Problemas de selección

Fundamentos de la programación y la informática
Grado en ingeniería aeroespacial en vehículos aeroespaciales 2019-2020
Escuela Técnica Superior de Ingeniería de Telecomunicación
Universidad Rey Juan Carlos

Cambios en el documento

15 de noviembre. Práctica 4.4. Donde decía *la magnitud será un número real* ahora dice *la magnitud será un número entero*

Práctica 4.1. Temperatura

En wikipedia podemos leer lo siguiente:

- Hipotermia: Cuando la temperatura axilar es inferior a 36 ºC.
- Febrícula: Cuando la temperatura axilar se encuentra entre 37.0 °C y 37.5 °C.
- Fiebre: Cuando la temperatura axilar se encuentra entre 37.5 °C y 41 °C.
- Hiperpirexia: Cuando la temperatura axilar es igual o mayor que 41 °C.

La medicina no es una ciencia exacta así que posiblemente esta definición es válida. Pero en informática es una ambiguedad inaceptable, porque si la temperatura es exactamente 37.5 ºC no sabemos si es febrícula, fiebre o ambas. Y si la temperatura es exactamente 41 ºC , también hay ambigüedad entre fiebre e hiperpirexia.

Escribe un programa llamado `~/fpi/practica04/temperatura.pas` que contenga una función que

1. Tenga como argumento una temperatura

2. Devuelva la cadena de texto *hipotermia*, *temperatura normal*, *febrícula*, *fiebre* o *hiperpirexia*, según corresponda. Aplica el criterio definido anteriormente, pero resuelve las ambigüedades (de la manera que creas conveniente)

Observa que si la temperatura es superior a la hipotermia pero inferior a la febrícula, es normal.

El cuerpo principal del programa invocará 5 veces a esta función, con diferentes valores

Práctica 4.2. Aeropuertos

Como seguramente sabes, casi todos los aeropuertos del mundo tienen un código de tres letras mayúsculas denominado *código IATA* que se usa, por ejemplo, en las etiquetas del equipaje.

Puedes consultarlos aquí https://en.wikipedia.org/wiki/IATA_airport_code

Elige 5 aeropuertos cualquiera y escribe un programa llamado `~/fpi/practica04/aeropuertos.pas` según la siguiente especificación

1. El programa tendrá una función que reciba como argumento una cadena de texto con un código IATA de aeropuerto, y que devuelva
   - Una cadena de texto con el nombre del aeropuerto, si es uno de los 5 que la función reconoce
   - La cadena "Aeropuerto desconocido" en otro caso
2. Esta función estará basada en sentencias if encadenadas (else-if)
3. El programa tendrá un cuerpo principal que invocará a la función al menos dos o tres veces, para probarla.

Práctica 4.3. Precondiciones en el programa múltiplos

Copia tu práctica `~/fpi/practica03/multiplos.pas` en `~/fpi/practica04/multiplos.pas`. En este último fichero:

- Añade funciones que comprueben las precondiciones. Si no se cumplen, el programa principal mostrará un error y no invocará a tu función principal.
- Añade funciones que compruebe las postcondiciones. Si no se dan, el programa mostrará un mensaje de error.

(Aunque en un entorno más realista, raramente comprobaríamos estas postcondiciones porque es obvio que se cumplirán si el programa está medianamente bien escrito)

Práctica 4.4. Fase de vuelo

Consideremos que las fases posibles de vuelo de un avión comercial son las siguientes:

- Estacionamiento: velocidad nula.
- Despegue: velocidad no nula, menor de 150 nudos y aceleración positiva
- Ascenso inicial: velocidad entre 150 y 240 nudos y aceleración positiva
- Ascenso final: velocidad mayor de 240, menor o igual a 520 nudos y aceleración positiva
- Crucero: velocidad mayor de 520 nudos (sin importar la aceleración)
- Descenso inicial: velocidad mayor de 300 nudos, menor o igual a 520 nudos, aceleración negativa
- Descenso final: velocidad mayor o igual a 140 nudos, menor o igual a 300 nudos, aceleración negativa
- Aterrizaje: velocidad no nula, menor de 140 nudos y aceleración negativa

Escribe un programa llamado `~/fpi/practica04/fases_vuelo.pas` según la siguiente especificación

1. A partir de la velocidad y la aceleración, indicará la fase de vuelo correspondiente.
2. La velocidad que reciba podrá estar especificada en metros por segundo, en kilómetros por hora o en nudos.
   - La magnitud será un número entero.
   - La unidad estará expresada con una cadena de texto, que podrá tomar los valores `m/s`, `km/h` o `kn`. (Exactamente así, en minúsculas). Si la unidad es incorrecta, el programa mostrará un mensaje de error y no la fase de vuelo.
3. Internamente, el programa trabajará en nudos. Así que en caso de que la unidad de entrada sea distinta, lo primero que hará el programa es convertirlo (observa que esto es el preproceso).
4. La magnitud y las unidades de la aceleración no son relevantes, así que estará expresada con un booleano que valdrá *TRUE* si es positiva y *FALSE* si es negativa o nula.

5. El programa no leerá nada del teclado: usa constantes para los valores de entrada.

Revisión de los nombres de los ficheros

Ejecuta `~mortuno/revisa practicas fpi` para comprobar que los nombres de los programas son los correctos.