A HashMap is a collection that associates two kinds of objects, a key and a value. Examples are:

- A dictionary
  (key = word, value = meaning)
- A phone book
  (key = name, value = number)

A HashMap is:

- unordered
- index by keys
- variable size

Note: there can be only one value for each key, but values may be repeated for different keys.

The HashMap is part of the Java class library and so must be imported before it is used:

```java
import java.util.HashMap;
```

The API docs can be found at:

http://download.oracle.com/javase/7/docs/api/java/util/HashMap.html
HashMap definitions

HashMaps take two type parameters – the key type and the value type:

HashMap<String, Integer> mapStringToNumber;
HashMap<Book, Author> mapBookToAuthor;

As with ArrayLists, we cannot have HashMaps of primitive types (either key or value).

Construction

To construct an HashMap we will usually use the simple no-parameter constructor:

```java
// create an empty map
HashMap<String, Integer> marks = new HashMap<String, Integer>();
```

Adding items

A newly constructed HashMap is empty. We can add items using the put method.

```java
marks.put("Jim", 90);
marks.put("Fred", 55);
```

Overwriting items

Each key has only one value. Adding a new one overwrites the old:

```java
marks.put("Fred", 65);
```
Getting items

We can read items using the `get` method.

```java
marks.get("Jim");
// returns 90
marks.get("Jane");
// returns null
```

Integers, `ints` and `null`

An important difference between Integers and `ints`.

Integer variables are object references and so they can be `null`.

`int` variables are numbers and cannot be `null`.

This sometimes causes problems with maps:

```java
int mark = marks.get("Jim");
// Integer 90 is converted to int
mark = marks.get("Jane");
// ERROR:
// Cannot convert null to int
```

Solution

If there is doubt, always check the value first:

```java
Integer mark = marks.get("Jane");
if (mark == null)
    // handle the null case
else
    // handle non-null case
```
Iterating

We cannot use the for-each loop directly on the HashMap:

```java
for (String who: marks)
{
    // ERROR: will not compile
}
```

However we can iterate over the key set:

```java
for (String who: marks.keySet())
{
    Integer value = marks.get(who);
    // do something with who/mark
}
```

Other methods

There are other methods that you should read about in the API:

- `containsKey` - check if a key is in the map
- `size` - how many entries in the map
- `remove` - remove an entry by key

Practise reading the API docs.