Java

```
package Package;
import java.util.HashMap;
import java.lang.reflect.Method;
public class Class_01 {
        public static void main(String[] args) {
                Class<?> clas = HashMap.class;
                Method[] mtd = clas.getDeclaredMethods();
                for(Method m: mtd) {
                        System.out.print(m.getName() + ", ");
                }
        }
}
```

Output

```
remove, remove, size, get, put, values, clone, clear, hash, isEmpty, replace, replace, replaceAll,
merge, newHashMap, entrySet, putAll, putIfAbsent, compute, readObject, writeObject, forEach,
containsKey, computeIfAbsent, keySet, containsValue, getOrDefault, computeIfPresent,
tableSizeFor, loadFactor, capacity, putVal, treeifyBin, comparableClassFor,
compareComparables, resize, putMapEntries, getNode, newNode, afterNodeAccess,
afterNodeInsertion, replacementTreeNode, removeNode, afterNodeRemoval, reinitialize,
internalWriteEntries, calculateHashMapCapacity, prepareArray, keysToArray, valuesToArray,
replacementNode, newTreeNode,
```

Javascript

```
const obj = {};
const methods = Object.getOwnPropertyNames(Object.getPrototypeOf(obj)).filter(method =>
typeof obj[method] === 'function');
console.log('Dictionary Methods:');
methods.forEach(method => console.log(method));
```

Output

```
Dictionary Methods:
constructor
__defineGetter__
__defineSetter__
hasOwnProperty
__lookupGetter__
```

```
__lookupSetter__
isPrototypeOf
propertyIsEnumerable
toString
valueOf
toLocaleString
```

Python

```
# Get all dictionary methods using dir()
dict_methods = dir(dict)

# Filter methods that do not start with '__' (those are dunder methods)
public_dict_methods = [method for method in dict_methods if not method.startswith('__')]

# Print all dictionary methods
for method in public_dict_methods:
    print(method)
```

Output

```
clear
copy
fromkeys
get
items
keys
pop
popitem
setdefault
update
values
```