Computing fundamentals
Lesson 1: Computational thinking 1

Introduction
In this lesson students are introduced to Computational Thinking through a team giant paper aeroplane challenge. They create and test a giant paper aeroplane and create instructions (an algorithm) for someone else to follow.

You will need
Lesson plan, presentation, large sheets of strong poster paper and normal paper, clear tape, coloured marker pens, stickers (optional), tape measure (optional)

Learning objectives
- To understand what 'computational thinking' is
- To develop computational thinking skills
- To write an accurate algorithm

Lesson summary approx 60 minutes
1. Team giant aeroplane challenge (15 minutes)
2. Flying time (10 minutes)
3. Writing instructions (10 minutes)
4. Sharing instructions (5 minutes)
5. Introducing computational thinking (15 minutes)
6. Review & wrap up (5 minutes)

1. Introduction: Team giant paper aeroplane challenge (15 minutes)
1. Introduce students to the challenge (**slide 2**)
2. Split the students into teams and give out large pieces of poster paper, a roll of tape, coloured marker pens and any stickers
3. Set a visible timer and give teams 10 minutes to create their paper aeroplanes (adjust the time to suit your students).
4. Students may want more instructions, however emphasise that you are looking for them to use their skills of problem solving, experimentation and debugging. If any groups really struggle, you could give them the first instructions from https://www.bbc.co.uk/newsround/26050831 as a starting point.

2. Testing & Flying (10 minutes)
Once the time is up, go to a suitable area to test the aeroplanes (awarding prizes if you wish!).

3. Writing instructions (10 minutes)
1. Explain that you want them to write instructions for someone else to follow to create a giant paper aeroplane **(slide 3)**
2. Give out large pieces of paper and ask teams to write the steps for creating the paper aeroplane (encourage creativity in how they do this, just emphasising that someone else should be able to follow it).
3. Remind them they may also need to make adjustments from their own aeroplane if this wasn't very successful!

Published by Micro:bit Educational Foundation microbit.org

4. Sharing instructions (5 minutes)

1. Give teams time to share and compare their instructions with other teams.
2. Discuss how they have each approached the challenge, discussing what they noticed from comparing their instructions (eg. were they more/less detailed, which were easiest to follow etc) and highlighting there are different ways to write the instructions to meet the same goal.

5. Introducing Computational thinking (15 minutes)

1. Introduce the term 'computational thinking' (**slides 4 & 5)**.
2. Ask students to share any knowledge and understanding they currently have around computational thinking and introduce the learning objectives if you wish **(slide 6).**
3. Use **slides 7 - 16** to explain each term to students (the first slide of each term), then asking them to consider in their teams how they used that skill (the second slide - click to reveal some examples for each concept).

6. Wrap up (5 minutes)

1. Give out prizes for the giant aeroplane challenge if appropriate.
2. Invite students to share their learning from today, informally assessing their knowledge and understanding of computational thinking concepts covered in the lesson, revisiting the learning objectives on **slide 17** if you wish.

Extension ideas

1. You could extend the flying section significantly to include collecting, analysing and presenting data using spreadsheets.
2. If your students have prior experience of designing algorithms, or you wish to extend the algorithmic understanding in this lesson, ask students to progress from using natural language in their algorithm to pseudocode and/or flowcharts.
3. Ask students to create an algorithm of any activity they do in their daily lives (e.g. getting dressed, walking their dog, playing their favourite game etc).

Differentiation

Support:

- Consider groupings carefully to enable all students to participate.
- Give students suitable support with their algorithms, encouraging them to use images and simple language if helpful, or giving them statements and/or images to sequence.

Stretch & challenge:

- Challenge students to create more detailed, accurate algorithms with less room for ambiguity.
- See also extension activities.

Opportunities for assessment:

- Informal assessment of students' algorithms and answers to questions.
- Informal observation of students' participation during the team activity and discussion.

Published by Micro:bit Educational Foundation microbit.org