II-C

1.

III-B

1.

Codicioso: O(n²\*log(n)): Pasamos todas las ciudades: O(n) y en cada ciudad medimos las distancias hacía las otras ciudades. Como las opciones diminuyen con cada ciudad pasada el coste es O(n\*log(n)). Anadidas las dos costes: O(n)+O(n\*log(n)) = O(n²\*log(n))

Exhaustivo: O(n!): Crea todas las permutaciones (o circuitos) de las ciudades que son n! en total. Después compara cada circuito que también es O(n!). Dadas las reglas de la notación Big-O nos quedamos con O(n!) + O(n!) = O(n!)

Repeated greedy: Esta función aplica el algoritmo codicioso con coste O(n²\*log(n)) a todos los n nodos que sería O(n²\*log(n)) + O(n). Luego compara todos los mejores circuitos con empieze en los n nodos: O(n). En total es, teniendo en cuenta las reglas de Big-O, O(n²\*log(n)) + O(n) + O(n) = O(n²\*log(n)).

2.

La solución greedy no es óptima para un grafo:

[[0 1 3 2]

[1 0 5 4]

[3 5 0 6]

[2 4 6 0]]

Solución greedy desde el nodo 0:

Nodos: 0-–-1---3---2

Distancias: 0 + 1 + 4 + 6 = 11

Una solución óptima es:

Nodos: 3---1---0---2

Distancias: 0 + 4 + 1 + 3 = 8