WHAT IS A PROBLEM STATEMENT?

A problem statement is a clear description of a challenge or goal that learners need to solve. It presents a situation that requires computational thinking and/or programming to resolve.

A problem statement tells learners what needs to happen, but not exactly how to make it happen. Learners must figure out the solution themselves.

When problem statements are well-crafted, learners stop seeing lessons as tasks to complete and start seeing themselves as capable problem-solvers who can help others - a powerful foundation for lifelong learning in computer science and beyond.

In Coding and Robotics, we solve problems!

1 The Structure of a Good Problem Statement - Essential Elements:

1. **A Clear Goal/Challenge**
   What needs to be achieved?
2. **Context or Scenario** (when appropriate)
   Why does this matter? Who needs this solution?
3. **Constraints or Rules**
   What are the limitations or requirements?
4. **Success Criteria** (sometimes implicit)
   How will we know the problem is solved?

WHY USE PROBLEM STATEMENTS?

1. Develops Computational Thinking
Problem statements require learners to:

- **Decompose** (break the problem into smaller steps)
- **Recognise patterns** (see **what** repeats or is similar)
- **Abstract** (focus on important information)
- **Create algorithms** (design step-**by**-step solutions)
- **Debug** (test and fix when it doesn't work)

2. Promotes Active Learning
When faced with a problem, learners must:

- Think **critically**
- Make **decisions**
- Try different **approaches**
- Learn from **mistakes**

This is much more powerful than passively following instructions.

3. Builds Problem-Solving Skills
Real programmers and engineers don't get told exactly what to do - they're given problems to solve. Problem statements prepare learners for authentic problem-solving.

4. Encourages Multiple Solutions
Good problem statements often have more than one correct solution. This:

- Values different **thinking** approaches
- Promotes **creativity**
- Enables **differentiation** (some solutions are simpler, others more complex)
- Creates opportunities for rich discussion: "How did you solve it? How is yours different from mine?"

5. Makes Learning Meaningful
Problems give **context** and **purpose** to coding and robotics. Instead of learning "what is a loop," learners think "I need to make this action repeat - how can I do that?"

6. Aligns with Real-World Application
In real life, we don't follow recipes - we solve problems. Problem statements mirror how coding and robotics are actually used in the world.

What Makes a Good Problem Statement for Foundation Phase?
A high-quality problem statement for Foundation Phase should:

- **Be age-appropriate**: Use simple language and familiar contexts
- **Create empathy**: Feature relatable characters or situations
- **Be specific**: Clearly define what needs to be accomplished
- **Motivate learning**: Create genuine interest in solving the problem
- **Connect to real life**: Use scenarios children can relate to
- **Support learning objectives**: Directly address the competencies being taught
- **Be scaffolded**: Provide appropriate challenge without overwhelming
- **Encourage agency**: Make learners feel they can help/solve the problem

PEDAGOGICAL OPPORTUNITY

In Grade 2, problem statements in lesson plans can occasionally be used as an opportunity to teach or guide learners to develop aspects of computational thinking, such as abstraction and even pattern recognition.

For example (***abstraction***), when a problem statement has unnecessary detail, then guide learners to identify what is essential.

Grade 2 Example: *"Lerato has a red robot. She got it for her birthday last year. Her robot needs to move forward 3 steps to pick up a blue ball. The ball is her favourite. It's a sunny day outside."*

Teacher guidance:

- "Let's find the information we NEED to program the robot. What must we know?"
- Circle or highlight essential details together
- Cross out unnecessary information

For example (***pattern recognition***) through ‘same or different’)), help learners see that abstraction involves recognising what matters and what does not.

Grade 2 Example: Show three problem statements for three scenarios:

- A cat needs to cross the street
- A robot needs to cross the classroom
- A game character needs to cross the river

Ask: "What's the SAME in all these problems, even though the details are different?"
(Answer: They all involve moving from one side to another)

Why this works: Learners see that we can ignore surface details (cat/robot/character, street/classroom/river) and focus on the core pattern (crossing from A to B).

The "Story vs. Code" Comparison
Show learners how we translate rich contexts into simple instructions.

Grade 2/3 Process:

1. Read a story-based problem together
2. Ask: "If we told this to a robot, what would we say?"
3. Create two columns on the board:

Story Version	Robot Instructions
"The hungry caterpillar wants to eat the apple"	Move to apple, eat
"It's a red, juicy apple on a green leaf"	(not needed for coding!)

Grade 2 Problem Statements - Gradual Release: From Teacher Guidance to Independence
Teacher can follow the guidelines below to gradually release guidance and to become independent in terms of interpreting and identifying main ideas in problem statements (become problem solvers)

Grade 2 Progression:

- **Week 1-2**: Teacher models identifying main ideas out loud
- **Week 3-4**: Learners identify main ideas with teacher (whole class)
- **Week 5-6**: Pairs work together with teacher support
- **Week 7+**: Individuals try with peer checking

For Assessment:
Use problem statements to assess:

- Can learners identify what the problem is?
- Can they suggest appropriate solutions?
- Do they connect problem to learned concepts?
- Can they explain their problem-solving process?

Remember
Effective problem statements are the heart of inquiry-based learning. By improving problem statements to be more engaging, specific, and aligned with learning intentions, we create authentic reasons for foundation phase learners to engage with coding and robotics concepts.

CONCLUSION

Problem Statements

- Create emotional investment through relatable characters
- Provide clear, achievable goals
- Embed computational thinking language naturally
- Connect practical activities to theoretical concepts
- Respect learner agency and capability
- Build progressively in complexity
- Reflect real-world applications