DIGITAL SCHOOLHOUSE together with NINTENDO SWITCH

+44 (0) 207 534 0586
contact@digitalschoolhouse.org.uk

Teaching Guide.

Let's Play – Code Combat: Playing with Python

Introduction

Code Combat is a multiplayer game to help people learn to program. Through solving puzzles and defeating ogres players progress through the game levels to learn increasingly complex programming concepts.

This workshop uses the Code Combat resource to help set the context for the day. Pupils begin the workshop by registering and playing a few levels of the game for themselves, before being asked to apply their computational thinking skills to attempt to decompose the main elements of the game. The magic of computing is then introduced through magic tricks to teach sequencing and algorithmic thinking.

Pupils then return to the computer to apply their skills to move into Python and learn the basics of creating 'Hello World' style programs. A sample file is provided, and pupils are encouraged to decompose the given program and use logical reasoning to identify what the program does before making their own modifications to personalise it. Opportunities are then given to enable pupils to teach themselves some new skills based on suggested online tutorials. Through working with their peers they then use what they've learnt to create something new and inspirational. Before the end of the day the pupils return to Code Combat to complete the 'create your own level' challenge to help consolidate their learning for the day.

ukie PlayStation. SEGA Warwickshire County Council UBISOFT

+44 (0) 207 534 0586
contact@digitalschoolhouse.org.uk

Computing Programmes of Study Links

2. Pupils should be taught to:

2.1. design, write and debug programs that accomplish specific goals, including controlling or simulating physical systems; solve problems by decomposing them into smaller parts

2.2. use sequence, selection, and repetition in programs; work with variables and various forms of input and output

2.3. use logical reasoning to explain how some simple algorithms work and to detect and correct errors in algorithms and programs

Progression Pathway bands covered

ALG = Algorithms: Pink, Yellow, Orange, Blue

Reference

PA1	Understands what an algorithm is and is able to express simple linear (non-branching) algorithms symbolically.
PA2	Understands that computers need precise instructions.
PA3	Demonstrates care and precision to avoid errors
YA1	Understands that algorithms are implemented on digital devices as programs
YA2	Designs simple algorithms using loops, and selection i.e. if statements.
YA3	Uses logical reasoning to predict outcomes.
YA4	Detects and corrects errors i.e. debugging, in algorithms.
OA1	Designs solutions (algorithms) that use repetition and two-way selection i.e. if, then and else.
OA2	Uses diagrams to express solutions.
OA3	Uses logical reasoning to predict outputs, showing an awareness of inputs.
BA1	Shows an awareness of tasks best completed by humans or computers.

+44 (0) 207 534 0586
contact@digitalschoolhouse.org.uk

P&D = Programming & Development: Pink, Yellow, Orange, Blue

Reference

PP1	Knows that users can develop their own programs and can demonstrate this by creating a simple program in an environment that does not rely on text
PP2	Executes, checks and changes programs
PP3	Understands that programs execute by following precise instructions
YP1	Uses arithmetic operators, if statements, and loops, within programs.
YP2	Uses logical reasoning to predict the behaviour of programs
YP3	Detects and corrects simple semantic errors i.e. debugging, in programs.
PP1	Knows that users can develop their own programs and can demonstrate this by creating a simple program in an environment that does not rely on text
PP2	Executes, checks and changes programs
PP3	Understands that programs execute by following precise instructions
YP1	Uses arithmetic operators, if statements, and loops, within programs.
YP2	Uses logical reasoning to predict the behaviour of programs

+44 (0) 207 534 0586
contact@digitalschoolhouse.org.uk

Computational Thinking Strands

AL – Algorithmic Thinking

Ref.	Activity
A1	Writing instructions that if followed in a given order (sequences) achieve a desired effect
A4	Writing instructions that choose between different constituent instructions (selection) to achieve a desired effect;
A5	Writing instructions that repeat groups of constituent instructions (loops/iteration) to achieve a desired effect;

EV – Evaluation

Ref.	Activity
E1	Assessing that an algorithm is fit for purpose;
E2	Assessing whether an algorithm does the right thing (functional correctness);
E8	Assessment of whether a system gives an appropriately positive experience when used (user experience);

DE – Decomposition

Ref.	Activity
D1	Breaking down artefacts (whether objects, problems, processes, solutions, systems or abstractions) into constituent parts to make them easier to work with

+44 (0) 207 534 0586
contact@digitalschoolhouse.org.uk

Learning Outcomes

1. Able to progress through several levels of gameplay, thereby being able to control game characters through simple instructions.
2. Be able to use decomposition to identify the key aspects of the code combat game
3. Understand why instructions are important in computing
4. Understand what instructions can do within the code combat game
5. Understand what role instructions/algorithms play in the wider aspect of computing
6. Understand what is meant by the term ‘sequence’
7. To be able to write a sequence of instructions as an algorithm
8. To be able to write a sequence of instructions within Python
9. Be able to input simple instructions within Python
10. Be able to run an existing Python programme and identify what it does and how it is working by examining the code and running tests
11. Be able to edit and customise existing code
12. Be able to solve simple errors and within their code
13. Be able to use online tutorials to be able to self-teach new syntax and programming concepts
14. To be able to share what they have learnt with their peers
15. To be able to use what they have learnt to append and modify existing code in order to refine their solution
16. To be able to use what they have learnt to return to the Code Combat environment and create new game levels suitable for their peers to play

+44 (0) 207 534 0586
contact@digitalschoolhouse.org.uk

Session Overview

SESSION 1

Session Content / Activity	Resources Used	Prog. Pathway	Comp. Thinking	Computing POS Link
Welcome and settle pupils and explain objectives of the day	DSH_WelcomeIntroduction.ppt			
Introduce pupils to the Code Combat game. Encourage pupils to register so that they can save their gameplay and create their own levels later.	Let's Play Code Combat.ppt Codecombat.org		A1	2.2, 2.3
Allow pupils to spend time playing the game; they should aim to play at least 3 levels.	Let's Play Code Combat.ppt	ALG PA1, PA2, PA3, YA1, YA2 P&D PP1, PP2, PP3, YP2	A1	2.1, 2.2, 2.3
Encourage pupils to discuss aspects of the game, its objectives and how it works. In particular encourage them to discuss what is happening and why they are giving instructions to their characters.	Let's Play Code Combat.ppt	ALG PA1, PA2, PA3, YA1, YA3, OA3 P&D PP1, PP2, PP3, YP2	A1, D1	2.1, 2.3
Discuss the importance of commands, instructions and introduce the term algorithms. Encourage pupils to then play another few levels of the game, asking them to pay particular attention to the commands they are writing.	Let's Play Code Combat.ppt	ALG PA1, PA2, PA3, YA1, YA3, OA3 P&D PP1, PP2, PP3, YP2	A1	2.2

Digital Schoolhouse, 18a Blackbull Yard, 24-28 Hatton Wall, London EC1N 8JH.

+44 (0) 207 534 0586
contact@digitalschoolhouse.org.uk

Stop the pupils to deliver the 'Joker in the Pack' magic trick. Encourage them to try the trick for themselves without revealing the secret. Can they work out how the trick works? Then reveal the algorithm, discuss the importance of it and allow them to try the trick for themselves. If time, allow them to see if they can modify the algorithm to change the trick.	Let's Play Code Combat.ppt	ALG PA1, PA2, PA3, YA1, YA2, YA3, YA4, OA3, BA1	A1, D1	2.2, 2.3
Welcome and settle pupils and explain objectives of the day	DSH_WelcomeIntroduction.ppt			
Introduce pupils to the Code Combat game. Encourage pupils to register so that they can save their gameplay and create their own levels later.	Let's Play Code Combat.ppt Codecombat.org		A1	2.2, 2.3
Allow pupils to spend time playing the game; they should aim to play at least 3 levels.	Let's Play Code Combat.ppt	ALG PA1, PA2, PA3, YA1, YA2 P&D PP1, PP2, PP3, YP2	A1	2.1, 2.2, 2.3
Encourage pupils to discuss aspects of the game, its objectives and how it works. In particular encourage them to discuss what is happening and why they are giving instructions to their characters.	Let's Play Code Combat.ppt	ALG PA1, PA2, PA3, YA1, YA3, OA3 P&D PP1, PP2, PP3, YP2	A1, D1	2.1, 2.3
Discuss the importance of commands, instructions and introduce the term algorithms. Encourage pupils to then play another few levels of the game, asking them to pay particular attention to the commands they are writing.	Let's Play Code Combat.ppt	ALG PA1, PA2, PA3, YA1, YA3, OA3 P&D PP1, PP2, PP3, YP2	A1	2.2

+44 (0) 207 534 0586
contact@digitalschoolhouse.org.uk

Stop the pupils to deliver the 'Joker in the Pack' magic trick. Encourage them to try the trick for themselves without revealing the secret. Can they work out how the trick works? Then reveal the algorithm, discuss the importance of it and allow them to try the trick for themselves. If time, allow them to see if they can modify the algorithm to change the trick.	Let's Play Code Combat.ppt	ALG PA1, PA2, PA3, YA1, YA2, YA3, YA4, OA3, BA1	A1, D1	2.2, 2.3

SESSION 2

Session Content / Activity	Resources Used	Prog. Pathway	Comp. Thinking	Computing POS Link
Recap the magic trick and what they learnt. Reinforce the concept of sequences of instructions.	Let's Play Code Combat.ppt	ALG PA1, PA2, PA3, YA1, YA2, YA3, YA4, OA3, BA1	A1, D1	2.2, 2.3
Introduce pupils to Python. Once in the 'shell editor' encourage pupils to try entering data for themselves. Can they work out what input generates an error and what generates a valid output? Lead the pupils in a discussion to generate a list of valid inputs and what Python will accept.	Let's Play Code Combat.ppt Python IDLE	ALG PA1, PA2, PA3, YA1, YA2, YA3, YA4, OA1, OA3, BA1 P&D PP1, PP2, PP3, YP1, YP2, YP3	A1	2.2, 2.3
Using slides 16 – 18 encourage pupils to try their own calculations and use Python like a calculator. Can they work out what is happening? How is it working – what is happening behind the scenes in order for Python to be able to produce the results?	Let's Play Code Combat.ppt	ALG PA1, PA2, PA3, YA1, YA2, YA3, YA4, OA1, OA3, BA1 P&D	A1	2.1, 2.2, 2.3

+44 (0) 207 534 0586
contact@digitalschoolhouse.org.uk

		PP1, PP2, PP3, YP1, YP2, YP3		
Use slides 19 – 22 to introduce pupils to text entry in Python, allow them to discover which of the options on slide 21 is in the correct format. Why is it important for the exact syntax to be so accurate and correct? Why is an error generated if it is entered incorrectly? If we humans can understand it, why can't the computer? Because computers are extremely 'stupid' machines. They have to follow precise instructions written exactly in a way that the computer is programmed to recognise it, otherwise it won't recognise it at all. *"If you saw Santa wearing blue clothes and with a black beard would you still recognise it as Santa Clause?"*	Let's Play Code Combat.ppt	ALG PA1, PA2, PA3, YA1, YA2, YA3, YA4, OA1, OA3, BA1 P&D PP1, PP2, PP3, YP1, YP2, YP3	A1	2.1, 2.2, 2.3
Run the file 'Jokes.py', without giving the pupils any instructions tell them to run the program. What is happening? Can they guess the code behind game? Once pupils have had the opportunity to share their thoughts and try and work out what's happening, then show them how to open the file in edit mode. *(Right click Edit with IDLE)*	Let's Play Code Combat.ppt Jokes.py	ALG PA1, PA2, PA3, YA1, YA2, YA3, YA4, OA1, OA3, BA1 P&D PP1, PP2, PP3, YP1, YP2, YP3	A1, D1, E2	2.1, 2.2, 2.3
Using slides 24 – 27 give pupils a chance to familiarise themselves with the code. Does the code meet their expectations? Then encourage pupils to add to and edit the existing jokes and customise it for their own. Give them the opportunity to 'play' with the code and make it their own.	Let's Play Code Combat.ppt	ALG PA1, PA2, PA3, YA1, YA2, YA3, YA4, OA1, OA2 OA3, BA1 P&D PP1, PP2, PP3, YP1, YP2, YP3	A1, D1, E2, E8	2.1, 2.2, 2.3

+44 (0) 207 534 0586
contact@digitalschoolhouse.org.uk

By trying their peers modified programs, encourage pupils to test that their program is working the way they wanted it to and to provide critical feedback to their peers. Pupils should spend some time discussing end experimenting with possible modifications to their code For lower ability pupils you may wish to suggest some possible modifications for the pupils to explore and consider.	Let's Play Code Combat.ppt	ALG PA1, PA2, PA3, YA1, YA2, YA3, YA4, OA1, OA3, BA1 P&D PP1, PP2, PP3, YP1, YP2, YP3	A1, D1, E1, E2, E8	2.1, 2.2, 2.3

SESSION 3

Session Content / Activity	Resources Used	Prog. Pathway	Comp. Thinking	Computing POS Link
By providing pupils with a starting point of suggested online tutorials and possibly YouTube videos, encourage them to independently try out the tutorials to learn something new. For more able pupils give them free reign to explore and try out new syntax as they wish. This may be a good opportunity to introduce them to Trinket https://trinket.io/python/bbe92df381 Once pupils have had the opportunity to try some tutorials, encourage them to begin thinking about their Jokes program. Can they use something they have learnt to modify their program further? They may wish to try this out in pairs.	Let's Play Code Combat.ppt	ALG PA1, PA2, PA3, YA1, YA2, YA3, YA4, OA1, OA3, BA1 P&D PP1, PP2, PP3, YP1, YP2, YP3	A1, D1	2.1, 2.2, 2.3

+44 (0) 207 534 0586
contact@digitalschoolhouse.org.uk

Further develop the work done here by moving onto slide 28. If pupils have already been working in pairs, then encourage them to share with another pair. Can they teach each other something new? Can the use their new knowledge to modify their programs even further? Or perhaps by collaboratively writing an entirely new program all together.	Let's Play Code Combat.ppt	ALG PA1, PA2, PA3, YA1, YA2, YA3, YA4, OA1, OA3, BA1 P&D PP1, PP2, PP3, YP1, YP2, YP3	A1, A4, A5, D1, E1, E2, E8	2.1, 2.2, 2.3
The following activities can be considered an extension activity if there is time, or if there is a need to move on. Some pupils and classes may become completely engaged in the previous activity and therefore, you may wish to allow them greater time to work on their creations instead.				
Pupils should move back onto Code Combat and try some of the additional levels that develop alternative skills. They can also try to create their own levels for their peers to play. Encourage them to work this out for themselves.	Let's Play Code Combat.ppt	ALG PA1, PA2, PA3, YA1, YA2, YA3, YA4, OA1, OA3, BA1 P&D PP1, PP2, PP3, YP1, YP2, YP3	A1, A4, A5, D1, E1, E2, E8	2.1, 2.2, 2.3
As a plenary set a Think/Pair/Share activity to encourage pupils to share three new things they have learnt today.	Let's Play Code Combat.ppt			2.3

+44 (0) 207 534 0586
contact@digitalschoolhouse.org.uk

Files/Resources

Filename	Resource Type	Purpose/Description
Codecombat_trailer	Video	Video to provide a context and introduction to the game
Jokes.py	Python Source File	Sample file for pupils to explore and edit
Jokes-selection.py	Python Source File	Sample file for pupils to explore and edit
Let's Play Code Combat	PowerPoint	Main teaching PowerPoint for the workshop

PLEASE NOTE: The activities outlined in this workshop pack are a suggested outline of how the workshop can be delivered. It is envisaged that teachers will adapt the resources and the organisation of them according to the needs of their class.