CI-2125 Computación I

Practica 8
Contenido: typedef y struct

1. Defina tipos enumerados para los siguientes conjuntos de datos
a) Días de la semana
b) Deportes en equipos
c) Meses del año

Solución:

enumdia {domingo,lunes,martes,miercoles,jueves,viernes,sabado}
enum deportes {futbol,baloncesto,beisbol,voleibol,balonmano,hockey,waterpolo}
enum mes
{enero,febrero,marzo,abril,mayo,junio,Julio,agosto,septiembre,octubre,noviembre,diciembre}

2. Proponga estructuras de datos que simulen las siguientes situaciones:
a) Información académica de un estudiante
b) Datos de los equipos del mundial de futbol

Solución:

a)

```
typedef struct est{
        int carnet;
        char nombres[50];
        char apellidos[50];
        int cedula;
        int edad;
        float índice;
        int codcarrera;
        char carrera[20];
}ESTUDIANTE;
```

b)

```
#define MAX 30

typedef struct miembr{
        char nombre[50];
        int edad;
        char posición[20];
        int nummundialesjugados;
        int numpartidosjugados;
        int golesAnotados;
} MIEMBRO;
```

```
typedef struct datos{
        char nombre[50];
        char país[30];
        char continente[20];
        char grupo;
        int numcopasganadas;
        int ranking;            // Posición en la tabla de la FIFA
        int nummiembros;
        MIEMBRO integrantes[MAX];
}EQUIPO;
```

3. Suponga que los datos de una persona son cédula, nombre, apellido y edad. Si se define un registro que contenga los datos de una persona, defina el algoritmo de un subprograma que inicialice cada campo del registro con valores leídos por teclado. Defina un segundo subprograma que imprima los valores de cada campo del registro. Escriba el algoritmo del programa principal que lea los datos de 5 personas y calcule el promedio de edad. El programa imprime el promedio y la persona de edad máxima. Escriba el programa en C equivalente usando typedef para definir las estructuras.

Programa Principal:
ENTRADAS:
    n (entero) //numero de personas a procesar
PRECONDICIÓN:
    0<n<MAX
POSTCONDICIÓN:
    Imprime los datos de la persona con edad máxima y el promedio de la edad

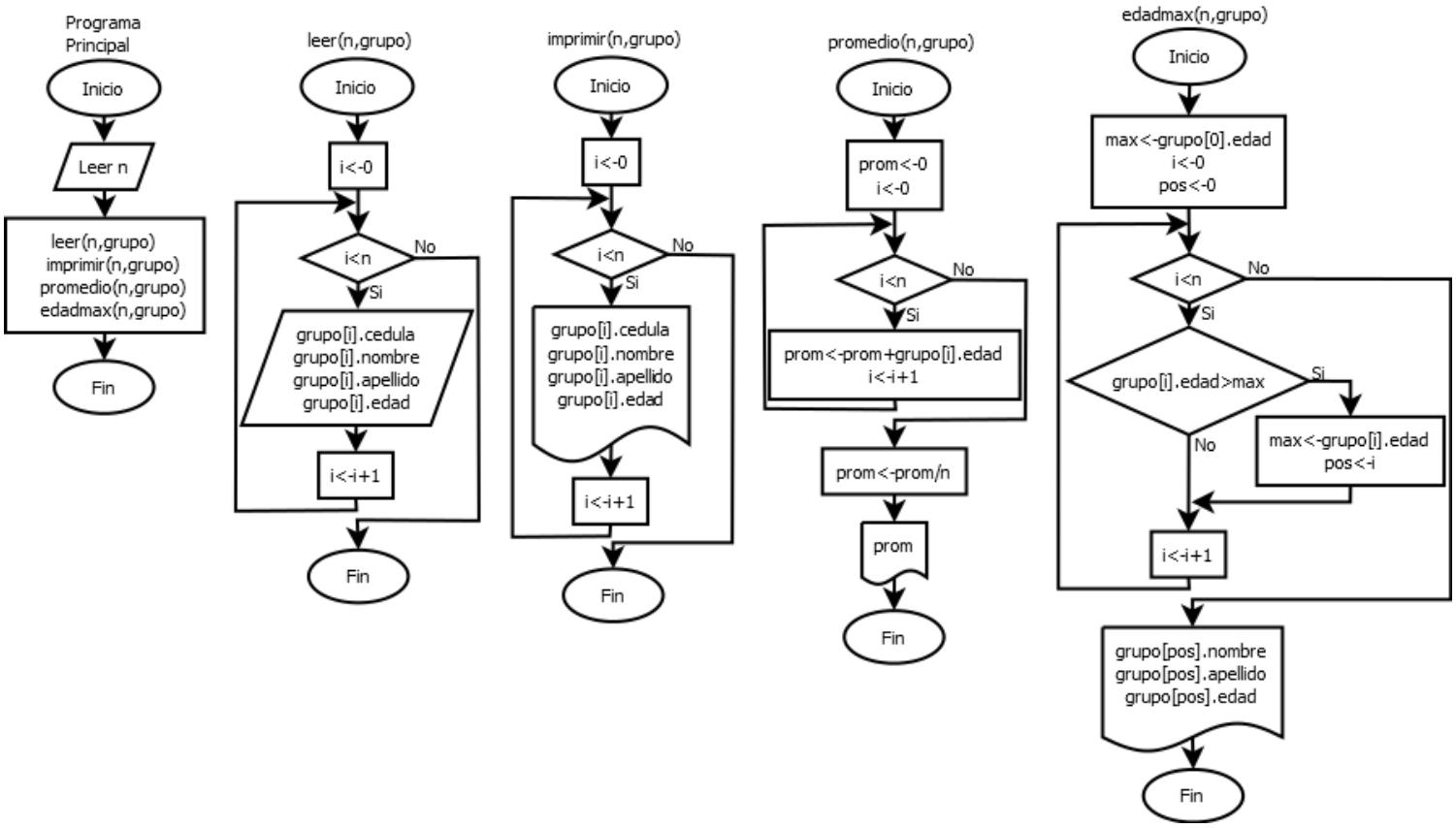
Procedimiento Leer:
ENRADAS:
    n (entero)
    grupo (arreglo de estructura)
PRECONDICION:
    0<n<MAX
SALIDAS:
    grupo (arreglo de estructura)
POSTCONDICION:
    grupo contiene los valores leídos

Procedimiento imprimir:
ENTRADAS:
    n (entero)
    grupo (arreglo de estructura)
PRECONDICION:
    0<n<MAX
POSTCONDICION:
    Imprime en pantalla los datos de las n personas

Procedimiento promedio:
ENTRADAS:
n (entero)
grupo (arreglo de estructura)
PRECONDICION:
0<n<MAX
POSTCONDICION:
Imprime en pantalla el promedio de las edades

Procedimiento edadmax:
ENTRADAS:
n (entero)
grupo (arreglo de estructura)
PRECONDICION:
0<n<10
POSTCONDICION:
Imprime la persona con edad máxima

Solución:



4. Escriba un algoritmo que lea N fechas y las inserte de manera ordenada en un arreglo. Suponga que se tiene definido el tipo fecha como un registro donde el día es un entero, el mes es de tipo

enumerado y el año es un entero. El programa finaliza imprimiendo las fechas de manera ordenada. Escriba el programa en C equivalente usando typedef para definir las estructuras y el tipo enumerado.

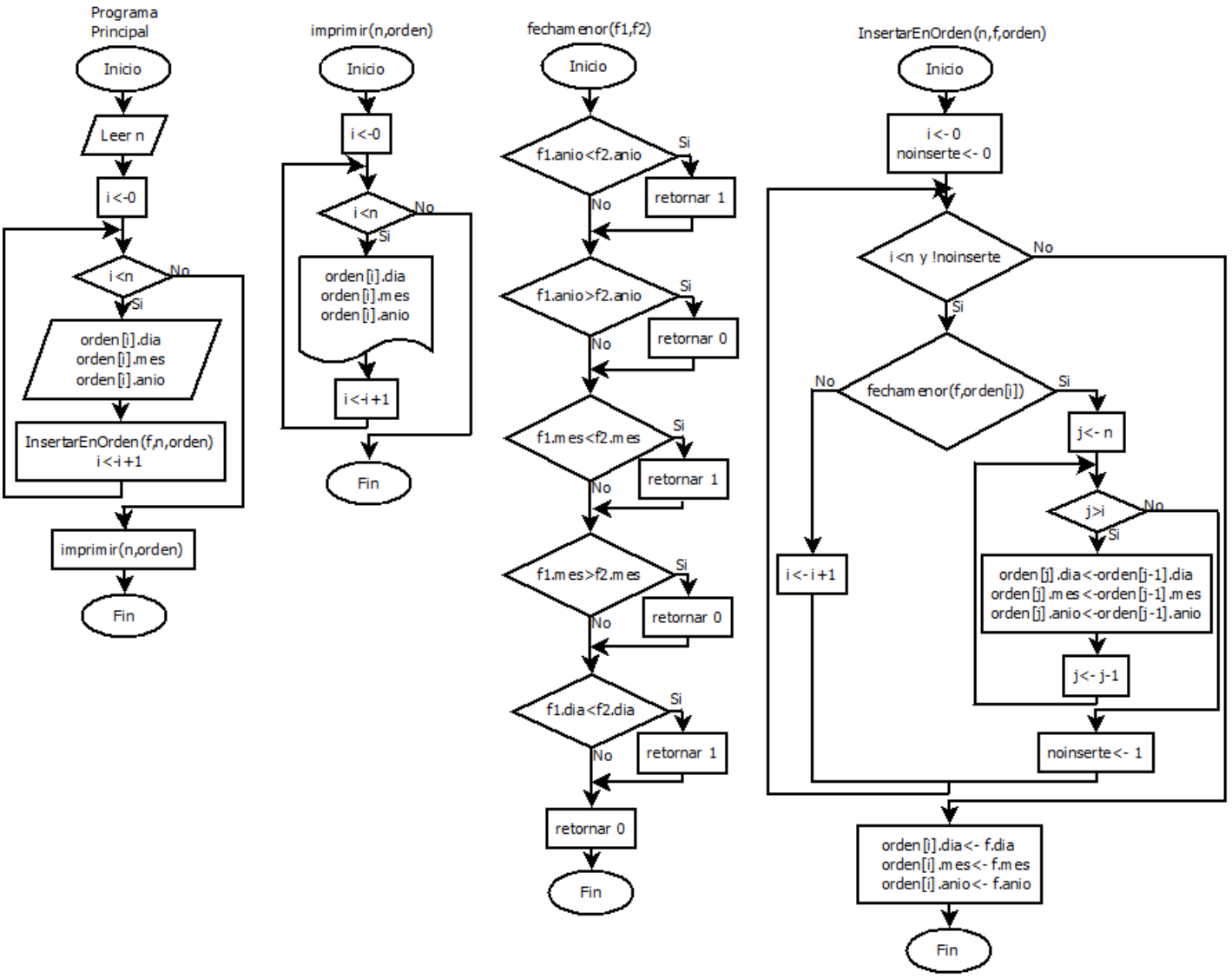
Solución:

Programa Principal:
ENTRADAS:
    n (entero) //numero de fechas a ordenar
    orden (arreglo de FECHAS) // estructura de fecha
PRECONDICIÓN:
    0<n<MAX
POSTCONDICIÓN:
    Escribe en pantalla las fechas ordenadas

Procedimiento imprimir:
ENTRADAS:
    n (entero)
    orden (arreglo de FECHAS)
PRECONDICION:
    0<n<100
POSTCONDICION:
    Escribe en pantalla los elementos del arreglo

Función fechamenor:
ENTRADAS:
    f1 (FECHA)
    f2 (FECHA)
PRECONDICION:
    0<f1.dia<=31 y 0<f2.dia<=31 y f1.anio>0 y f2.anio>0
    //el mes no hace falta chequearlo por ser tipo enumerado
SALIDAS:
    fechamenor (entero)
POSTCONDICION:
    Retorna 1 si la fecha f1 es menor a la fecha f2 de lo contrario retorna 0

Procedimiento InsertarEnOrden:
ENTRADAS:
    n (entero)
    f (FECHA)
    orden (arreglo de FECHAS)
PRECONDICION:
    0<n<100 y 0<f.dia<=31 y f.anio>0 y el arreglo orden está ordenado
SALIDAS:
    orden(arreglo de FECHAS)
POSTCONDICION: Se ingresa la fecha f en el arreglo de manera que este queda ordenado

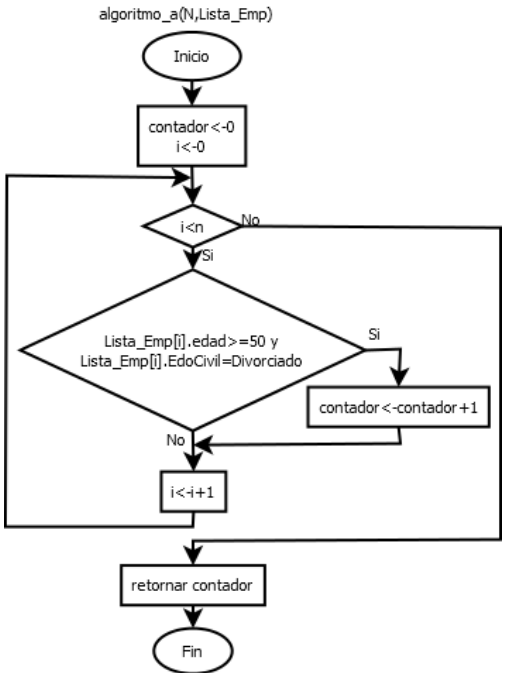


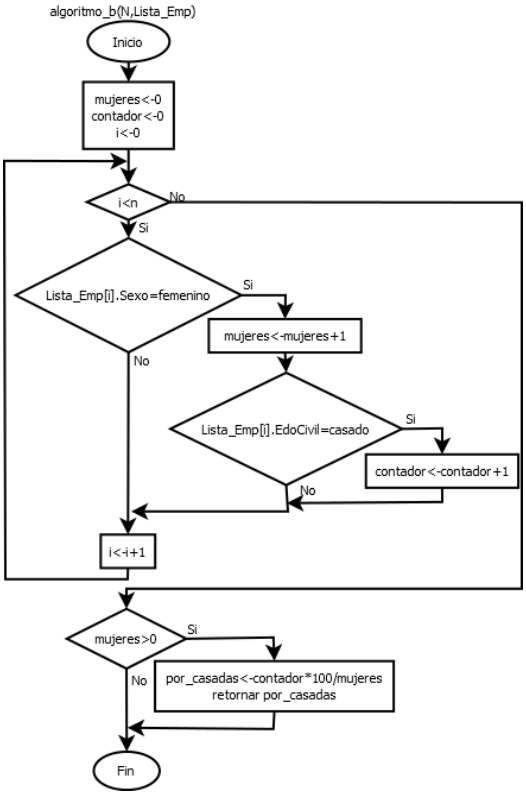
5. Dadas las siguientes declaraciones:

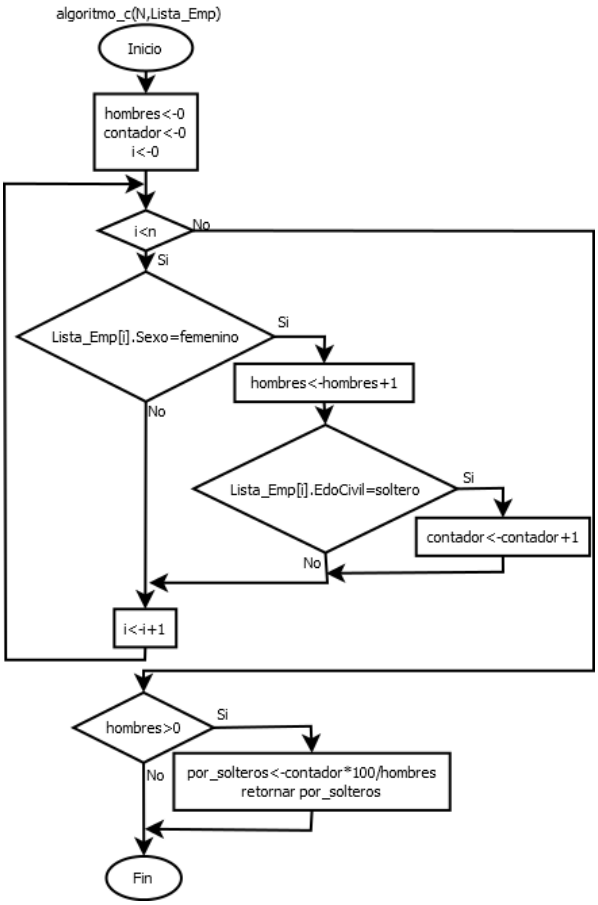
```
#define N 100
typedef enum Estado {Soltero,Casado,Viudo,Divorciado};

typedef struct Empleados {
      char Apellido[20],Nombre[20];
      int Edad;
      enum Sexo {Masculino,Femenino};
      ESTADO EdoCivil;
}Tipo_Emp;

Tipo_Emp Lista_Emp[N];
```

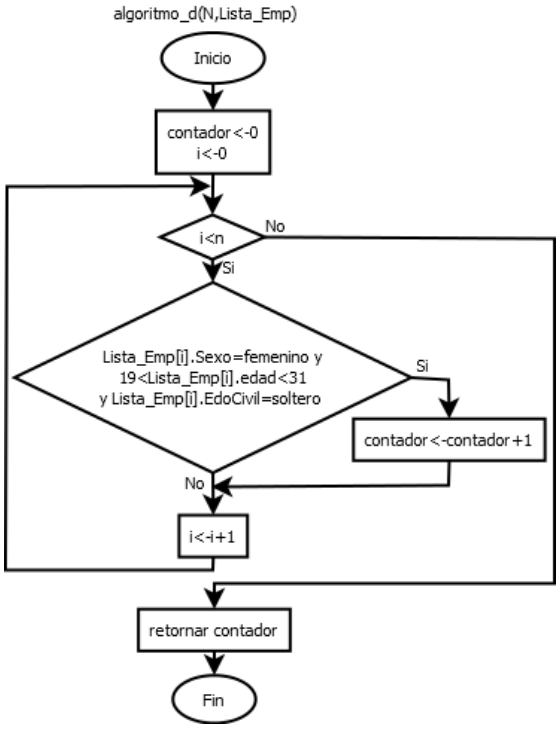
Escriba algoritmos para funciones o subprogramas que respondan:

a) ¿Cuántos empleados mayores de 50 años están divorciados?
b) ¿Cuál es el porcentaje de mujeres casadas?
c) ¿Cuál es porcentaje de empleados solteros?
d) ¿Cuántas mujeres solteras están entre los 20 y 30 años de edad?






algoritmo_c(N,Lista_Emp)
Inicio
hombres<-0
contador<-0
i<-0
i<n
No
Si
Lista_Emp[i].Sexo=femenino
Si
hombres<-hombres+1
No
Lista_Emp[i].EdoCivil=soltero
Si
contador<-contador+1
No
i<-i+1
hombres>0
Si
No
por_solteros<-contador*100/hombres
retornar por_solteros
Fin


algoritmo_d(N,Lista_Emp)
Inicio
contador<-0
i<-0
i<n
No
Si
Lista_Emp[i].Sexo=femenino y
19<Lista_Emp[i].edad<31
y Lista_Emp[i].EdoCivil=soltero
Si
contador<-contador+1
No
i<-i+1
retornar contador
Fin

6. Suponga que los datos de una ASIGNACIÓN en el comprobante de nómina de un empleado son bono de transporte, bono de alimentación y bono de vacaciones, todos números reales. Luego las DEDUCCIÓNES son el seguro y el ahorro habitacional, también números reales. Además los datos del EMPLEADO son el código (entero), si está activo o no, el sexo, el sueldo por hora (real) y las horas trabajadas. Escriba algoritmos para los siguientes subprogramas

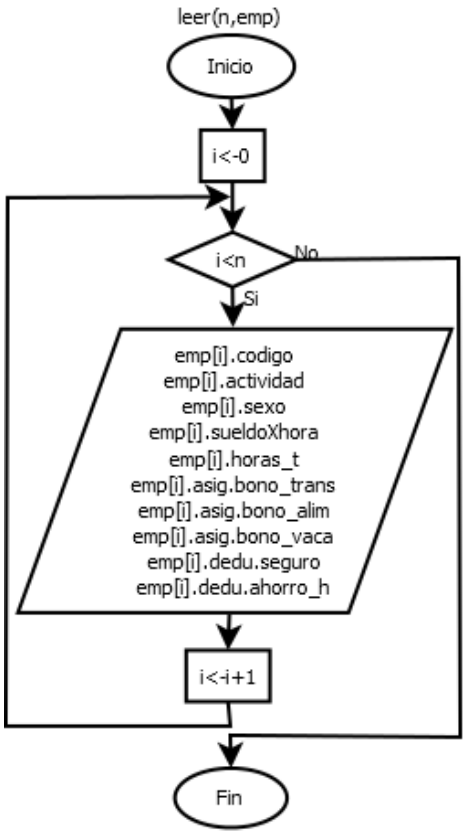
a) Inicializar los datos de un empleado que incluyen las asignaciones y deducciones con valores leídos.

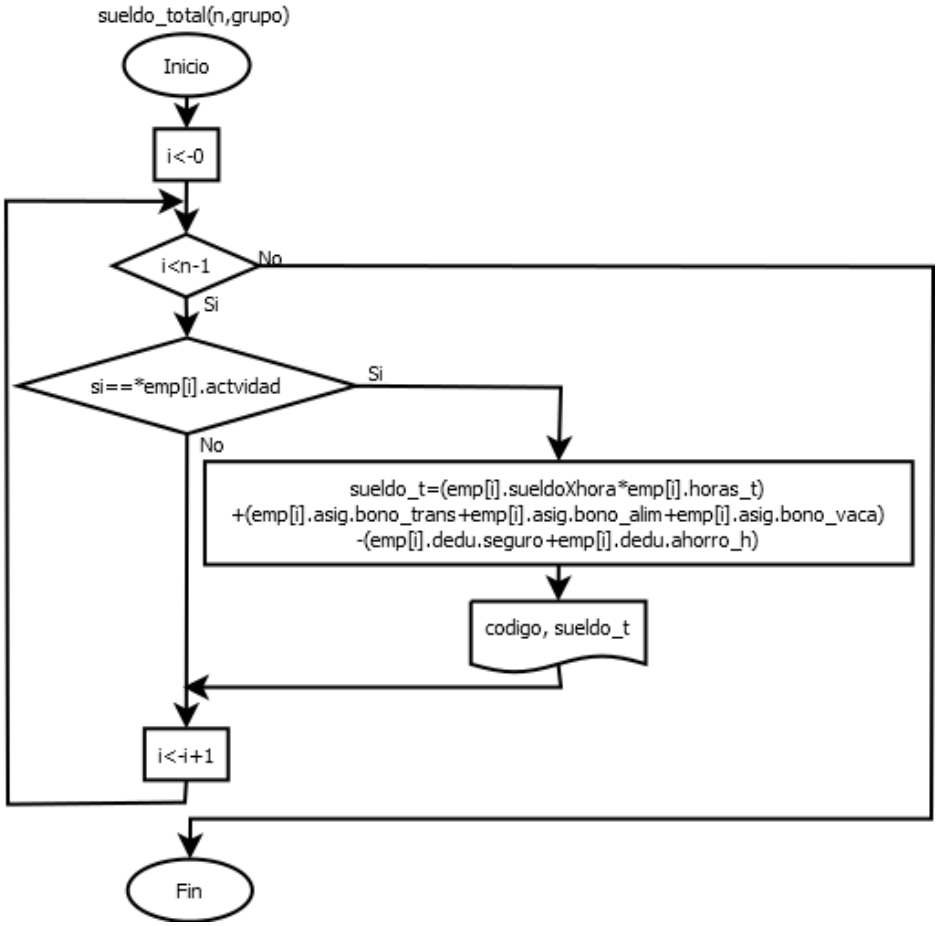
b) Calcular el sueldo de los empleados activos, dado que ST = (SH * HT) + A – D, siendo las variables las siguientes: ST = Sueldo Total, SH = Sueldo por Hora, HT = Horas Trabajadas, A = Asignaciones, D = Deducciones

Solución:

Procedimiento Leer:
ENTRADAS:
    n (entero)
PRECONDICION:
    0<n<MAX
SALIDAS:
    emp (arreglo de EMPLEADOS)
POSTCONDICION:
    emp contiene los valores leidos

Procedimiento sueldo_total:
ENTRADAS:
    n (entero)
    emp (arreglo de estructura)
PRECONDICION:
    0<n<MAX
POSTCONDICIONES:
    Imprime el sueldo total de los empleados activos


leer(n,emp)
Inicio
i<-0
i<n
No
Si
emp[i].codigo
emp[i].actividad
emp[i].sexo
emp[i].sueldoXhora
emp[i].horas_t
emp[i].asig.bono_trans
emp[i].asig.bono_alim
emp[i].asig.bono_vaca
emp[i].dedu.seguro
emp[i].dedu.ahorro_h
i<-i+1
Fin


sueldo_total(n,grupo)
Inicio
i<-0
i<n-1
No
Si
si==*emp[i].actvidad
Si
No
sueldo_t=(emp[i].sueldoXhora*emp[i].horas_t)
+(emp[i].asig.bono_trans+emp[i].asig.bono_alim+emp[i].asig.bono_vaca)
-(emp[i].dedu.seguro+emp[i].dedu.ahorro_h)
codigo, sueldo_t
i<-i+1
Fin