LAB 3 REPORT

Optimal Substructure - We can calculate the maximum grade for a certain number of classes by taking the maximum result of the previous class. We have to take into consideration all hours for the current class we are considering and compare these results to the previous result at the current hour we are considering. Maximizing the grade between all of these comparisons will give the maximum grade possible.

Recursive Definition - The recursive function G[class,hour] is the highest grade possible at that point. We will also define G[class,hour] equal to 0 when class is equal to zero.

G[class,hour] = max{max{G[class-1,hour - i] + fclass(i)}, G[class-1][hour]} where 1<=i<=hour
While running this algorithm we have a second matrix which represents the hours used by a particular class. This will keep track of the values throughout its execution in order to reconstruct the solution after finding the maximum score. H[class,hours] = -1 if G[class][hour] < G[class-1][hour] and H[class,hours] = the current time if G[class][hour] > G[class-1][hour]

Iterative Algorithm -

```
func(hours)
  Initialize G[class, hour]
  For all classes
        For all hours
              Find i that maximizes G[class-1][hour-i] + grade of class at i
              G[class][hour] = max of the maximum of above

Find solution by finding best G for each class
Return solution
```

Testing Solution - MyGradeFunction was used to test the case where studying for a class for the maximum time resulted in a maxGrade but studying for two classes separately resulted in a higher overall grade. This tests the program for spending its time wisely to maximize the overall grade.