A SELF-STUDY APPROACH TO LEARNING

The intention with Craig'n'Dave programming resources is that students make progress independently. New concepts are introduced through practical activities by engaging with sample code. Open-ended problems are sufficiently differentiated with a points-based system to enable all learners to make progress independently at their level. There is no need for a teacher to stand at the front of the class and teach the keywords and concepts in each objective in a traditional way. Students should make progress through the resources on their own, at their own pace. The role of the teacher is to maintain that pace, provide individual interventions when students are stuck, review completed objectives and track progress.

Students that are absent from class are not disadvantaged and can even continue their work at home. If a question is too difficult to answer, or a problem is too difficult to solve these can be left to be discussed with the teacher. The student can still move on to the next objective or problem.

FUNCTIONS FIRST

The use of comments and sensible variable names is expected from students from the outset. It therefore seems odd not to also introduce structured programming using subroutines from the start too.

This does undoubtably mean the initial small programs students write become unnecessarily complex, but good habits introduced early are not so easily forgotten.

Although avoiding input and output within a function is desirable, for simplicity this is sometimes used. It is also regularly seen in exam mark schemes.

PRIOR KNOWLEDGE

Craig'n'Dave resources assume no prior knowledge, so they are suitable for all learners regardless of their programming experience. Students that have studied a text-based language at Key Stage 3 and/or GCSE will no doubt find the objectives familiar, although it is less likely they have adopted a functions first approach.

The Python resources are written with GCSE in mind, with C# being more suitable for A level. It is extremely unlikely that a GCSE student will complete every problem presented in these resources due to the volume of work and because of the differentiated approach. At A level it is suggested students use the same resources in an unfamiliar language, perhaps undertaking some of the problems they did not solve at GCSE.

.

NAMING CONVENTIONS

Although experienced programmers use different naming conventions for different purposes within a program, such as PascalCase, camelCase, snake_case and kebab-case, Craig'n'Dave resources use PascalCase throughout for consistency.

With PascalCase, each letter of a new word in an identifier is uppercase.

BREAKS AND OPTIMAL SOLUTIONS

Commands exist in many languages to break out of iterations or jump to new sequences, but these are not considered good practice for creating structured code. Therefore break, goto and equivalent commands are never used.

Many programs can be written more elegantly with modern functions and frameworks. However, this is an introduction to programming that also prepares students for the algorithms they will see in exams. Therefore, logical instead of optimal code is often used. E.g. using .sort() in answer to a question about sorting algorithms gains no marks!

OLD AND NEW APPROACHES

It is recognised that different exam boards exemplify slightly different keywords and approaches that students could be using when coding. An example is operator overload concatenation or string formatting.

```
Sol = 299792458
OUTPUT("The speed of light is " + Sol + "m/s")
```
older approach and less performant

Could also be written like this:

```
Sol = 299792458
OUTPUT("The speed of light is {0} m/s",Sol)
```
contemporary approach and more performant

The Craig'n'Dave approach is to use a coding style that primarily matches exam board expectations but with more modern approaches being a consideration too. These are often not easy decisions to make when there are many possible ways to write a program! We introduce students to both operator overloading concatenation (in objective 1) and string formatting (in objective 3).

KEY TERMINOLOGY

"The syntax error was because of a missing string qualifier before the operator in the statement that concatenates the two variables."

Students become far better programmers if they understand what statements like this mean. In teaching we are encouraged to consider reading age and simplifying language for comprehension. With programming, by having a commanding vocabulary of the words associated with code, not only does it enable students to understand other programmers and articulate their approaches, but it also helps to understand new concepts. For example, if you know that an operator can add two integer variables, then understanding overloading operators becomes easier too.

Key terminology is used throughout Craig'n'Dave resources with each new term is explained and summarised.