Some Whig History of Computing

~200BC Euclid — the first non-trivial algorithm (to calculate **gcd**)

~1200 Fibonacci — introduction of "Arabic" numbers, easy algorithms for arithmetic

1900 Hilbert — Find an *algorithm* which determines if a Diophantine equation has a solution.
BIG Question: What is an *algorithm*?

~1930 Gödel— A logical system cannot prove its own consistency.
Algorithm $\equiv$ Recursive function

~1938 Church — Algorithm $\equiv$ λ calculus
Turing — Algorithm $\equiv$ Turing machine
Unsolvable Problems. (Problems which DON'T have algorithms.)
(Solvable Problems have algorithms.)
"Reduction": Problem B is unsolvable because if B were solvable then A would be solvable,
BUT, we know A is unsolvable, so B is unsolvable.

~1940s Neural Nets: models of computation which are strictly weaker than Turing machines.

~1950s Grammars: models of computation which are weaker than Turing machines and stronger than neural nets.

~1950s-60s Some problems "Harder" than others.
Reductions among problems: e.g. MULTIPLICATION $\equiv$ SQUARING .
A.I. : many problems can be reduced to "theorem proving".
Edmonds — "reasonable" problems have polynomial time algorithms.

~1970s Cook — "theorem proving" is the "Hardest" problem in a class of problems.
(SAT is NP-complete.)
Karp — all those *reductions* show that many interesting problems are NP-complete.
Garey & Johnson — a long list of NP-complete problems.

~1980s Can something similar to NP-complete tell us why some problems are hard to parallelize?
(Not entirely successful)

∼1990s Approximation Algorithms / Probabilistic Algorithms

Some problems can be approximated (quickly),

BUT some problems can be approximated (quickly) only if P = NP.

If we can't always get the correct answer, can we do so most of the time and/or do so quickly most of the time?

(New Complexity Classes – Complexity Zoo)

Interactive Protocols — Two or more Provers and Verifiers

Probabilistic Proof Systems — New characterization of NP

(Zero-knowledge Proof Systems)

(Probabilistically checkable Proofs)

(Quantum Computers ??)