Se realiza una tabla comparativa con los resultados obtenidos por cada uno de los algoritmos al efectuar las pruebas con entradas de texto:

Entropía:

Algoritmos de hash:	Entropía:
Algoritmo de hash creado	330
SHA1	206,797
SHA256	330,87
MD5	165,437

Al observar dichos valores se puede destacar lo siguiente:

- La entropía de los algoritmos es diferente, esto se debe a que al generar el hash los largos de los caracteres van variando dependiendo de qué algoritmo se está utilizando.
- El algoritmo MD5 posee la menor entropía al poseer un menor largo en su hash, por lo que existe menos variedad (dispersión) en su cifrado.
- El algoritmo de hash creado y SHA256 (prácticamente dan valores casi iguales) son los que poseen la mayor entropía, esto se debe al generar un hash más largo por lo que al tener una entropía más grande posee una mayor dispersión (variedad) en cuanto al cifrado.
- El algoritmo creado genera una mayor entropía que otros algoritmos como SHA1 y MD5 por lo que sería ideal para su uso al momento de cifrar las passwords.
- La entropía se calcula utilizando el valor de la base utilizada junto con el largo del hash, por lo mismo es directamente proporcional a dichos parámetros.
- Recordar que la entropía se refiere a una medida de dispersión que posee la contraseña, esta se calcula con la base utilizada y el largo
- Según Nist se estima que la entropía de una password debería tener un mínimo de 30 bits.
- Las bases utilizadas en SHA1, MD5 Y SHA256 es de 36, ya que dan valores numéricos del 0 al 9, además, solamente se compone de minúsculas al analizar los valores de hash obtenidos.
- La base utilizada en el algoritmo de hash creado es de 64.

En resumen para los cálculos de hash (vistos anteriormente) se tiene:

Algoritmos de hash:	Largo de password hasheada:	Base utilizada:
Algoritmo de hash cread	55	64
SHA1	40	36
SHA256	64	36
MD5	32	36

Rendimiento:

Se calcula el tiempo que demoran en ejecutarse los algoritmos para las diferentes pruebas, dando lo siguiente:

Algoritmos de hash	Prueba 1 (1 entrada de texto)	Prueba 2 (10 entradas de texto)	Prueba 3 (20 entradas de texto)	Prueba 4 (50 entradas de texto)
Algoritmo creado	0.00014 s	0.00082 s	0.00185 s	0.00402 s
SHA1	4.673e-05	0.000195 s	0.000452 s	0.00133 s
SHA256	5.698e-05 s	0.000291 s	0.000640 s	0.00147 s
MD5	3.314e-05 s	0.000205 s	0.000308 s	0.000814 s

Al ver los valores arrojados se destaca lo siguiente:

- El algoritmo creado demora más (es más lento) que los demás algoritmos de hash. Esto se debe a que realiza más procedimientos con los valores de entrada.
- MD5 demora menos (es más rápido) que los demás algoritmos en general, sobre todo a medida que se van aumentando la cantidad de entradas se va notando más la diferencia.
- SHA1 demora poco (es rápido) en un principio pero luego al ir aumentando la cantidad de las entradas este va demorando más.
- SHA256 es el segundo algoritmo que más demora en ejecutarse solo por debajo del algoritmo de hash creado, este va aumentando considerablemente al aumentar la cantidad de entradas de texto.

- Mediante los datos se puede decir que MD5 es el algoritmo que trabaja de forma más rápida.

A continuación se efectúa una tabla con los procedimientos realizados por cada algoritmo de hash:

Algoritmos de hash	Funcionamiento
Algoritmo creado	1. Lee las entradas y las transforma en base 64 2. Transforma en ascii 3. Transforma en hexadecimal 4. Aplica un padding (en caso de ser necesario) 5. Efectúa la salida (se muestra el valor de hash)
SHA1	1. Lee las entradas 2. Convierte la entrada a una de 40 caracteres, la cual está en hexadecimal que es una suma de comprobación de 160 bits.
SHA256	1. Lee las entradas 2. Convierte la entrada a una de 64 caracteres, que está en hexadecimal con una codificación de 256 bits (32 bytes).
MD5	1. Lee las entradas 2. Convierte la entrada a una de 32 caracteres, que está en valor hexadecimal correspondiente a una suma de comprobación de 128 bits.

Por lo mismo, se pueden ver dichas diferencias en los algoritmos en términos del funcionamiento, los cuales afectan en el rendimiento y en la entropía obtenida (visto anteriormente), ya que si bien tienen el mismo objetivo (el cual es generar un hash a partir de una entrada), estos trabajan de distintas formas.

En síntesis, si bien el algoritmo creado posee una mayor entropía a diferencia de otros algoritmos de hash, este toma más tiempo ya que realiza una serie

de pasos (base 64, ascii, hexadecimal y padding) para poder finalmente hashear una entrada, por lo que no es tan eficiente en términos de tiempo de ejecución, pero de igual manera cumple su objetivo el cual es poder garantizar confidencialidad e integridad en los datos, en este caso en las credenciales al poder efectuar el hash (también cumple con lo solicitado por el gremio de dueños de sitios webs). Además, se comprobó mediante pruebas realizadas que las entradas dan hashes únicos a cada entrada (al tener entradas diferentes o similares, dan hashes distintos y únicos, efecto avalancha), por lo que se evitan colisiones.

También se destaca que en el caso de querer aumentar la entropía del algoritmo de hash creado, se debe por ejemplo, aumentar el largo de este o la base utilizada, pero a la vez se puede intuir que trabajara de forma más lenta (tendrá un mayor tiempo de ejecución el algoritmo). En caso de querer disminuir dicha entropía, se debe disminuir el largo del hash o en su defecto utilizar una base más pequeña, por lo que tendrá una menor dispersión (variación), pero a la vez trabajara de forma mas rapida (en términos de tiempo de ejecución). Por ende, todos estos factores se deben tomar en cuenta al momento de efectuar una modificación del algoritmo para ya sea aumentar o disminuir la entropía.