The simple uniform hashing assumption assumes that any given element is equally likely to hash into any of the m slots, independently of where any other element has hashed to. Since non-empty sets A, B are finite, then that means there exists finitely many different functions.

With domain A and codomain B, there exists $|S| = |B|^{|A|}$ functions, where S is the set of all different functions, and |S| is the number of different functions that exist.

As the problem mentions, we pick a function uniformly from S at random, so the probability of picking either one function is $\frac{1}{|S|}$. Since for each function within S, it would always map a particular value in domain A to a particular value in codomain B, and the only randomness that exists is the choice of choosing a function f from the set of functions S.

Hence, for a random function $f \in$ S, such that f: U → {0, … , m – 1}, when give a value a ∈ U, it would always map the same a to the same b ∈ {0, … , m – 1}. Since there are $|B|^{|A|}$ functions in total, it means that there are $\frac{|B|^{|A|}}{m}$ functions that would map the same a ∈ U to the same b ∈ {0, … , m – 1}. Hence, the probability of hashing any given element to a slot, for all k ∈ U and i ∈ {0, … , m – 1} is

$$\Pr\{f(k) = i\} = \frac{|B|^{|A|}/m}{|B|^{|A|}} = \frac{|B|^{|A|}}{m} \cdot \frac{1}{|B|^{|A|}} = \frac{1}{m}$$

Hence, the random function has the simple uniform hashing property.