

EJERCICIO DE PREPARACIÓN DEL EXAMEN DE TECNOLOGÍA DE LA PROGRAMACIÓN. CURSO 05/06

Este ejercicio esta propuesto para que se haga un repaso de casi toda la materia dada en la asignatura y que, por tanto, suponga que realizar el mismo correctamente sea una garantía de tranquilidad de cara a afrontar el examen práctico de la misma. El enunciado del ejercicio es el siguiente:

La pastelería *MuyFriki S. L.* (Sin Limite) esta pensando en presentarse al concurso para obtener el premio *PMFDA (pastelería Mas Friki Del Año)*, para lo cual necesita presentar un sistema informático que simule el funcionamiento de su nueva fabrica automatizada de productos, sistema que se deberá hacer siguiendo las especificaciones puestas por el dueño de la pastelería (el señor *Cabello de Ángel*), y que se detallan en este documento.

Antes de pasar a la descripción de lo que hay que hacer, conviene destacar que en este modelo, para simplificar, solo se contemplarán dos tipos de productos a elaborar: El *turrón duro INTEL* (que proviene de *INteresante Turrón ELaborado*) y las *Magdalenas AMD* (que viene de *Apetitosas Magdalenas Dulces*). Las características de ambos elementos son:

Turrón INTEL:

- Peso de cada barra: Las barras oscilarán entre los 400 y 600 gramos, según cuantas nueces *CORE* (CORTadas y REquemadas) incorpore y de la consistencia de la masa.
- Ancho de la barra: Debido a un error en la operación de división en la maquina que corta las piezas, el ancho oscilará entre los 7 y los 9 cms.
- Longitud de la barra: Igualmente oscilará entre 10 y 15 cms.
- Grosor: El grosor de cada barra variará entre 3 y 4 cms.
- Sabor: Solo se fabrican 3 tipos de sabores inventados por esta pastelería y patentados en todo el mundo, cuyo nombre corresponde a una receta secreta que produce sabores especiales irresistibles que la han hecho muy famosa. Todos los sabores se identifican por un código numérico y son los siguientes: *Silicio* (Nº 1), *Cobre* (Nº 2) y *Neón* (Nº 3).
- Concentración de CORES (nueces): Oscilará entre el 50% y el 75%. Aunque pudiera parecer que si, para simplificar se supondrá que no hay relación entre el numero de CORES del turrón y su peso.

Magdalenas AMD:

Al igual que los turrones, las magdalenas también están sometidas a la tiranía y al mal hacer de nuestra maquinaria con defectos en las operaciones de división:

- Peso de la magdalena: Oscilará entre 50 y 65 gramos.
- Altitud de la magdalena: Entre 8 y 11 cms.
- Sabor: Se fabricarán magdalenas de los 3 sabores dados anteriormente.

Añadir a estos elementos todos los métodos que se consideren necesarios para la realización del ejercicio. Obligatoriamente los elementos deberán incluir los siguientes métodos:

- Índice de calidad: El índice de calidad de una magdalena es un numero REAL que se obtiene mediante la suma de $(\text{Peso}/50) + (\text{Altitud}/9)$. El de un turrón se obtiene mediante la formula: $(\text{Peso}/500) + (\text{concentración}/60) + (\text{longitud}/12)$.

- Turrón Extreme Edition: Este método exclusivo de los turrónes devolverá *true* si todos los parámetros del mismo alcanzan el valor máximo posible (evidentemente, dentro de este cálculo no se incluye el sabor).
- magdalena Dual Core: Este método devuelve *true* si una magdalena pesa exactamente 65 gramos.

Con estos elementos se requiere:

Construir una clase que sea una "cinta de distribución friki", implementada con un array, que tenga las siguientes características:

- Podrá contener tanto magdalenas como turrónes (el alumno deberá determinar que forma de hacer esto es la que mejor se adapta a sus conocimientos y a este problema).
- Tamaño máximo indicado por el constructor. La cinta no crece, si se intentase meter un número de elementos superior al espacio disponible, deberá rechazarse la inserción mediante una excepción creada al efecto.
- Los elementos nunca se borran de la lista uno a uno, sino que la lista se vacía de una sola vez, entera. Al mismo tiempo, existe un indicador que dice cual es el siguiente elemento a obtener.
- Operador sobrecargado ">>" (sobre la lista) devuelve el elemento de la lista apuntado por el indicador mencionado anteriormente, avanzando dicho indicador una posición (por lo tanto, la siguiente llamada a ">>" sacara el siguiente elemento al ultimo sacado, etc).
- Operador sobrecargado "<<" (sobre la lista) que insertará un elemento a continuación del ultimo insertado (lanzando una excepción si no hay más sitio).
- Operador sobrecargado "<<" (sobre un objeto *ostream*): Imprimirá la totalidad de la lista, con cada uno de sus elementos.

Además de la clase anterior, se requerirá implementar una clase "controlador de operaciones friki", que recibirá cada uno de los elementos de la cinta anterior (de uno en uno), y cuya principal función será comprobar la validez de los elementos. Concretamente sus funciones serán las siguientes:

- Si lo que se recibe es un turrón, y pasa el proceso de validación, entonces se meterá en la "Lista de turrónes a distribuir".
- Si lo que se recibe es una magdalena, y pasa el proceso de validación, se meterá en la "Lista de magdalenas a distribuir".
- Si cualquier elemento no pasa el proceso de validación, se incluirá en la "Pila de elementos a destruir".

Una magdalena se considera válida si su altura es igual o superior a 9 cms, mientras que un turrón será valido si su peso es superior a los 500 gramos, su concentración de cores mayor del 60% y su longitud superior o igual a 12 cms.

La lista de turrónes a distribuir se puede implementar de la forma que mas le guste al alumno (incluyendo la posibilidad de usar *std:vector* u otra clase similar de la *std*), y deberá tener solamente las siguientes operaciones:

- Add: Para añadir elementos a la lista.
- Contar el número de unidades en la lista.
- Peso total de todos los elementos de la lista (suma de los pesos de todos los turrónes).
- Concentración media de cores de los turrónes existentes en la lista.
- Longitud media de las barras.

- Operador []: Para acceder a cada elemento.
- Numero de turroneos *extreme edition* existentes.
- Empaquetar: Vaciar la lista, suponiendo que todos los elementos han sido empaquetados para su distribución.

Al igual que la anterior, la lista de magdalenas es también de implementación libre como la anterior, y contendrá solamente las siguientes operaciones:

- Add: Para añadir elementos a la lista.
- Numero de unidades en la lista.
- Numero de magdalenas de cada sabor (se puede hacer implementando una operación por sabor o de cualquier otra forma que se os ocurra).
- Peso máximo: El peso de la magdalena que mas pesa dentro de la lista.
- Numero de magdalenas *dual core* existentes.
- Empaquetar: Vaciar la lista, suponiendo que todos los elementos han sido empaquetados para su distribución.

Por último, la pila de elementos a destruir deberá implementarse con punteros y contendrá solamente las siguientes operaciones:

- Operador "+" (evidentemente, para meter elementos).
- Consumir en el fuego del monte del destino: Destruirá todos los elementos de la pila.
- Operador "<<": Para sacar su contenido por pantalla.

El escenario de prueba que se propone para ejecutar conjuntamente todos estos elementos es:

- Crear una cinta de 8 posiciones:
- Crear un numero variable (entre 7 y 12) de turroneos y magdalenas, con parámetros aleatorios entre los límites dados. Se puede crear un 50% de magdalenas y un 50% de turroneos, de distintos sabores también aleatorios (recordar que los sabores se codifican con números).
- Introducir estos elementos en la cinta.
- Pasar cada elemento por el controlador de operaciones, para que determine si es o no válido.
- Una vez procesados todos los elementos, hacer lo siguiente:
 - Para la lista de turroneos: Sacar el numero de unidades, peso total, longitud media y concentración de COREs por pantalla y empaquetar los elementos, vaciando la lista.
 - Para la lista de magdalenas: Sacar por pantalla el numero de magdalenas total y por cada sabor, así como el peso máximo y empaquetar, vaciando la lista.
 - Para la pila de elementos a destruir: Volcar su contenido por pantalla y destruir posteriormente.

Ampliaciones (Opcionales):

- Escribir los elementos que se sacan de ambas listas y de la pila en ficheros aparte.
- Repetir este proceso múltiples ciclos (en la practica básica solo se requiere hacer esta secuencia de operaciones una vez, aunque se puede fácilmente volver a hacer todas las operaciones de nuevo formando un ciclo que se repita un numero arbitrario de veces).

Consejos/Normas:

- Abordar el ejercicio por pasos, no intentar hacer todo a la vez. Conviene además pensar que parte es mejor hacer primero y que parte después, para establecer un orden que os facilite su realización.
- Se recomienda hacer un pequeño esquema en papel de las clases y métodos a construir, para tener una idea del funcionamiento y arquitectura antes de empezar a programar.
- Usar herencia y polimorfismo en alguna parte del ejercicio (a elección del alumno).
- Allá donde se permita libertad de implementación, pensar cual es la forma de implementar esa parte que menos problemas puede dar.
- ¡El examen no será tan largo como este ejercicio! Y tampoco contemplara todas estas cosas, solo una parte de las mismas, en función del modelo que os toque.
- **Pista:** Como no hemos visto la forma de averiguar el tipo de un objeto, se puede incluir en la clase un método "*getTipo*", que devuelva si es un turrón o una magdalena.
- Evidentemente, se admitirán todo tipo de preguntas acerca de la realización de este ejercicio vía e-mail, tutorías, foro, etc.
- Ningún pastelero friki, magdalena o turrón ha sido dañado en la elaboración de este enunciado :).