Tarea 1
INFO053 - Estructuras de Datos y Algoritmos.

Académico: Héctor Ferrada.
Instituto de Informática, Universidad Austral de Chile.

Mayo 04, 2018

Entrega. Debe enviar todo en un archivo, tarea1Alumno1Alumno2.zip (máximo 2 integrantes), con su implementación (que incluya el Makefile) y un informe breve, por correo con asunto INFO053 TAREA 1 a **jorge.delgado01@alumnos.uach.cl**, a más tardar el viernes 25 de Mayo.

Informe. De a lo más 3 páginas, el cual debe explicar la lógica que uso en sus implementaciones y la justificación de ello, p.ej. porque no otra alternativa más eficiente (la eficiencia es un factor importante en la evaluación). También entregar las conclusiones y anexar complementos si los hubiese (como gráficas).

Problema 1. Arreglo no Decreciente

Dado n leído desde los argumentos del programa, cree un arreglo de enteros no dececientes $A[0..n-1]$; tal que $A[i+1] = A[i] + inc$; $\forall\ i, 0 \leq i \leq n-1$; con $A[0] = 0$. El valor inc es un valor aleatorio, tal que $0 \leq inc < M$, donde M debe encontrarse definido en la cabecera del programa (p.ej #define M 5). Se pide:

1. Cree una función int posAprox(int x, "más todos los parámetros que necesite") que, con una lógica sencilla, devuelva la posición aproximada de x en A ($\forall\ x \geq 0$).
   Indicación: no es necesario recorrer el arreglo para esto. Imagine cuando usted necesita buscar una palabra en un diccionario, la página que abre al comienzo es precisamente el valor dado por posAprox.

2. Cree una función bool isXinA(int *A, int n, int x, int *pos, int *c); que retorne verdadero si x esta en A y en *pos su posición, o falso si no esta. Para esto use la función anterior y a partir del valor dado por posAprox() recorra secuencialmente la menor cantidad de celdas del arreglo hasta determinar si x esta o no en A y escriba en *c la cantidad de celdas que le fue necesario recorrer en su búsqueda.

3. Implemente en bool bbXinA(int *A, int n, int x, int *pos, int *c); la típica búsqueda binaria en un ciclo for o while (que realiza a lo más $\approx \lg n$ iteraciones) y escriba en *c la cantidad de celdas que le fue necesario comparar en su búsqueda (asegúrese que ambas funciones retornan las mismas posiciones).

4. En un ciclo for de $t = 10^6$ repeticiones, invoque a ambas funciones: isXinA(int *A, int n, int x, int *$p1$, int *$c1$) y bbXinA(int *A, int n, int x, int *$p2$, int *$c2$) con el mismo valor aleatorio para x, $0 \leq x \leq A[n-1]$ y calcule el promedio para $c1$ y $c2$. ¿Qué puede decir sobre estos resultados?

Problema 2. Pilas y Colas

En un hospital muy desorganizado, se acaba de producir una gran emergencia. Han llegado al mismo instante cientos de enfermos y heridos por diversas causas, muchos de ellos provenientes de accidentes de transito debido a una colición multiple en la autopista central del país. Esto ha llevado a organizar una política para atender a los más graves primeramente. Se reconocen 3 niveles de prioridad: 1, 2 y 3, siendo el 1 los de mayor gravedad y los 3 los de menor urgencia. Para el grupo más prioritario también influirá la edad, siendo los niños (menores 15 años) los más prioritarios, luego los de 75 a 90 años (suponga 90 como edad máxima), luego de 60 a 75 y luego el resto en cualquier orden. Suponga que los n pacientes están inicialmente en una gran cola que almacena a cada paciente con su edad y prioridad; Se pide deshacer esta gran cola y almacenar a los pacientes en estructuras separadas según los siguientes criterios:

1. En una Cola $Q3$ deben quedar los pacientes de prioridad 3, dejando a los pacientes en el mismo orden que tenía la cola inicial.

2. En una Cola $Q2$ deben quedar los pacientes de prioridad 2, dejando a los niños al frente de la cola (para que sean atendidos primero), luego a los ancianos mayores de 75, luego a los ancianos mayores de 60 y luego el resto de los pacientes en el mismo orden que tenía la cola inicial.

3. En pilas $P_1, P_2, ..., P_6$ a los pacientes de prioridad 1 que son los más importantes en términos vitales. En P_1 deje a los niños, en P_2 a los mayores de 15 y menores de 30, en P_3 a los de entre 30 y menores de 45, P_4 para el rango $[45 - 60)$, P_5 para $[60 - 75)$ y los más mayores en P_6.

La clasificación y construcción de las pilas debe ser hecha de forma extremadamente eficiente, por tanto una vez que se tenga a un paciente de prioridad 1, no se debe preguntar por la edad que tenga —ya que una consulta $if()$ es una operación muy costosa en términos de tiempo y se necesita hacer esto más rápido. Por tanto debe de idear una forma para enviar a un paciente de prioridad 1 directamente a la cola P_i que corresponda a su edad sin preguntar por la edad que este tenga.