Coursework commentary 2018–2019

CO3325 Data compression

General remarks

This course offers two coursework assignments. Both are technical and require programming implementation.

The coursework assignments offer instructions and guidelines. A good submission should follow these closely.

This final year technical module relies upon your knowledge and skills gained from previous years. Successful submissions often report that, as suggested, students should:

- complete exercises to understand the relevant concepts before attempting the coursework assignment tasks
- review the software engineering approaches (see subject guide CO2226 Software engineering, algorithm design and analysis, Volume 2) in order to complete a software development journey as a professional would
- divide the coursework assignments into smaller tasks and complete them in a series of stages with milestones such as:
    1. Working on the given problem specifications → designing rough solutions in block diagrams.
    2. Designing fine solutions in flowcharts →finalising the design with general analysis in class diagrams or pseudocodes.
    3. Testing.
    4. Evaluation of the software.

High quality coursework reflects good efforts. It should be well-presented, starting with a considered design, offering comprehensive technical details and justification, and making wise choices of data structures and sensible decisions for efficient approaches to problem solving. The report should be logically structured and clearly written with insightful discussions on technical issues. The programming style should be neat with sufficient comments and explanation, and screenshots to highlight the execution process of the programs.

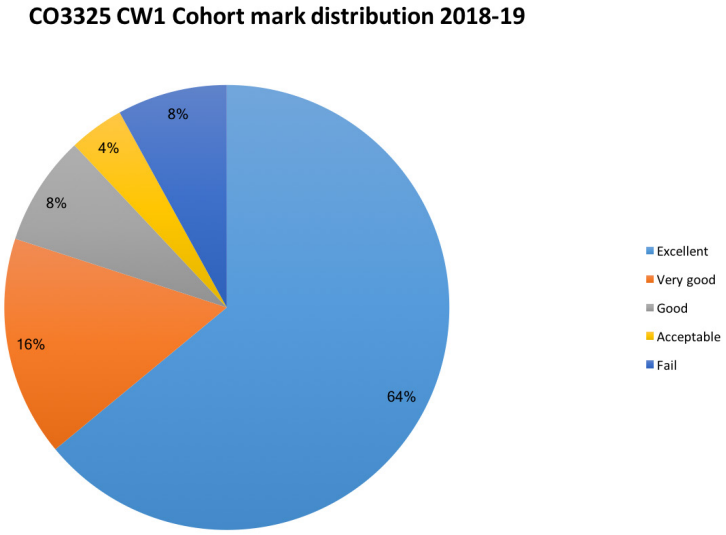
Coursework commentary 2018–2019

CO3325 Data compression

Coursework assignment 1

The first coursework assignment focuses on implementation of the LZW compression algorithms.

See 2018–2019 CW1 cohort mark distribution below:



Comments on specific questions

It was a great pleasure to see more outstanding courseworks again this year. In addition to a general high standard of submissions, extra efforts were obviously made to extend some aspects of the coursework. For example, studies and implementations included new algorithms, some efficient data structures, original work from scratch, and automatic demonstration.

Most students successfully implemented the LZW algorithms, and some explored alternative data structures for the dictionary. They not only gained extra marks, but also an interesting and rewarding learning experience beyond the textbooks. For example, a student wrote: '*It's the first Java program I've made that I feel really proud of.*'

A few students found programming difficult, at least at the beginning. They were honest about their lack of confidence in their programming skills. Some even said that it was the first time they had developed such a program from scratch.

However, many students improved their programming skills from this coursework assignment. They wrote, for example, *'there [have] been many struggles particularly to have a concrete structure of how the program is to be buil[t]. In my time doing this assignment I have noticed that doing the encoding and decoding manually written on a rough paper first helped me understand a lot on how the logic of the program has to be, the flowcharts and the UML Class diagram has also helped me greatly in this assignment as it gave me [a] strong*

idea on how the program has to be structured and coded.'

Almost all the students who attempted the coursework assignment concluded that they had learnt a lot, both in understanding the compression algorithms and in software development. For example, one recalled, *'Implementing the LZW prototype was a little more challenging [than] was expected. The difficulty level might have been initially overlooked due to the simple appearance of the algorithms. Completing the java program however was rewarding and enlightening.'*

One student found the assignment a little vague with only a few explicit questions, and no specified format requirements. This, however, is quite common in real-world practice. Customers often do not know what they require precisely, and it is part of a programmer's job to help them design a specific use-case and requirements for the project system. This assignment required many decisions to be made, and a good submission offers justification for these decisions.

Understanding is a prerequisite of programming. One student wrote, '*For me the course work was very straight forward but firstly I needed to fully understand dictionary encoding to begin designing my program. I used my study guide for all my research and referenced the pseudo code for my implementation. Coding and testing the program was really interesting and fun to see how the program would react to different input[s] and provide different outputs for encoding and decoding. I have no questions overall, I thought this was a really nice coursework.*'

One very rare mistake made this year was that a student submitted a coursework assignment for another module. This results in scoring a mark of 0. You should **always double check** that the file you have uploaded is the file you meant to upload.

Coursework commentary 2018–2019

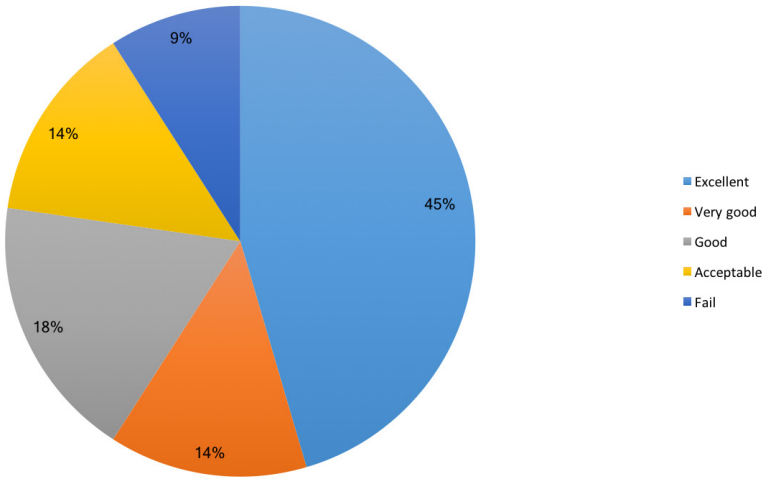
CO3325 Data compression

Coursework assignment 2

The second coursework assignment is dedicated to the performance of different compression algorithms.

See 2018–2019 CW2 cohort mark distribution below:

CO3325 CW2 Cohort mark distribution 2018-19



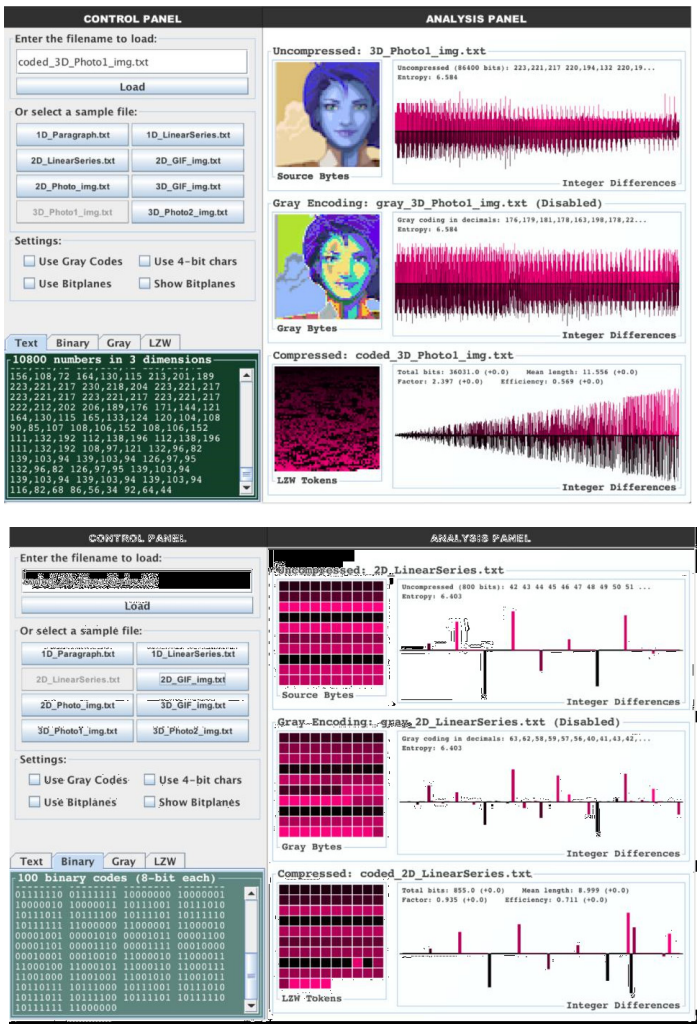
Comments on specific quesitons

This coursework assignment seemed to be more attractive to creative students. Students reported to have spent more time on Coursework assignment 2 than they did on Coursework assignment 1, reflecting their interests, effort and dedications. Many students went the extra mile to learn from their own experimental program(s). They learnt not only compression but also programming and problem solving. They wrote, for example, *'In this assignment I built the most advanced program of my academic career while learning how Gray codes greatly improve compression performance when implemented using bitplanes.'*

Some students seemed to be short of time. In this case, students could just use their own compression programs developed in Coursework 1, and implement another compression algorithm only if time permitted.

A good program design could start from the user interfaces. For example, it is wise to decide first what the input and output would be. Most students used the input.txt and output.txt auxiliary files for their compression system. This is a good choice due to flexibility.

Quite a lot of students were very creative indeed. They applied the data compression ideas nicely on various compression problems, asked their own questions, identified new sub-problems, and found their own answers and solutions. Some focused on demonstration programs as a proof of concept. For example, the following figures are copied from an excellent piece of coursework by Alexis Parizeau:



In the above work, the student went the extra mile. Not only were the experiments carried out as required, but a convenient tool was also developed as a by-product for repeating this work and conducting more experiments for the research community.

The use of graphics is helpful for the design. For example, one student wrote:

'I found using flowcharts made scripting the algorithms that much easier to follow the sequence of events needed. I found these of value in ensuring I captured the necessary elements of my methods and class.'

Students who attempted the Coursework assignment 2 generally felt that their programming skills had improved. Again, almost all the students said that they had learnt a lot from completing the coursework assignment, both in understanding the compression algorithms and in software development. They wrote:

'Throughout this assignment I was able to do a lot of research before implementing anything and then [to] be able to do what I enjoy doing most by being creative.'

'Overall this assignment gave me a much stronger background in data compression and also in programming as it gave me the privilege to work on many different areas to construct a lossless compression using java.'

'Overall a very fun and interesting assignment. Felt very open-ended which gave room to exercise our creative outlet.'