ORAL FEEDBACK FRAMEWORK

Not all these questions will be suitable for all programs. Be selective.

ALL STUDENTS:

1. Approach

   To what extent does the program meet its objective?

   a. Has the student created a program that runs and produces the desired output?

   b. Has the student used the most appropriate code structures?

      E.g., *match* vs *if*, *for* vs *while, list* vs *variables*.

2. Comprehension

   To what extent does the student understand the code they have written?

   a. Can the student answer the questions in the "investigate" section?

   b. Can the student answer questions about their code?

      (See program comprehension framework overleaf)

3. Maintainability

   To what extent and how consistently has the student used best practices in creating readable code?

   a. Has the student used subprograms and whitespace?

   b. Has the student used comments to explain each subprogram, condition & iteration?

   c. Has the student used sensible identifier and subprogram names?

STRONGER STUDENTS:

4. Scalability

   To what extent is the solution suitable for an increased data set?

   a. Has the student created reusable program components?

      E.g. local variables, parameter passing & return statement.

   b. Has the student used appropriate data structures?

      E.g. arrays/lists instead of multiple variables?

   c. Is the solution time and space efficient?

5. Robustness

   To what extent is the program easy to use?

   a. Has the student used validation to prevent the program crashing?

   b. Has the student made the program user friendly?

      E.g. accepting both upper & lowercase inputs.

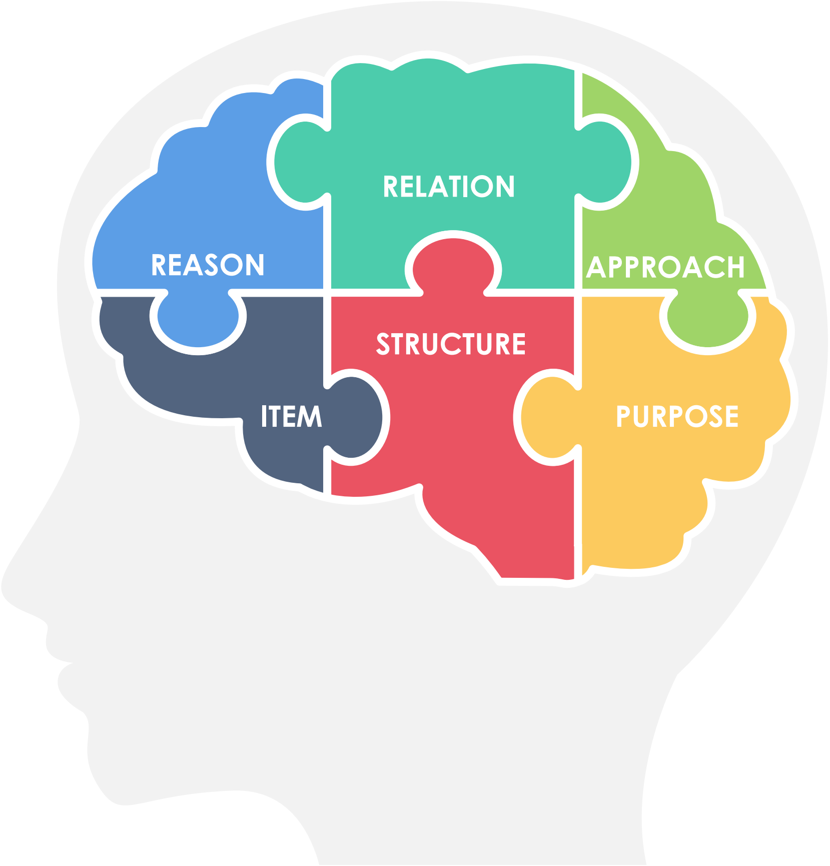
   c. Has the student prevented exception errors?

      E.g. file not found.

PROGRAM COMPREHENSION FRAMEWORK

Ask students questions about their programs.

- Lower order questions: item structure & purpose.
- Higher order questions: reason, relation & approach.



ITEM

Programming terminology.

STRUCTURE

Syntax of lines of code.

PURPOSE

What the item or structure achieves, returns, or outputs.

REASON

The reason why an item or structure is used.

RELATION

How items or structures relate to each other.

APPROACH

Evaluating the merits of alternative ways of solving the problem.

ITEM	Can you identify in your code an example of a/an:	• Argument • Constant • Data type: string, integer, float, Boolean • Identifier • Iteration • Operator • Parameter • Qualifier • Reserved word • Selection • Sequence • Variable
STRUCTURE	Can you explain the syntax of the command at line…	What are the arguments of the function for? Why are brackets or quotes used?
PURPOSE	Can you explain what the lines of code between line x and y do?	
REASON	Why did you use a…?	
RELATION	What would happen if you moved the code from line x to be on line y instead? Can you explain the relationship between this argument and this parameter?	Exploring local and global variables, code within and outside loops.
APPROACH	What alternative approach could you have taken with this section of code?	Explore efficiency and scalability.