A Bag for Juliane Lesson 2

Curriculum links:
- **Computing**: computational thinking: algorithms, logical thinking, decomposition, abstraction, iteration, loops, selection, variables, testing, debugging, user needs, evaluation
- **Art and Design** drawing and craft, **Science** Day and Night / Sensors, **Design & technology** Product design, **Citizenship** Respect, road safety, communities, **PSHE** understanding others, mental health and wellbeing, **Geography**, understanding the world, sustainability

**Skills**: Empathy, designing, creative thinking, problem solving, prototyping, team working, presenting

**Resources**: Teacher Guide (presentation download, lesson plan download, worksheet download), large sheets of paper and coloured pens, rough paper.

**Background**: It is assumed that you have first completed the safety introductory lesson and A Bag for Juliane Lesson 1.

**Introduction**
In this lesson students create an annotated paper prototype for their bag for Juliane, including a detailed algorithm showing instructions of how it will work and feature labels to explain why they think it will help Juliane.

**Teacher guide:**
**Learning objectives**
- To create a paper prototype of a bag to help Juliane feel safe on her journey to and from school
- To take into account user requirements
- To write, test and debug detailed, accurate and efficient algorithms including iteration, selection and variables

**Agenda**
- Introduction (5 minutes)
- Paper prototyping (15 minutes)
- Algorithm design (15 minutes)
- Testing and refining algorithms (10 minutes)
- Showcase (10 minutes)
- Wrap up (5 minutes)

**Introduction**
- Give out students’ work from last lesson and invite them to recap their task and ideas (slide 2)
- Explain that you have had some information from Julianne’s school and there are some things they would like you to ensure are included in the bags (slide 3).
- Ask students to check their designs against this list and If necessary, discuss briefly how they will add these requirements to their design. Remind them they will be using micro:bit and as appropriate guide them to be creative about the kinds of things they could make (e.g. using additional LEDs / motors etc).
- Share the learning objectives if you wish on slide 4.

**Main activities**
Paper Prototype

- Introduce prototyping to students if necessary (slide 5) and explain they will create a paper prototype first, then prototype the computerised elements using the MakeCode editor.
- Give out large pieces of paper and pens and ensure students understand what they need to include on their paper prototype (slide 6). Invite them to sketch out and label their bag first, before moving on to the algorithm design.

Algorithm design

- Once students have completed their basic prototype, they can move on to writing the algorithm for the computerised elements using micro:bit. There is a sample algorithm on slide 7 showing a flashing light for Juliane’s bag that lights up when she turns it on, it is dark and she is moving. This uses micro:bit’s light sensor and accelerometer, iteration, selection and variables. If necessary, recap these concepts with students and encourage them to write efficient algorithms (slide 8).
- Ask students to write their initial algorithm on rough paper so they can test and debug it before transferring a final version to their paper prototype.

Testing and debugging algorithms

- Ask students to work with their peers to test and debug their algorithm, focussing on helping each other to write efficient algorithms and ensuring their bag designs meet the requests from Julien’s school.
- Once they have a final, accurate version, they can add this to their paper prototype, highlighting where they have used iteration, selection and variables to create an efficient algorithm.

Showcasing paper prototypes

- Invite students to stick their paper prototypes on the wall and present it to their peers, explaining their design, algorithm and how they worked out any tricky problems. If time allows, invite other students to give constructive feedback (e.g. WWW / EBI).

Lesson wrap up

- Ask students to think/pair/share why they have used iteration, selection and variables in their algorithms and how considering user needs is important.
- Explain that next lesson they will use their algorithms to create their micro:bit prototype.
- If you wish, revisit the learning objectives (slide 9).

Extension / homework

- Students could add their work today to an assessment portfolio and, if they have not used it before, explore the MakeCode editor ready for next lesson.

Differentiation

Support:

- Encourage students to focus on a simple algorithm using everyday language rather than pseudocode if necessary. You could also give out text instructions and ask them to sequence them into the correct order for their algorithm.

Stretch & challenge:

- Students can be challenged to create multiple, highly detailed, accurate algorithms for more advanced features they are including and making good use of iteration, selection and variables.

Opportunities for assessment:

- Informal assessment during showcase and questioning.
- More formal assessment if wished, of students’ paper prototypes, algorithms, focussing on the efficient and effective use of iteration, selection and variables.