
EJERCICIOS CAPÍTULO 12

Ejercicio 1

Un programa toma como entrada un fichero cuyo formato de registro es el siguiente:

Numero-empleado	Nombre-empleado	Meses-Trabajo	Directivo
-----------------	-----------------	---------------	-----------

donde:

- **Nmero-empleado** es un campo de números enteros positivos de 3 dígitos (excluido el 000).
- **Nombre-empleado** es un campo alfanumérico de 10 caracteres.
- **Meses-Trabajo** es un campo que indica el número de meses que lleva trabajando el empleado; es un entero positivo (incluye el 000) de 3 dígitos.
- **Directivo** es un campo de un solo carácter que puede ser «+» para indicar que el empleado es un directivo y «-» para indicar que no lo es.

El programa asigna una prima (que se imprime en un listado) a cada empleado según las normas siguientes:

- P1 a los directivos con, al menos, 12 meses de antigüedad
- P2 a los no directivos con, al menos, 12 meses de antigüedad
- P3 a los directivos sin un mínimo de 12 meses de antigüedad
- P4 a los no directivos sin un mínimo de 12 meses de antigüedad

Se pide:

- 1.1. Crear una tabla de clases de equivalencia (las clases deberán ser numeradas) en la que se indiquen las siguientes columnas en cada fila:
 - Condición de entrada que se analiza
 - Clases válidas y
 - Clases no válidas que se generan para la condición
 - Regla heurística que se aplica para la generación de las clases de la fila
- 1.2. Generar los casos de prueba (especificando la entrada en todos los casos y la salida esperada sólo en los casos válidos) para las clases creadas usando la técnica de particiones de equivalencia, indicando en cada caso las clases que cubre. Enunciar la regla se aplica para derivar los casos a partir de las clases de equivalencia.

Ejercicio 2

Se ha ideado el siguiente pseudocódigo para cumplir la especificación del problema anterior.

```
Begin {programa}
  Print ("Ilmo.Sr.Director General:");
  Read (registro_FICH);
  Prima = 0;
  While (no FF de FICH) do
    Begin {while}
      If (meses_FICH>=12)
        Then
```

```

        If (directivo_FICH="+")
            then Prima=1000
            else Prima=75
    Else
        If (directivo_FICH="+")
            then Prima=500
            else;
        Print (num_FICH, nombre_FICH, Prima);
        Read (registro_FICH)
    End {while};
    Print ("S.e.u.o.")
End {programa}.

```

Se han añadido algunas líneas de cabecera y de final al listado de las primas, y se han asignado valores concretos a las primas P1, P2, P3 y P4. El fichero de entrada se denomina FICH y los nombres de los campos son más o menos iguales.

Se pide:

Suponiendo que en la anterior prueba de clases de equivalencia el programa se ejecuta sólo con los dos registros siguientes (de casos válidos), consecutivamente, en el orden dado (el símbolo # indica un carácter blanco):

```

123  Fernández#    009 +          Registro 1
456 Fernando## 013 -          Registro 2

```

- 2.1 Comprobar si se cumple la cobertura de sentencias indicando, en su caso, cuáles de ellas no se ejecutan, empleando la numeración de líneas del código que se ofrece. En caso de que no se cumpla, añadir el mínimo número de registros adicionales para que se cumpla la cobertura.
- 2.2 Comprobar si se cumple la cobertura de decisiones, creando una tabla donde se marque los valores que adopta cada decisión, identificada con la numeración de líneas de código que se ofrece (por ej., la decisión "meses-FICH>=12" será la decisión 7). En caso de no cumplirse la cobertura, añadir el mínimo número de registros adicionales para que se cumpla. Los casos a añadir deben definir la entrada y la salida esperada al ejecutarse junto a los dos casos de caja negra (registros 1 y 2). Emplear el formato de salida que se incluye en el código del programa.

Ejercicio 3

Dado el siguiente diagrama modular (véase la Figura 0-1.), que podría constituir un diseño alternativo para el problema anterior, y suponiendo que no se admite ningún tipo de agrupación de módulos para hacer pruebas de unidad:

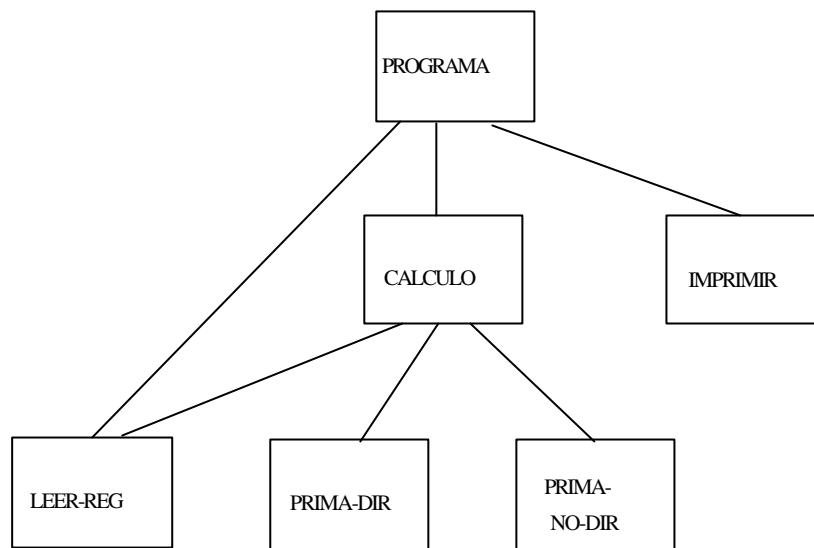


Figura 0-1. Diseño modular sobre el que aplicar el ejercicio

SE PIDE:

3.1. Calcular el número mínimo de módulos ficticios subordinados y módulos impulsores para los supuestos de integración descendente, ascendente y sandwich (respóndase sobre el cuadro adjunto).

Tipo de integración	Nº de Módulos Subordinados	Nº de módulos Impulsores
Ascendente		
Descendente		
Sandwich o mixta		

3.2. Dibujar la secuencia de pasos para realizar la integración tipo sandwich hasta el último

3.3 Realizar los mismos cálculos y dibujos para alguno de los ejercicios de diseño estructurado del capítulo 8.