Control 2 - Forma B
INFO08 - Taller Estructuras de Datos y Algoritmos

Académicos: Erick Araya, Héctor Ferrada.
Instituto de Informática, Universidad Austral de Chile.
Agosto 13, 2019

Heap (3.0 pts.)

Ejecución: ./problemaB1 n

En el fuente problemaB1.cpp está definido el método *makeMaxHeap*, que es el mismo que se estudió en clases. Este método convierte al arreglo de enteros X en un *maxheap*. También existe una estructura de tipo *VAL*, la cual posee dos campos que se deben inicializar con los valores indicados en el mismo código, esto es al crear los datos:

```
struct valores {
    int val1;       //un entero aleatorio en el intervalo [0,3M]
    int val2;       //Si val1 < 0: un entero aleatorio en el intervalo [M,2M].
                    //Sino un entero aleatorio en el intervalo [−M,M]..
}; typedef struct valores VAL;
```

Se pide que en la rutina *main* usted debe crear un arreglo de largo n, de tipo *VAL*, llenar el arreglo con datos aleatorios. Luego debe invocar a la función *makeMaxHeapVAL(VAL *X, int n);* la cual debe de convertir el arreglo X en un *maxheap* de tipo *VAL*, considerando que el valor de la prioridad de cada celda corresponde al valor absoluto de la suma entre *val*1 y *val*2.

Hashing (3.0 pts.)

Ejecución: ./problemaB2 n m

Un *motif*, en una secuencia de ADN, es un pequeño substring que tiene una función biológica relevante. Ud deberá clasificar secuencias de ADN por la cantidad de *motif*'s que poseen. En el fuente problemaB2.cpp se define un diccionario (de nombre *DNA*) con las 4 letras para secuencias de ADN, y el *motif* declarado como: char $MOTIF[] =$ "AG", que usará en las búsquedas. Considere como una ocurrencia del *motif* tanto a "AG" como "GA". Ej.: "$ATGAG$" tiene 2 ocurrencias: "GA" y "AG". Así, una secuencia de largo m puede contener como máximo $m-1$ *motif*'s. Usted utilizará una estructura de *hash* para esta clasificación, con una tabla de tamaño m, donde la clave de cada cadena es el número de *motif*'s que esta contiene. Use listas enlazadas para las colisiones, en base a la siguiente estructura:

```
struct Sequence {
    char *seq;    //una combinación de m letras del diccionario DNA
    struct Sequence *next;
}; typedef struct Sequence ADNSEQ
```

Se pide que en la rutina *main* cree la tabla *hash* T (inicialice con NULL todas sus celdas). Luego genere n secuencias de ADN aleatorias de largo *mobastias@anid.cl* e invoque al método de inserción: *int insertInT*(*ADNSEQ* **T, int m, char *s). Codifique este método para que inserte s en su estructura de *hash*, donde la *key* de cada cadena es el número de *motif*'s que contiene. El fuente incluye al método *void printT*(...); el cual le ayudará a visualizar lo que tiene en su estructura.