Offline coding activities

To support key aspects of Computational Thinking

Discovery EDUCATION™ | CODING™

Computational Thinking — key aspects
Offline activities

Computational Thinking lies at the heart of the new computing curriculum. Computing is not just about being able to write code, it's about teaching pupils to 'think like a coder'. There are five key aspects to Computational Thinking: logic, algorithms, decomposition, patterns, and abstraction — and all of these can be supported with offline activities.

Learn more about Computational Thinking:
http://www.discoveryeducation.co.uk/servlet/file/store66/item847933/Computational-Thinking_Jeannette-M-Wing.pdf

LOGIC

The ability to reason logically, and use what you already know to enable you to make predictions, is an essential skill in terms of being able to design and implement computer programs. The KS1 curriculum expects pupils to be able apply logical reasoning to predict the behaviour of simple programs, while at KS2 pupils should be taught to use it to explain how simple algorithms work and to detect and correct errors.

Making predictions (10 mins approx.)

Draw attention to the logical reasoning pupils use across the curriculum, such as in literacy; when they are predicting the actions or behaviour of a story character, or in science; when they predict the outcome of a science experiment. Encourage them to talk about the information they've used to draw their conclusion and make a distinction between this kind of reasoning and a straightforward guess.

Paired problems (10 minutes approx.)

Give pupils a logic problem to solve, such as the following:

Four children have different favourite meals.
Ben can't stand chicken.
Claire and David both like Italian food.
David is allergic to cheese.

Who prefers which meal?

	Chicken tacos	Spaghetti and meatballs	Cheese and tomato pizza	Sausages and mash
Anna				
Ben				
Claire				
David				

Pupils should work in pairs to solve it, so that they have to explain their reasoning. They could also annotate their finished grid, to show how they reached the conclusions they did. Similar problems can be found online (search for logic grid puzzles), or more able pupils could be challenged to invent their own. This would also work well with a Sudoku grid (again, there are plenty of examples online), with children working in pairs to solve and then explain their reasoning. Other puzzle types you could explore are Kakuro and Pixel puzzles.

Where are we going? (15 minutes approx.)

Give pupils a set of instructions for moving around the playground or classroom (take five steps forward, do a quarter turn clockwise, go backwards for three steps, etc.) Ask them to predict where they will end up after they follow the instructions. Repeat using a range of starting points. You could also try this with bee-bots or similar programmable toys, or pupils could write their own sets of instructions to give to each other and make predictions.

Codebreakers (15 minutes approx.)

Show pupils some code (on screen, or give them a printout) and ask them to 'read' it then predict what the program will do. Extend this by including a 'bug' or mistake, and seeing whether they can spot and correct it. This could be done using a programming environment they are familiar with (e.g. block coding in Espresso) or as a way of introducing them to other languages; for example, they could look at some HTML code and use it to predict what a web page would look like, or try to work out what some python commands mean by comparing code with what's displayed on screen when it runs.

ALGORITHMS

An algorithm is simply a set of steps that need to be followed in order to solve a problem. These aren't limited to computing, and there are plenty of opportunities for creating and exploring them in other subjects.

For example, pupils could create or follow algorithms when:

- working with recipes in Design and Technology
- using a step-by-step process to solve a particular type of maths problem (e.g. two digit by one digit multiplication)
- creating gymnastics or dance routines in PE
- explaining the steps in a science experiment
- writing instructions in literacy

These could be represented in a variety of ways: as lists, numbered diagrams, storyboards, or even videos or animations to explain the steps in a particular process.

Other activities to help develop an understanding of algorithms include:

Barrier games

Working in pairs, pupils sit back to back (or with some sort of barrier in place) so they can't see each other's work. One pupil, the 'sender', does a drawing (e.g. of a house, or a monster made of simple shapes) and then gives instructions to a 'receiver', who must replicate the same drawing simply by

following the instructions given. Squared paper could help children to describe size and positions more accurately.

You could link this in with positional language or names of 2D shapes in Maths, or simply use grid paper and colour in squares to make a particular design. As a variation, give each pupil a map and ask the 'sender' to mark some landmarks in, then explain to the 'receiver' what they are and where to put them.

Flow charts

Algorithms can be represented in different ways: as lists, drawings, or even (for more complex problems) with a flow chart. Show pupils some examples of flow charts and ask them to try and work out what each one represents. Ask them to create their own, e.g. they could draw a flow chart to explain how to play a favourite playground game.

You could even mark a simple flow chart out with chalk in the playground, and have pupils physically follow the lines and arrows.

Supporting resources: Example flow charts on **Pages 11-12**.
Description of the flow charts on **Page 10**.

Word search algorithm

Help pupils understand the difference between computer algorithms and more general instructions by giving children a word search puzzle to solve. When they have finished, ask them to share the strategies they used to find the words. Did they hunt randomly, or take a more systematic approach?

Ask them to consider what instructions they would give a computer to solve the same problem.

DECOMPOSITION

Decomposition — or converting a problem or task into a series of smaller ones — is a key element of designing an algorithm or program. For example, to be able to code a computer game, pupils will need to think about different elements (such as how the player will control the main character or object, the scoring system, enemies and any obstacles, such as walls), and then write code that controls each of these different elements.

Practising breaking complex tasks into smaller, manageable parts can help with this.

Off to school (15 minutes approx.)

Introduce young children to a character (puppet / soft toy) and explain that he / she needs help with getting ready for school. Tell them you want them to give him some instructions to help.

Ask children: what are some of the things you do before you come to school? As each suggestion is made (i.e. have breakfast, get dressed, brush your teeth), ask children to break these down into parts and give the correct instructions (e.g. if he needs to brush his teeth, we should tell him to: put the toothpaste on the brush, brush his teeth, rinse the brush). Concentrate on a small number of steps and simple language, using actions to help reinforce the steps. Chant the instructions and actions several times, before repeating with a different activity (e.g. getting dressed).

Dance sequence (15 minutes approx.)

Teach pupils a dance that has a series of steps, and then talk about how this can be broken down into parts, to make it easier to remember. Children could come up with different names for the different parts, or a different child could be in charge of learning a particular part and then teaching the rest of their group.

Getting organised (10 minutes approx.)

Set pupils the task of organising an event, e.g. a class party, and discuss different ways the workload could be broken up. Will they put one group in charge of food, and another on entertainment? How will they make sure they have a range of food — what if girls brought savoury dishes, and boys brought sweet ones? This could also be used to plan a class assembly with different sections, or a class trip.

What's going on? (20 minutes approx.)

Challenge older pupils to think about a complex, computerised system — such as a pedestrian crossing or, to extend more able pupils; the automatic checkout at a supermarket. Ask them to work in groups to discuss and record an algorithm for this process. Compare results, looking at the different ways that different groups decomposed the system, and how they recorded their algorithm.

Fermi problems (15 minutes approx.)

These are problems that initially sound impossible to answer, but a reasonable solution can be reached if they are broken down into a series of related questions whose answers can be estimated. A famous example is: 'How many piano tuners are there in Chicago?', which can be answered by estimating (or making assumptions) about the population of Chicago, how many households (on average) have a piano, how often pianos need to be tuned, how many can be tuned in a day, etc.

Set pupils a similar problem (e.g. how many hairs are there on a dog?; how many mobile phone calls are being made at this exact moment?), and ask them to think of ways it could be solved, by breaking it down into simpler questions.

Supporting resources: Additional Fermi problem examples on **Page 13**.
Supporting link: https://www.mathcircles.org/files/Fermi_Estimates_Lesson_Plan.pdf

PATTERNS

The ability to recognise patterns is important in computing, to help with efficiency. If programmers can spot similarities between different problems, or even between the steps of an algorithm, they can reuse the same solutions or pieces of code instead of always starting from scratch. Pupils might be surprised to learn that this is common practice. There are plenty of opportunities to explore pattern across the curriculum, and it will already be part of many subjects, particularly in Maths, Science and Art and design.

The following activities could also help:

Image sort (10 minutes approx.)

Give pupils a set of images to sort into groups. They could then swap with a partner, who should search for patterns and try to identify the categories their partner has used. A set of animal pictures could be sorted by size (e.g. bigger than a human, smaller than a human), type (e.g. mammals, insects, reptiles), domestic or wild, number of syllables in their name, whether they start with a letter in the first or second half of the alphabet, etc. Extend the activity by using a Venn diagram to sort the pictures into overlapping groups.

Going on a picnic (10 minutes approx.)

Play a game where children suggest food they might bring on a picnic. Decide on a rule (e.g., pupils must suggest food which starts with the same letter as their name) and challenge the children to guess it by telling them whether or not they will be able to come, depending on whether their suggested item fits the rule (e.g. Sally can come if she brings sausages, but Tim wants to bring bananas so he can't). You could vary this by only letting them come if their suggested food has the same number of syllables as their name, has a double letter somewhere in it, is not yellow, etc.

Spelling sort (15 minutes approx.)

Give pupils a set of words that use different spellings to make the same sound, or give them a silly sentence that contains the words and ask them to identify the words that contain the sound. Challenge

them to sort these into groups according to spelling pattern. Older pupils could do this with prefixes and suffixes; e.g. ant /ent (or –able /-ible).

Extend the activity by asking children to look for similarities within groups, to see whether they can make generalisations about when the different spellings are most likely to be used. (e.g. k is used for the 'c' sound before e, I and y.)

Appendix 1 in the National Curriculum for English contains (non-statutory) rules and guidance that could support this kind of activity.

Supporting resources: *Silly sentence* examples on **Page 14**.

ABSTRACTION

Abstraction involves removing unnecessary information. It helps us to solve a problem by focusing only on the relevant details. An example is the London Tube map, which shows the order of stations along the different lines, as well as the point at which they intersect, but doesn't include information about the distance between them or their relative locations.

Quick draw (10 minutes approx.)

Play a game where one pupil has to draw an object in the classroom as quickly as possible, while the rest of the class guesses what it is (as in the board game Pictionary). Give them a time limit to see how many drawings they can guess in one minute, or have teams race against each other. Afterwards, compare the drawings with the original objects and look at which details did / didn't need to be included. What is the simplest version that can still be recognised?

Give me a clue

Vary the quick draw game (above) by instead asking pupils think of an object they can see, and giving clues about what it is. Again, focus on giving the least detail (or the fewest words) possible to describe the object, and comparing different versions.

The Seven Bridges of Königsberg puzzle

This is a famous maths problem, which involves visiting every part of a town by crossing each of its seven bridges only once. Investigate the problem with pupils, looking at how it can be simplified by drawing a diagram and focusing only on relevant aspects. This could lead to further work investigating networks and Euler paths.

Supporting links: http://nrich.maths.org/2484
http://www.mathsisfun.com/activity/seven-bridges-konigsberg.html

EVALUATION

Evaluation is an important part of computing and involves checking that the suggested solutions are fit for purpose. Do they do what they are supposed to? Do they do it effectively? Do they do it efficiently?

These skills can be developed right across the curriculum by including opportunities for peer and self-assessment. You could ask pupils to compare end results against success criteria, make judgements about their own or others' work, and suggest improvements. Whatever strategies or structures you already use for this could be used to evaluate finished coding projects.

- *Two stars and a Wish* — Come up with two good points and one potential improvement.
- *What Went Well / Even Better If* — This could be used for both the finished product and also to help pupils reflect on how well they worked together, or the approach they took.
- *PMI* — What were the Positives, Minus points, and Interesting aspects of the solution?

ALGORITHMS: FLOW CHARTS ACTIVITY - DICE GAME DESCRIPTIONS

Flow charts on following pages.

3 or more – Two player game

The object of the game is to get 3 or more of a kind. The player with the highest score after a fixed number of rounds is the winner.

Roll 5 dice. You must have 2 of a kind to continue playing. If you don't your score for this round is zero, and the dice go to the other player.

If you rolled 3, 4 or 5 of a kind on that first roll, score as below:

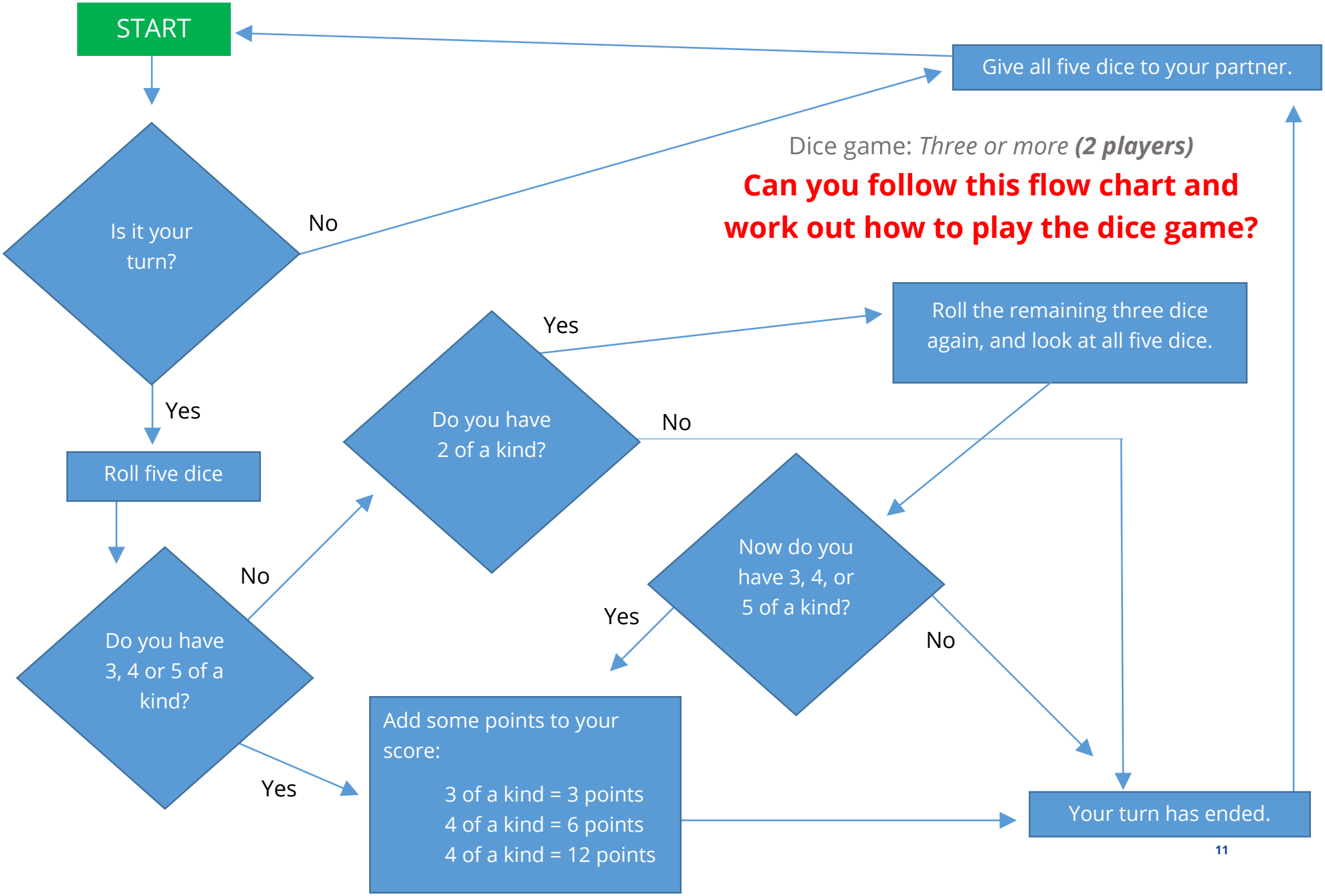
3 of a kind = 3 points
4 of a kind = 6 points
5 of a kind = 12 points

If you rolled only 2 of a kind, you have one more turn to improve your score. Put those 2 dice aside and roll the others again. If you succeed, score as above. If you don't, your score for the round is zero.

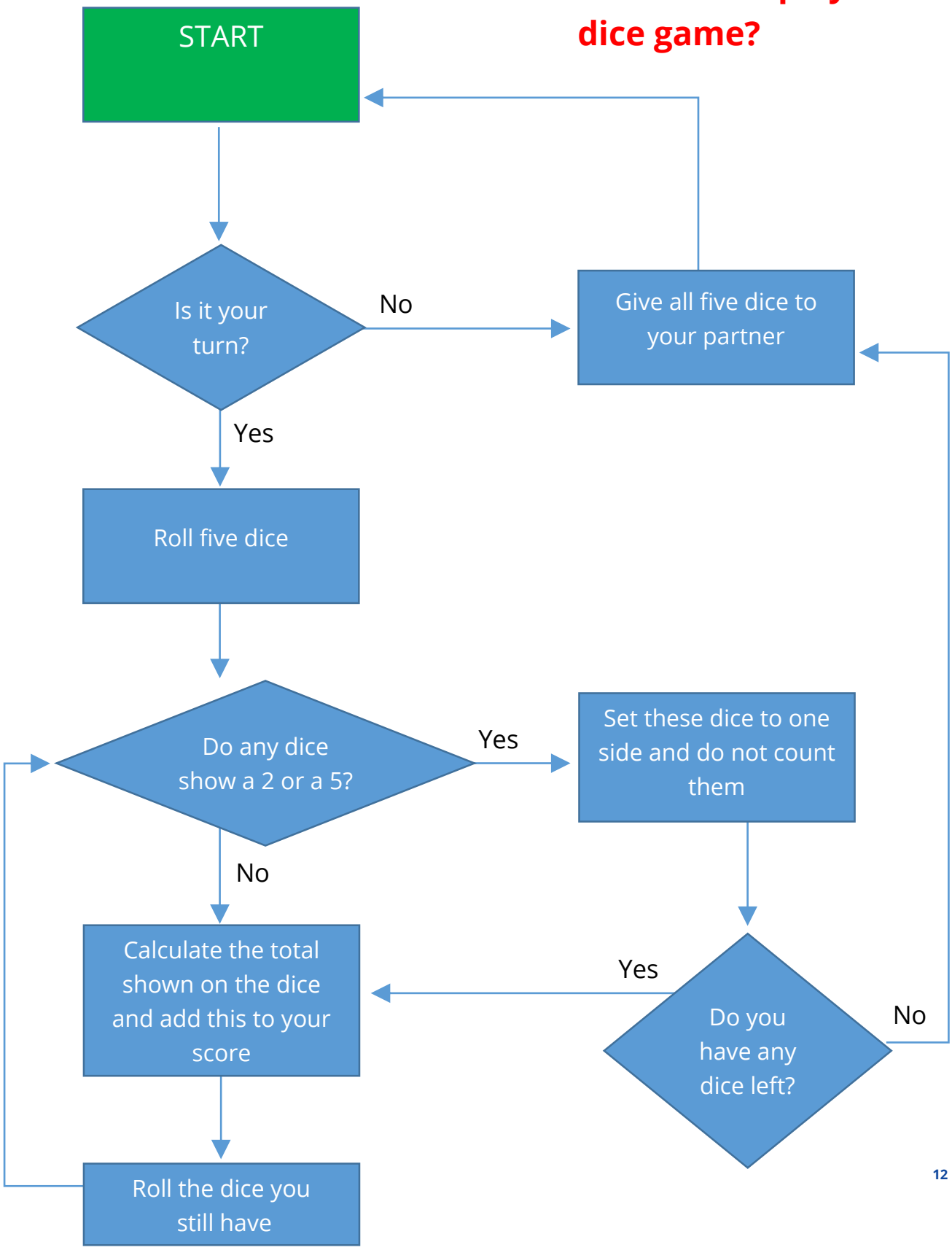
Stuck in the Mud – Two player game

The player with the highest score after a set number of rounds is the winner.

Roll 5 dice. Any values of 2 or 5 don't count, and those dice become 'stuck in the mud' and can no longer be used. Add up the total (excluding 5s and 2s), and then roll any of the dice which are not 'stuck' again. Again, discount any 2s or 5s, but add the new total to your score. Keep going, continuing to roll the dice and add their value to your total for as long as you have dice left. When all five of your dice are 'stuck in the mud', it is the next player's turn.


START
Give all five dice to your partner.
Dice game: *Three or more* ***(2 players)***
Can you follow this flow chart and work out how to play the dice game?
Is it your turn?
No
Yes
Roll five dice
Do you have 3, 4 or 5 of a kind?
No
Yes
Add some points to your score:
3 of a kind = 3 points
4 of a kind = 6 points
4 of a kind = 12 points
Do you have 2 of a kind?
Yes
Roll the remaining three dice again, and look at all five dice.
No
Now do you have 3, 4, or 5 of a kind?
Yes
No
Your turn has ended.

Can you follow this flow chart and work out how to play the dice game?



DECOMPOSITION: FERMI PROBLEMS - EXAMPLES

How many piano tuners are there in Chicago?

How many hairs are there on an average dog?

How many mobile phone calls are being made at this exact moment?

If everyone in our school lay head to toe in a straight line, where in the UK could the last person's toes be?

How many words are there in the book you are currently reading? How many letters? How many times does the letter 'e' appear?

How many grains of rice are there in a 1kg bag?

How old would you be if you have lived for exactly 1 million seconds?

How many blades of grass are there on a football pitch?

If you had a stack of £2 coins which was as tall as you, how much money would you have?

If you had your weight in 5p pieces, how much money would you have?

How fast do your fingernails grow?

How much water does your household use each week?

If you wanted to fill an Olympic size swimming pool with milk, how many cows would you need?

How many times would your bicycle wheel turn if you rode from home to school?

How many tennis balls are used during the Wimbledon championship?

How far can a brand new 2B pencil write?

How many tennis balls would fit in a bathtub?

How far have the pair of shoes you are currently wearing walked?

How many sheets of A4 paper would it take to cover the floor of your classroom?

How long is a ball of string?

PATTERNS: SPELLING SORT ACTIVITY
SILLY SENTENCES

What sound is being repeated in these sentences? Circle all of the words which contain this sound, then sort these words into groups according to the spelling pattern used to make the sound. Can you add any more words to each group?

Ryan decided to set an alarm for five past midnight as he wanted to try and fly his kite at night, but when he woke up and switched on the light he got a big fright – there in the corner of his bedroom was a giant tiger, who seemed ready for a fight and looked like he might bite.

Joe and Flo had planned to see a show, but then Joe broke a bone his toe while roaming in the woods so he phoned Flo to say she'd have to go alone. She groaned and said "Oh no! I don't want to go on my own, maybe I'll just stay at home alone."

Sue wished she knew who had untied the loops of her new blue shoelaces, because whoever it was had made her shoes so loose that when the wind blew, the two shoes flew off her feet and landed in Timbuktu where a hungry gnu mistook them for fruit and chewed on them all afternoon.

Why not try writing some silly sentences of your own?