

# >>> Assignment #4 for Computer Networks (CNT 4004) <<<

## Due on 11/03/10 at the beginning of class

This assignment covers material from chapter 4 of the textbook and from class lecture.

### **Problem #1**

Implement max-min scheduling. Appendix A contains a template for you to use (you can copy-paste the template to your .c file, which should be named maxmin.c). You need to insert variables and code where so indicated. The program header shows a sample execution.

### **Problem #2**

Is the following a correct description of Dijkstra's shortest path algorithm? If it is not, correct it. "To determine that shortest path from node A to node B in a connected graph add edges (hops) starting at A in order of least edge (hop) cost until node B is reached. This will be the lowest cost path."

### **Problem #3**

Do problem P24 (page 430) in the textbook, *but change the edge costs from v to u to be 7, from u to w to be 6, from w to x to be 10, and from v to t to be 6*. You do not have to show a table similar to table 4.3, you only need to give the order of edges (hops) added step-by-step and the path cost from node x for each node added. If there is more than one solution (that is, multiple paths with equal least cost), given them all. Identify an edge by its two end vertices (e.g., hop xz is the edge between nodes x and z).

### **Problem #4**

Do problem P1 (page 424) in the textbook.

### **Problem #5**

Do problem P6 (page 426) in the textbook.

### **Problem #6**

Do problem P9 (page 427) in the textbook.

### **Problem #7**

We have talked a little about switch architectures – the key "hardware" within IP routers (and also LAN switches we will learn). Who invented the input buffered (also called input queued) switch and what is the main problem with this switch architecture? How is this problem solved? I do not believe that the answer is in the textbook (I suggested firing-up scholar.google.com – never used scholar.google.com you say?). All referenced papers must be properly cited (that is, use correct bibliographic style).

### **Note:**

The TA and I are here to help you! Make use of help if you need it.

## Appendix A – maxmin\_template.c for Problem #1

```

//===== file = maxmin.c =====
// Program to implement max-main scheduling
//=====
// Notes: 1) Takes input from console and writes output to console
//         2) Limited to MAX users (set in #define)
//         3) Users must be entered in order (lowest to highest) by
//            by demand
//-----
// Example execution:
//
// ----- maxmin.c -----
// - Program to implement maxmin scheduling -
// -----
// Capacity =====> 10.0
// Number of users =====> 4
// User demand from lowest to highest for 4 users...
// Demand for user # 0 =====> 1.0
// Demand for user # 1 =====> 3.0
// Demand for user # 2 =====> 6.0
// Demand for user # 3 =====> 7.0
// -----
// Allocated capacity is:
// User # 0 = 1.000000
// User # 1 = 3.000000
// User # 2 = 3.000000
// User # 3 = 3.000000
// -----
// Build: bcc32 maxmin.c
//-----
// Execute: maxmin
//-----
// Author:
//-----
// History: KJC (10/07/10) - Genesis of maxmin_template.c
//=====
//----- Include files -----
#include <stdio.h>           // Needed for printf()
#include <stdlib.h>         // Needed for ato*()
#include <assert.h>         // Needed for assert()

//----- Defines -----
#define MAX 10              // Maximum number of users

//===== Main program =====
void main(void)
{
    char    inString[256];    // Input string
    double  capacity;        // Capacity of system
    int     numUser;         // Number of users
    double  demand[MAX];     // User demand
    double  allocate[MAX];   // User allocations
    // *** insert other variables as needed here ***
    int     i;              // Loop counter

    // Output banner
    printf("----- maxmin.c ----- \n");
    printf("- Program to implement maxmin scheduling - \n");
    printf("----- \n");

    // Prompt for capacity
    printf("Capacity =====> ");
    scanf("%s", inString);
    capacity = atof(inString);
}

```

```

// Prompt for number of users
printf("Number of users =====> ");
scanf("%s", inString);
numUser = atoi(inString);
assert(numUser <= MAX);

// Prompt for demand for each user
printf("User demand from lowest to highest for %d users... \n", numUser);
for (i=0; i<numUser; i++)
{
    printf(" Demand for user #%2d =====> ", i);
    scanf("%s", inString);
    demand[i] = atof(inString);
}

// Clear allocate vector
for (i=0; i<numUser; i++)
    allocate[i] = 0.0;

// Main scheduling loop

// *** Insert max-min scheduling code here ***

// Output allocations
printf("----- \n");
printf("Allocated capacity is: \n");
for (i=0; i<numUser; i++)
    printf(" User #%2d = %f \n", i, allocate[i]);
printf("----- \n");
}

```