Universidad De Buenos Aires

Facultad De Ingeniería

66.20 Organización De Computadoras

Trabajo Práctico 0

Integrantes:
Daniel Fernandez - 93083
Nicolas Ortoleva - 93196
Maximiliano Schultheis - 93285

8 de Abril de 2014

Índice

1. Diseño e implementación

2. Código Fuente

2.1. Código fuente C

```
#include <stdio.h>
#include <stdlib.h>
#include <getopt.h>
#include <string.h>
#include <stdbool.h>



bool encoderActivo = true;
bool decoderActivo = false;

FILE* finput = NULL;
FILE* foutput = NULL;



char* vecHexa [] = {"0","1","2","3","4","5","6","7","8","9","A","B","C","D","E","F"};


static struct option long_options[] = {
         {"version", no_argument, 0, 'v'},
     {"help", no_argument, 0, 'h'},
     {"input", required_argument, 0, 'i'},
     {"output", required_argument, 0, 'o'},
     {"action", required_argument, 0, 'a'},
     {0, 0, 0, 0}
};


char* encoder( int numInt){

        int highNible = numInt & 0xf0;
        int lowNible = numInt & 0x0f;
        int primerNum = highNible >> 4;
        int segundoNum = lowNible;

        char* primerChar = vecHexa[primerNum];
        char* segundoChar = vecHexa[segundoNum];
        char* valorHexa = malloc( sizeof(char)*3 );

        strcpy(valorHexa,primerChar);
        strcat(valorHexa,segundoChar);

        return valorHexa;
}


int correrReferencia ( int numInt ){
        if( numInt > 47 && numInt < 58)
                return numInt - 48 ;

        else if( numInt > 64 && numInt < 71)
                return numInt - 55 ;

```

```
55          else if( numInt > 96 && numInt < 103)
56                  return numInt - 87 ;
57          else {
58                  fprintf(stderr,"Contiene caracteres que no pertenecen al codigo Hexa\
                        n");
59                  exit(1);
60          }
61
62  }
63
64  char decoder ( int numPri, int numSeg){
65
66          int valor1 = correrReferencia( numPri );
67          int valor2 = correrReferencia( numSeg );
68
69          int highNible = valor1 << 4;
70          highNible =  highNible & 0xf0;
71          valor2 = valor2 & 0x0f;
72
73          char caracter = highNible | valor2;
74
75          return caracter;
76
77
78  }
79
80  void procesarArchivos(FILE* finput, FILE* foutput){
81
82          char* string;
83          char c;
84          int caracter2;
85          int caracter = fgetc(finput);
86
87          while (caracter != EOF){
88                  if(encoderActivo){
89                          string = encoder(caracter);
90                          if(foutput != NULL) fputs(string , foutput);
91                          else printf("%s",string);
92                          free(string);
93                  }
94                  else{
95
96                          caracter2 = fgetc(finput);
97                          c = decoder(caracter,caracter2);
98                          if(foutput != NULL) fputc(c,foutput);
99                          else printf("%c",c);
100
101
102                 }
103                 caracter = fgetc(finput);
104         }
105
106     fclose(finput);
107     if(foutput != NULL) fclose(foutput);
108 }
109
110 void EntradaEncoderStandar(){
111
112         int c = getchar();
113         while( c != EOF){
114                 char* string = encoder(c);
```

```
115                 printf("%s",string);
116                 free(string);
117                 c = getchar();
118         }
119 }
120
121 void EntradaDecoderStandar(){
122         int c = getchar();
123         int c2 = getchar();
124         while( c != EOF){
125                 char string = decoder(c,c2);
126                 if( foutput != NULL ) fputc(string,foutput);
127                 else printf("%c",string);
128                 c = getchar();
129                 c2 = getchar();
130         }
131
132 }
133
134
135 void comprobarAction(char* optarg){
136         if( strcmp ( optarg, "encode" ) == 0 ){
137                 encoderActivo = true;
138                 decoderActivo = false;
139         }
140         if ( strcmp ( optarg, "decode" ) == 0 ){
141                 encoderActivo = false;
142                 decoderActivo = true;
143         }
144
145 }
146
147
148 void opciones( int argc , char** argv ){
149
150         int option_index = 0;
151         int option = getopt_long ( argc, argv, "vhi:o:a:", long_options, &
                option_index);
152         if( option == -1 ){
153                 EntradaEncoderStandar();
154                 return;
155         }
156         while ( option != -1 ){
157
158                 switch ( option ){
159                         case 'v':
160                                         printf("66.20-Organizacion de Computadoras TP
                                             Version 0.0\n");
161                                         break;
162                         case 'h':
163                                         printf(" -v, --version, Shows the version of
                                            TP. \n");
164                                         printf(" -h, --help , Show help \n");
165                                         printf(" -i, --input, Location of the input
                                            file\n");
166                                         printf(" -o, --output, Location of the output
                                             file\n");
167                                         printf(" -a, --action, Program action: encode
                                             (default) or decode \n");
168                                         break;
169
```

```
170                             case 'i':
171                                             finput = fopen(optarg,"r");
172                                             if(finput == NULL ){
173                                                     fprintf(stderr,"Error al abrir el
                                                            archivo input %s\n",optarg);
174                                                     exit(1);
175                                     }
176                                     break;
177                             case 'o':
178                                             foutput = fopen(optarg, "w");
179                                             if (foutput == NULL){
180                                                     fprintf(stderr,"Error al abrir el
                                                            archivo output %s \n",optarg);
181                                                     exit(1);
182                                             }
183                                                     break;
184                             case 'a':
185                                             comprobarAction(optarg);
186                                             break;
187
188                             default:
189                                             break;
190                     }
191
192
193                     option = getopt_long ( argc, argv, "vhi:o:a:", long_options, &
                            option_index);
194
195         }
196
197
198     }
199
200
201     void controlarOpciones(){
202
203              if (finput != NULL){
204                     procesarArchivos(finput,foutput);
205             }
206             else{
207                     if(encoderActivo) EntradaEncoderStandar();
208                     else EntradaDecoderStandar();
209             }
210
211     }
212
213     int main (int argc, char** argv){
214             opciones( argc , argv);
215             controlarOpciones();
216             printf("\nSe completo con exito la operacion\n");
217             return 0;
218
219     }
```

2.2. Código assembly MIPS

3. Conclusiones