Endnotes

1. Various properties make hashing an interesting tool to ensure data integrity and prevent forgery:

1. A given set of data (input) will always give the same hash (output). However, a small change in the input – such as a single change in letter cases or in punctuation – will completely and drastically change the resulting hash. This is crucial when it comes to verifying the integrity of data. If a piece of information produces different hashes for the sender and the receiver, this means that it has been tampered with during transit. Likewise, if a record at rest changes hash between the time of its creation and the time of verification, this indicates that it has been subject to modification during the interval. This makes it useful to immediately detect forgery when it occurs.
2. It is infeasible to determine the original input on the basis of its hash value. The probability to "guess" it is so low that it requires a daunting amount of computational power.
3. Comparing an output with an input is very rapid, which is very useful when it comes to digesting big data files and in cases where responsiveness is required – e.g. in user's identity verification processes; when users type their password to access a service, the system immediately hashes the typed password and compares it to the stored hash.
4. Hash functions are said to be "puzzle-friendly": if an output is obtained by combining two sets of input, it is nearly impossible to identify the value of one of them if you already know the second.
5. Hash functions are "collision-resistant": the likelihood that two different inputs would randomly give the same output is extremely limited, which is another aspect that contributes to data integrity.

2. In the case of the proof-of-work protocol used, for example, by Bitcoin, the term used is that the block is "mined".

3. See https://iota.readme.io/docs/what-is-iota