HWA CHONG INSTITUTION
C2 PRELIMINARY EXAMINATION 2013

COMPUTING
Higher 2

16 September 2013 **Paper 1 (9597 / 01)** **1300 -- 1615 hrs**

Additional Materials:
Electronic version of `DECIMAL.txt` data file
Electronic version of `MARKS.txt` data file
Electronic version of `PAS.txt` data file
Electronic version of `EVIDENCE.docx` document

READ THESE INSTRUCTIONS FIRST

Type in the `EVIDENCE.docx` document the following:

- Candidate details
- Programming language used

Answer **all** questions.

The maximum mark for this paper is 100.

All tasks must be done in the computer laboratory. You are not allowed to bring in or take out any pieces of work or materials on paper or electronic media or in any other form.

All tasks and required evidence are numbered.

The number of marks is given in brackets [ ] at the end of each task.

Copy and paste required evidence of program listing and screen shots into the `EVIDENCE.docx` document.

At the end of the examination, print out your `EVIDENCE.docx` and fasten your printed copy securely together.

1. The algorithm for converting a decimal number to a 8-bit binary number is as follows:

- Keep dividing the decimal number by 2 until the quotient is zero.
- Use the reminders in reverse order as the digits of the converted number.
- Pad leading zeros to make it up to 8-bit binary number, if needed.

Note that the largest decimal number that can be stored using 8 bits is 255.

Example: Convert decimal number 18 to binary.

```
18 / 2  →  quotient : 9,   remainder:  0
 9 / 2  →  quotient:  4,   remainder:  1
 4 / 2  →  quotient:  2,   remainder:  0
 2 / 2  →  quotient:  1,   remainder:  0
 1 / 2  →  quotient:  0,   remainder:  1
```

If you read from last remainder to first (from bottom to top), you get `10010`, which are 5 bits, so you need to add three zeros to the left. Hence 18 (decimal) = `00010010` (8-bit binary).

Task 1.1
Write program code for the `DecimalToBinary` function using the following specification.

```
FUNCTION DecimalToBinary (DecimalNumber : INTEGER): STRING
```

The function has a single parameter `DecimalNumber` and returns a 8-bit binary string result.
Use the sample data provided in the text file `DECIMAL.txt` and paste this into your programming code.

Evidence 1: Your `DecimalToBinary` program code. [7]

Evidence 2: One screenshot showing the output from running the program code for the data in `DECIMAL.txt`. [5]

A **bit shift** is a procedure whereby the bits in a binary string are moved to the left or to the right. For example, we can shift the bits in the string `1011` two places to the left to produce the string `1110`. Note that the leftmost two bits are wrapped around to the right side of the string in this operation.

Task 1.2
Write program code with the following specification:
- Input a 8-bit binary string
- Validate the input
- Shift the input string one place to the left, wrapping the leftmost bit to the rightmost position
- Output the resulting binary string.

Evidence 3: Your program code. [7]

Task 1.3
Draw up a list of **four** suitable tests and provide screenshot evidence for your testing.

Evidence 4: Annotated screenshots for each test data run. [4]

Use the strategy of the decimal to binary conversion and the bit shift left operation defined above to code an encryption algorithm. The algorithm should add 1 to each character's numeric ASCII value, convert it to a 8-bit string, and shift the bits of this string one place to the left. A single space character in the encrypted string separates the resulting bit strings.

For example, to encrypt a word 'AM'

character 'A':
Add 1 to ASCII value of 'A' → 65 + 1 = 66
Convert 66 to 8-bit binary string → 01000010
Shift the binary string 1 place to the left → 10000100

character 'M':
Add 1 to ASCII value of 'M' → 77 + 1 = 78
Convert 78 to 8-bit binary string → 01001110
Shift the binary string 1 place to the left → 10011100

Hence 'AM' is encrypted as '10000100 10011100'

Task 1.4
Write program code which does the following:
- The user inputs a word to be encrypted
- Encrypt the word using the above encryption algorithm
- Output the encrypted word.

Evidence 5: Your program code. [9]

Evidence 6: Produce two screenshots showing the output of 'DAD' and 'HELLO' by the user. [2]

2. The following is a pseudocode algorithm for a recursive function that returns the factorial of a non-negative integer `n`. The algorithm works for most input but is missing out on one edge case.

```
FUNCTION Factorial(n)
     if n = 1
          result = 1
     else
          result = n * Factorial(n-1)
```

Task 2.1

Write program code for this algorithm including the amendment you would make to:

- make the function works for all cases
- adhere to good programming style

Verify that your program works by testing it for suitable values of `n<=100`.

Evidence 7: Your program code. [5]

Evidence 8: Two screenshots showing the output from running the program code for `n=0` and `n=50`. [2]

Task 2.2

While concise and elegant, recursive functions are more computationally resource intensive, and break down when the input size is large (e.g. when `n=1000` for the above recursive factorial function). To improve its efficiency and ability to handle large input sizes, convert the recursive factorial function into an iterative one.

Evidence 9: Your converted program code. [7]

Evidence 10: One screenshot showing the output from running the program code for `n=1000`. [1]

3. In a training school, every student has to take a core module. An application is to be created to compute the highest mark, the average mark of the module and assign the grades to the students.

A text file `MARKS.txt` holds the records of the students' performance in the module. Each student record takes up one line and has three data fields: `student ID`, `student name` and `final mark`. Each data field is delimited with a comma (",") .

For example, `S0101,ALFRED ARUMUGAM,33`

Task 3.1
Write program code for a procedure `HighestAndAverage` which does the following:

- the program reads from file `MARKS.txt`
- compute and display on screen:
    - the highest mark and the name(s) of the student(s) who achieved it
    - the average mark of the module, rounded to 2 decimal places.

Evidence 11: Your `HighestAndAverage` program code. [12]

Evidence 12: One screenshot showing the output from running the program code. [4]

Task 3.2
Write a program to assign the grades to the students in `MARKS.txt` and generate a new file called `GRADES.txt`.

Note: Amend your `HighestAndAverage` program code from Task 3.1 to return the highest mark and the average mark of the module so that it can be used in Task 3.2.

The grades (`M`, `F`, `P`) are assigned as follows:

`M`	:Merit. The student with the highest mark in the module
`F`	:Fail. Students who score less than 10 marks below the module average
`P`	:Pass. All other students who did not get `M` or `F`

The file `GRADES.txt` has the same format as `MARKS.txt`. Each student record in `GRADES.txt` has five data fields: `student ID`, `student name`, `final mark`, `average mark` of the module (rounded to 2 decimal places) and the `grade` assigned.

For example, `S0101,ALFRED ARUMUGAM,33,71.64,F`

Evidence 13: Your program code. [8]

Evidence 14: Your `GRADES.txt`. [3]

4. Several tests are available to determine the readability of a book for a target age group. One such test by McLaughlin is described below. The test is carried out on sample sets of sentences from a book.

McLaughlin's (1969) SMOG Grade

1. Select 10 consecutive sentences near the beginning, 10 consecutive sentences near the middle and 10 consecutive sentences near the end of the book.

2. In the 30 sentences selected, count every word of 3 or more syllables. These are polysyllabic words (PSW).

3. Find the square root of the number of polysyllabic words (PSW) counted in step 2.

4. Add 3 to the square root.

5. This gives the SMOG grade.

A word is any sequence of characters delimited by white space or punctuation marks, whether or not it is an actual English word. A syllable may be counted using the following rules: Each group of adjacent vowels (a, e, i, o, u, y) counts as one syllable (for example, "ea" in "read" counts as one syllable but the "e…a" in "beta" counts as two syllables). However, an "e" at the end of a word does not count as a syllable. The letter "y" is to be treated as a vowel in the counting of syllables. For example, "floppy" is to be counted as two syllables and "byte" is counted as one syllable. Also, each word has at least one syllable, even if the previous rules give a count of zero.

Task 4.1
Write program code to find the SMOG grade for a book, `PAS.txt`, and display the results on screen. The output should be similar to this:

```
Sample text name: PAS.txt
No. of PSW:
Square root of PSW:
SMOG grade:
```

The text file `PAS.txt` is provided for you. The sentences of the book are stored line by line in the file.

Add comment lines at the start of your program code to explain the method that you used to get the 10 consecutive sentences near the middle of the text file when computing the SMOG grade.

Evidence 15: Your program code. [20]

Evidence 16: One screenshot showing the output from running the program code. [4]

-- END OF PAPER ---