Primitive and Reference types

* Primitive: int, bool : stores values in the storage location.
* Reference: Int[], string: stores a reference to the LOCTATION where the data is stored.

Generic programming

* Type casting
* Wrapper classes allow primitives to be auto boxed into objects
* Generic types: “ArrayList<Shape> shapes = new Arraylist<T>()”, where T is a placeholder for type, in this case we would replace it with Shape.
* Run time/compile time error differences, compile time errors are always better because it will always encounter the error. Run time errors could get hidden and then cause issues later down the road.
* An ArrayList<Circle> IS NOT an ArrayList<Shape>. Even though Circle is a Shape object

Algorithm Analysis

* N-> Problem size. Linear, Quadratic, and logarithmic, cubic, and more complexity classes etc.
* For program performance: **Algorithmic complexity** is the most important, then: code quality, machine speed, and complier/OS.
* Big-O analysis takes the slowest component, or dominant term, example: 3N^2 + 13N + N in big O notation is O(N^2), where we get rid of constant coefficients.

Lambda expressions and Logs

* (str1, str2) -> str2.compareTo(str1)
* Logarithms, y=10^x, x = log(y)
* In CS we usually assume log base 2, although it doesn’t really matter in Big-O
* O(F(N)) for Big O, T(N) for timing, such that F(N) T(N).
* T(N)/F(N)= R , where R should be a constant. If R grows, underestimated F(N), and vice versa if R shrinks. Constant means we were correct.
* Binary Search, the cutting half method. List must be sorted. By doing ½ maximizes guaranteed elimination.

REVIEW

* NlogN is best case for an unknown distributed list sort

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | Selection | Insertion | Shell | Merge | quick |
| B | N^2 | N |  | NlogN | NlogN |
| A | N^2 | N^2 | N^5/4 | NlogN | NlogN |
| W | N^2 | N^2 | N^3/2 | NlogN | N^2 |

* How to choose sorting?:
* **Notes for A5: mergesort and Quicksort, must switch to insertion sort for small lists.**
* **Mergesort pass a tempary list of the right size. Analysis is 50% of the score**

Tuesday REVIEW:

**A Queue is an ADT.**

It provides access to the oldest added item.

Functionality:

• Offer (enqueue) – add item to back

• Poll (dequeue) – remove and return item

from the front.

• Peek – return front item without removing

**A Stack is an ADT.**

It provides access to the most recently

added item.

Functionality:

• Push – add item to top

• Pop – remove and return top item

• Peek – return top item without removing