**Exercise no**:: 9

**Date**:: 23/11/2020

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Aim:**  Fill the missing words.

**Program:**

Print(‘\n—dictionaries’)  #Output: -- dictionaries

D = {‘a’: 1, ‘b’: 2}

Print(d[‘a’]) #Output: 1

Del d[‘a’]

# iterate

D = {‘a’: 1, ‘b’: 2}

For key, value in d.items():

    Print(key, ‘:’, value)

For key in d:

    Print(key, d[key])

# d.fromkeys(iterable[,value=None]) -> dict: with keys from iterable and all same value

D = d.fromkeys([‘a’, ‘b’], 1)

Print(d) #Output: {‘a’: 1, ‘b’: 1}

# d.clear() -> removes all items from d

D = {‘a’: 1, ‘b’: 2}

d.clear()

print(d) #Output: {}

# d.items() -> list: copy of d’s list of (key, item) pairs

D = {‘a’: 1, ‘b’: 2}

Print(d.items()) #Output: [(‘a’, 1), (‘b’, 2)]

# d.keys() -> list: copy of d’s list of keys

D = {‘a’: 1, ‘b’: 2}

Print(d.keys()) #Output: [‘a’, ‘b’]

# d.values() -> list: copy of d’s list of values

D = {‘a’: 1, ‘b’: 2}

Print(d.values())  #Output: [1, 2]

# d.get(key,defval) -> value: d[key] if key in d, else defval

D = {‘a’: 1, ‘b’: 2}

Print(d.get(“c”, 3)) #Output: 3

Print(d) #Output: {‘a’: 1, ‘b’: 2}

# d.setdefault(key[,defval=None]) -> value: if key not in d set d[key]=defval, return d[key]

D = {‘a’: 1, ‘b’: 2}

Print(‘d.setdefault(“c”, []) returns ‘ + str(d.setdefault(“c”, 3)) + ‘ d is now ‘ + str(d))

#Output: d.setdefault(“c”, []) returns 3 d is now {‘a’: 1, ‘b’: 2, ‘c’: 3}

#d.pop(key[,defval]) -> value: del key and returns the corresponding value. If key is not found, defval is returned if given, otherwise KeyError is raised

D = {‘a’: 1, ‘b’: 2}

Print(‘d.pop(“b”, 3) returns ‘ + str(d.pop(“b”, 3)) + ‘ d is now ‘ + str(d))

#Output: d.pop(“b”, 3) returns 2 d is now {‘a’: 1}

Print(‘d.pop(“c”, 3) returns ‘ + str(d.pop(“c”, 3)) + ‘ d is still ‘ + str(d))

#Output: d.pop(“c”, 3) returns 3 d is still {‘a’: 1}

# sort on values

Import operator

X = {1: 4, 5: 4, 4: 4}

Sorted\_x = sorted(x.items(), key=operator.itemgetter(1), reverse=True)

#Output: print(‘sorted(x.items(), key=operator.itemgetter(1)) sorts on values ‘ + str(sorted\_x))

# max of values

D = {‘a’:1000, ‘b’:3000, ‘c’: 100}

Print(‘key of max value is ‘ + max(d.keys(), key=(lambda key: d[key])))

#Output: key of max value is b

**Result:** The program has been successfully verified