Metodología de la Programación

Grado en Ingeniería Informática

Curso 2022/2023

Durán Obregón, Alejandro
Ruíz Miñán, José Antonio
Ruíz del Pino, Alejandro
Vázquez Vera, Luis Enrique

Proyecto Modularidad

ESI-SHARE

Índice general

1. Introducción

Este es un ejemplo de plantilla para la generación de la documentación asociada al desarrollo de un programa. Se puede usar como guía, pero puede sufrir todas las modificaciones que considere oportunas para mejorar su documento.

Es necesario incluir una portada, dispone de cuatro ejemplos. Puede comprobar el resultado de cada uno de ellos en los ficheros PDF. En el fichero .tex está por defecto el ejemplo de Portada1.

Se debe añadir un índice. Puede observar en el código cómo se puede incluir en la sección destinada a tal fin.

A lo largo del documento también puede surgir la necesidad de añadir una imagen. Puede ver cómo hacerlo en el código de las portadas.

El código fuente del módulo *Escribir* es el siguiente:

```
#include "acceso.h"

//Prototipo: void acceso();
//Precondicin: Tener inicializada la variable "
   numUsuarios", con el nmero de usuarios mximos del
   fichero, y la estructura "usuario", con datos ledos
//desde el fichero "usuarios.txt", y haber introducido
   el nmero correspondiente en el men.
//Postcondicin: Autenticacin de las credenciales
   introducidas, si coinciden con alguna de la base de
   datos, pues se accede al programa.
//Si el usuario es "usuario", acceder al men de
   usuario, y si es "administrador", acceder al men
   de admin.

void acceso()
{
    int i, j=0, k=0, encontrado=0, encontrado2=0,
       encontrado3=3, preg=0;
    char usua[6], contra[9], c;

    printf("Introduzca sus credenciales de acceso:\
       nUsuario:\n");
    pregunta(usua, 6);

    for(k=0; k<numUsuarios&&encontrado2==0; k++)
```

```
{
    if(strcmp(usua, usuario[k].usuario)==0) //
       Comprueba si el usuario est en la base de
       datos.
    {
        encontrado2=1;
    }
}

if(encontrado2==1) //Si el usuario est en la base
   de datos, salta aqu.
{
    while(encontrado3>0&&encontrado==0) //Sistema
       de 3 intentos para introducir la contrasea
        correctamente.
    {
        printf("Contrasena:\n");
        fflush(stdin);
        while ((c=getch())!='\r'&&j<8)
        {
            if (c=='\b'&&j>0)
            {
                j--;
                printf("\b \b");
            }
            else if (c != '\b')
            {
                contra[j++] = c;
                printf("*");
            }
        }
        contra[j] = '\0';

        for(i=0; i<numUsuarios||encontrado!=0; i++)
            //Comprueba si el usuario y la
           contrasea son correctos.
        {
            if(strcmp(usuario[i].usuario,usua)==0)
            {
                if(strcmp(usuario[i].contrasena,
                   contra)==0)
```

```
{
    if(strcmp(usuario[i].perfil,"
        usuario")==0) //Si el
        usuario, tiene perfil de "
        usuario", salta al men de
        usuario.
    {
        encontrado=1;
        menuUsuario(i);
    }
    else if(strcmp(usuario[i].
        perfil,"administrador")==0)
        //Si el usuario, tiene
        perfil de "administrador",
        salta al men de admin.
    {
        encontrado=1;
        menuAdmin(i);
    }
}
else
{
    printf("\nLa contrasena
        introducida es incorrecta.\n
        "); //Si la contrasea no
        es correcta, se imprime esta
         frase.
    encontrado3--; //Se resta 1
        intento, hasta llegar a 0.
    if(encontrado3==0)
    {
        printf(" Intentos agotados!
            :(\n");
        system("PAUSE");
    }
    else
    {
        *contra=NULL;
        j=0;
        printf("Queda(n) %i
            intentos.\n",
```

```
                                    encontrado3); //Imprime
                                    un contador de intentos,
                                     para saber cuntos
                                    quedan.
                                system("PAUSE");
                                system("cls");
                                printf("Introduzca sus
                                    credenciales de acceso:\
                                    nUsuario:\n");
                                printf("%s\n", usua);
                            }
                        }
                    }
                }
            }
        }
        else
        {
            printf("Usuario no encontrado en nuestra base
                de datos.\n"); //Si el usuario no est en la
                 base de datos.
            system("PAUSE");
        }
}
```

2. Introducción

Este es un ejemplo de plantilla para la generación de la documentación asociada al desarrollo de un programa. Se puede usar como guía, pero puede sufrir todas las modificaciones que considere oportunas para mejorar su documento.

Es necesario incluir una portada, dispone de cuatro ejemplos. Puede comprobar el resultado de cada uno de ellos en los ficheros PDF. En el fichero .tex está por defecto el ejemplo de Portada1.

Se debe añadir un índice. Puede observar en el código cómo se puede incluir en la sección destinada a tal fin.

A lo largo del documento también puede surgir la necesidad de añadir una imagen. Puede ver cómo hacerlo en el código de las portadas.

El código fuente del módulo *Escribir* es el siguiente:

```
#include "acceso.h"

//Prototipo: void acceso();
//Precondicin: Tener inicializada la variable "
   numUsuarios", con el nmero de usuarios mximos del
   fichero, y la estructura "usuario", con datos ledos
//desde el fichero "usuarios.txt", y haber introducido
   el nmero correspondiente en el men.
//Postcondicin: Autenticacin de las credenciales
   introducidas, si coinciden con alguna de la base de
   datos, pues se accede al programa.
//Si el usuario es "usuario", acceder al men de
   usuario, y si es "administrador", acceder al men
   de admin.

void acceso()
{
    int i, j=0, k=0, encontrado=0, encontrado2=0,
       encontrado3=3, preg=0;
    char usua[6], contra[9], c;

    printf("Introduzca sus credenciales de acceso:\
       nUsuario:\n");
    pregunta(usua, 6);

    for(k=0; k<numUsuarios&&encontrado2==0; k++)
```

```
{
    if(strcmp(usua, usuario[k].usuario)==0) //
       Comprueba si el usuario est en la base de
       datos.
    {
        encontrado2=1;
    }
}

if(encontrado2==1) //Si el usuario est en la base
   de datos, salta aqu.
{
    while(encontrado3>0&&encontrado==0) //Sistema
       de 3 intentos para introducir la contrasea
        correctamente.
    {
        printf("Contrasena:\n");
        fflush(stdin);
        while ((c=getch())!='\r'&&j<8)
        {
            if (c=='\b'&&j>0)
            {
                j--;
                printf("\b \b");
            }
            else if (c != '\b')
            {
                contra[j++] = c;
                printf("*");
            }
        }
        contra[j] = '\0';

        for(i=0; i<numUsuarios||encontrado!=0; i++)
            //Comprueba si el usuario y la
           contrasea son correctos.
        {
            if(strcmp(usuario[i].usuario,usua)==0)
            {
                if(strcmp(usuario[i].contrasena,
                   contra)==0)
```

```
{
    if(strcmp(usuario[i].perfil,"
        usuario")==0) //Si el
        usuario, tiene perfil de "
        usuario", salta al men de
        usuario.
    {
        encontrado=1;
        menuUsuario(i);
    }
    else if(strcmp(usuario[i].
        perfil,"administrador")==0)
        //Si el usuario, tiene
        perfil de "administrador",
        salta al men de admin.
    {
        encontrado=1;
        menuAdmin(i);
    }
}
else
{
    printf("\nLa contrasena
        introducida es incorrecta.\n
        "); //Si la contrasea no
        es correcta, se imprime esta
         frase.
    encontrado3--; //Se resta 1
        intento, hasta llegar a 0.
    if(encontrado3==0)
    {
        printf(" Intentos agotados!
            :(\n");
        system("PAUSE");
    }
    else
    {
        *contra=NULL;
        j=0;
        printf("Queda(n) %i
            intentos.\n",
```

```
                                encontrado3); //Imprime
                                un contador de intentos,
                                 para saber cuntos
                                quedan.
                            system("PAUSE");
                            system("cls");
                            printf("Introduzca sus
                                credenciales de acceso:\
                                nUsuario:\n");
                            printf("%s\n", usua);
                        }
                    }
                }
            }
        }
    }
    else
    {
        printf("Usuario no encontrado en nuestra base
            de datos.\n"); //Si el usuario no est en la
             base de datos.
        system("PAUSE");
    }
}
```

3. Documentación de usuario

Documentación relacionada con las funciones del sistema pero no se hace referencia a nada relativo a su implementación. Está orientado a las personas que van a usar el sistema no a los que lo van a mantener.

3.1. Descripción funcional

- Debe describir de forma simple el propósito del sistema.
- En esta sección se debe incluir una breve descripción funcional sobre lo que hace y no hace el sistema.
- Se pueden incluir pequeños ejemplos para dar una visión general de las características del sistema.

3.2. Tecnología

Un listado con una breve descripción de las tecnologías usadas en el proyecto: aplicaciones, frameworks, librerías, arquitecturas,... Por ejemplo, se puede añadir información sobre el tipo de programa, desarrollador, última versión estable y enlace a la página web oficial, entre otros detalles. También se pueden añadir los iconos representativos de las herramientas.

Si se necesita hacer referencia a algunos documentos se incluyen las citas que sean necesarias (**??**).

3.3. Manual de instalación

Una breve descripción que explique como instalar el sistema y adecuarlo a las configuraciones particulares del *hardware*, indicando las características mínimas de *hardware* requeridas para el funcionamiento del programa.

3.4. Acceso al sistema

Debe contener una explicación sencilla de cómo iniciarse en el sistema y cómo salir del mismo. Así como la forma de salir de los problemas más habituales.

3.5. Manual de referencia

En esta sección se describe con detalle las ventajas que ofrece el sistema a los usuarios y cómo se pueden obtener mediante su uso. Debe describir, de una manera más completa y formal, el uso del sistema, así como las situaciones de error y los informes generados.

3.6. Guía del operador

Si el programa requiriera de operador entonces se incluiría esta sección. Debe describir cómo se debe de reaccionar antes determinadas situaciones que pueden darse durante el uso del sistema.

4. Documentación de sistema

Documentación referida a los aspectos del análisis, diseño, implementación y prueba del *software*, así como la implantación del mismo. En general, debe hacer referencia al sistema, ya que está orientada a los programadores que van a realizar el mantenimiento. Debe estar muy bien estructurada, desde lo más general a los más interno.

4.1. Especificación del sistema

Esta sección debe describir el análisis y la especificación de requisitos. Cómo se descompone el problema en distintos subproblemas y los módulos asociados a cada uno de ellos, acompañados de su especificación. También debe incluir el plan de desarrollo del *software*.

4.2. Módulos

Debe contener una descripción de cada módulo, su funcionalidad y la interfaz. También se podría añadir código fuente.

4.3. Plan de prueba

Debe describir todo el plan de pueba que se ha llevado a cabo. Se distinguen las pruebas realizadas a cada módulo y las pruebas de integración de los distintos módulos probándolo como un todo. Por último, también se debe incluir la descripción del plan de pruebas de aceptación.

Prueba de los módulos

Incluir toda la batería de pruebas realizadas. Pruebas de caja blanca y pruebas de caja negra.

Prueba de integración

Incluir toda la batería de pruebas realizadas. Pruebas de caja blanca y pruebas de caja negra.

Plan de pruebas de aceptación

Estas pruebas deben diseñarse con el usuario del sistema. Deben describir las pruebas que son necesarias pasar para que el sistema sea aceptado por el usuario final.

Si en algún punto del documento se quiere hacer referencia a algún documento es necesario incluir la cita donde corresponda, podemos tomar como ejemplo **?**.

5. Documentación del código fuente

En este punto se puede incluir toda la documentación interna que puede ser generada con Doxygen.

La documentación interna del programa, como se ha comentado anteriormente, comprende los comentarios del programa. Durante la fase de implementación es necesario comentar adecuadamente cada una de las partes del programa. Estos comentarios se incluyen en el código fuente con el objetivo de aclarar qué es lo que hace el fragmento de código, para explicar sus elementos.

Esta documentación ayuda a hacer más comprensible el código fuente a los programadores que deban trabajar en él, facilitando las tareas de mantenimiento. No sólo proporciona una mayor legibilidad en la actualización y modificación del código, sino también en la depuración del mismo.