HashMap<String,Integer> map =new HashMap<>();

        map.put("a", 1);

        map.put("b", 2);

        map.put("c", 3);

        map.put("d", 4);

//hashMap遍历方法1

         Iterator<String> iterator = map.keySet().iterator();

         while (iterator.hasNext()) {

          Object key = iterator.next();

          System.out.println("map.get(key) is :"+map.get(key));

         }

System.out.println(map.containsValue(1) );

          System.out.println(map.containsKey("a") );

          System.out.println(map.size() );

          System.out.println(map.remove("a") );

          System.out.println(map.size() );

          System.out.println(map.containsKey("a") );

          System.out.println(map.containsValue(1) );

System.out.println(map.isEmpty());

获得key对应的值:map.get(key);

修改对应的值：map.replace(key , new\_value);

# [HashMap的基本使用](https://www.cnblogs.com/hanlk/p/11229058.html)

### 常用方法

首先，我们应该知道HashMap类实现了Map接口，所以实现了Map常用的一些方法。

（1） 插入键值对数据

public V put(K key, V value)

（2）根据键值获取键值对值数据

public V get(Object key)

（3）获取Map中键值对的个数

public int size()

（4）判断Map集合中是否包含键为key的键值对

public boolean containsKey(Object key)

（5）判断Map集合中是否包含值为value的键值对

boolean containsValue(Object value)

（6）判断Map集合中是否没有任何键值对

public boolean isEmpty()

（7）清空Map集合中所有的键值对

public void clear()

（8）根据键值删除Map中键值对

public V remove(Object key)

### 键值对数据遍历

（1）将Map中所有的键装到Set集合中返回

//public Set<K> keySet();  
Set<K> set=map. keySet()

（2）返回集合中所有的value的值的集合

// public Collection<V> values();  
Collection<V> c=map.values()

（3）将每个键值对封装到一个个Entry对象中,再把所有Entry的对象封装到Set集合中返回

// public Set<Map.Entry<K,V>> entrtSet();  
Set<Map.Entry<K,V>> entrys=map.entrySet()