Procesador de paquetes.

Objetivos.

- Aprender a implementar y utilizar el TAD Cola.
- Aprender a usar los tipos std::stack y std::list de la STL de C++.

Descripción.

Desarrolla un programa que simule un procesador de paquetes. Para ello se implementará un buffer con una capacidad máxima de paquetes que será gestionado con una cola (sin prioridad). Para implementar la cola puedes elegir entre usar dos pilas o una lista con acceso O(1) al último elemento. Para ambos TADs puedes utilizar los templates std::stack o std::list de la librería STL.

Cuando un paquete llega al procesador pueden ocurrir las siguientes circunstancias:

- Que esté vacío el buffer, por lo que el procesador pasa a procesar el paquete de forma inmediata.
- Que no esté vacío el buffer, pero haya aún espacio en él. En ese caso el paquete se encola para ser procesado en orden de llegada.
- Que el buffer esté lleno, en ese caso el paquete será desechado “drop”.

Notar que pueden llegar varios paquetes en el mismo instante de tiempo.

El programa debe mostrar para cada paquete, el instante de tiempo en el que comenzó a ser procesado o -1 indicando que el paquete fué desechado.

En el ficheros de tests, la primera fila indica el tamaño del buffer y el número de paquetes que se simulan. A continuación cada paquete (uno por línea) está modelado por una tupla de valores “a p” donde ‘a’ es el tiempo medido en milisegundos de llegada y ‘p’ es el tiempo en milisegundos necesario para procesar el paquete.

Por ejemplo si el fichero test de entrada es:

2 4
1 2
2 2
3 2
3 1

(La primera fila indica usar un <tamaño buffer>=2, <número de paquetes>=4, y las siguientes filas indican los 4 paquetes (uno por fila) con el formato <tiempo de llegada> <tiempo_procesado>)

La salida debería ser:
1
3
5
-1

Explicación: El primer paquete que llegó en tiempo 1, como el buffer estaba vacío pasó a procesarse directamente, con tiempo de terminación 3. Después llegó un segundo paquete en tiempo 2, que fue procesado en tiempo 3, cuando el proceso del paquete 1 terminó. Como el primer paquete aún no había terminado cuando llegó, el paquete se encoló con tiempo de terminación 5 (el tiempo de terminación del último paquete 3 + su tiempo de proceso 2).

Posteriormente, llegó un tercer paquete en tiempo 3. El paquete 1 ya había terminado su proceso (en tiempo 2) por lo que se desencoló. El paquete 2 aún no había terminado su proceso (termina en tiempo 5) pero como queda un hueco libre en el buffer, el paquete 3 se encola para ser procesado en tiempo 5 y con tiempo de terminación 7. En el mismo tiempo 3 llegó un cuarto paquete. Los paquetes 2 y 3 aún no habían terminado su proceso (terminan en tiempo 5 y 7 respectivamente) y como no quedaba sitio en el buffer (están los paquetes 2 y 3) se desechó (-1).