Night sensor lesson 2

Curriculum links:

- **Computing:** Algorithms (pseudocode & flowcharts), decomposition, logical thinking, debugging
- **Science** Day and Night / Sensors, **Design & technology** Product design, **Citizenship** Road safety

Skills: Designing, analysing, problem solving, team working.

Resources: Teacher Guide: downloads: presentation, lesson plan, algorithms worksheet printed on A3 paper, large sheets of paper, coloured pens.

Background:
It is assumed that you have first completed Lesson 1 of the Night Sensor activity.

Introduction
This lesson focussed on the key concepts of decomposition and algorithms (pseudocode and flowcharts) as students begin to build their 'Night Sensor' example - a wearable device that will give an audio and visual reminder to a child to 'be safe, be seen' at nightfall.

Teacher guide:

Learning objectives

- To decompose a large problem into smaller, component parts
- To write a detailed, accurate algorithm using pseudocode and flowcharts and including selection
- To test and debug algorithms and understand why this is important

Agenda

- Introduction (10 minutes)
- Algorithm design (10 minutes)
- Team algorithms (15 minutes)
- Testing and refining algorithms (10 minutes)
- Flowcharts (10 minutes)
- Wrap up (5 minutes)

Introduction

- Invite students to share what they recall about the child road safety at night and recap some of their ideas from last lesson (slide 2).
- Explain you will be taking them through an example which may help them to refine their ideas, or think of new ones using micro:bit.
- Introduce the 'Night Sensor' (slide 3) and explain this is the goal/problem you will be focussing on in this lesson. Discuss why it will be particularly helpful for children with hearing or visual impairments who are more vulnerable on the roads.
- Highlight you will be focussing on the key Computational Thinking skills of decomposition and algorithms and share the learning objectives if you wish (slides 4 & 5).

Main activities

Algorithm design

- Show students the goal/problem again (slide 6) and depending on students' experience either as a class or in pairs write a basic, broad algorithm to solve the problem (example on slide 7). Ask questions to test understanding and encourage discussion such as:
    - will this be detailed enough for someone to code?
    - what would we need to do to make it detailed enough?
    - which parts are ambiguous and how would we address that?
- Highlight they have just decomposed the problem into smaller parts and explain they will now design more detailed, accurate algorithms for each part using pseudocode (slide 8).

Team algorithms

- Split the class into their teams and give out the algorithm worksheets (ideally on A3 paper).
- Explain you want them to write a detailed algorithm using pseudocode that someone could follow accurately to write the code for the Night Sensor. They may wish to split the team down to write different sections on rough paper before putting it all together. An simple example is given on slide 9 and a more advanced including a start and stop button and on and off sounds on slide 10.
- Depending on previous experience, you may need to spend time going through input and output devices (slide 11) and iteration, loops and selection (slide 12).

Testing and refining algorithms

- Pair up teams and ask them to swop algorithms, test them and give feedback to the other team, comparing and debugging as necessary.
- Come up with a final algorithm as a class, ensuring an appropriate level of detail and accuracy according to your students' experience and ability and addressing any misconceptions.
-

Flowcharts

- Invite students to share why creating a flowchart algorithm can also be helpful alongside or instead of pseudocode (graphical representation of algorithm, can be easier for a person to follow and to see decision points).
- Remind students if needed of the standard flowchart symbols (slide 13) and give each team a large sheet of paper and coloured pens. Ask them to create a flowchart for the algorithm and then share, test and debug as necessary as a class (example on slide 14 for the basic algorithm).

Lesson wrap up

- Explain next lesson they will use their algorithms to code the Night Sensor next lesson.
- Ask students a variety of questions to test their understanding (suggestions on slide 15).
- If you wish, use the learning objectives to check progress and understanding (slide 16).

Extension / homework

- Students can add to their learning log and/or working wall if using.
- Students could write pseudocode or flowchart algorithms to solve other everyday problems they encounter (e.g. getting out of bed, doing homework, deciding what film to watch, feeding a pet).

Differentiation

Support:

- Consider groupings carefully to ensure all students are able to participate.
- Encourage students to focus on writing a clear, simple algorithm and supply them with a print and cut out version of the basic algorithm to sequence into place.

Stretch & challenge:

- Students can be challenged to include additional detail and elements in their algorithms. For example, can they include the stop/start button and audio sounds straight away in their algorithm and flowchart?

Opportunities for assessment:

- Informal assessment of team algorithms (pseudocode and flowcharts) and use questioning to assess understanding and progress throughout and in lesson wrap up.