Control 1
INFO08 - Taller Estructuras de Datos y Algoritmos

Académicos: Erick Araya, Héctor Ferrada.
Instituto de Informática, Universidad Austral de Chile.
Mayo 25, 2020

Problema 1 (3.5 Pts.)

Ejecución: **./problema1** n k (*asuma que n es múltiplo de k*)
El fuente C1_2021_P1.cpp contiene los siguientes structs:

```
struct nodeList{
    int val;
    nodeList* next;
};
```

```
struct celdaS{
    int val;
    nodeList* p;
};
```

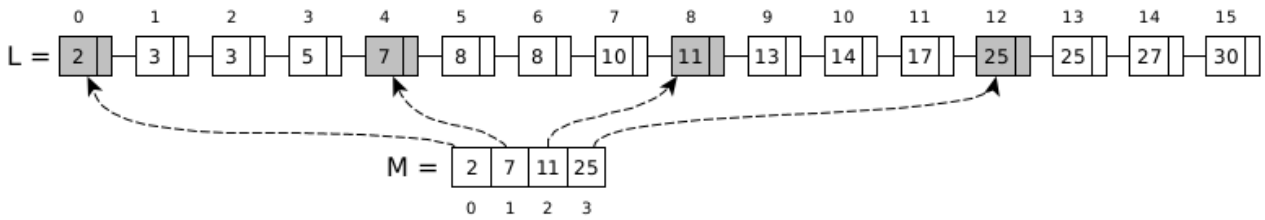
En este problema usted deberá realizar búsquedas de enteros almacenados en una lista enlazada L, de tipo nodeList, ordenada ascendentemente. Para agilizar las búsquedas construirá un arreglo M, de tipo celdaS, con algunas *muestras* de L. Cada celda de M posee también un puntero p al nodo de L que ha sido muestrado. Así, al buscar un dato x podrá hacer primero una búsqueda binaria de x en M para encontrar la mayor muestra $\leq x$ y por medio del puntero p ir directamente al nodo muestreado en L, para continuar su búsqueda con un recorrido secuencial en L a fin de determinar si existe o no x.

Realice:

(1.5 pts) i- En el main(), cree la lista L con n enteros aleatorios (positivos y menores que MAX, MAX definido en el fuente). Puede usar el método ***inserInList***(...) incluido en el fuente y que inserta en orden ascendente. Luego cree el arreglo $M[0 \ldots n/k - 1]$ de la siguiente manera: para cada uno de los n/k nodos de L cuya posición (de izquierda a derecha) sea múltiplo de k, es decir las posiciones $\{0, k, 2k, 3k, \ldots\}$, guarde en el campo $M[i].val$ el entero del nodo de L de posición $i \cdot k$, $\forall\ i = 0, 1, 2, \ldots, \dfrac{n}{k} - 1$, y en $M[i].p$ la dirección de memoria del nodo muestreado.

Imprima sus estructuras y luego pida al usuario el ingreso de enteros positivos e invoque al método ***buscaEntero***(L, M, n, k, x) por cada valor x ingresado. Este método debe buscar x y retornar verdadero o falso según se encuentre o no x en la lista L. Además debe imprimir si el dato fue encontrado o no. Permita que el usuario ingrese un 0 para terminar sus búsquedas y la ejecución del programa.

Como ejemplo de construcción, para $n = 16$ y $k = 4$ su estructura debe quedar como se muestra en la figura, en donde los nodos sombreados son los valores muestreados y que se deben almacenar en M, y las flechas desde M hacia L representan a cada puntero p de las celdas de M:



(2.0 pts) ii- Cree el método ***void buscaEntero***(nodeList *L, celdaS *M, int n, int k, int x); el cual debe buscar x de la siguiente manera:
1- Implemente una búsqueda binaria (BB) de x en M, que encuentre el mayor índice m tal que: $M[m].val \leq x$ (si $M[0].val < x$, entonces que $m = 0$). Imprima la muestra encontrada.
2- Luego prosiga su búsqueda secuencialmente en la lista L a partir del nodo $M[m].p$ y determine si x está o no en L. Note que esta búsqueda no puede visitar más de $k + 1$ nodos.

En el fuente dispone de los métodos para impresión tanto de la lista, ***printList***(...), como de las muestras, ***printMuestras***(...). Puede crear métodos adicionales si los necesita.

Ejemplo de ejecución.

```
 ./problema1 16 4
n = 16, k = 4
Lista L = 1 4 5 23 25 28 39 40 41 43 55 69 70 72 79 95
M[0..3] = 1 25 41 70
Ingrese valor a buscar, x: 30
Muestra encontrada por la BB = 25, luego se visitaron 3 nodos en L
30 No encontrado :(
Ingrese valor a buscar, x: 55
Muestra encontrada por la BB = 41, luego se visitaron 3 nodos en L
55 Encontrado !!
Ingrese valor a buscar, x: 0
### Fin Problema 1 ###
```

Problema 2 (2.5 Pts.)

Ejecución: ./problema2 n b

En el código se encuentra declarada la estructura nodeDoubleList, así como las expresiones MIN (1000000) y MAX (10000000).
El objetivo del problema es usar el contenedor vector de la STL. Primero ha de crear e imprimir un vector V de n elementos aleatorios entre MIN y MAX-1 (n es ingresado como argumento en la ejecución del programa). Las funciones a usar son ***creaVectorV***() y ***printVector***(). Mediante el vector V, ha de crear un nuevo vector X, que contendrá también n elementos. La función a usar es ***creaVectorX***(). Cada elemento $X[i]$ se obtendrá de la suma de los dígitos IMPARES de $V[i]$, haciendo uso de la función RECURSIVA ***reduceRecursivo***(). El nuevo vector también ha de imprimirse usando ***printVector***(), que ya está codificado.
Con los elementos de X, ha de crear e imprimir una lista doblemente enlazada usando los métodos ya codificados: ***appendToDoubleListR***() y ***printDoubleList***().
Finalmente, el programa ha de generar un nuevo número aleatorio, entre MIN y MAX-1 que, haciendo uso de la función ***reduceRecursivo***(), ha de crear el valor de un nuevo nodo a insertar en la lista doblemente enlazada ANTES del nodo b (b ingresado como argumento en la ejecución del programa). La función a usar para insertar el nuevo nodo es ***insertBeforeDoubleList***().

La siguiente imagen muestra dos ejemplos de la salida esperada para 12 datos:

```
C:\> ./problema2 12 21
Vector V:
5289383 1930886 8692777 5636915 5747793 2238335 9885386 3760492 3516649 2641421 9202362 1490027
Vector X:
20 19 30 23 38 14 23 21 18 2 14 17
Lista doblemente enlazada con valores del vector X:
20 - 19 - 30 - 23 - 38 - 14 - 23 - 21 - 18 - 2 - 14 - 17
Nuevo número aleatorio: 1368690
Número reducido a insertar en la lista, antes de 21: 13
Lista final:
20 - 19 - 30 - 23 - 38 - 14 - 23 - 13 - 21 - 18 - 2 - 14 - 17

C:\> ./problema2 12 5
Vector V:
5289383 1930886 8692777 5636915 5747793 2238335 9885386 3760492 3516649 2641421 9202362 1490027
Vector X:
20 19 30 23 38 14 23 21 18 2 14 17
Lista doblemente enlazada con valores del vector X:
20 - 19 - 30 - 23 - 38 - 14 - 23 - 21 - 18 - 2 - 14 - 17
Nuevo número aleatorio: 1368690
Número reducido a insertar en la lista, antes de 5: 13
El nodo con valor 5 no existe!!
Lista final:
20 - 19 - 30 - 23 - 38 - 14 - 23 - 21 - 18 - 2 - 14 - 17
```

Suba sus respuestas a siveduc en una carpeta comprimida cuyo nombre sea su rut (miRut.zio) a la tarea Control1.