COMP20010

Data Structures and Algorithms I

01 Introduction

Dr. Aonghus Lawlor
aonghus.lawlor@ucd.ie

Formalities

Timetables

Lecture	Tuesday	09:00	110mins	FS G01 (AG)
Lecture	Thursday	10:00	110mins	G-15 (AG)
Lab	Monday	14:00	110mins	B002/B003
Lab	Friday	11:00	110mins	B002/B003

Timetables

Lecture	Tuesday	09:00	110mins	FS G01 (AG)
Lecture	Thursday	10:00	110mins	G-15 (AG)
Lab	Monday	14:00	110mins	B002/B003
Lab	Friday	11:00	110mins	B002/B003

Moodle Enrolment Key:
COMP200102018

Assessment

- Continuous Assessment 30%
  - 2 Assignments (2x12%=24%)
  - End of module code repository (6%)

- End of Semester Exam: 70%

How to Learn

- Lectures
- Tutorials
- Lab sessions
- Homework and coding experiments
- Assignments
- Online

Plagiarism & UCD Computer Science

- **Plagiarism is a serious academic offence**
  - [Student Code, sections 6.2 & 6.3] or [UCD Registry Plagiarism Policy] or [CS Plagiarism policy and procedures]
- Our staff and demonstrators are **proactive** in looking for possible plagiarism in all submitted work
- Suspected plagiarism is reported to the CS Plagiarism subcommittee for investigation
  - Usually includes an interview with student(s) involved
  - 1st offence: **usually** 0 or NG in the affected components
  - 2nd offence: may be referred to the **University disciplinary committee**
- Student who enables plagiarism is equally responsible

http://www.ucd.ie/registry/academicsecretariat/docs/plagiarism_po.pdf

http://www.ucd.ie/registry/academicsecretariat/docs/student_code.pdf

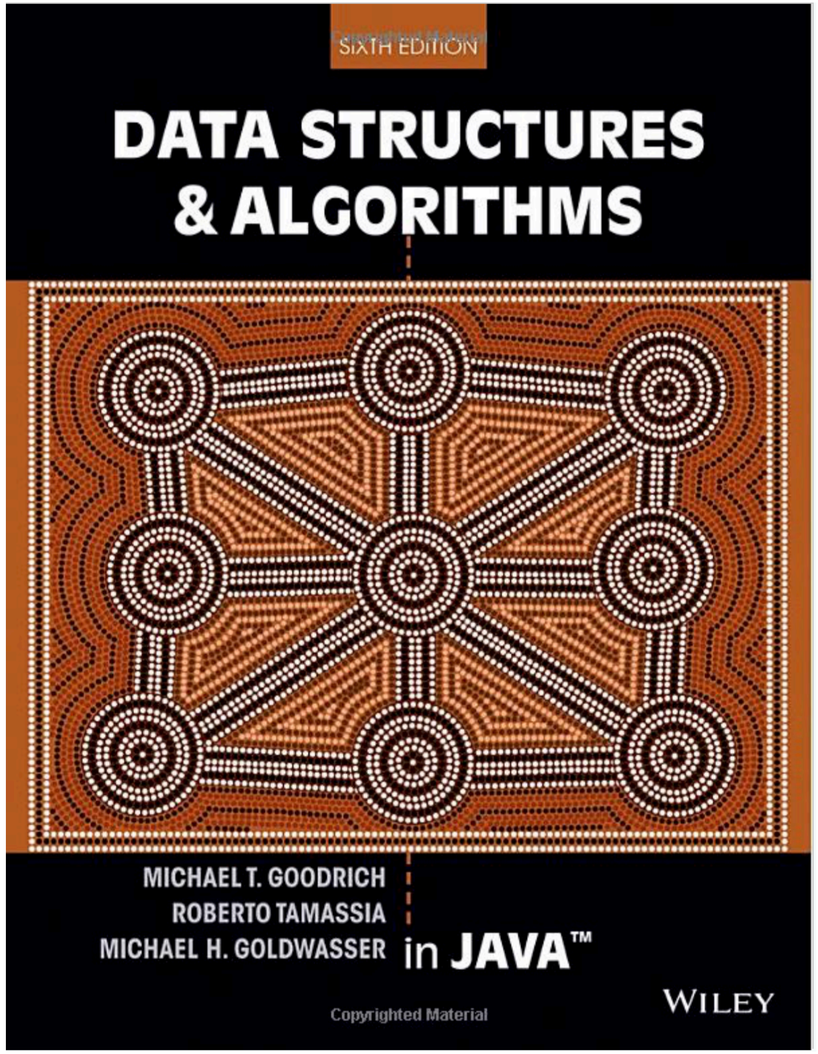
http://libguides.ucd.ie/academicintegrity

Course Timetable COMP20010

Week	Topic	Assignment
7	Intro	
	Arrays, ADT's, Generics	
8	Linked Lists	
	Doubly Linked, Circularly Linked	A1
9	Analysis of Algorithms	
10	Recursion	
	Timing, Performance	A2
11	Stacks, Queues, Deques, Priority Queues	
12	Maps, hash functions	A3
	Review	

Books

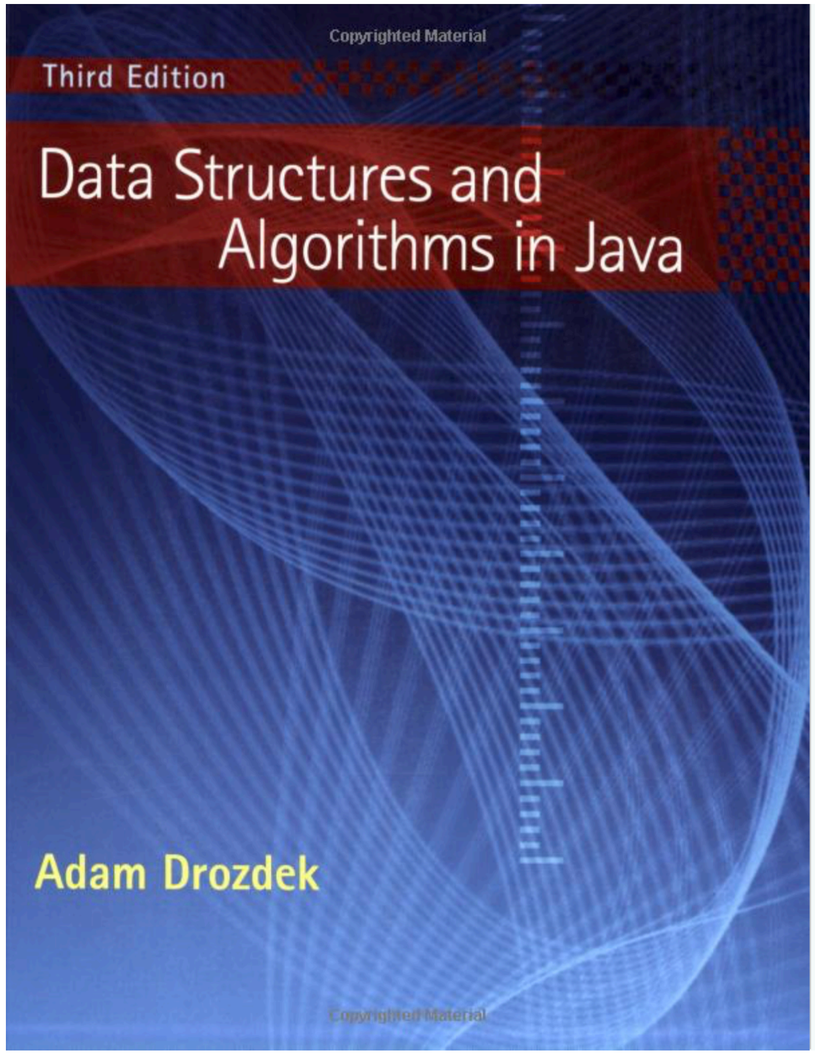
Books



Data Structures and Algorithms in Java

by Michael T. Goodrich (Author), Roberto Tamassia (Author), David M. Mount (Author)

Books



Data Structures and Algorithms in Java

by Adam Drozdek (Author)

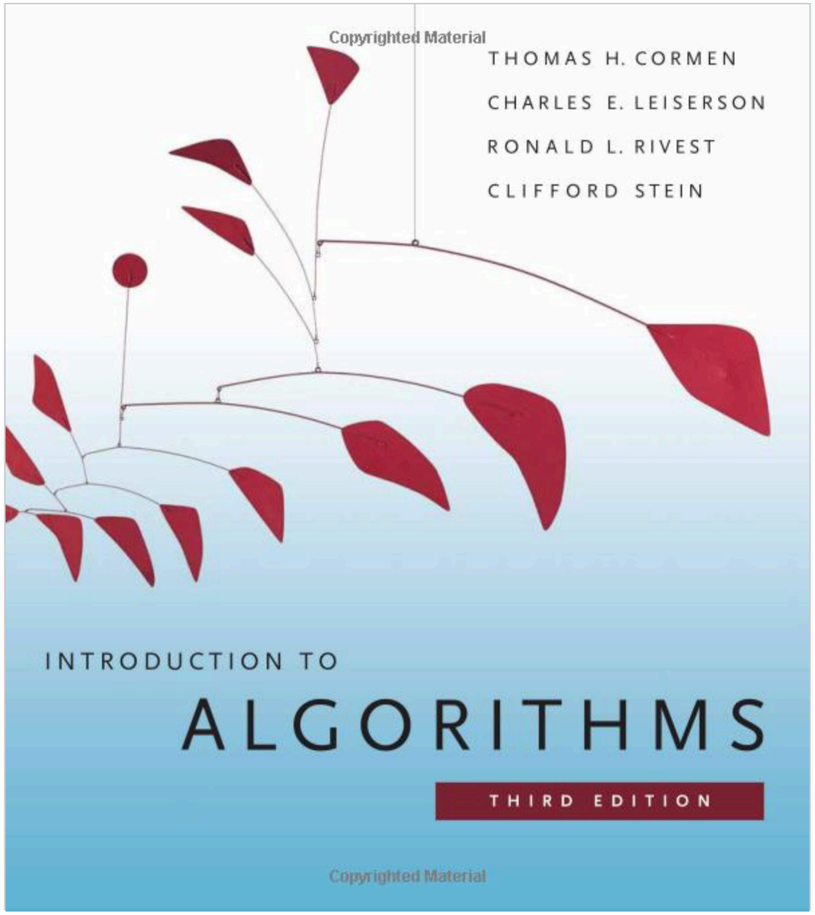
Books

Algorithms

by

Robert Sedgewick (Author), Kevin Wayne (Author)

Books



Introduction to Algorithms

by

Thomas H. Cormen (Author), Charles E. Leiserson (Author), Ronald L. Rivest (Author), Clifford Stein (Author)

Overview

Why data structures and algorithms?

- Data Structures
- Algorithms
- Efficient Programming
- Designing, building, testing programs
- Programming tools (java, eclipse)
- Languages (Java, Scala, etc...)

Why data structures and algorithms?

Their impact is broad and far-reaching

- **Internet:** Web search, packet routing, distributed file sharing.
- **Biology:** Human genome project, protein folding.
- **Computers:** Circuit layout, file system, compilers.
- **Computer graphics:** Movies, video games, virtual reality.
- **Security:** Cell phones, e-commerce, voting machines.
- **Multimedia:** CD player, DVD, MP3, JPG, DivX, HDTV.
- **Transportation:** Airline crew scheduling, map routing.
- **Physics:** N-body simulation, particle collision simulation.

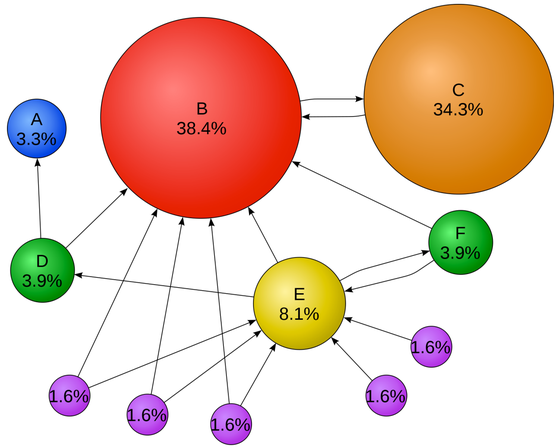
Profit!

Google

- PageRank algorithm developed by Larry Page and Sergey Brin at Stanford in 1996
- Tells us the probability that any person randomly clicking on links will arrive at a page
- Several iterative strategies are possible (linear algebra)
- Google Inc. is now a $131bn company with 58,000 employees

$$PR(u) = \sum_{v \in B_u} \frac{PR(v)}{L(v)}$$

the PageRank value for a page u is dependent on the PageRank values for each page v contained in the set Bu (the set containing all pages linking to page u), divided by the number L(v) of links from page v.



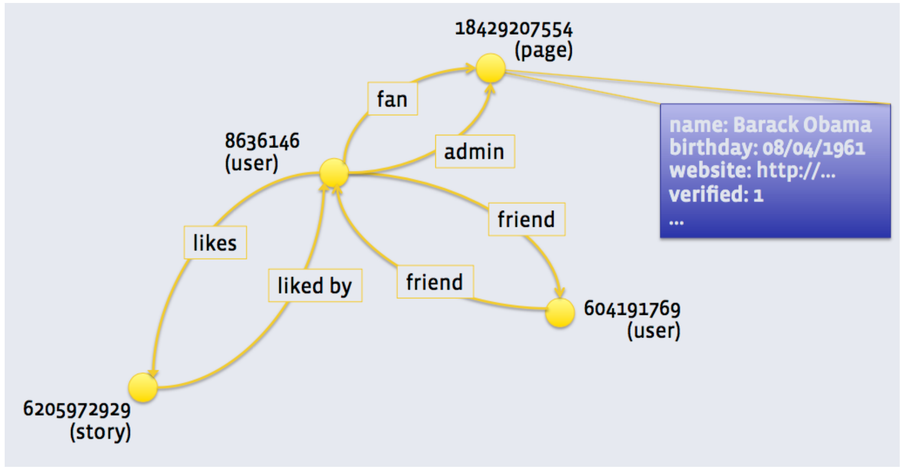
Google

- Store data persistently and efficiently
  - high availability
  - high read/write bandwidth

- Run large scale computations
  - don’t lose computations because of machine failure
  - distributed computations for speed

Facebook

- Facebook does graph processing on a massive scale
- Unicorn, Pregel graph engines
- 2.5billion pieces of new content every day
- 2.7billion likes added every day (8Tb/day)
- Unicorn is an online, in-memory social graph-



The Facebook graph is the collection of entities and their relationships on Facebook. The entities are the nodes and the relationships are the edges. One way to think of this is if the graph were represented by language, the nodes would be the nouns and the edges would be the verbs. Every user, page, place, photo, post, etc. are nodes in this graph. Edges between nodes represent friendships, check-ins, tags, relationships, ownership, attributes, etc.

Why data structures and algorithms?

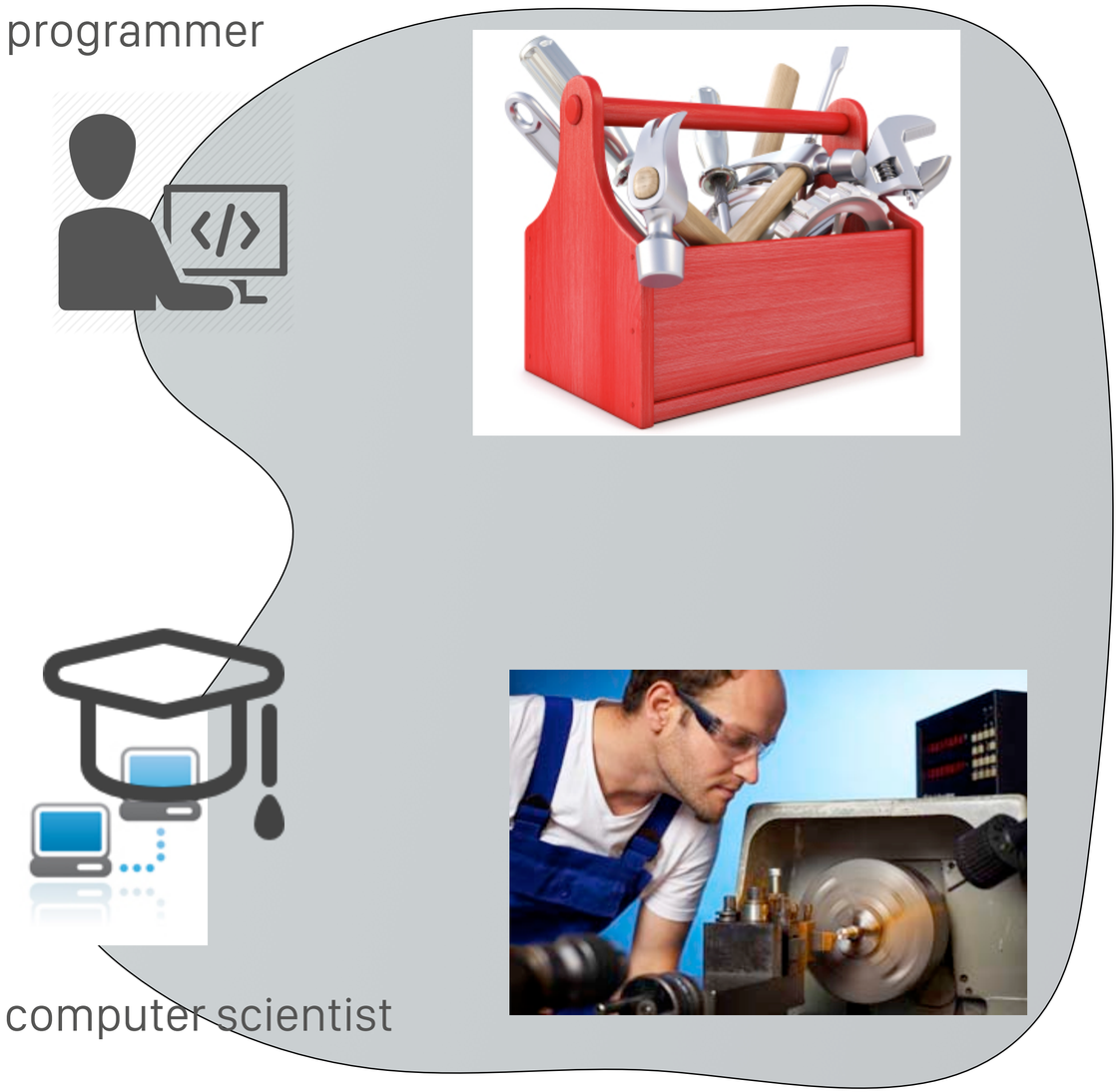
- data structures are (mostly) fundamentally simple
- obvious implementations do not scale well for number of items/ number of operations
- we need to care with how the data is organised
- we need to pay special attention to how it is accessed and manipulated

Why data structures and algorithms?

- Their impact is broad and far-reaching
- Old roots, new opportunities
- To be able to solve problems that could not otherwise be addressed
- For intellectual stimulation
- They may unlock the secrets of life and of the universe

Why bother?

- essential knowledge for all levels of programming
- active fields of research in computer science (graphs, spatial ds, distributed data, clustering, stream processing, high-dimensional data)



Why data structures and algorithms?

Why data structures and algorithms?

"...I will, in fact, claim that the difference between a bad programmer and a good one is whether he considers his code or his data structures more important. Bad programmers worry about the code. Good programmers worry about data structures and their relationships."

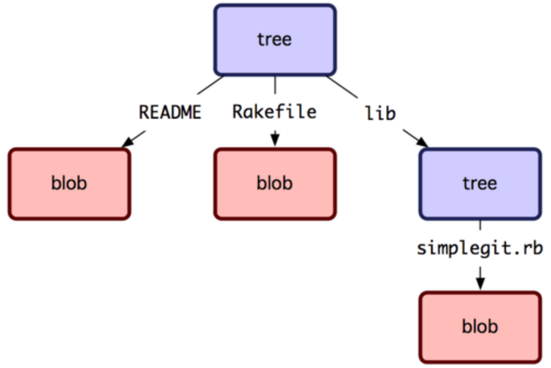
Linus Torvalds (2006-06-27). Message to Git mailing list.

Why data structures and algorithms?

"...I will, in fact, claim that the difference between a bad programmer and a good one is whether he considers his code or his data structures more important. Bad programmers worry about the code. Good programmers worry about data structures and their relationships."

Linus Torvalds (2006-06-27). Message to Git mailing list.

"Git is a content-addressable filesystem. It means that at the core of Git is a simple key-value data store. You can insert any kind of content into it, and it will give you back a key that you can use to retrieve the content again at any time."



“Interview Questions”

“Interview Questions”

Write a Java implemention of BubbleSort to sort an array of integers.

“Interview Questions”

Write a Java implemention of BubbleSort to sort an array of integers.

What is the difference between a Stack and Queue?

“Interview Questions”

Write a Java implemention of BubbleSort to sort an array of integers.

What is the difference between a Stack and Queue?

Describe an algorithm to find the largest sub-array of consecutive integers in an array?

“Interview Questions”

Write a Java implemention of BubbleSort to sort an array of integers.

What is the difference between a Stack and Queue?

Describe an algorithm to find the largest sub-array of consecutive integers in an array?

Write some Java code to implement the Stack ADT?

“Interview Questions”

Write a Java implemention of BubbleSort to sort an array of integers.

What is the difference between a Stack and Queue?

Describe an algorithm to find the largest sub-array of consecutive integers in an array?

Write some Java code to implement the Stack ADT?

What is a linked list?

“Interview Questions”

Write a Java implemention of BubbleSort to sort an array of integers.

Describe an algorithm to find the largest sub-array of consecutive integers in an array?

What is a linked list?

What is the difference between a Stack and Queue?

Write some Java code to implement the Stack ADT?

Describe an algorithm to determine if a linked list contains a loop?

“Interview Questions”

Write a Java implemention of BubbleSort to sort an array of integers.

What is the difference between a Stack and Queue?

Describe an algorithm to find the largest sub-array of consecutive integers in an array?

Write some Java code to implement the Stack ADT?

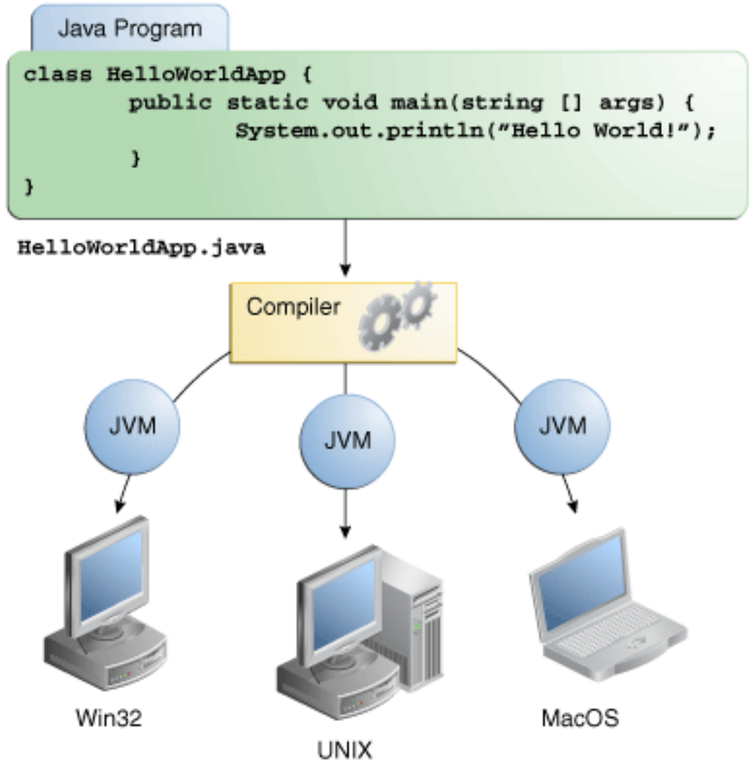
What is a linked list?

Describe a recursive algorithm to reverse a linked list?

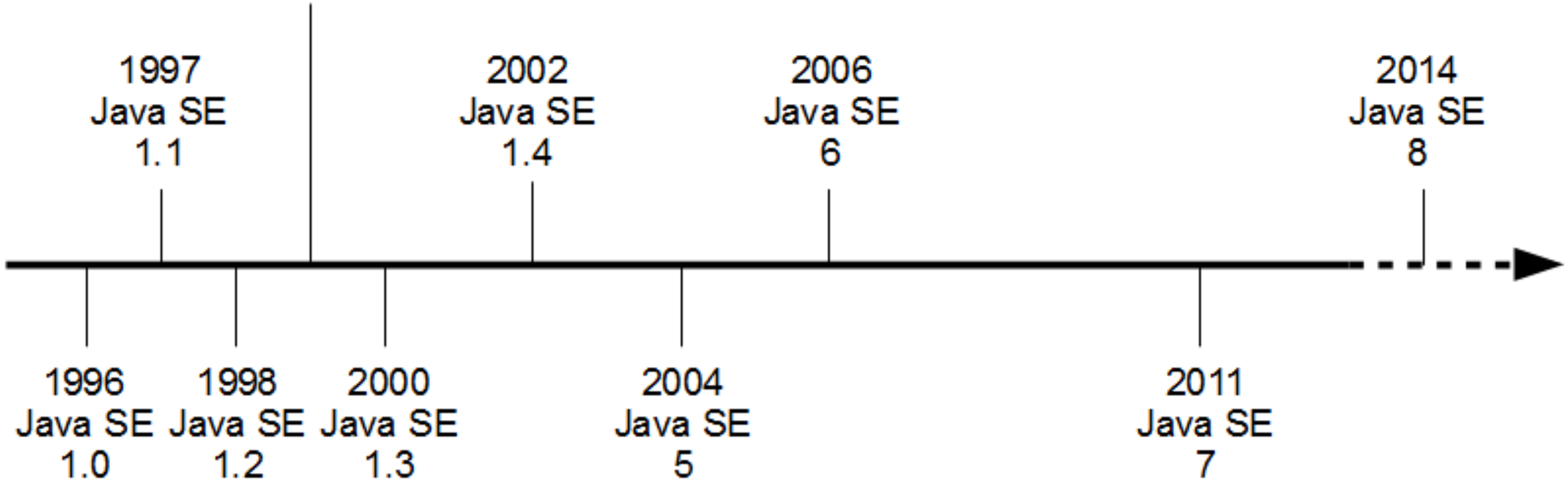
Describe an algorithm to determine if a linked list contains a loop?

CS Background

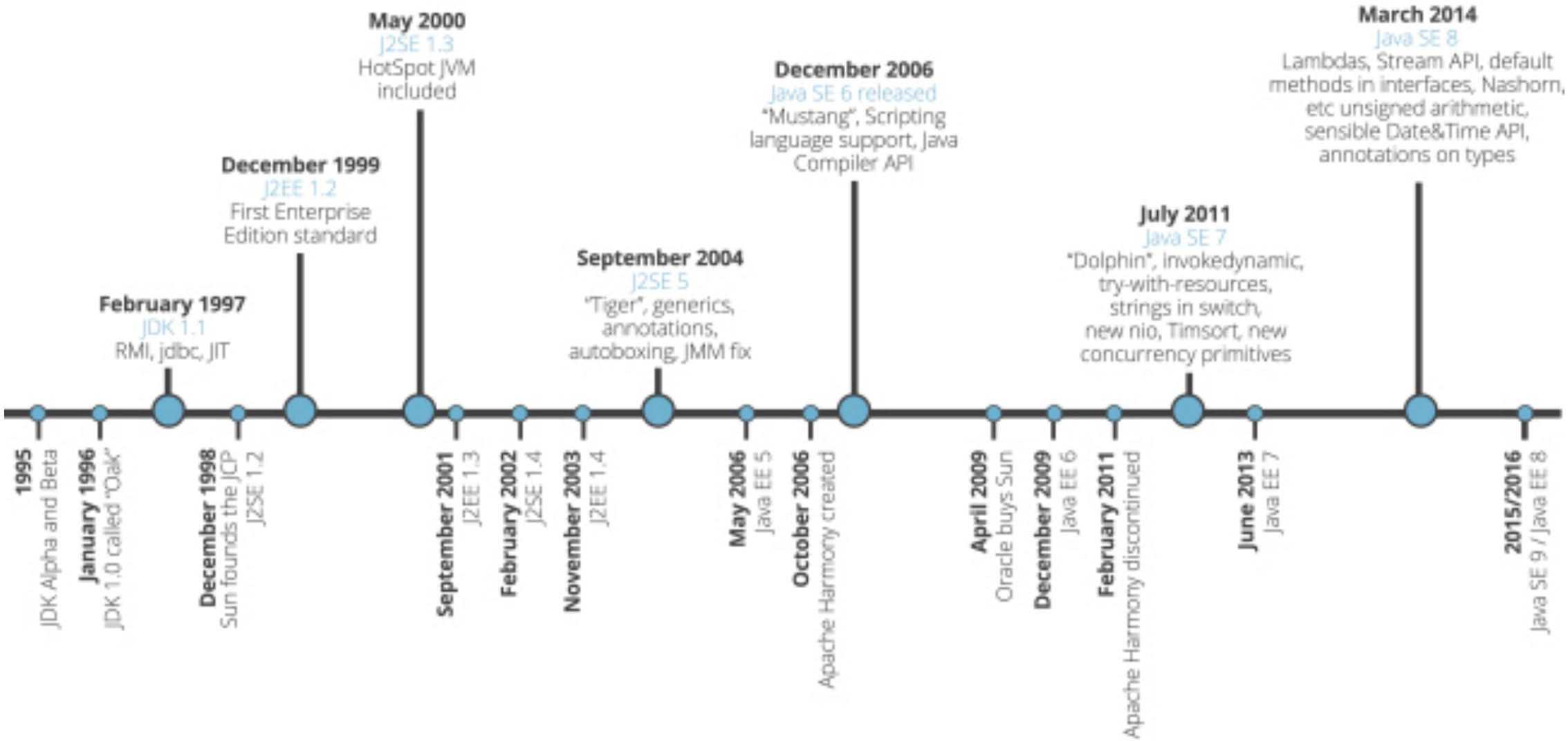
- In Java, all source code is first written in plain text files ending with the `.java` extension.
- Those source files are then compiled into `.class` files by the `javac` compiler.
- A `.class` file does not contain code that is native to your processor; it instead contains *bytecodes* — the machine language of the Java Virtual Machine (Java VM).
- The `java` launcher tool then runs your application with an instance of the Java Virtual Machine.



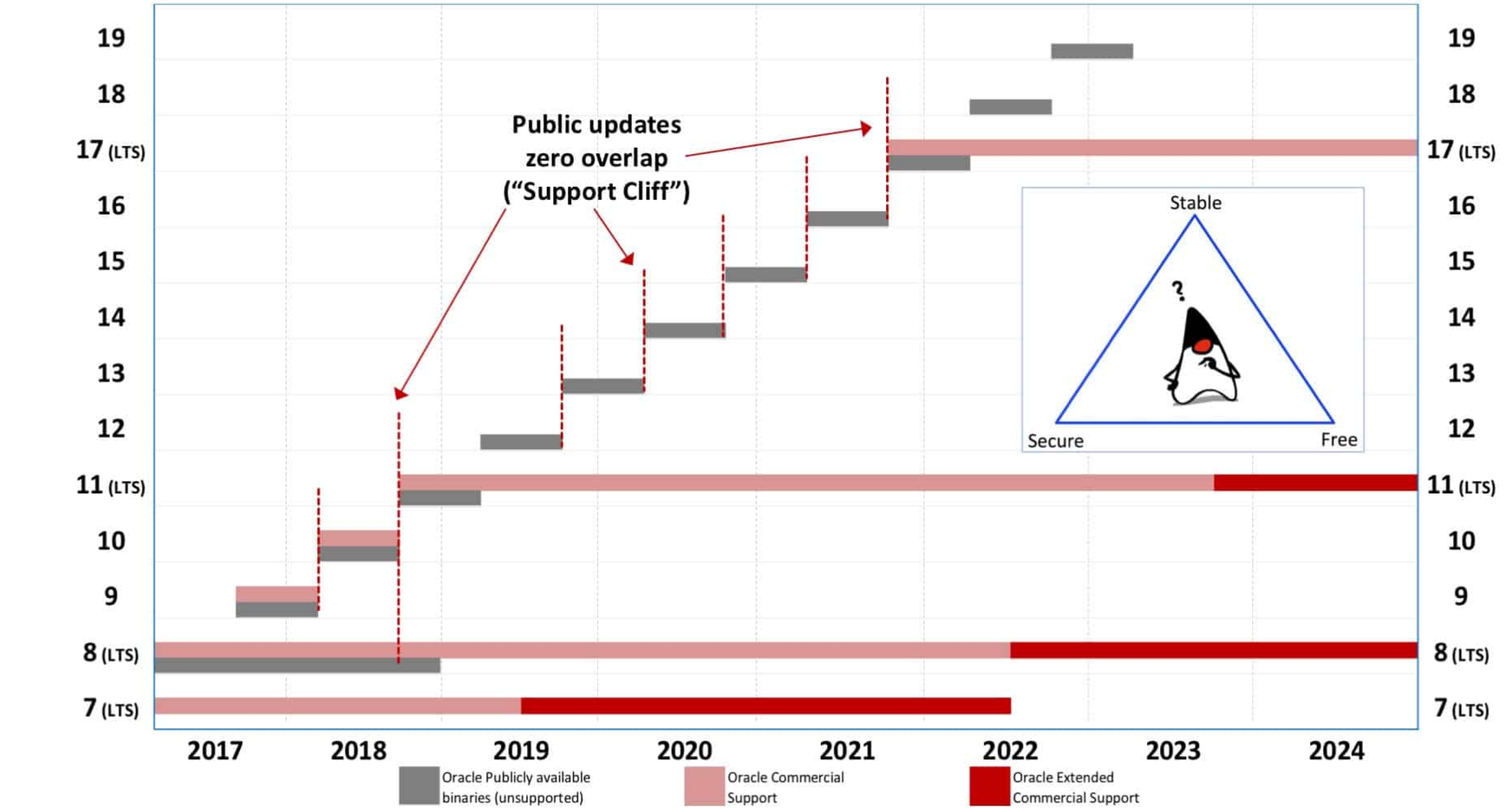
Java Timeline



Java Timeline

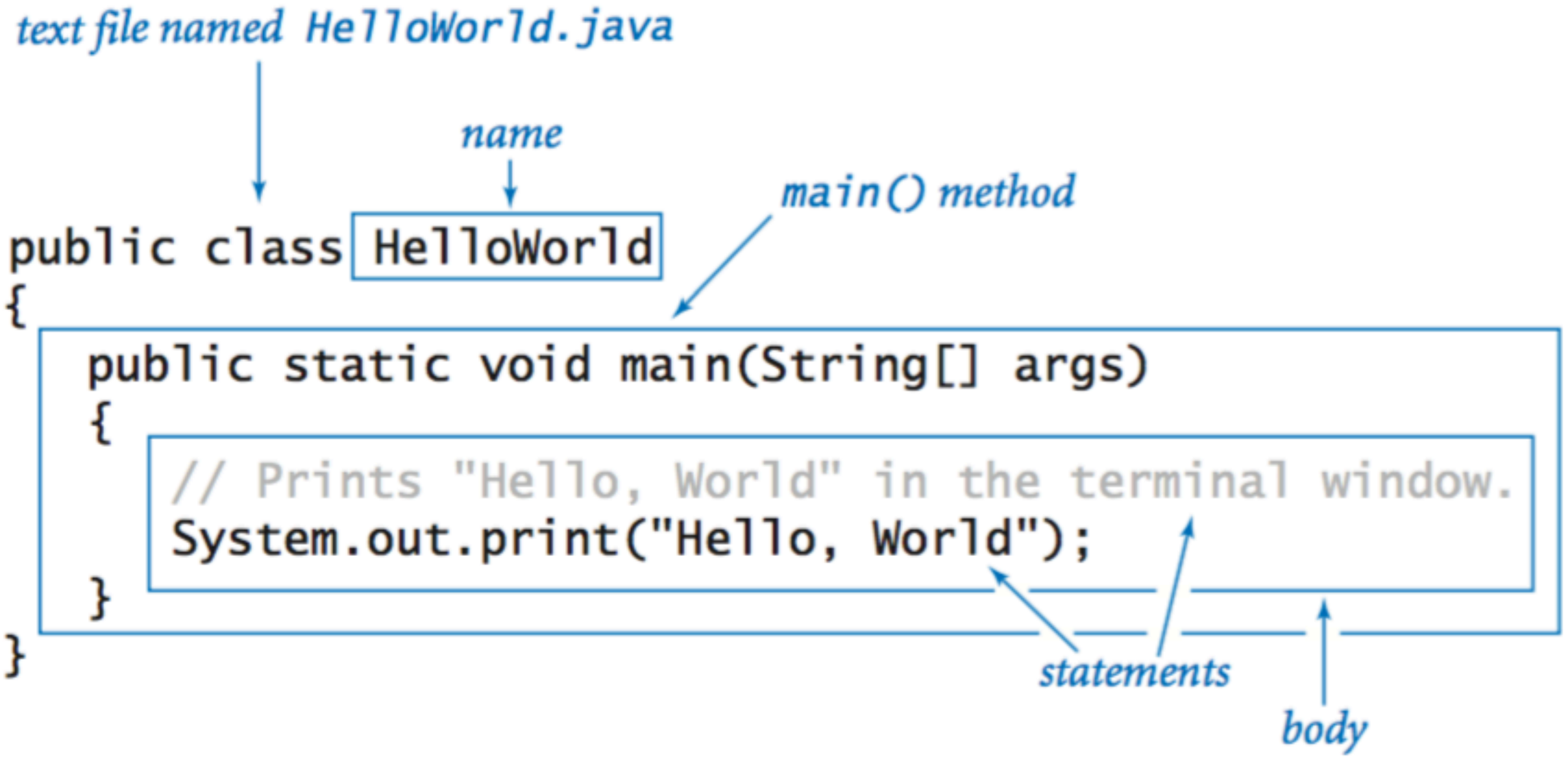


Java SE Lifecycle – 5+ Year Timeline

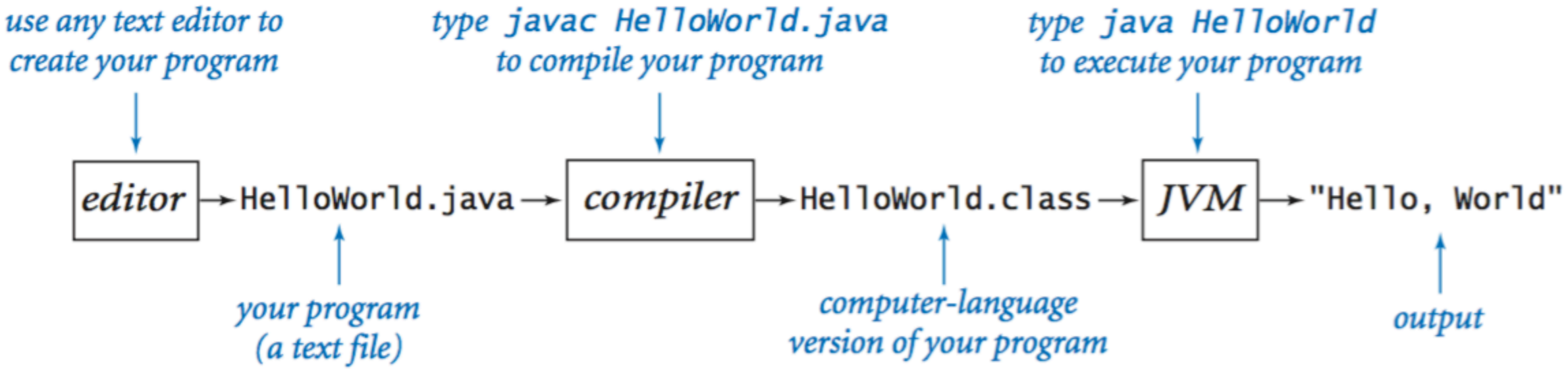


Java Recap

Hello World


text file named HelloWorld.java
name
main() method
public class HelloWorld
{
public static void main(String[] args)
{
// Prints "Hello, World" in the terminal window.
System.out.print("Hello, World");
}
}
statements
body

Editing, compiling, and executing



Built in data types

type	*set of values*	*common operators*	*sample literal values*
`int`	integers	`+ - * / %`	`99 12 2147483647`
`double`	floating-point numbers	`+ - * /`	`3.14 2.5 6.022e23`
`boolean`	boolean values	`&& || !`	`true false`
`char`	characters		`'A' '1' '%' '\n'`
`String`	sequences of characters	`+`	`"AB" "Hello" "2.5"`

Fundamental Data Types

Primitive Type	What It Stores	Range
`byte`	8-bit integer	–128 to 127
`short`	16-bit integer	–32,768 to 32,767
`int`	32-bit integer	–2,147,483,648 to 2,147,483,647
`long`	64-bit integer	-2^{63} to $2^{63} - 1$
`float`	32-bit floating-point	6 significant digits (10^{-46}, 10^{38})
`double`	64-bit floating-point	15 significant digits (10^{-324}, 10^{308})
`char`	Unicode character	
`boolean`	Boolean variable	`false` and `true`

The eight primitive types in Java

Fundamental Types

Integer Types

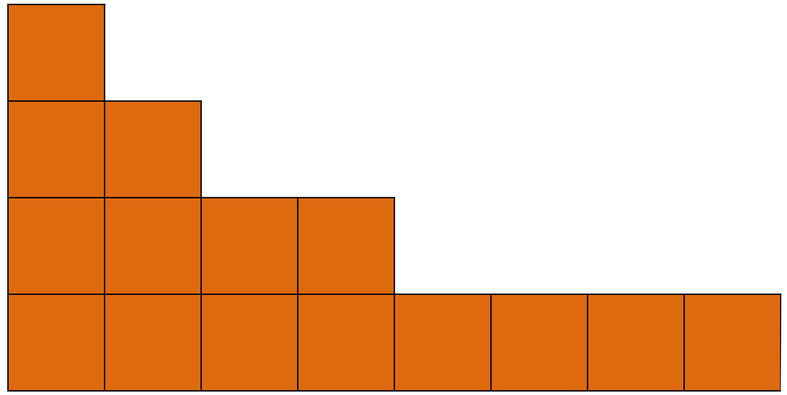
Size in bytes

byte

short

int

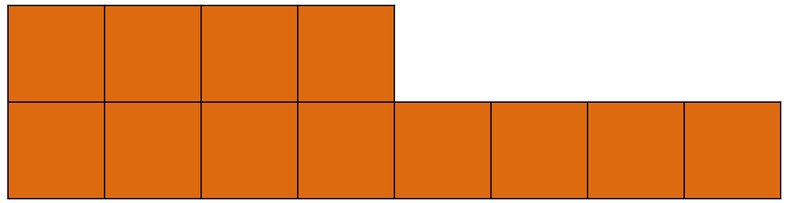
long



Floating Point

float

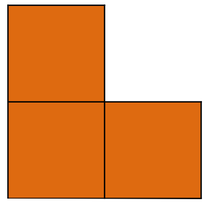
double



Others

boolean

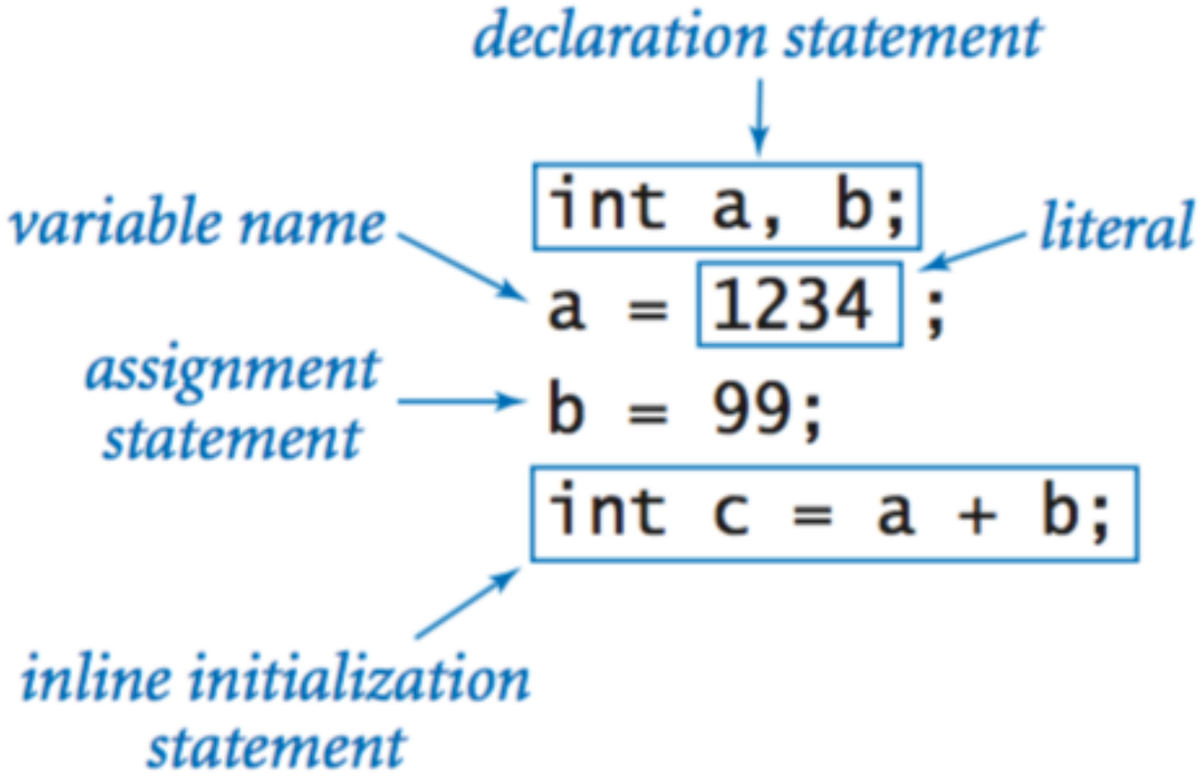
char



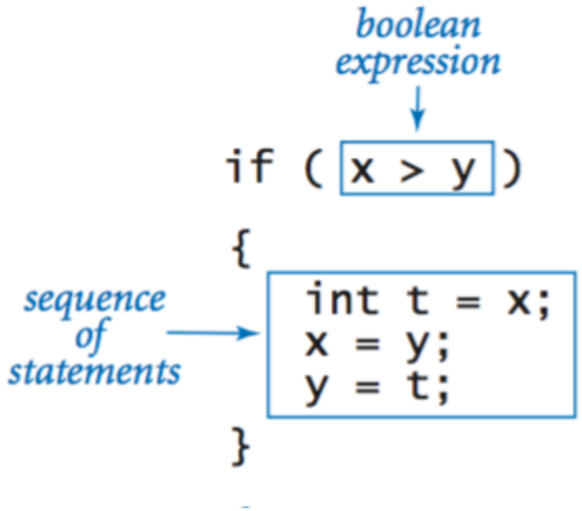
Fundamental Types

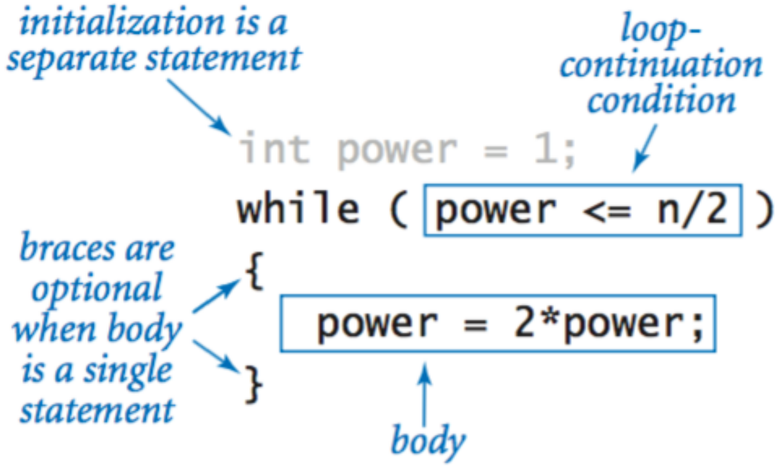
Integer Types		min	max
byte	2^8	-128	+127
short	2^16	-32768	32767
int	2^32	-2147483648	+21474836487
long	2^64	-9223372036854775808	9223372036854775808
Floating Point			
float		±3.40282347E+38	
double		±1.7976931348623157OE+308	
Others			
boolean		true or false	
char	2^16	0	65536

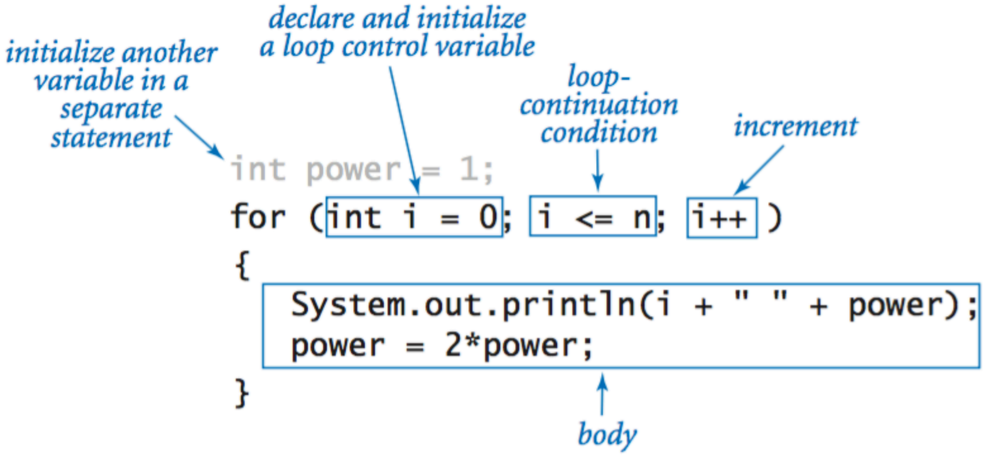
Declaration and assignment statements



Loops

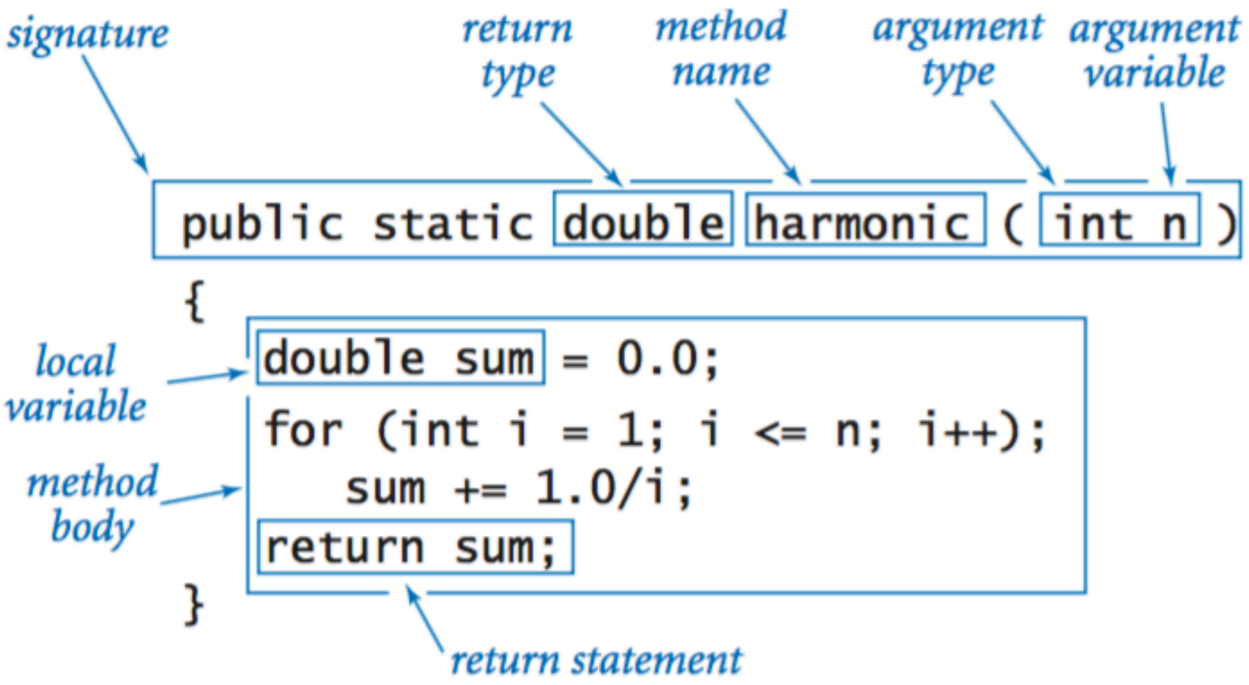







```
do
{  // Scale x and y to be random in (-1, 1).
   x = 2.0*Math.random() - 1.0;
   y = 2.0*Math.random() - 1.0;
} while (Math.sqrt(x*x + y*y) > 1.0);
```

Functions



absolute value of an `int` value	`public static int abs(int x)`
	`{`
	`  if (x < 0) return -x;`
	`  else  return  x;`
	`}`

absolute value of a `double` value	`public static double abs(double x)`
	`{`
	`  if (x < 0.0) return -x;`
	`  else  return  x;`
	`}`

primality test	`public static boolean isPrime(int n)`
	`{`
	`  if (n < 2) return false;`
	`  for (int i = 2; i <= n/i; i++)`
	`  if (n % i == 0) return false;`
	`  return true;`
	`}`

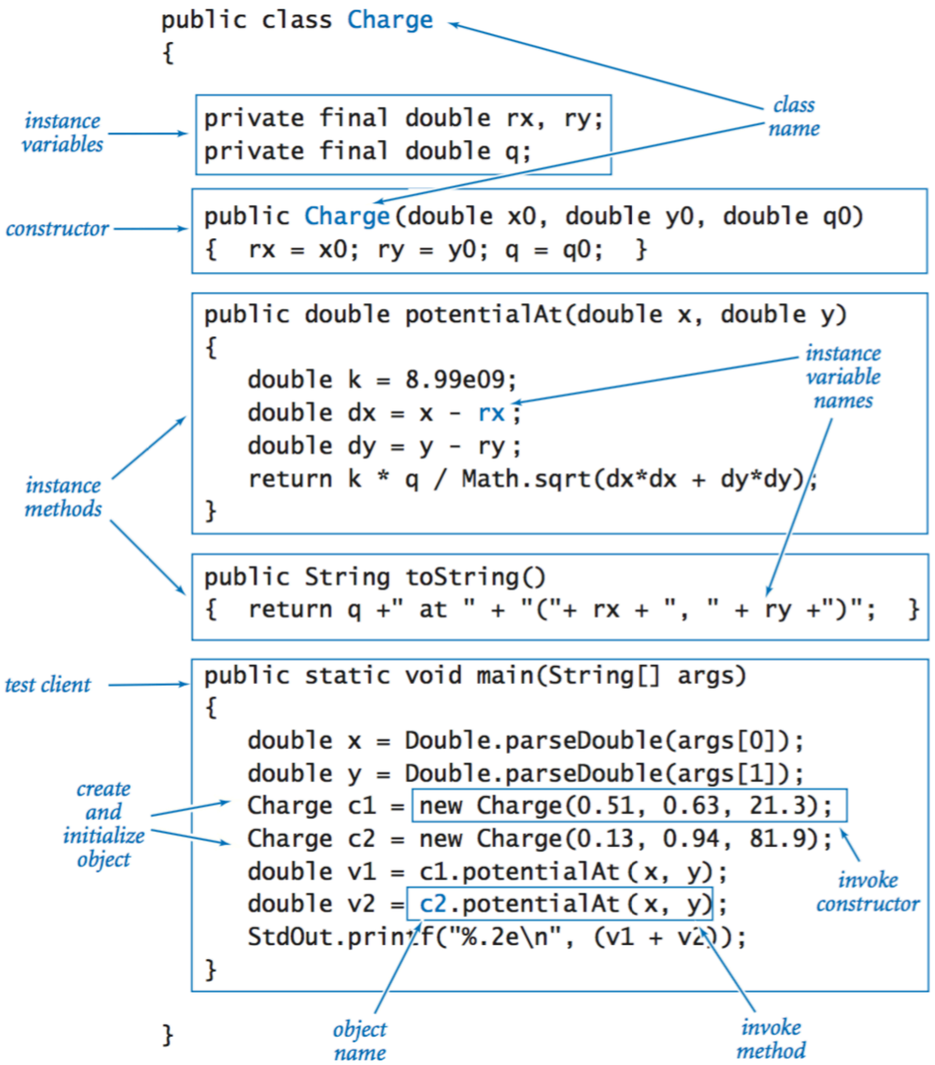
hypotenuse of a right triangle	`public static double hypotenuse(double a, double b)` `{  return Math.sqrt(a*a + b*b);  }`

harmonic number	`public static double harmonic(int n)`
	`{`
	`  double sum = 0.0;`
	`  for (int i = 1; i <= n; i++)`
	`  sum += 1.0 / i;`
	`  return sum;`
	`}`

uniform random integer in [0, n)	`public static int uniform(int n)` `{  return (int) (Math.random() * n);  }`

draw a triangle	`public static void drawTriangle(double x0, double y0,`
	`  double x1, double y1,`
	`  double x2, double y2 )`
	`{`
	`  StdDraw.line(x0, y0, x1, y1);`
	`  StdDraw.line(x1, y1, x2, y2);`
	`  StdDraw.line(x2, y2, x0, y0);`
	`}`

Classes

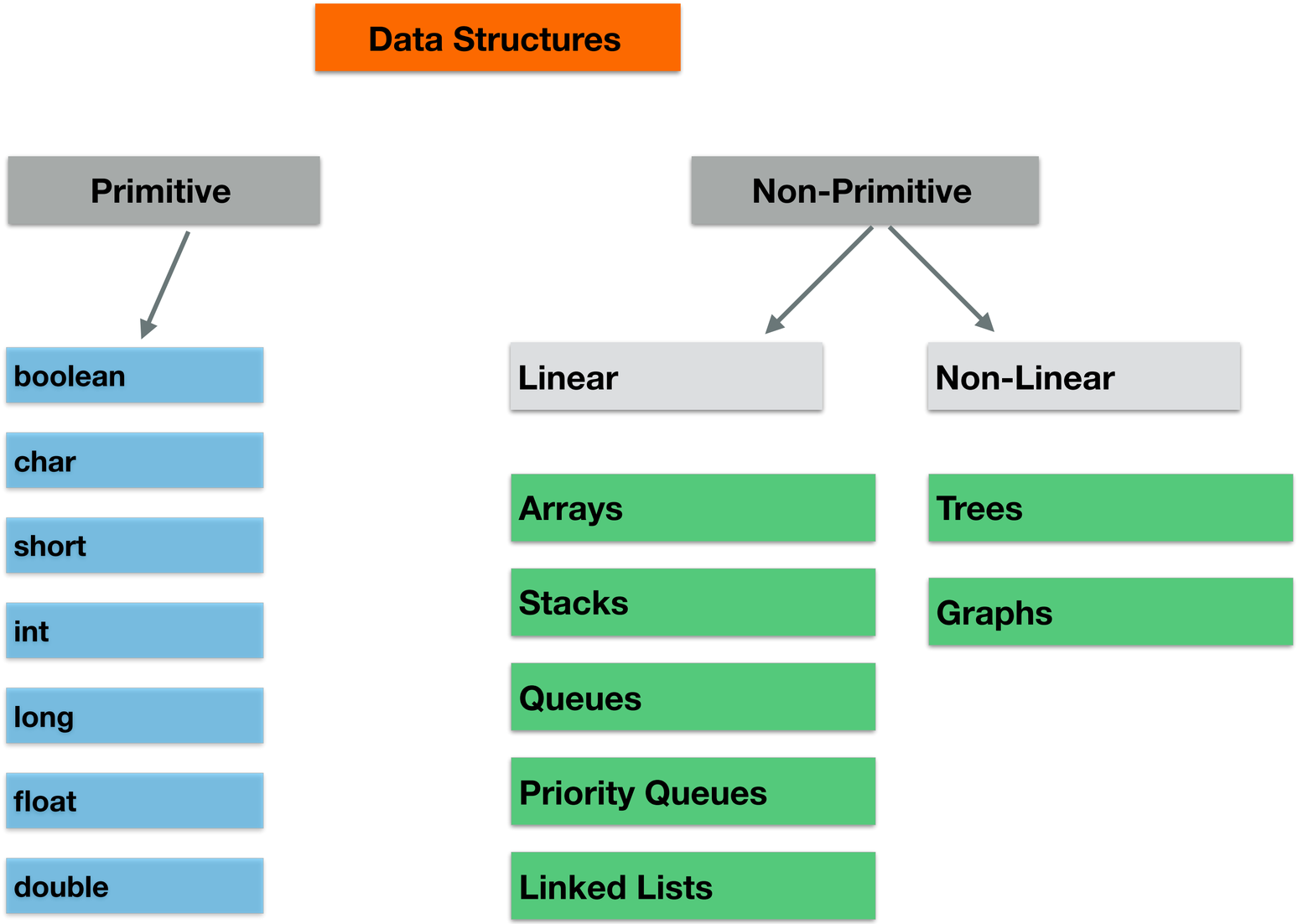

public class Charge
{
private final double rx, ry;
private final double q;
public Charge(double x0, double y0, double q0)
{ rx = x0; ry = y0; q = q0; }
public double potentialAt(double x, double y)
{
double k = 8.99e09;
double dx = x - rx;
double dy = y - ry;
return k * q / Math.sqrt(dx*dx + dy*dy);
}
public String toString()
{ return q +" at " + "("+ rx + ", " + ry +")"; }
public static void main(String[] args)
{
double x = Double.parseDouble(args[0]);
double y = Double.parseDouble(args[1]);
Charge c1 = new Charge(0.51, 0.63, 21.3);
Charge c2 = new Charge(0.13, 0.94, 81.9);
double v1 = c1.potentialAt(x, y);
double v2 = c2.potentialAt(x, y);
StdOut.printf("%.2e\n", (v1 + v2));
}
}
instance variables
class name
constructor
instance variable names
instance methods
test client
create and initialize object
invoke constructor
object name
invoke method

Generics

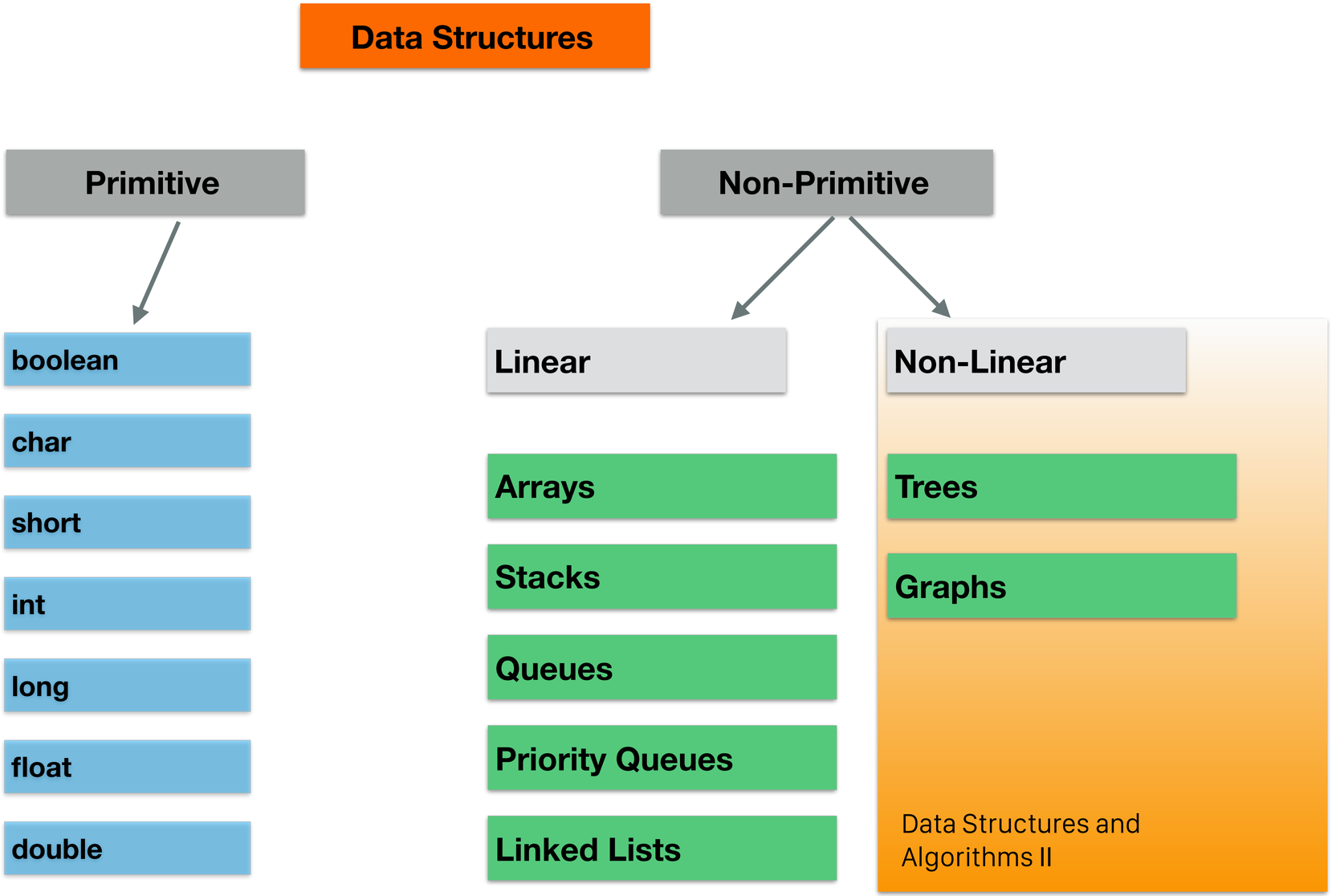
```
// before generics
List words = new ArrayList();
words.add("Hello ");
words.add("world!");
String s = ((String)words.get(0))+((String)words.get(1))
assert s.equals("Hello world!");


// with generics
List<String> words = new ArrayList<String>();
words.add("Hello ");
words.add("world!");
String s = words.get(0)+words.get(1); // no explicit casts
assert s.equals("Hello world!");
```

Overview of Data Structures



Overview of Data Structures



Overview Algorithms and Java

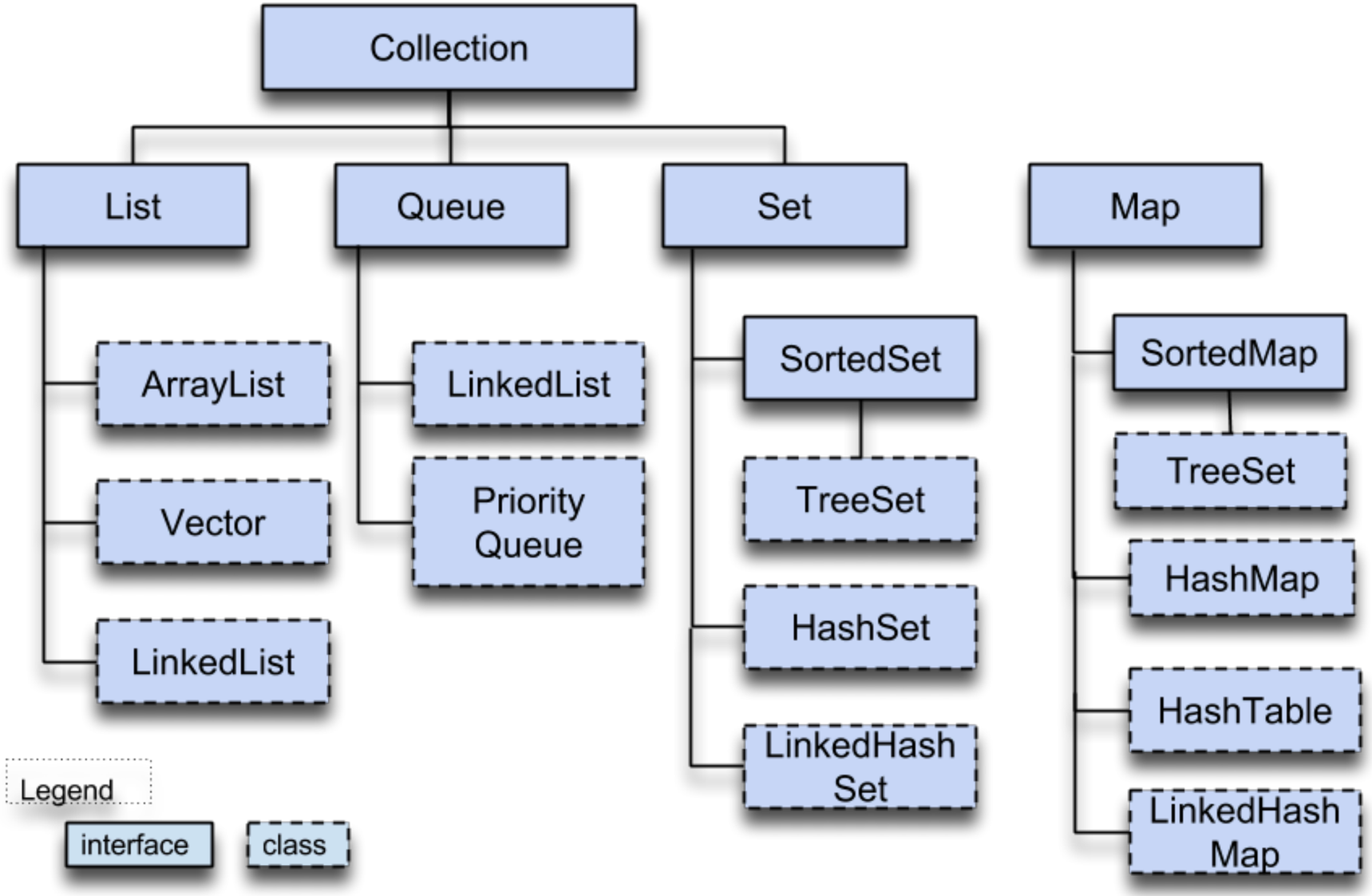
Algorithms

- Complexity O(n)
- Arrays
- Lists
- Stacks
- Queues
- Stacks
- Maps

Java

- Programming elements
- Expressions
- Control Flow
- Classes
- Writing, Testing, Debugging
- Generics
- Collections Framework

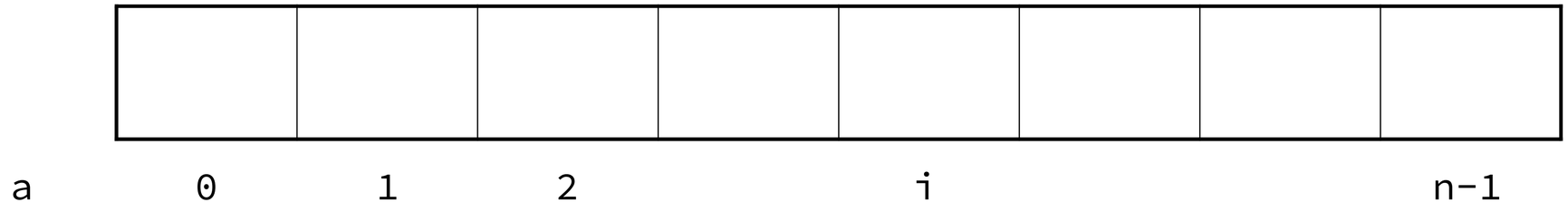
Java Collections



Arrays

Array Definition

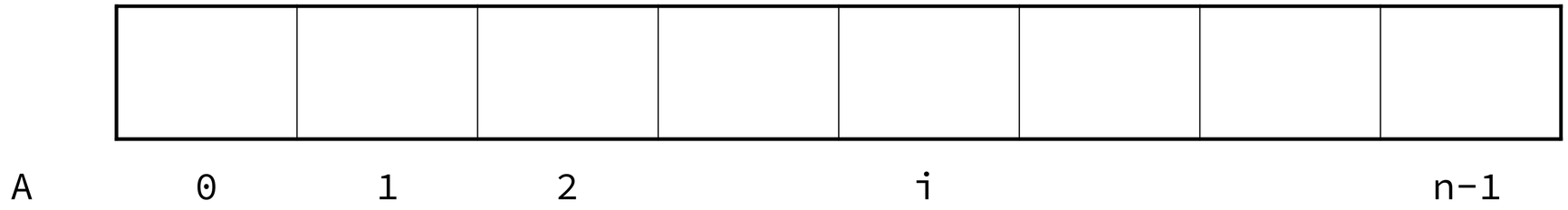
- An **array** is a sequenced collection of variables all of the same type.
- Each variable, or **cell**, in an array has an **index**, which uniquely refers to the value stored in that cell.
- The cells of an array, **a**, are numbered 0, 1, 2, and so on.
- Each value stored in an array is often called an **element** of that array.



Array Length and Capacity

Since the length of an array determines the maximum number of things that can be stored in the array, we will refer to the length of an array as its **capacity**.

In Java, the length of an array named **a** can be accessed using the syntax **a**.length. Thus, the cells of an array, **a**, are numbered 0, 1, 2, and so on, up through a.length–1, and the cell with index k can be accessed with syntax a[k].



Declaring Arrays

The first way to create an array is to use an assignment to a literal form when initially declaring the array, using a syntax as:

```
elementType[] arrayName = {initialValue0, initialValue1, …, initialValuen-1};
```

The elementType can be any Java base type or class name, and arrayName can be any valid Java identifier. The initial values must be of the same type as the array.

Declaring Arrays

- The second way to create an array is to use the **new** operator.
  - However, because an array is not an instance of a class, we do not use a typical constructor. Instead we use the syntax:

```
elementType[] arrayName = new elementType[n];
```

n is a positive integer denoting the length of the new array.

The **new** operator returns a reference to the new array, and typically this would be assigned to an array variable.

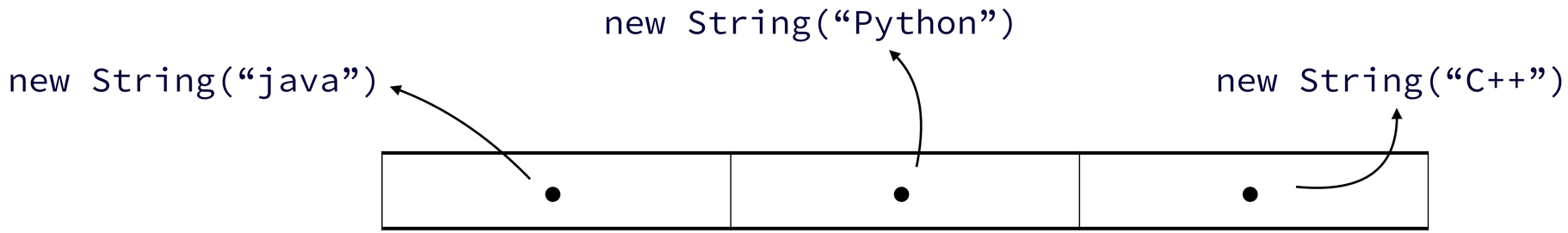
Arrays

- Arrays can be used to store primitives
  - boolean, byte, char, short, int, long, float, double

1	0	0	1	0

Or references to objects:

```
String[] s = new String{“java”, “C++”, “Python”};
```



Multidimensional Arrays

- In Java, the syntax for two-dimensional arrays is similar to the syntax for one-dimensional arrays, except that an extra index is involved:

```
int[][] a = new int[10][10];

int[][] b = { { 1, 2, 3 }, { 4, 5, 6, 9 }, { 7 }, };
```

Direct initialisation

- Java does not actually have two-dimensional arrays
- The elements in a 2D array of type `int[][]` are variables of type `int[]`.
- A variable of type `int[]` can only hold a pointer to an array of `int`. So, a 2D array is really an array of pointers, where each pointer can refer to a one-dimensional array.

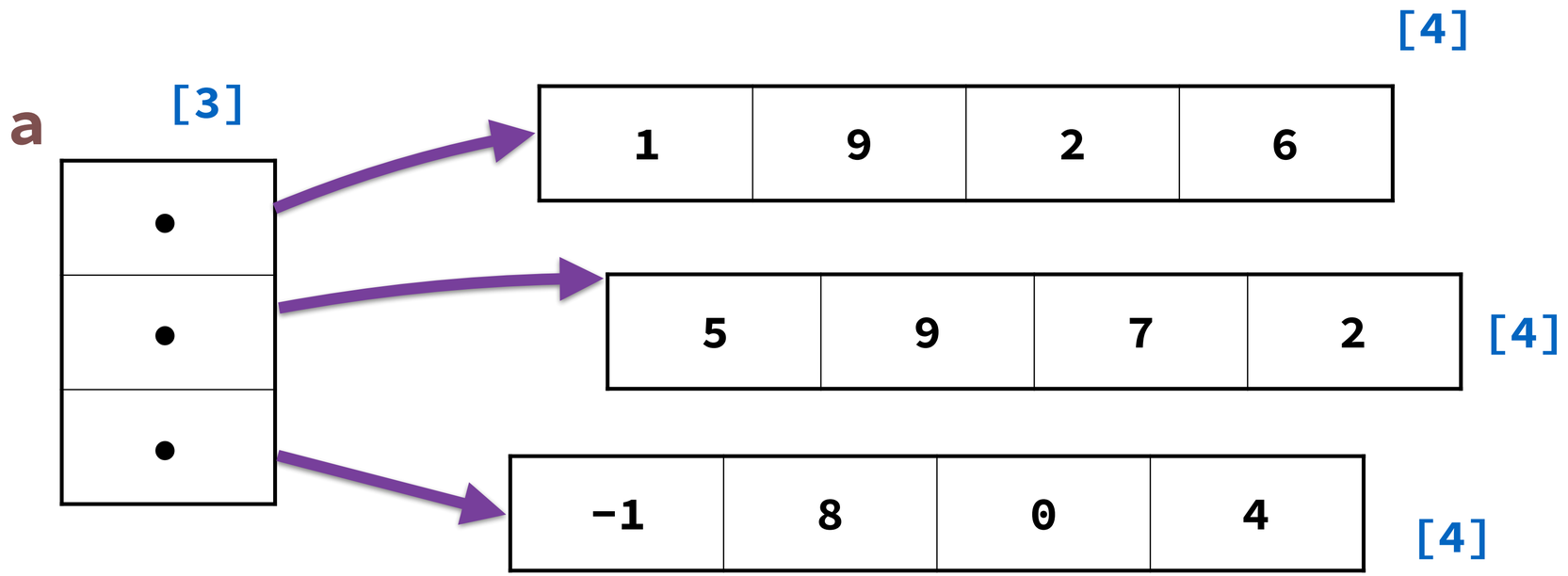
Multidimensional Arrays

rows

columns

```
int[][] a = new int[3][4];
```

1	9	2	6
5	9	7	2
-1	8	0	4



Multidimensional Arrays

- Higher dimensional arrays:

```
int[][][] a = new int[10][10][10];
```

- Straightforward extention of 2D syntax

java.util.Arrays

- The Arrays class of the java.util package contains several static methods that we can use to fill, sort, search, etc in arrays. This class is a member of the Java Collections Framework and is present in java.util.arrays. Sample of methods:

public static String toString(int[] a)	The string representation consists of a list of the array's elements,
public static void sort(int[] a)	Sorts the specified array into ascending numerical order.
public static int[] copyOf(int[] original, int newLength)	Copies the specified array and length. It truncates the array if provided length is smaller and pads if provided
public static void fill(int[] a, int val)	Fills all elements of the specified array with the specified value.

Array Examples

- Creating arrays
- Different object types
- Iterating over arrays
- Passing arrays as arguments to functions

Programming Competitions

ACM ICPC

The **International Collegiate Programming Contest** is an algorithmic programming contest for college students. Teams of three, representing their university, work to solve the most real-world problems, fostering collaboration, creativity, innovation, and the ability to perform under pressure. Through training and competition, teams challenge each other to raise the bar on the possible. Quite simply, it is the oldest, largest, and most prestigious programming contest in the world.

World Finals ICPC 2019
Porto 31 Mar - 5 Apr

hosted by the University of Porto

acm International Collegiate Programming Contest

IBM event sponsor

interested? get in touch with me for more details...