
















Coding with Scratch: Learning Loops: Loops in Games

| | |
|--|---|
| <p>National Curriculum Design, write and debug programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them into smaller parts.</p> <p>Use sequence, selection, and repetition in programs; work with variables and various forms of input and output.</p> <p>Aim To design a simple catching game, making use of appropriate loops.</p> | <p>Lesson Duration It is estimated that this lesson will take approximately 60 minutes.</p>  |
| <p>Success Criteria</p> <p>I can solve a problem by decomposing it into smaller parts.</p> <p>I can design, write and debug algorithms to solve problems.</p> <p>I can identify the three types of loops in Scratch and select the most appropriate loop for a particular task.</p> <p>I can add a variable.</p> | <p>Key Vocabulary Algorithm, game, user, loop, debug, decomposition, variable.</p> |
| <p>Resources Lesson Pack</p> <p>PC devices, such as laptops, Chromebooks and/or tablets Scratch Online version accessed via</p> | <p>Preparation _____ - as required</p> <p>Scratch Blocks Bingo - one card per child and calling cards, cut up ready to use.</p> <p>Differentiated Design It Activity Sheet - one per child.</p> <p>Please access Lesson 6 (Catching Game) Scratch file within the _____</p> <p>In order to gather valuable data about how effective this unit has been and how much your children have understood the topic, we recommend completing the _____ at the end of this lesson.</p> |

Prior Learning: In the previous lessons, children will have become familiar with using a range of Scratch blocks to create algorithms. They have learnt about the three types of loops in Scratch and what they are used for. They have also had the opportunity to use the different loops within code.

Learning Sequence

| | | |
|--|---|---|
|  | <p>Remember It: Ask the children if they can remember some of the different categories of blocks found in Scratch. Recap on the different blocks that have been used so far in the unit by playing a fun game of Scratch Blocks Bingo.</p> <p>Use the Lesson Presentation to remind children about the three different loops that are available in Scratch and why loops are useful when writing code.</p> |  |
|  | <p>Game Design: Use the Lesson Presentation to initiate discussion about some of the different types of computer games that are available and allow children time to talk about the types of games they enjoy playing and why.</p> <p>Run Lesson 6 (Catch It) within the Scratch Project Area for children to see a simple catching game in action. Use the questions on the Lesson Presentation to encourage children to think more deeply about how the game is designed and played.</p> |  |
|  | <p>Decompose It: Introduce the term decomposition and ensure children understand that breaking a problem down into smaller, more manageable parts is an important skill in coding. Encourage children to think about how decomposition is used in everyday problem-solving too.</p> <p>Can you think of a time you used decomposition to help you solve a problem?</p> <p>Use the Lesson Presentation to show children how to decompose the catching game into more manageable tasks. For each algorithm, children are presented with the blocks required and challenged to think about the order in which they should be organised. Give children time to talk about the blocks and provide logical reasons for the order they should be arranged. Then view the possible algorithm that could be used to code that part of the game. For each algorithm, children will be asked to think about what is missing. They should be able to identify that a loop is needed for repetition and be able to suggest which loop would be most appropriate. Draw attention to the fact that loops have made it possible to repeat actions in the game without writing hundreds or even thousands of lines of code.</p> <p>In order to add a score to the game, children will be introduced to variables. A variable in Scratch can be described as an empty box that holds a value that can be stored in Scratch's memory. Demonstrate how to make a score variable and make it change by 1 each time the Ball sprite is caught in the Bowl.</p> <p>How would you have built these algorithms if there weren't any loops in Scratch?</p> |  |

| | | | | | | |
|---|--|---|--|--|---|---|
|  | <p>Design It: Use the Lesson Presentation to introduce the task. Ensure that children have access to the appropriate Design It Activity Sheet.</p> |  | | | | |
|  | <p>Children will copy the algorithms into Scratch to create a Catch It game. They will select sprites and backdrops to create an eye-catching game.</p> |  | <p>Children will copy the algorithms into Scratch, adding key details. They will select sprites, backdrops and sound effects to create an eye-catching game.</p> |  | <p>Children will copy the algorithms into Scratch, adding key details. They will change sprites and backdrops and add sound effects to create an eye-catching game. Children can then add an extra sprite to increase the difficulty level of the game.</p> |  |
|  | <p>Share It: Once children have completed their game designs, encourage them to share them with one another. Remind children that feedback on other people's work should be constructive and positive at all times.</p> | | | |  | |

Exploreit

Createit: Children use the _____ to create a simple Scratch game. Remind children to test their game and debug it if required.

Designit: Children to design a poster to advertise a computer game that they like to play, considering how they could encourage other people to want to play the game too. They must include a title, a description of the game, an age range and some eye-catching art work.

Developit: Challenge children to develop the catching game into a game for older children. Children should consider what themes older children may prefer and if they need to change the speed and complexity of the game. Use the [Game Design Market Research Activity Sheet](#) to collect information from the class before designing a new catching game for an older audience.

Assessment Notes:

Disclaimers:

External Links:

This resource contains links to external websites and/or external apps. Please be aware that the inclusion of any link in this resource should not be taken as an endorsement of any kind by Twinkl of the linked website and/or app, or any association with its operators. You should also be aware that we have no control over the availability of the linked pages and/or apps. If the link is not working, please let us know by contacting TwinklCares and we will try to fix it although we can assume no responsibility if this is the case. We are not responsible for the content of external sites and/or external apps.

Scratch Safety:

Showing or creating the flashing sprite effect could be problematic for children with conditions such as epilepsy. Discretion is advised.



Computing

Coding with Scratch: Learning Loops

Coding
with Scratch

Loops in Games



touching Bowl ▾

Question Marks

**This is Quizby.
He is a question mark who
loves to ask questions.**



When you see a question mark icon like this in the **Lesson Presentation**, it can be clicked on to reveal one of Quizby's questions.



The questions that appear next to these question marks will help you to think about the key learning throughout the lesson.

Aim

To design a simple catching game, making use of appropriate loops.

Success Criteria

I can solve a problem by decomposing it into smaller parts.

I can design, write and debug algorithms to solve problems.

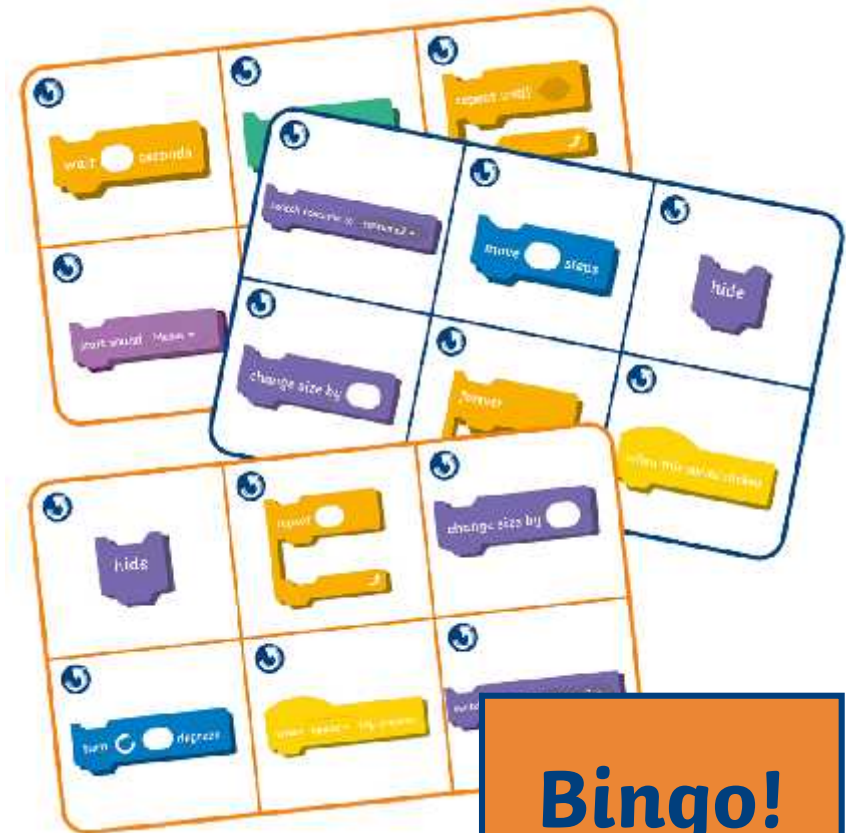
I can identify the three types of loops in Scratch and select the most appropriate loop for a particular task.

I can add a variable.

Remember It

There are so many different types of blocks in Scratch to choose from. Can you remember some of the loop blocks we have used in this unit?

Play **Scratch Block Bingo** and see who will be first to cover all their blocks. Don't forget to shout Bingo! if you want to be the winner.



Remember It

Here are three blocks that you have been learning about in this unit. Can you remember what these blocks do and why they are useful in coding?

In a count-controlled loop, the instructions are repeated a specific number of times.

In a **forever** loop, the instructions are repeated without end.

In a **repeat until** loop, the instructions are repeated until a certain condition is met.

These blocks are useful because they produce loops. Loops are a way of repeating a set of instructions. The computer checks the instructions inside the loop and repeats them. Loops save a computer programmer from having to write extra lines of code.

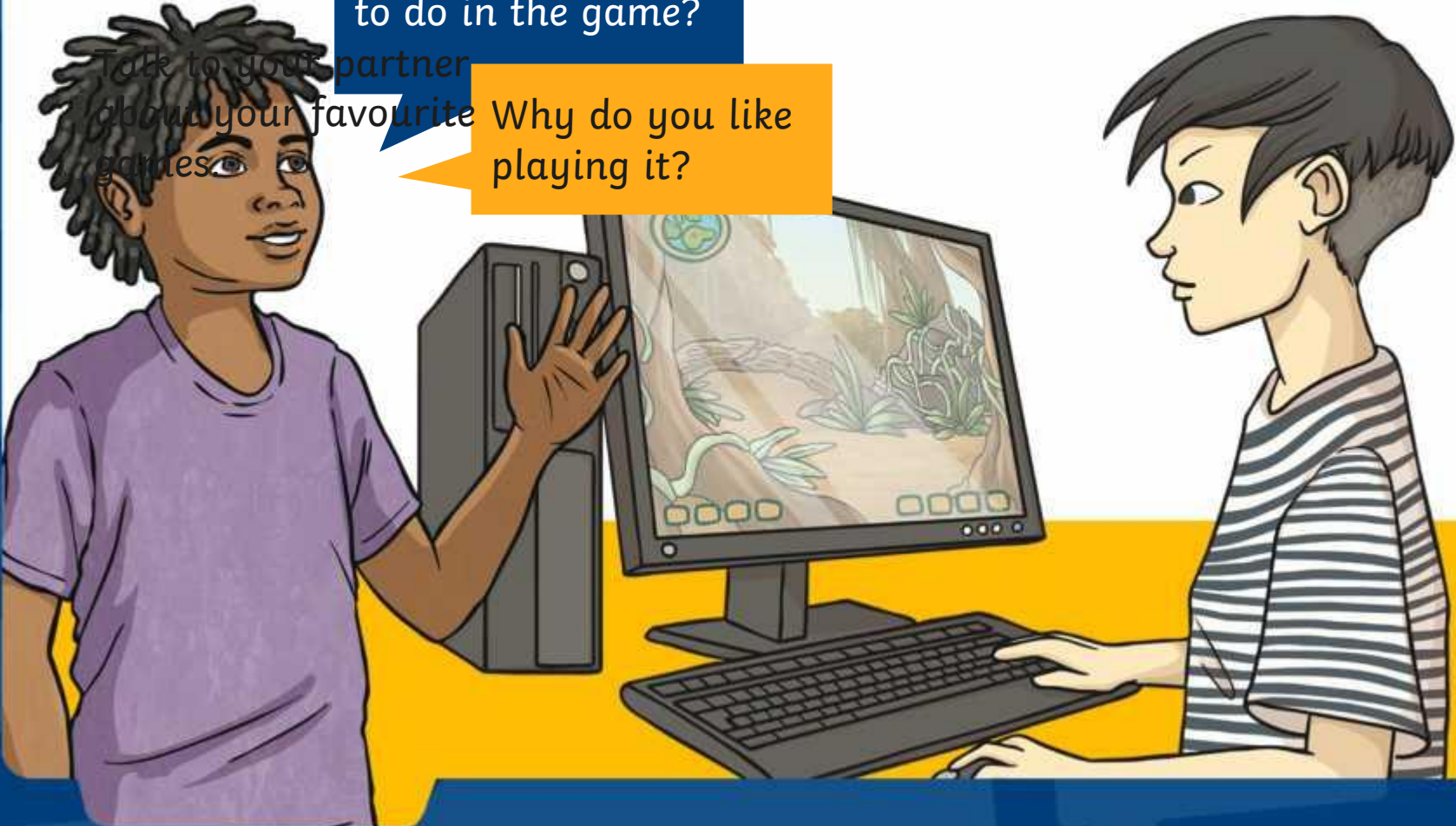
Game Design

Do you play computer games at home or in school?

What do you have to do in the game?

Talk to your partner about your favourite games.

Why do you like playing it?



Game Design

There are lots of different types of computer games to play. Have you played any of these different types?



Adventure
A game played as a character.



Target
A game that involves aiming at a target or catching an object.



Racing
A game where you race against other competitors.



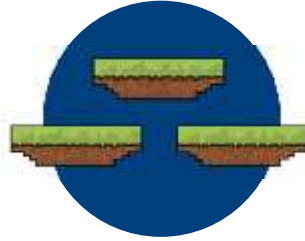
Clicker
A game where you click on objects to reveal rewards.

Adventure



A game based on solving puzzles.

Jumping



A game where you jump over objects or onto platforms.

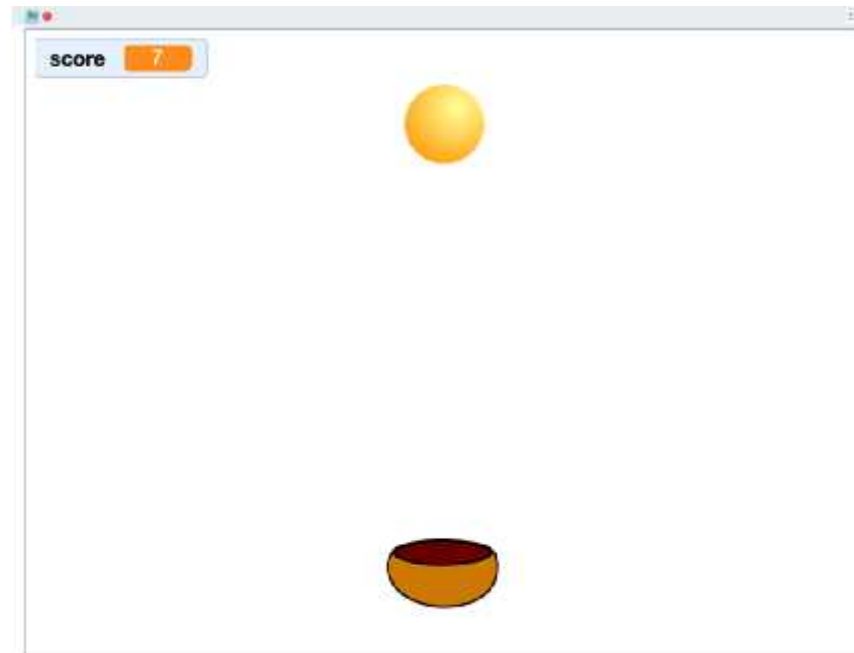
Simulator



A game that copies activities in the real world.

Game Design

In this lesson, you will be using Scratch to design and make a simple catching game for younger children. Have a look at [Lesson 6 \(Catch It\)](#) to find out what a catching game looks like. What is the aim of the game? How do you play it? How do you win the game?



Decompose It

There are lots of different things all happening at the same time in the **Catch It** game

Computer games are complicated things to make. Games designers can't write the code for all the different parts of the game at the same time. They have to break the game down into smaller sections and tackle each part in turn. This is called decomposition.

Decomposition means breaking things down into smaller parts making them easier to manage and understand.

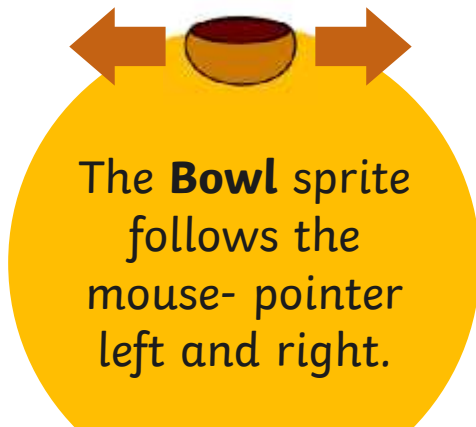
X

Can you think of a time you used decomposition to help you solve a problem?

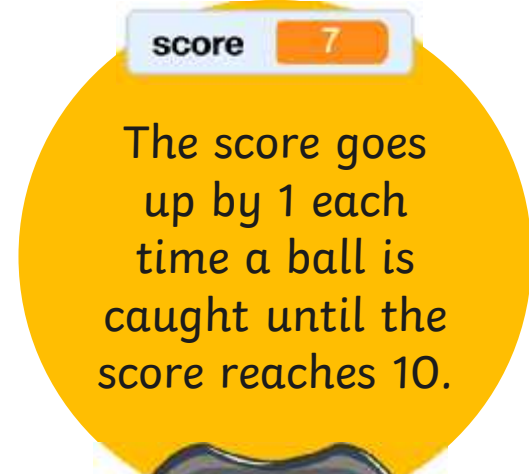
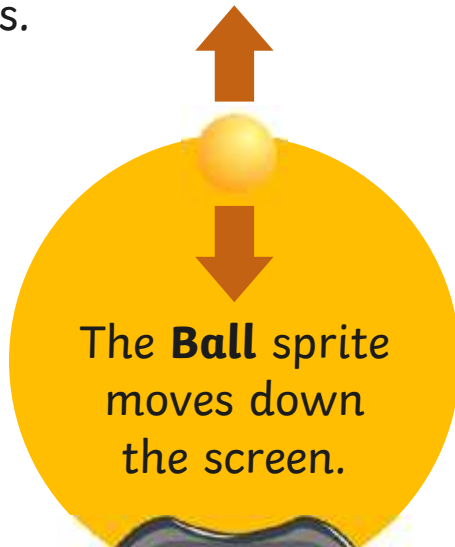


Decompose It

Talk About It: Have another look at
Talk to your partner or group about the different things that you can see happening in the **Catch It** game and then click on each controller to see if you spotted these things.



Scratch call it a mouse-pointer, you may refer to this as a cursor.



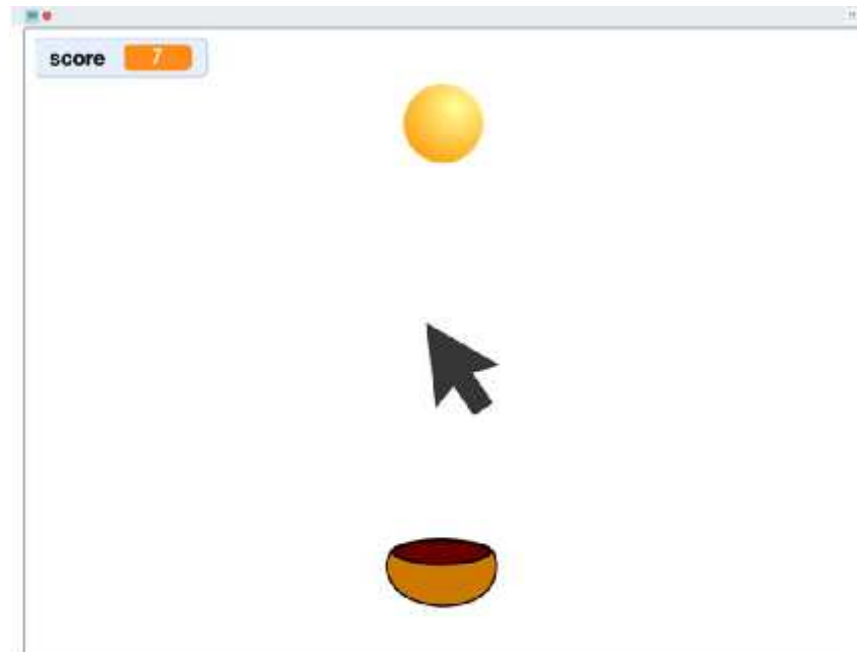
Decompose It

Let's have a look at how
each section could be
coded.

Decompose It

Step 1: Make a sprite follow the mouse-pointer left and right.

Did you Catch It from the Bowl
Bowl's position is positioned near
the bottom of the Stage. Stage had
mouse to catch the ball. The Bowl
is following the mouse-pointer.

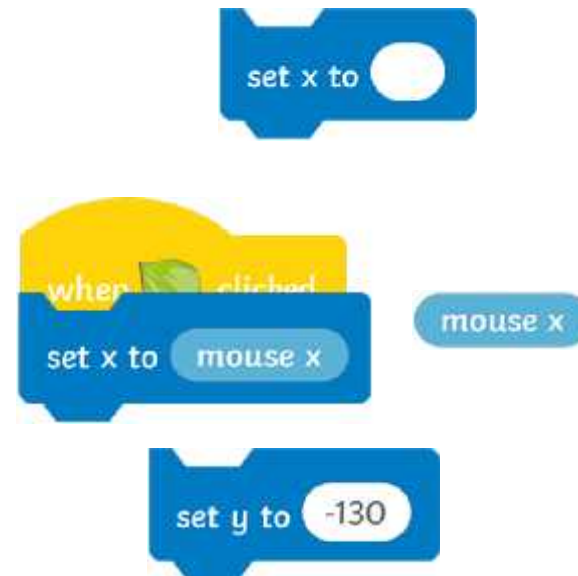


Decompose It

Step 1: Make a sprite follow the mouse-pointer left and right.

These are the blocks needed to make the **Bowl** sprite follow the mouse-pointer left and right but they are muddled up.

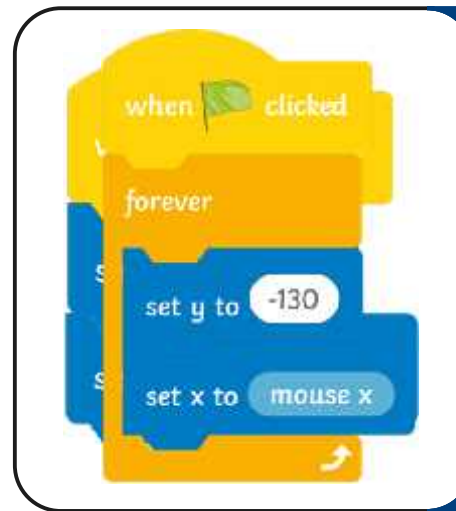
Can you work out what order they should be in?



Decompose It

Step 1: Make a sprite follow the mouse-pointer left and right.

Can you think which **repeat** block would be the best one to use to make a loop that always repeats?



That's right! A **forever** loop will make the sprite always follow the mouse-pointer.

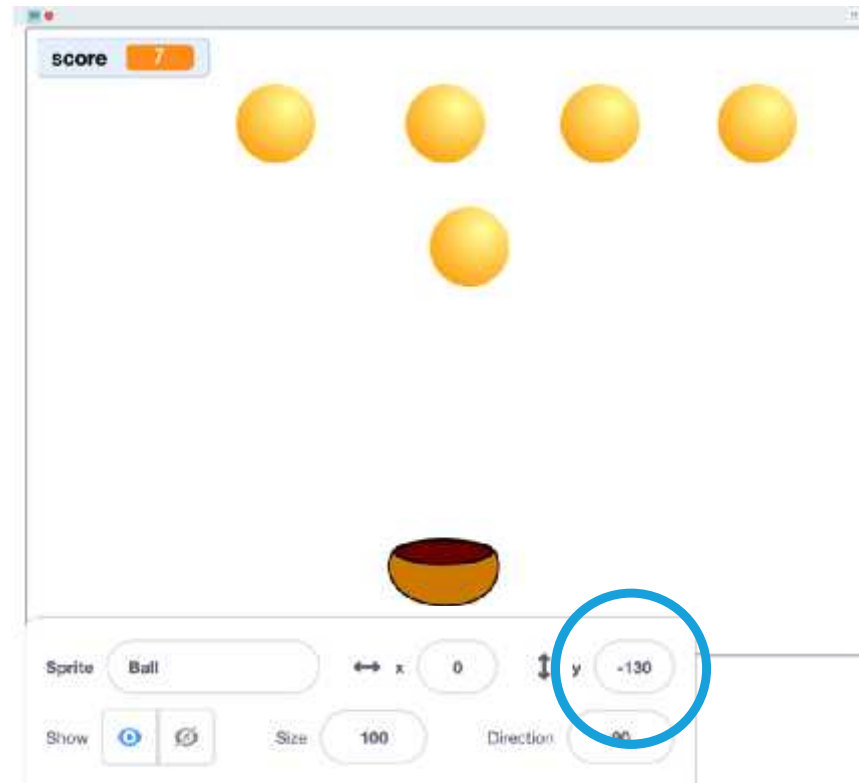
Look at your if you need some help remembering the repeat blocks.

Click on the algorithm above to see if you were right.

Decompose It

Step 2: Make a sprite fall down and then go back to the top of the **Stage**

Tip: As it goes, the **Ball** sprite falls down from the top to the bottom of the **Stage** grid. The **Ball** sprite then reappears at the top coordinate. You can see the **Sprite Pane** underneath the **Stage** area in Scratch. Try moving a sprite up and down and look at what happens to the **y** coordinate.



Decompose It

Step 2: Make a sprite fall down and then go back to the top of the **Stage**

Have a look at the blocks needed to make the ball travel up and down on the **Stage**.

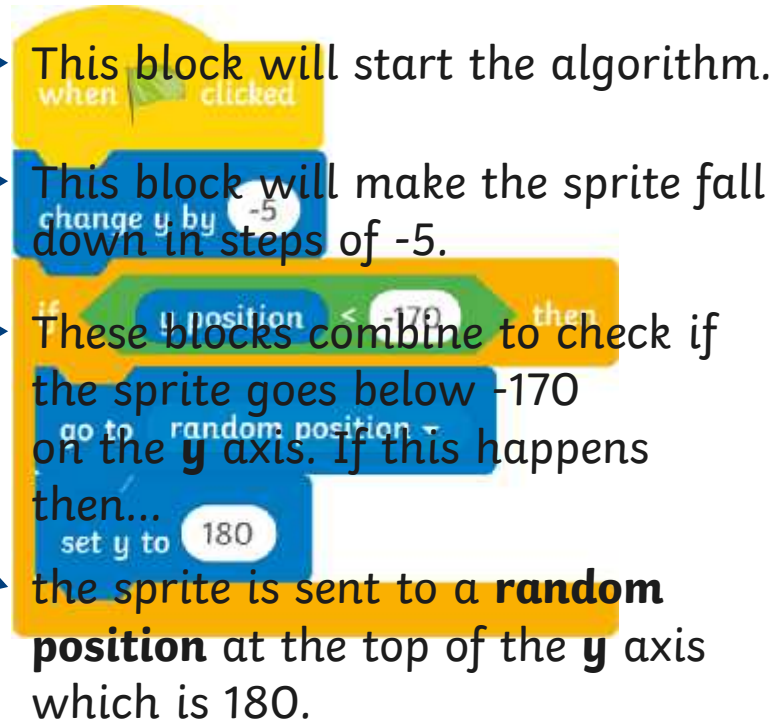
Can you think what order they might go in?



Decompose It

Step 2: Make a sprite fall down and then go back to the top of the Stage

How did you get on?
Have a look at the algorithm
and see if you put the blocks
in the same order.



Decompose It

Step 2: Make a sprite fall down and then go back

Ooops! Just like last time - something important is missing. In this algorithm, the **Ball** sprite will only fall once.

What needs to be added to make the ball fall over and over again?



```
when green flag clicked
  forever loop
    change y by -5
    if y position < -170 then
      go to random position
      set y to 180
```

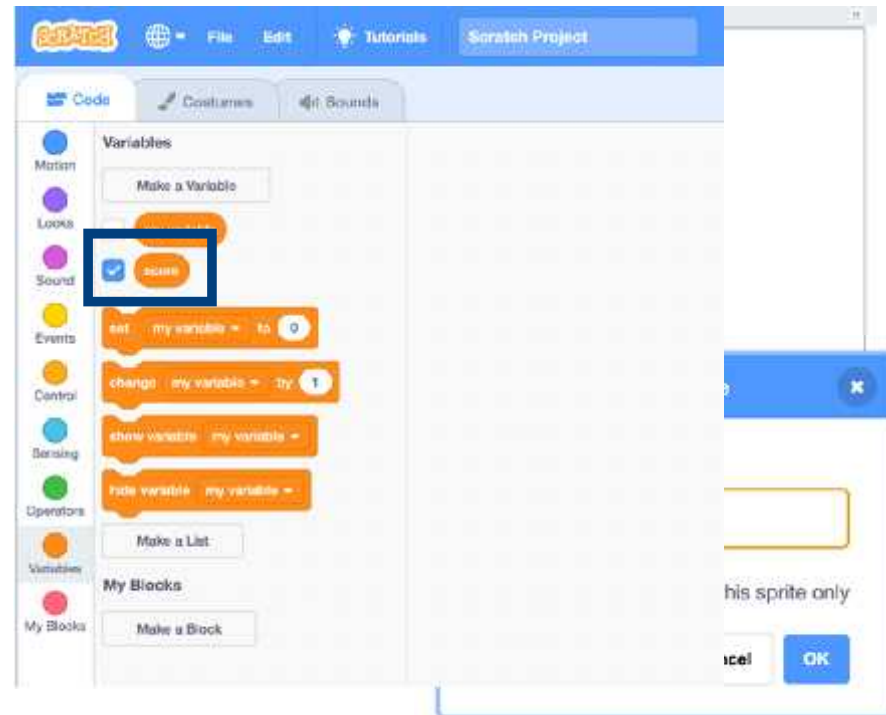
That's right!
Don't forget that important **forever** loop to repeat the instructions endlessly.

Click on the algorithm above to see if you are right.

Decompose It

Step 3: Add a score when the **Ball** is caught in the **Bowl**

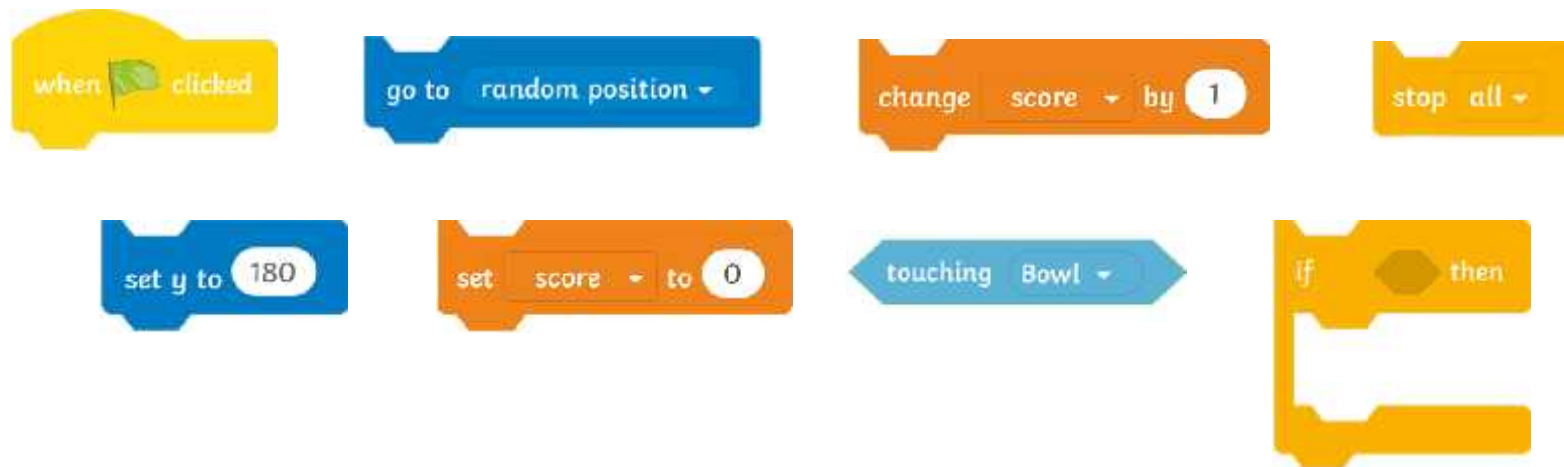
Adding a score to the game is a simple task. In the **Code** area, click on the **Variables** category. A dialog box will appear, allowing you to create a new variable. Name the variable **score** and set its initial value to **0**. This variable will be used to track the number of catches scored in the game.



Decompose It

Step 3: Add a score when the **Ball** is caught in the **Bowl**

Here are the blocks for coding the score and ending the game. What order do you think these blocks should be in?



Decompose It

Step 3: Add a score when the **Ball** is caught in the **Bowl**

Here is the algorithm for coding the score and ending the game.

Once again, there is something missing from this algorithm. Can you guess what it might be?

You guessed it! The algorithm needs a loop to make it work correctly. This time the loop needs to repeat the instructions until the score reaches 10. Which loop will be the best one to add?



Click on the algorithm above to see if you are right.

Decompose It

Step 3: Add a score when the **Ball** is caught in the **Bowl**

Did you spot that a **repeat until** block was needed this time? A **repeat until** loop will check the instructions and repeat them until the score reaches 10. Once the score reaches 10 the condition has been met and the loop will stop.



Decompose It

Decomposing the game into smaller parts has made it easier to write all the algorithms needed.

```
when clicked
  forever
    set y to -130
    set x to mouse x
```

```
when clicked
  change y by -5
  if y position < -170 then
    go to random position
    set y to 180
```

```
when clicked
  set score to 0
  repeat until score = 10
    if touching Bowl then
      change score by 1
      go to random position
      set y to 180
  stop all
```

Design It



The **Catch It** game was designed for younger children to play. In market research, users reported that they thought the game was good for young children but they didn't think the game was eye-catching or interesting enough.

Can you be a games designer and make the game more exciting for users? Follow the instructions on the **Design It Activity Sheet** and create the best game you can. Good luck!

Design It

To design a simple catching game, making use of appropriate loops.

Task 1: Choose a theme

Have a look at the backdrops and the sprites in Scratch. Which theme will you choose for your game? Try to pick backdrops and sprites that work well together. Select a sprite that will fall down the **Stage**. Select another sprite that will catch the falling sprite. Add them to your **Stage**.

Can you think of an exciting name for your game?

Task 2: Write the game

Create these algorithms in Scratch. Add in the missing information to change how fast the sprite falls, what the top score will be and how much the score will change each time a sprite is caught.

Top Tip: Remember that you will have to create your own 'score' variable before you can add it into the algorithm. Go to the **Variables** category in the **Block Palette** and choose **Make a Variable**. Call the variable 'score'.

Catching Sprite

Falling Sprite

Challenge:

Can you add a sound effect whenever the falling sprite is caught?

Aim

To design a simple catching game, making use of appropriate loops.

Success Criteria

I can solve a problem by decomposing it into smaller parts.

I can design, write and debug algorithms to solve problems.

I can identify the three types of loops in Scratch and select the most appropriate loop for a particular task.

I can add a variable.



**Coding
with Scratch**

Design It

To design a simple catching game, making use of appropriate loops.



Task 1: Choose a theme

Have a look at the backdrops and the sprites in Scratch. Which theme will you choose for your game? Try to pick backdrops and sprites that work well together. Select a sprite that will fall down the **Stage**. Select another sprite that will catch the falling sprite. Add them to your **Stage**.

Can you think of an exciting name for your game?

Task 2: Write the game

Create these algorithms in Scratch. Test your game to see if it works correctly. Remember to test and debug if you find any errors.

Top Tip: Remember that you will have to create your own 'score' variable before you can add it into the algorithm. Go to the **Variables** category in the **Block Palette** and choose **Make a Variable**. Call the variable 'score'.

Catching Sprite

```
when clicked
  forever
    set y to -130
    set x to mouse x
```

Falling Sprite

```
when clicked
  forever
    change y by -5
    if y position < -170 then
      go to random position
      set y to 180
```

```
when clicked
  set score to 0
  repeat until score = 10
    if touching Bowl then
      change score by 1
      go to random position
      set y to 180
  stop all
```

Challenge:

Can you change the speed of the falling sprite and make it fall faster?

Design It

To design a simple catching game, making use of appropriate loops.



Task 1: Choose a theme

Have a look at the backdrops and the sprites in Scratch. Which theme will you choose for your game? Try to pick backdrops and sprites that work well together. Select a sprite that will fall down the **Stage**. Select another sprite that will catch the falling sprite. Add them to your **Stage**.

Can you think of an exciting name for your game?

Task 2: Write the game

Create these algorithms in Scratch. Add in the missing information to change how fast the sprite falls, what the top score will be and how much the score will change each time a sprite is caught.

Top Tip: Remember that you will have to create your own 'score' variable before you can add it into the algorithm. Go to the **Variables** category in the **Block Palette** and choose **Make a Variable**. Call the variable 'score'.

Catching Sprite

```
when clicked
  forever loop
    set y to -130
    set x to mouse x
```

Falling Sprite

```
when clicked
  forever loop
    change y by 1
    if y position < -170 then
      go to random position
      set y to 180
```

```
when clicked
  set score to 0
  repeat until score = 10
    if touching Bowl then
      change score by 1
      go to random position
      set y to 180
  stop all
```

Challenge:

Can you add a sound effect whenever the falling sprite is caught?



Design It

To design a simple catching game, making use of appropriate loops.



Task 1: Choose a theme

Have a look at the backdrops and the sprites in Scratch. Which theme will you choose for your game? Try to pick backdrops and sprites that work well together. Select a sprite that will fall down the **Stage**. Select another sprite that will catch the falling sprite. Add them to your **Stage**.

Can you think of an exciting name for your game?

Task 2: Write the game

Create these algorithms in Scratch. Add in the missing information to change how fast the sprite falls, what the top score will be and how much the score will change each time a sprite is caught.

Top Tip: Remember that you will have to create your own 'score' variable before you can add it into the algorithm. Go to the **Variables** category in the **Block Palette** and choose **Make a Variable**. Call the variable 'score'.

Catching Sprite

```
when clicked
forever
  set y to -130
  set x to mouse x
```

Falling Sprite

```
when clicked
forever
  change y by 0
  if y position < -170 then
    go to random position
    set y to 180
```

```
when clicked
set score to 0
repeat until score = 0
  if touching Bowl then
    change score by 0
    go to random position
    set y to 180
stop all
```

Challenge:

Can you add a sound effect whenever a falling sprite is caught?



Super Challenge:

Can you add another falling sprite into the game to make the game harder to play?



Design It Possible Solutions

Children's individual solutions to creating the code may vary but trial and error and experimentation with ideas should be encouraged.



```
when clicked
  forever
    change y by -5
    if y position < -170 then
      go to random position
      set y to 180
```

To make the falling sprite fall faster, children will need to decrease the value of the negative number from -5. They could change it to -10 or -12.



```
when clicked
  set score to 0
  repeat until score = 10
    if touching Bowl then
      change score by 1
      go to random position
      set y to 180
      start sound Jump
  stop all
```

Children can add a sound effect of their choice into this algorithm to create a sound whenever a sprite is caught.

Design It Possible Solutions



```
when clicked
  set score to 0
  repeat until score = 10
    if touching Bowl then
      change score by 1
      go to random position
      set y to 180
      start sound Jump
  stop all
```

Children can add a sound effect of their choice into this algorithm to create a sound whenever a sprite is caught.

To add a new sprite, children can click on the falling sprite in the **Sprite Pane** and duplicate it or they can choose a different sprite and write the code for it. Encourage children to think about how they might need to edit the speed of the falling sprite when there is more than one to catch.

Design It

To design a simple catching game, making use of appropriate loops.



Task 1: Choose a theme

Have a look at the backdrops and the sprites in Scratch. Which theme will you choose for your game? Try to pick backdrops and sprites that work well together. Select a sprite that will fall down the **Stage**. Select another sprite that will catch the falling sprite. Add them to your **Stage**.

Can you think of an exciting name for your game?

Task 2: Write the game

Create these algorithms in Scratch. Test your game to see if it works correctly. Remember to test and debug if you find any errors.

Top Tip: Remember that you will have to create your own 'score' variable before you can add it into the algorithm. Go to the **Variables** category in the **Block Palette** and choose **Make a Variable**. Call the variable 'score'.

Catching Sprite

```
when clicked
  forever
    set y to -130
    set x to mouse x
```

Falling Sprite

```
when clicked
  forever
    change y by -5
    if y position < -170 then
      go to random position
      set y to 180
```

```
when clicked
  set score to 0
  repeat until score = 10
    if touching Bowl then
      change score by 1
      go to random position
      set y to 180
  stop all
```

Challenge:

Can you change the speed of the falling sprite and make it fall faster?

Design It

To design a simple catching game, making use of appropriate loops.



Task 1: Choose a theme

Have a look at the backdrops and the sprites in Scratch. Which theme will you choose for your game? Try to pick backdrops and sprites that work well together. Select a sprite that will fall down the **Stage**. Select another sprite that will catch the falling sprite. Add them to your **Stage**.

Can you think of an exciting name for your game?

Task 2: Write the game

Create these algorithms in Scratch. Add in the missing information to change how fast the sprite falls, what the top score will be and how much the score will change each time a sprite is caught.

Top Tip: Remember that you will have to create your own 'score' variable before you can add it into the algorithm. Go to the **Variables** category in the **Block Palette** and choose **Make a Variable**. Call the variable 'score'.

Catching Sprite

```
when clicked
  forever loop
    set y to -130
    set x to mouse x
```

Falling Sprite

```
when clicked
  forever loop
    change y by 1
    if y position < -170 then
      go to random position
      set y to 180
```

```
when clicked
  set score to 0
  repeat until score = 10
    if touching Bowl then
      change score by 1
      go to random position
      set y to 180
  stop all
```

Challenge:

Can you add a sound effect whenever the falling sprite is caught?



Design It

To design a simple catching game, making use of appropriate loops.



Task 1: Choose a theme

Have a look at the backdrops and the sprites in Scratch. Which theme will you choose for your game? Try to pick backdrops and sprites that work well together. Select a sprite that will fall down the **Stage**. Select another sprite that will catch the falling sprite. Add them to your **Stage**.

Can you think of an exciting name for your game?

Task 2: Write the game

Create these algorithms in Scratch. Add in the missing information to change how fast the sprite falls, what the top score will be and how much the score will change each time a sprite is caught.

Top Tip: Remember that you will have to create your own 'score' variable before you can add it into the algorithm. Go to the **Variables** category in the **Block Palette** and choose **Make a Variable**. Call the variable 'score'.

Catching Sprite

```
when green flag clicked
forever loop
  set y to -130
  set x to mouse x
```

Falling Sprite

```
when green flag clicked
forever loop
  change y by 1
  if y position < -170 then
    go to random position
    set y to 180
```

```
when green flag clicked
set score to 0
repeat until score = 10
  if touching Bowl then
    change score by 1
    go to random position
    set y to 180
stop all
```

Challenge:

Can you add a sound effect whenever a falling sprite is caught?



Super Challenge:

Can you add another falling sprite into the game to make the game harder to play?



Design It Possible Solutions

Children's individual solutions to creating the code may vary but trial and error and experimentation with ideas should be encouraged.



```
when clicked
  forever
    change y by -5
    if y position < -170 then
      go to random position
      set y to 180
```

To make the falling sprite fall faster, children will need to decrease the value of the negative number from -5. They could change it to -10 or -12.



```
when clicked
  set score to 0
  repeat until score = 10
    if touching Bowl then
      change score by 1
      go to random position
      set y to 180
      start sound Jump
  stop all
```

Children can add a sound effect of their choice into this algorithm to create a sound whenever a sprite is caught.

Design It Possible Solutions



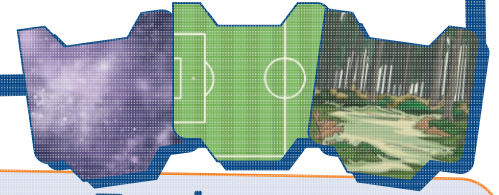
```
when clicked
  set score to 0
  repeat until score = 10
    if touching Bowl then
      change score by 1
      go to random position
      set y to 180
      start sound Jump
  stop all
```

Children can add a sound effect of their choice into this algorithm to create a sound whenever a sprite is caught.

To add a new sprite, children can click on the falling sprite in the **Sprite Pane** and duplicate it or they can choose a different sprite and write the code for it. Encourage children to think about how they might need to edit the speed of the falling sprite when there is more than one to catch.

Game Design Market Research

Challenge: Can you create a new version of the Scratch Catching Game aimed at older children. It will need an exciting new theme. Do older children like space, animal or sports themes? Add some themes to the tally chart and carry out market research in your class or school to find out what the most popular themes for a computer game are. You could have a look at the sprites in Scratch to think of some themes to add to your tally chart.



| Theme | Tally | Total |
|-------|-------|-------|
| | | |
| | | |
| | | |
| | | |
| | | |

The most popular theme for a new computer game is _____

The least popular theme for a new computer game is _____

Game Design Market Research



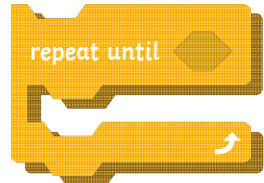


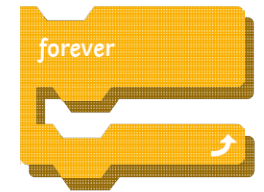
Challenge: Can you create a new version of the Scratch Catching Game aimed at older children. It will need an exciting new theme. Do older children like space, animal or sports themes? Add some themes to the tally chart and carry out market research in your class or school to find out what the most popular themes for a computer game are. You could have a look at the sprites in Scratch to think of some themes to add to your tally chart.





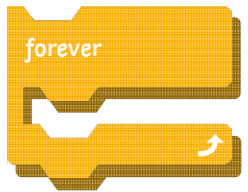







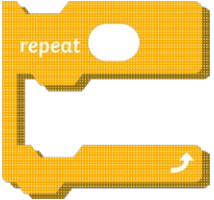

| Theme | Tally | Total |
|-------|-------|-------|
| | | |
| | | |
| | | |
| | | |
| | | |



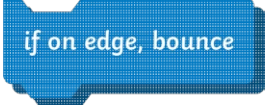

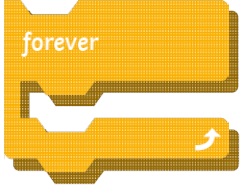
The most popular theme for a new computer game is _____

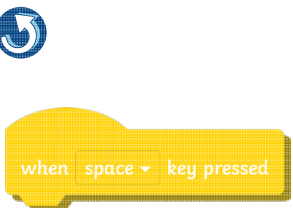

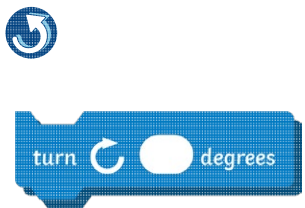
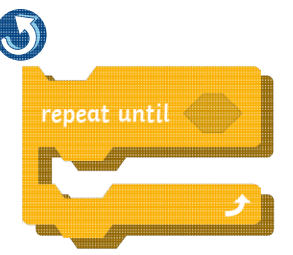
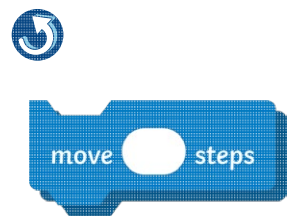
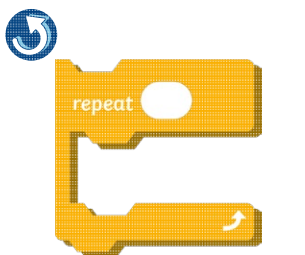
The least popular theme for a new computer game is _____


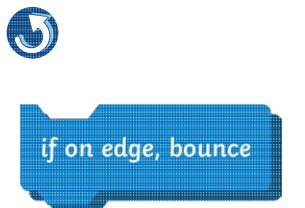

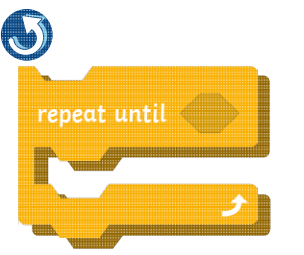
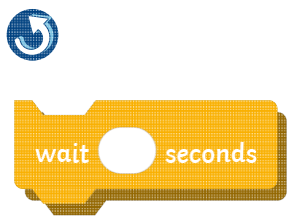
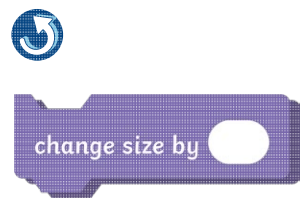
| | | |
|---|---|--|
|  |  |  |
|  |  |  |


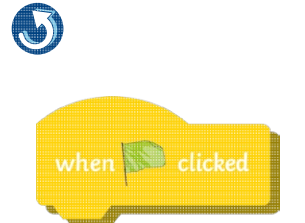

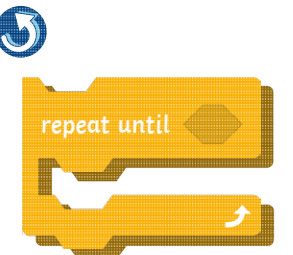
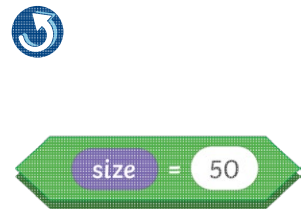

| | | |
|---|---|---|
|  |  |  |
|  |  |  |

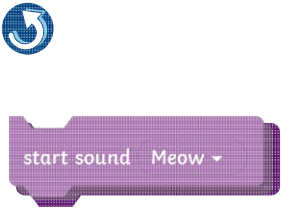
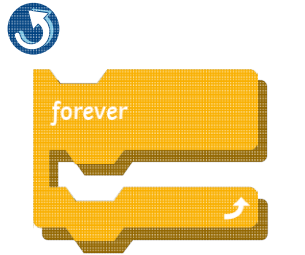

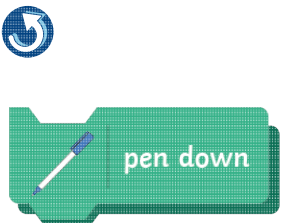
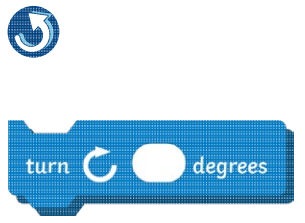

| | | |
|---|---|--|
|  |  |  |
|  |  |  |


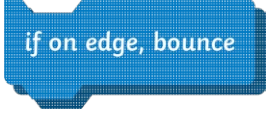
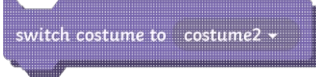


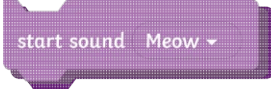
| | | |
|---|---|---|
|  |  |  |
|  |  |  |



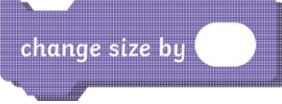


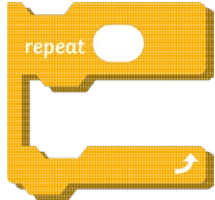
| | | |
|---|---|--|
|  |  |  |
|  |  |  |







| | | |
|---|---|---|
|  |  |  |
|  |  |  |


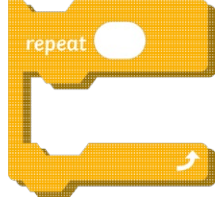



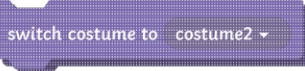
| | | |
|---|---|--|
|  |  |  |
|  |  |  |





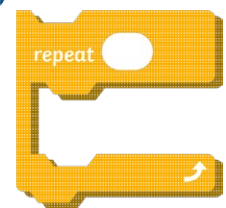

| | | |
|---|---|---|
|  |  |  |
|  |  |  |







| | | |
|---|---|--|
|  |  |  |
|  |  |  |




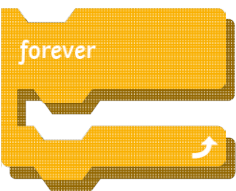


| | | |
|---|---|---|
|  |  |  |
|  |  |  |







| | | |
|---|---|--|
|  |  |  |
|  |  |  |

| | | |
|---|---|---|
|  |  |  |
|  |  |  |

| | | |
|---|---|--|
|  |  |  |
|  |  |  |

| | | |
|---|---|---|
|  |  |  |
|  |  |  |

| | | |
|---|---|--|
|  |  |  |
|  |  |  |

| | | |
|---|---|---|
|  |  |  |
|  |  |  |

| | | |
|--|--|--|
| | | |
| | | |

| | | |
|--|--|--|
| | | |
| | | |

| | | |
|--|--|--|
| | | |
| | | |


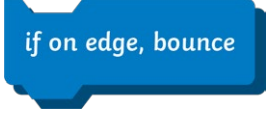
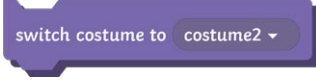


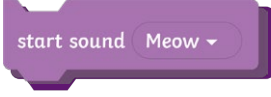
| | | |
|--|--|--|
| | | |
| | | |



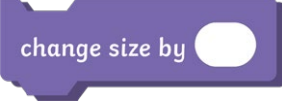


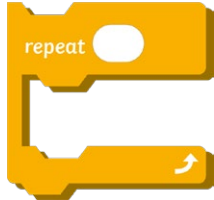
| | | |
|--|--|--|
| | | |
| | | |

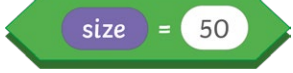


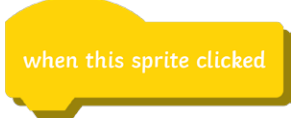

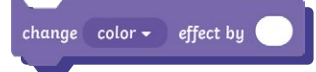
| | | |
|--|--|--|
| | | |
| | | |


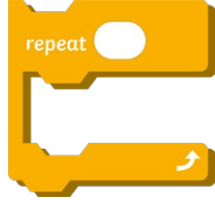



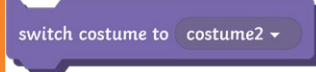
| | | |
|--|--|--|
| | | |
| | | |




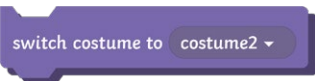
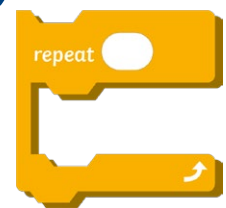

| | | |
|--|--|--|
| | | |
| | | |

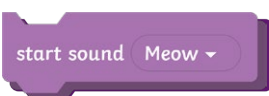





| | | |
|---|---|--|
|  |  |  |
|  |  |  |





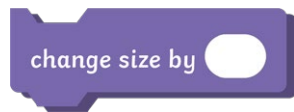

| | | |
|---|---|---|
|  |  |  |
|  |  |  |


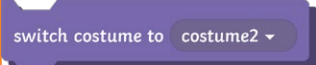


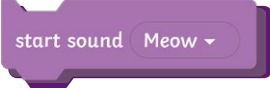

| | | |
|---|---|--|
|  |  |  |
|  |  |  |

| | | |
|---|---|---|
|  |  |  |
|  |  |  |

| | | |
|---|---|--|
|  |  |  |
|  |  |  |

| | | |
|---|---|---|
|  |  |  |
|  |  |  |

| | | |
|---|---|--|
|  |  |  |
|  |  |  |

| | | |
|---|---|---|
|  |  |  |
|  |  |  |

Scratch Blocks Bingo

Instructions

1. Print out and cut up the bingo boards and the caller cards.
2. Give out bingo cards to each player along with six counters.
3. Shuffle the calling cards and place them in a pile, face down on the table.
4. The bingo caller takes one card at a time and reads out the description of the Scratch block given on the card.
5. The bingo players look for the corresponding Scratch block on their bingo card. If they find it they can cover the block with a counter.
6. The winner is the first player to cover all their Scratch blocks and shout BINGO!

Calling Cards

A block to put the pen down

A block to change the colour of a sprite

A block to make the size of a sprite equal to 50

A block to make a sprite invisible

A block to make a sprite bounce off the edge of the **Stage**

A block to make a sprite grow or shrink by a certain amount

A block to make a sprite turn right

A block to make a sprite move forward a certain distance

A block to start a sound effect

A block to make a sprite stop and wait for a certain length of time

A block to make a sprite do something when it is clicked

A block to make a sprite change its costume

A block to make something happen when the spacebar is pressed

A block to make a sprite say something

A block to make something happen when the green flag is clicked

A count-controlled **repeat** block

A repeat **forever** block

A **repeat until** a certain condition is met block

Coding with Scratch: Learning Loops | Loops in Games

| | | |
|--|--|--|
| To design a simple catching game, making use of appropriate loops. | | |
| I can solve a problem by decomposing it into smaller parts. | | |
| I can design, write and debug algorithms to solve problems. | | |
| I can identify the three types of loops in Scratch and select the most appropriate loop for a particular task. | | |
| I can add a variable. | | |

Coding with Scratch: Learning Loops | Loops in Games

| | | |
|--|--|--|
| To design a simple catching game, making use of appropriate loops. | | |
| I can solve a problem by decomposing it into smaller parts. | | |
| I can design, write and debug algorithms to solve problems. | | |
| I can identify the three types of loops in Scratch and select the most appropriate loop for a particular task. | | |
| I can add a variable. | | |

Coding with Scratch: Learning Loops | Loops in Games

| | | |
|--|--|--|
| To design a simple catching game, making use of appropriate loops. | | |
| I can solve a problem by decomposing it into smaller parts. | | |
| I can design, write and debug algorithms to solve problems. | | |
| I can identify the three types of loops in Scratch and select the most appropriate loop for a particular task. | | |
| I can add a variable. | | |

Coding with Scratch: Learning Loops | Loops in Games

| | | |
|--|--|--|
| To design a simple catching game, making use of appropriate loops. | | |
| I can solve a problem by decomposing it into smaller parts. | | |
| I can design, write and debug algorithms to solve problems. | | |
| I can identify the three types of loops in Scratch and select the most appropriate loop for a particular task. | | |
| I can add a variable. | | |

Coding with Scratch: Learning Loops | Loops in Games

| | | |
|--|--|--|
| To design a simple catching game, making use of appropriate loops. | | |
| I can solve a problem by decomposing it into smaller parts. | | |
| I can design, write and debug algorithms to solve problems. | | |
| I can identify the three types of loops in Scratch and select the most appropriate loop for a particular task. | | |
| I can add a variable. | | |

Coding with Scratch: Learning Loops | Loops in Games

| | | |
|--|--|--|
| To design a simple catching game, making use of appropriate loops. | | |
| I can solve a problem by decomposing it into smaller parts. | | |
| I can design, write and debug algorithms to solve problems. | | |
| I can identify the three types of loops in Scratch and select the most appropriate loop for a particular task. | | |
| I can add a variable. | | |

Coding with Scratch: Learning Loops | Loops in Games

| | | |
|--|--|--|
| To design a simple catching game, making use of appropriate loops. | | |
| I can solve a problem by decomposing it into smaller parts. | | |
| I can design, write and debug algorithms to solve problems. | | |
| I can identify the three types of loops in Scratch and select the most appropriate loop for a particular task. | | |
| I can add a variable. | | |

Coding with Scratch: Learning Loops | Loops in Games

| | | |
|--|--|--|
| To design a simple catching game, making use of appropriate loops. | | |
| I can solve a problem by decomposing it into smaller parts. | | |
| I can design, write and debug algorithms to solve problems. | | |
| I can identify the three types of loops in Scratch and select the most appropriate loop for a particular task. | | |
| I can add a variable. | | |