Universidad Simón Bolívar
Departamento de Computación y Tecnología de la Información
CI-2692 - Laboratorio de Algoritmos y Estructuras II
Septiembre-Diciembre 2014

Proyecto 2 (20 %)

Entrega: Martes de semana 8 **antes de las 11:59 PM** en el espacio de su grupo en el Aula Virtual.

1. Introducción

En este proyecto se pide que escriba un simulador de la planificación de operaciones en una empresa de mensajería. El simulador debe tomar una secuencia de paquetes de un archivo de entrada y simular su entrega de acuerdo a los parámetros abajo descritos. Debe implementar las clases que se especifican en las secciones siguientes.

Su programa se ejecutará mediante el comando especificado a continuación. Su módulo principal debe tener el mismo nombre.

```
> python3 simulador.py <arch-entrada> <arch-salida>
```

2. Descripción del simulador

2.1. Clase `Paquete`

Los objetos de esta clase representan un paquete, debe contener como mínimo los campos:

- `destinatario` (string): El destinatario del paquete, es un identificador único dado en la entrada del programa
- `prioridad` (entero): Prioridad del paquete, va del 1 al 5 siendo 1 la prioridad más alta.
- `capacidad` (entero): Capacidad requerida para llevar este paquete.

- `duracion` (entero): Tiempo que va a ocupar entregar este paquete.

Estos datos son leídos por la instrucción `paquete` del archivo de entrada (ver sección 3.1).

2.2. Clase cola de prioridades

Debe implementar el tipo cola de prioridades en una clase. Esta clase debe encapsular su funcionalidad. Al leer una instrucción *paquete* desde el archivo de entrada, se debe crear la estructura `paquete` e introducirla en el lugar adecuado de la cola de prioridades. Las prioridades van de 1 a 5, siendo la prioridad 1 la más alta. Esta estructura debe ser implementada sobre una lista enlazada.

2.3. Clase `Mensajero`

Este simulador tiene 4 mensajeros. Debe crear una clase para representar a los mensajeros del simulador. El mensajero tiene como mínimo los campos:

- `identificador`: Entero entre 1 y 4 identificando al mensajero actual.
- `paquetesEmpilados`: Es una pila (usted debe implementar la clase pila sobre lista enlazada) donde se van a empilar el paquete y los subpaquetes (si los tiene) tomados de la cola de prioridades. Al realizar la entrega del paquete que se encuentra en el tope de la pila, éste se desempila y se da inicio al siguiente.

La entrega de un paquete sólo se debe iniciar si hay capacidad suficiente disponible en el simulador. Se considera un paquete como entregado una vez hayan pasado suficientes unidades de tiempo desde el inicio de la entrega.

2.3.1. Subpaquetes

Algunos paquetes tienen la particularidad de que deben ser divididos en cierto número de subpaquetes. Cuando se cargue en el mensajero un paquete con uno o más subpaquetes, esos subpaquetes deben crearse y empilarse inmediatamente en el mismo mensajero. Éstos no requieren capacidad adicional, utilizan la capacidad del paquete origen.

Los subpaquetes heredan los atributos de su padre, salvo el destinatario. El nombre de un subpaquete se crea agregando al destinatario del padre el string "`-N`", donde N es un número agregado secuencialmente a cada hijo, empezando por cero. Los subpaquetes se deben empilar de manera inversa, es decir, se empilan en el orden $N-1, N-2, \ldots, 1, 0$.

Ejemplo:

Si se tiene al paquete p con 3 subpaquetes, la pila debe quedar:

```
p <-Tope de la pila
p-0
p-1
p-2
```

2.4. Clase Simulador

Esta es la clase principal del simulador, se encarga de:

- Leer el archivo de entrada: El formato del archivo de entrada está especificado en la sección 3.1.
- Llevar el control del tiempo: El simulador debe representar el momento actual de la simulación y notificar a cada mensajero cada vez que avanza el tiempo. Este tiempo no se refiere a una unidad de tiempo (mseg., etc.) en particular, sino a una indicación de la secuencia en la que ocurren los eventos. El tiempo en el simulador se cuenta a partir de 1.
- Asignar los paquetes (ver sección 2.1) de la cola de prioridades (ver sección 2.2) a los mensajeros de manera que se asigne el paquete al mensajero con el número más bajo disponible.
- Llevar la cuenta de la capacidad disponible. Este simulador tendrá disponible 1024 unidades de capacidad.
- Imprimir cada 100 unidades de tiempo un listado de los paquetes en la cola de prioridades y en la pila de cada mensajero. Ver en la sección 3.2 el formato de impresión.

  El simulador no toma tiempo para ejecutar sus acciones, ya sea agregar paquetes a la cola de prioridades ni empilar en el mensajero.

3. Archivos de Entrada y Salida

3.1. Archivo de entrada

El simulador lee del archivo especificado en el parámetro `arch-entrada` los paquetes que va a entregar. Cada paquete viene separado por una línea de la siguiente manera:

```
paquete destinatario prioridad capacidad duración subpaquetes
```

Donde:

- `paquete`: la palabra clave que identifica que esta línea introduce un paquete nuevo.
- `destinatario`: un identificador de tipo `string` alfanumérico de este paquete, no contiene espacios en blanco.
- `prioridad`: La prioridad para la entrega del paquete, de tipo entero con valores entre 1 y 5.
- `capacidad`: La capacidad requerida por el paquete, de tipo entero.
- `duracion`: La cantidad de unidades de tiempo del reloj que requiere este paquete para ser entregado, de tipo entero.
- `subpaquetes`: La cantidad de subpaquetes que tiene el paquete, de tipo entero, mayor o igual a 0.

3.2. Archivo de salida

Toda la ejecución del programa será registrada en un archivo de salida especificado en el parámetro `arch-salida`. En este archivo se registrarán las acciones mencionadas en las secciones anteriores.

3.3. Empilado de subpaquetes

Cuando un paquete tenga subpaquetes, se debe imprimir en el archivo de salida que se están empilando:

```
<tiempo> Empilando <N> subpaquetes del paquete <destinatario_del_paquete> en el mensajero <mensajero>.
```

Esto se debe imprimir ANTES de la impresión que indica el inicio de la entrega.

3.4. Entrega de paquete

Cuando el mensajero sale a entregar un paquete escribirá en el archivo de salida.

```
<tiempo>Iniciando paquete <destinatario> por el mensajero <mensajero>.
```

Al finalizar de ejecutar escribirá:

```
<tiempo>Finalizando paquete <destinatario> por el mensajero <mensajero>.
```

Note que el tiempo de entrega incluye el tiempo inicial y el final, ejemplo: Si un paquete que tarda 10 unidades de tiempo inicia en el tiempo 1 imprimirá lo siguiente:

```
1 Iniciando paquete <destinatario> por el mensajero <mensajero>
10 Finalizando paquete <destinatario> por el mensajero <mensajero>
```

3.5. Acción Listar

Cada 100 unidades de tiempo, se debe imprimir en el archivo de salida lo siguiente:

```
<tiempo> Listado
destinatario prioridad capacidad duración <estado>
Fin listado
```

Donde cada línea es un paquete que está en la cola de prioridades o lo tiene un mensajero, incluyendo los que están en la pila, ordenada por orden alfabético. `<estado>` puede ser:

- `Cola de prioridades` si el paquete está en la cola de prioridad.
- `Mensajero <N>` si el paquete está empilado en el mensajero `<N>`.

En los tiempos múltiplos de 100 el listado aparece antes de las impresiones por inicio o fin de entrega y empilado de paquetes, reflejando el estado del simulador antes de las acciones en el mensajero en ese tiempo.

3.6. Fin de la entrega

Al finalizar la entrega de todos los paquetes el simulador escribe al archivo de salida una señal de fin, que ocurre una unidad de tiempo después del último paquete en ser entregado.

```
<tiempo> Fin
```

4. Corrida de ejemplo

4.1. Archivo de entrada

```
paquete p1 1 100 1000 0
paquete p2 1 900 2000 0
paquete p3 2 1000 10 0
paquete p4 3 100 100 2
paquete p5 4 10 10 0
```

4.2. Archivo de salida

Nota: en esta salida sólo se muestra el listado en el tiempo 700 para facilitar la lectura. En una salida real se imprimiría el listado cada 100 unidades de tiempo.

```
1 Iniciando paquete p1 por el mensajero 1.
1 Iniciando paquete p2 por el mensajero 2.
1 Iniciando paquete p5 por el mensajero 3.
10 Finalizando paquete p5 por el mensajero 3.
700 Listado
p1 1 100 1000 mensajero 1
p2 1 900 2000 mensajero 2
p3 2 1000 10 Cola de prioridades
p4 3 100 100 Cola de prioridades
Fin listado
1000 Finalizando paquete p1 por el mensajero 1.
1001 Empilando 2 subpaquetes del paquete p4 en el mensajero 1.
1001 Iniciando paquete p4 en por mensajero 1.
1100 Finalizando paquete p4 por el mensajero 1.
1101 Iniciando paquete p4-0 por el mensajero 1.
1200 Finalizando paquete p4-0 por el mensajero 1.
1201 Iniciando paquete p4-1 por el mensajero 1.
1300 Finalizando paquete p4-1 por el mensajero 1.
2000 Finalizando paquete p2 por el mensajero 2.
2001 Iniciando paquete p3 por el mensajero 1.
2010 Finalizando paquete p3 por el mensajero 1.
2011 Fin
```

5. Requerimientos de la Entrega

1. El código debe estar adecuadamente documentado, lo cual incluye nombre de variables y métodos auto-explicativos en la medida de lo posible. Recuerde que el propósito de la documentación no es repetir información ya expresa en el código.

2. Todos sus archivos de código deben tener un encabezado con los nombres y números de carné de ambos miembros del equipo.

3. La entrega del proyecto debe realizarse ANTES de la fecha y hora indicadas al inicio del enunciado. Se recomienda realizar la entrega con antelación y considerar imprevistos como fallas de conectividad.

6. Preguntas y consultas

Se recomienda hacer preguntas con la mayor antelación posible. A fin de realizar las consultas de manera pública y persistente, coloque sus preguntas en el foro del Aula Virtual.