Geometry Problems
Examples for Typical ACM Instances

Elementary Geometry
- application of sine/cosine formulas
- spherical coordinates
- distances between lines, line segments, polygons, etc.
- area formulas

Algorithmic Geometry
- convex hull
- intersection testing
- iterative approach for maximising/minimising objective function
Traps

- numerical problems (rule of thumb: use double, not float)
- input can contain collinear points or same points twice
- slope of a line can become infinity

Tips

- sometimes brute force is good enough
- can you apply a plane-sweep/scan-line algorithm?

http://www.faqs.org/faqs/graphics/algorithms-faq/
http://mathworld.wolfram.com
http://www.ics.uci.edu/~eppstein/junkyard/
Floating Point Numbers

IEEE 754

- sign bit, biased exponent, significand (mantissa)
- float: 32 bit, sign 1, exponent 8, significand 23 (about 7 decimal digits)
- double 64 bit, sign 1, exponent 11, significand 52 bit (about 16 decimal digits)
- long double, **nonstandard** 96bit storage but only 80bit precision, sign 1, exponent 15, significand 63 (in Java use BigDecimal)
- only subset of the reals. e.g. $\frac{1}{3}$ not representable
  $\Rightarrow$ rounding to nearest
Floating Point Numbers

Example: $\pi$

- $\pi$ rounded to 24 bits: 11.001001000011111101101
- $\textit{significand} = 1.100100100001111101101$
- $\textit{exponent} = 1$
- $\Rightarrow \pi = 1.100100100001111101101 \cdot 2^1$
Absolute vs. Relative Error

**Absolute Error**

- During arithmetic floating-point operations a loss of precision can occur.
- How can one test the equality of two numbers?
- Absolute error: \(|a - b| < \epsilon\).
Absolute vs. Relative Error

Absolute Error
- During arithmetic floating-point operations a loss of precision can occur.
- How can one test the equality of two numbers?
- Absolute error: $|a - b| < \epsilon$.

Example
- 32-bit floating point calculation with an expected result of 10000. Precision of 6 decimal digits ($\epsilon = 0.000001$).
- However, the result is 10000.000977 (nearest number to 10000 using 32-bit float)
- Absolute error: 0.000977 = 977 $\cdot$ $\epsilon$.
- Solution: Use relative error.
Absolute vs. Relative Error

Relative Error

- Relative error: \(|(a - b)/b| < \epsilon\).
- In our case: 0.0000000977 < 0.000001
Relative Error

- Relative error: \(|(a - b)/b| < \epsilon\).
- In our case: 0.0000000977 < 0.000001

Summary

- Absolute error doesn’t work for big positive and negative numbers.
- Relative error doesn’t work not for numbers around 0. Why?
  - Two numbers are equal if: \(|a - b| < \epsilon\) or \(|(a - b)/b| < \epsilon\).
- Smallest epsilon for 32-bit float: \(1.19209 \cdot 10^{-07}\).
- Smallest epsilon for 64-bit float: \(2.22045 \cdot 10^{-16}\).
Closest Pair of Points

Algorithms

Voronoi diagram
Delaunay triangulation
or (easier)
Divide and Conquer
Plane-Sweep / Scan-Line

Geometry Problems
Closest Pair of Points

Algorithms

- Voronoi diagram
- Delaunay triangulation
- or (easier)
- Divide and Conquer
- Plane-Sweep / Scan-Line
Convex Hull

Algorithms

- Voronoi Diagram (points of unbounded regions)
- Gift Wrapping
- Graham Scan
- Quickhull
- Divide and Conquer
- Incremental
- Double-Chain
Convex Hull

Algorithms

- Voronoi Diagram (points of unbounded regions)
- or (easier)
- Gift Wrapping
- Graham Scan
- Quickhull
- Divide and Conquer
- Incremental
- Double-Chain