EE 2331 Data Structures and Algorithms, Semester B, 2009/10

Tutorial 8: Generating Bit Sequence

Week 7 (11[h] March, 2010)

The tasks of tutorial exercises are divided into three levels. Level 1 is the basic tasks. You should have enough knowledge to complete them after attending the lecture. Level 2 is the advanced tasks. You should able to tackle them after revision. Level 3 is the challenge tasks which may be out of the syllabus and is optional to answer. I expect you to complete task A, B and C in the tutorial.

Outcomes of this tutorial
1. Able to generalize a problem using recursion
2. Able to distinguish between iteration and recursion

Recursion is a technique commonly used to solve problems with incremental or step-by-step nature.

In this tutorial, we will apply iteration and recursion to work on the same problem.

Task A (Level 1): 3-bit sequence

A function generate2Bit() has been given to you. The function is based on iteration to generate all combinations of 2-bit sequence, {00, 01, 10, 11}. Now follow this function to write your generate3Bit() function to generate the combinations of 3-bit sequence.

Expected Output:

```
Enter the number of bit: 3
000
001
010
011
100
101
110
111
```

Task B (Level 1): 4-bit sequence

Use the same iterative approach, implement the function generate4Bit() to generate all combinations of 4-bit sequence.

Expected Output:

```
Enter the number of bit: 4
0000
0001
0010
0011
0100
0101
0110
0111
1000
1001
1010
1011
1100
1101
1110
1111
```

Task C (Level 2): *n*-bit sequence

Now can you generate the combinations of *n*-bit sequence? Can you still use iteration to complete the task? Implement the function generateNBit(int n) to generate sequence for *n*-bit using recursion. Your function should work for all positive *n*.

Expected Output:

```
Enter the number of bit: 6
000000
000001
...
111110
111111
```

Discussion: Can you implement task c using iteration? Why or why not? How to generate the decreasing bit sequence?