

Función de hash

Definimos una **función de hash** h como una función

$$h : \mathbb{N}_0 \rightarrow [0, m - 1]$$

Con m el largo de una tabla de hash

Pero una función de hash puede no estar ligada a una tabla

Función de hash

El dominio de las claves puede no ser \mathbb{N}_0 . Llamémoslo D

Una función de hash H se define entonces como sigue

$$H : D \rightarrow R, \quad R \subseteq \mathbb{N}_0$$

Y a la definición anterior la llamaremos **método de ajuste**

$$h : \mathbb{N}_0 \rightarrow [0, m - 1]$$

Función cualquiera

Cualquier función $f : D \rightarrow R$ cumple con que

$$x = y \rightarrow f(x) = f(y)$$

Esto significa que

$$f(x) \neq f(y) \rightarrow x \neq y$$

Colisiones

Decimos que el hash H tiene una **colisión** cuando

$$x \neq y \quad \wedge \quad H(x) = H(y)$$

El ajuste h puede producir una **colisión en la tabla** si

$$x \neq y \quad \wedge \quad h(H(x)) = h(H(b))$$

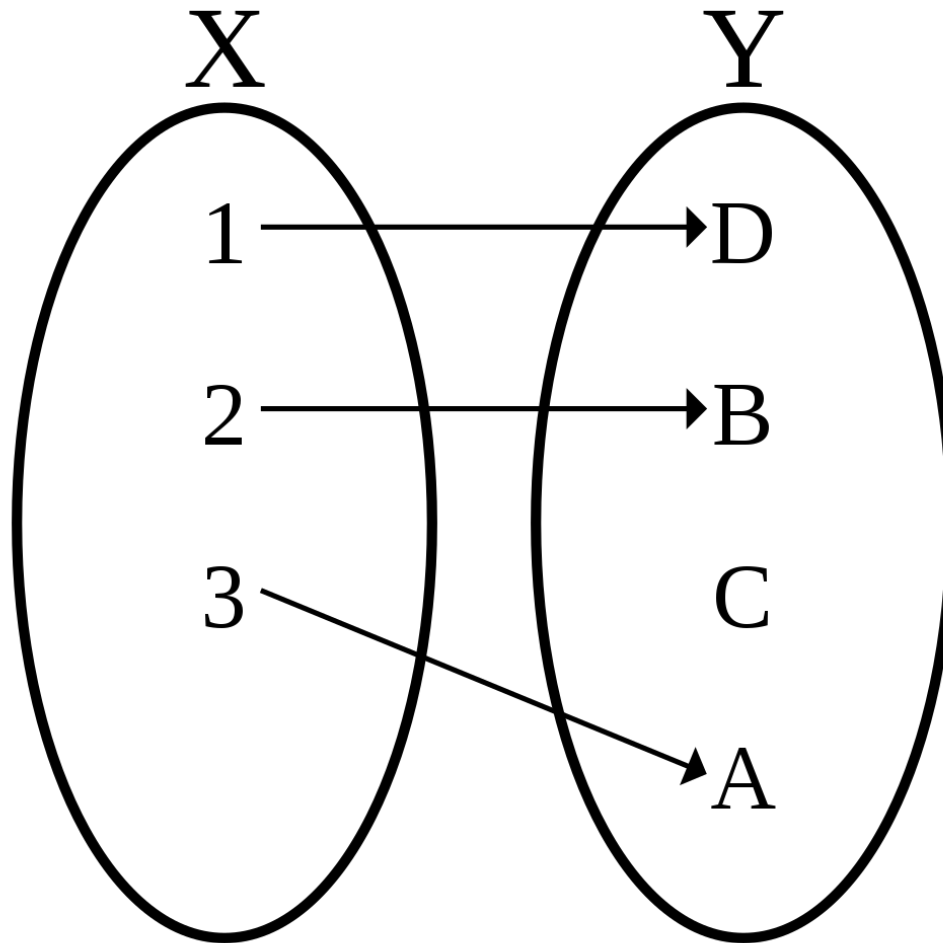
Propiedades de una función

Una función cualquiera $f : D \rightarrow R$ puede ser:

- Inyectiva
- Sobreyectiva
- Biyectiva

¿Qué significa esto para una función de hash H ?

Función Inyectiva



Función Inyectiva

Si la función de hash H es inyectiva, entonces

$$x \neq y \rightarrow H(x) \neq H(y)$$

¿Qué ventajas tiene esto? ¿Tiene alguna desventaja?

¿Es posible que la función de ajuste h sea inyectiva?

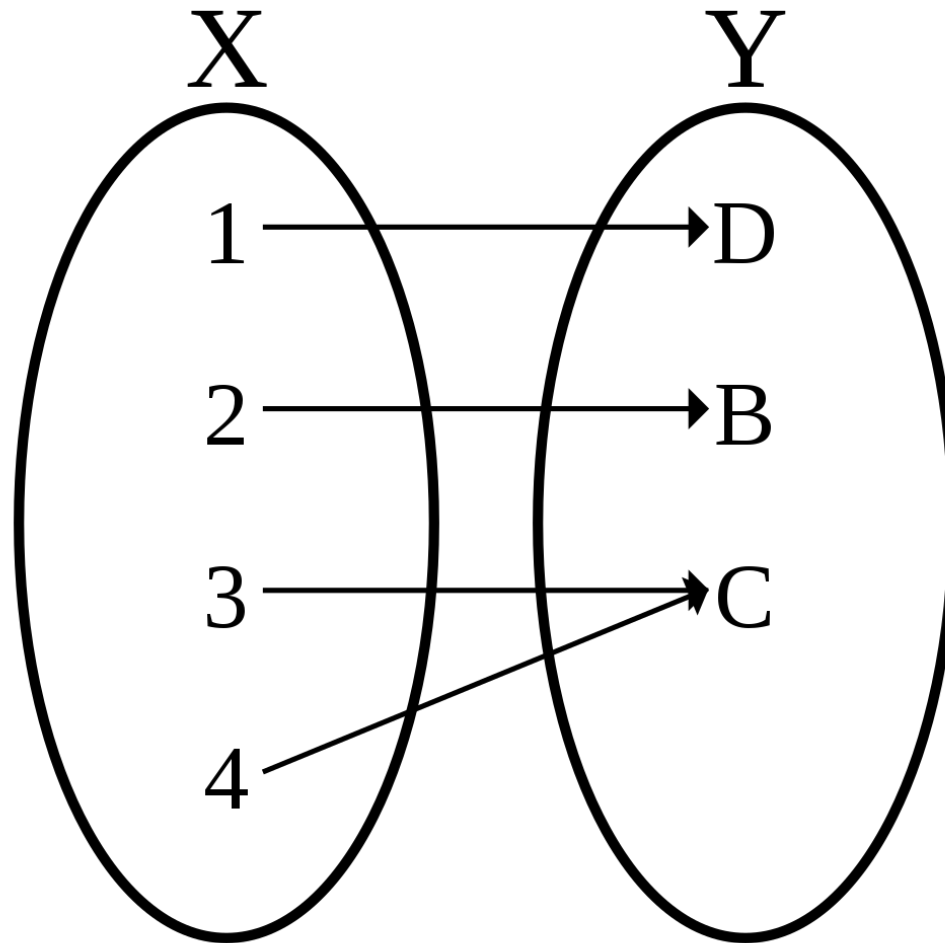
Función Perfecta

Una función de hash inyectiva se dice **perfecta**

La única posibilidad de colisiones es por el ajuste

Podemos comparar por hashes y olvidarnos de la clave

Función Sobreyectiva



Función Sobreyectiva

Si la función de hash H es sobreyectiva, entonces

$$\forall r \in R, \exists d \in D$$

$$H(d) = r$$

¿Qué ventajas tiene esto? ¿Tiene alguna desventaja?

¿Es posible que la función de ajuste h sea sobreyectiva?

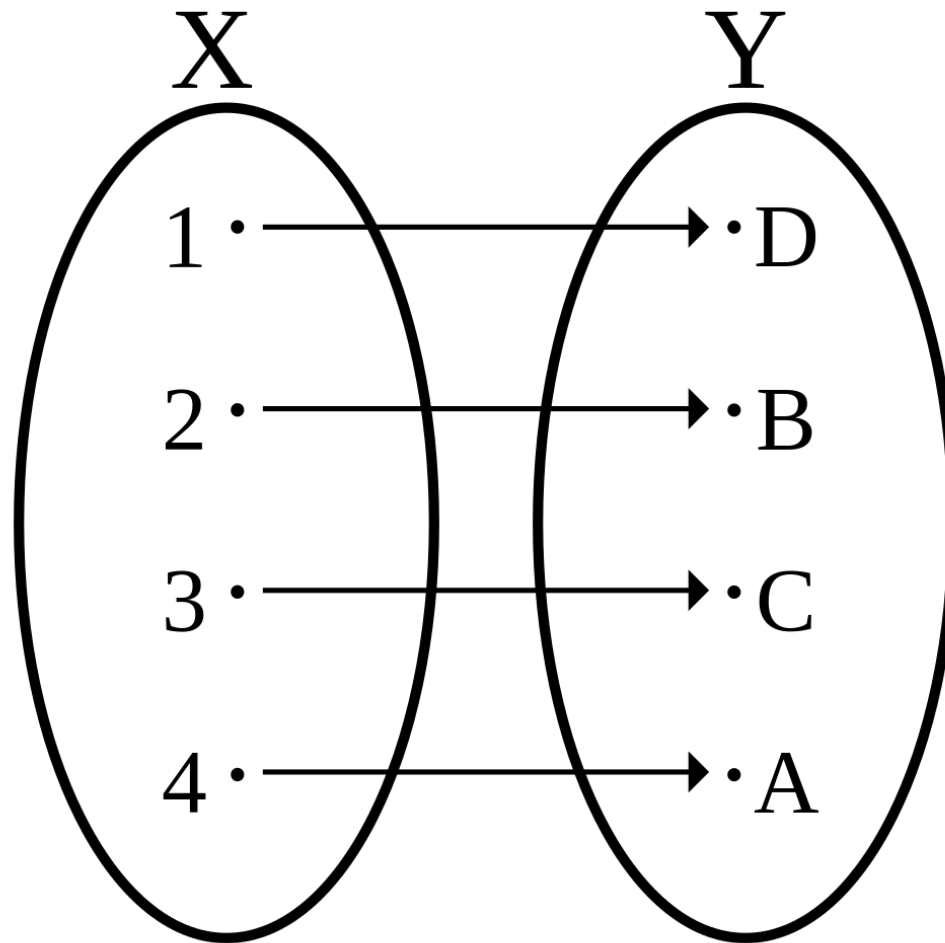
Función Compacta

Una función de hash sobreyectiva se dice **compacta**

Una función puede ser más o menos compacta según cuantos elementos de R quedan sin preimagen

El ajuste h debe ser compacto sí o sí

Función Biyectiva



Función Biyectiva

Si la función de hash H es sobreyectiva, entonces

- Es inyectiva
- Es sobreyectiva

¿Qué significa esto?

Función Invertible

Para funciones continuas, una función biyectiva es **invertible**

Pero una función de hash es discreta

Basta con que sea inyectiva para poder invertirla

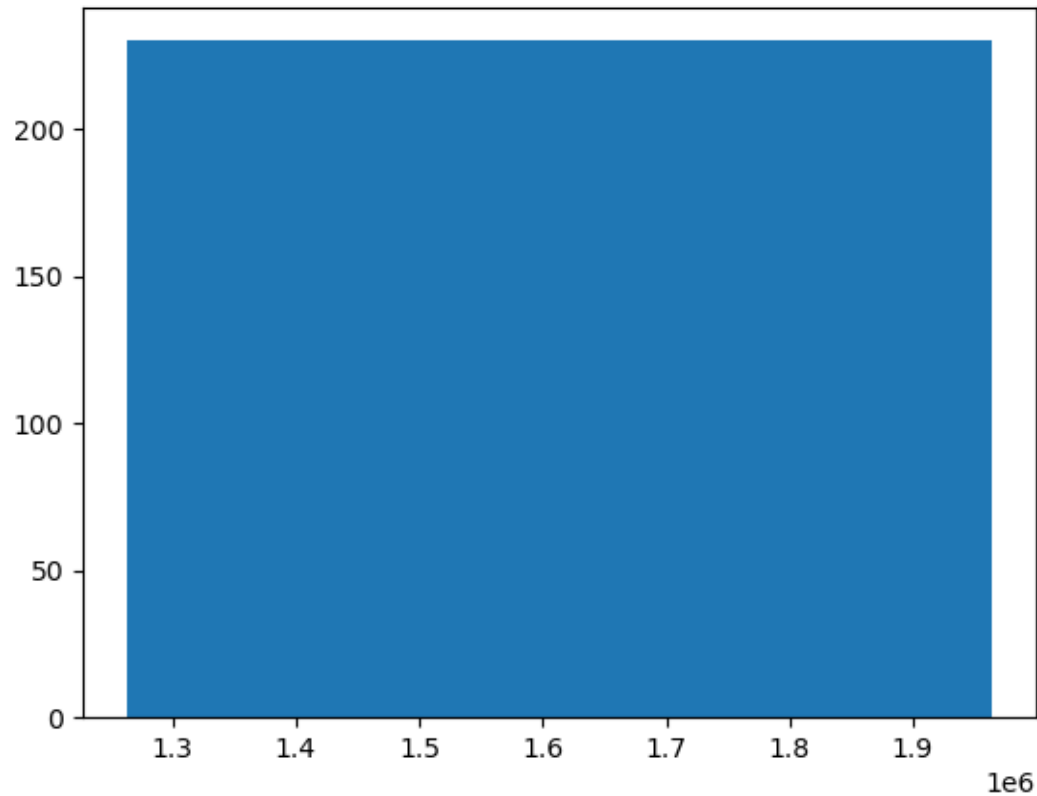
Propiedades de un hash

Una función de hash puede tener otras propiedades:

- Distribución uniforme
- Eficiente
- Incremental
- Efecto avalancha

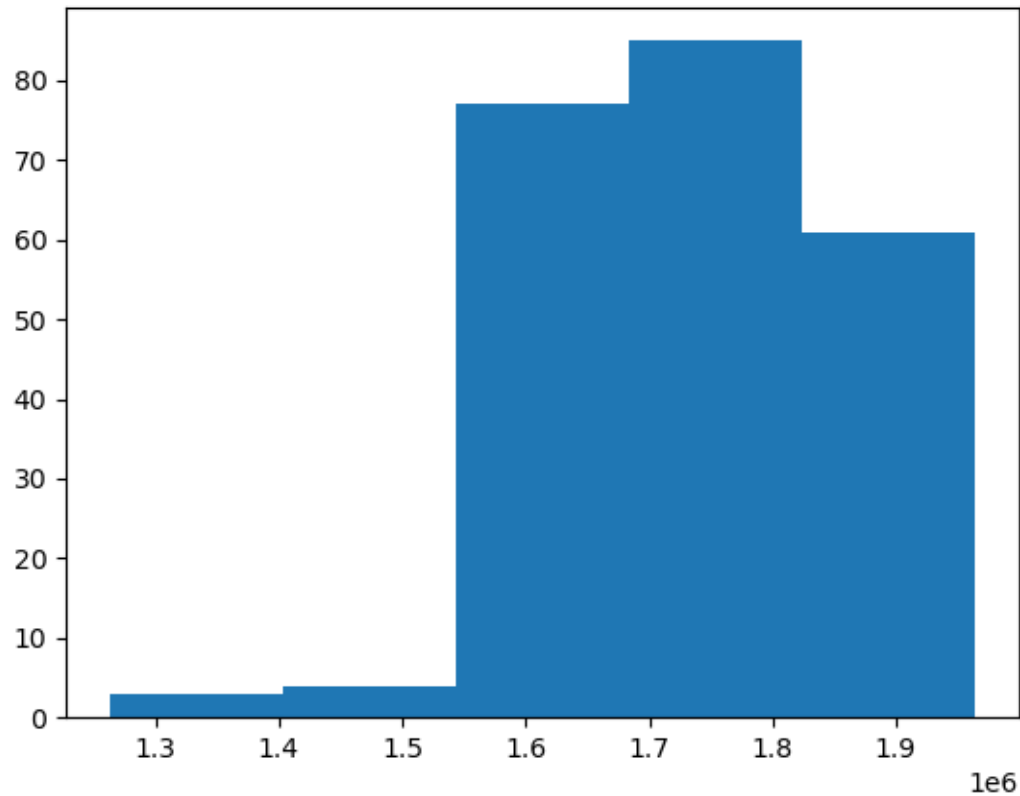
También afectan al comportamiento de una tabla de hash

Histograma: N° Alumno



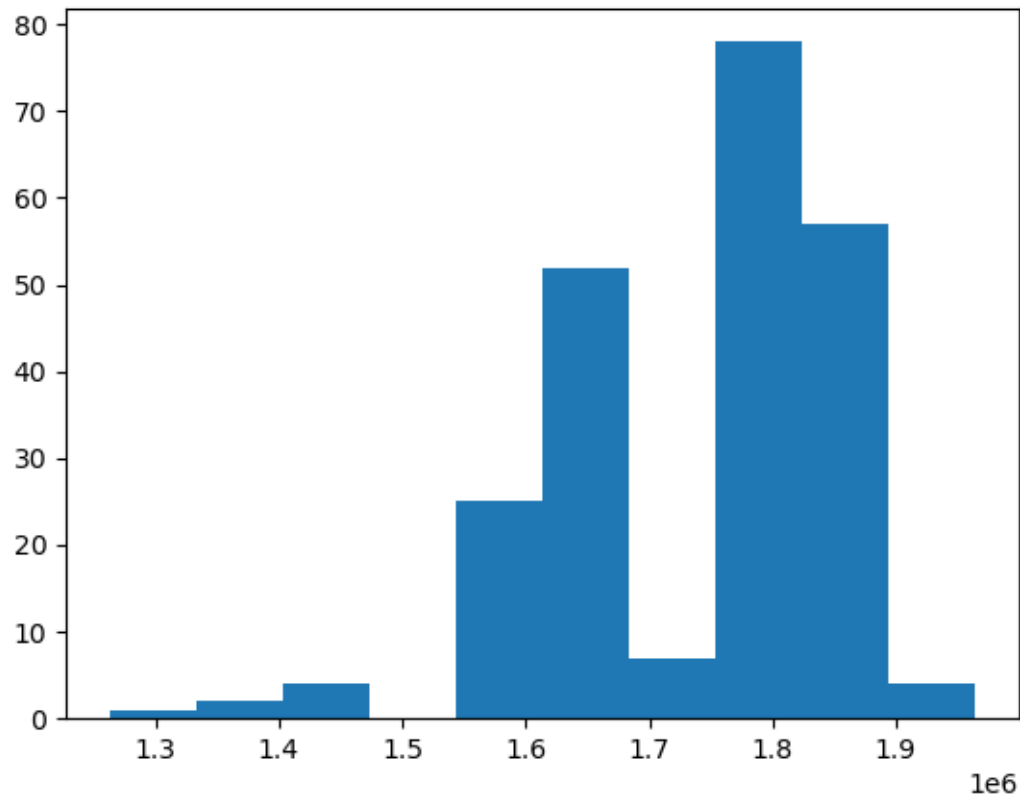
1 bin

Histograma: N° Alumno



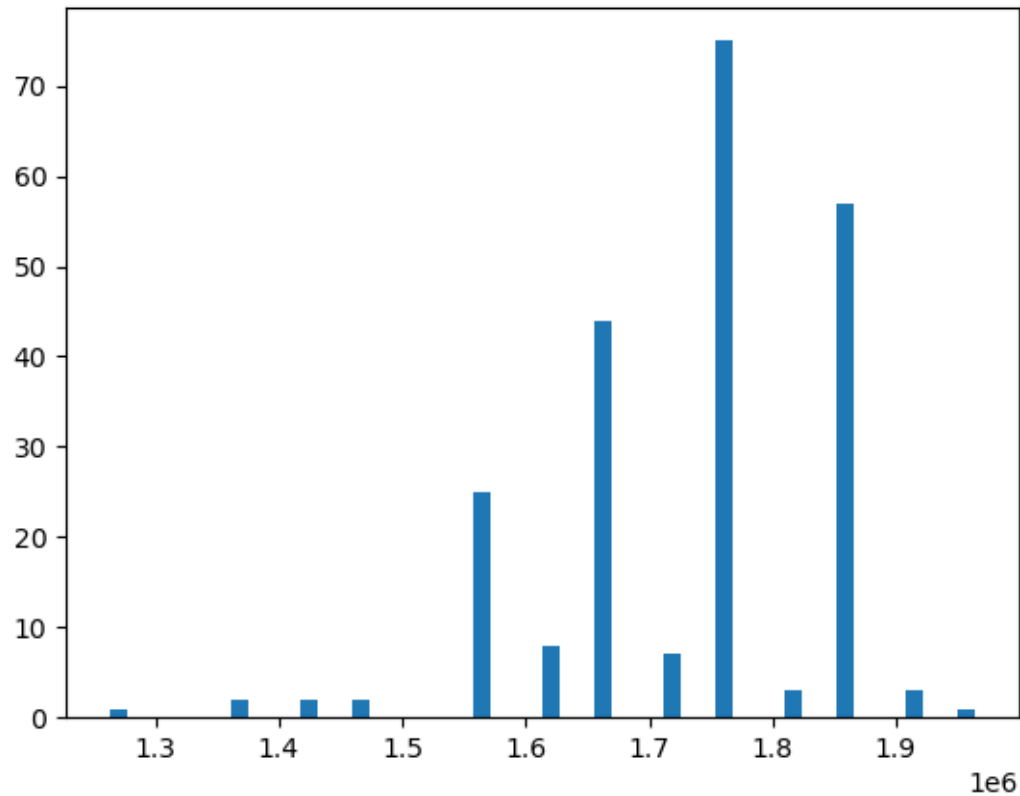
5 bins

Histograma: N° Alumno



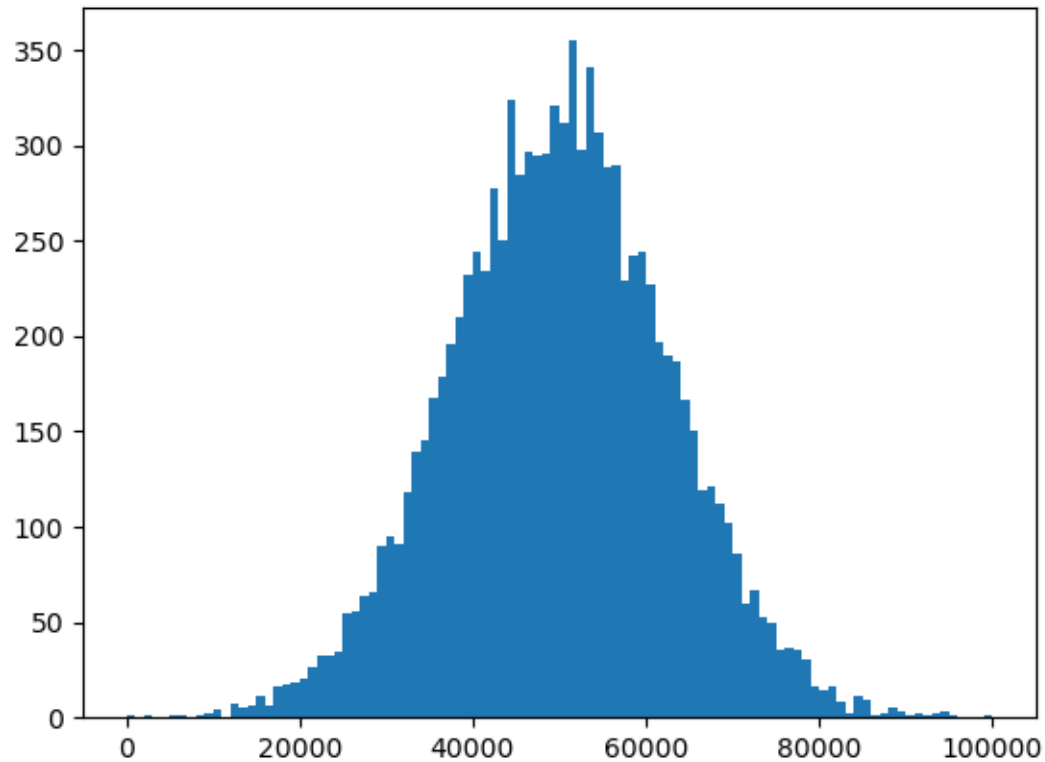
10 bins

Histograma: N° Alumno

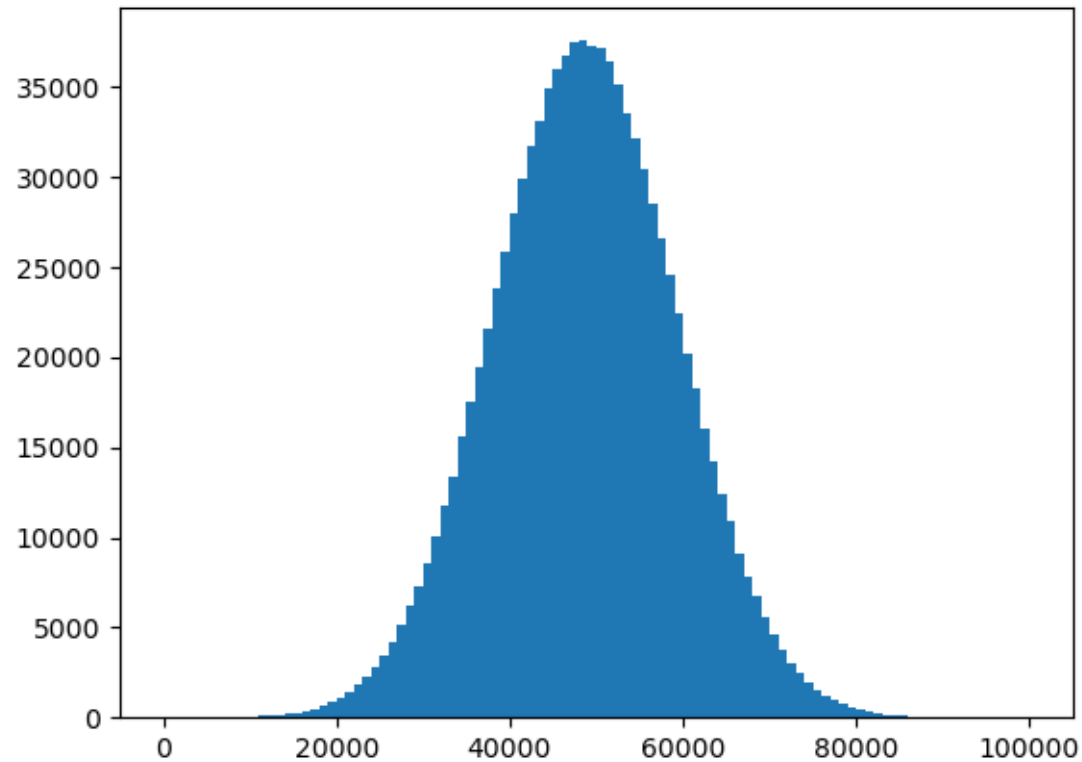


50 bins

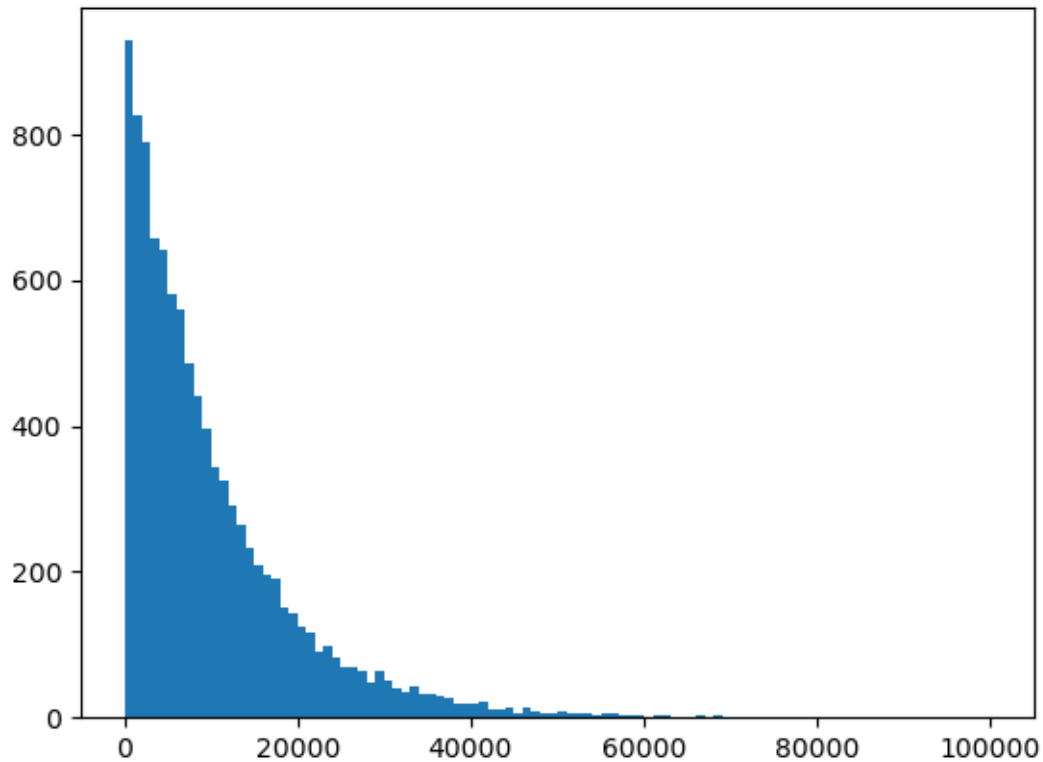
Distribución Normal



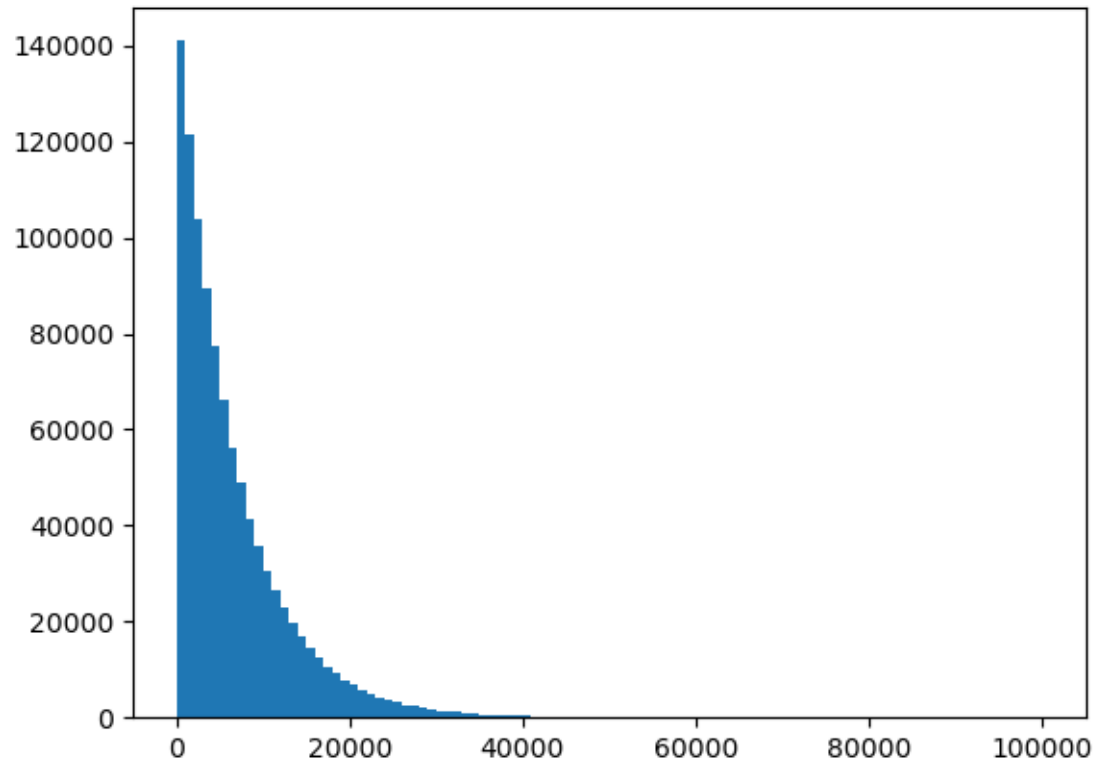
Distribución Normal



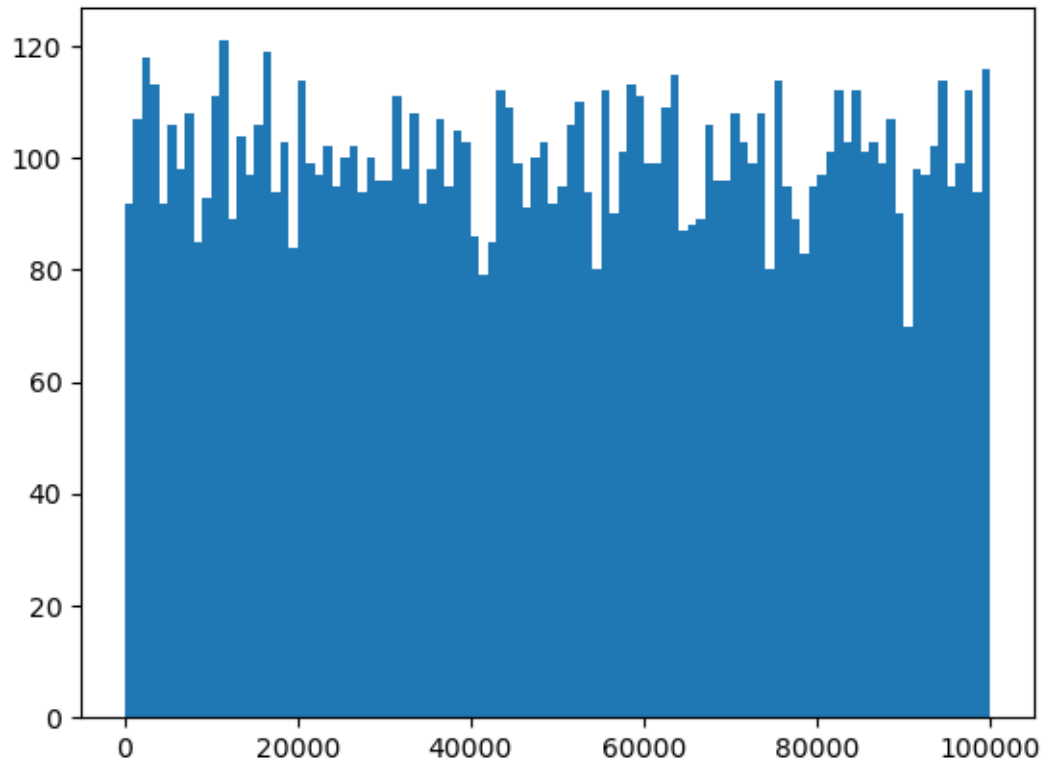
Distribución Exponencial



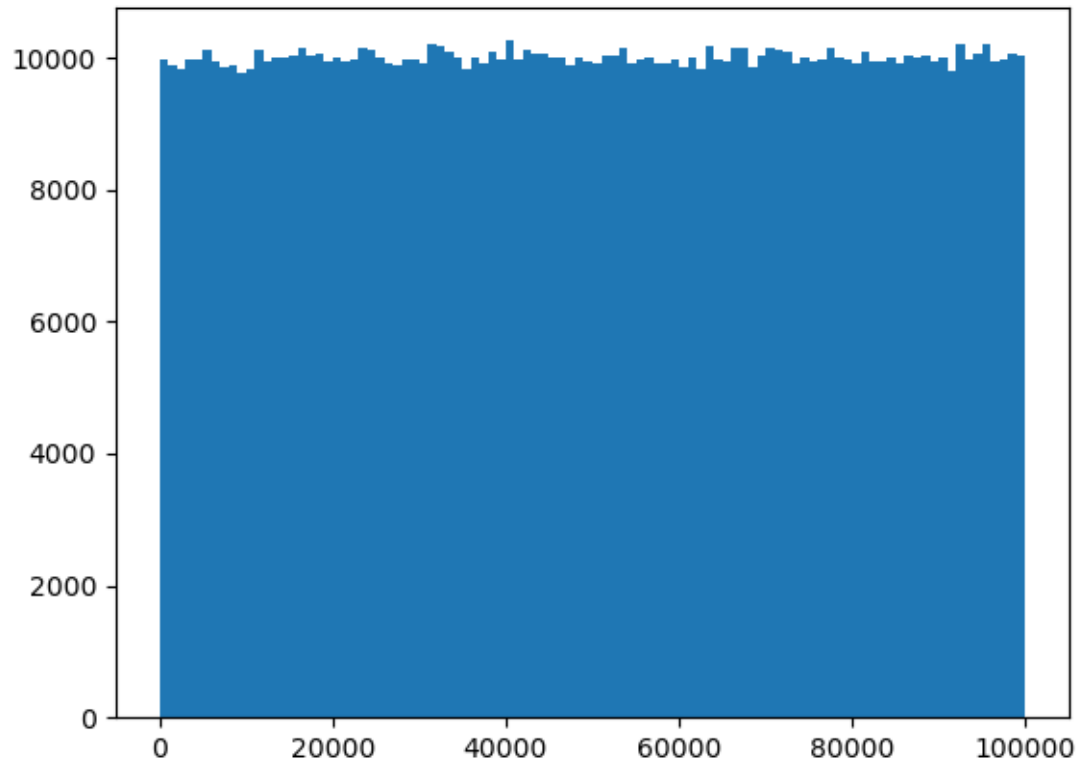
Distribución Exponencial



Distribución Uniforme



Distribución Uniforme

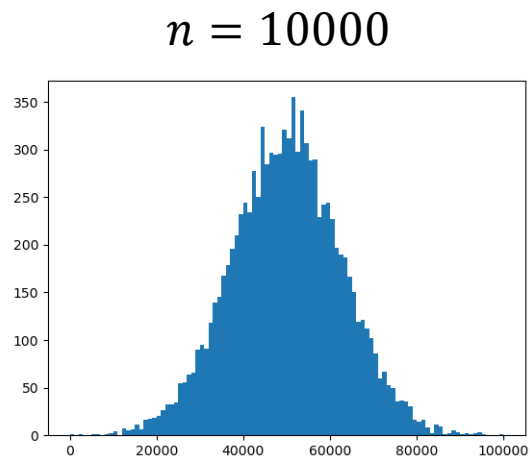


Distribución vs Tabla

¿Qué efecto tiene en la tabla la distribución del hash H ?

¿Qué efecto tiene en la tabla la distribución del ajuste h ?

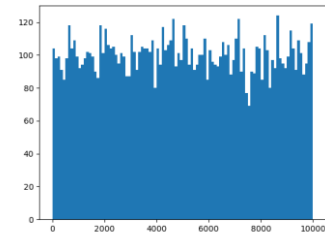
Método de la división



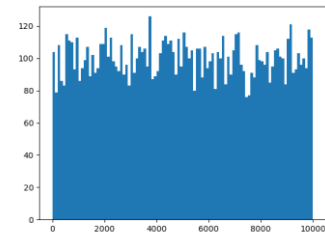
$$h(k) = k \bmod m$$

$h(k)$

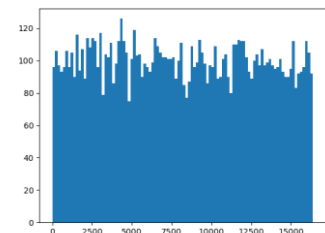
$m = 10000$



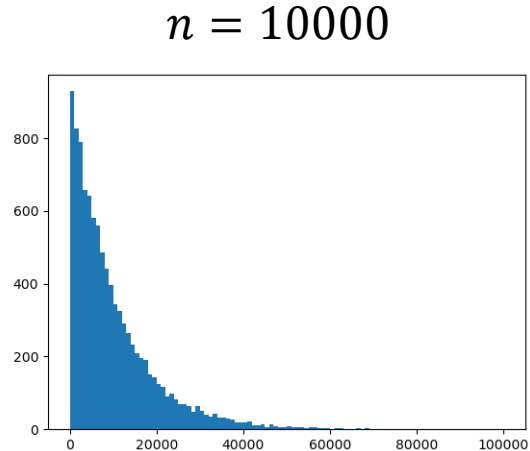
$m = 10007$



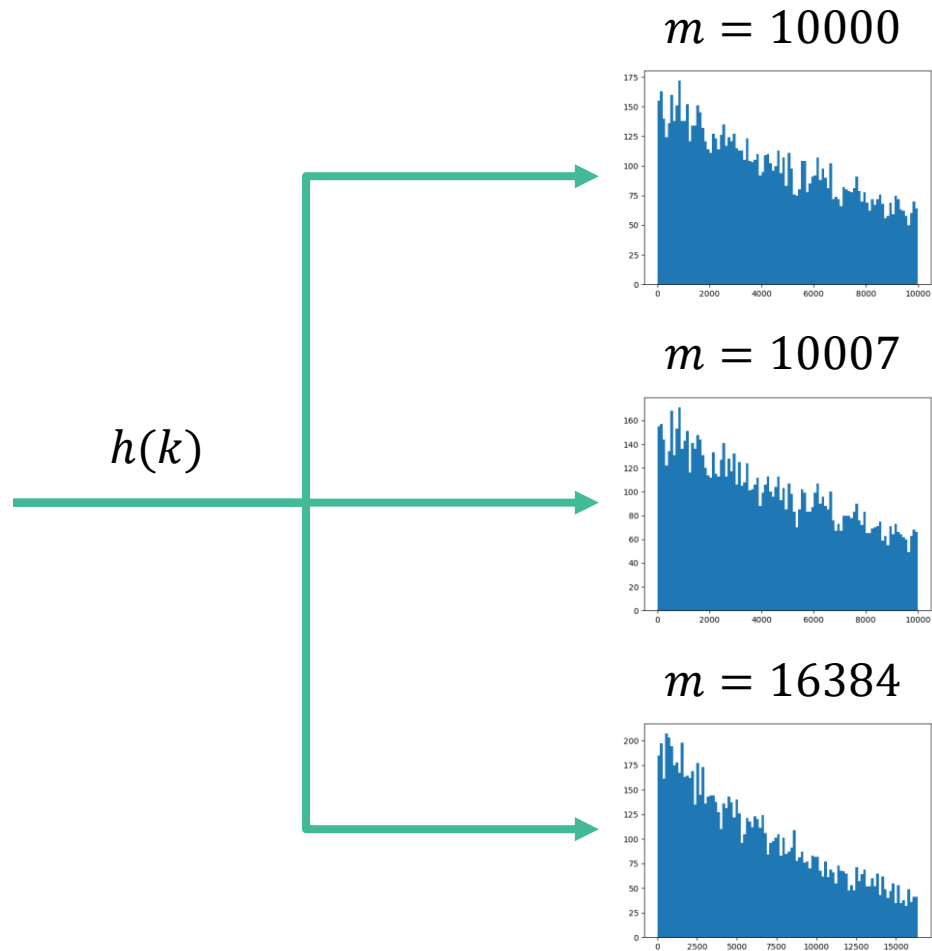
$m = 16384$



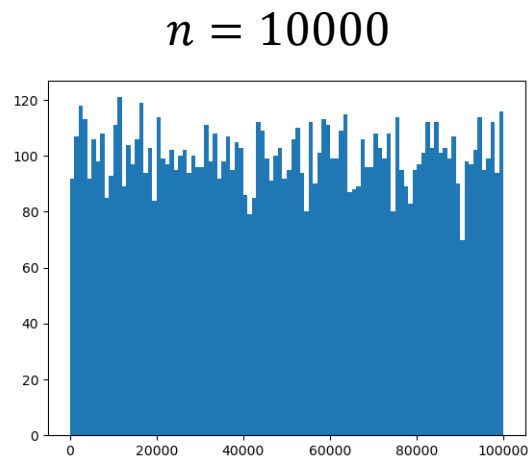
Método de la división



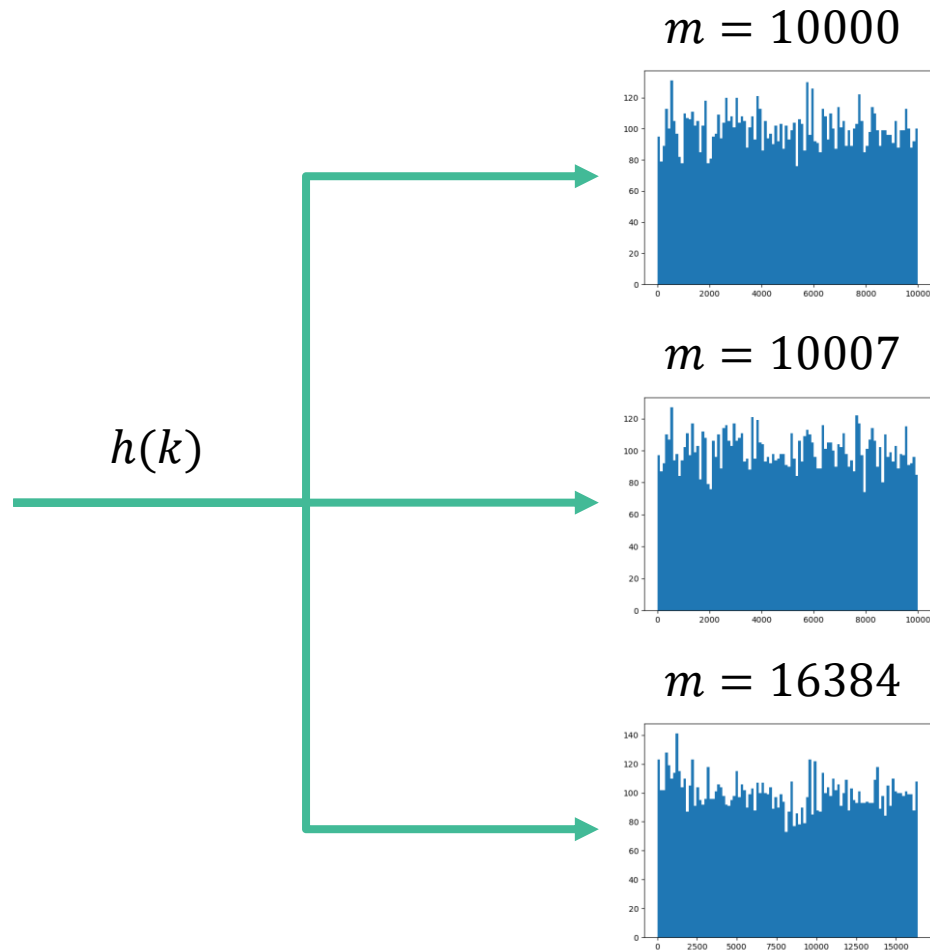
$$h(k) = k \bmod m$$



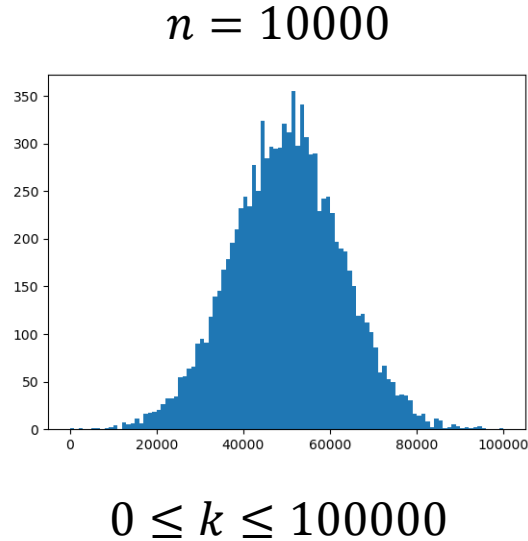
Método de la división



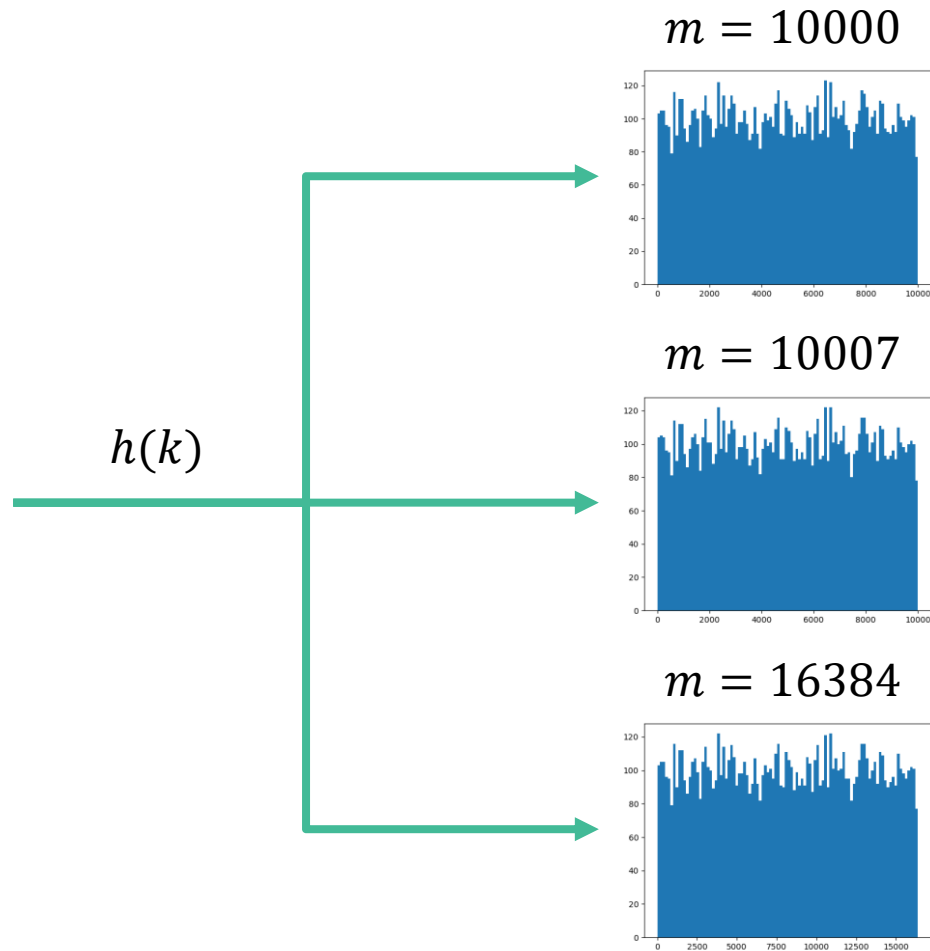
$$h(k) = k \bmod m$$



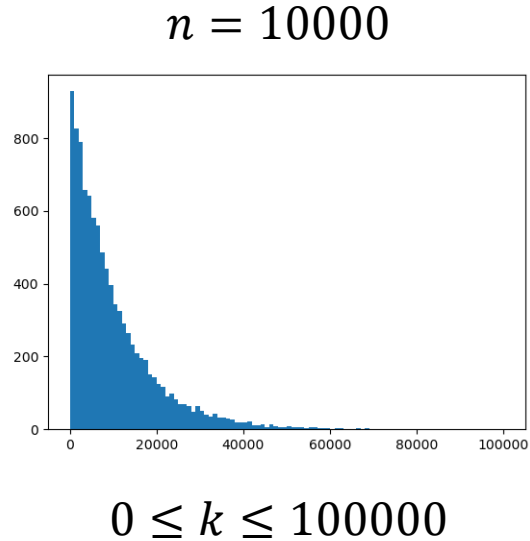
Método de la multiplicación



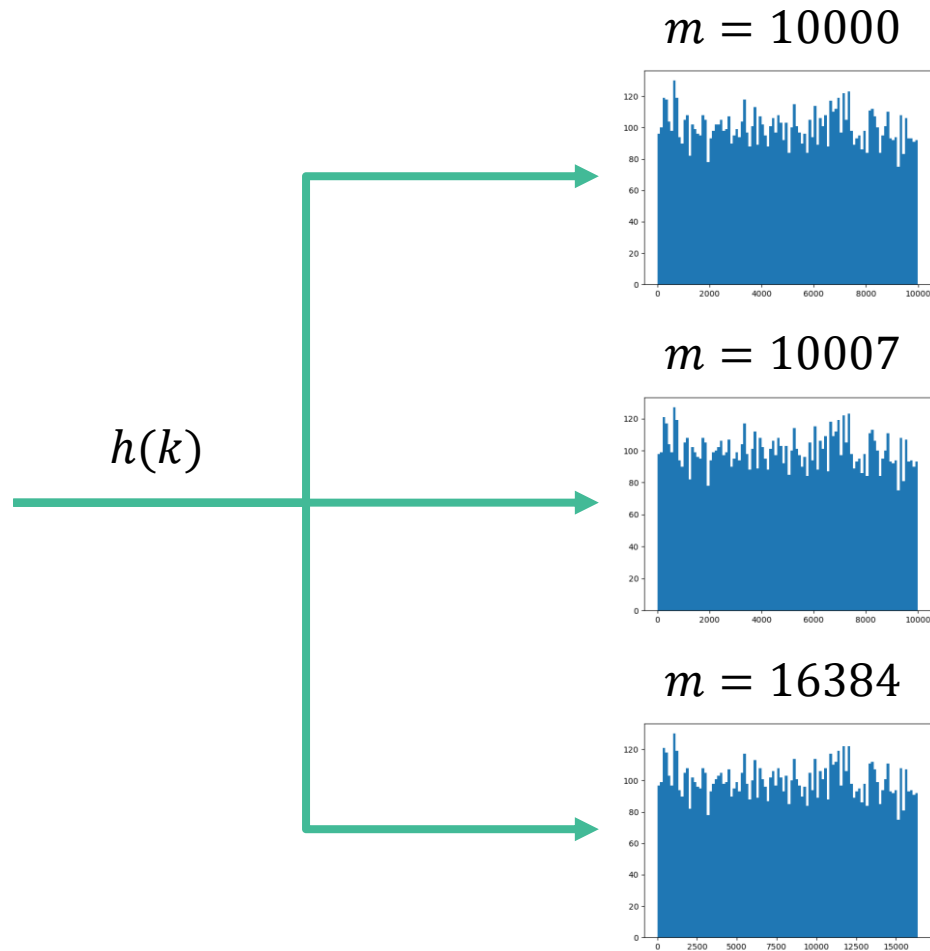
$$h(k) = \lfloor (\varphi k \bmod 1) \cdot m \rfloor$$



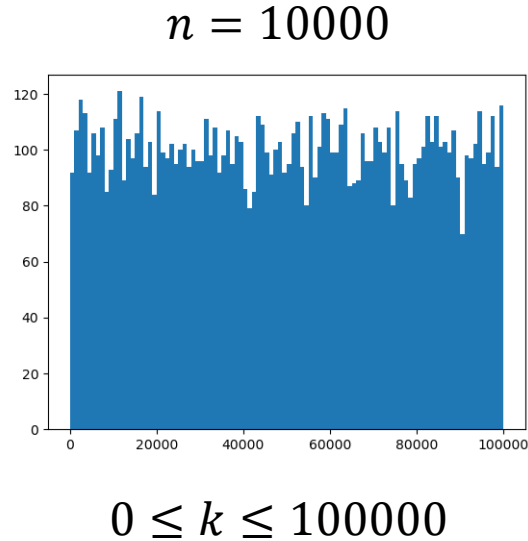
Método de la multiplicación



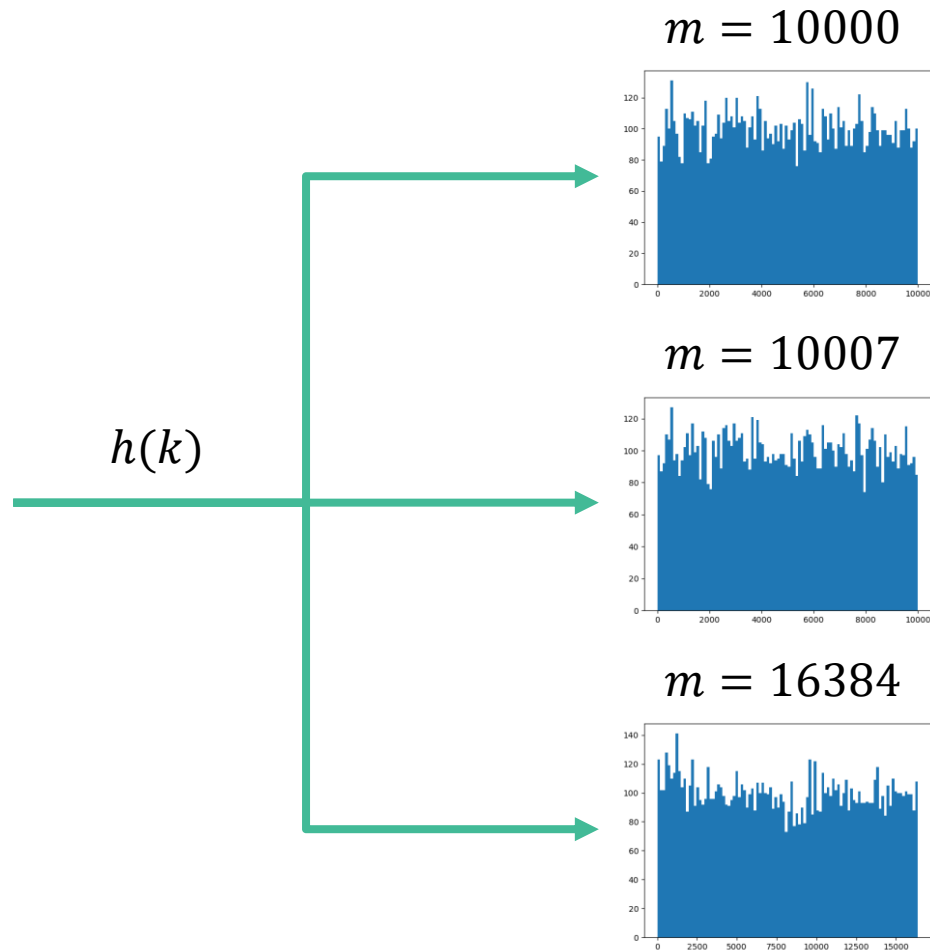
$$h(k) = \lfloor (\varphi k \bmod 1) \cdot m \rfloor$$



Método de la multiplicación



$$h(k) = \lfloor (\varphi k \bmod 1) \cdot m \rfloor$$



Uniformidad de H

Es importante que H sea uniforme

Es difícil uniformizar los datos con el ajuste h

Si H es uniforme, entonces es muy fácil que h sea uniforme

Eficiencia

En una tabla se llama la función de hash para cada operación

La complejidad de la función de hash debe ser $\mathcal{O}(d)$,
con d el tamaño del dato hasheado

Hash Incremental

Si y tiene mucho en común con x , llamemos a z su diferencia

H se dice **incremental** si $H(y)$ se puede expresar como

$$H(y) = G(z, H(x))$$

El costo de calcularlo es $\mathcal{O}(|z|)$

Hash de strings

Sea $X = [x_1, x_2, x_3, \dots, x_n]$ un string de n datos

Si a cada dato x_i le damos una interpretación numérica,

Podemos interpretar X como un número en base b :

$$H(X) = x_1 b^{n-1} + x_2 b^{n-2} + \dots + x_{n-1} b^1 + x_n b^0$$

Hash de strings

$$H(X) = x_1 b^{n-1} + x_2 b^{n-2} + \dots + x_{n-1} b^1 + x_n b^0$$

$$H(X) = \sum_{i=1}^n x_i b^{n-i}$$

Rolling hash

¿Cómo obtener $H(X[5 : 8])$ a partir de $H(X[4 : 7])$?

Con $n = 10$

$$X[4 : 7] = [x_4, x_5, x_6, x_7]$$

$$X[5 : 8] = [x_5, x_6, x_7, x_8]$$

Rolling hash

¿Cómo obtener $H(X[5 : 8])$ a partir de $H(X[4 : 7])$?

$$H(X[4 : 7]) = x_4 b^6 + x_5 b^5 + x_6 b^4 + x_7 b^3$$

$$H(X[5 : 8]) = x_5 b^5 + x_6 b^4 + x_7 b^3 + x_8 b^2$$

$$H(X[5 : 8]) = H(X[4 : 7]) - x_4 b^6 + x_8 b^2$$

Efecto avalancha

Para x e y muy similares, si $f(x)$ es muy distinto a $f(y)$

Entonces la función f tiene **efecto avalancha**

¿Cuál de los ajustes estudiados cumple con esto?

Efecto avalancha

El método de la multiplicación tiene efecto avalancha

Esto es útil cuando se insertan muchas claves muy similares:
estas se reparten a lo largo de la tabla

Para distribuciones muy concentradas, ayuda a uniformizar