# Effective C++

## Performance Assignment

### This summative assessment is designed to test your ability to develop solutions to real-world problems. It assesses your skills in applying the techniques covered in the module and the wider course, i.e. performance considerations, multi-threading and use of C++ generally.

This assignment is individual work and contributes 40% of the overall coursework mark for this module.

### Assignment Overview

The application provided (see BlackBoard) performs a sequence of image processing tasks on a collection of bitmap images.

You are to:

1. Enhance this application so that it performs additional image processing tasks (see below).
2. Optimise your application so that it executes as fast as possible. Your application MUST process the images as fast as possible and in real time.
3. Supply an explanation of the **code structure**, **its operation** and a **reflection on which elements of your course have informed your choice of solution for this assessment**. For this, describe the major functions and any classes you construct and what these contribute to the application, as well as any major language features that have also contributed. Also describe any algorithms employed and major design decisions. Text and diagrams are acceptable. For your reflective component, please describe any relevant modules and why they have been useful in helping you arrive at a solution to this problem.

### Additional Image Processing Tasks

As it stands, the application applies a simple brightening function to each image. The application must perform this as well as the following tasks:

1. Rotate each image clockwise by 90 degrees.
2. Convert each image to greyscale (i.e. remove the colour information to make the image appear as though it was shot with black-and-white film).
3. Scale each image to reduce its size by half, in both horizontal and vertical dimensions. To do this, reduce the size of the image then apply the **bilinear filtering algorithm** to produce a resulting image that is a good quality interpretation of the original.

### Additional Information

The image filenames are hard coded into the application as text strings. The application then works its way through these, brightening the images and saving the results. The source files are JPGs and the destination files (after processing) are PNGs. The images are of different dimensions (width and height) and this needs to be taken into consideration when processing the files.

Your application MUST process ONLY these files. Do not substitute different images in place of these. The order in which these are processed and the order in which the image processing tasks are performed is up to you.

You can apply any C++ optimisation strategies to improve the performance of your application. You can also use assembly language or GPU processing, as well as C++, if you wish.

The skeleton source code contains some performance measuring routines that will be used to measure your application's execution time. Do not alter this part of the program.

You may use third-party libraries. If you do this, you MUST provide references and describe where and when your application depends on these. Missing references will be treated as plagiarism.

### Competition and Mark Allocation

If your application correctly and successfully implements the four required image processing tasks, then it will be subjected to a performance test. It will be executed and compared to other assignment submissions (which are equally correct and successful) and marks will be awarded in relation to how well it performs with respect to the fastest application.

Please make sure your code is commented and easy to read – if the structure of your code is incomprehensible then you may be disqualified.

### Overall Mark Allocation

|  |  |
| --- | --- |
| Correct image rotation | 2 |
| Correct greyscale conversion | 2 |
| Correct image scaling | 2 |
| Implementation of bilinear filtering | 4 |
| Marks available for the performance competition | 20 |
| Explanation of code structure and operation  Marks will be awarded for depth of explanation in terms of algorithms used, general structure of source code, major functions/classes, function call hierarchy, use of STL and run-time issues. | 10 |
| Total | 40 Marks |

### Submission Check List

Please check you have submitted the following:

|  |  |
| --- | --- |
| Source code and project files on disk, compilable for Visual Studio 2013 on University Machines. Ensure this compiles and works. |  |
| Summary of code structure (functions, classes, and algorithms). |  |

Ensure that your submission (DVD, CD, Dropbox or other method) is free from errors, and that your code compiles successfully.

MAKE SURE THE APPLICATION COMPILES AND EXECUTES ON UNIVERSITY COMPUTERS. **If you use third-part libraries, make sure ALL required files are included.** Failure to submit the required files, or if your application will not compile on University machines, WILL result in zero marks being awarded.

### Hand-in Date

Friday 22nd April 2016

# Effective C++

## Performance Assignment

## Marking and Feedback

### This formative assessment is designed to test your ability to develop solutions to real-world problems. It assesses your skills in applying the techniques covered in the module so far, i.e. performance considerations, multi-threading and object-oriented design.

|  |  |
| --- | --- |
| **Student Name:** | **Mark** |
| Correct image rotation | 2 |
| Correct greyscale conversion | 2 |
| Correct image scaling | 2 |
| Implementation of bilinear filtering | 4 |
| Marks available for the performance competition | 20 |
| Description of code structure and operation | 10 |
| **Total** |  |