# Workshop HashTable

*In computing, a hash table (hash map) is a data structure that implements an associative array abstract data type, a structure that can map keys to values. A hash table uses a hash function to compute an index, also called a hash code, into an array of buckets or slots, from which the desired value can be found. During lookup, the key is hashed and the resulting hash indicates where the corresponding value is stored.*

## Overview

Create a **HashTable class** that should have the needed functionality for a hash table, such as:

* **hash(key: str)** - a function that should figure out **where** to store the key-value pair
* **add(key: str, value: any)** - adds a **new key-value** pair usign the hash function
* **get(key: str)** – **returns the value** corresponding to the **given key**
* additional **"magic"** methods, that will make the code in the **example** work correctrly

The HashTable should have an **attribute** called **array** of **type: list**, where all the values will be stored. Upon **initialization** the **default length** of the array should be **4**. After each addition of an element if the HashTable gets too **populated**, **double** the **length** of the array and **re-add** the existing data.

**You are not allowed to inherit any classes.** Feel free to implement your own functionality (and unit tests) or to write the methods by yourself.

## Examples

|  |  |
| --- | --- |
| **Test Code** | **Result** |
| table = HashTable()  table["name"] = "Peter"  table["age"] = 25  print(table)  print(table.get("name"))  print(table["age"])  print(len(table)) | <hash\_table.HashTable object at 0x00000185F4F08580>  Peter  25  4 |