Metodología de la Programación
Grado en Ingenería Informática

Liga Fantástica

¸Laura Guerrero, Victor Moreno, Alberto Valderas y Carlos Vidal

Trabajo sobre modularidad
Escuela Superior de Ingeniería, Abril 2021

Índice general

Capítulo 1

Introducción

1.1. Descripción funcional

En este programa, cada participante configurará sus propias plantillas de fútbol con los jugadores disponibles de equipos reales de primera división. Al final de cada jornada, el usuario asignará una valoración, que se sumará a la del resto de futbolistas de la plantilla para conformar una puntuación general de la misma. Esta puntuación general de la plantilla será la empleada para designar el usuario ganador de la liga.

1.2. Descomposición del problema

El problema principal se descompone en diferentes subproblemas para facilitar la producción del programa completo. En primer lugar encontramos la carga de los archivos en memoria principal para poder trabajar con ellos de una forma más eficiente, del cual se encarga el módulo de "Carga". Posteriormente encontramos la administración de usuarios del sistema y la redirección a las distintas opciones dependiendo del tipo de usuario de lo que se encarga "Usuario". A continuación encontramos la edición de los parámetros del juego de lo que se ocupa "Configuración". Antes de acabar encontraríamos la edición y administración de las plantillas por parte del usuario, el juego en sí, de lo que se encargaría el módulo "Equipo". Finalmente para actualizar todos estos datos y guardarlos sin que se pierda nada el módulo de "Carga" se encargaría de volcar estos datos a los archivos.

1.3. Descripción de cada módulo implementado en C

1.3.1. Módulo Usuario - Victor Moreno

En el módulo usuario se ha implementado una función que permite registrarse al individuo, y una vez iniciado sesión el programa te informa sobre el tipo de usuario y los permisos que este conlleva. En el caso de ser usuario normal, el programa hace uso del módulo equipos. En el caso de ser cronista, puedes editar la puntuación de los jugadores, así como ver la puntuación actual de los mismos. Por último en el caso de ser administrador, el programa hace una llamada al módulo configuración.

1.3.2. Módulo Equipo - Alberto Valderas

En este módulo, se ha desarrollado la información perteneciente a las plantillas, con su respectiva evaluación global. Para ello se permite a los usuarios crear, editar o borrar plantillas, así como ver las mismas, su puntuación o incluso el usuario creador de estas.

1.3.3. Módulo Configuración - Laura Guerrero

En este módulo, se ha implementado una función mediante la cual se le pregunta al administrador qué valor desea editar y si quiere editar más de un valor.

1.3.4. Módulo Carga - Carlos Vidal

Este módulo se encarga de cargar los archivos de texto en las estructuras de cada módulo para que estas puedan ser usadas posteriormente.

Capítulo 2

Instalación y ejecución del programa.

2.1. Proceso de instalación.

Para la instalación del programa deberemos descargar el archivo comprimido y posteriormente descomprimirlo para poder hacer uso de los archivos de su interior.

2.2. Proceso de ejecución del programa.

Para ejecutarlo, necesitaremos abrir el proyecto en un compilador y compilar la función main, la cual se encargará de compilar el resto de funciones en el momento que las necesite.

Capítulo 3

Pruebas de Software

3.1. Casos de prueba

En este apartado estudiaremos el funcionamiento del programa a distintos de niveles, de menos a mayor complejidad. Empezando por las funciones individuales de cada módulo para finalizar con el funcionamiento del programa completo.

3.1.1. Prueba de funciones individuales

carga.c

1. (void limpiar) Funciona correctamente.

2. (int contar lineas) Funciona correctamente.

3. (void carga-equipos) Funciona correctamente.

4. (void carga-configuracion) Funciona correctamente.

5. (void carga-usuario) Funciona correctamente.

6. (void carga-futbolistas) Funciona correctamente.

7. (void carga-plantillas) Funciona correctamente.

8. (void descarga-equipos) Funciona correctamente.

9. (void descarga-configuracion) Funciona correctamente.

10. (void descarga-usuario) Funciona correctamente.

11. (void descarga-plantillas) Funciona correctamente.

12. (void descarga-futbolistas) Funciona correctamente.

13. (void descarga-equipos) Funciona correctamente.

configuracion.c

1. (void config) Funciona correctamente.

equipo.c

1. (int numero-plantillas) Funciona correctamente.

2. (void equipos) Funciona correctamente.

usuario.c

1. (void usuario) Funciona correctamente.

3.1.2. Prueba de los módulos

1. Módulo carga: Funciona correctamente.

2. Módulo configuración: Funciona correctamente.

3. Módulo equipos: Funciona correctamente.

4. Módulo usuario: Funciona correctamente.

3.1.3. Prueba de integración entre módulos

El paso de los valores así como la llamada a las funciones necesarias se realiza correctamente.

3.1.4. Prueba de programa

El programa compila pero no se ejecuta correctamente, imposibilitando el jugar de forma satisfactoria al juego "Liga Fantástica".

3.2. Prueba de ruta básica

3.2.1. Módulo Carga

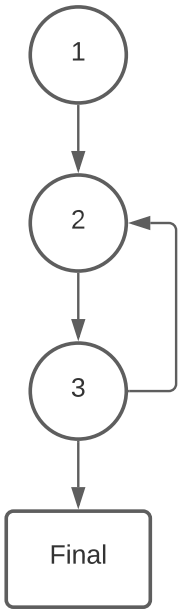


Figura 3.1: Diagrama de flujo de limpiar.c

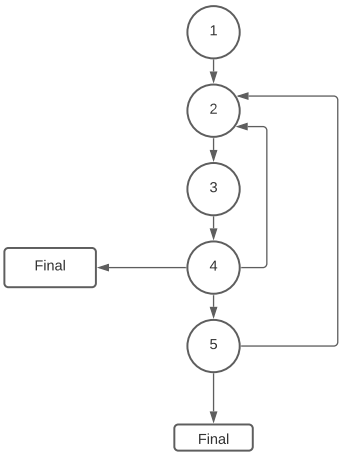
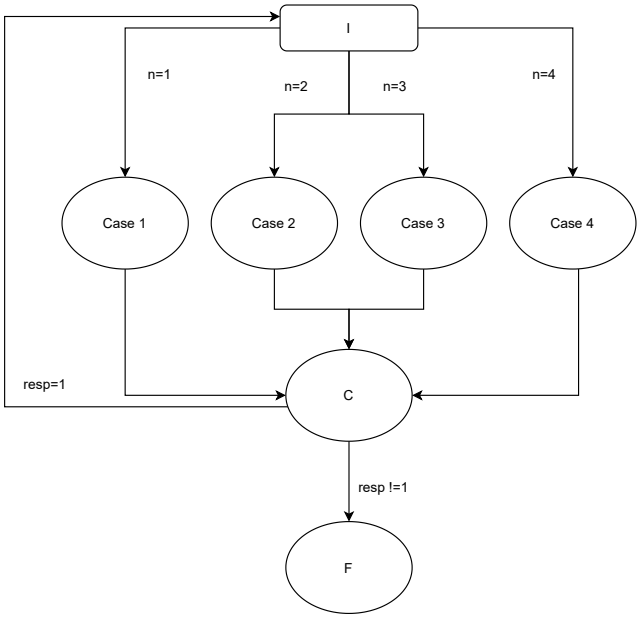


Figura 3.2: Diagrama de flujo de contarlineas.c

Cuadro 3.1: Cálculo de prueba de ruta básica

Camino	Entrada	Salida
X	X	X

3.2.2. Módulo Configuración



(a) Diagrama de flujo

```
void config (){
    int n, precio_min, precio_max, n_max_plant, n_max_jugadores, resp;
    configuraciones configuracion_tot[3];
    do{
        printf ("Que desea editar? \n\n 1-precio minimo | 2-precio maximo | 3- Numero max de plantillas | 4-
Numero max de jugadores\n");
        scanf ("%d", &n);
        switch (n) {
        case 1:
            printf ("Cual desea que sea el nuevo precio minimo?\n");
            scanf("%d", &precio_min);
            configuracion_tot [n-1].valor_campo= precio_min;
            break;
        case 2:
            printf ("Cual desea que sea el nuevo precio maximo?\n");
            scanf ("%d", &precio_max);
            configuracion_tot [n-1].valor_campo= precio_max;
            break;
        case 3:
            printf ("Cual desea que sea el nuevo numero maximo de plantillas?\n");
            scanf ("%d", &n_max_plant);
            configuracion_tot [n-1].valor_campo=n_max_plant;
            break;
        case 4:
            printf ("Cual desea que sea el nuevo numero maximo de jugadores?\n");
            scanf ("%d", &n_max_jugadores);
            configuracion_tot [n-1].valor_campo=n_max_jugadores;
            break;
        }
        printf ("Desea editar algo mas?\n \t (1) Si/ (2) No\n");
        scanf ("%d", &resp);
    } while (resp==1);
}
```

(b) Código del módulo configuracion.c

Figura 3.3: Código y diagrama del módulo configuracion.c

Cuadro 3.2: Cálculo de prueba de ruta básica con complejidad ciclomática de 5 [V(G)]=5

Camino	Entrada	Salida
I, 1 C, F	n=1, resp≠ 1	F
I, 2 C, F	n=2, resp≠ 1	F
I, 3 C, F	n=3, resp≠ 1	F
I, 4 C, F	n=4, resp≠ 1	F
I, 1 C, I, 2, C, F	n=1, resp=1, n=2, resp≠ 1	F

3.2.3. Módulo Equipo

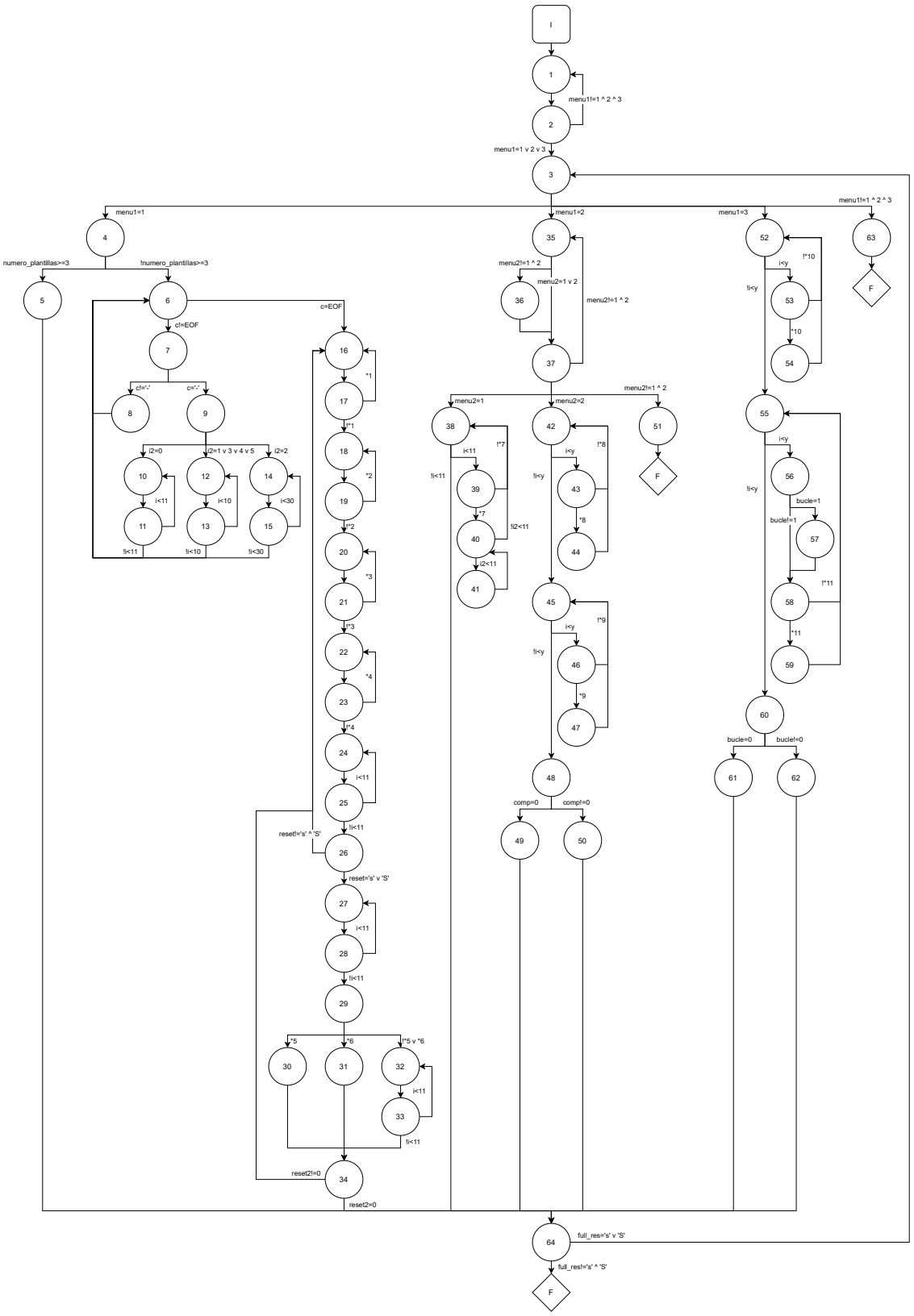


Figura 3.4: Diagrama de flujo.

```
void equipo(int x)
{
	int menu1, menu2, i=1, i2=0, reset2, y, borrar, bucle=0, editar, comp=0;
	char c, reset, full_res;

	do
	{
		do
		{
			printf("Crear nueva plantilla (1) | Ver/editar mis plantillas (2) | Borrar plantilla (3)\n");
			fflush(stdin);
			scanf("%i", &menu1);
			if(menu1!=1&&menu1!=2&&menu1!=3)
				printf("Comando no valido\n");
		}while(menu1!=1&&menu1!=2&&menu1!=3);

		switch (menu1)
		{
		case '1':
			if(numero_plantillas(x)>=3)
				printf("Ya tienes el numero maximo de plantillas");
			else
			{
				printf("Como quieres que se llame la nueva plantilla? (sin espacios, max 21 caracteres) --> ");

				fichero=fopen("plantillas .txt","r");
				y=contar_lineas(fichero);
				fclose(fichero);

				total_plantillas=(plantilla *)realloc(total_plantillas, (y+1)*sizeof(plantilla));

				fflush(stdin);
				scanf("%s", &total_plantillas.nombre_plantilla);

				total_plantillas.identificador_plantilla=y+1;
				total_plantillas.identificador_prop_plantilla=x;

				i=1;
				i2=0;

				fichero=fopen("jugadores.txt","r");
```

```
				printf("Codigo  Equipo  Nombre          Valor(M)  Puntos  Puntos totales\n\n");
				while((c = getc(fichero)) != EOF)
				{
					if(c!='-')
					{
						putchar(c);
						i++;
					}
					else
					{
						switch(i2)
						{
						case 0:
							for(i; i<11; i++)
								printf(" ");
							i=0;
							break;

						case 1:
						case 3:
						case 4:
						case 5:
							for(i; i<10; i++)
								printf(" ");
							i=0;
							break;

						case 2:
							for(i; i<30; i++)
								printf(" ");
							i=0;
							break;
						}

						i2++;
						if(i2==6)
							i2=0;
					}
				}

				fclose(fichero);
```

```
do
{
    do
    {
        printf("\n\n");

        do
        {
16          printf("Introduzca el codigo de su portero (numeros 1, 5, 9 ,13...) --> ");
            fflush(stdin);
            scanf("%i", &total_plantillas.jugadores[0]);
            if(total_plantillas.jugadores[0]%4!=1||total_plantillas.jugadores[0]>80)
                printf("Codigo no existente o no es portero.\n");
17      }while(total_plantillas.jugadores[0]%4!=1||total_plantillas.jugadores[0]>80);

        do
        {
18          printf("Introduzca el codigo de sus cuatro defensas (separados por espacios)
(numeros 2, 6, 10 ,14...) --> ");
            fflush(stdin);
            scanf("%i %i %i %i", &j[1], &j[2], &j[3], &j[4]);

        if(total_plantillas.jugadores[1]%4!=2||total_plantillas.jugadores[2]%4!=2||total_plantillas.jugadores[3]%4!=2||total_plantillas.jugadores[4]%4!=2||t
otal_plantillas.jugadores[1]>80||total_plantillas.jugadores[2]>80||total_plantillas.jugadores[3]>80||total_plantillas.jugadores[4]>80)
                printf("Codigo/s no existente/s o no es/son defensa/s.\n");

19  }while(total_plantillas.jugadores[1]%4!=2||total_plantillas.jugadores[2]%4!=2||total_plantillas.jugadores[3]%4!=2||total_plantillas.jugadores[4]%4!
=2||total_plantillas.jugadores[1]>80||total_plantillas.jugadores[2]>80||total_plantillas.jugadores[3]>80||total_plantillas.jugadores[4]>80);

        do
        {
20          printf("Introduzca el codigo de sus tres centrocampistas (separados por espacios)
(numeros 3, 7, 11 ,15...) --> ");
            fflush(stdin);
            scanf("%i %i %i", &j[5], &j[6], &j[7]);

        if(total_plantillas.jugadores[5]%4!=3||total_plantillas.jugadores[6]%4!=3||total_plantillas.jugadores[7]%4!=3||total_plantillas.jugadores[5]>80||tot
al_plantillas.jugadores[6]>80||total_plantillas.jugadores[7]>80)
                printf("Codigo/s no existente/s o no es/son centrocampista/s.\n");

21  }while(total_plantillas.jugadores[5]%4!=3||total_plantillas.jugadores[6]%4!=3||total_plantillas.jugadores[7]%4!=3||total_plantillas.jugadores[5]>80|
|total_plantillas.jugadores[6]>80||total_plantillas.jugadores[7]>80);

        do
        {
22          printf("Introduzca el codigo de sus tres delanteros (separados por espacios)
(numeros 4, 8, 12 ,16...) --> ");
            fflush(stdin);
            scanf("%i %i %i", &j[8], &j[9], &j[10]);

        if(total_plantillas.jugadores[8]%4!=0||total_plantillas.jugadores[9]%4!=0||total_plantillas.jugadores[10]%4!=0||total_plantillas.jugadores[8]>80||to
tal_plantillas.jugadores[9]>80||total_plantillas.jugadores[10]>80)
                printf("Codigo/s no existente/s o no es/son delantero/s.\n");

23  }while(total_plantillas.jugadores[8]%4!=0||total_plantillas.jugadores[9]%4!=0||total_plantillas.jugadores[10]%4!=0||total_plantillas.jugadores[8]>80
||total_plantillas.jugadores[9]>80||total_plantillas.jugadores[10]>80);

        printf("\n");

24      for(i=0; i<11; i++)
25          printf("%i ", total_plantillas.jugadores[i]);

        printf("\n\n\nSon correctos estos codigos? [s/n] --> ");
        fflush(stdin);
        scanf("%c", &reset);
26  }while(reset!='s'&&reset!='S');

    i=0;

27  for(i=0; i<11; i++)
28      total_plantillas.presupuesto+=total_futbolistas[total_plantillas.jugadores[i]].precio_futbolista;

29  if(total_plantillas.presupuesto>toda_config[1].valor_campo)
    {
30      printf("Su plantilla se pasa del presupuesto por %i millones. Por favor, vuelva a elegir sus
jugadores.", total_plantillas.presupuesto-toda_config[1].valor_campo);
        reset2=1;
    }
    else if(total_plantillas.presupuesto<toda_config[0].valor_campo)
    {
31      printf("Su plantilla esta por debajo del presupuesto por %i millones. Por favor, vuelva a elegir
sus jugadores.", toda_config[0].valor_campo-total_plantillas.presupuesto);
        reset2=1;
    }
    else
    {
32      for(i=0; i<11; i++)
33      total_plantillas.puntuacion_plantilla+=total_futbolistas[total_plantillas.jugadores[i]].valoracion_total;
        printf("Su plantilla ha sido creada correctamente.");
```

```
                reset2=0;
            }
34      }while(reset2!=0);
        }
        break;

case 2:
        do
        {
            printf("Ver plantillas (1) | Editar plantilla (2)\n");
            fflush(stdin);
            scanf("%i", &menu2);

35          if(menu2!=1&&menu2!=2)
36              printf("Comando no valido");
37      }while(menu1!=1&&menu2!=2);

        switch(menu2)
        {
        case 1:
            fichero=fopen("plantillas.txt","r");
            y=contar_lineas(fichero);
            fclose(fichero);

38          for(i=0; i<y; i++)
39              if(total_plantillas[i].identificador_prop_plantilla==x)
                {
                    printf("Codigo:%i\nNombre:%s\nValor total:%i Millones\nPuntuacion total:%i
puntos\nJugadores:\n");
40                  for(i2=0; i2<11; i2++);
                    {
41                      printf("\t%s - %s - Puntuacion individual: %i\n",
total_futbolistas[total_plantillas.jugadores[i2]].nombre_futbolista,
total_equipos[total_futbolistas[total_plantillas.jugadores[i2]].identificador_equipo_pertenece-1].nombre_equipo,
total_futbolistas[total_plantillas.jugadores[i2]].valoracion_total);
                    }
                    printf("\n");
                }

            break;

        case 2:
            printf("Que plantilla quiere editar? (Escriba el codigo de plantilla)\n");
            fichero=fopen("plantillas.txt","r");
            y=contar_lineas(fichero);
            fclose(fichero);

42          for(i=0; i<y; i++)
43              if(total_plantillas[i].identificador_prop_plantilla==x)
44                  printf("%i %s\n", total_plantillas[i].identificador_plantilla, total_plantillas[i].nombre_plantilla);

            fflush(stdin);
            scanf("%i", &editar);

45          for(i=0; i<y; i++)
            {
46              if(total_plantillas[i].identificador_prop_plantilla==x&&total_plantillas[i].identificador_plantilla==editar)
                {
47                  comp=1;
                    printf("Como quiere que pase a llamarse esta plantilla? (sin espacios, max 21 caracteres) --> ");
                    limpiar(total_plantillas[i].nombre_plantilla);
                    fflush(stdin);
                    scanf("%s", &total_plantillas[i].nombre_plantilla);
                }
            }
48          if(comp==0)
49              printf("Plantilla inexistente o no es suya.\n");
            else
            {
50              comp=0;
                printf("Plantilla renombrada correctamente.\n");
            }
            break;

51      default:
            printf("Ha ocurrido un error. El programa se cerrara.");
            exit(1);
        }
        break;

case 3:
        printf("Que plantilla quiere borrar? (Escriba el codigo de plantilla)\n");

        fichero=fopen("plantillas.txt","r");
```

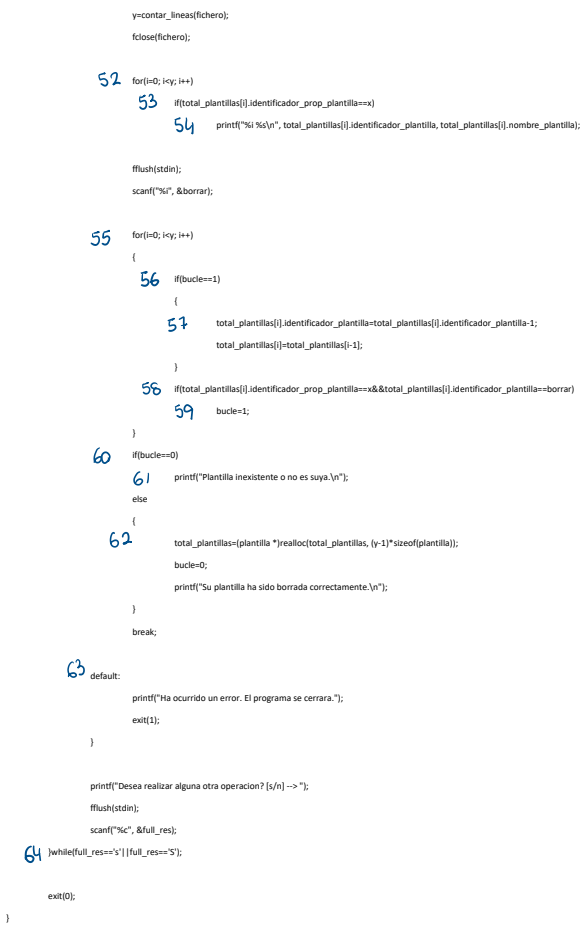


Figura 3.8: Leyenda de lo que representa cada nodo.

Nº de cada condición	Que valores representa
1	total_plantillas.jugadores[0]%4!=1 v total_plantillas.jugadores[0]>80
2	total_plantillas.jugadores[1]%4!=2 v total_plantillas.jugadores[2]%4!=2 v total_plantillas.jugadores[3]%4!=2 v total_plantillas.jugadores[4]%4!=2 v total_plantillas.jugadores[1]>80 v total_plantillas.jugadores[2]>80 v total_plantillas.jugadores[3]>80 v total_plantillas.jugadores[4]>80
3	total_plantillas.jugadores[5]%4!=3 v total_plantillas.jugadores[6]%4!=3 v total_plantillas.jugadores[7]%4!=3 v total_plantillas.jugadores[5]>80 v total_plantillas.jugadores[6]>80 v total_plantillas.jugadores[7]>80
4	total_plantillas.jugadores[8]%4!=0 v total_plantillas.jugadores[9]%4!=0 v total_plantillas.jugadores[10]%4!=0 v total_plantillas.jugadores[8]>80 v total_plantillas.jugadores[9]>80 v total_plantillas.jugadores[10]>80
5	total_plantillas.presupuesto>toda_config[1].valor_campo
6	total_plantillas.presupuesto<toda_config[0].valor_campo
7	total_plantillas[i].identificador_prop_plantilla=x
8	total_plantillas[i].identificador_prop_plantilla=x
9	total_plantillas[i].identificador_prop_plantilla=x ^ total_plantillas[i].identificador_plantilla=editar
10	total_plantillas[i].identificador_prop_plantilla=x
11	total_plantillas[i].identificador_prop_plantilla=x ^ total_plantillas[i].identificador_plantilla=borrar

Figura 3.9: Leyenda de lo que representa cada condición del diagrama.

Cuadro 3.3: Cálculo de prueba de ruta básica inviable de calcular por la complejidad del módulo

Camino	Entrada	Salida
X	X	X

3.2.4. Módulo Usuario

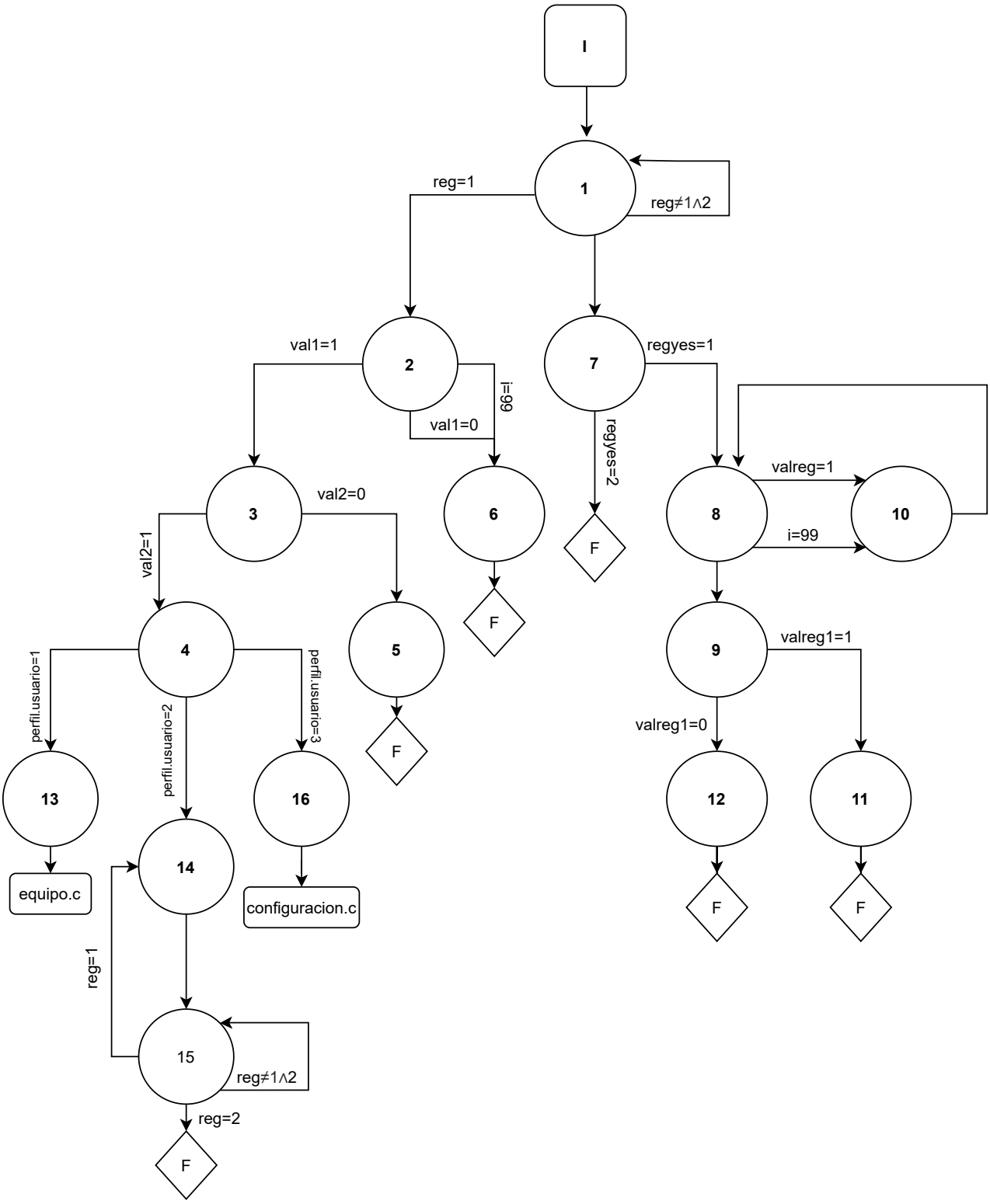


Figura 3.10: Diagrama de flujo.

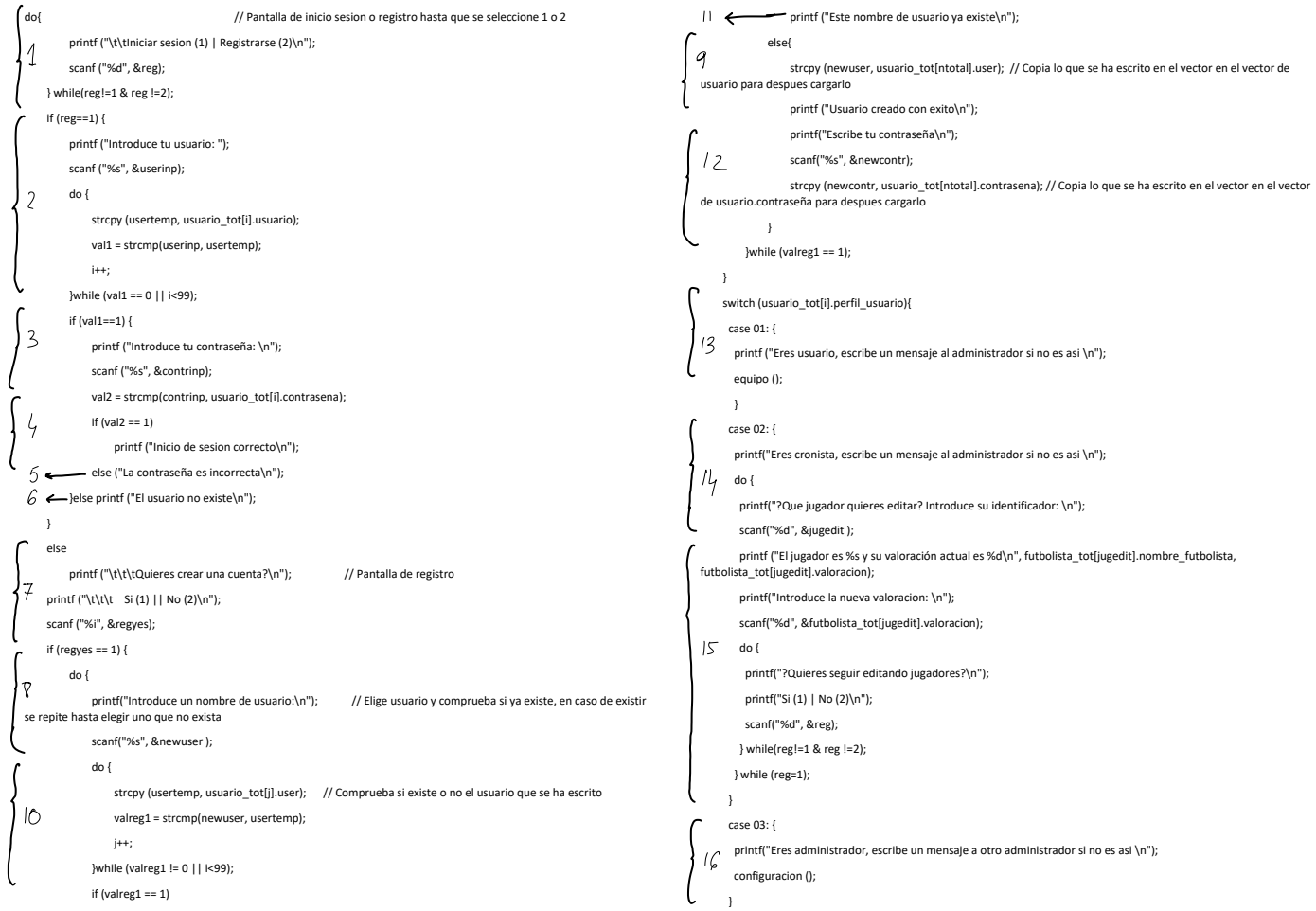


Figura 3.11: Leyenda de lo que representa cada nodo.

Cuadro 3.4: Cálculo de prueba de ruta básica con complejidad ciclomática de 12 [V(G)]=12

Camino	Entrada	Salida
1, 2, 3, 4, 14, 15, F	reg=1, val1=1, val2=1, perfil.usuario=2, reg=1, reg=2	F
1, 2, 3, 4, 14, 15, 14, 15, F	reg=1, val1=1, val2=1, perfil.usuario=2, reg$\neq$ (1&2), $reg = 2$	F
1, 2, 3, 4, 14, 15, 15, F	reg=1, val1=1, val2=1, perfil.usuario=2, reg=2	F
1, 2, 3, 4, 13, equipo.c	reg=1, val1=1, val2=1, perfil.usuario=1	equipo.c
1, 2, 3, 4, 16, configuracion.c	reg=1, val1=1, val2=1, perfil.usuario=3	configuracion.c
1, 2, 3, 5, F	reg=1, val1=1, val2=0	F
1, 2, 6, F	reg=1, val1=0 $\parallel i = 99$	F
1, 8, 9, 12, F	reg=2, regyes=1, valreg=0, valreg1=0	F
1, 8, 9, 11, F	reg=2, regyes=1, valreg=0, valreg1=1	F
1, 8, 10, 9, 12, F	reg=2, regyes=1, valreg=1 $\parallel i = 99, valreg1 = 0$	F
1, 7, F	reg=2, regyes$\neq$ 1	F
1, 1, 2, 3, 5, F	reg$\neq$ (1&2), $val1 = 1, val2 = 0$	F