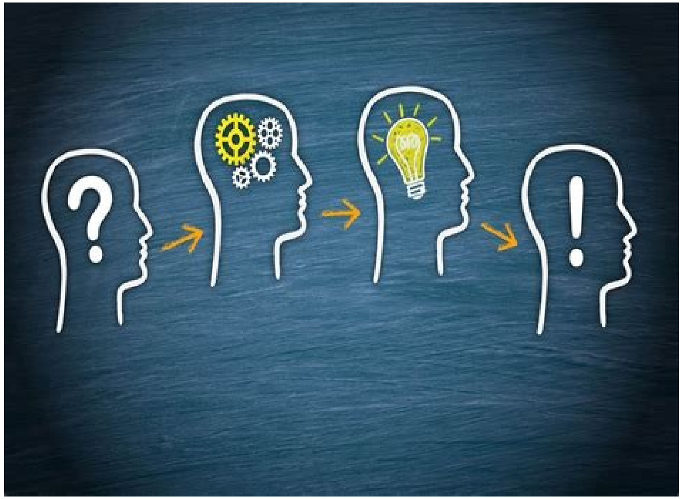
Sommario

- What is problem solving?
- Why it is important?
- Few steps
  - Understand the problem
  - Sample data
  - Pseudocode
  - Code
  - Debug or Optimization

Problem solving?

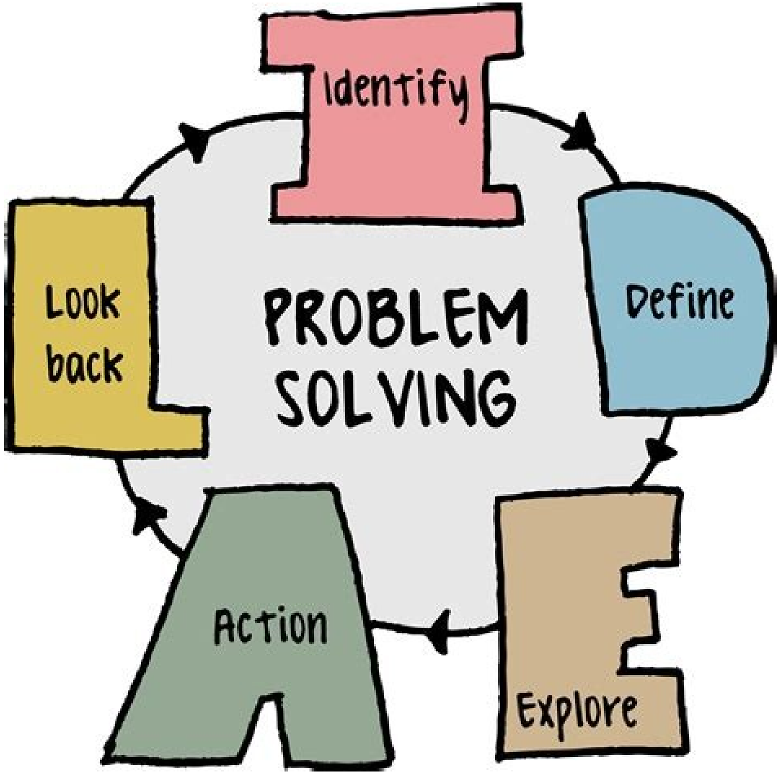
- What is?
  - Transform the description of a problem into a solution!
- Why it is important?
  - Engineers use problem solving skills, **constantly**
    - Not computer programming only!
  - Direct impact on the pace of your professional progress and career growth



What is the right approach?

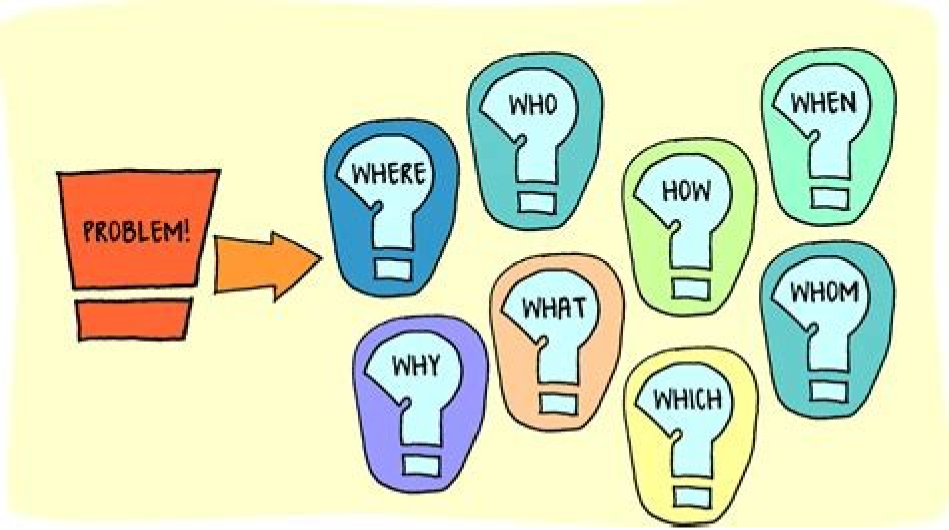
UNIVERSITÀ DI PARMA

- Do not rush in coding!
  - If you have a time limit this can appear captivating
  - But it is wrong!
- 4 steps to follow in the next slides



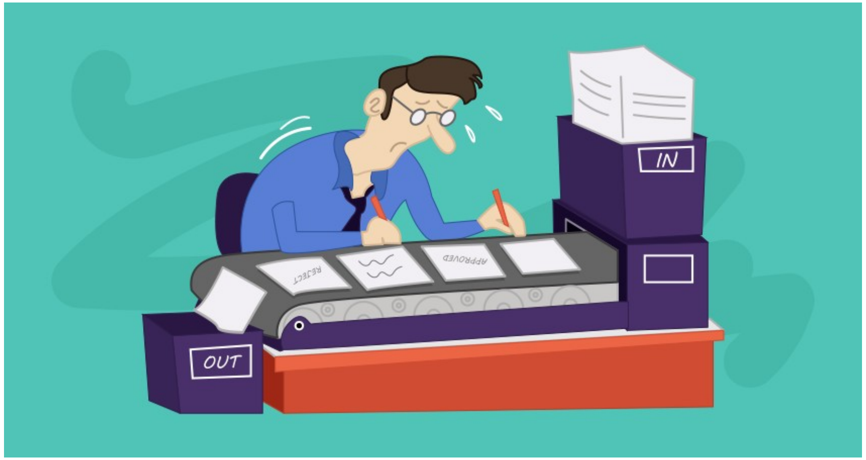
1. Understand the problem: read

- Again, do not rush in coding!
- You can not solve a problem you do not understand
- Read through the problem
- Clarify any part
  - Maybe ask or discuss!
- Once you have read
  - Read it again
  - Up to three times...



1. Understand the problem: simulate

- Work through the problem manually
  - What are the inputs?
  - What are the outputs?
- Consider sample inputs
- Try to analyze the output
  - Initially simple inputs
  - Then add complexity
  - Also consider corner/edge cases



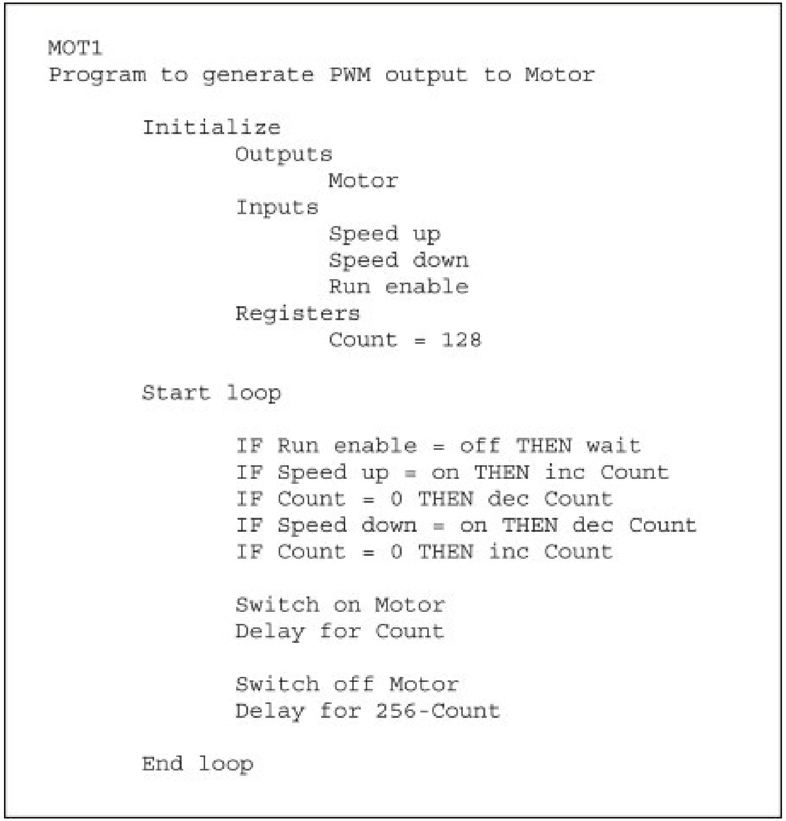
2. Devise a plan for solving the problem: top-down approach

UNIVERSITÀ DI PARMA

- Step a step plan to solve the problem
- Think about how you would solve it as a human
- Break down the problem
  - Small sub-problems or chunks
  - Solve them one by one
  - Connect the solution of each sub problem as input for others
- Edge and corner problems are dealt by the proposed solution?

2. Devise a plan for solving the problem: pseudocode

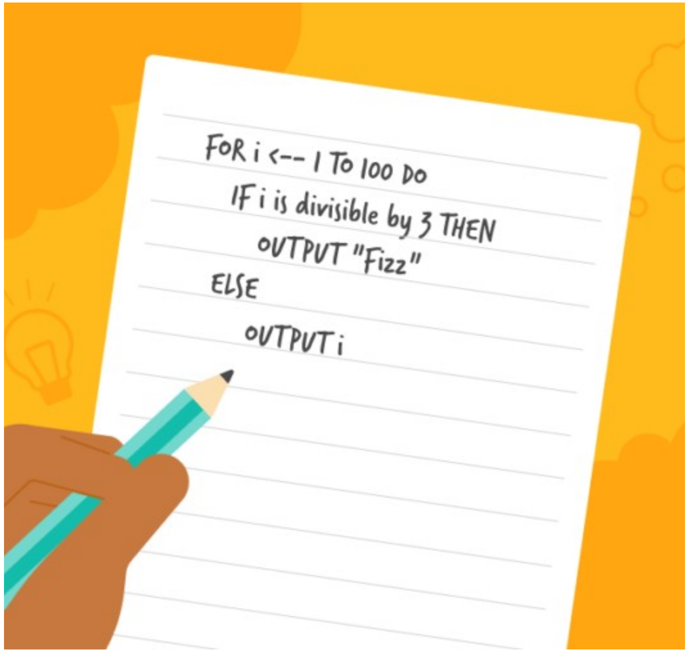
- Initially write down your steps in natural language
- Then translate your steps in pseudocode
  - Having final language in mind can help



```
MOT1
Program to generate PWM output to Motor

        Initialize
                Outputs
                        Motor
                Inputs
                        Speed up
                        Speed down
                        Run enable
                Registers
                        Count = 128

        Start loop

                IF Run enable = off THEN wait
                IF Speed up = on THEN inc Count
                IF Count = 0 THEN dec Count
                IF Speed down = on THEN dec Count
                IF Count = 0 THEN inc Count

                Switch on Motor
                Delay for Count

                Switch off Motor
                Delay for 256-Count

        End loop
```

3. Carry out the plan: write actual code

- Translate the pseudocode or algorithm description in programming language
  - Eventually you can code!
- Do not optimize code too early
  - Not required for us...



```
FOR i <-- 1 To 100 DO
   IF i is divisible by 3 THEN
      OUTPUT "Fizz"
 ELSE
      OUTPUT i
```

3. Carry out the plan: debug

- For sure the initial release will not work
- Debug
  - Look at compiler warnings
  - Add some print
  - Again, independently debug each part

4. Look back over what you've done: refactoring

- Not necessary in our case
- Reflect on your solution and, maybe, simplify your steps
  - Are improvements possible?
  - What other approaches are possible?
  - Is there anything you can generalize?