0-) a-) Vicente Antonio Espinosa Gonzalez

b-) Me comprometo a no preguntar ni responder dudas de la prueba a nadie que no sea parte del cuerpo docente del curso, ya sea de manera directa o indirecta

Vicente

1-) a-) La complejidad de inserción sort es $O(n+K)$.

Con n el largo de arrays, y K el numero de permutaciones que hay. En este caso hay K elementos por arrays. lo que significa que, si el array estuviera ordenado de mayor a menor (peor caso), habrian $\binom{K}{2}$ permutaciones, o sea, $\frac{K^2-K}{2}$, entonces la complejidad seria $O\left(K + \frac{K^2-K}{2}\right)$

$\approx O(K^2)$, pero como tenemos n/K listos, se debe multiplicar $= O(K^2 \cdot n/K) = O(Kn)$ //

b-) Sabemos que debemos recorrer todos los elementos, $\frac{n}{K} \cdot K$, lo que da complejidad $O(n)$, y luego el agregar cada elemento al heap tiene una complejidad de $O(\log n/K)$, por lo tanto, se tiene una

Complejidad final de $O(n \log n/K)$

c-) merge insert sort → $O(nK + n \log(n/K))$

merge sort → $O(n \log n)$

El máximo K con el que merge insert y merge sort corren en el mismo tiempo es $K = 1$

$$O(n \cdot 1 + n \log \tfrac{n}{1}) = O(n + n \log n) \simeq O(n \log(n))$$

d-) Lo único que exigís podría ser relevante para este algoritmo podría ser el hecho de si K divide de manera exacta en n, ya que si no es así, no habrían $\frac{n}{K}$ listas con K elementos y por lo tanto los calculos no serían correctos.