Prácticas
Estructuras de Datos Lineales

Elementos de Programación y Estructuras de Datos
Ma. Leticia Blanco Coca

8 de diciembre de 2014

Para todos los ejercicios debes utilizar adecuadamente Estructuras de Datos.

1. Dada una Lista de Simple Enlace:

   a) Contar la cantidad de elementos que tiene la lista.

   b) Aplicar un **Functor**que reduzca los elementos de la lista. Todo **Functor**, sabe reducir de dos modos: si recibe dos elementos devuelve uno y si recibe uno devuelve otro. Por ejemplo sea **f** un Functor, entonces se puede enviar los siguientes mensajes a **f**: **f.reducir(elem1, elem2)** o **f.reducir(elem)**. En ambos casos el resultado es un elemento.

   c) Invertir los enlaces de la Lista.

   d) Cocatenarla con otra lista de simple enlace. Debes tener cuidado de no tener elementos repetidos en la lista resultante.

   e) Dividirla en dos listas de longitudes lo mas similares posible. Asume que no puedes usar el metodo contar elementos.

   f) Que contiene elementos Comparables, realiza los métodos que permiten insertar lo elementos de forma ordenada ascendentemente.

2. Se desea modelar una solución para resolver el problema de atención al cliente en una entidad pública. Se sabe que los clientes vienen a la empresa por distintos motivos, básicamente por 5 tipos de quejas: q1, q2, q3, q4 y q5. Para ello la institución ha dispuesto 3 ventanillas de atención: vent1, vent2 y vent3. Las ventanillas vent1 y vent3 se especializan en atender quejas del tipo q2 y q5, en caso de emergencias y cuando existe demanda pueden atender quejas q3. La vent2 atiende las quejas q1, q3 y q4. La única vez que una ventanilla vent1 o vent3 atienden una queja q3, es cuando la vent2 solicita ayuda. En este caso se da la prioridad de atención al cliente con queja q3 en ya sea la ventanilla vent1 o vent3.

   a) Describe que estructura(s) de datos se adecúa mejor para resolver el problema.

b) Describe la estructura de los elementos que se almacenaran en la(s) estructura(s) de datos que elegiste.

c) Implementa la funcionalidad necesaria para permitir la llegada de los clientes a la entidad.

d) Implementa la funcionalidad necesaria para permitir la atención de los clientes.

e) Si además el gerente de la entidad requiere saber cual ha sido la productividad de cada una de las ventanillas, indica que cambios realizarias a tu anterior solución.

3. Dada una pila

   a) Invierte los elementos de la Pila

   b) Busca si existe un elemento

   c) Se quiere saber cual es el elemento que está en la base de la Pila

   d) Imagina un modo que permita migrar los elementos de la pila en una cola, de modo que el tiempo en el que ingresaron los elementos en la pila sea lo que rija el orden de inserción en la cola.

4. Se tiene una mesa redonda tradicional para compartir comida china. Luis es uno de los comensales que comparten de la comida, se desea saber en instantes especificos de tiempo qué plato tiene delante de él. Dado que los platos se encuentran en una plataforma que gira en el centro de la mesa, lo único que se sabe es que se tienen **n** comensales, y cada uno de ellos ha pedido una variedad distinta de plato. Entonces uno de los comensales puede servirse de alguno de los platos, para ello debe girar la plataforma central hasta tener el plato de su elección al frente. En ese instante de tiempo, obviamente es probable que Luis tiene delante de él algún plato particular, es en éste momento que el programa debería sacar un reporte de qué plato tiene en frente Luis.

   a) Describe que estructura(s) de datos se adecúa mejor para resolver el problema.

   b) Describe la estructura de los elementos que se almacenaran en la(s) estructura(s) de datos que elegiste.

   c) Implementa la funcionalidad necesaria para permitir la petición de un plato específico por uno de los comensales.

   d) Implementa la funcionalidad necesaria para permitir saber qué plato de comida tiene Luis al frente.

5. Sean A, B y C conjuntos de elementos, se desea:

   a) Realizar la union de A y B en C

   b) Realizar la intersección de A y B en C

   c) Realizar la diferencia de A y B en C

6. Dados dos polinomios, se desea:

   a) Sumarlos

   b) Multiplicarlos

7. Se desea saber si un mapa se puede colorear con dos colores, dado que se tiene la información de las conexiones entre las regiones del mapa. Las conexiones son duplas que se representan de la siguiente forma: $(reg1, reg2)$. Cada región tiene nombre y color de pintado (si estuviera pintado). Tu trabajo es decidir si el mapa puede ser o no pintado de dos colores.

8. En un gran restaurante de tres estrellas michelines, se sirven desayunos y como es de suponer, la presentación es un detalle que no debe descuidare a tiempo de servir todas las comidas. Una de las opciones de desayuno que tiene es panqueques franceses, el chef elabora un pedido de n panqueques y va poniéndolos en un platillo a medida que los cocina, al finalizar el chef deberia entregar el plato con los n panqueques puestos de forma que el panqueque de mayor diametro vaya siempre abajo y el de menor arriba. Ayuda a chef para saber cuantos movimientos se necesitan hacer para poder ponerlos en el orden estipulado, recuerda que como los panqueques están uno sobre otro, por lo que para cambiarlos de lgar probablemnete necesites voltear una pila de panqueques.

9. (60 ptos.) Se desea tener una aplicación que permita registrar las solicitudes de una tienda que sólo atiende a clientes via correo electrónico. Los pedidos que llegan deben tener un cliente, dirección - zona y una lista de objetos que se pide y cada objeto debería tener un peso que defina su posibilidad de atención y en base a la suma de los pesos de los objetos de los pedidos se define el orden de atención de estos pedidos. Los pedidos jamás pesan 100 ptos. y lo mínimo que pesan es 10 ptos. las solicitudes se organizan de acuerdo al peso que tienen, del 10 al 19, del 20 al 29, y asi sucesivamente del 90 al 99. Cada vez que se atiende el pedido, se debe a su vez registrar el pedido de acuerdo a la dirección del cliente. Al finalizar el día, la tienda, saca un listado que tiene los pedidos organizados por dirección - zona, cosa que al día siguiente su personal de reparto se haga cargo de la entrega de un conjunto de pedidos, se sabe que el personal de reparto se divide el trabajo por zonas

   a. (10 ptos.) Elige la(s) estructura(s) de datos más adecuada para representar el problema.

   b. (10 ptos.) Describe la estructura de los elementos que viven en la estructura de datos elegida.

   c. (20 ptos.) Escribe el (los) método(s) necesario(s) para poder atender el requerimiento de un cliente y registrarlo para su posterior entrega. Debes cuidar que la estructura que elijas debe ser lo más eficiente posible.

d. (20 ptos.) Escribe el (los) método(s) necesario(s) para poder encontrar la lista que permita planificar las entregas cada día.

10. (40 ptos.) Dada una cadena se pide encontrar su cadena extendida. Esta cadena se extiende por las vocales fuertes que tiene (se llaman vocales fuertes a la **a, e y o**), cada vez que aparece una vocal fuerte en la cadena esta se debe anteponer con las vocal debil más cercana y posponer la vocal debil mas lejana a ella. Por ejemplo: la cadena **extiende** se transforma en **ieuxtieundieu**, la única regla es que si ya una de las vocales debiles la antepone o pospone (en el orden) ya no debería considerarse, en el ejemplo, ante la segunda aparición de la **e** en exti**e**nde, ya esta antepuesta con **i** por lo tanto ya no es necesario ponerle de nuevo la **i**. Otro ejemplo es la palabra **cuando** que se transforma en **cuiaundiou**, en este caso la primera vocal débil mas cercana es **i** no **u**, por lo que se antepone la **i** a la **a** y se pospone la **u**.

    a. (10 ptos.) Elige la(s) estructura(s) de datos **MÁS ADECUADA** para representar el problema.

    b. (10 ptos.) Describe la estructura de los elementos que viven en la estructura de datos elegida.

    c. (20 ptos.) Escribe el (los) método(s)necesario(s) para dada una cadena encontrar la extendida.

11. (30 ptos.) En una fiesta de colores se ha decidido, invitar a personas que tiene que tener una pareja. Cada persona invitada, "elige" a su pareja y se supone que esta pareja debería venir con el mismo color de ropa que la persona que la eligió. Por supuesto, en esta tarea de elegir parejas es posible que una persona necesite vestirse de mas de un color - lo cual no es permitido.

    Dada una secuencia de personas invitadas y sus "elegidas", indica cuáles son las parejas posibles, e indica cuáles personas NO PUEDEN ir a la fiesta - aquellas que se les pide vestirse de más de un color.

    a) (5 ptos.) elige la estructura de datos mas adecuada para resolver el problema

    b) (5 ptos.) elige la estructura de los elementos que se almacenarán en tu estructura de datos

    c) (20 ptos.) escribe el (los) método(s) necesario(s) para indicar las parejas válidas y las personas que quedan fuera de la fiesta.

12. (45 ptos.) La embotelladora **bolem** necesita un programa que le ayude a distribuir sus productos estrella: **Koka&Kolla y Fantas-tiK** por la ciudad. Para ello acude a la UMSS en busca de personas que le solucionen el problema. Se decide que la modalidad de elección de los estudiantes que serán contratados para resolver el problema serán aquellos que puedan resolver el siguiente problema más sencillo.

La empresa tiene un camión que debe repartir sus productos en la ciudad a sus distribuidores. Los pedidos de productos se anotan en una cola. Para cada pedido se anota la cantidad de unidades y el distribuidor. Un distribuidor puede ser mayorista o minorista. El mayorista siempre pide el producto por cajas. El minorista las pide por cantidad de botellas. De cualquiera de ellos, **bolem** anota el nombre del distribuidor (que se denomina razón social) y su dirección. Estos pedidos deben ser puestos en la cola de espera por prioridad: los mayoristas son prioritarios a los minoritarios.

Para realizar el despacho se procede como sigue: hay que llenar el camión con cajas de bebidas. El camión puede llevar hasta 20 cajas de bebidas (cada caja tiene 12 botellas), aunque la empresa está pensando en la posibilidad de comprar un camión con mayor capacidad. El camión se llena con tantos pedidos como sea posible en el orden en que fueron catalogados para su atención. Es posible que un pedido no se pueda satisfacer totalmente, ya que el camión no tiene la suficiente capacidad como para atenderlo de una sola vez, entonces se carga el camión con tantas cajas como sea posible, dejando anotada la cantidad que queda pendiente a ser despachada.
En base a esta descripción se debe:

a. Elegir las estructuras de datos más **ADECUADAS** que permitan representar los pedidos y el camión de la empresa. **(5 puntos)**

b. Realiza un modelo que muestre las estructuras elegidas, la empresa y los elementos de las estructuras. **(10 puntos)**

c. Escribir el código en Java de los métodos necesarios para realizar una carga del camión a partir de los pedidos. **(30 puntos)**

Caso Prueba 1 Un ejemplo para el inciso c) es:

```
------------------------------------------------------------
*** LISTA DE PEDIDOS POR ORDEN DE LLEGADA***

Los Molinos c. flores 123, total pedido: 10, quedan: 10, unidad: botellas
Don Pele av. asia 4321, total pedido: 10, quedan: 10, unidad: cajas
Los Molinos c. flores 123, total pedido: 10, quedan: 10, unidad: botellas
Salteneria Tucuman av. argentina 10, total pedido: 50, quedan: 50, unidad: botellas
Supermercado Lo Caro c. francia 440, total pedido: 30, quedan: 30, unidad: cajas

PEDIDOS ALMACENADOS POR PRIORIDAD

Don Pele av. asia 4321, total pedido: 10, quedan: 10, unidad: cajas
Supermercado Lo Caro c. francia 440, total pedido: 30, quedan: 30, unidad: cajas
Los Molinos c. flores 123, total pedido: 10, quedan: 10, unidad: botellas
Los Molinos c. flores 123, total pedido: 10, quedan: 10, unidad: botellas
```

```
Salteneria Tucuman av. argentina 10, total pedido: 50, quedan: 50, unidad: botellas

PRIMERA SALIDA DEL CAMION
Cargando 10 cajas.
Cargando 10 cajas. Quedan 20 cajas pendientes al pedido 2

SEGUNDA SALIDA DEL CAMION
Cargando 20 cajas. Quedan pedidos pendientes

TERCERA SALIDA DEL CAMION
Cargando 1 cajas.
Cargando 1 cajas.
Cargando 5 cajas. No quedan pedidos pendientes
------------------------------------------------------------
```

13. **Fisgon**, ha tratado de capturar lo que se transmite en la red muchas veces, pero siempre ha fallado, debido a que **Vac** siempre encuentra alguna forma de evitar que esto suceda. **Vac** lo que hace es cifrar los mensajes que se transmiten y asi evitar que sean vistos por **Fisgon**. Se conoce que **Vac** cifra los mensajes por el siguiente método:

    a) El borra todos los espacios y marcas de puntuación del mensaje original

    b) El reemplaza todas las letras sucesivas idénticas por una instancia de la letra

    c) El inserta pares de letras idénticas en distintos lugares muchas veces y de forma aleatoria.

    La tarea que se debe hacer es el proceso inverso, es decir hay que tratar de restaurar un mensaje cifrado. Para esto, hay eliminar todos los pares de letras idénticas insertadas en el tercer paso del procedimiento.

    Entrada: Como entrada se tiene un texto que contiene un mensaje cifrado por **Vac**. El mensaje consiste de puras letras en minúsculas.

    Salida: La salida de restaurar el mensaje es el mensaje restaurado

    Ejemplo:

wwstdaadierfflitzzz	stierlitz
gguiesdrrkkkfoofi	uiesdki
eeexafaafmccessndd	examen

    a) Define la estructura de datos adecuada para representar el problema. (5 ptos)

    b) Define la estructura de los elementos de tu estructura de datos. (5 ptos)

    c) Escribe el (los) método(s) necesarios para dado un mensaje cifrado, me permita restaurar y obtener el mensaje restaurado. (30 ptos)

14. (40 ptos.) Una empresa de gestión de campa nas, se encarga de hacer entre muchas cosas, la elaboración de banderines multicolores, los mismos que deben cumplir que jamas en la tira resultante de banderines se tenga dos consecutivos del mismo color. Para ello, disponen de una secuencia de banderines de los colores de los que se desea construir la tira de banderines. Esta secuencia se da en forma desordenada y aletaoria en colores y numero, jamas en esta secuencia se tienen mas de 3 banderines de un mismo color seguidos.

    Por ejemplos: si se tiene la siguiente entrada de banderines de color, donde r = rojo, v = violeta y a = anaranjado:

    ```
    arrvvrvrarvaaarvva
    ```

    La salida correcta que permita tener una tira de banderines tricolor mezclado seria:

    ```
    arvrrvrvrarvaravava
    ```

    En la entrada, se tiene la misma cantidad de banderines de distintos colores, en el ejemplo de cada color se tiene 6 banderines, por lo que, siempre es posible tener una tira de banderines que cumplan las condiciones especificadas.

    Por otro lado, si se tiene la siguiente secuencia:

    ```
    arrvvrvrrvaaaraavv
    ```

    La salida correcta que permita tener una tira de banderines tricolor mezclado seria:

    ```
    aravrrvrvrvaravava
    ```

    a. Escoge la estructura de datos adecuada para resolver el problema (10 ptos).

    b. Describe la estructura de los elementos que almacenarás en las estructuras que elegiste en el inciso a (5 ptos.).

    c. Escribe el (los) método(s) necesario(s) para poder dad una secuencia de entrada de banderines conseguir la tira de banderines correcta (25 ptos).

15. (40 ptos.) Los navegadores web estándares continenen características para moverse hacia atras y hacia adelante entre las páginas recientemente visitadas. Una manera de implementar estas caracteristicas es usar dos pila para guardas el rastro de las paginas que pueden ser alcanzadas moviendose hacia atrás o hacia adelnate.

    Los siguientes comandos necesitan ser provistos:

BACK: Poner la pagina actual en el cima de la pila **adelante**. Retirar la pagina del tope de la pila **atras**, haciendo de ella la nueva pagina actual. Si la pila **atras** esta vacia, el comando es ignorado.

FORWARD: Poner la pagina actual en el tope de la pila **atras**. Retirar la pagina del tope de la pila **adelante**, haciendo de ella la nueva pagina actual, Si la pila **adelante** es vacia, el comando es ignorado.

VISIT $< url >$**:** Poner la pagina actual en el tope de la pila **atras**, y hacer que el URL especificado sea la nueva página actual. La pila **adelante** es vaciada.

QUIT: Permite salir del navegador.

Asuma que el navegador inicialmente carga la página web en la URL **http://www.acm.org/**

Este problema tiene varios ejemplos de como debería comportarse tu programa.

A continuación se muestra un ejemplo que tiene la siguiente secuencia de ejecución:

```
VISIT http://acm.ashland.edu/
VISIT http://acm.baylor.edu/acmicpc/
BACK
BACK
BACK
FORWARD
VISIT http://www.ibm.com/
BACK
BACK
FORWARD
FORWARD
FORWARD
QUIT
```

La salida seria:

```
http://acm.ashland.edu/
http://acm.baylor.edu/acmicpc/
http://acm.ashland.edu/
http://www.acm.org/
Ignored
http://acm.ashland.edu/
http://www.ibm.com/
http://acm.ashland.edu/
http://www.acm.org/
http://acm.ashland.edu/
```

```
http://www.ibm.com/
Ignored
```

Como podrás notar, las palabras: **VISIT, BACK, FORWARD, QUIT**, indican las diferentes acciones que se pueden hacer en el navegador. Y cada linea de la salida responde a cada una de las instrucciones. Cuando no es posible, ya sea ir adelante o atrás se debe generar un mensaje **Ignored**.

a. (10 ptos.) Escribe el (los) método(s) necesario(s) para poder responder la acción **VISIT**.

b. (10 ptos.) Escribe el (los) método(s) necesario(s) para poder responder la acción **BACK**.

c. (10 ptos.) Escribe el (los) método(s) necesario(s) para poder responder la acción **FORWARD**.

d. (10 ptos.) Escribe el (los) método(s) necesario(s) para poder responder la acción **QUIT**.

16. (30 ptos.) Lulú una niña muy inquieta, para navidad quiere regalarle un collar a su mejor amiga, pero no quiere que sea uno más del montón por lo que ha decidido comprar piedras de distinto color y unir piedras de distintos colores cada una. Pero ella jamas quiere formar el collar juntando piedras de color morado y negro, naranja y negro, ni juntar dos piedras del mismo color, pues le parece de mal gusto. Debes ayudar a Lulú a indicar si la cantidad y variedad de piedras que ha comprado le servirán para hacer un collar como ella quiere y cual seria la disposición de las piedras. En caso de no poder hacer el collar debes decir "No es posible".

    Por ejemplo:

    ```
    Caso 1:
    ENTRADA:
    {morado, negro, verde, morado, verde, negro, negro, morado}
    SALIDA:
    No es posible
    Caso 2:
    ENTRADA:
    {morado, negro, amarillo, verde, naranja, negro, rojo, blanco, amarillo}
    SALIDA:
    {morado, amarillo, negro, verde, naranja, rojo, negro, blanco, amarillo}
    Caso 3:
    ENTRADA:
    {morado, negro, negro, morado, verde, verde, amarillo, naranja, blanco, rojo}
    SALIDA:
    {morado, verde, negro, verde, negro, amarillo, morado, naranja, blanco, rojo}
    ```

    a. (10 ptos.) Elige la(s) estructura(s) de datos **MÁS ADECUADA** para representar el problema.

b. (10 ptos.) Describe la estructura de los elementos que viven en la estructura de datos elegida.

c. (20 ptos.) Escribe el (los) método(s)necesario(s) para dada la lista de piedras que tiene Lulú, intentar generar un collar de acuerdo a las necesidades y condiciones de Lulú.

17. (30 ptos.) Jolly Jumpers, es una notación que se asigna a una serie de números cuya distancia permite realizar saltos mas pequeños que la cantidad de números que tiene la secuencia. Por ejemplo: 3, 4, 5, 7, 2, 3 es un Jolly Jumper, ya que la cantidad de números que tiene es 6 y los saltos que debe hacer entre número y número siempre son menores que 6; del 3 al 4 salto de 1, del 4 al 5 salto de 1, del 5 al 7 salto de 2, del 7 al 2 salto de 5 y así sucesivamente.

    Realiza el programa que dado una serie de números enteros positivos, me permita decidir si es un Jolly Jumper o no y además indicar el (los) lugares donde se ha generado ruptura del Jolly Jumper. Es de más decir que una secuencia de 1 elemento es Jolly Jumper!!!

    Sobre la base de esta descripción se pide:

    a) (5 ptos.) identificar las estructuras de datos MAS adecuadas para modelar el dominio del problema.

    b) (5 ptos.) definir de forma clara la estructura de los elementos que contendrán las estructuras de datos antes elegidas

    c) (20 ptos.) implementar el (los) método(s) necesario(s) para dada una secuencia de números decidir si es o no Jolly Jumper y ademas registrar las rupturas en la secuencia si existiesen

    Por ejemplo:

```
ENTRADA                          SALIDA
3 4 5 7 2 3                      3 4 5 7 2 3          Jolly Jumper
2 4 6 1 7 12 24 21               2 4 6 1 7 12 24X 21  No Jolly Jumper
3 10 23 43 2 12                  3X 10X 23X 43X 2X 12 No Jolly Jumper
```

18. (40 ptos.) Se desea tener una aplicación que permita registrar las solicitudes de una tienda que solo atiende a clientes vía correo electrónico.

    Los pedidos que llegan deben tener un cliente, dirección - zona y una lista de items que se pide y cada item debería tener un peso que defina su posibilidad de atención y en base a la suma de los pesos de los objetos de los pedidos se define el orden de atención de estos pedidos. Los pedidos jamás pesan 100 ptos o más. y lo mínimo que pesan es 10 ptos.

    Las solicitudes se organizan de acuerdo al peso que tienen, del 10 al 19, del 20 al 29, y asi sucesivamente del 90 al 99. Cada vez que se atiende el pedido, se debe a su vez registrar el pedido de acuerdo a la dirección del cliente.

Al finalizar el día, la tienda saca un listado que tiene los pedidos organizados por dirección - zona, cosa que al día siguiente su personal de reparto se hará cargo de la entrega de un conjunto de pedidos, se sabe que el personal de reparto se divide el trabajo por zonas.

a. (5 ptos) Elije la(s) estructura(s) de datos más adecuada(s) para resolver el problema.

b. (5 ptos) Describe la estructura de los elementos (T) que viven en la estructura de datos elegida.

c. (15 ptos) Escribe el(los) método(s) necesario(s) para poder atender el requerimiento de un cliente y registrarlo para su posterior entrega. Debes cuidar que la estructura que elijas debe ser lo más eficiente posible.

d. (15 ptos) Escribe el(los) método(s) necesario(s) para poder encontrar la lista que permita planificar las entregas cada día

19. (30 ptos.)Se ha desarrollado una estructura de datos lineal que pretende mejorar la búsqueda de un dato. Esta estructura se llama ListaHilada, que es básicamente de un solo enlace al siguiente elemento, pero cada cuatro elementos éstos están enlazados de manera también simple hacia adelante. El modelo de la ListaHilada se muestra en la Figura 1.

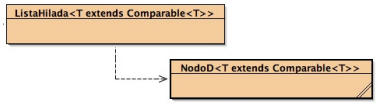

Figura 1: Modelo ListaEnlazada

Un ejemplo de cambio de estado en la ListaHilada haciendo inserciones se muestra en la Figura 2. Todos los elementos en la ListaHilada se tienen

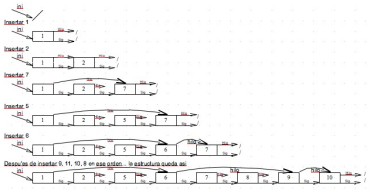

Figura 2: Ejecución de inserciones en ListaHilada

ordenados ascendentemente.

En el **Código 1** se tiene detalle de los atributos y operaciones de la ListaHilada.

```
public class ListaHilada<T extends Comparable<T>>
{
    private NodoD<T> ini;
    public ListaHilada()
    { ini = null; }
    public boolean vacia()
    { return ini == null; }
    public void insertar(T d)
    {   NodoD<T> p = new NodoD<T>(d);
        if (vacia())  ini = p;
        else          insertar(ini, p, 1, ini);}
    private void insertar(NodoD<T> q, NodoD<T> p, int i, NodoD<T> h)
    {// metodo que inserta a p caminando con q
    // y dependiendo del estado del hilo de h
    // y la posicion de i se actualiza el hilo}
    public T buscar(T d)
    {// HACER ESTO}
}
public class NodoD<T extends Comparable<T>> implements Comparable<NodoD<T>>
{
    private T dato;
    private NodoD<T> sig, hilo;
    public NodoD(T d){...}
    public NodoD<T> sig(){...}
    public NodoD<T> hilo(){...}
    public void sig(NodoD<T> s){...}
    public void hilo(NodoD<T> s){...}
    public void dato(T s){...}
    public T dato(){...}
    public int compareTo(NodoD<T> otro){...}
    public String toString(){...}
}
```

Realiza el método **T buscar(T d)**, que busca un elemento y lo devuelve si lo encuentra, caso contrario devuelve null. El método debe ser eficiente, aprovecha los enlaces de los hilos....

20. (30 ptos.) La sombra de un ArbolBB de tama no **d**, es aquel ArbolBB que puede formarse considerando los elementos que se encuentran hasta el nivel **d**. Realiza un método **ArbolBB**$< T >$ **sombra(ArbolBB**$< T >$ **orig, int d)**, que construya la sombra del ArbolBB **orig** dado que el tama no de la sombra es **d**.

21. (20 ptos.) Una librería quiere tener almacenados los items que tiene a la venta. Cada uno de estos items esta dentro de una clasificación y tiene precio por menor y por mayor, además de la unidad. Por ejemplo: hojas

existe una variedad, pero está clasificada como hojas. Las hojas por paquete tiene un precio y otro por caja. En este ejemplo el item es hojas, la descripción podría ser "de color" la unidad de venta por menor es paquete y la unidad de venta al por mayor es caja.

La libreria quiere tener velocidad cuando pregunta acerca de un item y su precio, por supuesto, el programa deberia indicar las posibilidades de oferta que tiene para vender.

a. (5 ptos.) Elige la estructura de datos más adecuada para representar el problema.

b. (5 ptos.) Describe la estructura de los elementos que viven en la estructura de datos elegida.

c. (10 ptos.) Escribe el (los) método(s) necesario(s) para poder atender el requerimiento de un cliente. Por ejemplo: hojas bond oficio.

22. (40 ptos.) Dada una cadena, se pide encontrar su patrón de composición reducido. Por ejemplo: si se tiene $aaaabbbbccdebbbffss$ el patrón reducido será $a4b4c2d1e1b3f2s2$.

    a. (10 ptos.) Elige la(s) estructura(s) de datos **MÁS ADECUADA** para representar el problema.

    b. (10 ptos.) Describe la estructura de los elementos que viven en la estructura de datos elegida.

    c. (20 ptos.) Escribe el (los) método(s) necesario(s) para dada una cadena encontrar la cadena patrón reducida.

23. (20 ptos.) Dada una Lista Circular de Simple Enlace tipificada como: $ListaCSE < Integer >$, se pide separar los elementos en dos Listas Circulares de Simple Enlace, dado un dato **n**, de acuerdo al siguiente criterio, todos los elementos menores a **n** se ponen en una de las listas y los que no lo son, en la otra.

24. (30 pts) Dada una torre que contiene elementos de dos tipos, se necesita organizar los elementos de la misma, de forma tal que siempre existan elementos intercalados. Se tiene como condicion de manejo que jamas se ingresa tres elementos del mismo tipo consecutivamente. La torre esta implementada en base a una pila, realice el (los) método(s) necesario(s) que permitan ingresar elementos a la torre de acuerdo a las especificaciones descritas.

25. La asignación de memoria es uno de los problemas más interesantes que existen dentro el desarrollo de los sistemas operativos y el mecanismo de GARBAGE COLLECTION también lo es. El GARBAGE COLLECTION consiste en liberar espacio de memoria que no es utilizado en un espacio de tiempo X.

En general la forma de organizar la memoria es por “bloques” de UN tamaño específico. Cada que se quiere “utilizar memoria”, se busca si hay un bloque que permita atender la solicitud. La solicitud de asignación me indica que ha fallado sólamente si ningún bloque tiene espacio , caso contrario se dice que ha sido exitosa. Cuando una asignación ha sido exitosa, sólamente permanece en la memoria si se lo usa en un periodo no mayor a 10 (este sería nuestro X). Por supuesto hay que cuidar que el espacio libre del bloque sea consistente despues de una asignación. Se tienen tres datos muy importantes: a) el tamaño de una solicitud siempre es menor que el tamaño de un bloque, b) que en un bloque jamás se asignan más de dos solicitudes y c) nunca dos solicitudes usando la memoria tienen el mismo tamaño.

Del mismo modo cuando se requiere acceder a un bloque, se indica que ha tenido éxito sólamente si el bloque está asignado con la solicitud buscada y si el tiempo de almacenamiento no ha pasado, caso contrario se dice que el requerimiento ha fallado.

Un dato importante es que jamás el bloque queda asignado por más de 10 periodos de tiempo sin haberlo utilizado, de tal manera que a tiempo de asignar el bloque se necesita que indiques el tiempo en el que se realizo la solicitud, del mismo modo cuando solicitas acceder a un bloque debes indicar el tiempo en el que estas intentando hacer el acceso.

Por ejemplo, si suponemos que se tiene una memoria organizada en 15 bloques cada uno de 500 de tamaño, todas libres al inicio y si se tiene un requerimiento de asignación que tiene el siguiente formato:

```
1 200
```

donde 1 es el tiempo en el que se pide asignar algun bloque de memoria y 200 es el tamaño de la solicitud.

La solicitud será exitosa y te dira en que bloque se asignó, en este caso se asigna en el primer bloque.

Si se tiene un requerimiento de acceso que tiene el siguiente formato:

```
7 10 200
```

donde 7 significa el tiempo en el que se requiere el acceso, 10 es el número de bloque que se quiere acceder y 200 es el tamaño de la solicitud, deberia decirme que el requerimiento falló ya que en ese bloque no hay nada asignado.

Pero si el requerimiento de acceso fuera:

```
7 1 200
```

el acceso es exitoso; ya que no ha pasado aún 10 periodos de tiempo y el bloque 1 está asignado con la solicitud 200.

Pero si el requerimiento de acceso fuera:

```
12 1 200
```

el acceso es exitoso; ya que no ha pasado más de 10 periodos de tiempo desde su asignación y/o último acceso

Pero si el requerimiento de acceso fuera:

```
27 1 200
```

el acceso falló; ya que ha pasado más de 10 periodos de tiempo desde su asignación y/o último acceso y el bloque 1 ya habria sido liberado automaticamente (GARBAGE COLLECTION).

Entonces si se tiene la siguiente secuencia de uso de memoria, considerando por ejemplo 15 bloques de 500:

ENTRADA	RESULTADO
1 200	exito
2 300	exito
3 400	exito
5 250	exito
6 350	exito
9 1 200	exito
13 1 250	fallo
20 2 300	fallo
21 1 200	fallo
22 1 250	fallo
22 150	exito
23 1 200	fallo
24 1 300	fallo
25 1 150	exito
30 2 300	fallo
31 3 400	fallo

Por supuesto, como podrás ver en el ejemplo, las solicitudes se hacen en orden cronológico. Puedes asumir que los números de bloque siempre son correctos.

a) elige la estructura de datos que más se adecue para resolver el problema. (5 ptos)

b) describe la estructura de los elementos que residiran en la estructura de datos que elegiste (realiza el modelo). (5 ptos.)

c) escribe el (los) método(s) que permitan asignar memoria a una solicitud(15 ptos.)

d) escribe el (los) método(s) que permitan acceder aun bloque de memoria. (15 ptos.)

26. En Las Vegas un juego muy popular es el blackjack que consiste en jugar con a lo sumo tres cartas contra el croupier (persona que reparte cartar y es personal del casino); el objetivo es que las cartas sumadas se aproximen sin sobrepasar a 21. Para ello la banca (se llama así a la mesa en la que se juega y se considera un maso de cartas completas) reparte 2 cartas a cada jugador incluyendo al crupier; los valores de las cartas son: el as vale 1 u 11, las figuras valen 10, y las cartas numéricas su valor natural; pudiendo cambiar el valor del as según la jugada.

    El jugador después de repartir las dos cartas tiene la posibilidad de plantarse (quedarse con las cartas que tiene) o pedir carta, sin pasarse del 21 ya que pierde automáticamente.

    Gana finalmente el que tenga el número más alto, cercano al 21, o saque un blackjack (sea igual a 21). Por su parte el croupier tiene reglas rígidas a las que atenerse: si su puntación inicial fuera 16 o menor está obligado a tomar otra carta, y se plantará siempre que su puntuación alcance 17 o mayor. Estas reglas las aplicará independientemente de las jugadas que tengan cada uno de los jugadores.

    El maso de cartas original, está formado por 'n' cartas las mismas que están barajadas de forma aleatoria; de este maso se va repartiendo la cartas los 'm'jugadores del juego. Jamás las cartas van saliendo del medio del maso siempre salen de la parte de arriba del maso que tiene las cartas volcadas para mantener la emoción del juego.

    Las cartas que hay en el maso son: (1 (as), 2, 3, 4, 5, 6, 7, 8, 9, 10, J, Q, K) cada uno de ellos en cuatro palos diferentes (C(corazon), D(diamante), T(trebol), E(espada)), en total existen 52 cartas en el maso original.

    Entonces en este juego hay dos rondas: la primera que permite repartir dos cartas a los jugadores, la segunda que permite repartir una carta más a los que deseen, salvo el croupier.

    Después de estas dos rondas se decide quién es el ganador del juego.

    a). elige las estructuras de datos adecuadas para resolver el problema (5 ptos.)

    b). describe la estructura de lo datos que contendrán tus estructuras de datos. (5 ptos.)

    c). escribe el(los) método(s) necesario(s) para realizar la repartija en la primera ronda a los 'm' jugadores, no olvides al croupier. (20 ptos.)

d). escribe el(los) método(s) necesario(s) para realizar la repartija en la segunda ronda a los 'm' jugadores, no olvides que las reglas del croupier son especificas y fijas en esta ronda. Para el resto de los jugadores puedes asumir que la solicitud de una carta más es aleatoria. (15 ptos.)

27. Se tienen las estructuras de datos ***p, b, a, l*** que son una pila de enteros, una bicola de enteros, un árbol binario de búsqueda de enteros y una lista de enteros, respectivamente:

```
Pila<Integer> p = new Pila<Integer>();
BiCola<Integer> b = new BiCola<Integer>();
ArbolBB<Integer> a = new ArbolBB<Integer>();
Lista<Integer> l = new ListaSE<Integer>();
```

Considera las siguientes instrucciones:

```
1        a.insertar(16);
2        a.insertar(20);
3        a.insertar(10);
4        a.insertar(40);
5        a.insertar(30);
6        vaciar(a.preorden(), p);
7        vaciar(p, b);
8        vaciar(b, a);
9        l = a.inorden();
```

El metodo vaciar es uno que permite pasar los elementos de una estructura (origen) a otra (destino), destruyendo la origen. Puedes asumir que su comportamiento es correcto.

a) muestra "graficamente" (dibuja), el estado de las estructuras ***a, p, b, l*** a medida que se realizan los procesos (10 ptos.)

b) la pila ***p*** está ordenada después de la instrucción 7? (5 ptos.)

c) al terminar el proceso (en la instrucción 9) el árbol ***a*** es el mismo en estructura que antes de los procesos vaciar? (5 ptos.)

28. En un juego denominado "*hola quien llama*? – *la llama que llama*", se ha planeado registrar todos los mensajes en un sistema. Lo que se desea es saber *que llama llama a que llama*. Todos los registros tienen un orden de aparición y se quiere saber *quien es la llama que llama*?, por lo que la posición de registro es importante. Por ejemplo, si se tienen los siguientes registros:

```
hola?
```

```
hola?
hola?
quien es la llama que llama?
quien es la llama que llama?
hola?
quien es la llama que llama?
quien es la llama que llama?
```

Se tendrá el siguiente reporte:

```
La llama 3
La llama 2
La llama 4
La llama 1
```

Considerando este ambito de aplicación:

a) (5 ptos) Elige la estructura de datos que se adecua mas para resolver el problema de manera eficiente

b) (5 ptos) Explica/define cuál es la estructura de los elementos que contendrá tu estructura de datos

c) (20 ptos.) Escribe el los métodos necesarios para permitir dado una lista de registros, definir *que llama llama*

29. Se quiere aumentar una operación a toda lista, que consiste en hacer un $split-division$ de la lista en otras mas pequeñas de tamaño n. Por ejemplo si se tiene la lista:

```
{2, 4, 5, 7, 2, 8, 10, 14, 19, 23, 59, 10, 14}
```

Y se quiere hacer un split de 2, se tendria el siguiente resultado:

```
{{2, 4}, {5, 7}, {2, 8}, {10, 14}, {19, 23}, {59, 10}, {14}}
```

Para realizar esta tarea puedes elegir la lista de tu preferencia: **ListaSE, ListaDE, ListaCSE o ListaCDE**. Lo único que debes cuidar es que si por ejemplo la lista original a la que le pides $split(n)$, es una ListaDE, entonces la lista resultante tambien lo será, además de cada una de las listas que aparecen en el resultado.

30. El manejo de los objetos en jdk, en particular, considera conceptos de asignación de memoria dinámica que puede ocasionar algunos funcionamientos "raros". Cuando una Cola de Prioridades permite realizar la operación $ver()$ retorna el dato del frente de la cola y desde donde lo solicitaron es posible cambiar el dato, por ejemplo observa el siguiente código:

```
 1 ColaPrioridad<Estudiante> cola = new ColaPrioridad<Estudiante>();
 2 Estudiante est;
 3 est = new Estudiante("Luis", 50);
 4 cola.encolar(est);
 5 est = new Estudiante("Maria", 30);
 6 cola.encolar(est);
 7 est = new Estudiante("Sara", 100);
 8 cola.encolar(est);
 9 est = new Estudiante("Jose", 10);
10 cola.encolar(est);
11 est = cola.ver();
12 est.setNota(50);
```

Para ubicar un estudiante en la cola, se utiliza su nota como prioridad. En la instrucción 11, se recupera el estudiante del frente y en la 12 se ha cambiado su información que es ademas su prioridad!!, cómo queda la cola? Sigue respetando la prioridad? Arregla lo que consideres necesario para que dinámicamente la cola pueda mantenerse siempre consistente, a pesar de que un dato que aun esta en la cola cambie.

31. Dado una Pila de elementos ordenables se te obtener los elementos en otra pila, de tal manera que los elementos queden ordenados de manera descendente, es decir el elemento mas pequeño en el tope de la pila. Recuerda no debes alterar las prestaciones de servicio de las pila