# Prog. Básica - Laboratorio 2 Instrucciones condicionales

**Nombre**: Markel Ferro **Fecha**: 26/09/2020

Establece para cada uno de los siguientes enunciados:

a) si es ambiguo el enunciado.

b) la correspondiente especificación.

c) una lista de casos de prueba.

d) el diseño de un algoritmo para llevarlo a cabo.

e) indicar qué lineas del algoritmo se ejecutan

**1. Resultado de la ONCE.** Crea un algoritmo que, dados dos números de cuatro cifras (el número de cupón premiado y el número de cupón que hay que comprobar), obtenga el premio que le corresponde al cupón a comprobar, sabiendo que si tiene los cuatro dígitos iguales (y en el mismo orden) le corresponden 100000 euros, si coinciden los tres últimos dígitos le corresponden 50000 euros y si son los dos últimos el premio son 3 euros. El resto de casos no tienen premio asociado.

**2. Segundo anterior.** Crea un algoritmo que, dada una hora válida (*horas*, *minutos* y *segundos*), calcule qué hora era un segundo antes. Por ejemplo, la hora anterior a 10:30:40 es 10:30:39.

**3. Añadir duración a una tarea.** Crea un algoritmo que, dada la hora de comienzo de una tarea y la duración en segundos de ésta, calcule la hora en que finalizará la tarea. La duración de la tarea puede ser de varias horas, pero nunca superior a media jornada (14400s). Las tareas incompletas continúan en la siguiente jornada. La jornada laboral es de 8:00 a 16:00.

**4. Día anterior.** Crea un algoritmo que, dada una fecha (*día*, *mes* y *año*) calcule el día anterior. Comienza considerando que todos los años no son bisiestos. Luego añade la condición de que el año puede ser bisiesto. Los años bisiestos son los que son múltiplos de 4 y no ser múltiplos de 100. Los múltiplos de 400 son una excepción, ya que, aun siendo múltiplos de 100, son bisiestos.

**5. Estación del año.** Crea un algoritmo que dados dos números enteros día y mes, indique a qué estación del año corresponde dicha fecha. Se considera que la primavera va desde el 21 de marzo hasta el 20 de junio; el verano se extiende desde el 21 de junio al 20 de septiembre; el otoño dura desde el 21 de septiembre hasta el 20 de diciembre; y el invierno ocurre entre el 21 de diciembre y el 20 de marzo.

1. **Resultado de la ONCE**

Crea un algoritmo que, dados dos números de cuatro cifras (el número de cupón premiado y el número de cupón que hay que comprobar), obtenga el premio que le corresponde al cupón a comprobar, sabiendo que si tiene los cuatro dígitos iguales (y en el mismo orden) le corresponden 100000 euros, si coinciden los tres últimos dígitos le corresponden 50000 euros y si son los dos últimos el premio son 3 euros. El resto de casos no tienen premio asociado

* 1. Especificación

Antes que nada, ¿es ambiguo el enunciado?

Después, rellena la especificación:

|  |
| --- |
| Datos?/Entrada?: cupon\_premiado, cupon\_comprobar : Integer;  Pre: Deben ser de 4 números los cupones    Resultados?/Salida?: Premio  Post: Dependiendo de las similitud de los cupones el premio cambia |

* 1. Casos de Prueba

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | Cupón 1 | Cupón 2 | Descripción | Premio |
| 1 | 4242 | 4242 | Todos los números son iguales | 100000 |
| 2 | 4242 | 3242 | Tres números son iguales | 50000 |
| 3 | 4242 | 3142 | Dos números son iguales | 3 |
| 4 | 4242 | 3132 | Un número es igual | 0 |
| 5 | 4242 | 3131 | Ningún número es igual | 0 |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |
|  |  |  |  |  |

* 1. Algoritmo

Texto

Descripción generada automáticamente

1. **Segundo anterior**

Crea un algoritmo que, dada una hora válida (*horas*, *minutos* y *segundos*), calcule qué hora era un segundo antes. Por ejemplo, la hora anterior a 10:30:40 es 10:30:39.

* 1. Especificación

Antes que nada, ¿es ambiguo el enunciado? No

Después, rellena la especificación:

|  |
| --- |
| Datos?/Entrada?: Horas: Natural (0..23). Minutos: Natural (0..59). Segundos: Natural (0..59)  Pre: Las tres están metidas dentro de tiempo con sus respectivos rangos    Resultados?/Salida?: Horas: Natural (0..23). Minutos: Natural (0..59). Segundos: Natural (0..59)  Post: Las tres están metidas dentro de tiempo con sus respectivos rangos con un segundo reducido |

* 1. Casos de Prueba

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| # | hh | mm | ss | Descripción | hh | mm | ss |
| 1 | 10 | 30 | 10 | Cambia *ss* | 10 | 30 | 9 |
| 2 | 10 | 28 | 00 | Cambia *ss y mm* | 10 | 27 | 59 |
| 3 | 10 | 00 | 00 | Cambia hh, mm y ss | 09 | 59 | 59 |
| 4 | 00 | 00 | 00 | Cambia hh, mm, ss | 23 | 59 | 59 |
|  |  |  |  |  |  |  |  |

* 1. Algoritmo

Texto

Descripción generada automáticamente

* 1. Simulación

|  |  |
| --- | --- |
| hh:mm:ss | Simulación: indica qué líneas del algoritmo se ejecutan |
| 10:30:10 | En el if de segundo\_Inicial da false con lo que salta al último else |

1. **Añadir duración a una tarea**

Crea un algoritmo que, dada la hora de comienzo de una tarea y la duración en segundos de ésta, calcule la hora en que finalizará la tarea. La duración de la tarea puede ser de varias horas, pero nunca superior a media jornada (14400s). Las tareas incompletas continúan en la siguiente jornada. La jornada laboral es de 8:00 a 16:00.

* 1. Especificación

Antes que nada, ¿es ambiguo el enunciado?

Después, rellena la especificación:

|  |
| --- |
| Datos?/Entrada?:  Pre:    Resultados?/Salida?:  Post: |

* 1. Casos de Prueba

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| # | hh:mm:ss | duración | Descripción | hh:mm:ss | día\_siguiente |
| 1 | 10:30:30 | 30 | medio minuto | 10:31:00 | false |
| 2 | 10:30:30 | 1800 | media hora | 11:00:30 | false |
| 3 | 10:40:30 | 3606 | una hora y 6 segundos | 11:40:36 | false |
| 4 | 15:50:10 | 3600 | cambiamos de jornada | 8:50:10 | true |
| 5 | 15:50:10 | 14400 | máxima duración de tarea | 11:50:10 | true |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

* 1. Algoritmo
  2. Simulación

|  |  |
| --- | --- |
| hh:mm:ss | Simulación: indica qué líneas del algoritmo se ejecutan |
| 10:30:10 |  |

1. **Día anterior**

Crea un algoritmo que, dada una fecha (*día*, *mes* y *año*) calcule el día anterior. Comienza considerando que todos los años no son bisiestos. Luego añade la condición de que el año puede ser bisiesto. Los años bisiestos son los que son múltiplos de 4 y no ser múltiplos de 100. Los múltiplos de 400 son una excepción, ya que, aun siendo múltiplos de 100, son bisiestos.

* 1. Especificación

Antes que nada, ¿es ambiguo el enunciado?

Después, rellena la especificación:

|  |
| --- |
| Datos?/Entrada?:  Pre:    Resultados?/Salida?:  Post: |

* 1. Casos de Prueba

|  |  |  |  |
| --- | --- | --- | --- |
| # | dd-mm-aaaa | Descripción | dd-mm-aaaa |
| 1 | 01-01-2006 | día 1 tras mes de 31 días | 31-12-2005 |
| 2 | 01-03-2006 | día 1 tras febrero de año no bisiesto | 28-02-2006 |
| 3 | 01-03-2000 | día 1 tras febrero de año bisiesto | 29-02-2000 |
| 4 | 01-05-2006 | día 1 tras mes de 30 días (1) | 30-04-2006 |
| 5 | 01-10-2006 | día 1 tras mes de 30 días (2) | 30-09-2006 |
| 6 | 13-03-2006 | un día cualquiera de un mes cualquiera | 12-03-2006 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

* 1. Algoritmo
  2. Simulación

|  |  |
| --- | --- |
| dd-mm-aaaa | Simulación: indica qué líneas del algoritmo se ejecutan |
|  |  |

1. **Estación del año**

Crea un algoritmo que dados dos números enteros *día* y *mes,* indique a qué estación del año corresponde dicha fecha. Se considera que la primavera va desde el 21 de marzo hasta el 20 de junio; el verano se extiende desde el 21 de junio al 20 de septiembre; el otoño dura desde el 21 de septiembre hasta el 20 de diciembre; y el invierno ocurre entre el 21 de diciembre y el 20 de marzo.

* 1. Especificación

Antes que nada, ¿es ambiguo el enunciado?

Después, rellena la especificación:

|  |
| --- |
| Datos?/Entrada?: día, mes: Positive  Pre: Día 28..31, Mes 1..12    Resultados?/Salida?: día\_del\_ano: Integer  Post: Se acciones (en este caso Puts) usando una lista de días vs estaciones |

* 1. Casos de Prueba

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| # | día | mes | Descripción | estación |
| 1 | 1 | 1 | 1 de enero = Invierno | Invierno |
| 2 | 20 | 3 | 20 de marzo = Invierno | Invierno |
| 3 | 21 | 3 | 21 de marzo = Primavera | Primavera |
| 4 | 20 | 6 | 20 de junio = Primavera | Primavera |
| 5 | 21 | 6 | 21 de junio = Verano | Verano |
| 6 | 20 | 8 | 20 de septiembre = Verano | Verano |
| 7 | 21 | 8 | 21 de septiembre = Otoño | Otoño |
| 8 | 20 | 12 | 20 de diciembre = Otoño | Otoño |
| 9 | 21 | 12 | 21 de diciembre = Invierno | Invierno |
| 10 | 31 | 12 | 31 de diciembre = Invierno | Invierno |

* 1. Algoritmo

Texto

Descripción generada automáticamente

* 1. Simulación

|  |  |  |
| --- | --- | --- |
| dd | mes | Simulación: indica qué líneas del algoritmo se ejecutan |
| 20 | 6 | Se ejecutan todas, menos en el case que se ejecute que encuentra el 171 |