

Rules To Problem Solving

By Richard Lowe

Rules To Problem Solving

Richard Lowe
articles@internet-tips.net

Internet Tips And Secrets <http://www.internet-tips.net>

Copyright (C) Richard Lowe Jr. and Claudia Arevalo-Lowe, 1999-2001.
Permission is granted to reprint the following article as long as no changes are made and the byline, copyright information, and the resource box is included. Please let me know if you use this article by sending an email to articles@internet-tips.net

Article Title: Rules To Problem Solving
Author: Richard Lowe, Jr.
Contact Author: articles@internet-tips.net
Publishing Guidelines: May be freely published w/bylines
Web Address: <http://www.internet-tips.net>
Autoresponder Address: article-211@internet-tips.net

I've been working in the world of computers for 23 years, and I've learned a lot about problems during that time. I've found a few rules which, if followed, make it easier to find, understand, correct and verify problems.

Rule #1: Don't assume you understand the problem. This is one of the classic mistakes of problem solving - you think you understand what's going on, but you didn't look deep enough or get enough information to really get it. Before starting to solve any problem, be sure you spend some time and be absolutely sure you understand exactly what's going in.

Rule #2: Don't assume that the person who reported the problem understands the problem either. In the computer field, I've found that users will report problems in many different, often bizarre ways. Sometimes they will describe it in such a manner that it appears to make sense, but actually what they are describing has no relation to the problem at all. Remember, most people do not understand computers and the related technology at all, so they tend to piece together descriptions based upon what they have heard, what they think they know and what people have told them.

Rule #3: Duplicate the problem. Always, always, always duplicate any problem before you start working on finding a solution. Why? See Rule #4. In addition, if you can make a problem occur again, there is a much better chance that you really do understand what's going on (rule #1).

Rule #4: You cannot know you have solved a problem unless you followed Rule #3. The only way to be sure that a problem is solved is to fix it, then exactly replicate what happened. The sequence is simple: duplicate the problem, fix the problem, then try and duplicate it again. If you've exactly

replicated the issue, then you can be reasonably sure you've fixed it.

Rule #5: Don't assume someone else understands the problem. If you need to delegate the problem to another person, or if you are receiving instructions from another person to solve the problem yourself, do not ever assume they understand what they are talking about. Always follow Rule #3 to be sure YOU understand the problem. Do not take anyone else's word for it. If you delegate the problem, make sure the person you give it to follows Rule #3.

Rule #6: Don't assume you have just one problem. Sometimes things are more complicated than they seem. It's never a good idea to assume that there is just one problem to be solved. Throughout the entire problem solving process, keep your eyes open and find any additional problems that you may see.

Rule #7: Don't assume there is more than one problem. Also, don't make the assumption there is more than one problem either. How do you follow rules #6 and #7? Just base your conclusions upon what exists, not upon your assumptions or what others have told you.

Rule #8: Don't assume there is a problem at all. Just because someone reports a problem does not mean there is actually a real problem. I remember when I got very upset because my car was making a strange noise. I brought it to the mechanic and had him spend hours checking my car to fix the noise. As it turned out, the noise was normal and was not a problem. Hours wasted when there was no problem at all. If the mechanic had followed Rules #2, #3 and #8, I would have been out of the shop in a few minutes.

Rule #9: Don't assume you don't have a problem either. Again, don't make assumptions. Base your conclusions upon what exists, not what you assume to exist.

Rule #10: Don't assume the problem is the same as an earlier problem. I manage a number of computer systems. One of the functions of these systems is to fax several thousand purchase orders to vendors over night. One day someone reported that they could not see any failures, and it's unheard of for no faxes to fail. I assumed, mistakenly, that this was a failure in the report, which had happened before. Thus I put the incorrect priority on the issue and didn't look at it until the afternoon. When I looked, I discovered to my horror that ALL faxes had failed (which caused the failure list to fail also, as it made an assumption that at least ONE fax would work). This caused incredible grief which could have been avoided had I actually looked instead of making an assumption.

Rule #11: Don't assume it's a computer error. Not all problems are caused by machines. You could spend countless hours trying to fix something that was actually a data entry error or had some other human cause.

Rule #12: Don't assume it's not a computer error. By now you should thoroughly understand this. Don't make assumptions. Look and form your conclusions based upon the evidence that exists.

Rule #13: Don't trust the documentation. Use technical documentation as a resource, but do not assume it is correct. Programmers are notorious for allowing their documentation to slip into uselessness. That's just the way the world is, so don't beat your head over it. Read any documents you can get your hands on, but also look at the code and anything else pertinent.

Rule #14: Don't assume it ever worked. Many years ago, I had the assignment to convert a plotting package from one computer system to another. It appeared to be a simple project (I violated Rule #15) so we just moved the code to the new machine and tried to run it. Several errors occurred (squares not square and triangles not triangular), and these did not occur on the original machine.

We spent months (literally!) trying to figure out what we did wrong. As it turned out, we violated rule #14. The code was in the middle of being modified, and the programmer who was doing the modifications quit and didn't tell anyone. Thus, the code we were using never worked, and thus, well, we didn't do anything wrong. Once we had the proper code (from an old backup) it really was very simple.

Rule #15: Don't assume it's simple or complex. Just remember it is what it is. Some problems are simple and some are complex. Don't assume either until you have done your analysis.

Rule #16: Don't assume maliciousness. If you find a human error, don't assume it was malicious. Generally, human errors are the result of incompetence - the person did not understand what he or she was doing. Start with training to correct human errors - you can move to harsher methods later if training doesn't work.

I hope these rules are of value to you in your problem solving endeavors.

List of articles available for reprint: <mailto:article-list@internet-tips.net>

NOTE: The following information must be included if you reprint this article:

Richard Lowe Jr. is the webmaster of Internet Tips And Secrets at <http://www.internet-tips.net> - Visit our website any time to read over 1,000 complete FREE articles about how to improve your internet profits, enjoyment and knowledge.

[Get-Articles.com : 1000's of reprintable business and internet marketing-related articles.](#)

[Submit your article for reprint.](#)