascii "r:c:w:H:o:Vh\000"
77 .align 2
78 $LC10:
79 .ascii "Error\n\000"
80 .align 2
81 $LC11:
82 .ascii "fatal: invalid center specification.\n\000"
83 .align 2
84 $LC12:
85 .ascii "fatal: invalid resolution specification.\n\000"
86 .align 2
87 $LC13:
88 .ascii "-\000"
89 .align 2
90 $LC14:
91 .ascii "w\000"
92 .align 2
93 $LC15:
94 .ascii "fatal: cannot open output file.\n\000"
95 .align 2
96 $LC16:
97 .ascii "P2\n"
98 .ascii "%d\n"
99 .ascii "%d\n"
100 .ascii "%d\n\000"
101 .align 3
102 $LC1:
103 .word 0
104 .word 1074790400
105 .text
106 .align 2
107 .globl main
108 .ent main
109 main:
110 .frame $fp,136,$31 # vars= 80, regs= 3/0, args= 32, extra= 8
111 .mask 0xd0000000,-8
112 .fmask 0x00000000,0
113 .set noreorder
114 .cplod $25
115 .set reorder
116 subu $sp,$sp,136
117 .cpstore 32
118 sw $31,128($sp)
119 sw $fp,124($sp)
120 sw $28,120($sp)
121 move $fp,$sp
122 sw $4,136($fp)
123 sw $5,140($fp)
124 lw $2,$LC0
125 sw $2,48($fp)
126 lw $2,$LC0+4
127 sw $2,52($fp)
128 l.d $f0,$LC1
129 s.d $f0,56($fp)
130 l.d $f0,$LC1
131 s.d $f0,64($fp)
132 sw $0,72($fp)
133 sw $0,76($fp)

```

```

134     sw    $0,80($fp)
135     sw    $0,84($fp)
136     sw    $0,88($fp)
137     la    $2,___sF+88
138     sw    $2,92($fp)
139     sw    $0,96($fp)
140     sw    $0,100($fp)
141     sw    $0,104($fp)
142 $L18:
143     sw    $0,16($sp)
144     lw    $4,136($fp)
145     lw    $5,140($fp)
146     la    $6,$LC9
147     la    $7,long_options.0
148     la    $25,getopt_long
149     jal   $31,$25
150     sw    $2,40($fp)
151     lw    $3,40($fp)
152     li    $2,-1      # 0xffffffffffffffff
153     bne   $3,$2,$L20
154     b     $L19
155 $L20:
156     lw    $2,40($fp)
157     addu   $2,$2,-63
158     sw    $2,112($fp)
159     lw    $3,112($fp)
160     sltu   $2,$3,57
161     beq    $2,$0,$L18
162     lw    $2,112($fp)
163     sll    $3,$2,2
164     la    $2,$L30
165     addu   $2,$3,$2
166     lw    $2,0($2)
167     .cpadd $2
168     j     $2
169     .rdata
170     .align 2
171 $L30:
172     .gpword $L29
173     .gpword $L18
174     .gpword $L18
175     .gpword $L18
176     .gpword $L18
177     .gpword $L18
178     .gpword $L18
179     .gpword $L18
180     .gpword $L18
181     .gpword $L24
182     .gpword $L18
183     .gpword $L18
184     .gpword $L18
185     .gpword $L18
186     .gpword $L18
187     .gpword $L18
188     .gpword $L18
189     .gpword $L18
190     .gpword $L18
191     .gpword $L18
192     .gpword $L18
193     .gpword $L18
194     .gpword $L18
195     .gpword $L28
196     .gpword $L18
197     .gpword $L18
198     .gpword $L18
199     .gpword $L18
200     .gpword $L18
201     .gpword $L18
202     .gpword $L18

```

```

203 .gpword $L18
204 .gpword $L18
205 .gpword $L18
206 .gpword $L18
207 .gpword $L18
208 .gpword $L26
209 .gpword $L18
210 .gpword $L18
211 .gpword $L18
212 .gpword $L18
213 .gpword $L27
214 .gpword $L18
215 .gpword $L18
216 .gpword $L18
217 .gpword $L18
218 .gpword $L18
219 .gpword $L18
220 .gpword $L25
221 .gpword $L18
222 .gpword $L18
223 .gpword $L22
224 .gpword $L18
225 .gpword $L18
226 .gpword $L18
227 .gpword $L18
228 .gpword $L23
229 .text
230 $L22:
231 lw $2,104($fp)
232 addu $2,$2,1
233 sw $2,104($fp)
234 addu $2,$fp,48
235 lw $4,optarg
236 move $5,$2
237 la $25,parseResolution
238 jal $31,$25
239 sw $2,100($fp)
240 b $L18
241 $L23:
242 lw $2,104($fp)
243 addu $2,$2,1
244 sw $2,104($fp)
245 lw $4,optarg
246 la $25,atof
247 jal $31,$25
248 s.d $f0,56($fp)
249 b $L18
250 $L24:
251 lw $2,104($fp)
252 addu $2,$2,1
253 sw $2,104($fp)
254 lw $4,optarg
255 la $25,atof
256 jal $31,$25
257 s.d $f0,64($fp)
258 b $L18
259 $L25:
260 lw $2,104($fp)
261 addu $2,$2,1
262 sw $2,104($fp)
263 lw $2,optarg
264 sw $2,88($fp)
265 b $L18
266 $L26:
267 lw $2,104($fp)
268 addu $2,$2,1
269 sw $2,104($fp)
270 addu $2,$fp,72
271 lw $4,optarg

```

```

272     move    $5,$2
273     la      $25,parseCenter
274     jal     $31,$25
275     sw      $2,96($fp)
276     b       $L18
277 $L27:
278     lw      $2,140($fp)
279     lw      $4,0($2)
280     la      $25,usage
281     jal     $31,$25
282     sw      $0,108($fp)
283     b       $L17
284 $L28:
285     lw      $2,140($fp)
286     lw      $4,0($2)
287     la      $25,version
288     jal     $31,$25
289     sw      $0,108($fp)
290     b       $L17
291 $L29:
292     la      $4,$LC10
293     la      $25,printf
294     jal     $31,$25
295     li      $2,-1      # 0xffffffffffffffff
296     sw      $2,108($fp)
297     b       $L17
298 $L19:
299     lw      $2,104($fp)
300     bne     $2,$0,$L32
301     lw      $2,140($fp)
302     lw      $4,0($2)
303     la      $25,usage
304     jal     $31,$25
305     sw      $0,108($fp)
306     b       $L17
307 $L32:
308     lw      $3,96($fp)
309     li      $2,-1      # 0xffffffffffffffff
310     bne     $3,$2,$L33
311     lw      $2,140($fp)
312     lw      $4,0($2)
313     la      $25,usage
314     jal     $31,$25
315     la      $4,___sF+176
316     la      $5,$LC11
317     la      $25,fprintf
318     jal     $31,$25
319     bgez    $2,$L34
320     li      $3,-2      # 0xfffffffffffffffe
321     sw      $3,108($fp)
322     b       $L17
323 $L34:
324     li      $2,-1      # 0xffffffffffffffff
325     sw      $2,108($fp)
326     b       $L17
327 $L33:
328     lw      $3,100($fp)
329     li      $2,-1      # 0xffffffffffffffff
330     bne     $3,$2,$L35
331     lw      $2,140($fp)
332     lw      $4,0($2)
333     la      $25,usage
334     jal     $31,$25
335     la      $4,___sF+176
336     la      $5,$LC12
337     la      $25,fprintf
338     jal     $31,$25
339     bgez    $2,$L36
340     li      $3,-2      # 0xfffffffffffffffe

```

```

341     sw  $3,108($fp)
342     b  $L17
343 $L36:
344     li  $2,-1      # 0xffffffffffffffff
345     sw  $2,108($fp)
346     b  $L17
347 $L35:
348     lw  $2,88($fp)
349     bne $2,$0,$L37
350     li  $3,-1      # 0xffffffffffffffff
351     sw  $3,108($fp)
352     b  $L17
353 $L37:
354     lw  $4,88($fp)
355     la  $5,$LC13
356     la  $25,strcmp
357     jal $31,$25
358     beq $2,$0,$L38
359     lw  $4,88($fp)
360     la  $5,$LC14
361     la  $25,fopen
362     jal $31,$25
363     sw  $2,92($fp)
364     lw  $2,92($fp)
365     bne $2,$0,$L38
366     lw  $2,140($fp)
367     lw  $4,0($2)
368     la  $25,usage
369     jal $31,$25
370     la  $4,___sF+176
371     la  $5,$LC15
372     la  $25,fprintf
373     jal $31,$25
374     bgtz $2,$L40
375     li  $2,-2      # 0xfffffffffffffffe
376     sw  $2,108($fp)
377     b  $L17
378 $L40:
379     li  $3,-1      # 0xffffffffffffffff
380     sw  $3,108($fp)
381     b  $L17
382 $L38:
383     lw  $2,48($fp)
384     blez $2,$L42
385     lw  $2,52($fp)
386     blez $2,$L42
387     b  $L41
388 $L42:
389     lw  $2,140($fp)
390     lw  $4,0($2)
391     la  $25,usage
392     jal $31,$25
393     b  $L43
394 $L41:
395     li  $2,255      # 0xff
396     sw  $2,16($sp)
397     lw  $4,92($fp)
398     la  $5,$LC16
399     lw  $6,48($fp)
400     lw  $7,52($fp)
401     la  $25,fprintf
402     jal $31,$25
403     bgez $2,$L44
404     li  $2,-2      # 0xfffffffffffffffe
405     sw  $2,108($fp)
406     b  $L17
407 $L44:
408     addu $3,$fp,48
409     addu $5,$fp,72

```



```

410    l.d $f0,64($fp)
411    s.d $f0,16($sp)
412    lw  $2,92($fp)
413    sw  $2,24($sp)
414    move $4,$3
415    lw  $6,56($fp)
416    lw  $7,60($fp)
417    la  $25,print
418    jal $31,$25
419 $L43:
420    lw  $3,92($fp)
421    la  $2,___sF+88
422    beq $3,$2,$L45
423    lw  $4,92($fp)
424    la  $25,fclose
425    jal $31,$25
426    bgez $2,$L45
427    li  $3,-2      # 0xfffffffffffffffe
428    sw  $3,108($fp)
429    b $L17
430 $L45:
431    sw  $0,108($fp)
432 $L17:
433    lw  $2,108($fp)
434    move $sp,$fp
435    lw  $31,128($sp)
436    lw  $fp,124($sp)
437    addu $sp,$sp,136
438    j $31
439    .end main
440    .size main, .-main
441    .rdata
442    .align 2
443 $LC18:
444    .ascii "%d\000"
445    .align 3
446 $LC17:
447    .word 0
448    .word 1073741824
449    .text
450    .align 2
451    .globl print
452    .ent print
453 print:
454    .frame $fp,128,$31    # vars= 88, regs= 3/0, args= 16, extra= 8
455    .mask 0xd0000000,-8
456    .fmask 0x00000000,0
457    .set noreorder
458    .cpload $25
459    .set reorder
460    subu $sp,$sp,128
461    .cprestore 16
462    sw  $31,120($sp)
463    sw  $fp,116($sp)
464    sw  $28,112($sp)
465    move $fp,$sp
466    sw  $4,128($fp)
467    sw  $5,132($fp)
468    sw  $6,136($fp)
469    sw  $7,140($fp)
470    lw  $2,128($fp)
471    l.s $f0,0($2)
472    cvt.d.w $f2,$f0
473    l.d $f0,136($fp)
474    div.d $f0,$f0,$f2
475    s.d $f0,24($fp)
476    lw  $2,128($fp)
477    addu $2,$2,4
478    l.s $f0,0($2)

```

```

479     cvt.d.w $f2,$f0
480     l.d $f0,144($fp)
481     div.d $f0,$f0,$f2
482     s.d $f0,32($fp)
483     lw  $2,132($fp)
484     l.d $f2,24($fp)
485     l.d $f0,136($fp)
486     sub.d $f2,$f2,$f0
487     l.d $f0,$LC17
488     div.d $f2,$f2,$f0
489     l.d $f0,0($2)
490     add.d $f0,$f0,$f2
491     s.d $f0,56($fp)
492     lw  $2,132($fp)
493     addu $2,$2,8
494     l.d $f2,144($fp)
495     l.d $f0,32($fp)
496     sub.d $f2,$f2,$f0
497     l.d $f0,$LC17
498     div.d $f2,$f2,$f0
499     l.d $f0,0($2)
500     add.d $f0,$f0,$f2
501     s.d $f0,64($fp)
502     li  $2,1      # 0x1
503     sw  $2,40($fp)
504 $L48:
505     lw  $2,128($fp)
506     addu $2,$2,4
507     lw  $3,40($fp)
508     lw  $2,0($2)
509     slt $2,$2,$3
510     beq $2,$0,$L51
511     b $L49
512 $L51:
513     li  $2,1      # 0x1
514     sw  $2,44($fp)
515 $L52:
516     lw  $2,128($fp)
517     lw  $3,44($fp)
518     lw  $2,0($2)
519     slt $2,$2,$3
520     beq $2,$0,$L55
521     b $L53
522 $L55:
523     sw  $0,72($fp)
524     sw  $0,76($fp)
525     sw  $0,80($fp)
526     sw  $0,84($fp)
527     l.d $f2,72($fp)
528     l.d $f0,72($fp)
529     mul.d $f0,$f2,$f0
530     s.d $f0,88($fp)
531     l.d $f2,80($fp)
532     l.d $f0,80($fp)
533     mul.d $f0,$f2,$f0
534     s.d $f0,96($fp)
535     li  $2,-1     # 0xffffffffffffffff
536     sw  $2,48($fp)
537 $L56:
538     lw  $2,48($fp)
539     slt $2,$2,255
540     beq $2,$0,$L57
541     l.d $f12,72($fp)
542     l.d $f14,80($fp)
543     la  $25,stop
544     jal $31,$25
545     bne $2,$0,$L57
546     l.d $f0,72($fp)
547     add.d $f2,$f0,$f0

```

```

548    l.d $f0,80($fp)
549    mul.d $f2,$f2,$f0
550    l.d $f0,64($fp)
551    add.d $f0,$f2,$f0
552    s.d $f0,80($fp)
553    l.d $f2,88($fp)
554    l.d $f0,96($fp)
555    sub.d $f2,$f2,$f0
556    l.d $f0,56($fp)
557    add.d $f0,$f2,$f0
558    s.d $f0,72($fp)
559    l.d $f2,72($fp)
560    l.d $f0,72($fp)
561    mul.d $f0,$f2,$f0
562    s.d $f0,88($fp)
563    l.d $f2,80($fp)
564    l.d $f0,80($fp)
565    mul.d $f0,$f2,$f0
566    s.d $f0,96($fp)
567    lw $2,48($fp)
568    addu $2,$2,1
569    sw $2,48($fp)
570    b $L56
571 $L57:
572    lw $4,152($fp)
573    la $5,$LC18
574    lw $6,48($fp)
575    la $25,fprintf
576    jal $31,$25
577    bgez $2,$L61
578    li $2,-2 # 0xfffffffffffffffe
579    sw $2,104($fp)
580    b $L47
581 $L61:
582    li $4,32 # 0x20
583    lw $5,152($fp)
584    la $25,fputc
585    jal $31,$25
586    bgez $2,$L62
587    li $2,-2 # 0xfffffffffffffffe
588    sw $2,104($fp)
589    b $L47
590 $L62:
591    lw $3,132($fp)
592    lw $2,44($fp)
593    sll $2,$2,1
594    mtc1 $2,$f0
595    cvt.d.w $f2,$f0
596    l.d $f0,24($fp)
597    mul.d $f2,$f2,$f0
598    l.d $f0,136($fp)
599    sub.d $f2,$f2,$f0
600    l.d $f0,$LC17
601    div.d $f2,$f2,$f0
602    l.d $f0,0($3)
603    add.d $f0,$f0,$f2
604    s.d $f0,56($fp)
605    lw $2,44($fp)
606    addu $2,$2,1
607    sw $2,44($fp)
608    b $L52
609 $L53:
610    lw $2,132($fp)
611    addu $3,$2,8
612    lw $2,40($fp)
613    sll $2,$2,1
614    mtc1 $2,$f0
615    cvt.d.w $f2,$f0
616    l.d $f0,32($fp)

```

```

617 mul.d $f2,$f2,$f0
618 l.d $f0,144($fp)
619 sub.d $f2,$f0,$f2
620 l.d $f0,$LC17
621 div.d $f2,$f2,$f0
622 l.d $f0,0($3)
623 add.d $f0,$f0,$f2
624 s.d $f0,64($fp)
625 li $4,10 # 0xa
626 lw $5,152($fp)
627 la $25,fputc
628 jal $31,$25
629 bgez $2,$L50
630 li $2,-2 # 0xfffffffffffffffe
631 sw $2,104($fp)
632 b $L47
633 $L50:
634 lw $2,40($fp)
635 addu $2,$2,1
636 sw $2,40($fp)
637 b $L48
638 $L49:
639 sw $0,104($fp)
640 $L47:
641 lw $2,104($fp)
642 move $sp,$fp
643 lw $31,120($sp)
644 lw $fp,116($sp)
645 addu $sp,$sp,128
646 j $31
647 .end print
648 .size print, .-print
649 .rdata
650 .align 2
651 $LC19:
652 .ascii "\000"
653 .text
654 .align 2
655 .globl parseResolution
656 .ent parseResolution
657 parseResolution:
658 .frame $fp,48,$31 # vars= 8, regs= 3/0, args= 16, extra= 8
659 .mask 0xd0000000,-8
660 .fmask 0x00000000,0
661 .set noreorder
662 .cplod $25
663 .set reorder
664 subu $sp,$sp,48
665 .cpstore 16
666 sw $31,40($sp)
667 sw $fp,36($sp)
668 sw $28,32($sp)
669 move $fp,$sp
670 sw $4,48($fp)
671 sw $5,52($fp)
672 lw $4,48($fp)
673 la $25,strlen
674 jal $31,$25
675 sltu $2,$2,3
676 beq $2,$0,$L65
677 li $2,-1 # 0xfffffffffffffffff
678 sw $2,28($fp)
679 b $L64
680 $L65:
681 lw $4,48($fp)
682 la $5,$LC19
683 la $25,strtok
684 jal $31,$25
685 sw $2,24($fp)

```

```

686     lw  $2,24($fp)
687     beq $2,$0,$L66
688     lw  $4,24($fp)
689     la  $25,atoi
690     jal $31,$25
691     move $3,$2
692     lw  $2,52($fp)
693     sw  $3,0($2)
694     move $4,$0
695     la  $5,$LC19
696     la  $25,strtok
697     jal $31,$25
698     sw  $2,24($fp)
699     lw  $4,24($fp)
700     la  $25,atoi
701     jal $31,$25
702     move $3,$2
703     lw  $2,52($fp)
704     addu $2,$2,4
705     sw  $3,0($2)
706     b   $L67
707 $L66:
708     li  $2,-1      # 0xffffffffffffffff
709     sw  $2,28($fp)
710     b   $L64
711 $L67:
712     sw  $0,28($fp)
713 $L64:
714     lw  $2,28($fp)
715     move $sp,$fp
716     lw  $31,40($sp)
717     lw  $fp,36($sp)
718     addu $sp,$sp,48
719     j   $31
720     .end  parseResolution
721     .size parseResolution, .-parseResolution
722     .rdata
723     .align 2
724 $LC20:
725     .ascii "+-i\000"
726     .text
727     .align 2
728     .globl parseCenter
729     .ent  parseCenter
730 parseCenter:
731     .frame $fp,64,$31    # vars= 24, regs= 4/0, args= 16, extra= 8
732     .mask 0xd0010000,-4
733     .fmask 0x00000000,0
734     .set  noreorder
735     .cplod $25
736     .set  reorder
737     subu $sp,$sp,64
738     .cprestore 16
739     sw  $31,60($sp)
740     sw  $fp,56($sp)
741     sw  $28,52($sp)
742     sw  $16,48($sp)
743     move $fp,$sp
744     sw  $4,64($fp)
745     sw  $5,68($fp)
746     li  $2,1      # 0x1
747     sw  $2,24($fp)
748     li  $2,1      # 0x1
749     sw  $2,28($fp)
750     li  $2,1      # 0x1
751     sw  $2,32($fp)
752     sw  $0,36($fp)
753     lw  $4,64($fp)
754     la  $25,strlen

```

```

755     jal $31,$25
756     move $3,$2
757     lw $2,64($fp)
758     addu $2,$3,$2
759     addu $2,$2,-1
760     lb $3,0($2)
761     li $2,105      # 0x69
762     beq $3,$2,$L69
763     li $2,-1      # 0xffffffffffffffff
764     sw $2,40($fp)
765     b $L68
766 $L69:
767     lw $2,64($fp)
768     lb $3,0($2)
769     li $2,45      # 0x2d
770     bne $3,$2,$L70
771     li $2,-1      # 0xffffffffffffffff
772     sw $2,24($fp)
773 $L70:
774     .set noreorder
775     nop
776     .set reorder
777 $L71:
778     lw $2,36($fp)
779     bne $2,$0,$L72
780     lw $4,64($fp)
781     la $25,strlen
782     jal $31,$25
783     lw $3,32($fp)
784     sltu $2,$3,$2
785     bne $2,$0,$L73
786     b $L72
787 $L73:
788     lw $3,64($fp)
789     lw $2,32($fp)
790     addu $2,$3,$2
791     lb $3,0($2)
792     li $2,43      # 0x2b
793     bne $3,$2,$L75
794     li $2,1       # 0x1
795     sw $2,36($fp)
796     b $L71
797 $L75:
798     lw $3,64($fp)
799     lw $2,32($fp)
800     addu $2,$3,$2
801     lb $3,0($2)
802     li $2,45      # 0x2d
803     bne $3,$2,$L77
804     li $2,1       # 0x1
805     sw $2,36($fp)
806     li $2,-1      # 0xffffffffffffffff
807     sw $2,28($fp)
808     b $L71
809 $L77:
810     lw $2,32($fp)
811     addu $2,$2,1
812     sw $2,32($fp)
813     b $L71
814 $L72:
815     lw $2,36($fp)
816     bne $2,$0,$L79
817     li $2,-1      # 0xffffffffffffffff
818     sw $2,40($fp)
819     b $L68
820 $L79:
821     lw $16,68($fp)
822     lw $4,64($fp)
823     la $5,$LC20

```

```

824     la $25, strtok
825     jal $31, $25
826     move $4, $2
827     la $25, atof
828     jal $31, $25
829     mov.d $f2, $f0
830     l.s $f0, 24($fp)
831     cvt.d.w $f0, $f0
832     mul.d $f0, $f2, $f0
833     s.d $f0, 0($l6)
834     lw $2, 68($fp)
835     addu $l6, $2, 8
836     move $4, $0
837     la $5, $LC20
838     la $25, strtok
839     jal $31, $25
840     move $4, $2
841     la $25, atof
842     jal $31, $25
843     mov.d $f2, $f0
844     l.s $f0, 28($fp)
845     cvt.d.w $f0, $f0
846     mul.d $f0, $f2, $f0
847     s.d $f0, 0($l6)
848     sw $0, 40($fp)
849 $L68:
850     lw $2, 40($fp)
851     move $sp, $fp
852     lw $31, 60($sp)
853     lw $fp, 56($sp)
854     lw $l6, 48($sp)
855     addu $sp, $sp, 64
856     j $31
857 .end parseCenter
858 .size parseCenter, .-parseCenter
859 .rdata
860 .align 2
861 $LC21:
862 .ascii "%s version: %s: \n\000"
863 .text
864 .align 2
865 .globl version
866 .ent version
867 version:
868 .frame $fp, 40, $31 # vars= 0, regs= 3/0, args= 16, extra= 8
869 .mask 0xd0000000, -8
870 .fmask 0x00000000, 0
871 .set noreorder
872 .cpload $25
873 .set reorder
874 subu $sp, $sp, 40
875 .cpstore l6
876 sw $31, 32($sp)
877 sw $fp, 28($sp)
878 sw $28, 24($sp)
879 move $fp, $sp
880 sw $4, 40($fp)
881 la $4, $LC21
882 lw $5, 40($fp)
883 la $6, VERSION
884 la $25, printf
885 jal $31, $25
886 move $sp, $fp
887 lw $31, 32($sp)
888 lw $fp, 28($sp)
889 addu $sp, $sp, 40
890 j $31
891 .end version
892 .size version, .-version

```

```

893 .rdata
894 .align 3
895 $LC22:
896 .word 0
897 .word 1074790400
898 .text
899 .align 2
900 .globl stop
901 .ent stop
902 stop:
903 .frame $fp,24,$31 # vars= 8, regs= 2/0, args= 0, extra= 8
904 .mask 0x50000000,-4
905 .fmask 0x00000000,0
906 .set noreorder
907 .cpload $25
908 .set reorder
909 subu $sp,$sp,24
910 .cpstore 0
911 sw $fp,20($sp)
912 sw $28,16($sp)
913 move $fp,$sp
914 s.d $f12,24($fp)
915 s.d $f14,32($fp)
916 l.d $f2,24($fp)
917 l.d $f0,24($fp)
918 mul.d $f4,$f2,$f0
919 l.d $f2,32($fp)
920 l.d $f0,32($fp)
921 mul.d $f0,$f2,$f0
922 add.d $f2,$f4,$f0
923 l.d $f0,$LC22
924 c.lt.d $f0,$f2
925 bclt $L83
926 b $L82
927 $L83:
928 li $2,1 # 0x1
929 sw $2,8($fp)
930 b $L81
931 $L82:
932 sw $0,8($fp)
933 $L81:
934 lw $2,8($fp)
935 move $sp,$fp
936 lw $fp,20($sp)
937 addu $sp,$sp,24
938 j $31
939 .end stop
940 .size stop, .-stop
941 .rdata
942 .align 2
943 $LC23:
944 .ascii "Usage: \n\000"
945 .align 2
946 $LC24:
947 .ascii "\t %s -h --help \n\000"
948 .align 2
949 $LC25:
950 .ascii "\t %s -V --version \n\000"
951 .align 2
952 $LC26:
953 .ascii "\t %s -r --resolution, permite cambiar la resolucion de "
954 .ascii "la imagen generada. El valor por defecto sera de 640x480"
955 .ascii " puntos \n\000"
956 .align 2
957 $LC27:
958 .ascii "\t %s -c --center, para especificar el centro de la imag"
959 .ascii "en, el punto central de la porcion del plano complejo di"
960 .ascii "bujada, expresado en forma binomica. El valor por defect"
961 .ascii "o sera 0 + 0i \n\000"

```



```

962     .align 2
963 $LC28:
964     .ascii "\t %s -w --width, especifica el ancho del rectangulo que"
965     .ascii " contiene la region del plano complejo que estamos por d"
966     .ascii "ibujar. El valor por defecto sera 4 \n\000"
967     .align 2
968 $LC29:
969     .ascii "\t %s -H --height, sirve, en forma similar, para especificar"
970     .ascii " el alto del rectangulo a dibujar. El valor por defecto"
971     .ascii " sera 4 \n\000"
972     .align 2
973 $LC30:
974     .ascii "\t %s -o --output, permite colocar la imagen de salida, "
975     .ascii "(en formato PGM) en el archivo pasado como argumento; o "
976     .ascii "por salida estandar -cout- si el argumento es \"-\" \n\000"
977     .align 2
978 $LC31:
979     .ascii "Examples: \n\000"
980     .align 2
981 $LC32:
982     .ascii "\t %s -o uno.pgm \n\000"
983     .align 2
984 $LC33:
985     .ascii "\t %s -c +0.282-0.01i -w 0.005 -H 0.005 -o dos.pgm \n\000"
986     .align 2
987 $LC34:
988     .ascii "\t %s -r 1x1 -o - \n\000"
989     .text
990     .align 2
991     .globl usage
992     .ent usage
993 usage:
994     .frame $fp,40,$31 # vars= 0, regs= 3/0, args= 16, extra= 8
995     .mask 0xd0000000,-8
996     .fmask 0x00000000,0
997     .set noreorder
998     .cpload $25
999     .set reorder
1000     subu $sp,$sp,40
1001     .cpstore 16
1002     sw $31,32($sp)
1003     sw $fp,28($sp)
1004     sw $28,24($sp)
1005     move $fp,$sp
1006     sw $4,40($fp)
1007     la $4,$LC23
1008     la $25,printf
1009     jal $31,$25
1010     la $4,$LC24
1011     lw $5,40($fp)
1012     la $25,printf
1013     jal $31,$25
1014     la $4,$LC25
1015     lw $5,40($fp)
1016     la $25,printf
1017     jal $31,$25
1018     la $4,$LC26
1019     lw $5,40($fp)
1020     la $25,printf
1021     jal $31,$25
1022     la $4,$LC27
1023     lw $5,40($fp)
1024     la $25,printf
1025     jal $31,$25
1026     la $4,$LC28
1027     lw $5,40($fp)
1028     la $25,printf
1029     jal $31,$25
1030     la $4,$LC29

```

```

1031 lw $5,40($fp)
1032 la $25,printf
1033 jal $31,$25
1034 la $4,$LC30
1035 lw $5,40($fp)
1036 la $25,printf
1037 jal $31,$25
1038 la $4,$LC31
1039 la $25,printf
1040 jal $31,$25
1041 la $4,$LC32
1042 lw $5,40($fp)
1043 la $25,printf
1044 jal $31,$25
1045 la $4,$LC33
1046 lw $5,40($fp)
1047 la $25,printf
1048 jal $31,$25
1049 la $4,$LC34
1050 lw $5,40($fp)
1051 la $25,printf
1052 jal $31,$25
1053 move $sp,$fp
1054 lw $31,32($sp)
1055 lw $fp,28($sp)
1056 addu $sp,$sp,40
1057 j $31
1058 .end usage
1059 .size usage,.-usage
1060 .ident "GCC: (GNU) 3.3.3 (NetBSD nb3 20040520)"

```