Portfolio 3 – Refactoring the Route Constructor

To cope with different format logs, gpxp::Route::Route() will need extending or overloading; before adding new functionality, the chief software architect has decided to refactor the constructor.

Introductory exercise

You should review the class (both data & function members) and consider the following

- Are the data members sufficient or does extra data need saving or is it just a different construction process?
- How should the existing constructor be modified to facilitate the extension – *look for abstractions*

The main task

The refactoring exercise will be the basis of the P3; it is worth 15% of the overall module credit.

The next phase of development involves extending the system so that GPS logs in different formats can be processed. Currently only GPX format is recognized, but the intention is to extend this to include one or more NMEA sentence types. This will involve modifying the Route constructor gpxp::Route::Route(std::string), but code review has indicated that better modularity would make integrating further formats more straightforward and the code more robust. It is the style and efficacy of the new modularity that will be assessed.

You should present your refactored code (the constructor, together with any new or revised functions) and suitable annotations. These should be the comments in the code, but they should not be 'regular' comments – the intention is to commentate on the revisions, which should refer to the changes and the reasoning behind them. Include abstraction by specification as well.

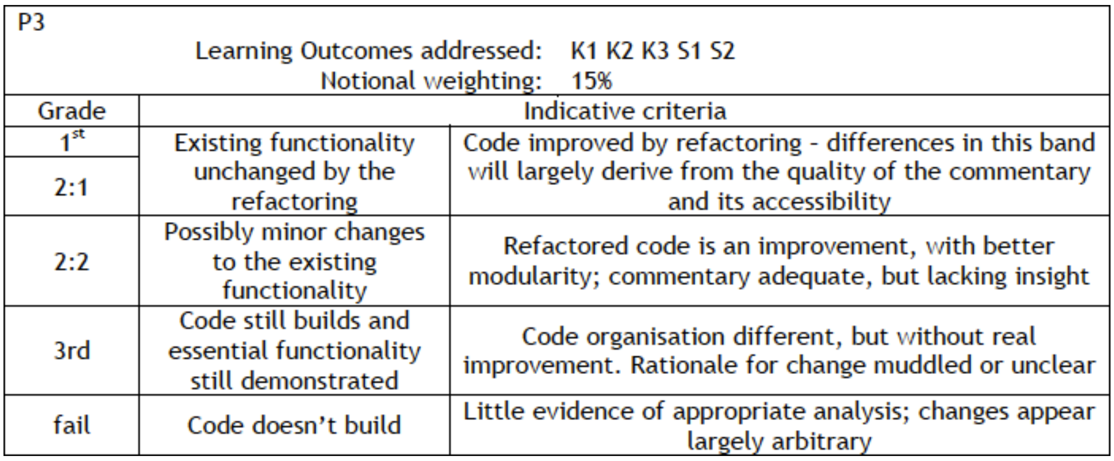
This can be individual work but it is probably most productive if done in pairs.

Submission details

Submit only the gpxp.cpp file. It should be uploaded to the ELP by 13 January 2017 (latest by 5pm).

Assessment details

This item is worth 15% of the module marks, allocated as a single grade; satisfactory refactoring is necessary, but the commentary in the report will generate most of the differentiation in the pass grades.



P3

Learning Outcomes addressed: K1 K2 K3 S1 S2

Notional weighting: 15%

Grade	Indicative criteria	
1st	Existing functionality unchanged by the refactoring	Code improved by refactoring - differences in this band will largely derive from the quality of the commentary and its accessibility
2:1		
2:2	Possibly minor changes to the existing functionality	Refactored code is an improvement, with better modularity; commentary adequate, but lacking insight
3rd	Code still builds and essential functionality still demonstrated	Code organisation different, but without real improvement. Rationale for change muddled or unclear
fail	Code doesn't build	Little evidence of appropriate analysis; changes appear largely arbitrary