Object Oriented Programming
CS F213

J. Jennifer Ranjani

email: jennifer.ranjani@pilani.bits-pilani.ac.in

Chamber: 6121 P, NAB

Consultation: Friday 4-5 p.m.

BITS Pilani

Pilani Campus

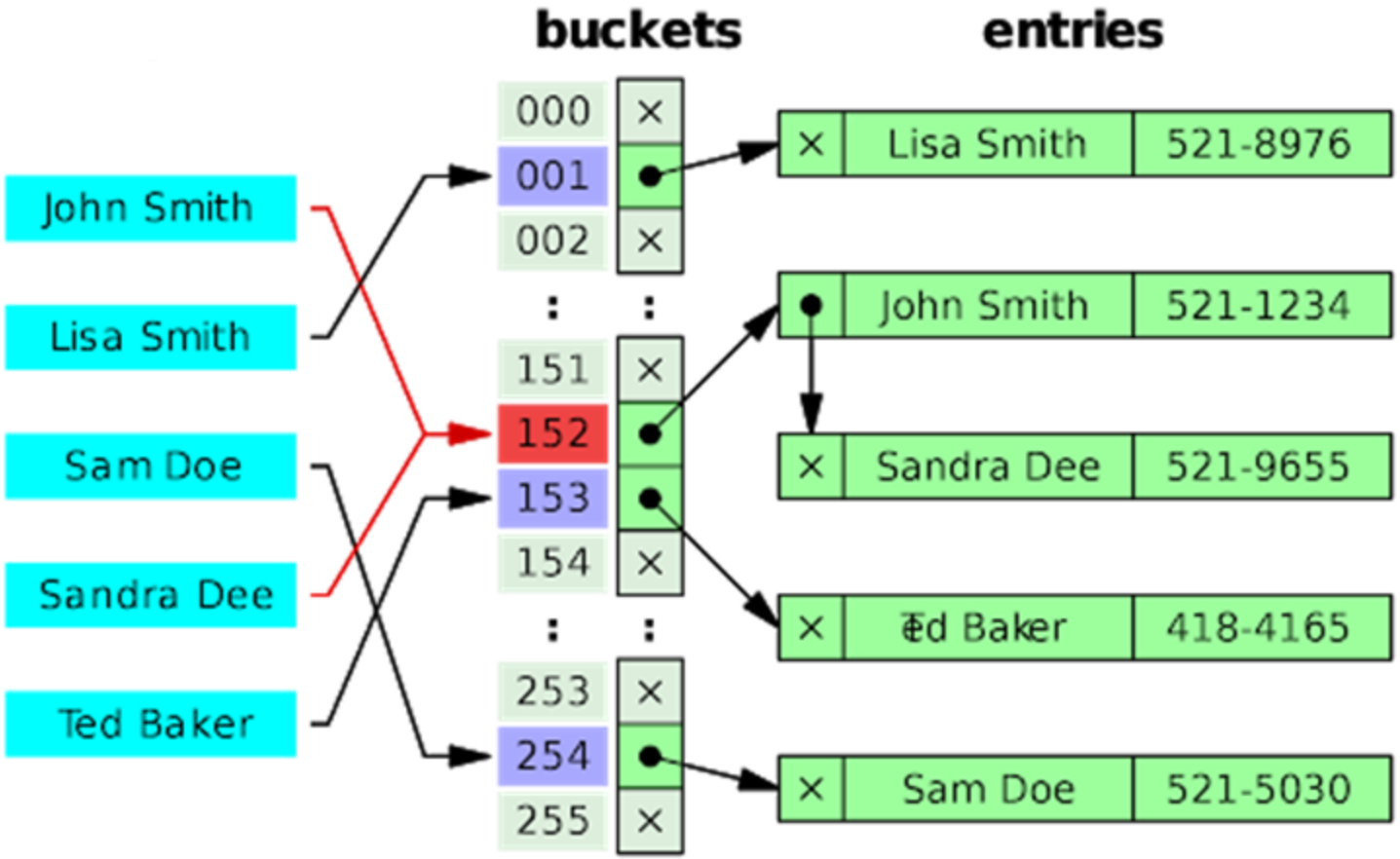
Recap

- List
  - Example
    - ArrayList - implements RandomAccess Interface, hence its is suitable if elements are to be retrieved frequently.
    - LinkedList – suitable if elements are added/removed frequently.
  - Insertion order is preserved
  - Duplicate elements are allowed
  - Multiple null values are allowed.

Recap

- Set
  - Example
    - HashSet, LinkedHashSet etc
      - » The time required for adding/removing/retrieving elements is constant even for large datasets.
  - Unordered collection
  - Duplicate values are not allowed

Example


buckets
entries
John Smith
Lisa Smith
Sam Doe
Sandra Dee
Ted Baker
000
001
002
151
152
153
154
253
254
255
Lisa Smith
521-8976
John Smith
521-1234
Sandra Dee
521-9655
Ted Baker
418-4165
Sam Doe
521-5030

General Contract for hashCode()

- During an execution, hashCode() invoked on an object always yields the same integer.
    - It might vary from one execution to another
- If two objects are equal, hashCode() on both should yield the same integer.
- Two unequal objects can also provide same integer when hashCode() is invoked.
    - hashCode() is basically used for identifying the bucket and equals() is used for identifying the position within the bucket.
- If equals() is overridden, it is mandatory to override hashCode() according to the contract.

innovate achieve lead

BITS Pilani

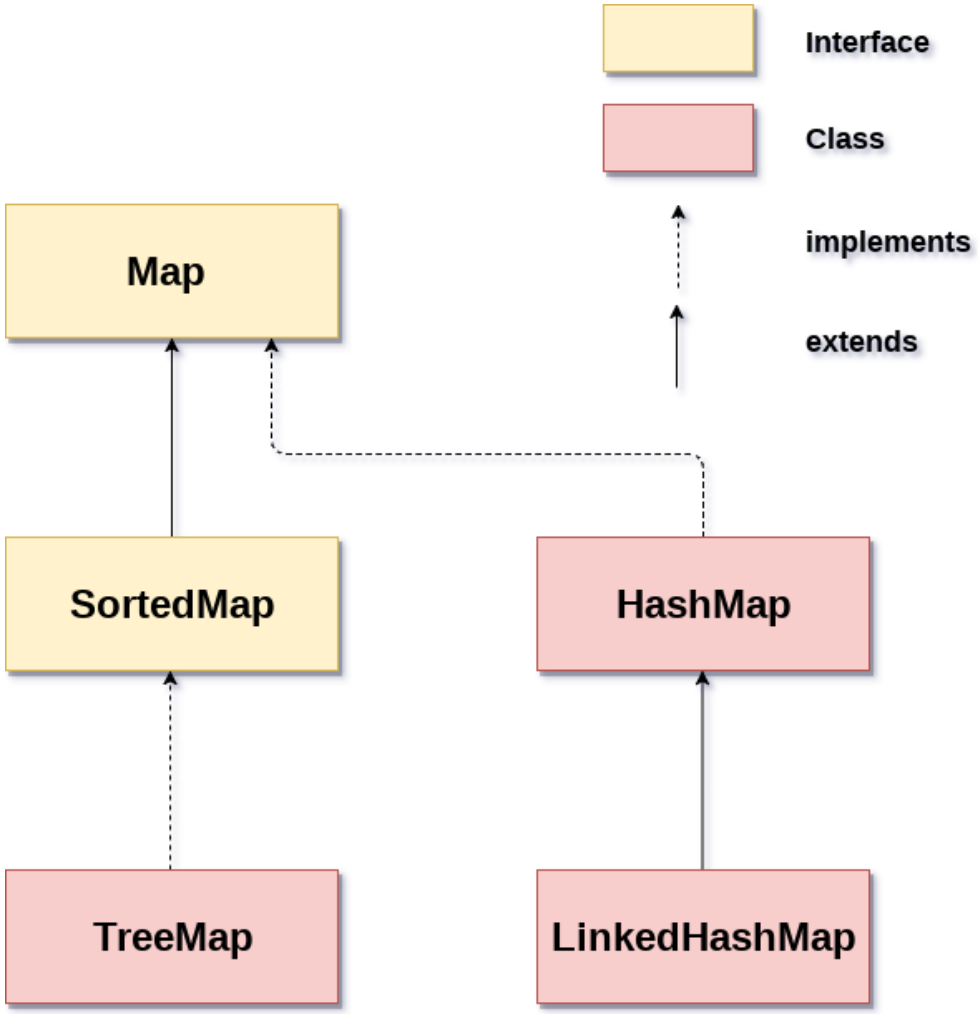
Pilani Campus

Map Interface

Map Interface

- A map contains values on the basis of key i.e. key and value pair.
- Each key and value pair is known as an entry.
- Map contains only unique keys.
- Map is useful if you have to search, update or delete elements on the basis of key.

Map Hierarchy



HashMap – no specific order

LinkedHashMap – maintains insertion order

TreeMap – maintains ascending order

Map Interface

Method	Description
Object put(Object key, Object value)	It is used to insert an entry in this map.
void putAll(Map map)	It is used to insert the specified map in this map.
Object remove(Object key)	It is used to delete an entry for the specified key.
Object get(Object key)	It is used to return the value for the specified key.
boolean containsKey(Object key)	It is used to search the specified key from this map.
Set keySet()	It is used to return the Set view containing all the keys.
Set entrySet()	It is used to return the Set view containing all the keys and values.

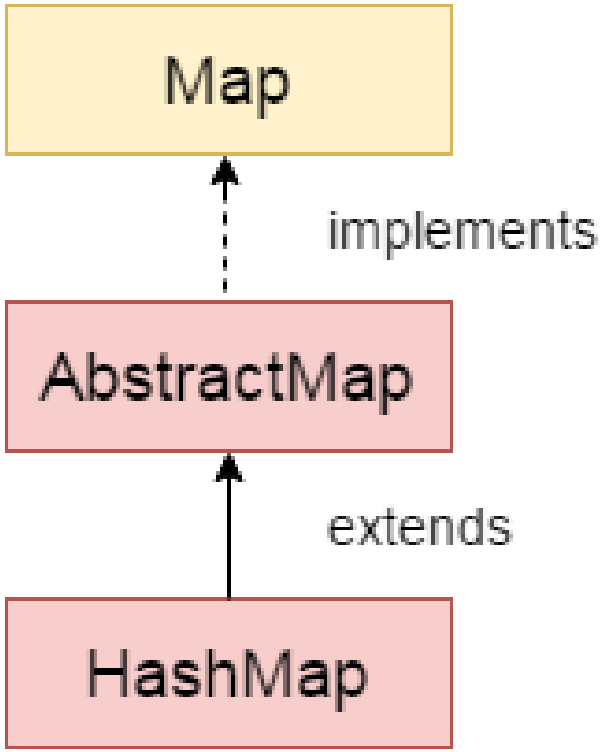
Map.Entry Interface

- Entry is the sub interface of Map.
- It provides methods to get key and value.

Method	Description
Object getKey()	It is used to obtain key.
Object getValue()	It is used to obtain value.

HashMap

- A HashMap contains values based on the key.
- It contains only unique elements.
- It may have one null key and multiple null values.
- It maintains no order.



Hash Map

```
class test {
public static void main(String[] args) {
  HashMap<Integer,String> hm=new HashMap<Integer,String>();
  hm.put(100,"Amit");
  hm.put(101,"Vijay");
  hm.put(102,"Rahul");
  hm.put(102,"RaKul");
  hm.put(null,null);
  hm.put(103,null);
  for(Map.Entry<Integer,String> m:hm.entrySet()){
   System.out.println(m.getKey()+" "+m.getValue());
  }
}
}
```

Output:

null null
100 Amit
101 Vijay
102 RaKul
103 null

innovate achieve lead

BITS Pilani

Pilani Campus

Wrapper Classes

Wrapper Class

- Converts a primitive into object (Autoboxing) and object into a primitive (unboxing)

Primitive Type	Wrapper class
boolean	Boolean
char	Character
byte	Byte
short	Short
int	Integer
long	Long
float	Float
double	Double

innovate achieve lead

BITS Pilani

Pilani Campus

Autoboxing and Unboxing (Java 5)

Wrapper Class & Boxing - Example

innovate achieve lead

```
int a = 50;
Integer i = Integer.valueOf(60);
Integer j = a;  // auto boxing
System.out.println(j.compareTo(i));
```

Output:
-1

```
Integer i = new Integer(50);
int a = i;  // unboxing
int b = Integer.numberOfLeadingZeros(i);
 System.out.println("Unboxing"+a+"Int Value"+b);
```

Output:
26

Note: Binary value of 50 is 0b110010; int is 32 bits long

Method overloading & Autoboxing

innovate achieve lead

```
class Boxing1{
  static void m(int i){System.out.println("int");}
  static void m(Integer i){System.out.println("Integer");}


  public static void main(String args[]){
   short s=30;
   m(s);
 }
}
```

Output:
int

Widening beats Boxing

Method overloading & Autoboxing

innovate achieve lead

```
class Boxing2{
  static void m(int i, int i2){System.out.println("int int");}
  static void m(Integer... i){System.out.println("Integer...");}

  public static void main(String args[]){
   short s1=30,s2=40;
   m(s1,s2);
 }
}
```

Output:
int int

Widening beats VarArgs

Method overloading & Autoboxing


```
class Boxing3{
  static void m(Integer i){System.out.println("Integer");}
  static void m(int... i){System.out.println("int...");}

  public static void main(String args[]){
   int a=30;
   m(a);
 }
}
```

Output:
Integer

Boxing beats VarArgs

Method overloading & Autoboxing

innovate achieve lead

```
class Boxing4{
  static void m(Long l){System.out.println("Long");}

  public static void main(String args[]){
   int a=30;
   m(a);
 }
}
```

Output:
Compile Time Error

Widening and Boxing cant be performed

Review Questions

```
static void print(int i,int j){System.out.println("int");}
static void print(Integer i,Integer j) {System.out.println("Integer");}
static void print(Integer... i){System.out.println("Var Integer");}
public static void main(String args[]){
 short s=30,t=50;
 Integer a=30,b=50,c=70;
// Place any of the following statements  }
```

What happens when the following print() are inserted in the code?

a. **print(s)**
b. **print(s,t)**
c. **print(a,b)**
d. **print(a,b,c)**

Review Questions

```
static void print(int i,int j){System.out.println("int");}
static void print(Integer i,Integer j) {System.out.println("Integer");}
static void print(Integer... i){System.out.println("Var Integer");}
public static void main(String args[]){
 short s=30,t=50;
 Integer a=30,b=50,c=70;
// Place any of the following statements  }
```

What happens when the following print() are inserted in the code?

a. **print(s)** **// Error**
b. **print(s,t)** **// int**
c. **print(a,b)** **//Integer**
d. **print(a,b,c)** **//Var Integer**