Ejercicio 8

a) Para lo que es el rendimiento determinístico, se entiende que el peor caso ocurre cuando (teniendo r arreglos que no contienen a x) estos r arreglos son los últimos que se consultan, al ser determinista, esta restricción se cumple

Notemos que para cada uno de los k elementos se realiza una búsqueda binaria y una comparación más, esto es, para un i, $\log_2(n_i)+1$.

Luego hay k arreglos

$$\Rightarrow \underbrace{\sum_{i=1}^{k}(\log(n_i)+1) \leq k(\log(\tfrac{n}{k})+1)}_{\text{identidad dada por el profe}} \quad ; \quad n = \sum_{i=1}^{k} n_i$$

Pero hay r (fijo) listas que no tienen al elemento x, entonces el peor caso será $(k-r+1)(\log(\frac{n}{k})+1)$ para el caso determinista.

b) El peor caso para el algoritmo aleatorizado se da cuando el hecho de que sea aleatorizado no es relevante, es decir, cuando $r=0$. En este caso, se tendrá que se tiene que realizar la búsqueda en todos los arreglos y no se detiene hasta cumplir esto. ~~El orden~~ La cota será

$$\sum_{i=1}^{k}(1+\log n_i)$$

c) Tomemos en cuenta que tenemos r arreglos que no contienen a x. Si se escoge uno arreglo al azar se tiene una probabilidad de

$$P(x \text{ no en } A_i) = \frac{r}{k}$$

y el programa termina.

Pero qué ocurre si a sacar este, no es del grupo (es decir, es un A_j donde A_j sí contiene a x). En este caso se vuelve a tener una oportunidad, pero esta vez

$$\mathbb{P}(x \notin A_1 \setminus A_j) = \frac{r}{k-1}$$

Y así sigue, siendo esta la unión de todas las probabilidades, siendo que la cota inferior de ~~que~~ este algoritmo ~~acabe es~~ es:

$$\sum_{i=1}^{k-r} \frac{r}{k-i} \cdot (1 + \log(n_i))$$

$k-r$ → porque termine antes (por r)

Que es la probabilidad de éxito (que termine).