UNIVERSITY OF EDINBURGH

COLLEGE OF SCIENCE AND ENGINEERING

SCHOOL OF INFORMATICS

COMPUTER PROGRAMMING MOCK

10:10 to 11:40, Thursday 9th November
15:20 to 16:50, Monday 13th November
12:20 to 13:50, Tuesday 14th November

Convener: ITO-Will-Determine
External Examiner: ITO-Will-Determine

INSTRUCTIONS TO CANDIDATES

1. **Answer all questions.**

2. **Questions in section A are each worth 10 marks, but are not necessarily of equal difficulty. You are advised to answer them in order.**

3. **Section B is worth 25 marks.**

4. **Consult the separate instruction sheet for details of how to obtain question files and submit your answers.**

Section A

This section contains three short questions worth 10 marks each, in which you are asked to implement either one function of a program, or a short complete program. *In the real exam, section A will have five questions.*

Little credit will be given for incomplete solutions.

1. You are given a file `chardemo.c`. Complete the file to produce a program which reads a character from the user, and prints out the character, its ASCII value (in decimal), and the upper case version of the character according to `toupper()`. Your program should print messages exactly as in the following example:

   ```
   Enter a character: b
   Your character is b
   Its ASCII value is 98
   Its uppercase version is B
   ```

   Recall that the 'ASCII value' of a `char` is simply its value as an integer. [*10 marks*]

   When you are ready, submit the file `chardemo.c`

2. In the template file `maxtofront.c`, you can see the function prototype:

   ```
     void maxToFront(int a[], int n)
   ```

   together with some `main` code which declares two arrays, and then makes calls to `maxToFront`, printing the results to standard output.

   Implement the `maxToFront` function so that it moves the largest element of `a` to the front of the array `a`, but leaves all other elements in their original relative order. If you implement the `maxToFront` function correctly, the output of the program (with the given `main` code) should look exactly like this:

   ```
   After maxToFront, b is 44, 6, 2, 4, 5, -10, -6, 5, 8, 2.
   After maxToFront, c is 44, 2, 3, -6, 4, 8, -2, 44, 9, 6, 1, 3, 4, -11, 0.
   ```

   If there are equal largest elements, your function should move just one of them, but it does not matter which one. ***When you are ready, submit the file*** `maxtofront.c` [*10 marks*]

3. C has a 'bitwise and' operator `&`, which applies to two integer values (e.g. `3 & 4`) and gives an integer value (for example, the expression `3 & 4` evaluates to `0`). Write a program `band.c` which reads a number `n` from the user, and then prints a table with `n` rows and columns: column j of row i (starting from zero in both cases) should contain i `&` j. An example of the output is:

```
Enter size of table: 5
  0  0  0  0  0
  0  1  0  1  0
  0  0  2  2  0
  0  1  2  3  0
  0  0  0  0  4
```

[*10 marks*]

When you are ready, submit the file `band.c`

Section B

This section contains one longer question worth 25 marks. *The real exam will have two questions in section B.* **Submissions that do not compile will receive little credit.**

1. In this question, you will produce an English to Pig Latin translator.

   Pig Latin is a method used by English-speaking children to disguise conversation from adults. It works by changing words as follows:

   - If a word begins with a vowel or `y-`, then `-way` is added to the end. For example, `apple` becomes `appleway`.
   - Otherwise, the initial consonant or consonants of the word are moved to the end, and `-ay` is added. For example, `pig` becomes `igpay`, `think` becomes `inkthay`, and `scram` becomes `amscray`.

   You are provided with a file `piglatin.c`, which contains a number of code sections to be completed. When completed, the program will translate the text given on standard input to Pig Latin, and print the result on standard output. The program processes text one line at a time. (Remember that standard input on a terminal can be terminated with the keystroke control-D at the start of a line.)

   (a) The function `int isVowel(char c)` takes a character and returns `1` if the character is a vowel, and `0` otherwise. A character is a vowel if and only if it is either upper or lower case `a,e,i,o,u,y`. Implement the function. [*5 marks*]

   (b) The function `void rotateWord(char w[], int n)` takes a character array `w` of length `n`, and rotates it one position, so that the letter in position `0` moves to position `n-1`, the letter in position `1` moves to position `0`, the letter in position `2` moves to position `1`, and so on. Thus `pig` becomes `igp`. Implement this function. [*10 marks*]

   (c) The function `void printPig(char *w)` takes a word in the string `w`, and prints the Pig Latin translation of the word. Implement this function, using the following logic: if `w` begins with a vowel, print `w` and then print `"way"`. Otherwise, use `rotateWord` to move letters to the end until the word begins with a vowel; then print it, followed by `"ay"`. [*10 marks*]

   You should now be able to run the program, and watch it translate your input to Pig Latin.

   When you are ready, submit the file `piglatin.c`