Exploring Programming

Level 1

Typically, pupils should show evidence of being able to:

- View a variety of existing simple programs and take part in a teacher-led discussion about choosing controls or events to include in a program. Use simple commands to investigate basic movements or events. *This might include using arrow keys to move a simple robot or an on-screen sprite round a maze.* (Explore)
- Create a basic program with the teacher's help. *This might include suggesting instructions to move a simple robot or sprites, for example along a path.* (Express)
- Know that digital methods can be used to communicate. (Exchange)
- Talk about the instructions they used, led by the teacher. (Evaluate)
- Demonstrate their simple program with the teacher's help. (Exhibit)

Level 2

Typically, pupils should show evidence of being able to:

- Choose some simple instructions to include in a program. Use simple controls or instructions to carry out simple actions in a program. *This might include entering instructions to control an object on screen, for example using Scratch.* (Explore)
- Create a basic program with the teacher's help. *This might include creating a sequence of instructions to control the movement of a character in Scratch.* (Express)
- Identify and talk about how to use different digital methods to communicate. (Exchange)
- Talk about how to improve their program, prompted by the teacher. (Evaluate)
- Save their program and/or demonstrate it to the class or group with the teacher's help. (Exhibit)

Level 3

Typically, pupils should show evidence of being able to:

- Research existing programs and search for and choose pre-programmed subroutines to edit and include in their own program. Plan or edit existing subroutines to solve simple problems in a programming environment. *This might include exploring the instructions available in a programming environment such as Scratch, including using more efficient statements such as 'repeat'.* (Explore)
- Create a program. *This might include creating or editing sequences of instructions to solve a given problem.* (Express)

- Use a contemporary digital method to communicate or contribute to a supervised online activity. *This might include sending an email or making a post to a wiki, blog or discussion thread. The email or post might be to a teacher.* (Exchange)
- Make some modifications to improve their program. *This might include editing sequences of instructions.* (Evaluate)
- Save their program with a filename and/or demonstrate it to the class or group. (Exhibit)

Level 4

Typically, pupils should show evidence of being able to:

- Research and select assets, such as code snippets, images or backgrounds to use when creating their program. Plan or edit an existing subroutine to solve the problem set in the task brief. *Typically, pupils use a programming environment to create an algorithm using a combination of simple and more complex instructions.* (Explore)
- Create a solution to a problem using a flowchart or pseudocode and then write the program in the chosen programming language, demonstrating a clear understanding of the audience and purpose. *Typical, pupils independently create a sequence of instructions to solve the problem given in the task brief.* (Express)
- Use one or more contemporary digital methods to communicate, exchange and collaborate in supervised online activities. *This might include emailing the completed program as an attachment or making several relevant posts to a wiki, blog or discussion forum.* (Exchange)
- Use appropriate ICT tools and features to improve their program. *This might include changing a sequence of instructions or using more efficient structures, documenting improvements where required.* (Evaluate)
- Save the program in a named folder, upload it to a class portfolio or publish it online. (Exhibit)

Level 5

Typically, pupils should show evidence of being able to:

- Research, select and evaluate a range of assets for programming, with an awareness of when it is better to use self-made assets. Select an appropriate programming language or environment and plan the required subroutines to solve the problem set in the task brief. *Typically, pupils use a storyboard to plan a game, using a wider range of features such as multiple levels, events and actions.* (Explore)
- Create a solution that combines a range of subroutines to solve the problem defined in the task brief, using a flowchart or pseudocode. Write the program in the chosen language, demonstrating a clear understanding of the audience and purpose. *Typically, pupils plan and create several subroutines, combining these to solve the problem given in the task brief.* (Express)

- Use a range of contemporary digital methods to communicate, exchange and share their information and solutions, collaborating online with their peers. *This might include working online to collaborate on designing and creating suitable algorithms to solve the problem, or discussing online about which language structures to include in their program to improve efficiency.* (Exchange)
- Use the 'plan, do, review' cycle to improve their program for the audience and purpose defined in the task brief. *This might include designing, drafting and making adjustments to their algorithms, creating a set of test criteria and using built-in functions to trace errors in code. They might also ask peers to test their program and give feedback to generate suggestions for improvements.* (Evaluate)
- Organise, store and maintain the program files, using appropriate naming conventions, in a personalised area to showcase learning digitally across the curriculum. (Exhibit)

Level 6

Typically, pupils should show evidence of being able to:

- Research, select and edit a range of subroutines, referencing the sources and justifying how they will help create a solution for the problem in the task brief. Explore the effects of using alternative subroutines. *This might include experimenting with different solutions to a problem, for example using alternative sorting or searching algorithms.* (Explore)
- Create planning documents to outline the proposed solution to the problem, including using flowcharts or pseudocode for all subroutines. Identify user requirements and plan, develop and test a well-documented program targeted at a specific audience that includes a range of more sophisticated language features and subroutines. *This might include a solution involving several subroutines, clearly indented and commented code, and a user guide.* (Express)
- Use a range of contemporary digital methods to communicate, exchange and share their information and solutions, collaborating with peers, experts and end users. *This might include collaborating on the program design and development with peers, for example using a discussion forum, where contributors submit and discuss alternative draft solutions or amendments.* (Exchange)
- Justify the algorithms they chose to complete the programming task, the alternatives they considered and the process they carried out in producing the program. Justify how their program meets the requirements of the specified audience and purpose. *This might include describing and justifying the project management process, including using test criteria or data for planned testing, error checking and peer review.* (Evaluate)
- Organise, store and maintain their work in a personalised area to showcase learning digitally across the curriculum. *This might include using version numbers to name files and assets.* (Exhibit)

Level 7

Typically, pupils should show evidence of being able to:

- Research, evaluate and adapt the most appropriate sources of information from available software libraries, referencing their sources and justifying how their choices will help create a solution for the problem in the task brief. Plan connecting subroutines and identify main inputs, outputs and parameters required. *This might include analysing a range of solutions for accuracy or efficiency and using appropriate development tools, such as a debugger, to routinely test and correct their code, justifying their decisions based on their original intentions and feedback from experts and end users.* (Explore)
- Create a requirements specification and formal plan for the proposed solution. Use the most appropriate design methods, language and validation techniques to create a sophisticated program suitable for the audience and purpose. *This might include using several self-written subroutines, with good use of commenting and indentation to improve readability.* (Express)
- Exploit contemporary communication methods to exchange, share and collaborate on their developed ideas with peers, experts and end users, contributing to a collaborative global environment. *This might include creating and maintaining shared documents to collaborate on solutions to coding problems.* (Exchange)
- Use, routinely, effectively and as an ongoing process, built-in functions, formal code review and planned testing to trace errors in code, improve code for efficiency, functionality or accuracy, and measure against agreed project objectives. Review the project management process to evaluate the methods used, including the testing process, and introduce changes to improve on these. (Evaluate)
- Manage and present a logically structured digital bank of work to showcase program development, taking account of format, portability, size and versioning. *This might include publishing the program, user guide and technical guide in a format suitable for distribution.* (Exhibit)

Pupils should demonstrate, when and where appropriate, knowledge and understanding of e-safety, including acceptable online behaviour.