The Design of efficient Algorithms

Definition: An algorithm is a precisely defined method for solving a problem that gives correct answers for all instances of the problem.

Examples:
Adding 2 numbers, multiplying 2 numbers dividing 2 numbers.
We all learned algorithms for these operations in elementary school.

The following is a very famous “problem” seeking a “good” algorithm to solve it. “Good” = efficient = fast.

The traveling salesman problem, or TSP for short, is this: given a finite number of "cities" along with the cost of travel between each pair of them, find the cheapest way of visiting each one of the cities, once and only once, and returning to your starting point.

Who wants to be a millionaire????

YOU????? OK! Solve this problem. No Kidding!

See
http://www.claymath.org/millennium/

Read it and then click on the link at the right “P vs. NP”
Q: Why is this problem important??????

A: This problem is “equivalent” to many others in the sense that a solution to TSP will be a solution to those others as well. One of the “equivalent problems” is that of breaking the secret codes that are used by government, military and commercial institutions.

Can you imagine what would happen if someone succeeded in “solving” TSP? Would you ever order anything on-line again?? What about the safety of your bank and credit card accounts, what about government secrets …..

Get it?????!!!
But … I will tell you a secret … shhhh … don’t tell … I CAN SOLVE IT!!!!!!!

Here is a simple solution!!!! (We can split the $1M prize. ☺)  
Solution:
1. Say there are 5 cities. Call them a,b,c,d,e. Lets say we want start from city a and return to city a.

2. So, every possible “tour” will be a list of cities, starting with “a” and ending with”a” and having four cities listed between them. For example: “a bcde a” or “a edbc a”. Now some of these will be possible (since there are roads that connect the cities in that order) and some will not be possible.

3. Make a list of all sequences of the right form: start with “a”, followed by some permutation (=arrangement) of the letters b,c,d,e, followed by “a”. There are 4! = 24 such sequences. (why? ….)

4. Go thru the list from top to bottom. For each entry see if the sequence represents a possible tour for the salesman (i.e. are there roads connecting the cities in that order). If it is a possible tour – calculate its cost, and compare it to the cost of the shortest tour found so far.

5. When you are finished going through the list, you will end up with the shortest one.

Is this an algorithm? Absolutely! – why?
Is it a “good” one? Absolutely not!!!! Why not? Well ….
Let’s say we have 101 cities. So we have to go thru a list with 100! entries (since the starting and ending cities are the same) How long will it take to go thru that list? Well……

0. There are 100! arrangements of the 100 cities.
1. Notice that 100! > 2^{100} i.e. 100 x 99 x 98 ... x 1 > 2 x 2 x 2 ... x 2.  
   100 terms  100 terms
2. Notice also that since 2^{10} = 1024 and 10^3 = 1000, 2^{10} is bigger than 10^3. 
3. So 2^{100} = (2^{10})^{10} > (10^3)^{10} = 10^{30}. so ....

4. A list with 10^{30} entries is shorter than a list with 100! entries. So … let’s see how long it would take to process the shorter list realizing that the list with 100! entries would take much longer.

5. Assume we can process 1 billion entries (i.e. suggested routes) a second (that’s a lot, by the way). Now, one billion = 10^9, so if we have 10^{30} entries it will take 10^{30} / 10^9 = 10^{21} seconds to process all those routes.

6. How much time is that? Do we have time to get a coffee? How about lunch? Maybe a walk around the block …? Let’s see.

7. How many seconds in a minute, hour, day, year?
   Sec in minute = 60 < 100 sec = 10^2
   Sec in hour = 60 x 60 < 100 x 100 sec = 10^2 sec.
   Sec in day = 60 x 60 x 24 < 100 x 100 x 100 sec.
   Sec in year = 60 x 60 x 24 x 366 < 100 x 100 x 100 x 1000

   So … number of seconds in a year is less than 10^2 x 10^2 x 10^2 x 10^3 = 10^9.

   So if we divide 10^{21} (the number of seconds needed to process the abbreviated list) by 10^9 (an overestimate of the number of seconds in a year)

   we get we get a low estimate on the number of years it would take to process the abbreviated list.

   Now, 10^{21} / 10^9 = 10^{12} years.

   This is 10^3 x 10^9 = a thousand billion years = one trillion years!

   Most estimates put the age of the universe at 15 billion years.
   Conclusion: To figure out the shortest route for just 100 cities in the traveling salesman problem using my “very clever” algorithm would take a considerably longer time than the present age of the universe.

   See why they didn’t give me the money????