#) Hashing

*) Hash Function
   ↳ It's a method for computing array Index
        from key.

   Ex:-)   Hash("Akhil") = 3 ← output (Index)
                          ↑
                     Input (key)

   Note:-) We use Index to store (key).

Complexity with Hf
   ↳ *) Computing Hash function. (complicated for complex Types).
   ↳ *) Equality Test (checking generated key == Existing key).
   ↳ *) Collision Re-solution :-)
        ↳ *) Two different key's win be hashed to Same key.

            Ex:-) Hash("Akhil") = ③ ⟶ "Collision".
                   Hash("Sundar") = ③

#) Importance of (Hf) in different Case's:-)

i) No - Space Limitation :-)
   *) If we don't have any problem with the "Space" then,
      we can generate many different key's as possible
      for Every array index.
           ↳ By This :⟶ i) We Can Create Long dynamic Array
                ↳ By this, "We can use (key) itself as a Index".
                     ↳ O(1) .

*) If we have No Time Limitation
   ↳ i) Hash Everything to Same Place.
   ↳ ii) Apply Sequential Search.
   Note:-) It's a bit Slow

#) Computing Hash Function

i) "Key" ⟶ [Hf] ⟶ ii) "Table Index"

Problem
   ↳ i) Every Key Need a Hash Function.
   ↳ ii) We should apply diff approach for "Every different Key Type".
        Note:-) ⟹ Basic Types:-) Int, float, double, String.
             ⟹ Custom Type :-) Implementing a diff approach for a Custom Type is a typical Task.

        ⟹ Key Characterstics of Hash Function
             i) Same Input ⟶ Same output (Deterministic)
             ii) Dynamic Size Input ⟶ Hash Should be Same Length.
             iii) Pre-Image Resistance ⟶ Hashvalue ⟶ key/x)
                                          key ⟶ Hvalue (✓)
             iv) Collision Resistance.
             v) Avalanche Effect:-) Input (small change) ⟶ Different output.

                  Ex:-) #) Simple Hash
                       i) Pick Input: 12345
                       ii) Pick Prime no (M) as modulus :-) ③1
                       iii) Hash value ⟹ 12345 ÷ 31 = ⑱

        # #) Collision Resolution Methods:-)

             ii) Seperate Chaining

             #) Linear Probing
                *) It's Called Open-Addressing.
                *) Idea:-) Size of an Array should be bigger than the "Key's".