|  |  |  |  |
| --- | --- | --- | --- |
| **University of Chester -** Undergraduate Programmes Assignment Specification  **Faculty of Science, Business & Enterprise**  **Department of Computer Science** | | | |
| **Module No**  CO5625 | **Module Title**  Further Programming and Algorithms | **Assessment No**  1 | **Weighting**  20% |
| **Submission Date**  11/01/24 at 17:30 | | | **Feedback due by**  08/02/24 |
| **Assignment Title**  Prototype | | | |
| **Learning Outcomes Assessed**  2. Analyse problems and design suitable solutions to them using real-world scenarios | | | |
| **Submission Information**  The TurnItIn submission box will have multiple parts. You must submit to the appropriate part   * A PDF file with all programming code from your project (in a monospace font), followed by a reference list in APA format. * A ZIP file containing the project. **This must be reduced to a manageable size using the scripts provided with the template project**   Both files must be named with your assessment (J number), e.g. J123456.pdf and J123456.zip. The name for each entry on TurnItIn must also be your assessment number.  Files submitted in an incorrect format will usually be marked as zero.  All components must be submitted to avoid receiving a mark of zero.  Any late work penalties for assignments will be calculated using the latest submission date/time. | | | |
| **Extensions**  Extensions should be requested through the online system available on the Registry services pages on [Portal](https://v3apps.chester.ac.uk/oed/#!start). Late work is penalised at the rate of 5 marks per day or part thereof.  **Academic Conduct** The material you submit must be your own work. Please avoid colluding with peers on your work. The penalties for breaching the academic conduct policy are severe. The minimum penalty is usually zero for that piece of work. Further information is available at Portal > Support Departments > Academic Services > Academic Standards > Academic Conduct > Information for Students > [Academic Conduct](https://portal1.chester.ac.uk/aqs/Pages/aqss-academic-conduct-information-students.aspx) | | | |
| **Generative AI The use of generative AI tools where not permitted will be treated as a breach of the academic integrity policy.**  **This assignment does not permit the use of any generative AI tools, including but not limited to ChatGPT, Bard, Copilot, Midjourney, and others.** | | | |
| **Referencing code**  Code adapted from third parties must be clearly referenced using comments to denote the start and end of the adapted code. You must also include an APA format reference in the PDF file.  **Example of referenced code**  *//code adapted from Thomson, 2012* **if** (someCharacter == **'z'** || someCharacter == **'Z'**) {  someCharacter -= 25;  } **else** {  someCharacter += 1; } //end of adapted code  **Example of reference entry in PDF file** Thomson, C. (2012). Rot-13 function in Java?. Stackoverflow. Retrieved October 25, 2021, from <http://stackoverflow.com/questions/8981296/rot-13-function-in-java> | | | |

**Assignment Brief**

You are required to implement a prototype quiz/game in which the user is repeatedly presented with the name of a river and must guess which of four continents it is in (Europe, Asia, Africa or South America). At the prototype stage, the following functionality should be implemented:

1. The program should read the four text files provided, containing rivers from each continent and store the data using a suitable data structure. **All file reads must be implemented in the constructor method of the River class** in accordance with the RAII design pattern. File reads must not occur elsewhere.
2. Implement a method which returns a randomly chosen river
3. Implement a method which returns the correct continent for a given river. This should return the empty string if the input is not a valid river.
4. Implement a method which takes two river parameters and returns a Boolean indicating whether or not they are in the same continent. This should return false if either parameter is not a valid river.
5. Implement a method which will provide the user with input and output for a single round of the game. This should include:
   1. The names of two rivers output on separate lines. These should be the first two lines of output and nothing else should appear on either of them.
   2. An opportunity for the user to enter their response (user instructions are already provided by a template UI so there is no need for any additional prompts here).
   3. The words ‘correct’ or ‘incorrect’ after user input. Inputs other than ‘s’, ‘d’ or ‘q’ should be considered incorrect. Scores and totals should be updated appropriately.
   4. The name of the correct continent for each river.
   5. The method should return false if the user quits (‘q’) and true otherwise

Examples of acceptable output can be found at the end of this brief. Keep it simple.

1. Implement further methods to return the score (number of correct answers) and total number of questions asked, as well as to reset the scores at the end of a game.

A template Visual Studio project will be provided with the necessary text files included, as well as the starter code for the project. This project has a ‘Feedback’ configuration which provides you with details of any issues you need to fix as well as awarding a provisional mark out of 75 for requirements 1-6. **It is unlikely that this mark will be changed after submission but the markers reserve the right to change it if you have done something particularly unusual or unexpected with your code**.

The game should work seamlessly once all the issues are resolved so you should spend most of your time running the feedback configuration. You should notice when running the debug configuration that the template project has an embryonic UI in place. Only the methods in parts 5 and 6 need to be completed to finish the quiz UI.

Implementation code for 1-4 should be written in the rivers.cpp and rivers.h files. Implementation code for 5 and 6 should be written in the game.cpp and game.h files. **In part 5, it is very important that you use the two parameters *in* and *out* instead of *cin* and *cout* for the playRound() method**. The relevant tests will fail and UI marks will not be awarded otherwise.

Each method signature in “rivers.h” contains a comment which you must complete, identifying the worst case time complexity of each of the methods as you have implemented them. You should complete these comments using O notation, where n is the number of rivers in the quiz (in other words, you can consider a river’s name to have a constant worst case length).

**Assessment Criteria**

Marks will be awarded according to the following scheme:

**Data class implementation(50%):**

Marks will be awarded as detailed by the feedback configuration, assuming issues have been resolved by reasonable means.

**UI implementation (25%):**

Marks will be awarded as detailed by the feedback configuration, assuming issues have been resolved by reasonable means.

**Time complexity (25%):**

Time complexity is indicated correctly for each of the 4 required methods (20%)

Method implementations are reasonably efficient in all cases (5%)

**Note that in general, regardless of feedback or any other outcomes, you will fail if your program does not read the data from the files provided.**

Example of acceptable UI outputs (template output in grey, student output in white):





