General Certificate of Education
Advanced Subsidiary Examination
June 2013

# Computing COMP1/PM

**Unit 1 Problem Solving, Programming, Data Representation and Practical Exercise**

# Preliminary Material

**To be given to candidates on or after Friday 1 March 2013, subject to the instructions given in the *Teachers' Notes* (COMP1/TN).**

**Information**

- This Preliminary Material comprises Instructions for Candidates and a Data File.
- A Skeleton Program is provided separately by your teacher and must be read in conjunction with this Preliminary Material.
- Candidates are advised to familiarise themselves with the Preliminary Material and Skeleton Program before the examination.
- Another copy of this Preliminary Material will be made available to you in the examination. You will also be given access to the Skeleton Program and Data File electronically at the start of the examination. You must **not** take any copy of the Preliminary Material, Skeleton Program or any other material into the examination room.

## INSTRUCTIONS FOR CANDIDATES

The question paper is divided into four sections and a recommendation is given to candidates as to how long to spend on each section. Below are the recommended timings for the 2013 examination.

### SECTION A

You are advised to spend no more than **35 minutes** on this section.
Questions will examine the specification content **not** specific to the **Preliminary Material.**

### SECTION B

You are advised to spend no more than **25 minutes** on this section.
You will be asked to create a new program **not** related to the **Preliminary Material** or **Skeleton Program**.

### SECTION C

You are advised to spend no more than **10 minutes** on this section.
Questions will refer to the **Preliminary Material** and the **Skeleton Program**, but will **not** require programming.

### SECTION D

You are advised to spend no more than **50 minutes** on this section.
Questions will use the **Skeleton Program** and the **Preliminary Material** and may require the **Data File diary.txt**.

This **Data File** must **not** be altered in any way prior to the examination.

### Electronic Answer Document

Answers for all questions for all sections must be entered into the word processed document made available to the candidate at the start of the examination and referred to in the question paper rubrics as the **Electronic Answer Document**.

### Preparation for the Examination

For your programming language you should ensure that you are familiar with this **Preliminary Material** and the **Skeleton Program**.

During the examination you will **not** be asked to modify, or explain, the code in the subroutine `DecryptUsingRailFence`.

## "SECRET MESSAGES"

The **Skeleton Program** in this Preliminary Material is a program for creating and reading secret messages.

**Table 1** gives definitions for some important terms that are used in this Preliminary Material.

**Table 1**

| Term | Definition |
|---|---|
| Sender | The person who has written a message that they wish to send to someone else. |
| Receiver | The person to whom a message is being sent. |
| Plaintext | The original message that is being sent. |
| Ciphertext | The encrypted version of the plaintext. |

Two of the main methods used for creating a secret message are *steganography* and *cryptography*.

**The Preliminary Material continues on the next page**

**Turn over ▸**

## STEGANOGRAPHY

Steganography is hiding the existence of a message so that only the sender and the receiver know of the existence of the message; no one else will know that there is a message being sent. Examples of steganography include using invisible ink to write a message and writing a message on an envelope and then placing a postage stamp in a position that hides the message.

Steganography can also be used by concealing data in computer files. An example of this is when a message is hidden in a larger piece of text or when certain pixels in an image have their colour changed to correspond to a letter in the alphabet.

### Steganography – every *n*th character

One way of hiding a message in a larger piece of text is to use every *n*th character in the larger piece of text to form the hidden message. **Figure 1** and **Figure 2** show examples of steganography that use the every *n*th character method.

**Figure 1**

Text the message is hidden in: `Oak.`
Value for *n*: 2

The 1st character is "`O`".
The 2nd character ("`a`") is skipped over.
The 3rd character is "`k`".
The 4th character "`.`" is skipped over.

This gives a hidden message of "`Ok`".

**Figure 2**

Text the message is hidden in: `Cow or pigeon`
Value for *n*: 4

The 1st character is "`C`".
The 2nd, 3rd and 4th characters ("`ow `") are skipped over.
The 5th character (1 + *n* = 5) is "`o`".
The 6th, 7th and 8th characters ("`r p`") are skipped over.
The 9th character (5 + *n* = 9) is "`i`".
The 10th, 11th and 12th characters ("`geo`") are skipped over.
The 13th character (9 + *n* = 13) is "`n`".

This gives a hidden message of "`Coin`".

## CRYPTOGRAPHY

Cryptography does not hide the existence of the message. Instead it changes the message in such a way that it can be read only by the sender and the receiver. Anyone else will be able to see that there is a message but will not be able to work out the meaning of the message. The sender and the receiver have to agree on the cryptography method that will be used for their communication. No one else will be able to read the message if the method chosen is sufficiently strong (complex). However, if the method chosen is too weak (simple) then other people may be able to work out the method that has been used and therefore be able to read the message.

Encryption is the process of taking a message and changing it so that it cannot be read by those who do not know the cryptography method being used. The original message is called the *plaintext* and the encrypted version of the message is called the *ciphertext*.

Decryption is the reverse of encryption and is the process of taking the ciphertext and changing it back into the plaintext.

### Cryptography – Caesar cipher

The Caesar cipher is a simple form of encryption where every letter in the plaintext is replaced by the letter that is a certain number of places from it in the alphabet. This cryptography method is named after the Roman leader Julius Caesar who used to send secret messages by changing every letter in his message to the letter that comes 3 places after it in the alphabet.

The *plaintext alphabet* is the alphabet used to write the original message and the *ciphertext alphabet* is the letters that are substituted in place of the letters in the plaintext alphabet.

Any non-alphabetic characters in the plaintext are kept the same in the ciphertext.

**Figure 4** shows an example of a Caesar cipher and **Figure 3** shows the plaintext and ciphertext alphabets that are used in the example shown in **Figure 4**. The plaintext alphabet has been shifted by 6 places to obtain the ciphertext alphabet.

**Figure 3**

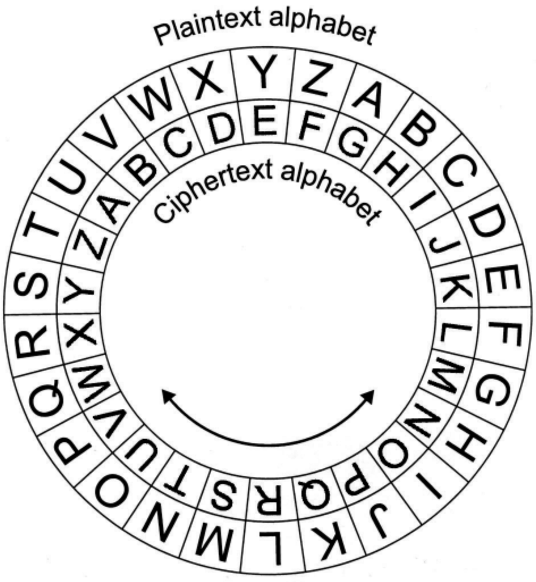
Plaintext alphabet
A B C D E F G H I J K L M N O P Q R S T U V W X Y Z
Ciphertext alphabet
G H I J K L M N O P Q R S T U V W X Y Z A B C D E F

**Figure 4**

Using a Caesar cipher with a key of 6 the plaintext "`HELLO THERE`" would be encrypted as "`NKRRU ZNKXK`".

**Turn over ▸**

### Cryptography – rail fence

Instead of changing the letters used in the plaintext, another way of encrypting is to change the order of the letters in the message. The rail fence is a method of encrypting which does this.

To encrypt using the rail fence method, a size for the rail fence (*n*) is chosen where *n* is less than the length of the plaintext message.

The plaintext message is then written out in a zigzag shape using *n* lines. The 1st character of the plaintext is written on the first line, the 2nd character on the 2nd line, ..., the $n^{th}$ character on the $n^{th}$ line. The $(n+1)^{th}$ character is then written on the 1st line, $(n+2)^{th}$ character on the 2nd line, ..., the $(n+n)^{th}$ character on the $n^{th}$ line. This process continues until all the characters of the plaintext have been written down in a zigzag shape.

The ciphertext is then produced by writing out each line (row) of the rail-fenced plaintext message.

**Figure 5** and **Figure 6** show examples of a rail fence being used to encrypt a plaintext message of "`Hello_there`". An underscore ( _ ) has been used to denote a space.

**Figure 5**

To encrypt using a rail fence of size 2:

```
Hello_there

     ↓

H l o t e e
 e l _ h r

     ↓

Hloteeel_hr
```

**Figure 6**

To encrypt using a rail fence of size 4:

```
Hello_there

     ↓

H   o   e
 e   _   r
  l   t   e
   l   h

     ↓

Hoee_rltelh
```

## THE SKELETON PROGRAM

In the **Skeleton Program** there is a menu containing nine options.

The first two options on the menu are about the plaintext (ie the original message). If the user selects *option a* they will be asked to enter the plaintext. If the user selects *option b* then the current value of the plaintext will be displayed.

The next two options on the menu are about the ciphertext. If *option d* is selected then the user will be asked to enter the ciphertext; if *option e* is selected then the current value of the ciphertext will be displayed.

The next two options on the menu are used to encrypt the plaintext. If *option g* is selected then the plaintext will be encrypted using a Caesar cipher (**Figure 3** and **Figure 4** show an example of this). The user will be asked to enter a value for the key and the ciphertext will then be created and displayed to the user. If *option h* is selected then the plaintext will be encrypted using a rail fence cipher (**Figure 5** and **Figure 6** show examples of this). The user will be asked to enter a value for the size of the rail fence and the ciphertext will then be created and displayed to the user.

The next two options on the menu are used to decrypt the ciphertext. If *option j* is selected then the program will attempt to decrypt the ciphertext using a Caesar cipher with the numeric key entered by the user. If *option k* is selected then the program will attempt to decrypt the ciphertext using a rail fence with the rail fence size entered by the user. If the user does not enter the correct value for the key / rail fence size (or the message was not encrypted using the method chosen by the user) then the plaintext will not be decrypted correctly and the user will not see the original message that was sent.

*Option n* on the menu is used to read a message hidden using steganography in the **Data File diary.txt**. The user is asked to enter a start position, an end position and a value for *n*. The program will then find a message that has been hidden using every *n*<sup>th</sup> character in the **Data File diary.txt** between the start and end positions (inclusive). Example: if the user enters a start position of 1, an end position of 208 and a value for *n* of 9 then the hidden message "`MeeT ME at the colD room`" will be revealed.

A copy of the contents of the **Data File diary.txt** can be found at the end of this **Preliminary Material**.

If the user *enters the letter q*, instead of selecting an option from the menu, then they will quit the program.

**The Preliminary Material continues on the next page**

**Turn over ▸**

## THE DATA FILE

```
Maia rolled her eyes.
Sir Terrence, Faisal, Maia and Ed had no money available at all. It
was about the bathroom's fetid reek (& limescale & growing mould)
that Derek and I let our Faisal organise our 09th meeting of the
year.  Maia wasn't happy that we were having another meeting as
flat meetings never went well but Faisal was right that we needed
one.  Pete had moved out at short notice and that had left a hole
in the kitty.  The bathroom definitely needed to be sorted - you
felt cleaner before having a shower than you did after having one
in this flat.

The next 7 pages were not readable.  The next extract that could be
read was found on line 665.
'Hello Pete, I know Maia and Terrence spoke to you on the phone,' I
said.
'They did,' Pete said.
'So, what are you going to do?'
'I can’t give you my share at the moment as I don't get paid until
Friday, but I know that I need to give you lot the money for the
last month and I will do.'
'Friday?'
'Yes, Friday.'
'I'll be in from five if you want to pop round then.'
'I get off work at six so I could get there for 8 if that’s okay?'
'No problem.'
'See you on Friday.'
I nodded my head, 'Friday.'
At the time I believed Pete - but he was going to let us down
again.

After this point there is very little that can be read, there is
another bit on line 1369.
As soon as I mentioned Brewood and Owen, O'Shanty got talking and
soon enough we knew what we needed to do.

These are all the extracts that can be made out from the diary - it
was very badly water-damaged. Hope this helps.
```

The **Data File diary.txt** will be available to you at the start of the examination.

If you are trying to find a hidden message by hand, it is worth noting that the **Data File** contains some characters that are not visible in the text above. There are carriage return and line feed characters at the end of lines where the enter key has been pressed, for example:

```
Maia rolled her eyes.
Sir Terrence,
```

There are 36 characters here. 34 of them are visible characters and there is one carriage return and one line feed after the full stop.

**END OF PRELIMINARY MATERIAL**