TECHNIQUES

In assessing **INGENUITY**, I make an on-balance assessment of:

- Structure and layout of your code is logical and easy to follow
- Comments in-line code (ideally following the conventions for your programming language)
- Names are meaningful and follow a consistent scheme (variables, methods, classes)
- Use of Input validation techniques
- Data is well structured (eg: use arrays/objects to avoid names like *player1, player2, player3*)
- Use of global variables is minimal or avoided altogether. Where used they have been well justified.
- Use of error checking to "fail gracefully" (file exceptions, network exceptions, user input exceptions etc)
- Use of presentation and logic code separation where practical (model-view-controller or similar)
- Use of an elegant abstraction of classes, properties and/or methods.
- Use of algorithmic thinking that is elegant &/or efficient in their operations. eg: You don't perform O(n^2) operations when O(n) is feasible.
- Use encapsulation to avoid dependence on, or creation of side-effects
- Functions are sufficiently modular so each achieves one task

In assessing **COMPLEXITY**, I make an on-balance assessment of:

- Use of files and/or databases to save/load persistent data
- Use of remote API's (such as REST APIs, socket connections)
- Use of imported functionality from 3rd party modules and/or libraries
- Use of dynamic data structures where appropriate (arraylists, stacks, queues, linkedlists, binary trees, objects)
- Use of key-value pair dynamic data structure (Hashtables in Java, Dictionaries in Python, serialised/deserialised JSON)
- Use of object orientated design where appropriate
- Use of multidimensional arrays and/or collections
- Use of event handlers, call backs, &/or promises
- Use of complex presentation frameworks or templating languages such as Cordova, Electron, Kivy, React, Angular, Handlebars, Jinga2
- Use of multiple programming languages (eg: Python or Java backend, non-trivial Javascript front end)
- Use of GPU programming, parallel processing, or multiple threads (if appropriate to the problem)
- Use of nested control structures where appropriate

Credit for these items is only awarded where the accompanying written work documents the use of the technique. No credit is awarded for techniques used but not described.

The more of these skills you are able to demonstrate (and articulate through your Criterion C document) through your IA the better.