Juan Carlos Fernández Caballero, jfcaballero@uco.es

Ejemplos de guía para usar expresiones regulares simples

Para el manejo de expresiones regulares, la GNU C Library incluye una serie de funciones (regcomp, regerror, regexec y regfree) que permiten comprobar si una cadena empareja una expresión regular. Estas funciones están incluidas en "regex.h" (Por favor, consulte los enlaces que se le proporcionan en la práctica):

Pasos:

1)
/* Compilar la expresión regular */
int *regcomp* (regex_t *restrict compiled, const char *restrict pattern, int cflags)

//*regcomp* returns 0 if it succeeds in compiling the regular expression; otherwise, it returns a nonzero error code.

Ejemplo:

```
char *regexValue = NULL;  // A rellenar con getopt_long(),
                          // será la palabra a buscar dentro de una frase

regex_t regex; //Tipo de dato para almacenar una expresion regular compilada

int value;  //Para recoger la devolucion de la funcion de compilacion
...

//Tratamiento expresion regular basica, lea la documentación de los enlaces de la practica
value = regcomp(&regex, regexValue, 0);
```

2)
/* Comprobar la expresión regular sobre la cadena pasada como argumento - Matching */
int *regexec* (const regex_t *restrict compiled, const char *restrict string, size_t nmatch, regmatch_t matchptr[restrict], int eflags)

//*regexec* returns 0 if the regular expression matches; otherwise, it returns a nonzero value.

3)
/* Capturar el tipo de error producido en caso de que falle la funcion de matching */
size_t *regerror*(int errcode, const regex_t *restrict preg, char *restrict errbuf, size_t errbuf_size);

4)
/* Liberar la memoria utilizada por las expresiones regulares */
void *regfree* (regex_t *compiled)

Ejemplo de los pasos:

/*Matching para expresion regular basica, lea la documentacion de los enlaces de la practica.
"buffer" será la cedena de texto del cliente recibida por la cola correspondiente*/

```
char *regexValue = NULL;
regex_t regex;
int comp, value;
char buffer[MAX_SIZE];
...
```

```
comp = regcomp(&regex, regexValue, 0);   //PASO 1
if( comp!=0 )
{
    printf("Error al compilar la expresion regular\n"):
    exit(0);
}
```

```
value = regexec(&regex, buffer, 0, NULL, 0);     //PASO 2
if( value==0 )
{
     strcpy(buffer,"Empareja");
    //Se enviará por la cola correspondiente
}
else if(value == REG_NOMATCH)
{
    strcpy(buffer,"No Empareja");
    //Se enviará por la cola correspondiente
}
else //Se produjo un error a la hora comprobar la expresion  //PASO 3
{
   regerror(value, &regex, msgbuf, sizeof(msgbuf));
   fprintf(stderr, "Falló el matching de la expresión regular: %s\n", msgbuf);
   /*El servidor debería mandar por la cola de emparejamientos algun mensaje al cliente para que
  tanto el servidor como el cliente finalizarán y cerraran estructuras abiertas antes de su fin*/
     ...
}
```

```
/* Liberar la expresión regular utilizada.  Puede incluir llamada en vez de aquí, en una función
   especial que sirva para terminar la ejecución del servidor.  Esa función puede ejecutarse cuando
   se reciban mensajes de capturas de señales por parte del cliente,  o por problemas de este tipo
   relativos a fallos de funciones Posix implementadas por Glibc.  */

regfree(&regex);  //PASO 4
```