Mathematical Proofs

$$\sum_{k=0}^{N} \log k \qquad \int_{0}^{\infty} e^{-x^2}\, dx$$

$$|a+b| \leqslant |a| + |b|$$

$$e^{\pi i} - 1 = 0$$

Carlos Moreno
cmoreno@uwaterloo.ca
EIT-4103

https://ece.uwaterloo.ca/~cmoreno/ece250

These slides, the course material, and course web site are based on work by Douglas W. Harder

Mathematical Proofs

- Today's class:
  - We'll investigate a few additional techniques to prove statements, including:
    - Direct proof
    - Proof by construction
    - Proof by enumeration (or exhaustion)
    - Proving the contrapositive
    - Proof by contradiction (very useful in general)
    - Proof by reduction (very useful in the context of algorithms)

Mathematical Proofs

- Direct proof
  - In general the simplest form to prove statements.
  - The result is directly obtained from the hypothesis (along with the basic axioms, etc.)
  - Example: proving that the square of an even number is even:

    Proof: An even number has the form $2k$. Its square is $(2k)^2 = 4k^2 = 2(2k^2)$, and so it is even.

Mathematical Proofs

- Proof by construction
  - Usually applicable to statements about existence of some entity — by explicitly constructing an example of such entity, the statement is proven.

Mathematical Proofs

- Proof by enumeration (or exhaustion)
  - The statement is proved by going over all possible conditions, and proving the statement for each one individually. Obviously, this is feasible when there is a reasonable number of conditions.
  - Example: prove the triangle inequality for real numbers (we won't really prove it — just sketch the procedure to illustrate the technique)

Mathematical Proofs

- Proof by enumeration (or exhaustion)

  In this case, the relevant possibilities are *a* and *b* being negative or non-negative, with the combinations of each one's magnitude being the larger.

  For example, if a is non-negative and b is negative, then we have: $|a| = a$ and $|b| = -b$, and $|a+b| = a+b$ if $|a| \geq |b|$ or $-(a+b)$ otherwise. In the first case, the inequality leads to $a+b \leq a-b$, or $b \leq -b$, which is true since b is negative. (etc. etc.)

Mathematical Proofs

- Proving the contrapositive
  - The contrapositive of the statement A $\Rightarrow$ B, is the *strictly equivalent* statement Not B $\Rightarrow$ Not A.
  - Since the two forms are equivalent, proving one proves the other one.
  - Example: Prove that if n^2 is even, then n is even.

Mathematical Proofs

- Proving the contrapositive
  - The contrapositive of the statement A $\Rightarrow$ B, is the *strictly equivalent* statement Not B $\Rightarrow$ Not A.
  - Since the two forms are equivalent, proving one proves the other one.
  - Example: Prove that if n^2 is even, then n is even. (Huh... didn't we already prove this?? Doesn't the direct proof example prove this one as well??)

Mathematical Proofs

- Proving the contrapositive
  - The contrapositive of this statement is: if n is odd, then n^2 is odd (if Not{n is even}, then Not{n^2 is even}), and this one is easy to show by direct proof:

    An odd number has the form 2k+1, and its square is $(2k+1)^2 = (4k^2 + 4k + 1) = 2(2k^2 + 2k) + 1$, and so is odd.

Proof by Contradiction

- There's this old (colloquial) adage that “you can't prove a negative”
- The rationale being more or less that proving something to be impossible or not to exists requires exhausting all possibilities in the Universe. Any attempt to show that it's not possible might show instead that one is unable to, and not that it can not be done.

Proof by Contradiction

- However, that old adage could not be more wrong!
- Arguments by contradiction are a quite remarkable way around that rationale!

Proof by Contradiction

- To prove a statement by contradiction, you assume that the statement is false (or equivalently, assume the negation of the statement to be true), and show, using deductive steps and axiomatically true arguments, that such assumption leads to some inconsistency (contradiction).

Proof by Contradiction

- If every step is correct and you did everything right (which is an obvious condition for any proof to be correct), then the only item in question is the initial assumption, so it must be the one that is proven false.
- But the initial assumption is that what you want to prove is false, and you then proved that *that* is false — thus, what we wanted to prove *is* proved.

Proof by Contradiction

- One of the most remarkable examples (IMHO) is the proof that $\sqrt{2}$ is not a rational number.
- It would seem (if we follow that old adage's line of thought) impossible to prove.... No matter how many digits you show (showing that there is no periodicity), you still don't know if later on periodicity will appear...
- The argument is completely different ... Let's assume that it is a rational number, and let p/q be the reduced form of that fraction.

Proof by Contradiction

- That is, *p* and *q* share no common factors, and it holds, of course, that

$$\left(\frac{p}{q}\right)^2 = 2 \Rightarrow p^2 = 2q^2$$

Proof by Contradiction

- That is, *p* and *q* share no common factors, and it holds, of course, that

$$\left(\frac{p}{q}\right)^2 = 2 \Rightarrow p^2 = 2q^2$$

So, p^2 is even... We already proved that this means that *p* is even; say, $p = 2k$ for some *k*. Then, what about *q* ... ?

Proof by Contradiction

- I'm sure you see where this is going.... Try to complete the argument (we'll work the complete example in class, of course).

Proof by Contradiction

- Another reasonably neat example:
- Prove that there are infinitely many prime numbers

Proof by Contradiction

- Another reasonably neat example:
- Prove that there are infinitely many prime numbers (and again, talk about a remarkably powerful argument — statements like these may sound like the perfect example for those that defend that old adage .... How can you convincingly argue that there are infinitely many of something without showing them all? How can you know that they won't stop after a certain value??)

Proof by Contradiction

- Philosophy aside, let's prove it!
- Assume there are finitely many primes:

$$p_1, p_2, p_3, \cdots, p_{n-1}, p_n$$

And consider the value obtained as the product of all primes + 1:

$$d = 1 + p_1 p_2 p_3 \cdots p_{n-1} p_n$$

Proof by Contradiction

- The value *d* is larger than the largest prime (since it is the largest prime times all the other primes + 1), so it can not be prime.
- But, as we proved last class, this means that *d* must be divisible by a prime — say, p_k (one of the *n* primes that exist). It could be divisible by more than one prime, but we only consider one (which anyway is all we proved last time!)
- So, $d = p_k \cdot m$, for some *m*.

Proof by Contradiction

- You *have* to see where this is going .... (can you complete the argument?)

Proof by Contradiction

- You *have* to see where this is going .... (can you complete the argument?)
- Hint: what else, apart from d, has p_k as a factor?
- (again, we'll look at the complete example in class)

Proof by Contradiction

- In both examples so far, the argument leads to a contradiction of some of the "background" assumptions — something that we know to be axiomatically true is contradicted by our assumption that the given statement is false, leading to the conclusion that the given statement then must be true.

Proof by Contradiction

- In both examples so far, the argument leads to a contradiction of some of the "background" assumptions — something that we know to be axiomatically true is contradicted by our assumption that the given statement is false, leading to the conclusion that the given statement then must be true.
- In some cases, we might end up contradicting the assumption itself, like in the next example.

Proof by Contradiction

- BTW, in some cases, we might end up contradicting the hypothesis — what does that look like? (I mean, what other technique for proofs would that look like?)

Proof by Contradiction

- Example: (multi-processor parallel processing)

  You have some processing totalling time T, and it can be split, in a continuous and arbitrary way, into N chunks, to be executed in parallel in N processors.

  Prove that the optimal execution time is reached when the task is split into N equal-size subtasks.

Proof by Contradiction

- This example illustrates an additional interesting aspect that sometimes makes the technique applicable: when we have to prove some condition about one particular combination among many, if we negate the condition, it may be easy to focus on one aspect of the negated condition that covers all other combinations. Trying to prove the "positive" form of the statement may end up being quite challenging by comparison!

Proof by Contradiction

- Anyway.... the proof:
- Assume, for a contradiction, that the the optimal time T_O is reached for a splitting into N subtasks that are not all equally-sized.
- If they're not equally sized, then we can pick the task of largest size (let's call it T_{MAX}) and the task of smallest size (let's call it T_{MIN})
- Notice that in this case, $T_O = T_{MAX}$ (the task is not complete until the subtask for T_{MAX} is done)

Proof by Contradiction

- Given these conditions, we can find a different splitting with lower execution time:
- Since $T_{MAX} > T_{MIN}$ (they can't be equal, since by assumption, we're splitting into tasks that are not equally-sized), then we can rearrange these two tasks so that each one takes $(T_{MAX} + T_{MIN})/2$
- But then, the T_O that we obtain (let's call it T'_O is lower than T_{MAX} (why? Detailed answer, please! There are several aspects involved)

Proof by Contradiction

- Summarizing: $T'_0 < T_0$, but this contradicts the assumption that the splitting that we had was optimal (since we're showing another one that gives a better execution time)
- Thus, the optimal splitting must be the one with N equally-sized tasks.

Proof by Contradiction

- Summarizing: $T'_0 < T_0$, but this contradicts the assumption that the splitting that we had was optimal (since we're showing another one that gives a better execution time)
- Thus, the optimal splitting must be the one with N equally-sized tasks.
- (Careful: there is a severe flaw in this proof; although very easy to fix — can you see it? Even better, can you fix the argument?)

Proof by Reduction

Proof by Reduction

- In the context of algorithms, *reduction* refers to implement one algorithm in terms of another one.
- If we implement algorithm A in terms of algorithm B, we say that A reduces to B.
- This has obvious theoretical applications, but also practical ones — maybe an algorithm is readily available, and it may be easier to reduce to that one rather than implementing from scratch some new algorithm that we require.

Proof by Reduction

- It may be helpful for proofs about execution time of some algorithms.
- For example, if algorithm A is known to be "hard", and we can find a reduction from A to B, then that proves that B can not be easy (in any case, it can not be easier/faster than A).
- Right? If B was easy, then A would also be easy, contradicting the known fact (or in any case the assumption) that A is hard

Proof by Reduction

- There's actually a subtlety in here, that ruins that argument (as stated)... Can you see what the problem is?

Proof by Reduction

- There's actually a subtlety in here, that ruins that argument (as stated)... Can you see what the problem is?
- Subtlety aside, let's see one rather neat example (the subtlety I'm referring to is really one extra condition that needs to be added, and that extra condition doesn't ruin the applicability of the technique):

Proof by Reduction

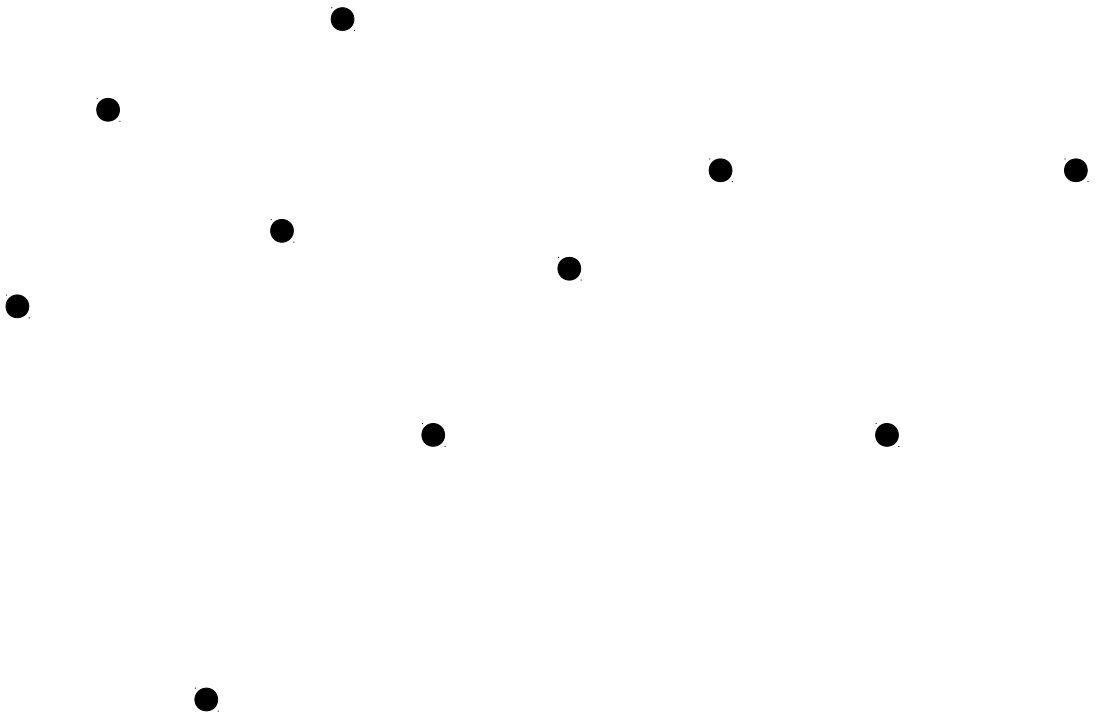
- Prove that an algorithm to find the convex hull of *n* points in the plane can not run in a worst-case better than an amount proportional to n log(n).

Proof by Reduction

- Preliminary definitions:
- Convex hull of a set of points: the smallest (in area) convex region that includes all the points.
- It is a known fact that for a set of points, the convex hull is always a convex polygon — the following example illustrates this:

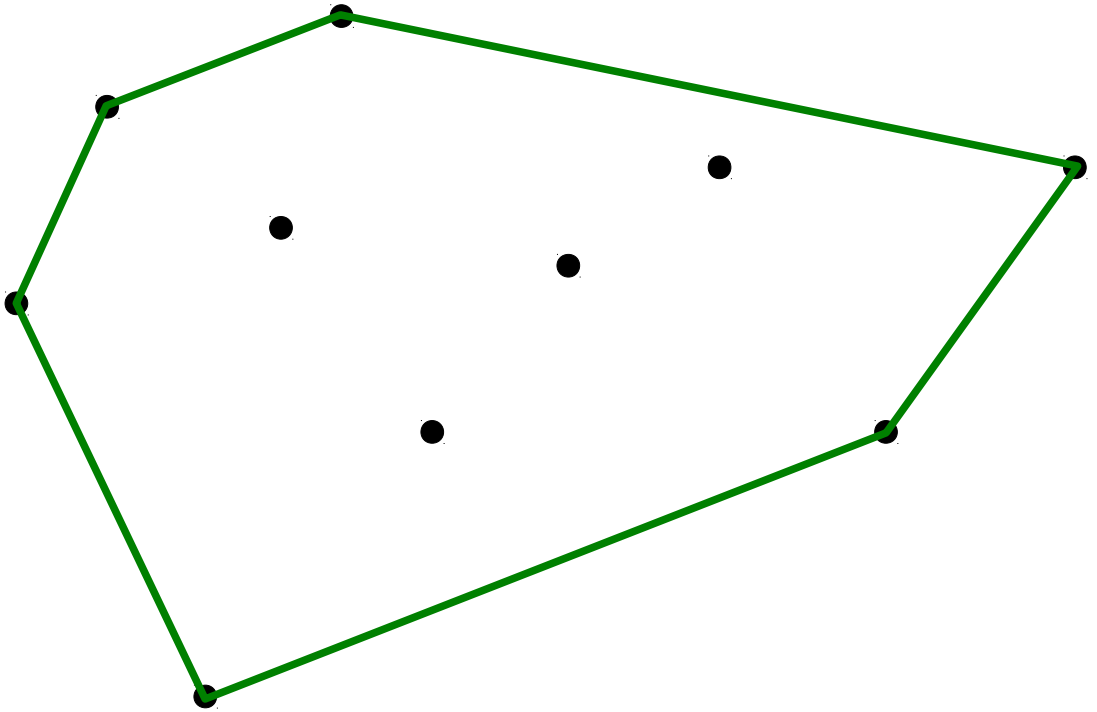
Proof by Reduction

Consider the following set of points:



Proof by Reduction

Their convex hull:



Proof by Reduction

- Preliminary definitions:
- An important detail is that an algorithm to compute the convex hull must output the vertices of the polygon *in sequence*. If not, we don't really have the convex hull — we would have a set of points that requires extra work to determine what the convex hull really is.

Proof by Reduction

- Another preliminary:

  It is a proven fact that (under certain conditions, operating on a single processor computer with standard assumptions on memory, assembler instruction set, etc.) that no sort algorithm can sort a set of *n* arbitrary values in a worst-case time better than an amount of time proportional to n log(n)

Proof by Reduction

- We're now ready to prove the statement.
- As you already suspect, we're going to reduce sorting to computing a convex hull.
- That is, we're going to find a way to sort a set of values given an algorithm that computes a convex hull.
- This will clearly require to somehow construct points from the given values (not a big challenge, really...)

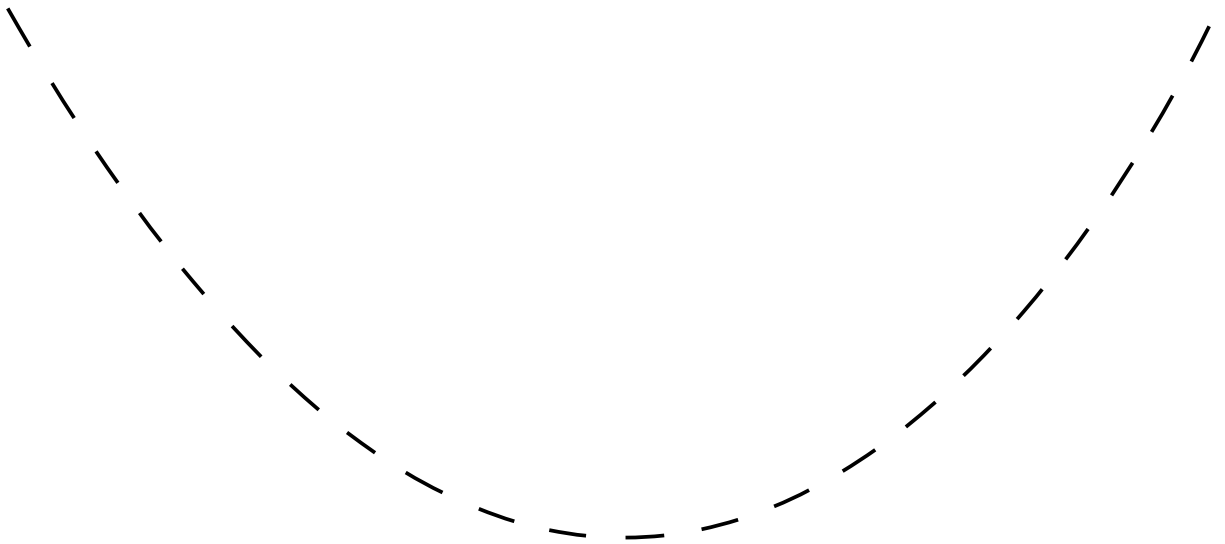
Proof by Reduction

- The real challenge is finding a way to construct points in a way that the convex hull will output something that we know will be useful for the purpose of sorting (and thus, useful for the purpose of our argument / proof)

- Hint: What is the convex hull of a bunch of points that lie on the parabola given by $y = x^2$?

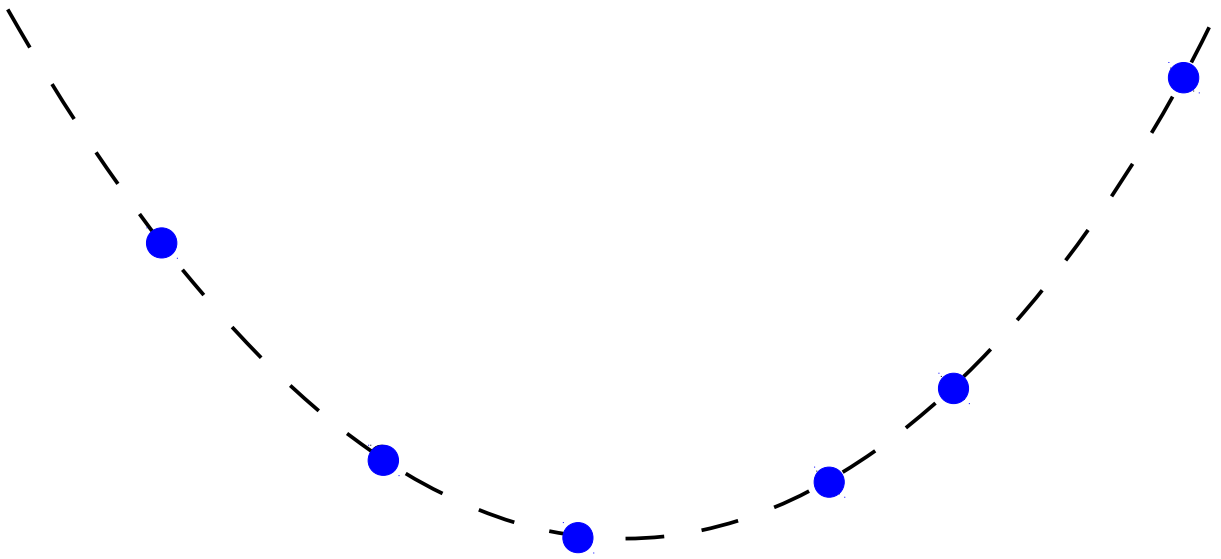
Proof by Reduction

- Let's try and answer that question with an example:



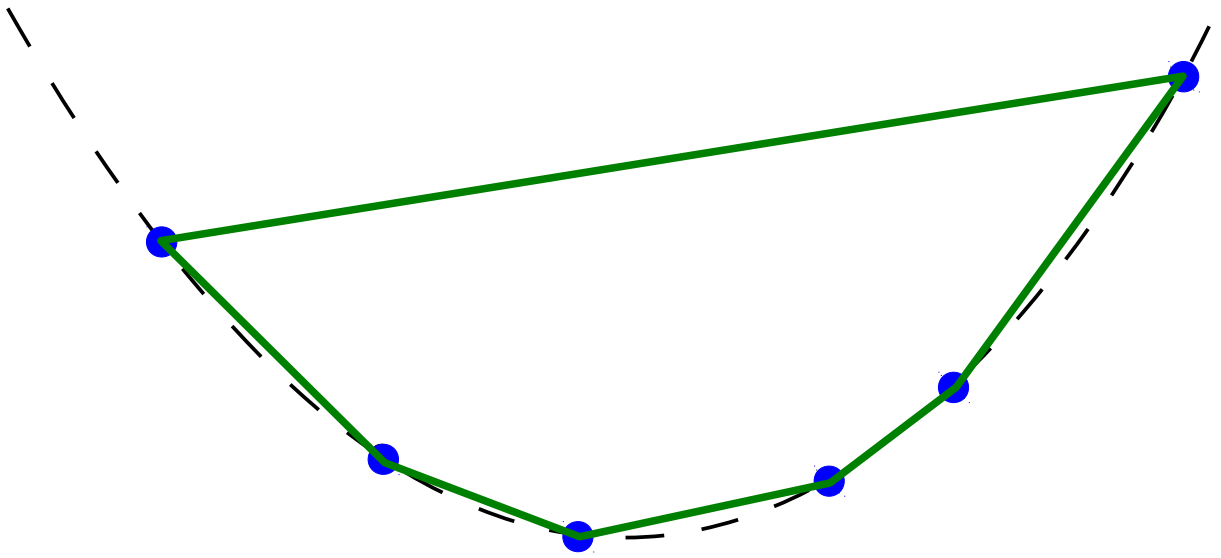
Proof by Reduction

- Let's try and answer that question with an example:



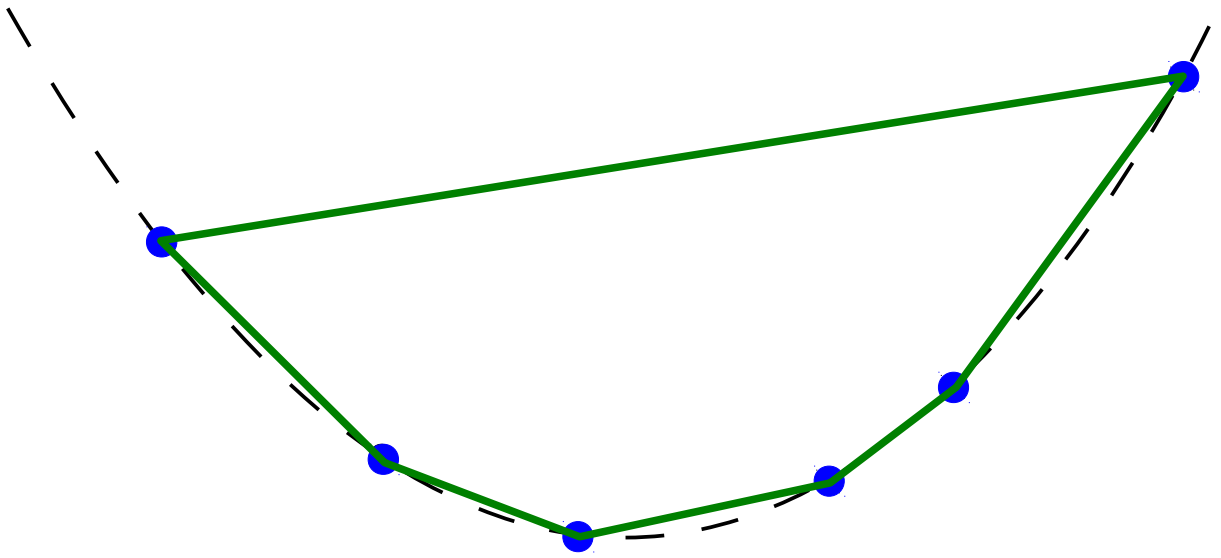
Proof by Reduction

- Let's try and answer that question with an example:



Proof by Reduction

- Key detail being: the convex hull always contains every point (the parabola is convex!)



Proof by Reduction

- The other key detail being what we already said: any convex hull algorithm would have to output those points in sequence.
- But in this case, the sequence involves the points in order by their x-coordinate (right?)
- So, we're done ... (right? You *have* to see where this is going!!)

Proof by Reduction

- The other key detail being what we already said: any convex hull algorithm would have to output those points in sequence.
- But in this case, the sequence involves the points in order by their x-coordinate (right?)
- So, we're done ... (right? You *have* to see where this is going!!)
- Ok, one extra hint, but then you complete the argument....

Proof by Reduction

- We probably want the input values (the ones that need to be sorted) to be the x-coordinates of the points that we're going to feed as input to the convex hull algorithm....

- (as usual, we'll complete this and discuss the details in class)