Array Video 1(intro)

overview

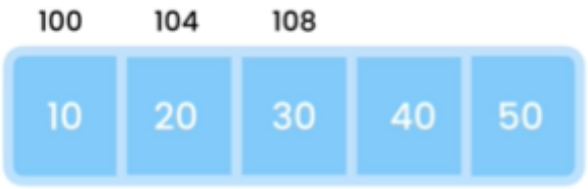
array in java

building an array

Video 2

Understanding Arrays

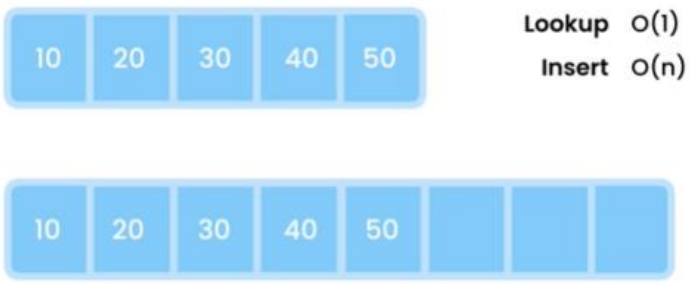
Array is the **simplest** data structure



In Array insertion we first need to resize the array of n elements. That's why it's time complexity is O(n).

Array in java store 4 digits that's why

100, 104, 108 etc.



Delete best case O(1) → remove from last index

Delete worst case O(n) → remove from last index

Video 3:- Working with Arrays in Java

```
package com.codewithmosh;

public class Main {
    public static void main(String[] args) {
        int[] numbers = new int[3];
        System.out.println(numbers);
    }
}
```

```
/Library/Java/JavaVirtualMachines/jdk-12.0.1.jdk/C
[I@38af3868

Process finished with exit code 0
```

Above picture has an Array type int

and I@.... Has two parts I means array type integer and @.... has tell the memory location where it stored.

//code

int [ ] num = new int [3];

Here we defined an array of integer with length 3

System.out.println(Arrays.toString(num));

// this line above print [0,0,0]

//so far we just declare the array but don't take any input in the array. That's why we get to see [0,0,0]

Note:

1. Array's index started with 0.
2. Negative index will give exception.

//Messy way to initialize items in array (time consuming)

int [ ] num = new int [3];

num [0] = 10;

num [1] = 20;

num [2] = 30;

System.out.println(Arrays.toString(num));

// this line above print [10, 20, 30]

//Cleanest method of making an array and initialized items in that array

int [ ] num = {10,20,30};

System.out.println(Arrays.toString(num));

// this line above print [10, 20, 30]

System.out.println(num.length);

//this will print 3

//if we wanted to add more items we have to create a new array with greater size and add the extra items on that resized array.

```
Main.java   Array.java
3   public class Main {
4       public static void main(String[] args) {
5           Array numbers = new Array( length: 3);
6           numbers.insert( item: 10);
7           numbers.insert( item: 20);
8           numbers.insert( item: 30);
9           numbers.insert( item: 40);
10          numbers.removeAt( index: 3);
11          System.out.println(numbers.indexOf(10));
```

Array items inserted via .insert method

Array size will updated automatically for .insert method.

removeAt(index); //this will remove the item from the given index.

Array two formation (Dynamic Purpose:Util Package):-

1. Vector-(100% growth every time its full)-Synchronized –Single thread Access
2. ArrayList (50% growth every time its full)

Some Useful built in Methods for indexes in Arraylist:-

```
indexOf(Object o)                                  int
lastIndexOf(Object o)                              int
contains(Object o)                             boolean
retainAll(Collection<?> c)                     boolean
containsAll(Collection<?> c)                   boolean
toString()                                      String
Press Ctrl+. to choose the selected (or first) suggestion and insert a dot afterwards  Next Tip
```

indexOf only find first occurrence of that value in the list.

size() methods returns how many items exists in the arraylist.

toArray() ➔ it makes list → Regular Array object.

Arrays

Exercises

1- Extend the Array class and add a new method to return the largest number. What is the runtime complexity of this method?

Solution: Array.max()

2- Extend the Array class and add a method to return the common items in this array and another array.

Solution: Array.intersect()

3- Extend the Array class and add a method to reverse the array. For example, if the array includes [1, 2, 3, 4], after reversing and printing it, we should see [4, 3, 2, 1].

Solution: Array.reverse()

4- Extend the Array class and add a new method to insert an item at a given index:

```
public void insertAt(int item, int index)
```

Solution: Array.insertAt()

```
package com.codewithmosh;


public class Array {

  private int[] items;

  private int count;


  public Array(int length) {

    items = new int[length];

  }


  public void insert(int item) {

    resizeIfRequired();


    items[count++] = item;
```

```
}

public void insertAt(int item, int index) {
  if (index < 0 || index > count)
    throw new IllegalArgumentException();

  // Note that I've extracted the logic for
  // resizing the array into this private
  // method so we can reuse in insert() and
  // insertAt() methods.
  //
  // This also made our code cleaner and
  // more readable.
  resizeIfRequired();

  for (int i = count - 1; i >= index; i--)
    items[i + 1] = items[i];

  items[index] = item;
  count++;
}

private void resizeIfRequired() {
  if (items.length == count) {
    int[] newItems = new int[count * 2];
```

```
    for (int i = 0; i < count; i++)
      newItems[i] = items[i];

    items = newItems;
  }
}

public void reverse() {
  int[] newItems = new int[count];

  for (int i = 0; i < count; i++)
    newItems[i] = items[count - i - 1];

  items = newItems;
}

public int max() {
  // O(n): Because we have to iterate over
  // the entire array to find the largest
  // number. This number may be at the end
  // of the array (worst case scenario).
  int max = 0;
  for (int item : items)
    if (item > max)
      max = item;
```

```
  return max;
}
```

//Very Hard intersect a[10,20,30],b[30,40,10] method will print {30,10}

```
public Array intersect(Array other) {
  var intersection = new Array(count);

  for (int item : items)
    if (other.indexOf(item) >= 0)
      intersection.insert(item);

  return intersection;
}

public void removeAt(int index) {
  if (index < 0 || index >= count)
    throw new IllegalArgumentException();

  for (int i = index; i < count; i++)
    items[i] = items[i + 1];

  count--;
}

public int indexOf(int item) {
  for (int i = 0; i < count; i++)
```

```
    if (items[i] == item)
      return i;

  return -1;
 }

 public void print() {
  for (int i = 0; i < count; i++)
    System.out.println(items[i]);
 }
}
```

Exercise build an array class just do this fundamental tasks.(This is covered above)

1. Insert items dynamically
2. Delete items from specific indexes
3. If user give me index of any item I will search for the item in the given array and return found or not found.