**Program 3 Report**

Xiaoyong Liang

XL5432

**Problem**

Choose the most optimal number of hours to spend on each class during final projects given certain GradeFunction, number of classes, maximum number of hours, and max grade.

**Solution**

1)Given numOfClass N, totalHours H, gradeFunction gf(i, h)

2)Let MaxGrade(i, h) be the maximum total grade for this problem.

3) Construct a 2D array where MaxGrade(i,h) gives the maximum grade for total *i* classes and *h* hours. BestHr(i, h) gives the best hour *subHr* for class *i* given total hour is *h*.

4) Set MaxGrade(0, h) to 0 for all *h*. (Base Case)

5) Set BestHr (i, h) to 0 for all *i* and *h*. (Base Case)

6) Set MaxGrade(i, 0) to sum of grade function of class id *j* for *j* from 1 to *i*. (Base Case)

7) For MaxGrade(i, h), spend subHr hour on course *i* for some subHr from 0 to *h*.

8) Compute each MaxGrade(i,h), and BestHr(i,h) with the recursive definition below (in row major order with classes as rows).

9) At (i = N, h = H), trace back the value of BestHr(i, H) subHr. Then go to (i-1, h-subHr) and repeat until reaching the start of the array.

**Recursive Definition**

MaxGrade(i, h) = Max for *subHr* from 0 to *h*:

gf(i,k) + MaxGrade(i-1, h- subHr)

BestHr(i, H) is then *subHr*

or relation is T(H) = T(H - 1) + O(1) where the sub-problem is constructed so that with each recurrence, this algorithm reduces the number of remaining hour to find by 1.

This solution provides the optimal substructure because it always produces the optimal next hour schedule at each iteration given an optimal schedule for current hour. With correct base cases(which is obvious and can be proved very easily), this solution is optimal.

**Tests**

Beside the tests for the solution, there are basic unit tests in this project to ensure

1. The program does break with unintentional inputs.
2. Cap gf(i,h) at maxGrade anyway even if it produces grades less or equal to maxGrade
3. Total hours are always spent, with no “leisure time” even if it already produces optimal solution.

For the correctness of the solution, I created test cases where the last class id produces the most grade per hour. In testing so, I found a bug where the base case is wrong for MaxGrade(i,0): instead of the sum of gf(j,0) for j=1->i, it was gf(i,0).

Some of the test cases are included in UnitTests.java (some of them is unit tested where others are tested by inspection).