NATIONAL UNIVERSITY OF SINGAPORE
SCHOOL OF COMPUTING

Practical Examination 1 (PE1) for Semester 1, AY2011/2
CS1010 — Programming Methodology

17 September 2011 Time Allowed: 2 hours

INSTRUCTION TO CANDIDATES

1. You are only allowed to read this cover page. Do **not** read the question paper until you are told to do so.
2. This paper consists of **2** exercises on **6** pages. Each exercise constitutes 50%.
3. This is an open-book exam. You may bring in any printed material, but **not** electronic devices, including but not limited to laptop, thumb-drive, electronic dictionary and calculator. You are to switch off/silence your mobile phone and keep it out of view.
4. You will be logged into a special NUSNET account at the beginning of the PE. Do not log off until the end of the PE. Do not use your own NUSNET account.
5. A plab account slip will be issued to you at the beginning of the PE. Bring your matriculation card for identification when you collect it. Please leave your matriculation card on the desk in front of you throughout the PE.
6. You are to write your program in the given **plab account**. The host name is **plab0** (not sunfire!). No activity should be done outside this plab account.
7. You do not need to submit your programs to CodeCrunch. We will retrieve your programs and submit them to CodeCrunch after the PE.
8. Skeleton programs are already residing in your plab account. Please leave the programs in the home directory, and use the same program names as specified in the paper. Do **not** create subdirectory to put your programs there or we will not be able to find them!
9. **Only your source codes (.c programs)** from your plab account will be collected after the PE. Hence, how you name your executable files is not important.
10. Please read carefully and follow all instructions in the question. If in doubt, please ask. Raise your hand and the invigilator will attend to you.
11. Any form of communication with other students, or the use of unauthorised materials is considered cheating and you are liable to disciplinary action.
12. Please save your programs regularly during the PE.
13. When you are told to stop, please do so **immediately**, or you will be penalised.
14. At the end of the PE, please **log out from your plab account** and **shut down the PC**.
15. Please check and keep your belongings (esp. matriculation card) before you leave.
16. We will make arrangement for you to retrieve your programs after we have finished grading. Grading may take a week or more.

ALL THE BEST!

CS1010 AY2011/2 Semester 1
Practical Exam 1 (PE1)

Advice on Writing Your Programs – Please read!

- You are advised to spend some time thinking over the tasks to design your algorithms, instead of writing the programs right away.
- You are **not** allowed to use arrays, recursion or string functions from string.h. If in doubt, please check with the lecturer.
- If you write a function, you must have a function prototype, and you must put the function definition after the main function.
- You may write additional function(s) not mentioned in the question, if you think it is necessary.
- Any variable you use must be declared in some function. You are not allowed to use global variables (variables that are declared outside all the functions).
- You may assume that all inputs are valid, that is, you do not need to perform input validity check.
- Manage your time well! Do not spend excessive time on any exercise.
- The rough marking schemes for both exercises are given in the next page.
- Correctness of program is not solely based on the result on CodeCrunch, or based on your own runs on the plab account. If there are mistakes such as "uninitialized variable" or other errors that are not caught by the compiler, even if you managed to run your program with correct result, your program is not considered correct and marks will be deducted.

Rough Marking Scheme For Each Exercise [50 marks]

1. Style: 10 marks
    - Top of program: Are name, matriculation number, plab account-id, DG and description filled?
    - Is there a description for every function apart from the main function?
    - Proper indentation and naming of variables?
    - Appropriate comments wherever necessary?
2. Design: 10 marks
    - Correct definition and use of functions?
    - Presence of function prototypes?
    - Is algorithm simple or unnecessarily complicated?
3. Correctness: 30 marks (10 test data sets; 3 marks each)
4. Deductions:
    - Program cannot be compiled: Deduct 10 marks
    - Compiler issues warning with –Wall: Deduct 5 marks.
    - Use of arrays, recursion, built-in string functions from string.h: Deduct 10 marks
    - Use of global variables: Deduct 10 marks

Exercise 1: Which Century? (50 marks)

You are helping primary school pupils understand the concept of century. Write a program **century.c** to read in a year and determine which century that year falls in. You may assume that the year entered is always a positive integer.

Four sample runs are shown below, with user input shown in **bold**.

```
Enter year: 78
The year 78 falls in the 1st century.
```

```
Enter year: 1900
The year 1900 falls in the 20th century.
```

```
Enter year: 4112
The year 4112 falls in the 42nd century.
```

```
Enter year: 1109
The year 1109 falls in the 12th century.
```

You must provide a function **`printCentury()`** (you need to determine its return type and formal parameter(s) yourself) to print the line of output.

As your output requires an *ordinal suffix* (“st”, “nd”, “rd”, or “th”), you must provide a function **`printOrdinal()`** for this purpose (again, you need to determine its return type and formal parameter(s) yourself).

The table below shows what ordinal suffix should be used on a number. Note that special suffices (“st”, “nd”, “rd”) are given to numbers ending with 1, 2, or 3, except those ending with 11, 12, or 13 (shown in bold).

1st	2nd	3rd	4th	….	10th
11th	**12th**	**13th**	14th	….	20th
21st	22nd	23rd	24th	….	30th
…	…	…	…	…	…
101st	102nd	103rd	104th	….	110th
111th	**112th**	**113th**	114th	….	120th
121st	122nd	123rd	124th	….	130th
…	…	…	…	…	…
1401st	1402nd	1403rd	1404th	….	1410th
1411th	**1412th**	**1413th**	1414th	….	1420th
1421st	1422nd	1423rd	1424th	….	1430th

If you do not provide the required functions stated above, marks will be deducted on design.

Skeleton Program:

A skeleton program **century.c** is available in your plab account and is shown below.

```
// CS1010 AY2011/2 Semester 1
// PE1 Ex1: century.c
// Name:
// Matriculation number:
// plab account-id:
// Discussion group:
// Description:

int main(void)
{
   int year; // user’s input

   printf("Enter year: ");

   return 0;
}
```

Exercise 2: Who Are the Winners? (50 marks)

Citizens of Zakadaha hold an annual Gagalafa festival to celebrate the harvest of their prized produce, the well-sought after cocoa beans Kokomoko. A lucky draw is held during the festival. Every participant is given a lucky draw number. Each year, the organizer decides on two non-zero digits, the *factor-digit* and the *must-have-digit*. These two digits need not be distinct.

A winning lucky draw number is a number that is a multiple of *factor-digit* as well as contains the *must-have-digit*.

In this exercise, you are to write a program **winners.c** to read in the following three inputs:

- The factor-digit, which is a non-zero digit (1 – 9).
- The must-have-digit, which is also a non-zero digit (1 – 9).
- The number of participants, *n*. Lucky draw numbers will be numbered from 1 to *n* inclusively.

You may assume that all inputs are valid.

For example, if factor-digit is 3, must-have-digit is 5, and the number of participants is 100, then the number of winners is 6 (the winning numbers are 15, 45, 51, 54, 57 and 75).

Your program is to count the number of winners whose lucky draw number is a multiple of factor-digit as well as contains the must-have-digit.

Three sample runs are shown below, with user inputs shown in **bold**.

```
Enter factor-digit: 3
Enter must-have-digit: 5
Enter number of participants: 100
Number of winners: 6
```

```
Enter factor-digit: 9
Enter must-have-digit: 1
Enter number of participants: 15
Number of winners: 0
```

```
Enter factor-digit: 7
Enter must-have-digit: 7
Enter number of participants: 200
Number of winners: 5
```

You must write additional function(s) besides the main function, or marks will be deducted on design.

Skeleton Program:

A skeleton program **winners.c** is available in your plab account and is shown below.

```
// CS1010 AY2011/2 Semester 1
// PE1 Ex2: winners.c
// Name:
// Matriculation number:
// plab account-id:
// Discussion group:
// Description:

int main(void)
{
   int factor;

   printf("Enter factor-digit: ");
   printf("Enter must-have-digit: ");
   printf("Enter number of participants: ");

   return 0;
}
```

=== END OF PAPER ===