Golosi

El problema

La empresa Golosi se dedica a empaquetar *chuches* en *cajitas sorpresa*. Cada *chuche* tiene un *valor calórico*, así como un *valor nutricional* (ambos valores son enteros no negativos). De esta forma, los valores calórico y nutricional de una *cajita* son, respectivamente, la suma de los valores calóricos y la de los valores nutricionales de las *chuches* que contiene.
Para producir las cajitas, Golosi utiliza una *hilera de producción* donde las diferentes *chuches* se disponen secuencialmente, una detrás de otra. Las cajitas en sí pueden contener cualquier número de *chuches*, pero, por restricciones del proceso de producción, las *chuches* en una cajita deben aparecer colocadas en posiciones consecutivas de la hilera de producción.
Debes desarrollar un algoritmo iterativo **eficiente** que, dados (i) un vector con los valores calóricos de las *chuches* en la hilera de producción; (ii) un segundo vector con los valores nutricionales de dichas *chuches*; y (iii) un *umbral calórico* (número entero no negativo), determine el máximo valor nutricional que pueden tener las cajitas cuyos valores calóricos no excedan el umbral calórico proporcionado.
En el desarrollo de este algoritmo puedes suponer que la hilera de producción tiene, como mínimo, una *chuche*. También puedes suponer que los valores calóricos de las *chuches* que aparecen en la hilera no exceden nunca el umbral calórico.

Trabajo a realizar

Para realizar el control se proporciona un archivo `plantilla.cpp` que contiene un programa que lee por la entrada estándar casos de prueba, y escribe por la salida el máximo valor nutricional conseguible.
Cada caso de prueba consta de cuatro líneas:

- En la primera línea aparece el número **n** de chuches que hay en la hilera de producción.
- En la segunda línea aparecen, en orden, los valores calóricos de cada chuche en la hilera.
- En la tercera línea aparecen, en orden, los valores nutricionales de cada chuche en la hilera.
- En la cuarta línea aparece el umbral calórico para las cajitas.

Los casos de prueba terminan con una línea que contiene `-1`.
A continuación, se muestra un ejemplo de entrada/salida:

Entrada	Salida
10 5 2 3 4 1 2 4 7 6 5 3 5 7 5 2 4 5 6 7 4 7 5 1 3 2 4 5 5 4 1 2 3 6 -1	**12** **10**

Vuestro trabajo consiste en:

- Especificar formalmente el algoritmo que calcula el máximo valor nutricional que puede conseguirse, rellenando los huecos correspondientes en los comentarios que rodean a `mejor_cajita`.
- Diseñar **sistemáticamente** una implementación C++ para dicho algoritmo. Debéis describir el diseño en el comentario habilitado a tal fin en el archivo proporcionado, siguiendo el método de diseño presentado en clase.
- Completar la función `mejor_cajita` con la implementación del algoritmo.
- Entregar vuestra solución a través del juez en línea de evaluación continua de la asignatura (`http://fal.fdi.ucm.es/`).

Importante:

- No modificar el código proporcionado. Únicamente deben responderse a los distintos apartados, en el interior de los comentarios, e implementar la función `mejor_cajita` (sí pueden incluirse funciones auxiliares, si se considera necesario).