# Respuestas a las preguntas de la tarea 1 – Hashing

## Decisiones y código más importante de cada ejercicio

### Ejercicio 1 – Prueba lineal

Lo más importante son las funciones H y G. Que en este caso son las únicas dependientes de esta prueba, lo demás es compartido con las demás pruebas de hashing. No fue muy difícil comparado con las otras pruebas, ya que era solo sumar los intentos.

### Ejercicio 2 – Prueba dependiente de clave

En esta prueba es lo mismo que en la anterior, pero fue más complicada. Aquí tuve que añadir el concepto de clave única como número, no como cadena de caracteres. Esto es para hacer el módulo correctamente y además, poder comparar entre números al buscar y no con cadenas.

Al final opté por construir esta clave a partir de la matrícula de cada coche, que es única. El problema estaba en convertir las letras a números sin perder el carácter identificativo (único) de la matrícula, y además, que cupiese en una variable lo menos grande posible. De esta forma, opté por convertir las letras a su valor numérico en ASCII. Como los números podían ir desde el 9999 hasta el 1111, y las letras nunca llegarían al valor 99 en ASCII, para que el número resultante fuese más pequeño, le di la vuelta. Concatené primero los números en ASCII y luego a la derecha de estos, los números de la matrícula (sin convertir a ASCII, porque ya son números). De esta forma quedaría la matrícula 9999-ZZZ como 424242-9999 en vez de 9999-424242. Luego, mediante la librería <limits.h> me aseguré de que este número no alcanzase el límite de un unsigned long.

## ¿Qué función H(x) se ha utilizado? ¿Y qué función G(x)?

### Ejercicio 1 – Prueba lineal

Para esta prueba se ha utilizado la función H como un módulo del tamaño total de la tabla con la clave numérica (que se construye a partir de la matrícula).

La función G básicamente lo que hace es ir sumando al resultado de la función H, el número de intentos y luego se le hace el módulo del tamaño total de la tabla para que sea circular y no se pase. Haciendo esto obtenemos “g”. Lo mismo que se haría en papel con la fórmula matemática.

### Ejercicio 2 – Prueba dependiente de clave

Al final se utiliza la misma función H que en la prueba lineal.

La función G ya cambia. Antes de nada, decir que incrementé el tamaño de la tabla a 256 para que fuese potencia de 2 y permitir un recorrido completo, además de lo que contaré a continuación. Primero, dividimos la clave que construimos a partir de la matrícula entre el tamaño de la tabla y tomamos la parte entera de esta división. Si el resultado es impar o cero, se le suma una unidad a este número, con esto nos aseguramos el recorrido completo. Luego sumamos al resultado de la función H el número anterior (d) multiplicado por el intento actual. A esto le hacemos el módulo del tamaño total y obtenemos “g”.

## ¿Cuántos accesos a la tabla hay que realizar para recuperar los registros con matrículas 9181XNT, 9841FRD y 6531FTQ?

### Ejercicio 1 – Prueba lineal

9181XNT: 1 intento.

9841FRD: 16 intentos.

6531FTQ: 245 intentos.

### Ejercicio 2 – Prueba dependiente de clave

9181XNT: 1 intento.

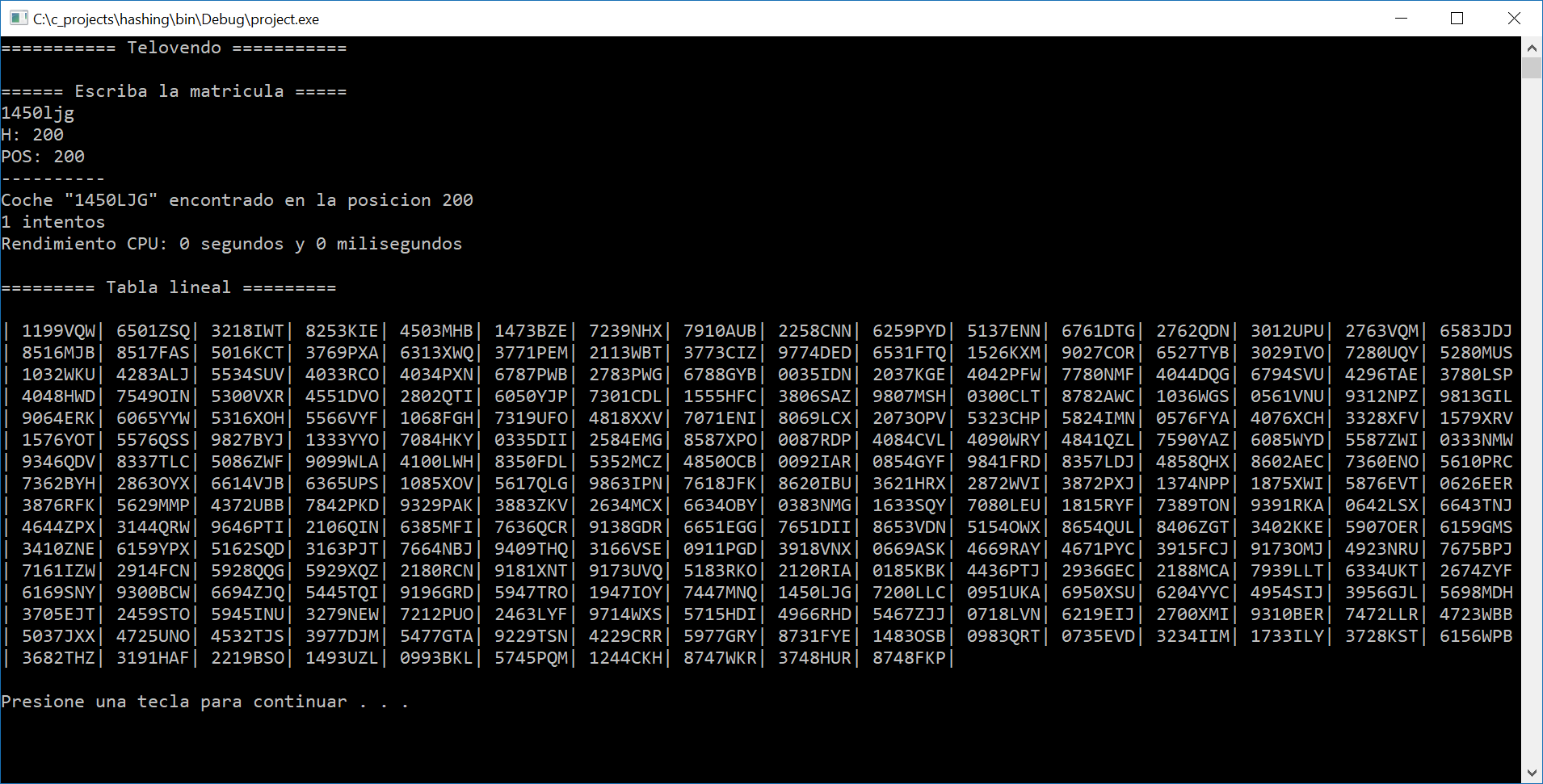
9841FRD: 7 intentos.

6531FTQ: 7 intentos.

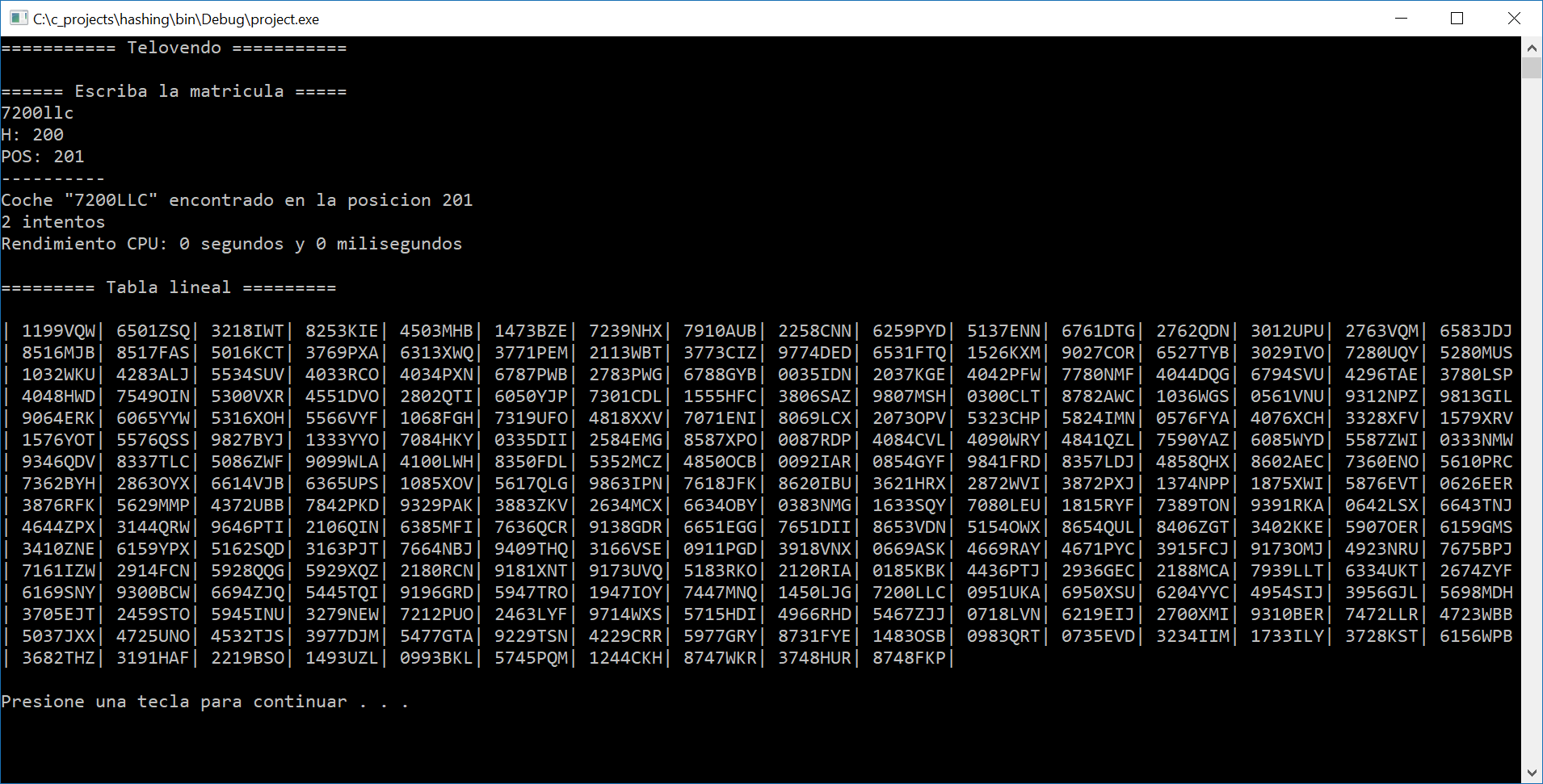
## Eliminar un registro que se haya insertado sin colisión. Probar a recuperar un registro que sí haya producido colisión con el que se acaba de eliminar. ¿Se recupera correctamente?

### Ejercicio 1 – Prueba lineal

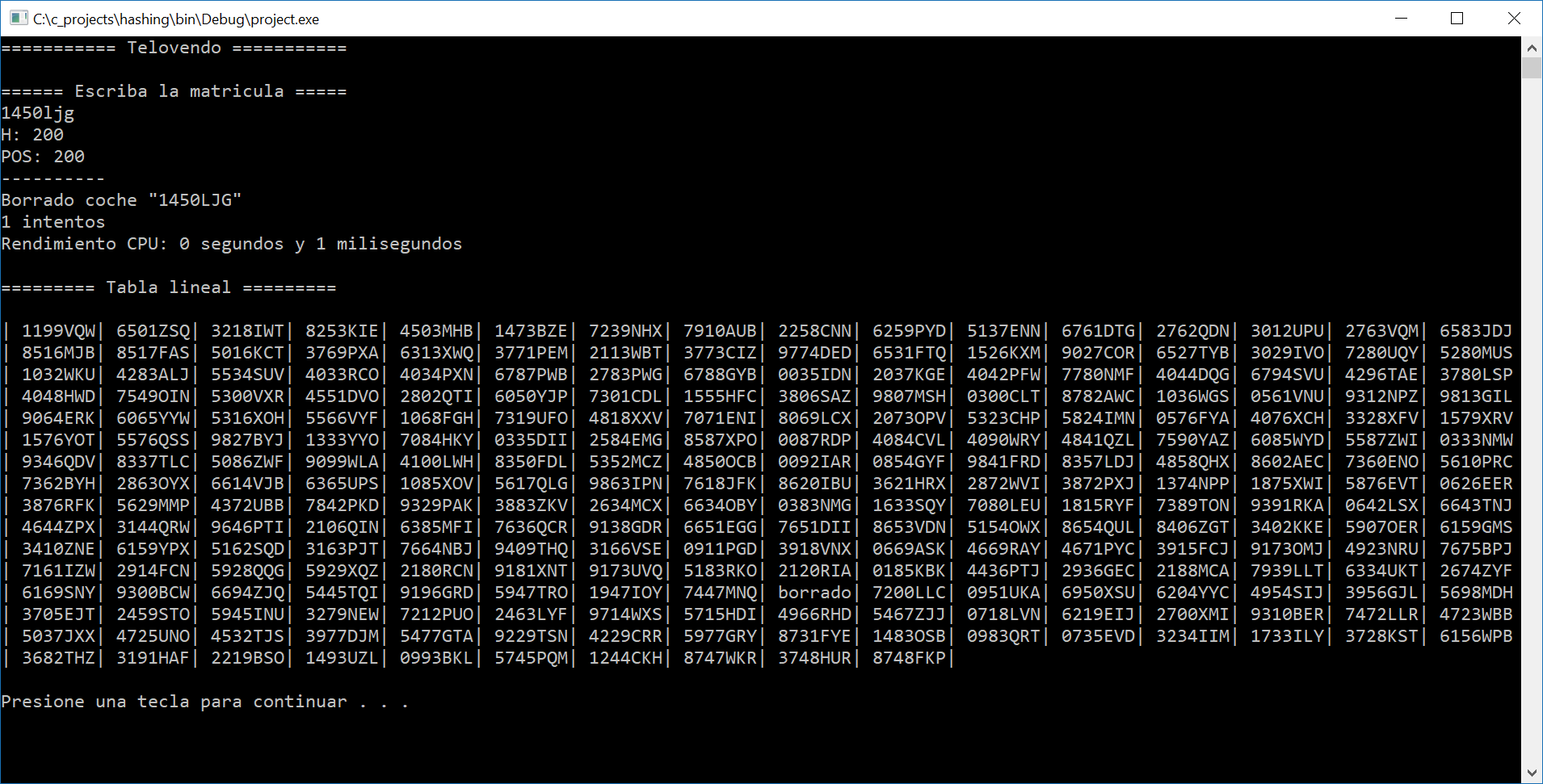
H (“1450LJG”, 1) = 200.



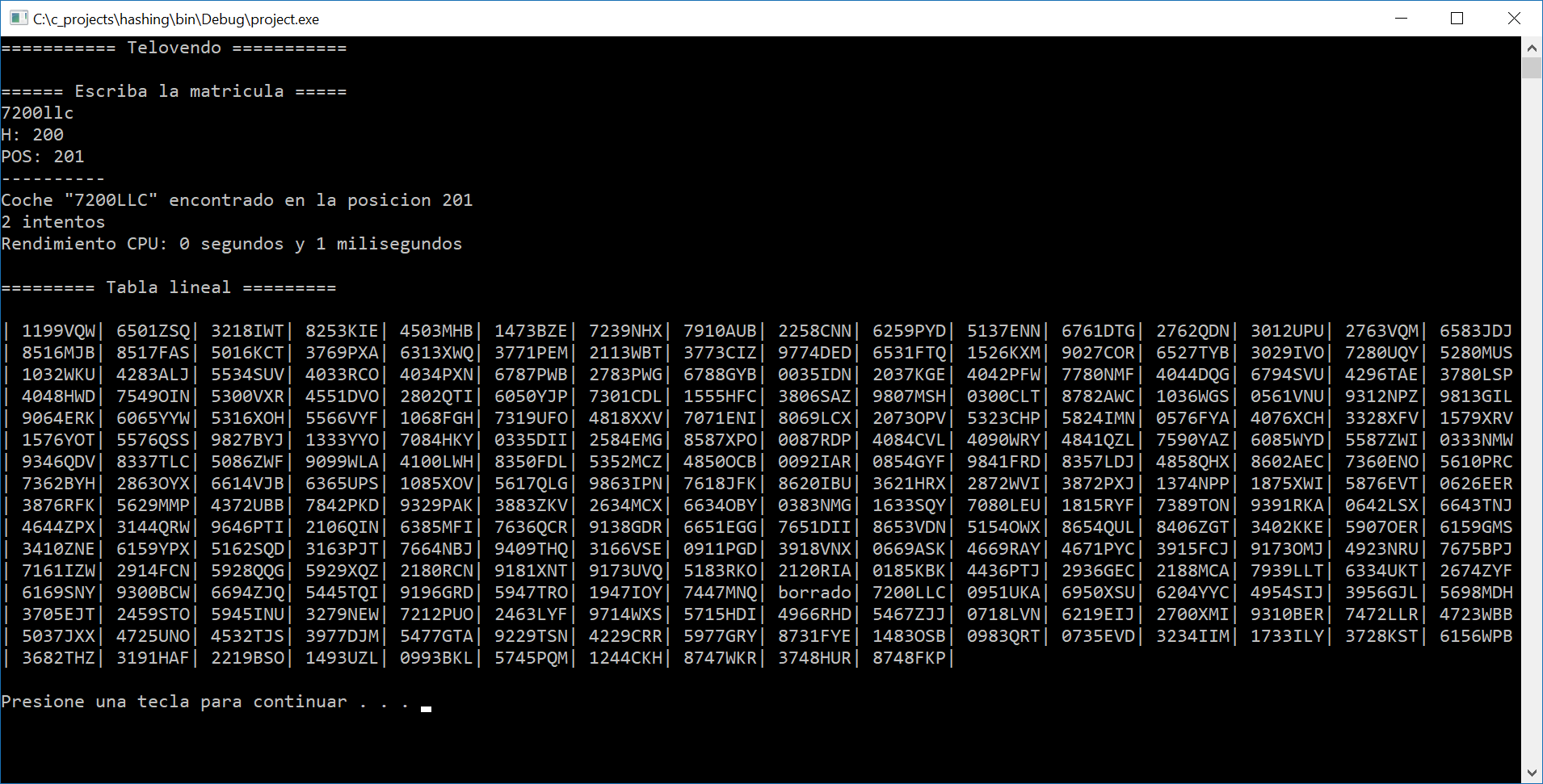
H (“7200LLC”, 1) = 200.   
H (“7200LLC”, 2) = 201.



Ahora lo borro:



Ahora recupero el de colisión:



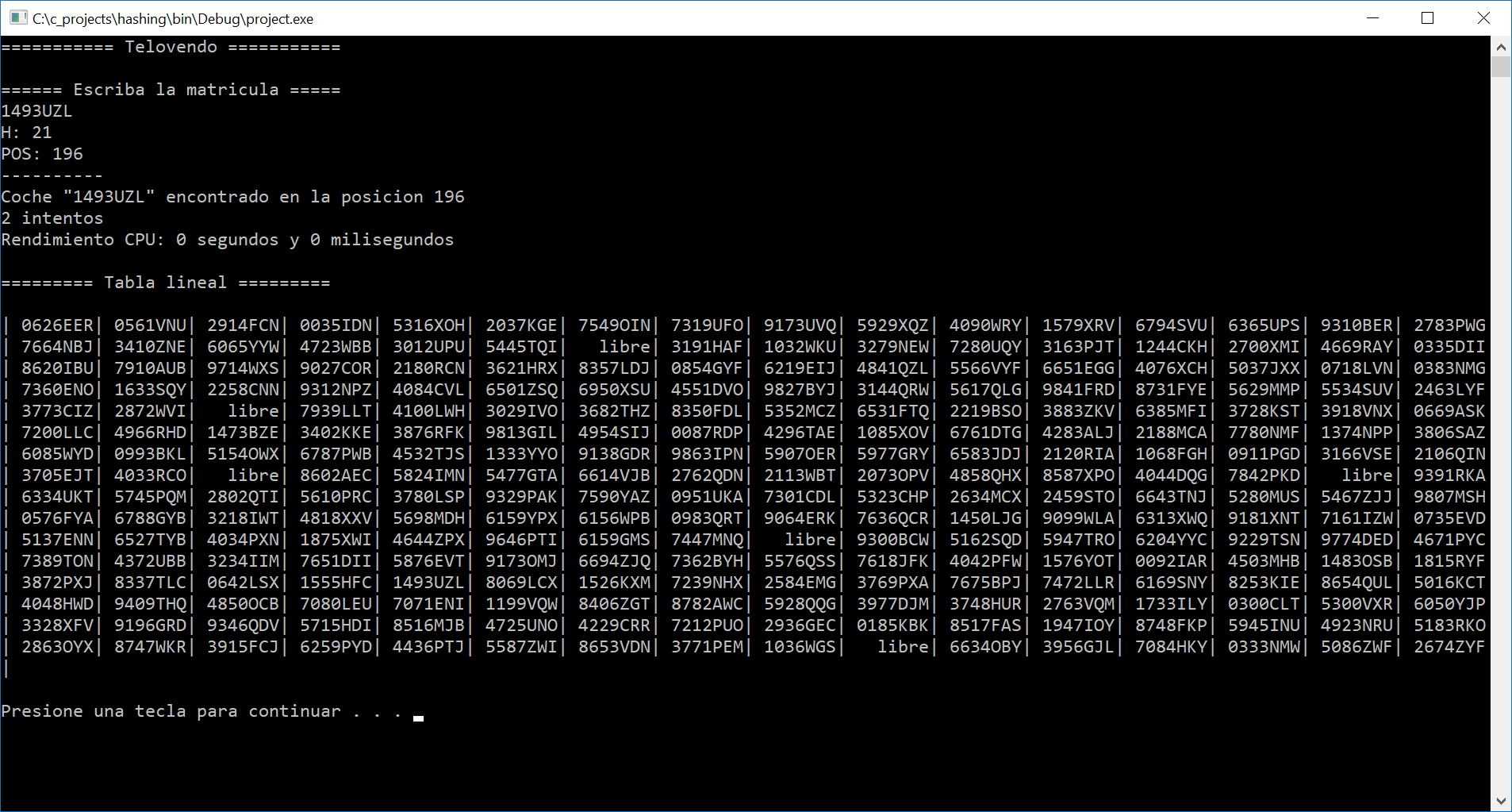
Por lo tanto, lo hace bien.

### Ejercicio 2 – Prueba dependiente de clave

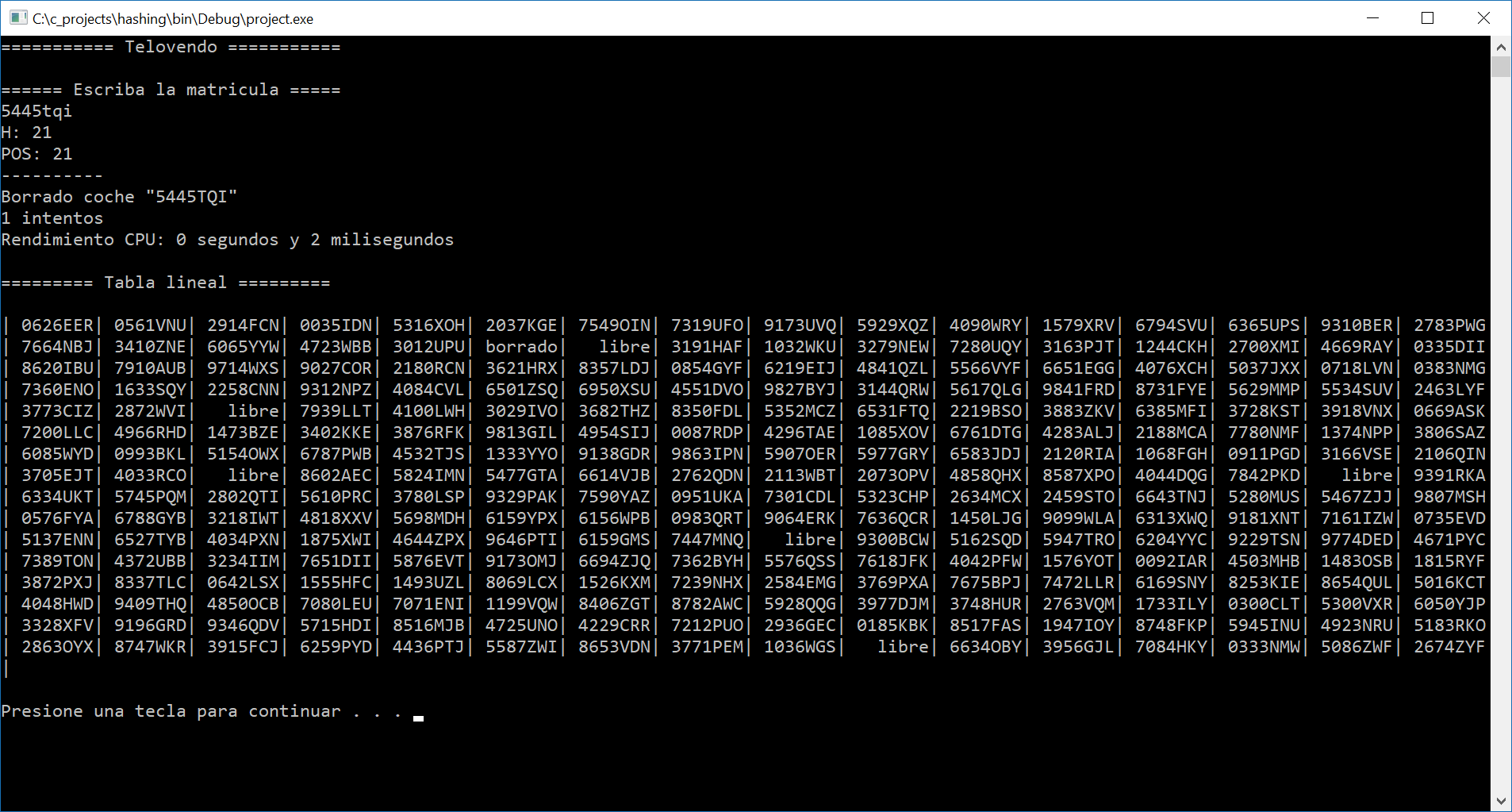
H (“5445TQI”, 1) = 21.



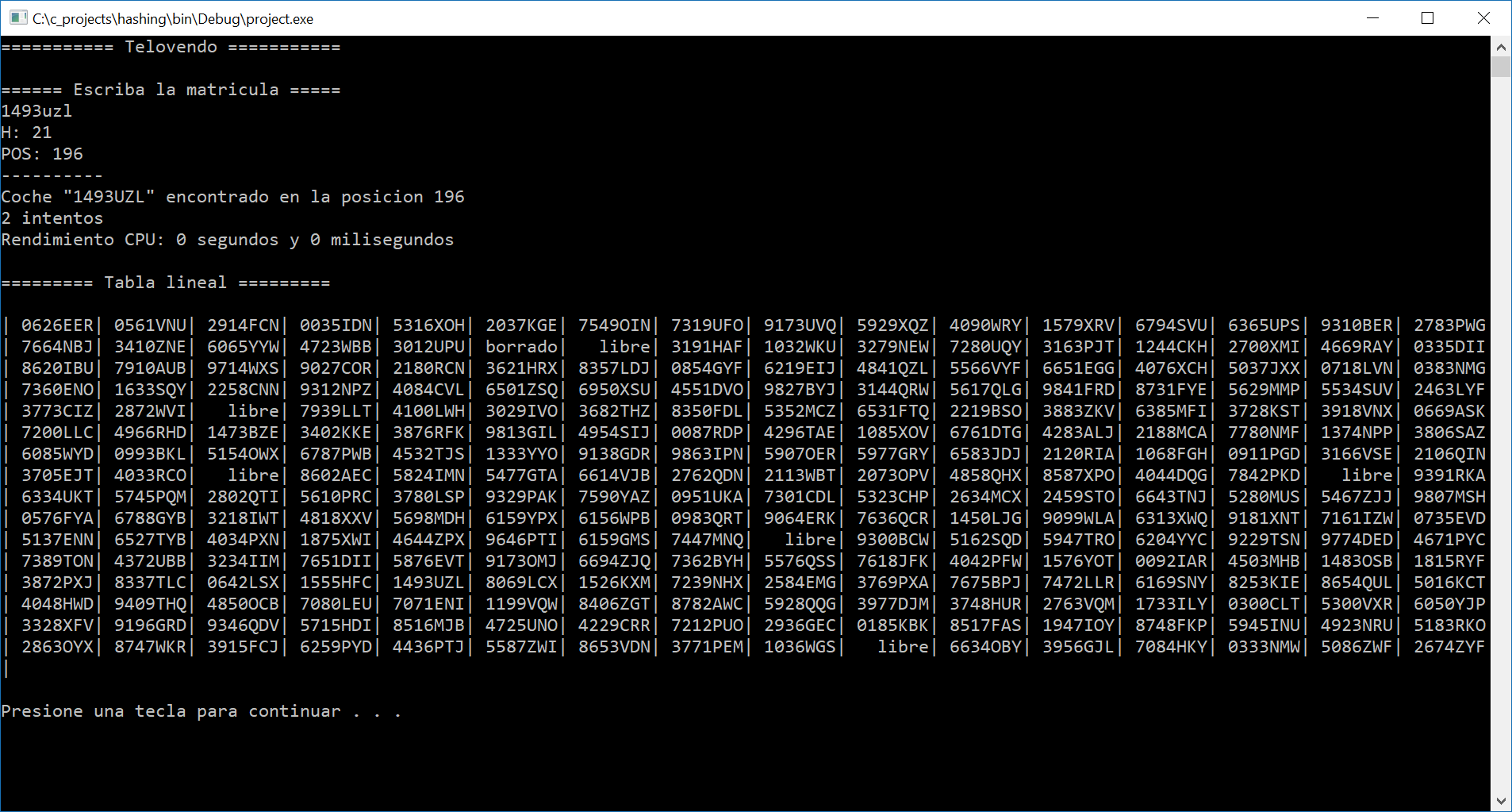
H (“1493UZL”, 1) = 21.  
H (“1493UZL”, 2) = 196.



Ahora lo borro:



Ahora recupero el de colisión:



Por lo tanto, lo hace bien.