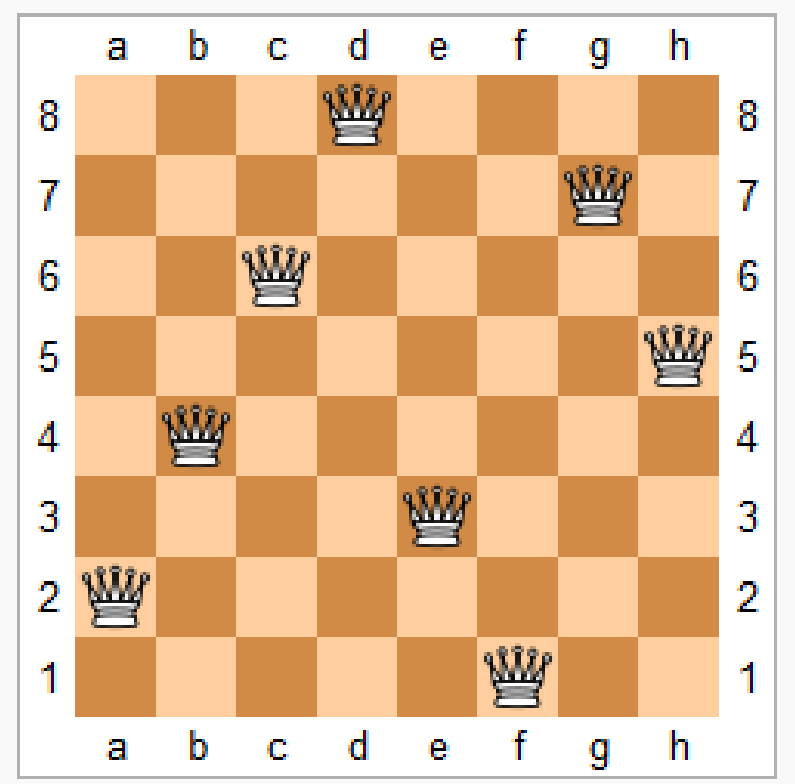
Parte I

A) Implementar un algoritmo de Hill Climbing (versión canónica) para resolver el problema de las n-reinas.



1. El algoritmo deberá ser capaz de encontrar solamente una solución para tableros de diferentes tamaños.
2. Una posible estructura para representar el tablero consiste en un arreglo de tamaño N, donde en cada posición hace referencia a una columna de tablero. Y cada valor hace referencia a una fila.
3. Se define una función objetivo $H(e)$ la cual contabiliza la cantidad de pares de reinas amenazadas para un tablero e.
4. Se deberá definir una variable que establezca el número máximo de estados que podrán ser evaluados.
5. El programa deberá devolver el tablero solución (únicamente la estructura que representa el tablero). Junto a la cantidad de estados que tuvo que recorrer el algoritmo para llegar a la solución. En caso de alcanzar el máximo de estados

evaluados, devolver la mejor solución encontrada y el valor correspondiente de la función H.

6. Replicar el punto 5 para los casos de las 4,8,10 reinas

B) Implementar el algoritmo Simulated Annealing para resolver el problema del punto A.

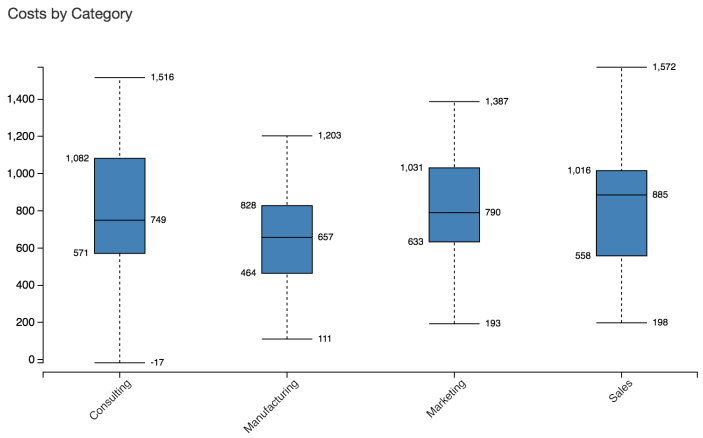
C) Implementar un algoritmo genético para resolver el problema del punto A. Ademas de la implementación en código del mismo, se deberán incluir detalles respecto a

1. Definición de los individuos de la población
2. Estrategia de selección
3. Estrategia de reemplazo
4. Operadores.

Parte II

A) Ejecutar cada uno de los algoritmos implementados en la parte I 30 veces y calcular para el caso de 4, 8,10,(**12,15)**? reinas:

1. El número (porcentaje) de veces que se llega a un estado de solución óptimo.
2. El tiempo de ejecución promedio y la desviación estándar para encontrar dicha solución. (se puede usar la función `time.time()` de python)
3. La cantidad de estados previos promedio y su desviación estándar por los que tuvo que pasar para llegar a una solución.
4. Generar un tabla con los resultados para cada uno de los algoritmos desarrollados y guardarla en formato .csv (comma separated value)
5. Realizar un gráfico de cajas (boxplot) que muestre la distribución de los tiempos de ejecución de cada algoritmo. (ver gráfico de ejemplo)



B) Para cada uno de los algoritmos, graficar la variación de la función h() a lo largo de las iteraciones. (Considerar solo una ejecución en particular)

C) Indicar según su criterio, cuál de los tres algoritmos implementados resulta más adecuado para la solución del problema de las n-reinas. Justificar.

Forma de entrega:

1. Dentro del repositorio en github con el nombre de **ia-uncuyo-2023** crear una carpeta con el nombre **tp5-busquedas-locales**.
2. colocar un archivo con el nombre **tp5-reporte.md** que contenga la respuesta a la pregunta A, B y C de la *parte II*
3. Dentro de dicha carpeta (**tp5-busquedas-locales/**) crear una nueva carpeta **code/** para el proyecto desarrollado en python.