codeproject.com

# The Beginner Programmer's guide to Problem Solving [With Example]

*Rajaraman Raghuraman*

11-14 minutes

---

Have you got this feeling that you are able to grasp the concepts of programming and you are able to understand what's a variable, what's a function, what are data types, etc. yet you find it difficult to solve problems in programming. Every beginner gets this feeling. I did too when starting out.

It is important to overcome this feeling at the earliest, otherwise it can form a mental block for you.


How it can be a mental block to you? Common sense says that the more you practice a certain skill, you get better at that skill as time progresses. Same goes with problem solving too. The more problems you solve, the better you become at problem solving. But when you get a feel that you are trying hard and still unable to solve a problem or find it extremely difficult, your confidence lowers. At this stage, either you stop solving problems or try to solve lesser number of problems.

The point is your curriculum or your professional work is generally designed in such a manner that the order of difficulty increases as time progresses. So, you are in a situation where you feel less confident in solving small problems but now tasked with solving bigger problems. And the cycle continues till it becomes a permanent mental block in you.

**Is it too late to start solving problems?**

No. If you have come to the realization that you need to improve your problem solving skills, you have made that good first step. Quite often our egos don't let us accept the obvious. It is

good to accept certain truth because that is the only way that we can improve ourselves.

**What can I do to become better at solving problems?**

**Remove the mental block first – Exercise your mind**

Your mind is your most powerful weapon. So you have to think you can actually solve the problem. So from today, think positively that you can solve any problem. But you will obviously start with small problems and go on to solve bigger problems.

As with every aspect in life, it starts with conditioning the mind. So, starting today, tell yourselves the following:

1. I can solve any problem that is put at me
2. I will commit at least 1-2 hours per day on solving problems alone for the next 30 days
3. I will never give up on any problem that is put at me, I will ask for help if required.1

**Understand the basic approach to problem solving**

Do you know one of the reasons for your struggle with problem solving? One reason might be due to lack of practice. But the main reason is because you have not understood the basics of problem solving especially in programming. Once you understand the approach to problem solving to the smallest of things, you can go ahead and solve bigger and more complex problems with confidence.1

Ever wondered how top tech companies like Google, Amazon solved the internet's biggest & hardest problems? The answer is simplicity. They solved problems at the basic level and then went on to solve bigger and bigger problems. You can do it too. But you need to be good at the basics.

## What do I need to understand before even trying to solve the problem?

**Understand the problem clearly – The Power of clarity**

You need to understand your problem clearly before even trying to solve it1. Lack of clarity at this stage will put you down. So make a conscious effort in understanding the problem more clearly. Ask questions like What, Why, When, Where, What if and How. Not all questions might be applicable to your problem, but it is important to ask questions to yourself at this stage before you go ahead trying to solve the problem.

**Visualize – The Power of visualization**

I am sure everyone of you is aware of what visualization is. Trying to picturize your thoughts. Have you ever imagined how some people can solve extra ordinary problems just by looking into those problems and they will instantly have a solution to it? And we don't even understand the problem fully? It is because they do it with their mind. They visualize the problem in their minds and they solve it in their minds itself. Visualization is a powerful tool in your mind.

But in order to get to that state, first you need to visualize the problem externally. That is where a pen and a paper/notebook (or) a white board comes into play1. Try to visualize the problem at hand and try to picturize the problem. That is also one of the steps to make sure that you understand the problem clearly.

There was a situation when I and my dear friend & colleague were discussing about a problem and we were literally going nowhere. This was actually when we each had around 7 years of experience in the industry. At that point, my friend said “Let’s put our points in board. If we don’t put it on the board, we will never get started”. And we started putting things on board. Things started to get more clear and raised more questions and ultimately became more clear.

That is the power of visualization. It really helps us to get started with our thinking. This visual thing works. Just try it out.

Your next question might be “I kinda get it, but I don’t. How do I visualize? What exactly do I visualize?”. Please read on to find out the answers.

## What is the basic approach to problem solving

### Step 1: Identify small problems

The major trick in problem solving is to identify and solve the smallest problem and then moving ahead with bigger ones. So how do you do it?

The answer is division of responsibility. Simply put, we need to identify parts that can stand on its own and identify a sequence in those responsibilities. And once you start breaking down the problems into smaller ones, then you can go ahead with the next step.

### Step 2: Solve the smaller problems one at a time

Now that you have identified the smaller problems, try to solve them. While solving them, make sure that you are focussing only on one problem at a time. That makes life much simpler for us. If you feel that this smaller problem is too big to solve on its own, try to break it down further. You need to iterate steps 1 to step 3 for each smaller problem. But for now, ignore the bigger problem and solve the rest of the problems.

Quick Tips

- Focus on one problem at a time. While you are focussing on one single problem
- It is ok to assume that other problems are solved
- It is ok to hardcode when coding a particular problem, but later you will resolve it in step 3.
- Solve the easier problems first, that will give you confidence and momentum until you get the confidence to solve the hardest problem first.

### Step 3: Connect the dots (Integration)

You have solved individual problems. Now it is time to connect the dots by connecting the individual solution. Identify those steps which will make the solution or the program complete. Typically in programming, the dots are connected by passing data that is stored in variables.

### Step 4: Try to optimize each step & across steps

Once you are completed with a working solution, try to optimize the solution with the best code that you can write. This comes only with practice. This trick can make a difference between a good programmer and a great programmer. But to get to this step, you need to be first good at steps 1 to 3.

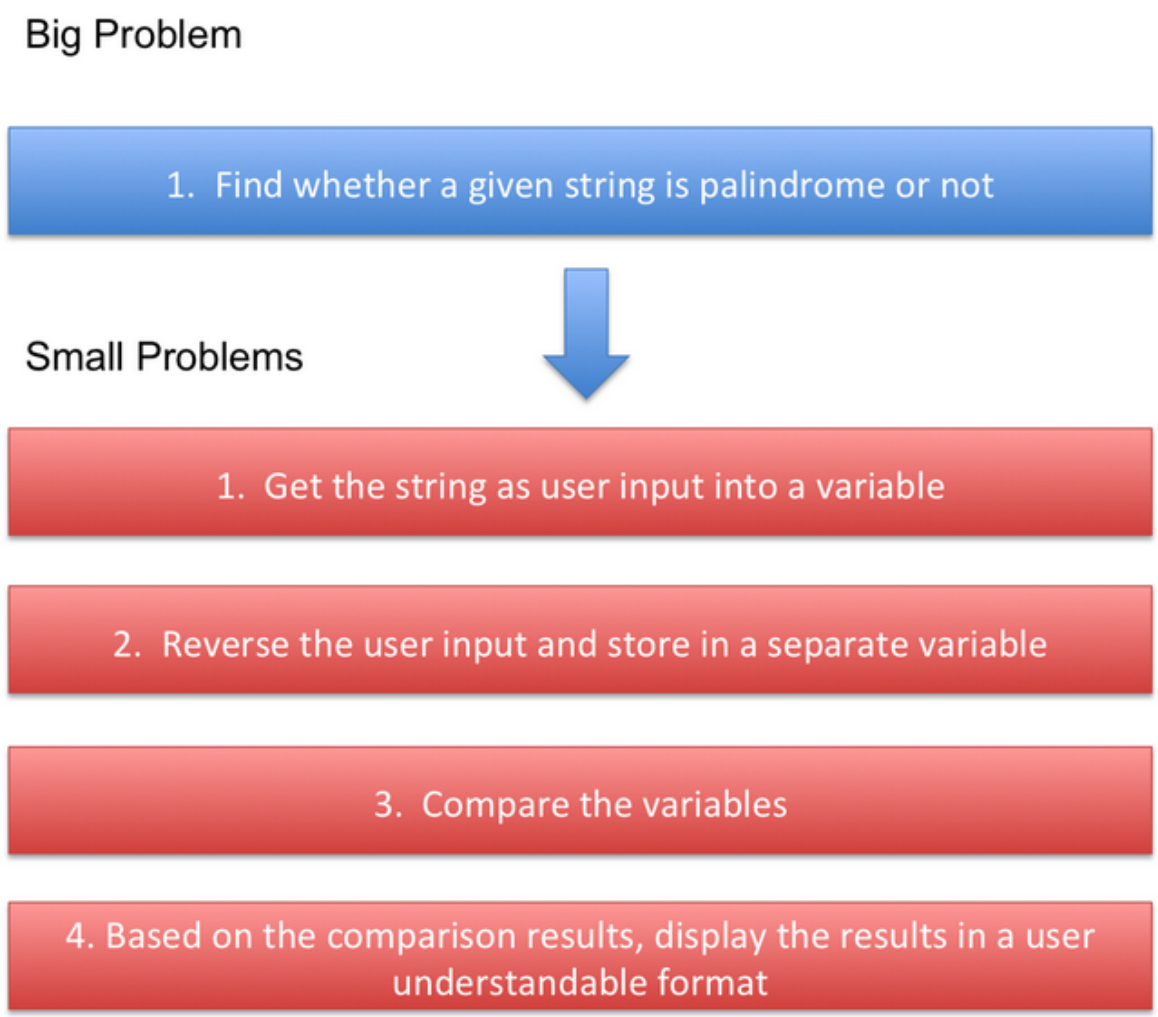
## Let’s take an example & walkthrough the problem solving approach

**Problem: Check if a user given string is a palindrome or not**

I will be using Python for this exercise (Although I have experience in1 C# and JAVA, I am also a Python beginner, so pardon any bad code). Let's iterate through our steps:

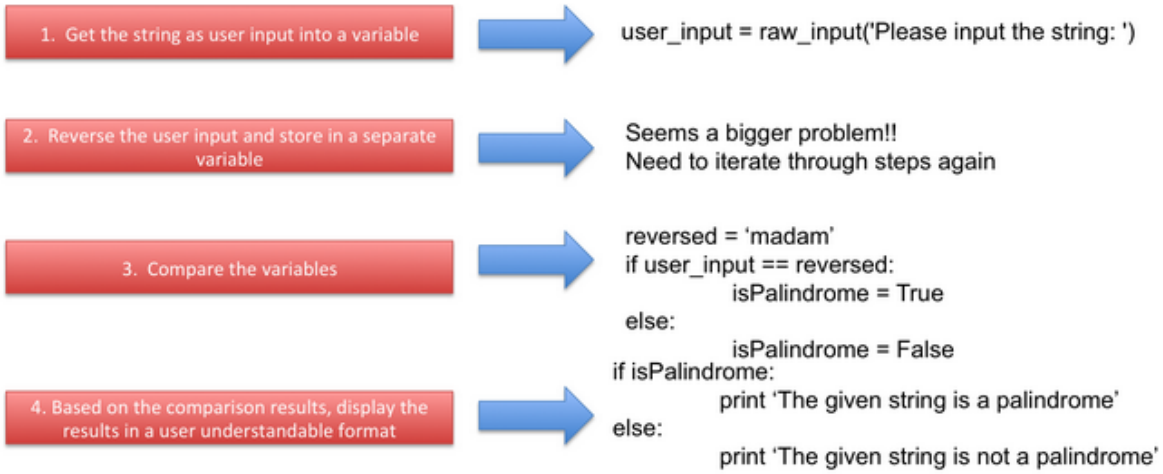
Let's call this as Level 1:

**Step 1: Identify smaller problems:**




**Step 2: Solve the small problems**

So each small problem will map to its corresponding solution as below:




Note: When solving the step (3. Compare the variables), I am doing 2 things:

- I am making an assumption that reversed is the variable name of the reversed string.
- I am hardcoding the variable name reversed to 'madam' to avoid compile time error
- If you execute the program at this state, you can input 'madam' and check if it is printing 'The given string is a palindrome' (And) you can input something else like 'dog' and check if it is printing 'The given string is not a palindrome'

**Step 3: Connect the dots (Integration)**

When we are trying to connect the dots, the only thing that is missing now is the variable reversed is hardcoded. For that to be set to the correct value, we need to break the small problem (Reverse the user input and store in a separate variable) into further smaller problems. Till that point we need to mark it as incomplete.

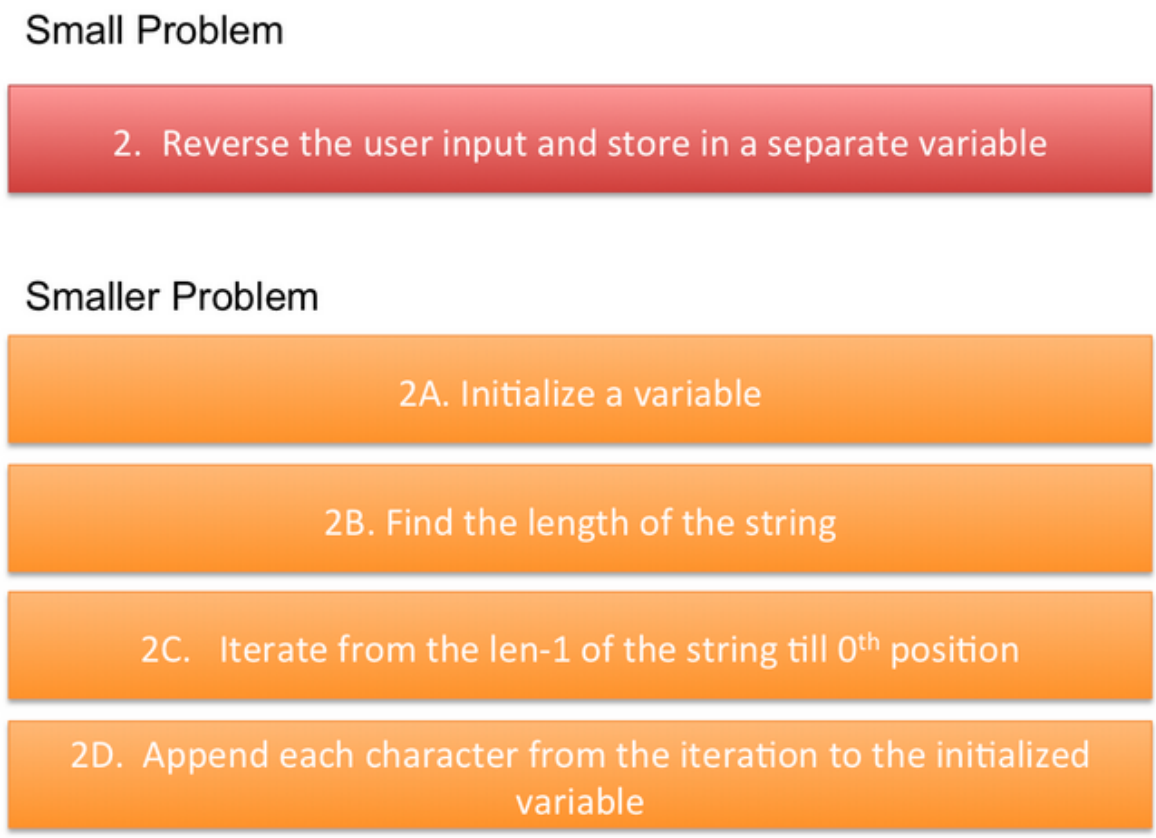
2 things still remain unsolved in Level 1:

- Solution for step 2 in the diagram (Reverse the user input and store in a separate variable)
- Connecting the dots once the solution for step 2 is found

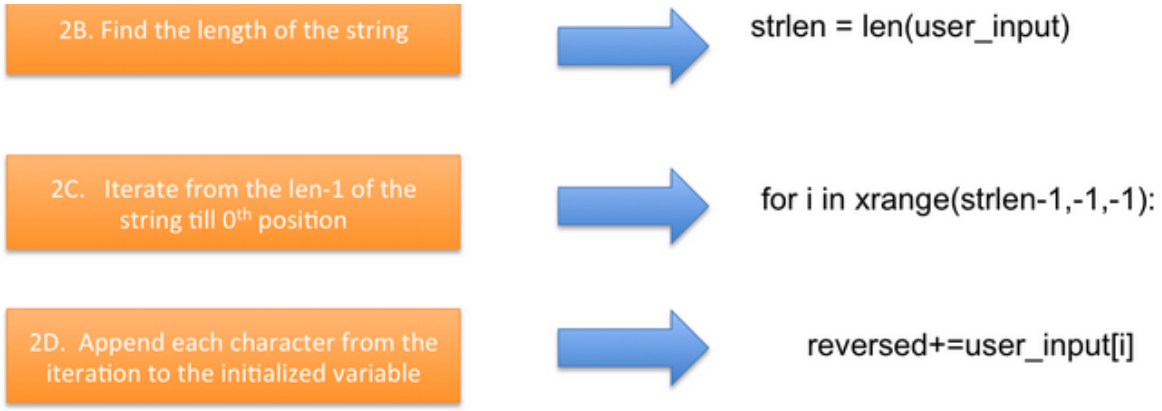
Iterating small problem 2 through our problem solving steps:

Let's call this Level 2:

**Step 1: Identify smaller problems**




**Step 2: Solve the small problems**




**Final code for this small problem will look like this:**

```
reversed = ''
strlen = len(user_input)

for i in xrange(strlen-1,-1,-1):
        reversed+=user_input[i]
```

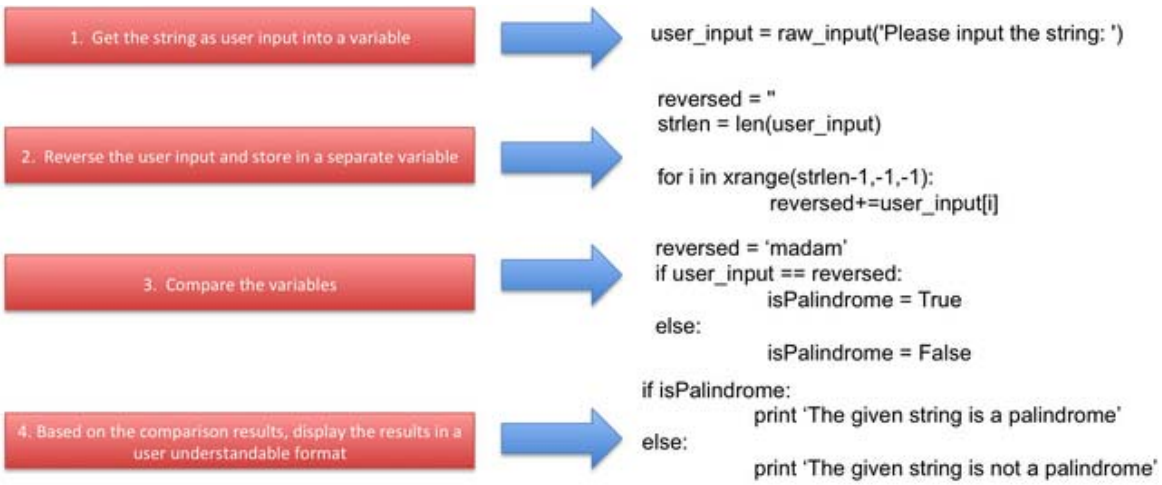
**Step 3: Connect the dots**

Here, we have already connected the dots. So we need not do anything extra in this step.

Now we have solved the smaller problems, which means Level 2 is over. Now we need to come back to Level 1.

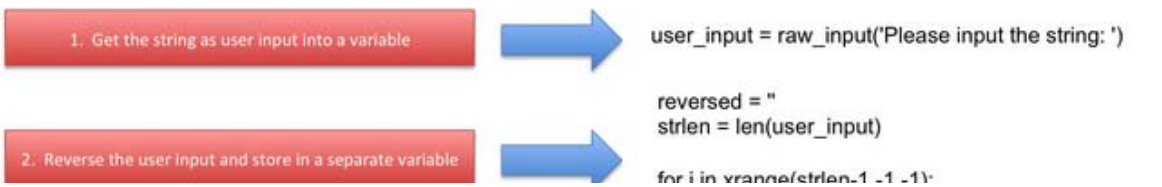
If you remember, 2 things remain in Level 1. One is solution for step 2 which we have found now. Two is connecting the dots.

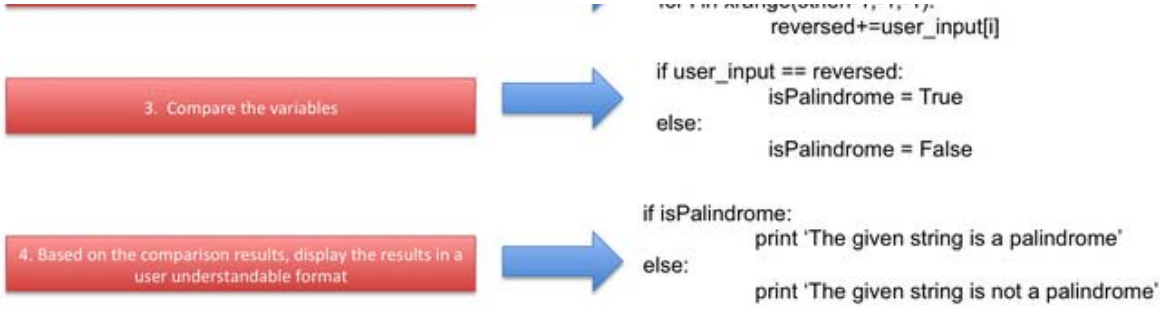
Now if we substitute the small problem 2 with the solution that we derived just now, we get something like this:




The thing that remains is connecting the dots.

So if we see what is the missing connection, the variable reversed is set twice. One to the solution of step 2 and another is hardcoded in step 3. So we can now remove the hardcoded value in step 3, in which case our code will become like this






If you see, we have actually solved our problem.

We are left with step 4 – Optimize each step and across steps

**Step 4: Try to optimize each step and across steps**

As you can see, there are many things that needs to be optimized for this code. I would leave you to optimize the code further. Come on, put on your thinking cap and try different solutions.

**BONUS STEP 5: Make the code robust**

By robust I mean,

1. Adding error & exception handling
2. Using better variable names
3. Adding user defined functions
4. Adding comments where necessary

Again, I would leave you to figure out how to do this step.

## Conclusion

- We saw just how we can solve problems using a step by step approach
- By solving smaller problems, I get into a momentum for solving bigger & tougher problems
- By focussing one problem at a time, I am eliminating distractions, thus allowing to better direct your efforts for that one problem rather than getting confused with many small problems at hand.
- If you understand this approach and practice, you will definitely go on to solve bigger problems and your confidence will raise.
- Beauty about breaking down the problem is that we can further convert each problem and sub problem into separate functions/modules thus making the code more modularized and maintainable.

Wait, You can't leave yet:

Now dear beginner programmers, take any problem and try to apply this approach. See the results for yourselves. Now, describe the following in the comments section:

- What problem you are solving?
- How did you break it down? (Even a snap of your notebook page or board will do!)
- The final code

- How did you feel and what did you learn from this exercise?

  Also remember, I am challenging you for the 30 day problem solving challenge.

  If you liked this blog post, please feel free to share it with your circles in social media.