UNIVERSITY OF SWAZILAND

FACULTY OF SCIENCE

DEPARTMENT OF ELECTRICAL & ELECTRONIC ENGINEERING

PROGRAMMING TECHNIQUES I

COURSE CODE – EE271

SUPPLIMENTARY EXAMINATION

JULY 2012

DURATION OF THE EXAMINATION - 3 HOURS

INSTRUCTIONS TO CANDIDATES

1. There are FIVE questions in this paper. Answer any FOUR questions only.
2. Each question carries equal marks.
3. Use correct notation and show all your steps clearly in any program analysis.
4. All programs should be well documented and indented for clarity.
5. Start each new question on a fresh page.

Question 1

a)

i. Explain the rationale for using iterative control structures in C programming. [1]

ii. Give three examples of iterative control structures. For each control structure given as an example, explain its semantics. [11]

b) Explain the difference between the *break* and *continue* statements. [4]

c) Explain the concept of *recursive functions*. Give an example of where this concept can be used. [3]

d) Explain the following terms in the C programming language:

(i) Variable [4]

(ii) Variable Scope [2]

Question 2

Answer each of the following. Assume that single-precision floating point numbers are stored in 4 bytes, and that the starting address of the array is at location 1002500 in memory. Each part of this question should use the results of previous parts where appropriate.

a. Define an array of type float called *numbers* with 10 elements, and initialize the elements to the values 0.0, 1.1, 2.2, 3.3, 4.4, 5.5, 6.6, 7.7, 8.8, 9.9. Assume the symbolic constant SIZE has been defined as 10. [2]

b. Define a pointer, *nPtr*, that points to an object of type float. [1]

c. Print the elements of array numbers. Print each number with 1 position of precision to the right of the decimal point. [4]

d. Give two separate statements that assign the starting address of array *numbers* to the pointer variable *nPtr*. [3]

e. Print the elements of array numbers using pointer/offset notation with the pointer nPtr. [5]

f. Assuming that *nPtr* points to the beginning of array numbers, what address is referenced by *nPtr + 8*? What value is stored at that location? [3]

g. Assumming that nPtr points to numbers[5], what address is referenced by nPtr-=4. What is the value stored at that location. [7]

Question 3

Using **only** *recursive* functions (NO repetition statements), write a program that displays the following checkerboard pattern: [25]

```
* * * * * * * * * *
* * * * * * * * * *
  * * *       * * *
    * * * * * * *
      * * * *
      * * * *
    * * * * * *
  * * *       * * *
* * * * * * * * * *
* * * * * * * * * *
```

Your program must use **only** three output statements, one (or more) of each of the following forms:

```
printf("* ");
printf(" ");
printf("\n");
```

Question 4

Carefully analyse the program shown in Figure 5 and determine its output. Show all working. [25]

```
#include <stdio.h>

int main(void) {

int m, n;
        int x = 1, y = 10;

        for(m=x; m<=y; m++) {
                for(n=x; n<=y; n++) {
                        if((m<=x+1)||(m>=y-1)) {
                                printf("* ");
                        } else {
                          ((y-n+1>=m-1)&&(y-n+1<=(m+1))) ?  printf("* ") : printf("  ");
                        }
                }
                printf("\n");
        }

        return 0;
}
```

Question 5

A company that wants to send data over the internet has requested you to write *two* C programs.

The *first* program encrypts the data so that it may be transmitted more securely and reduce the risk of unauthorised users reading it. All data is transmitted as *four-digit positive* integers. Your program should read a four-digit positive integer entered by the user, encrypt and print the number using the following *encryption scheme*:

Replace each digit with the result of adding 7 to the digit and getting the remainder after dividing the new value by 10. Then swap the first digit with the third, and swap the second digit with the fourth.

The *second* program inputs an *encrypted* four-digit integer and *decrypts* it (by reversing the encryption scheme above) to recover the *original* integer.

Your programs will be graded according to the following criteria:

i. *Correctness* – does the program produce the desired result i.e. for a specified number of lines and printing direction, can the program produce the correct Pascal triangle. [20]

ii. *Clarity* - proper indentation of program makes it easy to read. [1]

iii. *Sensible naming of variables* – make it easy to understand code when debugging. [2]

iv. *Proper use of comments* – comments also make the program easy to understand. [2]

END OF EXAM PAPER