Exercise 1 – Hash function for string: Sum of ASCII codes

```
def hash1(key_word, table_size):
        sum = 0
        for pos in range(len(key_word)):
                sum = sum + ord(key_word[pos])
        return sum % table_size
def main():
  print("table size is 13")
  for key_wd in ["cat","dog","god","abracadabra","abraabracad"]:
        print(key_wd, hash1(key_wd, 13))
```

```
table size is 13
cat 0
dog 2
god 2
abracadabra 3
abraabracad 3
```

Using the above hashing algorithm, which kind of keys will cause collisions?

Exercise 2 – Hash function for string: Weighted sum of ASCII codes

- Improve the previous algorithm by adding a weighting to each character (1 for the first, 2 for the second, ...).

```
def hash2(key_word, table_size):
        sum = 0
        for pos in range(len(key_word)):
                sum = sum + (pos+1) * ord(key_word[pos])
        return sum % table_size

def main():
  print("table size is 13")
  for key_wd in ["cat","dog","god","abracadabra","abraabracad"]:
        print(key_wd, hash2(key_wd, 13))
```

```
table size is 13
cat 4
dog 7
god 1
abracadabra 9
abraabracad 1
```

Exercise 3

- Insert the following items into the hash table below and indicate any collisions:
  - 11, 25, 63, 99, 12, 35, 54, 87, 66, 75, 91
- Hashing function:

$$h(item) = item \,\%\, 11$$

C C			C							C
11	12	35	25					63	75	54
0	1	2	3	4	5	6	7	8	9	10
99			91							87
66										