PREPARING A BACKUP PLAN

Alan O'Donohoe describes a range of backup activities teachers can fall back on when lessons can't proceed as planned

In those incredibly stressful teaching moments when technology fails you, what's the best strategy? Do you choose to wait in the hope that you can quickly resolve the issue, dive into fixing the problem, or abort your plans altogether? A technological issue may last only a few seconds, but time has an annoying habit of dragging painfully slowly as you try to diagnose and fix the problem while maintaining a sense of calm in the classroom. The goal of this article is to expand the range of lesson backup strategies you can rely on when lessons can't proceed as planned. The guide describes some tried-and-tested activities you can make use of with minimum preparation and zero reliance on tech.

Readers are advised to trial these activities with teaching groups at the earliest opportunity — don't wait for things to go wrong before trying them out! This will give you a less stressful opportunity to judge how well they work for you and allow you to refine and develop them. You may discover a particular favourite which you keep up your sleeve, ready to implement at

ELEVEN

This activity is loosely based on a game called Nim. Like Hack Attack (see page 86), you can play it anytime and anywhere, and it requires no resources. It's a problem-solving game that requires participants to apply logical deduction and pattern recognition. It is suitable for two or more players. Each individual round lasts about two minutes. As multiple rounds are played, players can develop and implement strategies.

Goal: players need to avoid being the player who is forced to say 'eleven'.

Instructions: all players stand up until they are eliminated from play, and the last player standing is the winner. The moderator points to the first player, who plays their move, followed by each subsequent player in turn. The first player can say up to three numbers in sequence, starting with number one — for example, 'one' or 'one, two, three'. The next player continues the sequence — for example, 'two' or 'four, five'. On their turn, any player can say up to three numbers at a time, but they must avoid reaching the number eleven. Players whose turn it is to say 'eleven' must do so and must sit down after being eliminated, at which point the next round begins again at the start of the sequence. In the final round, the two remaining players battle it out to see who the champion will be.

Variations: the teacher can introduce many variations into the game, and can also invite the class to predict likely outcomes. For example, you can divide the class into teams. In a two-team game, each team member takes their turn against a member of the opposing team. Teams can either score points for each win, or team members can be eliminated when they lose, until all the team members are out.

As learners quickly become deeply engaged in the game, they will need reminding to apply their skills, such as pattern recognition, logic, and prediction. There are other learning connections that can be made in the game, including abstraction, decomposition, algorithms, and flow control.

The advantage of this game, compared to Hack Attack, is that it's a fast-moving game that can be played in a few minutes. The rules are very easy to pick up, and students may also have already played something similar before.

ONE FALSE, TWO TRUE

This activity requires a bit of prior planning, but like the others listed, requires no additional resources and provides plenty of practical opportunities for students to apply computational thinking skills to an unplugged scenario.

Goal: students are challenged to identify which presented fact is false.

Instructions: the teacher starts by informing the class that they will present three brief facts about themselves, only two of which are true. For example, 'I used to work for GCHQ, I appeared on BBC News, and I appear in a famous rock star's autobiography.' Students are then asked to collaborate in small groups to agree which fact might be false and propose some questions for the teacher to answer. The teacher will take one question from each group. When answering the questions, warn students you will answer truthfully, but that you might avoid answering certain questions, particularly if you are asked, 'Which one is false?'

You should remind the class to approach this challenge from a computer science perspective in order to establish facts, for example by applying logical reasoning and abstraction.

When I play this game, I encourage students to demonstrate acute listening skills during the questioning, as well as computational thinking skills, and I praise them when they do. Sometimes, students are so focused on asking their own question that they don't hear the same question being asked by another student and answered. Some students will ask interesting questions that are not relevant (such as 'How old are you?'), in which case they need reminding of the goal of the activity!

It's important to reveal at the end which of the facts was false and, if time allows, to provide some feedback on the types of question the class asked. You can then ask students to devise their own three facts so that they can practise on each other. Discuss strategies with students, for example how they might best include two unbelievable truths as well as a plausible falsehood.

TILE ALGORITHMS

Goal: learners develop, test, and evaluate a turtle-style algorithm to be used in their classroom.

Instructions:

Design: ask all students in the class, without physically moving, to choose a start position for themselves in the room. They could, for example, identify either a floor carpet tile or a ceiling tile, but a piece of paper will also work. Then ask them to develop an algorithm, using only two movement controls, that will describe a path from their current position to the exit door.

They can use both of the following movement controls:

- FORWARD n, where 'n' is the number of tiles to move. FORWARD 2, for example, moves two tiles in the direction the player is facing.
- CLOCKWISE n, where 'n' represents one 90-degree clockwise rotation. CLOCKWISE 4, for example, rotates the player completely so that they are facing in the same direction as they started.

Testing: ask students to test their algorithms. It can be a little chaotic if you let them all test their algorithms at the same time. Instead, ask one student to follow instructions while another reads out the algorithm.

Prediction: ask students to predict what would happen if a student seated in another part of the class followed their algorithm. Inevitably, they only succeed if they start at the algorithm designer's origin.

Extension: ask students to work in groups. Explain that adding one more movement control to their algorithm could mean they could navigate from any position in the room to the exit. Students need to decide what the additional control is, and then develop and test an algorithm that uses it. The teacher may need to provide additional support by suggesting a control that can sense, such as collision detection. Remind students how, in addition to sequence, an algorithm might include conditionals and iteration. The additional control needed could be 'if touching object' or 'while not touching object'. This isn't completely flawless, as some room layouts could mean a person following the algorithm will be trapped in a corner forever!

>

HACK ATTACK

This activity is based on a modification of Werewolf, a game that has proved to be popular among the tech community (see **helloworld.cc/werewolfgame** for more on this). It lends itself to being played anytime and anywhere, with no special requirements for equipment. It's a deeply engaging problem-solving game that requires participants to apply computational concepts such as logical deduction and pattern recognition. It can be played with groups of different sizes, and each game can last 15 minutes or longer, depending on the number of players and how much discussion the moderator allows.

Goal: players collaborate to identify and then eliminate cyberthreat actors from their online community.

Instructions:

Night: at the start of the game, the teacher is the moderator and all other players take on the role of network users. The moderator instructs all users to log off the network and go to sleep, as it is night. Users must all close their eyes and put their heads down and listen attentively for suspicious-sounding activity. As users 'sleep', the moderator secretly nominates three players at random by gently tapping on their heads. They become cyberthreat actors (or hackers) who then wake up in silence and make themselves known to each other. Next, without speaking, they select one innocent user to be the victim of a coordinated cyberattack, which they communicate to the moderator before sleeping again.

Day: the moderator announces the arrival of daytime, and as all users wake from their sleep, the user who was targeted in the attack is identified to all as the victim. The moderator explains that the victim can no longer communicate with anyone, except to utter their final words, "I've been hacked." This player is out of the game now; they need to keep silent for the remainder of the game and are strictly forbidden from communicating with other players.

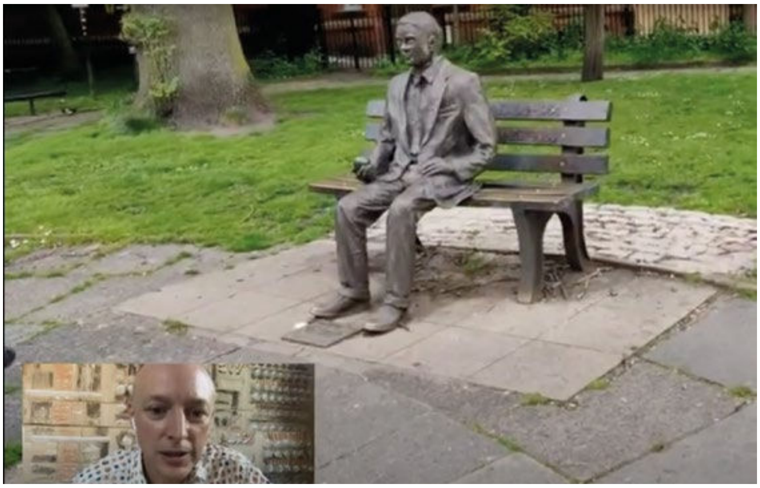
The moderator warns users that the hackers are cunning and deceptive. While they remain undetected, they pose a constant threat and will continue to target more innocent users. If challenged, they will protest their innocence and use lies and tricks to preserve their anonymity.

Discussion and elimination: users are urged to use logical reasoning and pattern recognition (computational thinking skills) to deduce which members of the group are likely to be the hackers. For example, focusing on pattern recognition, educators could prompt students to discuss questions such as: are any players behaving differently during the game? Which players appear to be most nervous? Has a player provided unnecessary detail in their answers? Have any night-time noises made you suspect a player? Users then nominate potential hackers and explain their reasoning. The moderator puts it to a vote and the player with the most votes is removed. The moderator explains: 'All their devices are confiscated and network connections terminated, meaning they can no longer communicate with others.' This player is now eliminated and their true identity is revealed. Unfortunately, this may result in an innocent user being eliminated, so users need to be careful who they accuse!

Night again: the process is repeated and the next innocent user is targeted by the hackers. If the users apply their computational thinking skills, they will eventually identify and eliminate all threat actors and their community will be safe. Learners will need reminding to apply their computational thinking skills. There are other learning connections that can be made in the game, including abstraction, decomposition, and algorithms. When the moderator is describing the activity, they should use terminology related to cybersecurity, such as coordinated, denial of service, traffic analysis, etc.

Extensions:

- You can introduce users with elevated roles and certain powers, such as ethical hackers (to whom moderators can secretly reveal the true identity of a single player during the night) and firewalls (which can secretly protect one player each night from an attack — either themselves or another player).
- Classes that are familiar with the game can propose changing some of the game mechanics and predict the outcomes. For example, you can increase the number of threat actors, or create additional roles. You could even ask learners to design, test, and evaluate an algorithm for an existing or a new player role. I have seen some students become so heavily invested in the game that they want to discuss their observations, theories, and hypotheses for many days after a game has stopped!



You can take students on virtual tours of places with computer science relevance, like a visit to the Alan Turing statue in Manchester, UK

Players collaborate in Hack Attack to eliminate cyberthreat actors from their community

the first sign that things are be about to go awry. It's likely that your favourite backup strategy will also become your learners' favourite — so watch out for children deliberately trying to sabotage the tech just so they can enjoy more of the same!

Some of the described activities have their origins in Victorian-style parlour games; I have modernised them and adapted them to help children develop computational thinking skills and apply computer science concepts to unplugged scenarios. All have been used successfully with children aged 5 to 18 years and above, some of which I'd prepared as a Plan B. (HW)

OTHER SUGGESTIONS

Deconstruction: challenge students to create an annotated diagram of a piece of technology or system to explain the key components and the connections between them (for example, a desktop PC, a mobile device, a classroom or school network, or the World Wide Web). Refer to examples of diagrams such as the London Underground map.

Textbooks: see the 'Emergency rations' section on page 91 of Hello World issue 17. The guide suggests an approach to using textbooks that could be used in case of emergency.

CSunplugged.org: this free website contains a wealth of activities that don't require technology. Note, though, that you will need the internet to access them!

Quiz: some teachers write their own quiz resources; others keep a book handy with some general knowledge questions they can quickly access when required.

A virtual tour: during the lockdown periods, I created some virtual tours to locations with links to computing (**helloworld.cc/exapeditions**). Each of these trips lasts 15 minutes, or longer if teachers use the questions included, and sites include Silicon Valley and the Alan Turing Memorial in Manchester. They do require an internet connection, video, and audio, so they may not be suitable for every tech failure.

ALAN O'DONOHOE

Alan has over 20 years' experience teaching and leading technology, ICT, and computing in schools in England. He runs **exa.foundation**, delivering professional development to engage digital makers, supporting computing teaching, and promoting the appropriate use of technology (**@teknoteacher**).