1. Cap 1: Introduction

1.1. Ejercicios 1.3.3.1

1.1.1. There are n webpages ($1 \leq n \leq 10M$). Page i has a page rank r_i. A new page ca be added or an existing page can be removed frequently. You want to pick the current top 10 pages with the highest page ranks, in order. Which method is better?

Load all n webpages page rank to memory, sort them in descending page rank ordern, and obtain the current top 10.

1.1.2. Given a list `L` with $100K$ integers, you need to frequently obtain `sum(i, j)`, i.e., the sum of `L[i] + L[i+1] + ... + L[j]`. Which data structure should you use?

Simple Array pre-processed with Dynamic Programming.

1.1.3. Given an $M \times N$ integer matrix $Q(1 \leq M, N \leq 70)$, determine if there exists a submatrix S of Q size $A \times B(1 \leq A \leq M, 1 \leq B \leq N)$ where $\texttt{mean}(S) = 7$. Which algorithm will not exceed $100M$ operations per test case in the worst case?

No sé.

1.1.4. Given a multiset S of $M = 100K$ integers, we want to know how many different integers that we can form if we pick two (no necessarily distinct) integers from S and sum them. The content of multiset S is prime numbers not more than $20K$. Which algorithm will not exceed $100M$ operations per test case in the worst case?

No sé

1.1.5. You have to compute the shortest path between two vertices on a weighted Directed Acyclic Graph (DAG) with $|V|, |E| \leq 100K$. Which algorithm(s) can be used?

- Breadth First Search.
- Dijkstra's

1.1.6. Which algorithm produces a list of the first $10M$ prime numbers with the best time complexity?

Sieve of Eratosthenes.

1.1.7. How to test if the factorial of n, i.e., $n!$ is divisible by an integer m ? $1 \leq n \leq 10^{14}$

- Test if $n!\,\%m == 0$

No funcionaría porque no se puede computar $10^{14}!$. Sin embargo, con mis conocimientos actuales, desconozco otra manera de hacerlo.

1.1.8. You want to enumerate all occurrences of a substring P (of length m) in a (long) string T (of length n), if any. Both n and m have a maximum of $1M$ characters. Which algorith is faster?

```
for (int i = 0; i < n-m; ++i){
    bool found = true;
    for (int j = 0; (j < m) && found; ++j  )
        if ((i+j >= n) || (P[j] != T[i+j]))
            found = false;
    if (found)
        printf("P is found at index %d in T\n", i);
}
```

1.1.9. Given a set S of N points scattered on a 2D plane ($2 \leq N \leq 5000$), find two points $\in S$ that have the greatest separating Euclidean distance. Is an $O(N^2)$ complete search algorithm that tries all possible pairs feasible?

Yes, such complete search is possible.

1.1.10. See Questin above, but now with a larger set of points: $2 \leq N \leq 200K$ and one additional constraint: The points are randomly scattered on a 2D plane

No sé

1.1.11. See the same Question above. We still have a set of $2 \leq N \leq 200K$ points. But this time there is no guarantee that the points are randomly scattered on a 2D plane.

No sé

1.2. Ejercicios 1.3.5.1

1.2.1. You receive a WA verdict for a very easy problem. What you should do?

- Create tricky test cases to find the bug.

Esto debido a que cuando el problema es muy fácil se tiende a omitir casos de prueba que no se toman en cuenta al momento de realizar la solución. La mayoría de veces estos casos especiales suelen ser un `if`, por lo tanto, son fáciles de sacar.

1.2.2. You receive a TLE verdict for your $O(N^3)$ solution. However, the maximum N is just 100. What you should do?

- Improve the performance of your solution (code optimizations/better algorithme)

A pesar de que el peor algoritmo que sea AC para un N de 100 es uno con $O(N^4)$, el TLE pasar por tiempo, entonces se buscaría una mejor optimización o mejor algoritmo para el código.

1.2.3. Follow up to Question above: What if the maximum N is 100 000?

Para un N tan largo de 100 000 se buscará un algoritmo con una complejidad menor a $O(n^{1.5})$. Es decir, la opción:

- Improve the performance of your solution (code optimizations/better algorithm).

1.2.4. Another follow up Question: What if the maximum N is 5000, the output only depends on the size of input N, and you still have four hours of competition time left?

Si la solución a la que pude llegar es de $O(N^4)$ y para un N máximo de 5000 se necesita un algoritmo de $O(n^2)$, entonces pasaría a un problema mucho más fácil teniendo en cuenta que faltando 4 horas de competencia es que hasta ahora ha empezado y toca ir por el problema de regalo de la competición.

1.2.5. You receive an RTE verdict. Your code (seems to) execute perfectly on your machine. What should you do?

Con este *Run Time Error* (RTE) yo verificaría la versión a la que estoy compilando (o ejecutando) mi lenguaje de programación y la pondría a comparación con la que está en el documento explicativo con las versiones de los lenguajes. También, si mi lenguaje requiere de compilarse, verifico con qué opciones estoy compilando el programa y las comparo con el aplicativo.

1.2.6. Thirty minutes into the contest, you take a glance at the scoreboard. There are *many* other teams that have solved a problem X that your team has not attempted. What should you do?

En las competencias siempre hay un problema de "*regalo*", por lo tanto, si veo que muchos equipos ya completaron el problema X, le diría al equipo (o a uno de ellos) que lo vaya solucionando si es muy fácil, o también, hacerlo entre todos, aunque si el equipo tiene buen nivel, se puede dejar a una persona haciendo el problema de regalo y los otros ir pensando los otros.

1.2.7. Midway through the contest, you take a glance at the scoreboard. The leading team (assume that it is not your team) has just solved problem Y. What should you do?

Seguir en el problema que estaba, si algo, le digo a un miembro del equipo que le eche una mirada al problema Y y si está posible de realizar, entonces le damos, si no, a seguir con el que estábamos.

1.2.8. Your team has spent two hours on a nasty problem. You have submitted several implementations by different team members. All submissions have been judged incorrect. You have no idea what's wrong. What should you do?

Podría tomar dos diferentes caminos:

1. Pedir un descanso en el equipos para tomar aíre y tener ideas más frescas.
2. Seguir a otro problema y dejar este para lo último de la competencia.

1.2.9. There is one hour to go before the end of the contest. You have 1 WA code and 1 fresh idea for *another* problem. What should you (or your team) do?

Si mi equipo tiene un nivel decente, pondría a la persona con la idea a hacer este nuevo problema y que los otros sigan debuggeando el problema con el WA.