

COURSEWORK ASSESSMENT SPECIFICATION

Details of Module and team

What Learning Outcomes are assessed?

What are my Deadlines and how much does this assessment contribute to my Module Grade?

What am I required to do in the assessment?

What are my assessment criteria? (What do I have to achieve for each grade?)

Can I get formative feedback before submitting ?
If so, how?

How and when do I submit this assessment?

How and when will I get summative feedback?

What skills might this work evidence to employers?

Work handed in up to five working days late will be given a maximum grade of 3LOW whilst work that arrives more than five working days will be given a mark of zero. Work will only be accepted beyond the five working day deadline if satisfactory evidence, for example, an NEC is provided.

Guidance on NEC submission and the appeals process can be found at

https://ntu.ac.uk/current_students/resources/student_handbook/appeals/index.html.

The University views **plagiarism and collusion** as serious academic irregularities and there are a number of different penalties which may be applied to such offences. The [Student Handbook](#) has a section on Academic Irregularities, which outlines the penalties and states that **plagiarism** includes:

'The incorporation of material (**including text, graph, diagrams, videos etc.**) derived from the work (published or unpublished) of another, by unacknowledged quotation, paraphrased imitation or other device in any work submitted for progression towards or for the completion of an award, which in any way suggests that it is the student's own original work. Such work may include printed material in textbooks, journals and material accessible electronically for example from web pages.'

Whereas **collusion** includes:

"Unauthorised and unacknowledged copying or use of material prepared by another person for use in submitted work. This may be with or without their consent or agreement to the copying or use of their work."

If copied with the agreement of the other candidate both parties are considered guilty of Academic Irregularity.

Penalties for Academic irregularities range from capped marks and zero marks to dismissal from the course and termination of studies.

To ensure that you are not accused of plagiarism, look at the sections on [Plagiarism Support](#) and [Turnitin support](#).

I. Assessment Requirements

Introduction to the Report

The aim of this coursework is work through the encoding, decryption and breaking of the RSA (Rivest, Shamir and Adleman) algorithm. This will be done through answering four questions, each of which carries an equal weighting towards your final mark. Your submission will take the form of a report in which you answer each question in turn. You will also be required to do some coding in MATLAB.

Submission Details

The assessment consists of a **single .pdf** submission to **Dropbox**, which should contain a report, followed by a copy of your MATLAB code as an appendix. Your work will be submitted to Turnitin as a means to check for plagiarism. **MATLAB code should be typed into the appendix to allow it to be checked by Turnitin. Do not submit screenshots from MATLAB or .m files as part of your report.** Included in the NOW Assessment folder is a sample *Question 0* with model solution which can be used as a template for your own work. Do not submit Q0 yourself.

You should use the **RSA Testing** Dropbox folder to make a test submission and check your Turnitin report (noting that the report can take a few hours to compile so do this early). The final submission should be placed in the **RSA Final Submission** folder. You may only submit once to the testing folder. You can submit more than once to final submission folder but you may only submit one file and this will overwrite any previous submission.

Questions 1-3 all require MATLAB coding. In each case the parts a), b), c),... will tell you what MATLAB code you need to write and parts i), ii), iii),... will explain what commentary you should look to include in your report. Non-submission of code for a given question, or incorrect code, will result in no credit being awarded for any answers in the commentary section that are directly related to having working MATLAB code.

The final report should not exceed four pages of typed A4 (11pt font size with 2.4cm margins). To achieve this, it is easiest to type up your report in Microsoft Word and save your final submission as a PDF. You are also welcome to use LaTeX but be aware that the page limit above, with formatting restrictions still applies and it is your responsibility to make sure that you meet these restrictions. However, **you must submit a .pdf.** *If you submit any other file type, you will be automatically docked one grade.*

All of your MATLAB code should be included as an appendix at the end of the report and should be separated on a question-by-question basis. All code should be given as MATLAB functions. The appendix will not contribute to your four page limit. You may (and are encouraged to) use external resources as long as they are referenced. Again, your references list does not contribute to the four page limit.

Referencing

Aspects of this assignment, particularly Question 4, will require you to use external resources. You are encouraged to use academic resources (papers, textbooks) where possible. Google Scholar is a great

place to search for academic material. Note, in particular, that whilst Wikipedia is a useful resource, it is a secondary resource. Rather than citing Wikipedia as a reference, use the citations on Wikipedia pages to guide you to original sources for you to reference.

You are expected to do your referencing in an academic style (Harvard or numeric). If you feel uncomfortable with referencing, you should seek guidance from the library at the earliest possible opportunity.

To encourage proper referencing, the following caps will be enforced:

- Submissions with references in a non-academic style will be capped at a **21HIGH**
- Submissions with a references list, but where references are not signposted in the main text, will be capped at a **22HIGH**.
- Submissions with no referencing at all will be capped at a **3HIGH**.

Collaboration Guidelines

Whilst the university guidelines on academic irregularity prohibit collusion, it is accepted that students will discuss ideas relating to both the coding and write-up aspects of this assignment with each other. This is not discouraged since benefits to learning can come out of sharing ideas. However, the following actions are strictly prohibited and will be acted upon if Turnitin suggests that any students breach these rules:

- **Plagiarism of Code** Be that from an online source or from a current/previous student.
- **Collusion in the write-up** Your report write-ups should be individual pieces of work. The best way to avoid issues here is to not show any part of your final write-up to another student. Using Turnitin, it is very easy to distinguish between students who have discussed ideas for their report but written up separately and students who have written up together.

Note that some of the questions in this assignment relate to the **Public Information** Excel file on NOW. This file splits the cohort into groups of four by student number and some questions ask you to solve your “group problem”. These groups only exist for the purposes of creating questions for you to answer. *You are reminded that this is an individual coursework and you should not be colluding with other members of your “group”.*

As a rule of thumb, you should consider discussing your work with your peers but not actually showing them any of your code/write-up. This will allow you to have the benefit of shared knowledge but will put you at no risk of receiving a Turnitin report that is a cause for concern. It is accepted that some of the code, particularly in Q1 and Q2 is likely to be flagged up by Turnitin, even where collusion/plagiarism has not taken place. Do not be concerned by this if you see this when submitting to the testing folder.

Coding Rules & Advice

Some of the coding questions in this assignment can be answered very easily using built-in MATLAB functions. As such, restrictions are placed on the built-in functions that you may use in your code.

You may use the built-in functions *mod()*, *length()*, *ceil()* and *floor()* at any time in your code, along with any standard operators such as +, *, -, /, ^ and their vector equivalents. Note that ^ can be used to find roots by using fractional powers. You may also use any functions that you have already created

from previous questions in the coursework. For example, if you create a function called *trialdiv()* for Question 1a), you may use the *trialdiv()* function in all future questions. No other built-in MATLAB function may be used unless specified. You may also use statements such as *for* and *while* loops, conditional *if* statements and the *break* and *return* commands.

If you are asked to check your code using a built-in MATLAB function, the built-in function should not form a part of the code given in your appendix but you should make some comment in your report about which values you checked your code on. Code using functions that are not permitted will result in marks being docked for the relevant question. The extent to which the code relies on these functions will determine how far the grade drops.

Note in particular that functions such as *disp()* and *fprintf()* are not permitted to display your answers. Instead, all of your answers should be given in function format. The question will specify what your input and output variables should be. Consider the example below of a simple function used to add up two numbers.

```
function c = add(a,b)
c = a+b;
end
```

In this function, *a* and *b* are the inputs and *c* is the output.

Once you have working code, you should also try to make it as *efficient* as possible. This means reducing the amount of work that MATLAB has to do to process your code. This can be measured in **arithmetic operations**, a concept that is explained in Question 1. Note that shorter code is not necessarily more efficient, for example, if you have a particularly long loop in the code.

If you cannot get some code to work, try to provide an explanation of where you think you are going wrong. You may be able to receive some credit for this.

Commentary Advice

The commentary questions determine what should go in your report. These are the questions that actually determine your grade. The MATLAB code is simply a means to an end to answer these questions. Assuming your code works and follows the rules given above, the only impact that your code will have on your grade is the level of efficiency that you have achieved. Any question that has no commentary answer given will receive a grade of ZERO, regardless of the quality of any code that is submitted for the question.

Your report should not look to explain what is happening in your code unless the commentary questions specifically request this. Some of the commentary questions mainly rely on using your code to produce some output but most require further investigation or thought.

Do not think of this as a coding assignment. The report is what constitutes the bulk of the marks so you should give yourself sufficient time to produce as good a write up as possible.

Question 1

Code: a) Write a function to test whether a natural number, n , is prime by trial division (testing from 2 to $n - 1$). Once a number is determined to be *not prime* you should break out of function immediately. Your output should be 0 if the number is not prime and 1 if the number is prime. You do not need to make allowances for the user inputting something that is not a natural number but your code should give the correct output for *every* natural number.

b) Your function in a) should take a while when testing certain large numbers (10 or more digits). Make a very simple modification to your function to speed it up. (You no longer have to test all the way to $n - 1$.) Test this new function is correct by comparing its outputs on some examples to those given by the built-in *isprime()* function in MATLAB.

c) Write a function to find the complete prime factorisation of a natural number, $n > 1$. Your output should be a vector containing all prime factors, listing them each as many times as they are a factor. For example, MATLAB should output

ans = 2 2 2 3

when asked to factorise 24. You do not need to make allowances for the user inputting something that is not a natural number > 1 but your code should give the correct output for *every* natural number > 1 . Test your function on some examples and compare your results to those given by the built-in *factor()* function in MATLAB.

Commentary: i) Explain the reasoning behind your chosen modification in b).

ii) Your function in c) will take longer to run for some numbers than others. Investigate what types of number take longer to factorise. Can you pinpoint a rule that specifically tells us whether one number is likely to take longer to factorise than another? You may wish to make yourself a MATLAB script to run your function and use the *Run and Time* setting in MATLAB to help you out with this. If you take this approach, you do not need to submit the code for your script.

iii) By considering the number of operations that MATLAB needs to perform to factorise a number, can you justify your claims in ii)? Each use of an arithmetic operator (+, -, *, /, ^) counts as one operation. Any use of a check, such as ==, < or > also counts as one operation. Finally, each use of the permitted functions is also considered to be one operation. The exception to this is *mod()* which you should research. You will not be able to find an exact formula for the number of operations required for any given n but you should be able to find lower and upper bounds in terms of n based on the rule you came up with in b).

iv) The number 1 is not generally considered to be a prime number. Does your work on this question provide evidence support or contradict that point of view? Briefly explain your reasoning.

Question 2

Code: a) Write a function to solve $HCF(a, b)$ by using Euclid's algorithm and find integers u and v such that $au + bv = HCF(a, b)$. Your input should be two natural numbers a and b . Your code does not need to account for the user inputting anything that is not a natural number but it should not rely on $a < b$ or vice versa. Your output should be a vector whose elements are $HCF(a, b)$, u and v respectively.

b) Write a function to solve the linear congruence $ax \equiv c \pmod{n}$, where a , c and n are known natural numbers. These three values should form the input and you do not need to account for the user inputting non-natural numbers. The output of the function should be a vector containing all integer solutions in the range $0 \leq x \leq n - 1$. If no solutions exist, the output vector should be empty.

Commentary: i) Did your code in a) require any special consideration to accommodate the possibility of your input having $a < b$ or $a > b$? Explain your reasoning.

ii) Open the **Public Information** file in the NOW folder and find the group containing your student ID number. Using your function in a), find the highest common factor of the last six digits of your ID, paired with each of the other student ID numbers within your assigned group. (For a group of five, you should find four HCFs.) For example, if your ID number is N0123456 and another member of your group has ID number N0654321, then find $HCF(123\ 456, 654\ 321)$ as one of your answers. For each of your pairs, find u and v . Check your answers using the built-in $gcd()$ function in MATLAB.

iii) Explain from a mathematical perspective how your function in a) has helped you to create your function in b). (What is the relationship between Euclid's extended algorithm and linear congruences?)

iv) Find solutions to the following using your function in b). If none exist, state why, giving reasoning specific to your example:

$$74\ 946x \equiv 5184 \pmod{330\ 389}$$

$$74\ 946x \equiv 5184 \pmod{655\ 678}$$

$$74\ 946x \equiv 5184 \pmod{983\ 517}$$

$$74\ 946x \equiv 5184 \pmod{162\ 383}$$

Question 3

Code: Let $n = pq$ where p and q are primes. Then $\varphi(n) = (p - 1)(q - 1)$ where $\varphi(n)$ is Euler's totient function. If e is coprime to $\varphi(n)$, then there exists d such that $ed \equiv 1 \pmod{\varphi(n)}$. Your functions from Question 2 should allow you to find d given e . Note that values of n and e in this form describe public keys in the RSA algorithm. Using this knowledge:

a) Write a function to compute the private decryption key from a given public encryption key for the RSA algorithm. Your input should be the known public key values n and e and your output should either be the private key value d or a private key vector containing n and d . Your code may assume that the n and e provided are from a working RSA public key and therefore are in exactly the form described above. In particular, note that n only has two prime factors. You may wish to call upon or adapt previous functions that you have created to complete this question.

b) Write a function to convert an encrypted number $c \equiv m^e \pmod{n}$ into $m \equiv c^d \pmod{n}$ where $0 \leq m < n$. You may assume that the private key is known when creating this function. Your input values should be c , n and d where c is either a single block or a vector of several blocks form a coded message. (No extra credit is given for being able to decrypt several blocks of code at once.) Your output should be the original message, m , in numeric form. Your code should also take account of the fact that MATLAB does not deal well with integer calculations $> 10^{16}$. Your code will only be used on RSA keys with $n < 10^6$.

***Hint:** This task is harder than it may seem at first because of the 10^{16} issue. You cannot just expect MATLAB to calculate c^d , even if you plan to take this value modulo n immediately after. Coming up with code to perform this calculation is tricky but the code itself can be very simple. Coming up with code to perform this calculation efficiently is much trickier. You are advised to come up with inefficient code first and then try to improve upon it.*

Commentary: i) Use your code to find the RSA private key (n,d) from the following public keys (n,e) :

(999 797, 949 951), (898 993, 198 097), (813 691, 534 233), (145 157, 3 649).

ii) Given n and e , how many operations does your function need to find p and q ? Given p , q , and hence $\varphi(n)$, how many further operations are needed to find d ? You will not be able to find an explicit formula for this but should be able to come up with lower/upper bounds. You should try to keep these bounds in terms of n for top marks. Comment on your results. In particular, what parts of your code are computationally expensive/inexpensive.

iii) Given that MATLAB does not deal well with integers greater than 10^{16} , give an upper bound for n for which your MATLAB code can be trusted to work and explain your reasoning. You should consider your 3b) code to answer this.

iv) Each "group" has been given a public key and an encrypted message on the **Public Information** spreadsheet on NOW. We use the coding **00** \leftrightarrow **space**, **01** \leftrightarrow **a**, **02** \leftrightarrow **b**, ... , **26** \leftrightarrow **z**, **27** \leftrightarrow ' and encrypt blocks of three letters at a time. For example, the message "come to me" would be encoded as 031513 050020 150013 050000 before encryption. Use your program from a) to find your group's private key and your program from b) to decrypt your group's message.

You do not need to write code to convert a string of numbers into the appropriate letters. This should be done manually. You should also remember that if your decryption reveals, "123", this translates as "000123" which translates to "space" "a" "w" prior to encoding.

Question 4

Commentary: Conclude your report by discussing whether you believe RSA to be a safe system for sending encrypted messages. Any claims that you make should be supported either by things that you have achieved earlier in this report, external academic resources or some combination of the two. If you have any other interesting observations that you have been unable to include previously in your report, you may include those here. Any comments from external sources should be referenced in an appropriate academic style.

You should be looking for around 1 page of material here. This question is far more nuanced than it first seems. To answer this question, think about what you have achieved in the first three questions of the assignment. Not just in the coding, but also when thinking about your commentary as well. Think carefully about what has been asked of you earlier in this assessment and ask yourself why you may have been asked to think about these things.

II. Assessment Criteria

Each of the four questions will be graded according to the criteria given below and given an NTU points score. (See the Grade Conversion table on NOW.) The overall grade for this assessment will be determined by the mean of points scores for each of the four questions. *Remember that not submitting a .pdf will result in a grade being docked and improper/no referencing will cap your overall grade.*

GRADE	CRITERIA
1 st	<p>Note that some extension to each of the following is expected for a 1EXC grade.</p> <p>Q1: Code is correct. The modification in b) is as good as possible and efficiencies have also been included in c). In the commentary the student has justified the modification clearly in i). Parts ii) and iii) reveal a thorough investigation coming to a clear conclusion about which numbers take longest to factorise and quantifying the computational expense. The answer to iv) provides a clear statement, backed up by evidence obtained in the process of doing this question.</p> <p>Q2: Code is correct and gives all output as expected. In the commentary i) is correct and well justified. The explanation of the connection between the two equation types is clear and mathematically grounded in iii) and iv) provides a clear explanation for any congruences without solutions that specifically relate to the question.</p> <p>Q3: Code all works and is very efficient. In ii) the student is able to consider the number of operations completely in terms of n and, as a result, is able to provide further insight into the computational expense of breaking RSA. This may require well-referenced external research. The correct answer is provided for iii) based on the information given in the question.</p> <p>Q4: Conclusion should reach a well-reasoned judgement about the security of RSA using multiple academic resources and drawing on the students' own work to back up claims made using external resources. The conclusion should probe several different lines of enquiry into the safety of RSA encryption. Referencing should be done correctly.</p>
2:1	<p>Q1: Code is correct but there may be clear inefficiencies, possibly in the choice of modification or, more likely, in c). In the commentary the student has justified the modification well in i). Parts ii) and iii) will show a good investigation into which numbers take longer to factorise, reaching a sensible conclusion backed up by good consideration of the operations required but this conclusion may not give a clear idea of expense in terms of n. The answer to iv) provides a clear statement, backed up by evidence obtained in the process of doing this question.</p>

	<p>Q2: Code is correct and gives correct answers. The commentary parts are all done well but there may be minor inaccuracies or parts that lack a little clarity. In particular, iii) may focus more on the process of coding than the mathematical relationship.</p> <p>Q3: Code all works but may have efficiency issues in b). In ii) the student has made an effort to consider the number of operations required but this may not be given completely in terms of n and may lack insight. Part iii) should either be answered correctly or have an incorrect answer with sensible reasoning.</p> <p>Q4: Conclusion should reach a well-reasoned judgement about the security of RSA using multiple academic resources and drawing on the students' own work. Referencing should be done well.</p>
2:2	<p>Q1: Code may contain minor errors but is largely correct, if inefficient. Commentary gives largely correct responses but quantifying operations iii) may prove beyond the student and iv) may have an answer that is more rooted in the definition of a prime number than the evidence of this question.</p> <p>Q2: Code may contain minor errors. The commentary is largely correct but some of the explanations outside of the coding are missing or contain clear inaccuracies. Part iii) is likely to be the main cause for concern here.</p> <p>Q3: Code for 3b) may not be working or is very inefficient but a) should be correct and some effort made to explain what may be going wrong in b) if it does not work. In the commentary, the student should have made an effort at ii) and considered what may be happening in iii) but answers may be incorrect or, in the case of ii), there may not be any attempt at all to quantify the number of operations.</p> <p>Q4: Conclusion focuses on security of the algorithm and comes to a reasonable judgement but this may be disjointed from the rest of the work completed, coming across more as just an exercise in external reading. Alternatively, the student may have done minimal external reading but has related the conclusion to their own work. There may also be some issues with the referencing style.</p>
3 rd	<p>Q1-3: Code contains errors, some may be missing altogether and commentary may lack supporting evidence but there should still some correct information here.</p> <p>Q4: Conclusion does address the issue of security but may use minimal, non-academic sources. Alternatively, the conclusion may read as more of a recap than an answer about security. May have serious referencing issues.</p>
Fail	<p>Q1-3: Question has been attempted but contains many errors and the commentary provides little work of merit.</p> <p>Q4: Only a very basic conclusion has been produced, using one non-academic source and not answering the key issue of security.</p>

ZERO	Q1-3: No commentary is attempted and/or no code is provided. Q4: Conclusion is missing or is of no merit.
-------------	--

III. Feedback Opportunities

Formative (Whilst you are working on the assignment)

All seminar sessions during this module will be dedicated to allowing students to improve their coding skills and to work on this assignment. These sessions will be unstructured and students are welcome to seek feedback from the tutor at any time during these sessions.

Summative (After you've submitted the assignment)

Individual feedback will be given in the form of Turnitin Grademark comments on your submissions, which will be made accessible to you after the marking period. General group feedback regarding what was done well and what common problems were will be given in a NOW video after the marking period.

V. Moderation

The Moderation Process

This assessment has been pre-moderated by Dr James Hind and an external examiner. Post-moderation will also take place by the same people. The moderation process is in accordance with standard university guidelines.

VI. Aspects for Professional Development

This assessment has clear implications for developing your coding skills but is also designed to increase awareness of the applications of even the most abstract mathematics in the real world. More importantly, whilst this coursework requires a significant amount of work, a significant amount of time is set aside in seminars for you to work on this at your own pace. Time management and workload prioritising are key aspects in producing this piece of work, and other coursework with similar hand-in dates, all at a high level.