Corte de Control Archivos

ALGORITMOS Y ESTRUCTURAS DE DATOS II

PRÁCTICO G3: AQUINO – BURGHARDT – PRINCICH

Archivo

En términos computacionales **es una colección de datos** que tiene un nombre y se guardan en dispositivos de almacenamiento secundario, los cuales pueden ser magnéticos, ópticos, electrónicos, etc.

Datos: Básicamente se refieren a los testimonios individuales relacionados con hechos, ya sean características de ciertos objetos de estudio o condiciones particulares de situaciones dadas. Los elementos individuales de los archivos se llaman datos o campos.

Registro: Es el conjunto completo de datos relacionados pertenecientes a una entrada. P. ejem. Un almacén puede retener los datos de sus productos en registros.

Archivos

ORGANIZACIÓN

- ✔Secuencial
- ✔Directo o aleatoria
- ✔Organización secuencial indexada

MODOS DE ACCESOS

- ✔Secuencial
- ✔Directo

Tipos de Archivos

DE TEXTO

- ✔Los datos están guardados en cadenas de texto.
- ✔El acceso es secuencial.

BINARIOS

- ✔Los datos están guardados en bits.
- ✔El accesos es directo.

Operaciones sobre archivos

Apertura de archivo:

La función fopen devuelve un puntero a un archivo.

FILE * **fopen**(nombre archivo, modo);

Modo	Significado
r	Abre un archivo de texto para lectura
w	Crea un archivo de texto para escritura
a	Abre un archivo de texto para añadir
rb	Abre un archivo binario para lectura
wb	Crea un archivo binario para escritura
ab	Abre un archivo binario para añadir

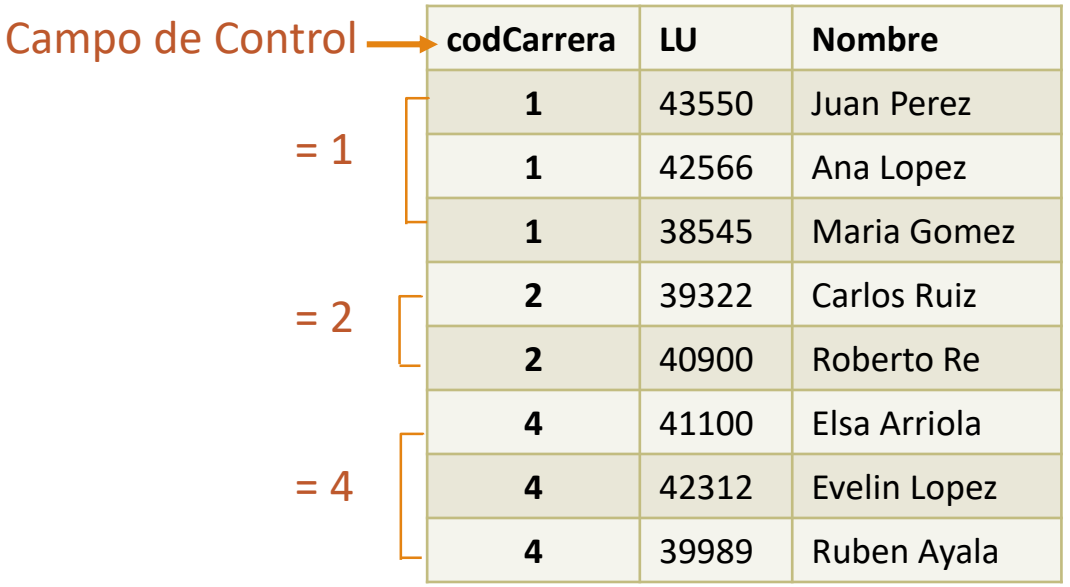
Cierre de archivo:

La función fclose devuelve un valor int, si es 0 significa que se cerro correctamente el archivo.

fclose(nombre archivo);

Leer un registro:

fread(nombre registro, tamaño registro, 1, nombre archivo);



codCarrera	LU	Nombre
1	43550	Juan Perez
1	42566	Ana Lopez
1	38545	Maria Gomez
2	39322	Carlos Ruiz
2	40900	Roberto Re
4	41100	Elsa Arriola
4	42312	Evelin Lopez
4	39989	Ruben Ayala

Corte de control

Corte de Control

En archivos secuenciales los registros están grabados en posiciones físicamente contiguas.

Para ciertas situaciones, resulta necesario ordenar los registros por un determinado campo(campo de control) antes de procesarlos.

El corte de control se utiliza cuando pueden detectarse, dentro del archivo ordenado, grupos de registros que corresponden a un mismo campo de control. Podemos detectar cuando finalizan los elementos de entrada de un grupo antes de comenzar el siguiente.

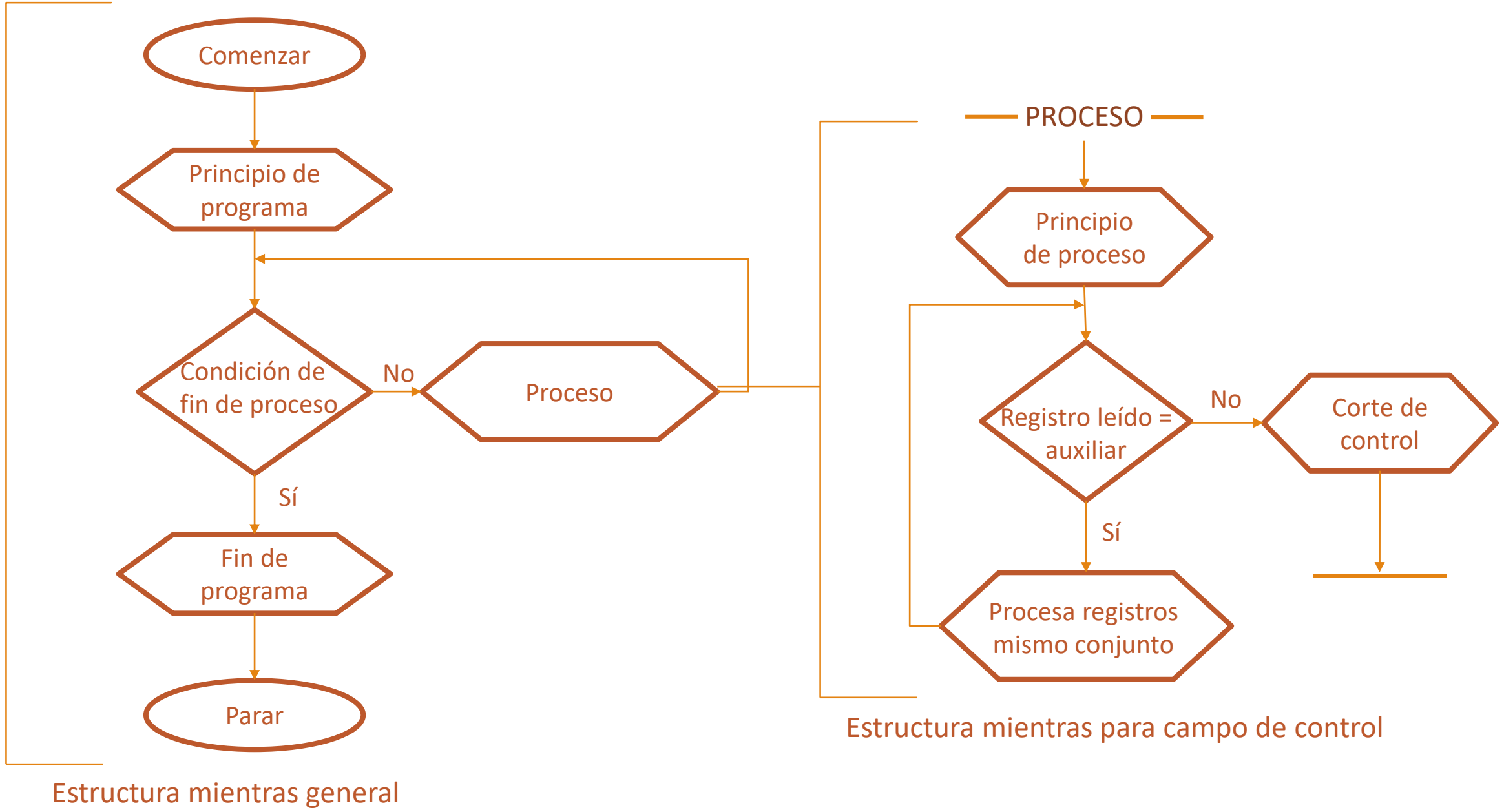
Campo de Control: Es el campo que identifica a cada subconjunto o grupo de elementos de entrada (registros) de un conjunto mayor de datos.

Definiciones a tener en cuenta …

- ✔ **Control:** significa mando, gobierno, dirección, dominio.
- ✔ **Control de Programa:** Es el mecanismo para dirigir, gobernar la ejecución de las instrucciones respetando la secuencia lógica establecida en el diagrama.
- ✔ **Corte de Control:** cuando se interrumpe el circuito de instrucciones que se estaban ejecutando.

Requisitos

- ✔ Ordenamiento de los datos de entrada.
- ✔ Que existan varios subconjuntos para que tenga sentido el corte de control.
- ✔ Que cada subconjunto tenga varios elementos o registros.



Estructura mientras general

Estructura mientras para campo de control

Ejemplo

Codificar un programa que realice un corte de control sobre un archivo de alumnos ordenados por código de carrera. Se debe:

Imprimir un detalle de los alumnos registrados en el archivo.

Mostrar, por cada carrera, los siguientes datos:

- Cantidad de alumnos.
- Cantidad de mujeres que cursan la carrera.
- Edad promedio de alumnos.

Al finalizar el programa se deberá informar:

- El total de alumnos.
- La edad promedio de los alumnos.

Formato del archivo:

Código de carrera	DNI	Código de sexo	edad	Nombre y Apellido

```
#include <stdio.h>
#include <string.h>

/* Tipos de datos personalizados */
typedef char tString[40];
typedef struct {
   int codCarrera, DNI;
   int codSexo, edad;
   tString nombreApellido;
}tRegistroAlumnos;

/* Declaración de prototipos */
void inicializacion();
void procesoCorte();
void finalizacion();

void principioCorte();
void unAlumno();
void finCorte();
```

```
/* Declaración de variables */
tRegistroAlumnos regAlumno;

FILE * archivo;

int codCarreraAnt;

int cantAlumnosTotal, cantAlumnosPorCarrera;
int cantMujeres;
int sumaEdadesPorCarrera, sumaEdadesTotal;

/* Bloque principal */
int main() {
      inicializacion();
      procesoCorte();
      finalizacion();
}
```

```
void inicializacion() {
	/*	Actividades al inicio del programa:
		1. Abrir archivo en modo lectura
		2. Leer el primer registro
		3. Guardar campo de control anterior
		4. Inicializar contadores/acumuladores generales/totales
		5. Escribir titulos */

	archivo = fopen("Alumnos.dat", "rb");
	 fread(&regAlumno, sizeof(tRegistroAlumnos), 1, archivo);

	cantAlumnosTotal = 0;
	sumaEdadesTotal = 0;
	//codCarreraAnt = regAlumno.codCarrera;	//Opción 2:Si guardamos el valor de anterior en esta función,
							// también debe realizarse esta acción en la función finCorte
	printf("Carrera\tDNI\tSexo\tEdad\tNombre y Apellido\n");
}
```

```
void procesoCorte() {
        /* Recorrer el archivo hasta el final (feof) */
        while (!feof(archivo)) {
                principioCorte();

                /* Con este while trabajamos con cada grupo  */
                while(!feof(archivo) && regAlumno.codCarrera == codCarreraAnt) {
                        unAlumno();
                        fread(&regAlumno, sizeof(tRegistroAlumnos), 1, archivo);
                }
                /* Cuando termina este while significa que terminó un grupo entonces
                realizamos las actividades del corte */

                finCorte();
        }
        /* Cuando termina este while significa que se terminó de recorrer el archivo */
}
```

```
void principioCorte() {
    /*    Actividades antes de recorrer los grupos: 1. Inicializar los contadores/acumuladores parciales */
    cantAlumnosPorCarrera = 0;
    sumaEdadesPorCarrera = 0;
    cantMujeres = 0;
    codCarreraAnt = regAlumno.codCarrera; //Opción 1: guardamos el valor del campo de control en codCarreraAnt
}

void unAlumno(){
    /* Actividades por grupo
            1. Actualizar contadores/acumuladores parciales
            2. Mostrar la línea del detalle en el caso que sea necesario
            3. Buscar el mayor/menor del grupo (parcial) en el caso que sea necesario
            4. Leer otro registro (leerRegistro) */
    cantAlumnosPorCarrera = cantAlumnosPorCarrera + 1;
    sumaEdadesPorCarrera = sumaEdadesPorCarrera + regAlumno.edad;
    if(regAlumno.codSexo == 2){
        cantMujeres = cantMujeres + 1;
    }
    printf("%d\t%d\t%d\t%d\t%s\n", regAlumno.codCarrera, regAlumno.DNI, regAlumno.codSexo, regAlumno.edad,
                                    regAlumno.nombreApellido);
}
```

```
void finCorte(){
    /*    Actividades por fin de corte de control
            1. Mostrar subtotales (línea de totales del grupo en el caso que el ejercicio lo solicite)
            2. Guardar la clave anterior(campo de control), en este caso codCarrera
            3. Calcular promedios/porcentajes parciales en el caso que el problema lo requiera
            4. Actualizar contadores/acumuladores generales
            5. Buscar el mayor/menor (general) en el caso que el problema lo requiera */

    cantAlumnosTotal = cantAlumnosTotal + cantAlumnosPorCarrera;
    printf("Cantidad de alumnos de la carrera %d: %d\n", codCarreraAnt, cantAlumnosPorCarrera);
    printf("Cantidad de mujeres en la carrera %d: %d\n", codCarreraAnt, cantMujeres);
    printf("Edad promedio de alumnos de la carrera %d: %.2f\n", codCarreraAnt,
                                                    (float)sumaEdadesPorCarrera/cantAlumnosPorCarrera);
    printf("-------------------------------------------------\n");

    sumaEdadesTotal = sumaEdadesTotal + sumaEdadesPorCarrera;

    //codCarreraAnt = regAlumno.codCarrera;    /*Opción 2: para esta opción volemos a guardar el valor del
                                                campo de control en anterior */
}
```

```
void finalizacion(){
	/*	Actividades al finalizar
			1. Mostrar totales
			2. Calcular promedios/porcentajes generales en el caso que el problema lo requiera
			3. Cerrar el archivo */
	printf("Cantidad de alumnos de todas las carreras: %d\n", cantAlumnosTotal);
	printf("Edad promedio de alumnos: %.2f\n\n", (float)sumaEdadesTotal/cantAlumnosTotal);

	fclose(archivo);
}
```