1. Las hash tables son arreglos donde cada posición puede almacenar una cierta cantidad de elementos según el factor de carga de la tabla. El factor de carga lo definimos como la proporción entre el número de elementos en toda la estructura y el número de posiciones disponibles de la misma (también llamadas cubetas o buckets). La inserción del elemento x calcula la posición o cubeta donde irá este elemento en función del mismo x. Supongamos una hash table que duplica su tamaño justo antes de que su factor de carga exceda 1. Cada operación que no duplica el tamaño toma un segundo y duplicar el tamaño toma $2n$ segundos, donde n es el número actual de elementos en la tabla. Si N (mayúscula) es el número actual de cubetas en la tabla, suponiendo que inicia vacía (i.e., con una única cubeta):

a. ¿Cuántos segundos en total tomaron todos los incrementos de tamaño hasta llegar al tamaño actual? Hint: si tenemos N cubetas ahora, antes del último incremento tuvimos N /2 , y antes de ese incremento N /4 , etc.

log2(N) nos dice cuantas veces se ha duplicado para alcanzar el numero de buckets actual.
Si 2n es el costo de duplicar la cantidad de buckets, entonces.

$$\text{Tiempo solo en duplicaciones } = \sum_{n=1}^{log2(N)} 2n$$

b. Sea i el número de inserciones que se han hecho para llegar a este punto. Provea (con su respectivo procedimiento y mediante notación asintótica) una cota superior para el tiempo de ejecución de las inserciones. Hint: use la respuesta al inciso anterior en su procedimiento. Tome en cuenta que están involucrados los costos de cada inserción y cada incremento de tamaño. Nota: tome en cuenta que se busca notación asintótica en función de i, pues buscamos acotar el tiempo de ejecución de una secuencia de inserciones.

Nota de aclaración : i puede ser menor o igual que n. porque i representa la cantidad de inserciones necesarias para expandir la tabla hasta N. en un ejemplo concreto: si N = 8 y n= 7; i seria 4, porque se requirieron 4 inserciones para que se expandiera la tabla hasta 8; en las inserciones n =5, n =6 y n=7 no se realizó ninguna expansión entonces 4 es el número de inserciones que nos llevó a N = 8 ;

Costo de inserciones + costo de duplicaciones.
Costo de las inserciones = $\sum_{x=1}^{i} 1 \;=\; i$

Costo de las duplicaciones hasta llegar a N => $\sum_{x=1}^{techo(log2(i))} 2x$

c. ¿Cuál es el costo amortizado por cada inserción, usando aggregate analysis?

Costo amortizado = Costo de las operaciones / número de operaciones

(n + $\sum_{x=1}^{techo(log2(n))} 2x$)/ n

1+ $(\sum_{x=1}^{techo(log2(n))} 2x \ ) /n$

Agregamos una operación de eliminación que también toma un segundo, pero que reduce a la mitad el tamaño de la tabla justo antes de que el factor de carga sea menor a 0.25. El costo real de una reducción de tamaño será N, i.e., la cantidad de cubetas antes de la reducción. Procedamos con el método del banquero/accounting method para análisis amortizado.

d. Indique, en función del número de cubetas (donde aplique), cuántos elementos debe haber en la tabla en cada uno de los siguientes casos: tabla nueva, tabla recién duplicada (considere N el nuevo tamaño) y tabla recién reducida (considere N el nuevo tamaño).

A) 1
B) (N/2) +1 porque al llegar a $N_{anterior}$ se llenó la estructura y al agregar uno más se duplico hasta llegar a N.
C) (N/2 -1) porque se debió llegar a un factor de carga menor del 0.25 del $N_{anterior}$ para que se dividiera.

e. Supongamos que el número de elementos en algún momento después del último cambio de tamaño es n y el número de cubetas es N.
¿Cuánto habré gastado, como mínimo, desde el cambio de tamaño si el último cambio de tamaño fue un incremento?
(n inserciones* 1 segundos) + (costo de las duplicaciones de espacio)

costo de las duplicaciones= $\sum_{x=1}^{(log2(N))} 2x$

costo mínimo = n + $\sum_{x=1}^{techo(log2(n))} 2x$ segundos

¿Cuánto, como mínimo, si fue una reducción? Hint: note que la cantidad de elementos n puede ser resultado de una combinación de inserciones y remociones después del último cambio de tamaño.
$N_{anterior}$ = N*2

Se debieron haber hecho al menos N+1 inserciones para que el N se haya duplicado antes de eliminar elementos y dividirse.

Se debieron haber hecho al menos [log2(N) +1] duplicaciones .

Se debieron haber hecho [(N+1) – techo(N/2)] sustracciones para llevar al factor de carga a 0.25 y que la división de tamaño fuera necesaria.

Se debió haber hecho al menos una división.

Entonces el costo mínimo es :
[N+1] + [$\sum_{x=1}^{(log2(N))+1} 2x$] + [(N+1)- techo(N/2)] + (N*2)   segundos

f. Si N es el número de cubetas actual, el próximo incremento ocurrirá cuando n = N + 1. ¿Cuánto me costará este nuevo incremento?

solo este incremento: 2N
y acumulado , incluyendo inserciones será : ($\sum_{x=1}^{(log2(N)+1)} 2x$) + (N+1)   segundos

¿Cuánto habré gastado, como mínimo, desde el incremento anterior para llegar a este punto?

en inserciones : 2^[Log2(N)] segundos - 2 ^ [Log2(N)- 1] segundos

Proponga costos amortizados para las operaciones de inserción y eliminación, tales que, a lo largo de una secuencia arbitraria de operaciones, ahorren lo necesario para pagar por el próximo incremento.
Costo amortizado de inserción de 3s para que con cada paso sobren 2s y se acumulen suficientes para cada inserción

Repita el procedimiento para pagar con ahorros por la próxima reducción. Finalmente, determine los costos amortizados para los cambios de tamaño.
¿Cuál será (en notación asintótica) la tasa de crecimiento más alta para una secuencia arbitraria de n operaciones?
Costo amortizado de 4/3s en cada sustracción para que se acumulen suficientes segundos al llegar al 0.25 de factor de carga para poder costear la división .

O(n)