**ECE 4318 Final Part 1 Individual**

**Date:** 12/5/20 Saturday

**Due Date:** 12/8/20, Tuesday, 10 p.m.

**Total Points:** 126 points

**Total Grade Points:** 10 grade points, 100 points = 10 grade points

**This is ECE 4318 final part 1. You are supposed to work alone.**

**Submit to Blackboard**

**Classes (Object Oriented Programming)**

1. **(40%) Class String**

You have learned C type string in ECE 1310 / ECE 114 and also C++ / C# / Java type string, which is an Object Oriented class.

C type string is when you declared a string to be a char array or a pointer to char like

char [20] = “Happy Birthday”; // you have to #include <cstring> or #include <string.h>

C++ / C# / Java type string is like

String s = “Happy Birthday”; // you need to use #include <string> if that is C++ (class String is intrinsic in C# and Java)

Now build a class myString from scratch that has virtually the same features as the class String in the object oriented programming languages OOP C++ / C# / Java etc.

1. **(5%) Build a class myString** (that has the default constructor, copy constructor etc.) with the necessary attributes (with the necessary public and private access identifiers as in OOP. You also need to make sure you can initialize strings as empty string or string like “Happy Birthday”.
2. **(5%) Add t**he method **Compare** that you can use to compare two strings (like s1.Compare(s2) where s1 and s2 are two strings declared using myString s1, s2; )
3. **(5%) Add** the method **Copy** that you can use to copy a string s2 to another string s1. You can also use the “=” operator for Copy (like in C++ with operator overloading).
4. **(5%) Add t**he method **Concatenate** so that you can concatenate a string s2 to another string s1 like s1.Concatenate(s2) (so if s1 = “Happy” and s2 = “ Birthday”, then s1.Concatenate (s2) is “Happy Birthday) (you can use “+” operator as the concatenate operator, so instead of s3 = s1.Concatenate(s2), you can have s3 = s1 + s2; for a simpler syntax)
5. **(5%) Add** the method **Length** so that you can find out the length of the string as object of your class called myString.
6. **(0% - 10%) Add** any other common string methods such as toUpperCase, to LowerCase, substring etc.
7. **(10%) Test your class myString** using at least three strings. Show that your methods in part (b) through (f) work with your test data. Suggestion: use strings from say “It was the best of times, it was the worst of time.,,,” from A Tale of Two Cities (Dickens), or “Four scores and seven years ago, our fathers brought forth on this continent, ..” from Gettysburg Address (Lincoln), or any other classics from English literature, American English literature etc.
8. **(5%) Compare *your test results with the results if you use the OOP class String*.**
9. **(0-10%) Discuss** the possibility of your myClass to be extended to process strings in languages other than English, such as Spanish, French, German (European languages) or more broadly some Asian languages.
10. **(43%) Class Matrix**

You are familiar with matrices, 2x2, 3x3 nxn, or even rectangular mxn from linear algebra or Mat 224.

MATLAB by Mathworks is a powerful tool that you can perform lots of matrix operations.

A matrix is actually just a two dimensional array (that you learned in ECE 1310 / ECE 114) with the new operations Add, Subtract, Multiply, Determinant, Inverse etc. You knew how to do such Matrix operations by hand from Mat 224 and had done similar operations using MATLAB.

1. (4%) Build a class myMatrix with the necessary constructors (and a two dimensional array as the data, where the array can be in int, double, etc., the array can be 2x2, 3x3, 3x5, or in general mxn where m and n may be equal or not).
2. (4%) Add the trivial methods Add, Subtract, ScalarMul (multiply a matrix by a constant)
3. (12%) Add the tricky method Multiply (if you are not careful, the product of two matrices using your code may be different from the answer you have by hand or by MATLAB)
4. (4%+) Add the method Determinant (which is trivial for 2x2, may be not that for 3x3, and for general nxn, you need maybe Gaussian Elimination algorithm). You can start here with just 2x2 and 3x3 matrices
5. (4%+) Add the method Inverse. Again, you can start with 2x2 and 3x3 matrices.
6. (10%) Test your class myMatrix with A = , B = , and C = =

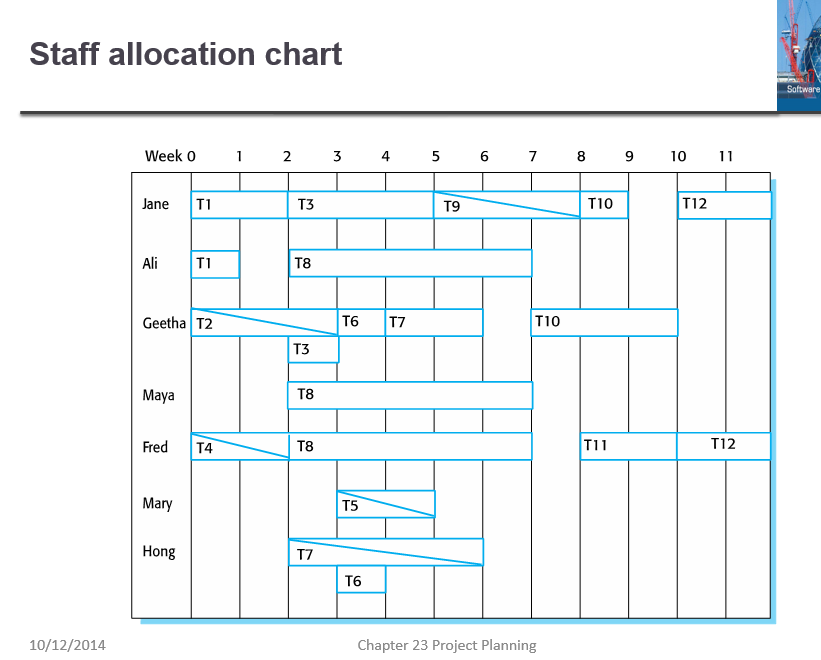
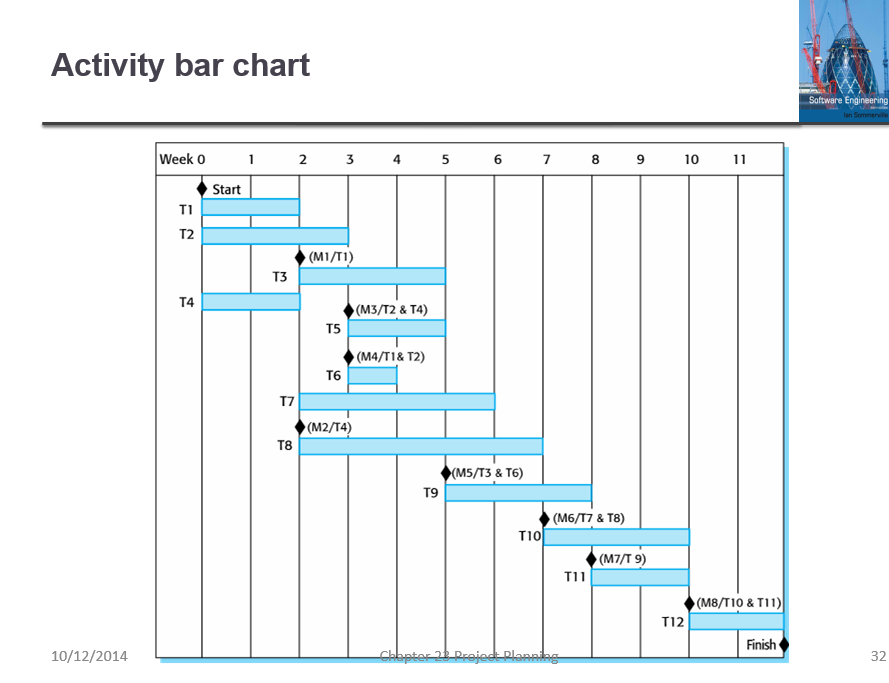
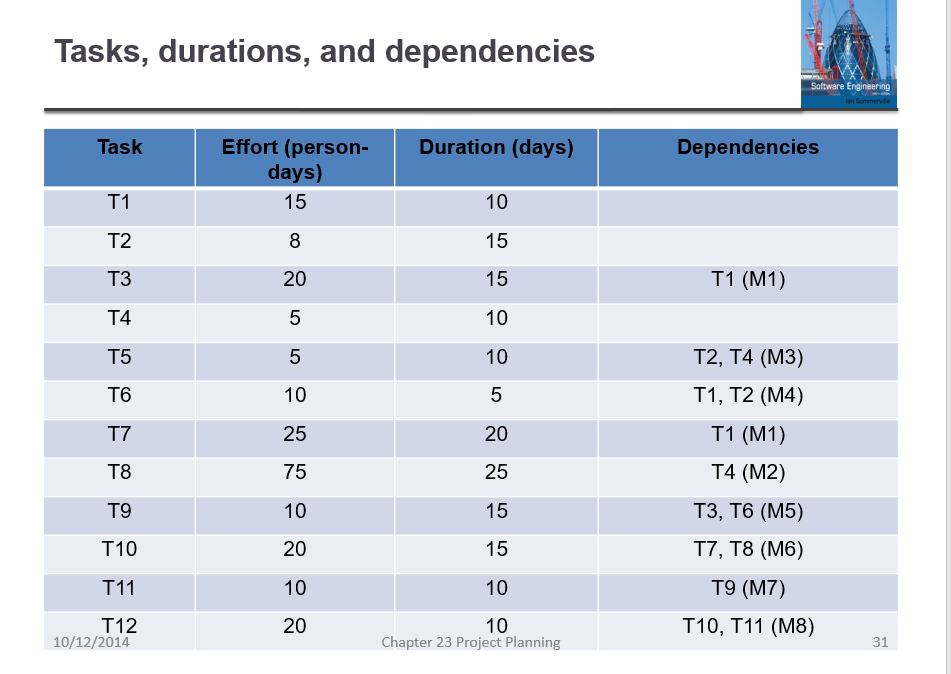
**Compute** AB (A multiplies B using your Multiply method of part (c)), BA, A10, B10, C10, and also , A100, B100, C100. All the powers of matrices here should be computed using your Multiply method in part (c).

1. (5%) Compare your test results of part (f) with those from either manually or from MATLAB etc.

**Scheduling (43%)**

1. **(43% +) Scheduling (in Projects)**

Recently you have learned scheduling (a project) from chapter 23 Project Planning with slides 31, 32, and 33 as follows:



Slide #31 #32 #33 of chapter 23, Sommerfeld

**Enlarge** these slides if necessary (I reduced them here) for you to read.

1. **(20%) Verify** if these 3 slides present consistent data. There is possibility that these 3 slides are not consistent. For example, it is possible that T1 shows efforts of 15 person-days and duration of 10 days in slide 31, but in slide 32 Gantt charts it shows longer or shorter than 10 days, also in slide 33, it is possible that the employees allocated to task T1 do(es) not really use 10 person days (and 15 calendar days).

In the case of T1, it seems fine since in slide 32 we do see that T1 is from the beginning of week 0 till the beginning of week 2, so duration of 10 days is checked. In slide 33 we see that Jane is working full time from start of week 0 till start of week 2 for 10 days and Ali is working full time from start of week 0 till start of week 1 for 5 days, hence efforts 15 person days (15 = 10 + 5) is also checked.

**Make a table** to facilitate your answering (and my grading).

Table 1. Task scheduling and staff allocation

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Task | Effort | Duration | Dependency | Staff 1 | Staff 2 | Staff 3 | Remark |
| T1 | 15 | 10 |  | Jane 100% | Ali 100% |  | OK 31,32,33 |
| T2 | 8 | 15 |  |  |  |  |  |
| T3 | 20 | 15 | T1 (M1) |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

**Expand the table** to 12 rows for the 12 tasks in slide 31. **Add** columns or ways to clearly show me that you have checked all 12 tasks (with all the staff of 7 employees)

1. (3% - 10%) **Discuss** on if part (a) can be done by computer software (for project manager and for someone who comes to audit the project should there be 200 tasks, 30 months, 50+ employees, for a multimillions project)
2. (12+%) **Assume** T1 is delayed 3 days and T2 is delayed 2 days. Find out how many other tasks are delayed. How late will the project be delayed?
3. (8%) *Compute* the man-weeks effort of the original project (if an employee works 3 weeks, it is 3 man-weeks, 2 employees work 3 weeks, it is 6 man-weeks). *Compute* also the man-weeks of the delayed project assuming T1 is delayed 3 days and T2 is delayed 2 days (then many tasks but maybe not all tasks and milestones are affected)
4. (0% - 6%) Discuss the possibility of the possibility of doing part (c) and (d) by software.