Algoritmos y Estructuras de Datos I
Primer Cuatrimestre 2018

Guía Práctica 8
Algoritmos sobre secuencias

Departamento de Computación
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

1. Secuencias de enteros

Ejercicio 1. ★ Especificar y escribir un programa para el siguiente problema. Dada una secuencia de valores enteros, encontrar la meseta más larga. Decimos que un intervalo de valores consecutivos de la secuencia es una *meseta* si todos estos valores son iguales.

Ejercicio 2. ★ Especificar y escribir un programa para el siguiente problema. Dado un entero $n \geq 0$, retornar una secuencia con todos las secuencias de longitud n cuyos elementos son enteros entre el 0 y el 9. Por ejemplo: si $n = 3$ debe retornar $\langle 0,0,0 \rangle$, $\langle 0,0,1 \rangle$, ..., $\langle 9,9,9 \rangle$

Ejercicio 3. Escribir un programa para el siguiente problema. Dada una secuencia, definimos una *"infrasecuencia"* como una secuencia cuyos elementos son algunos de los elementos de la secuencia original, y aparecen en el mismo orden que en dicha secuencia.
Por ejemplo:

- $\langle 0,2,4 \rangle$ es infrasecuencia de $\langle 0,1,2,3,4 \rangle$
- $\langle 0,0,0,1,1,1 \rangle$ es infrasecuencia de $\langle 0,0,0,1,1,1 \rangle$
- $\langle 0,0,0,0 \rangle$ no es infrasecuencia de $\langle 0,0,0,1,1,1 \rangle$ porque la secuencia no tiene cuatro ceros
- $\langle 1,0 \rangle$ no es infrasecuencia de $\langle 0,1 \rangle$ porque no hay un 1 antes de un 0 en la secuencia original.

Dada una secuencia de números enteros, encontrar la infrasecuencia creciente más larga, es decir, aquella en la que los números estén ordenados de menor a mayor y no haya otra con la misma propiedad que sea más larga.

Ejercicio 4. Especificar y escribir un programa para el siguiente problema. Dadas dos secuencias de enteros sin repetidos, retornar una secuencia de pares con el producto cartesiano entre ambas entradas.

Ejercicio 5. ★ Especificar y escribir un programa para el siguiente problema. Dadas dos secuencias de enteros de igual longitud, retornar la *distancia de hamming* (cantidad de elementos que son distintos entre ambas entradas).

Ejercicio 6. Especificar y escribir un programa para el siguiente problema. *Quickselect:* Especificar e implementar el problema de encontrar el índice donde se encuentra el k-ésimo menor elemento de un vector de enteros. Se define k-ésimo menor como el valor en la posicion k si el vector estuviese ordenado de menor a mayor.

Ejercicio 7. ★ Especificar y escribir un programa para el siguiente problema. Dada una secuencia de enteros no ordenada, retornar el indice del vector tal que la suma de los elementos a la izquierda es igual a la suma de elementos a la derecha. Si no existe tal indice, retornar el -1.

Ejercicio 8. *Compresión de secuencias:*[★] Escribir un programa para el siguiente problema. Dada una secuencia de enteros retornar una secuencia de secuencias de 2 elementos $\langle begin, length \rangle$ tales que *begin* representa el principio de la subsecuencia de elementos consecutivos y *length* su longitud.
Por ejemplo:

- $\langle 1,2,3,4,5 \rangle$ retorna $\langle \langle 1,5 \rangle \rangle$
- $\langle 1,2,3,10,11 \rangle$ retorna $\langle \langle 1,3 \rangle, \langle 10,1 \rangle \rangle$
- $\langle 1,2,3,100,102 \rangle$ retorna $\langle \langle 1,3 \rangle, \langle 100,1 \rangle, \langle 102,1 \rangle \rangle$
- $\langle 5,4,3,2,1 \rangle$ retorna $\langle \langle 5,1 \rangle, \langle 4,1 \rangle, \langle 3,1 \rangle, \langle 2,1 \rangle, \langle 1,1 \rangle \rangle$

¿Es posible a partir de la secuencia de secuencias obtenida reconstruir la secuencia original?

2. Strings

Ejercicio 9. Dadas dos strings, escribir un programa que decida si el primero es una subsecuencia del segundo.

Ejercicio 10. ★ Dados dos strings, escribir un programa que encuentre la subsecuencia de strings común más larga, es decir, la secuencia que es subsecuencia de ambas y que es de mayor longitud.

Ejercicio 11. Dado un entero L y un string de longitud mayor a L, escribir un programa que encuentre el substring de longitud L que es el primero en orden alfabético entre todos los substrings de longitud L.

Ejercicio 12. ★ Escribir un programa que dado un String que representa una frase retorne el orden inverso de las palabras en el string. Ejemplo: Dado `"Hola#Mundo#Maravilloso"` retorna `"Maravilloso#Mundo#Hola"`.

Ejercicio 13. Escribir un programa que dado una secuencia de Strings retorne el resultado de justificar las líneas: Ejemplo: Dado

```
"justifying#lines#by########"
"inserting#extra#blanks#is##"
"one#task#of#a#text#editor.#"
```

Se convierte en:

```
"justifying#####lines#####by"
"inserting#extra##blanks##is"
"one##task#of#a#text#editor."
```

Ejercicio 14. Dado un string s y un entero positivo $c > 0$, se desea partir el string s en líneas de a lo sumo c caracteres. Asuma que no existen palabras en el string s con mas de c caracteres.
Ejemplo:

```
"Esta#es#una#cadena#de#caracteres#a#partir#en#muchas#lineas"
```

Debe retornar con $c = 10$

```
"Esta#es"
"una#cadena"
"de"
"caracteres"
"a#partir"
"en#muchas"
"lineas"
```

3. Ordenamiento, búsqueda lineal y búsqueda binaria

Ejercicio 15. ★ Consideremos el siguiente programa de ordenamiento, llamado ordenamiento por burbujeo (*bubble sort*):

```
for( int i = 0; i < a.size()-1; i++ ) {
    for ( int j = 0; j < a.size()-1; j++) {
        if( a[j] > a[j+1] ) {
            swap(a, j, j+1);
        }
    }
}
```

a) Describir con palabras qué hace este programa.

b) Proponer un invariante y variante para el ciclo principal.

c) Proponer un invariante y variante para el ciclo interno.

d) ¿Cuántas veces se ejecuta el swap del ciclo interior como máximo (i.e. en el peor caso)?

Ejercicio 16. ★ Ordenar los siguientes vectores utilizando los algoritmos de *ordenamiento por inserción*, *ordenamiento por selección* y *ordenamiento por burbujeo*. Para cada uno, indicar cuál (o cuáles) de los algoritmos utiliza una menor cantidad de operaciones que los demás:

1. $\langle 1, 2, 3, 4, 5 \rangle$
2. $\langle 5, 4, 3, 2, 1 \rangle$
3. $\langle 1, 3, 5, 2, 4 \rangle$
4. $\langle 1, 1, 1, 2, 2, 2 \rangle$
5. $\langle 1, 2, 1, 2, 1, 2, 1, 2 \rangle$
6. $\langle 1, 10, 50, 30, 25, 4, 6 \rangle$

Ejercicio 17. Escribir un programa para resolver los siguientes problemas:

1. Dado un vector de enteros desordenado, contar la cantidad de veces que aparece el número 0.
2. Dado un vector de enteros desordenado, encontrar el número que más veces aparece en el vector.
3. Dado un vector de enteros desordenado, calcular la diferencia entre el mínimo elemento del vector y el máximo elemento del vector.
4. Para cada uno de los incisos anteriores, ¿Cómo podemos reducir la cantidad de iteraciones que realiza el programa si el vector está ordenado?
5. Resolver los tres primeros incisos para el vector $\langle 3, 1, -2, 0, , 2, -2, -2, -2, 3, 10, 0, 4 \rangle$. Si tenemos que resolver los tres ejercicios de manera consecutiva, ¿Es conveniente ordenar primero el vector?

Ejercicio 18. ★ Escribir un programa para resolver cada uno de los siguientes problemas:

1. Dado un vector cuyos elementos son todos cero o uno, calcular la suma de los elementos del vector.
2. Resolver el mismo problema que en el inciso anterior si se sabe que el vector está ordenado.
3. Resolver el mismo problema si se sabe que el vector está ordenado, y que en lugar de cero y uno, los posibles elementos del vector son 15 y 22.

Ejercicio 19. Escribir un programa que implemente este problema:

`proc reconstruye` (in a: $seq\langle\mathbb{Z}\rangle$, out b: $seq\langle\mathbb{Z}\rangle$) {
  `Pre` $\{|a| = \sum_{i=0}^{|a|-1} a[i]\}$
  `Post` $\{|a| = |b| \land ordenado(b) \land (\forall i : \mathbb{Z})\big(0 \leq i < |b| \rightarrow_L 0 \leq b[i] < |a|\big) \land (\forall j : \mathbb{Z})\big(0 \leq j < |a| \rightarrow_L a[j] = \#apariciones(b, j)\big)\}$
}

Ejercicio 20. Dar un programa que resuelva este problema:

`proc dosMitades` (inout a: $seq\langle\mathbb{Z}\rangle$) {
  `Pre` $\{|a| \geq 2 \land \neg ordenado(a) \land (\exists i : \mathbb{Z})\big(0 \leq i \leq |a| - 2 \land ordenado(subseq(a, 0, i)) \land ordenado(subseq(a, i, |a| - 1))\big)\}$
  `Post` $\{ordenado(a)\}$
}

Ejercicio 21. ★ *Merge:* Dadas dos secuencias ordenadas, especificar el problema de retornar el apareo de ambas secuencias. Escribir un programa que implemente la especificación.

Ejercicio 22. Dada una secuencia ordenada, un valor n y un valor k, retornar la secuencia con los k valores más cercanos a n.

Ejercicio 23. Dado el siguiente programa para ordenar un vector de enteros entre los índices `left` y `right`:

```
void sort(vector<int> &arr, int left, int right) {
      int i = left, j = right;
      int tmp;
      int pivot = arr[(left + right) / 2];

      /* partition */
      while (i <= j) {
            while (arr[i] < pivot) {
                  i++;
            }
            while (arr[j] > pivot) {
                  j--;
            }
            if (i <= j) {
                  tmp = arr[i];
                  arr[i] = arr[j];
                  arr[j] = tmp;
                  i++;
                  j--;
            }
      };

      /* recursion */
      if (left < j) {
            sort(arr, left, j);
      }
      if (i < right) {
            sort(arr, i, right);
      }
}
```

1. ¿Cuál debería ser la especificación de este programa?
2. Dado un vector ya ordenado de menor a mayor, ¿cuántas iteraciones se ejecutan?
3. Se desea utilizar este programa en un contexto en el cual la subsecuencia inicial del mismo está ordenada. Optimizar el programa para que aprovechar este contexto.

Ejercicio 24. ★ Dada una secuencia de enteros no ordenada, escribir el programa que retorne cuántos elementos deben ser intercambiados para que esté ordenada.