

arrays

- literal array = `int [] a = {3, 6, 10, 14};`
- often `array[i+1]` goes out of bounds
- `int array[] = new int array[3]`
 - ↳ all values equal to 0
- 2D array → ROW MAJOR
- `int [][] arr2D = new int [size][size]`
 - ↳ to find a row, dow `arr2D [row]`
 - ↳ to find column, use helper method

for loop

- check starting value (`x=0` or `x=1`)
- ★ how many times does a certain line run?
 - ↳ test even/odd numbers
- ★ `for(i=0; i < 3; i++)`
 - ↳ `i` will never be to 3
- can't modify arrays/lists in for each loops
- `for(i=0; i <= 3; i++)`
 - ↳ `i` can be 3

String Class

- `str.substring [start]`
 - if `start > str.length()`
 - ↳ out of bounds
- `str.substring [start, end]`
 - if `start > n`
 - or `end > str.length()`
 - ↳ out of bounds
- ★ `substring(0)` → prints whole word
- ★ `substring(length)` → prints nothing
- ★ if `end = str.length()`
 - ↳ then ok ✓
- +toString
 - ↳ runs automatically when call an object
 - ↳ if not overridden, prints memory location
- compareTo
 - ↳ alphabetical order
 - "a".compareTo("b") < 0
 - if a precedes → < 0 (goes first)
 - if same = 0

miscellaneous

- don't forget to initialize
- cannot cast an obj
- which action is best?
 - ↳ the specification must be clarified
- `(double)(3/4) = 0`
- To call `reverseArray` in `ArrayUtil` class
 - ★ `ArrayUtil.reverseArray (array)`
- DONT forget to boundry check!
- `==` → same object (for obj.)
- `(x==y) || (x>2) && (y=2) → true`
 - if true, it short circuits

the test

- don't be redundant
- when finding a post condition
 - ↳ it must describe the MAIN function
- always the simplest answer

ArrayLists

- `list.add (list.size())`
 - ↳ index out of bounds
- ★ MUST initialize using "new"
 - ↳ can only hold objects (no primitives)

Math class

- ★ `Math.random()`
 - ↳ returns `0 ≤ Math.random < 1`
 - ↳ `(int)(Math.random()*range) + offset`
 - ↳ 0-100 (inclusive)
 - ↳ `(int)(Math.random()*101)`

reminders

- if "The code throws a _____ error" is option
 - ↳ look for out of bounds errors
 - ↳ look for division by zero → arithmetic exception

errors

- **Syntax error**
 - * compiler cannot understand the code
- **Compile error**
 - * occurs while compiling
- **runtime error**
 - * incorrect results
 - * logic of code is incorrect
- **exceptions**
 - * during execution (runtime)
 - * disrupts normal flow
 - * will cause program to stop execution

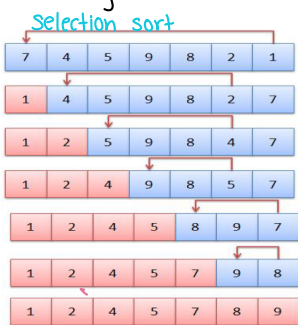
ex.

- arithmetic exception (\div by 0)
- illegal argument exception integer \rightarrow string val
- index out of bounds
- array index out of bounds
- null pointer `s = null r = "";`
`s.equals(r)` \rightarrow not initialized
- concurrent modification
- stack overflow \rightarrow computer memory is exhausted

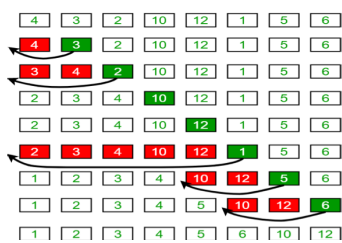
Classes

- **Static variables** shared by all instances of the class (objects)
 - \hookrightarrow accumulate total \star one for whole class
 - \hookrightarrow keep track of stats for objects
- instance methods
 - **accessor methods** - access obj but do not modify
 - **mutator method** - modifies obj.
- **Static method** - CANNOT use instance variables, only static ones
- **Overriding** \rightarrow 2 methods in ONE class with same name and different parameters
- **Overloading** \rightarrow 2 methods with same name/parameters, one in parent class, one in child class
- **final** \rightarrow the value will not change (all caps)
- **this** - `System.out.print(this)` \rightarrow prints the toString \hookrightarrow the current object
- **Alarsing** `Date d = new Date(1, 2, 3);`
`Date birthday = d;`
`d.changeDate()` \rightarrow also changes birthday
 to avoid prob `Date birthday = new Date(d.getMonth(), d.getDay(), d.getYear());`
 * in static methods must call an object (not automatic)

Sorting/search



insertion sort



\star fastest when short sorted list

\hookrightarrow go through and update curmin

\hookrightarrow go through entire array each time

\star search and swap

Sequential/linear search

* works on unsorted too

binary search

- * data must be sorted
- * starts in middle
 - \hookrightarrow eliminate half of the array each iteration
 - \hookrightarrow keeps going until desired value is found or all elements eliminated
- * more efficient than sequential/linear search
- * can be recursive/iteratively

\hookrightarrow what is the max amount of times to determine x is not in the array

first value of y where $2^y >$ array.length

data types

- doubles sometimes have rounding issues
 - \hookrightarrow floating-point numbers
- Primitives (int, double ...)
- Reference (String, Random, int[], String[][], Cat)

Recursion

• public static void recursion(int num)

num = 4

call stack

\hookrightarrow recursion(4)

\hookrightarrow recursion(3)

\hookrightarrow recursion(2)

⋮

\star check if the method prints or returns

num value

num = 4

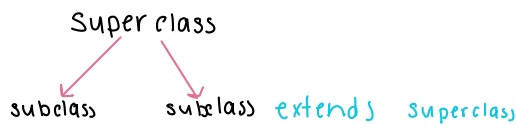
num = 3

num = 2

\star when you get to base case... work backwards from last call to the line after the recursive call

• inheritance

- **super** must be first line of constructor
 - ↳ if not, compile time error
- if a client program of SomeClass is trying to access private methods/variables
 - ↳ will cause error
- a subclass class inherits public vars/stats/methods ✱
 - ↳ or private ones only via public methods
- a superclass **CANNOT** access properties from child class
- subclass can **override** methods
- the subclass can invoke constructors from superclass via **super**
- if the subclass does not call parent constructor, default constructor is called.
- Composition is a has-a relationship
- inheritance is a is-a relationship
- Class obj = new subclass();
 - obj. method in subclass()
 - ↳ will not work b/c obj is a class type
- can declare a subclass as a class
- Constructors are not inherited!
 - ↳ Super class must have default
 - ↳ gets called if no constructor is subclass
 - ↳ other instance variables are set to 0 or null
- **super** → first line of constructor
- cannot override static methods
- **polymorphic** → a method overridden at least once
 - Student a = new grad(3)
 - object reference type object type
- if a method is NOT overridden, it must be in the Object Reference Type class (superclass)
- can downcast it to a class with that method



public void method (class name)

↓
anything that is class
OR subclass of class can work

fruit
↓
grape
↓
seedless grape

✓ fruit a = new seedless grape(...)
✗ seedless b = new grape(...)

```

public Subclass()
{
    super();
}
    
```

modulus

- 2 % 3
 - ↳ remainder is 2
- 5 % 10
 - ↳ remainder is 5
- % → same precedence as * and /