Computer Science 3711 (Fall 2005):
Assignment #2, Questions #2–6
Due: 4:00 PM on Friday, October 7, 2005

2. (10 marks) For each of the situations described below, state the corresponding computational problem in terms of its inputs and outputs along the lines of the statement of the sorting problem on page 5 of the textbook. Where possible, please phrase the entities in these problems in terms of abstract mathematical structures, *e.g.*, sets, functions, graphs, nodes, edges.

a) (5 marks) The Avalon Intelligence Agency (AIA) is looking to distribute $k > 0$ copies of its latest secret code handbook amongst its agents. Each agent knows a subset of all the other AIA agents and can share his or her copy of the codebook among them. The AIA Director wants as many agents as possible to have access to the new codebook. The AIA Internal Security Division wishes to minimize the chances of losing a codebook if an agent in a group that shares a codebook is compromised; hence, it has issued a directive that no agent in a group that shares a codebook can know any AIA agent in any other group that shares a codebook. The AIA Code Division wishes to know the size of the largest group of agents that can have access to a codebook subject to the wishes of the Director and the Internal Security Division.

b) (5 marks) The Canadian Aeronautics and Space Administration (CASA) is preparing to send a team of astronauts on a long-duration deep space mission. The Nutrition Division is responsible for preparing meals for the mission. Each meal consist of three courses, and each course satisfies the daily requirements of a subset of the official CASA list of daily required nutrients. A valid meal consists of three courses which, as a group, satisfy all daily requirements of nutrients on the CASA daily required nutrient list. The astronauts have been complaining about the lack of imagination in meals on the last several missions. Given a list of courses and their nutrient specifications, the Nutrition Division would like a list of all possible valid meals that can be built up from these courses.

3. (10 marks) Prove or disprove the following:

b) (5 marks) $f(n) = n(n + \log n)$ is $\Omega(n^2 \log n)$

c) (5 marks) $f(n) = 2^{n-d}$, where d is an integer constant greater than or equal to 10, is $\Theta(2^n)$

All proofs should give appropriate bounds on values of c and n_0.

4. (14 marks) Solve the following recurrences using the iteration method:

a) (7 marks)

$$T(n) = \begin{cases} 27 & n \leq 4 \\ T(n-4) + cn^2 & n > 1 \end{cases}$$

b) (7 marks)

$$T(n) = \begin{cases} 3 & n \leq 1 \\ 8T(n/2) + cn & n > 1 \end{cases}$$

5. (8 marks) For each of the algorithms below, derive a parameterized asymptotic worst-case time complexity function.

a) (4 marks)

```
sum = 0
tsum = -15
for i = 1 to n do
    x = P1(n)
    sum = sum - x + 5
    for j = 1 to n * n do
        y = x / (P2(n) + P1(n))
    if (P3(n))
        if (P4(n))
            tsum = tsum + tsum
     else
        for j = 1 to log(n) do
    if (P4(n))
                y = y * i - j
    tsum = tsum / y
sum = sum - tsum * tsum
```

b) (4 marks)

```
tsum = -15
for i = 1 to n do
    x = P2(n)
    sum = sum - x + 5
    for j = 1 to n * n * n do
        y = x / P3(n)
    if (P4(n))
        tsum = tsum * x
        if (P1(n) and not P4(n))
            tsum = tsum + tsum
    else
        for j = 1 to log(n) do
            y = y * i - j + P2(n)
    tsum = tsum / y
```

6. (8 marks) For each of the recursive algorithms below, derive a recurrence describing the exact running time of that algorithm.

a) (4 marks)

```
FUNKY-REC1(n, x)
    if (n <= 2)
        print x
    else
        for i = 1 to n * log(n) do
            if i is odd then
                x = x + i
            else
                x = x - 1
        FUNKY-REC1(n - 2, x)
        x = x * x
        FUNKY-REC1(n - 3, x - 1)
        for i = 1 to n do
            x = x / i
        FUNKY-REC1(n - 2, x)
```

b) (4 marks)

```
FUNKY-REC2(n, x, k)
    if (n <= 3)
        print (x - 13 * n)
    else if (n <= 10)
        for i = 1 to n * k do
            print x
    else if (n is odd)
        FUNKY-REC2(n / 3, x - 1, k)
        for i = 1 to n do
            x = x / i
        FUNKY-REC2(n / 3, x - 5, k)
    else if (n <= 6)
        for i = 1 to k * log(k) do
            print x / n
    else
        for i = 1 to n * log(n) do
            x = x * i
        FUNKY-REC2(n - 3, x * x, k)
```