HOW TO APPROACH ANY CODING PROBLEM?

Check out the full video on YouTube by
#TheLeanProgrammer

Check out this article on medium -

HOW TO APPROACH ANY CODING QUESTION
http://madhavbahl.tech/approach-question/
THE LEAN PROGRAMMER

1. UNDERSTAND AND ANALYSE THE PROBLEM

Read the problem, start thinking about it and if possible write the things that are given and the things that you need to find out on a piece of paper.

ASK YOURSELF -

1. Are you able to understand the question fully?
2. Would you be able to explain the question to a layman?
3. What and how many inputs are required?
4. What would be the output for those inputs
5. Can you separate out some modules or parts from the problem?
6. Do you have enough information? If not, try understanding the question again.

2. GO THROUGH THE SAMPLE INPUTS AND EXAMPLES THOROUGHLY

Going through some sample inputs and coming up with more examples sure helps you a lot to understand the problem well

START SMALL -

1. Take very simple examples and find the output
2. Take more complex and bigger inputs to see what will be the output, how many use cases do we want

THEN, HANDLE EDGE CASES

1. Try out the problem with no input, what should be the output now
2. Try out the problem with invalid input, what should be the output now

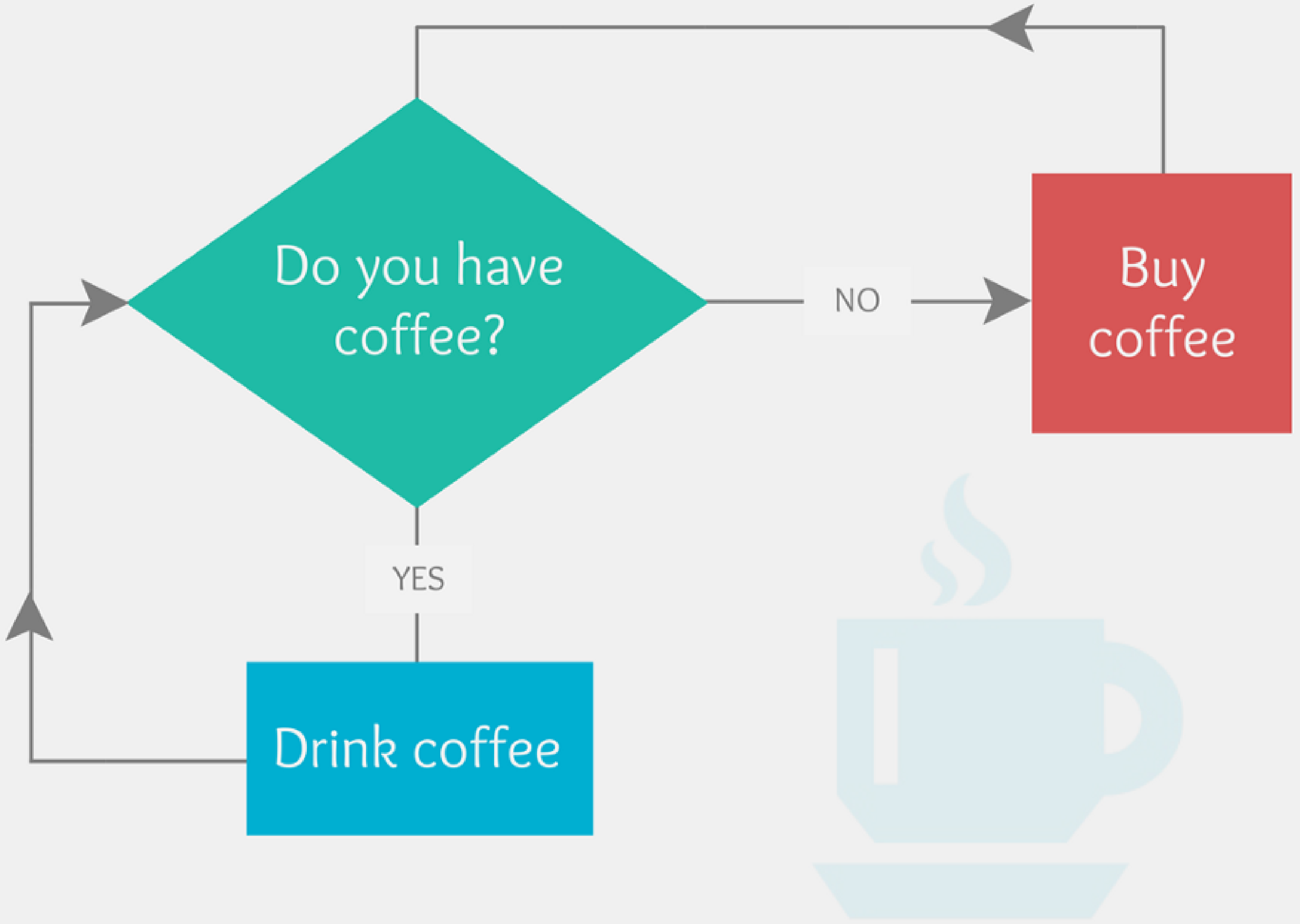
3. BREAK DOWN THE PROBLEM

This step is really very helpful when the problem at hand is very big.

TRY THE FOLLOWING -

1. Try to make a flow chart (or a UML, if it's too big to handle) for the problem in hand
2. Divide the problem into different modules or sub-problems
3. Try to make independent functions for each sub-problem
4. Connect those sub-problems by calling them in the required order, or as necessary
5. Try to use classes and objects while handling questions which try to implement some real-world problem (like management systems, etc.)

Should I Drink Coffee?



4. START SOLVING/CODING

Now that you've analyzed the problem it's time to actually write the code. If you can't solve the problem fully at once, try writing code for a simpler sub part, as you solve the problem, gradually you keep figuring out the problem in more detail, and you start getting new ideas.

ALWAYS KEEP THESE 3 THINGS IN MIND

1. The point where you started
2. Where are you right now?
3. What is your destination?

BONUS TIPS

Keep these in mind during the interview

- when you are giving some interview, do not waste time in figuring out the whole solution and then tell your interviewer
- keep simplifying the problem and keep telling your interviewer about how you are approaching the problem

1. Tell the interviewer how you are trying to start
2. Tell what method is there in your mind right now
3. Try solving the simple sub-parts first
4. You can start with brute force, there's nothing wrong with that!
5. Most likely, your interviewer will keep giving you hints on how can you improve
6. Use those hints to come up with a better solution
7. Keep talking to the interviewer

5. LOOK BACK AND LEARN MORE

Don’t just give up when you are done, give up when you feel content with the possible solution(s) and you’ve explored the problem completely!

ASK YOURSELF -

1. Does this code run for every possible input (including the edge cases)
2. Is there any other way to solve the problem?
3. Is the code efficient? Can it be more efficient?
4. Is the code readable?
5. If someone else showed you this code, would you be able to understand it?
6. Can the performance be improved?
7. Can some other algorithm be used which provides better results?

ALL THE BEST!