

TEST 2 REVIEW
MONDAY NOVEMBER 25

Implement a Map

keys	values
k1	v1
k2	v2
k3	v1

may have duplicates

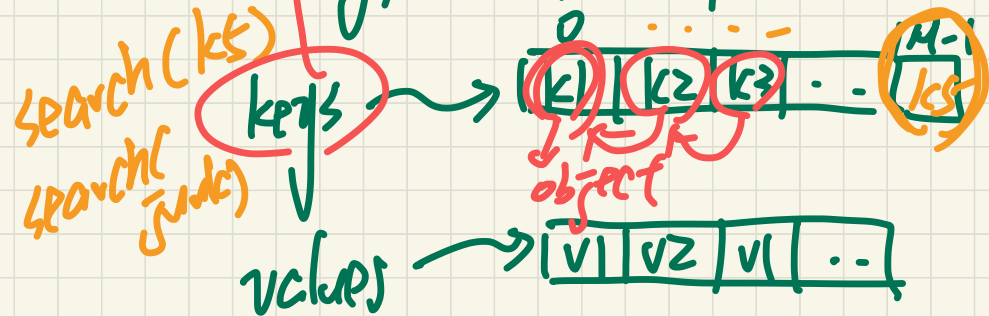
unique

map(k1)
map(k3)

v1
v1

size: N
↳ search: $O(N)$
scale
↳ size of map: $1M$

Strategy 1



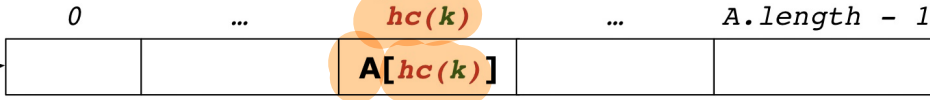
Strategy 2

Implementing a Hash Table via Hashing

any type of object $\leftarrow k$

hashing

hash code \rightarrow determines where to store the key.



- Converting k to $hc(k)$
- Indexing into $A[hc(k)]$

For illustration, assume $A.length$ is 11 and $hc(k) = k \% 11$

$hc(k) = k \% 11$	(SEARCH) KEY	VALUE
	1	D
25	3	C
3	14	F
	6	Z
	39	A
		C
		2

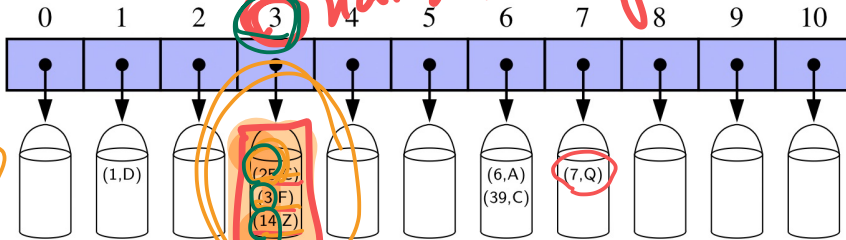
search(k)
 \rightarrow calculate $hc(k)$ $O(1)$
 $\rightarrow A[hc(k)]$ $O(1)$

collision. $O(1)$

same hash code
 distinct keys

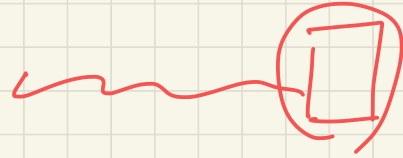
hash code \rightarrow used as index
 $hc(k)$ \rightarrow index
 $A[hc(k)]$ \rightarrow value

not going to be used directly as index



ArrayList < .. > list

list.add(..)

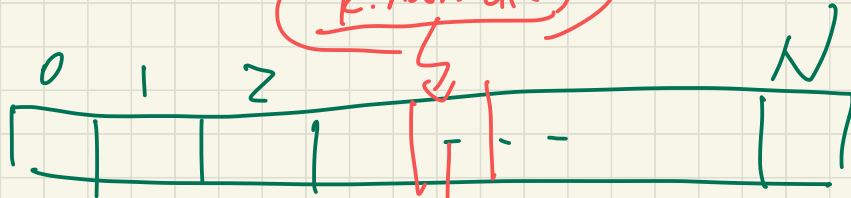


HashMap < Integer, String > table =

table.add(k, v);
↳ k.hashCode() →
ba

table.add(k2, v2);
↳ k2.hashCode() →
" " k.hashCode()

new HashMap<>();



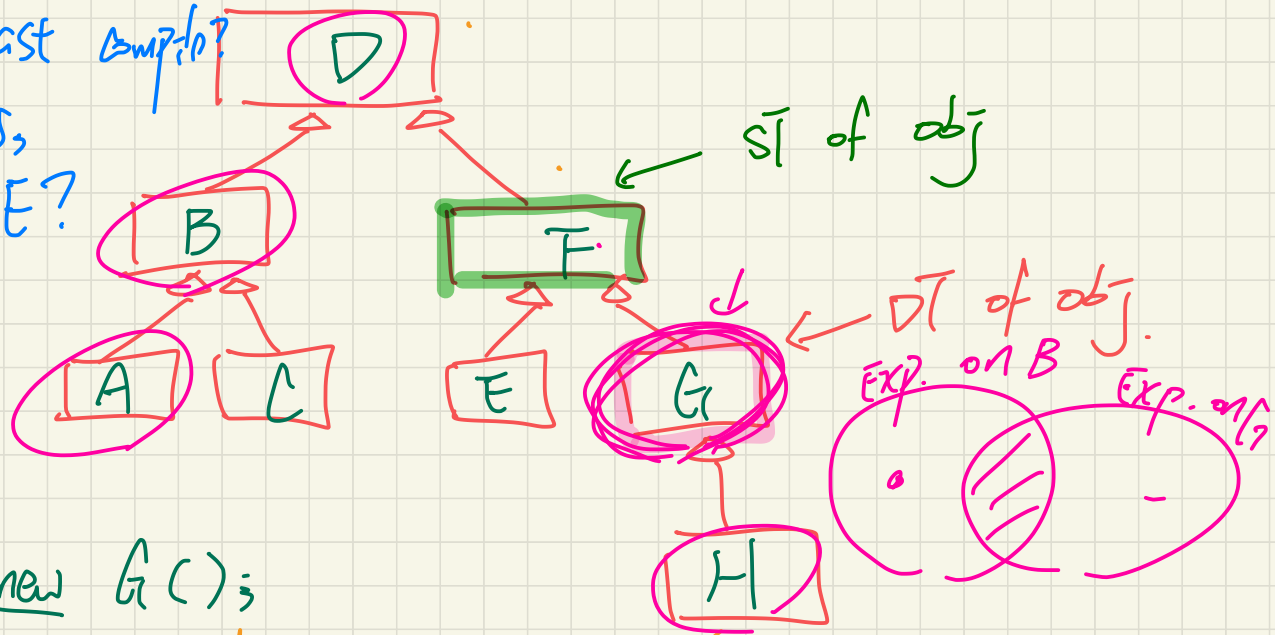
(k, v)
(k2, v2)

Bucket Array .

ArrayList < ArrayList < ^{Entry} > >

Q1. does the cast compile?

Q2. If it compiles, do we have CCE?



F obj = new G();

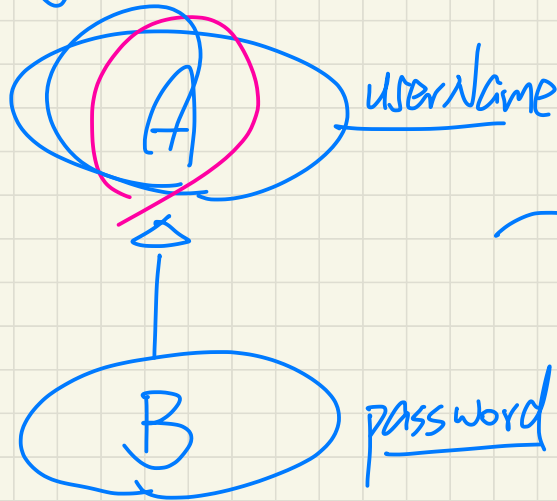
- ① (D) obj ✓ ∵ upward.
 - ② (G) obj ✓ ∵ downward
 - ③ (H) obj ✓ Compiles ∵ down
 - ④ (B) obj ✓
 - ⑤ (A) obj ✓
- X CCE occ.

When do we get a CCE?

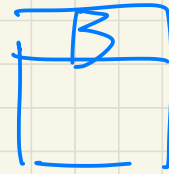
C (obj) des. of

1. When DI of obj is not prep. on
2. When DI of obj cannot fulfill

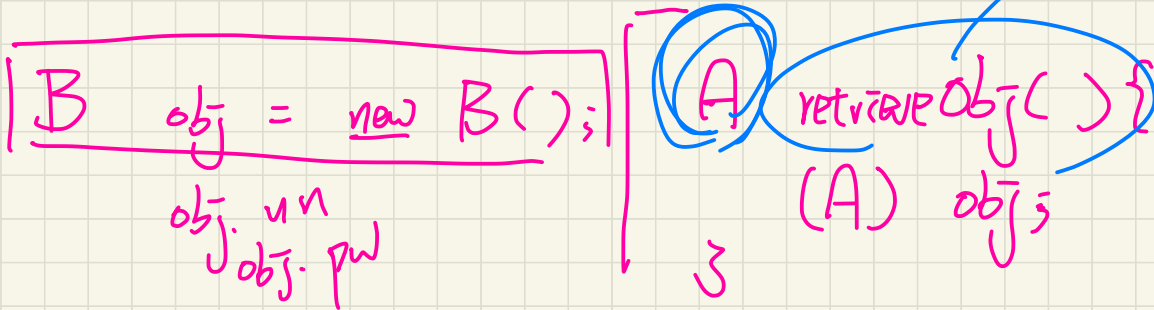
Upward Casting : Access Control



retrieveObj(). un
• password



caller would only retrieve on object of static type A.



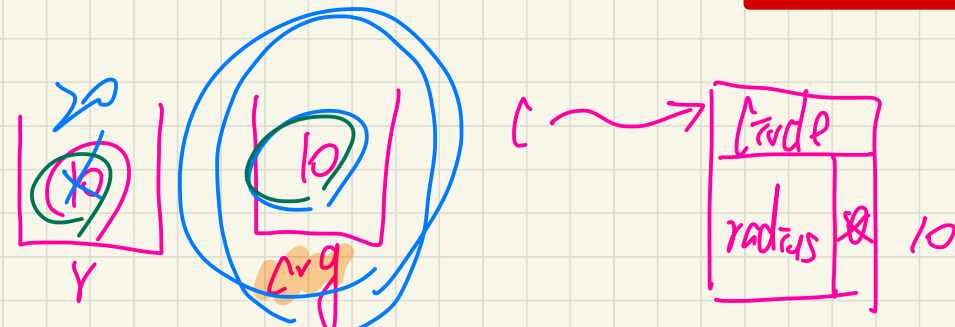
Call by Value: Primitive Argument

```
class Circle {  
    int radius;  
    void setRadius(int r) {  
        this.radius = r;  
    }  
}
```

Handwritten notes:
- A pink arrow points from the text "primitive parameter" to the parameter `int r`.
- The parameter `int r` is circled in pink.
- A green circle around `r` is labeled `r = arg`.
- A blue arrow points from the closing brace of `setRadius` to the text `r = 20;`.

```
class CircleUser {  
    ...  
    Circle c = new Circle();  
    int arg = 10;  
    c.setRadius(arg);  
}
```

Handwritten notes:
- The variable `arg` is circled in green.
- The value `10` is circled in orange.



Call by Value: Reference Argument

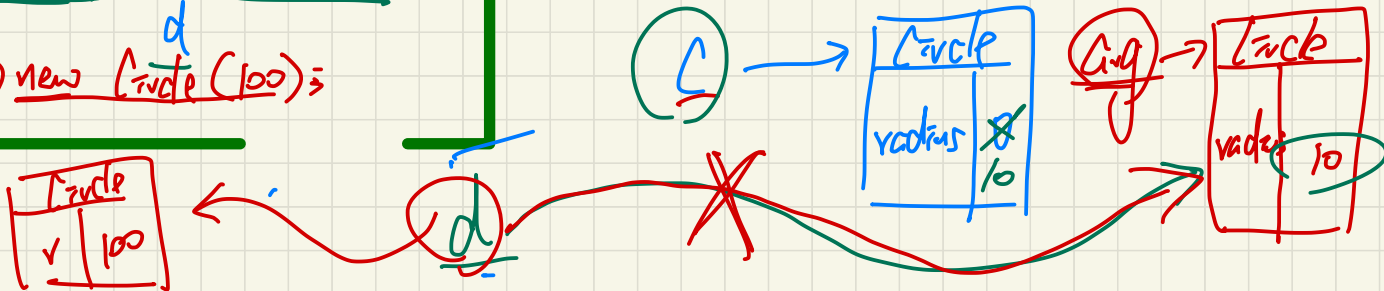
```
class Circle {  
  int radius;  
  Circle() {}  
  Circle(int r) {  
    this.radius = r;  
  }  
  void setRadius(Circle c) {  
    this.radius = c.radius;  
  }  
}
```

Reference Argument.

d() new Circle(100);

```
class CircleUser {  
  ...  
  Circle c = new Circle();  
  Circle arg = new Circle(10);  
  c.setRadius(arg);  
}
```

address of some code object.



```

class Circle {
  int radius;
  Circle() {}
  Circle(int r) {
    this.radius = r;
  }
  void setRadius(Circle o) {
    this.radius = o.radius;
  }
}

```

Implicitly:
 d = arg;

d

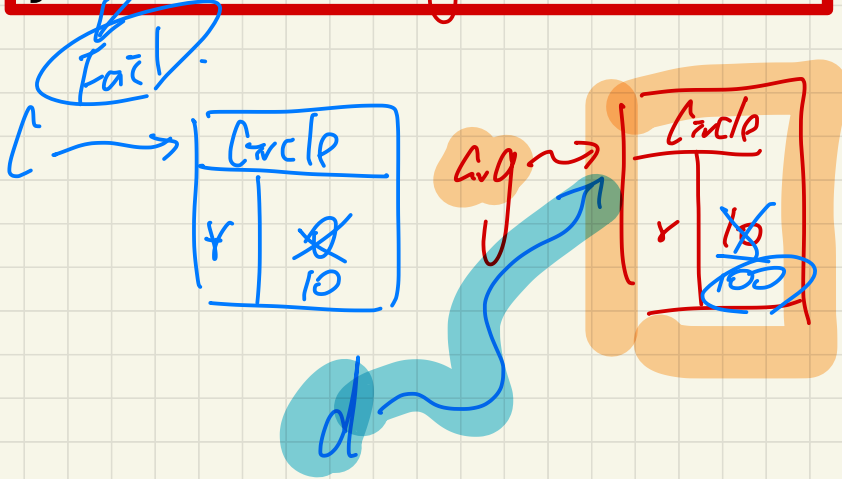
d.setRadius(100);

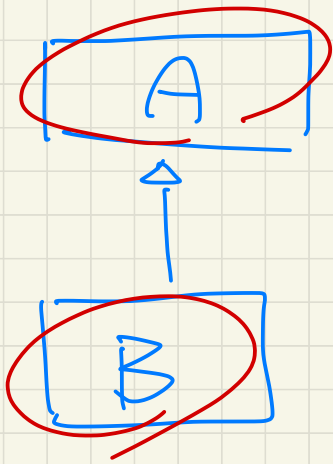
```

class CircleUser {
  ...
  Circle c = new Circle();
  Circle arg = new Circle(10);
  c.setRadius(arg);
}

```

assertTrue(arg.radius == 10);





① $(A) \quad oa = \dots;$

$(B) \quad ob = \dots;$

$oa = ob;$

$\times \quad ob = oa;$

not complete

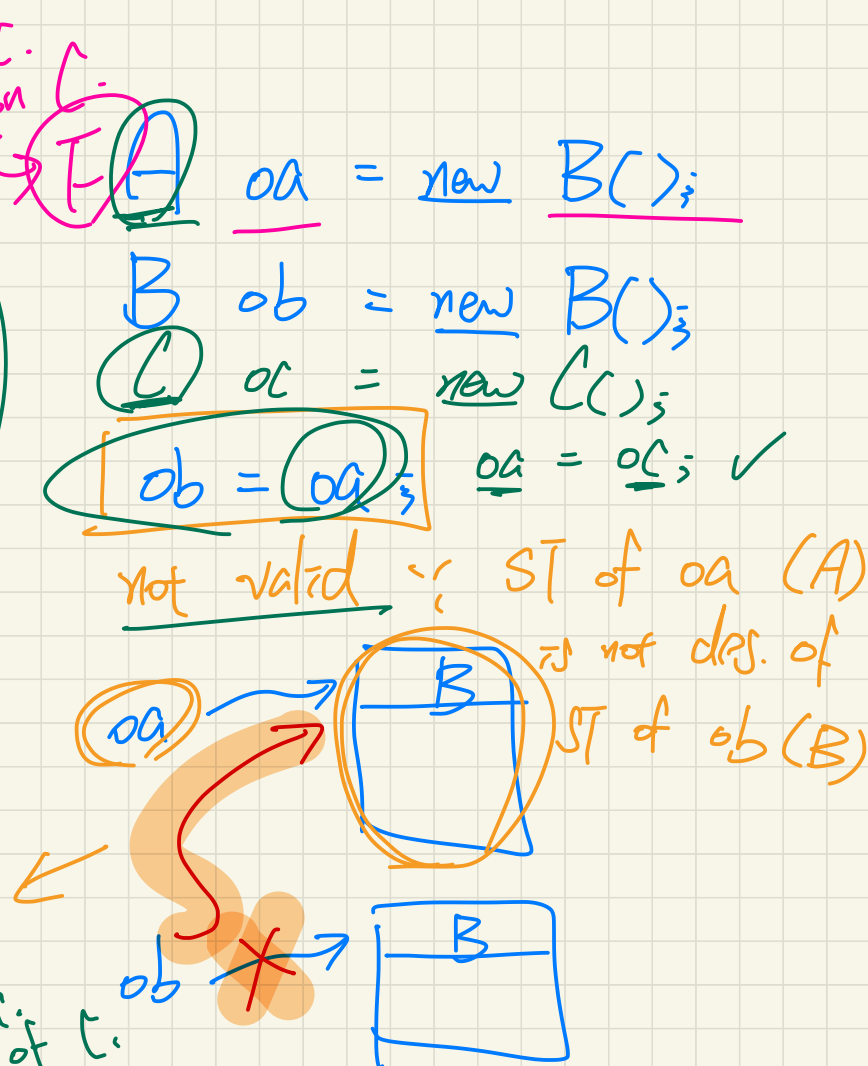
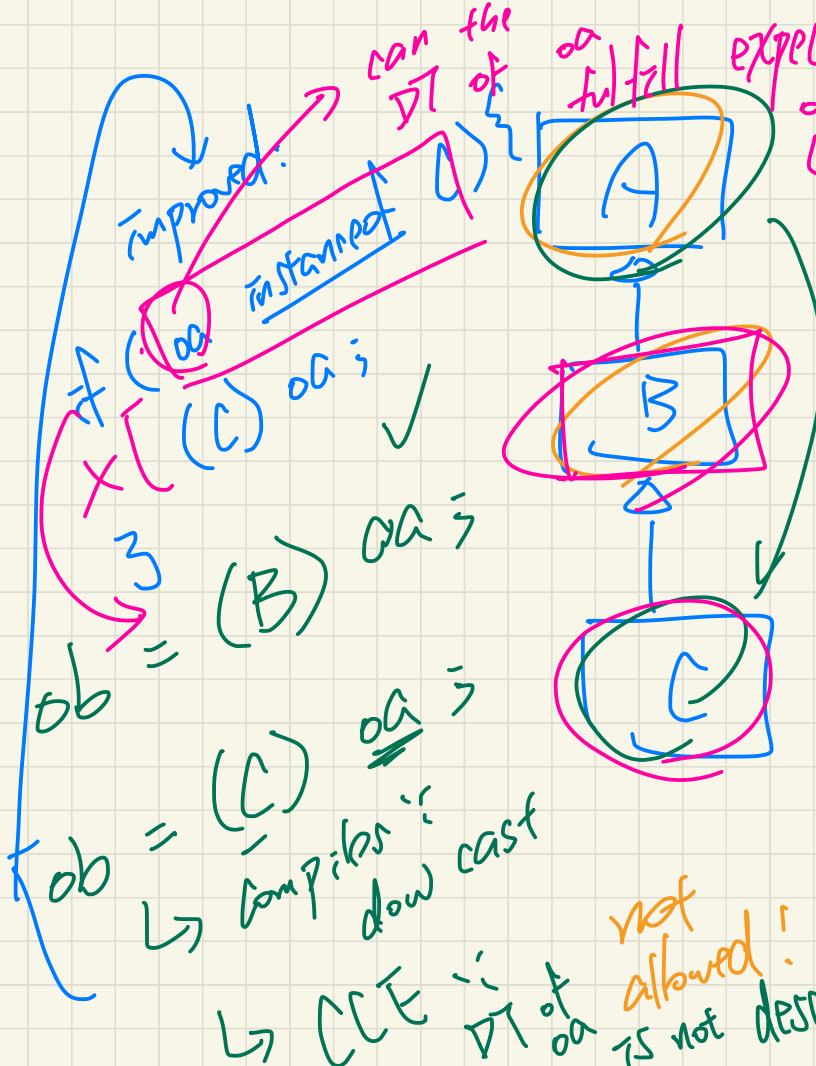
$\downarrow \quad \downarrow$
 $\underline{ST: B} \quad \underline{ST: A}$

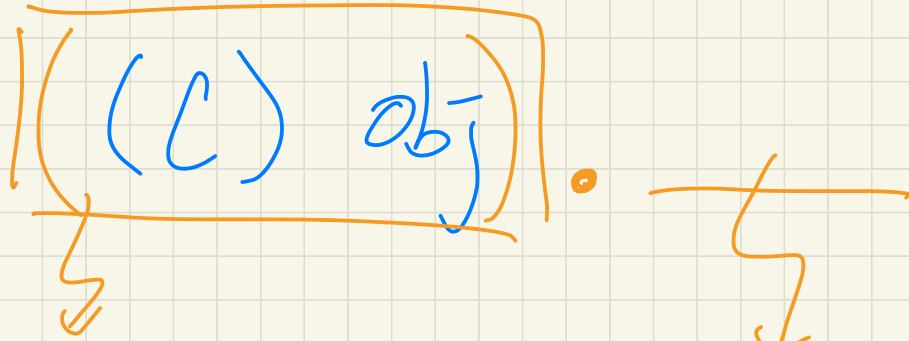
② ✓

$(A) \quad oa = \underline{\text{new } B();}$

$B \quad ob = \underline{\text{new } A();}$

new object which can fulfill exp. on B.





cast expression
has static type C

all expectations on C
all attributes and
methods declared
in ancestors of C .

Generic Parameters: ArrayList

```
class ArrayList<E> {  
    boolean add(E e)  
    E remove(int index)  
    E get(int index)  
}
```

→ declare a type parameter to be used in ArrayList class

list1.add(new Point(3, 4)); X

Caller of ArrayList

String s = list2.remove(2); X

```
ArrayList<String> list1 = new ArrayList<String>();  
ArrayList<Point> list2 = new ArrayList<Point>();
```

```
class ArrayList<X> {  
    boolean add(X e)  
    X remove(int index)  
    X get(int index)  
}
```

String
String
X
X

```
class ArrayList<X> {  
    boolean add(X e)  
    X remove(int index)  
    X get(int index)  
}
```

Point
Point
Point