**ECE 4318 Final**

**Date:** 12/4/21 Saturday

**Due Date:** 12/8/21 Wednesday. Please let me know if you need extension of 1 to 2 days

**Total Points: 132** points

**Total Grade Points:** 20 grade points, 100 points = 10 grade points

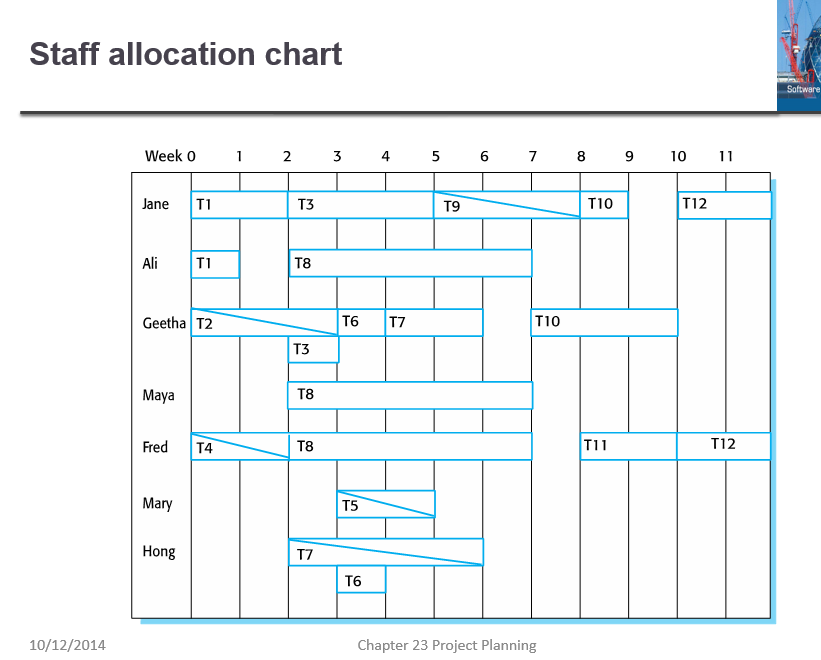
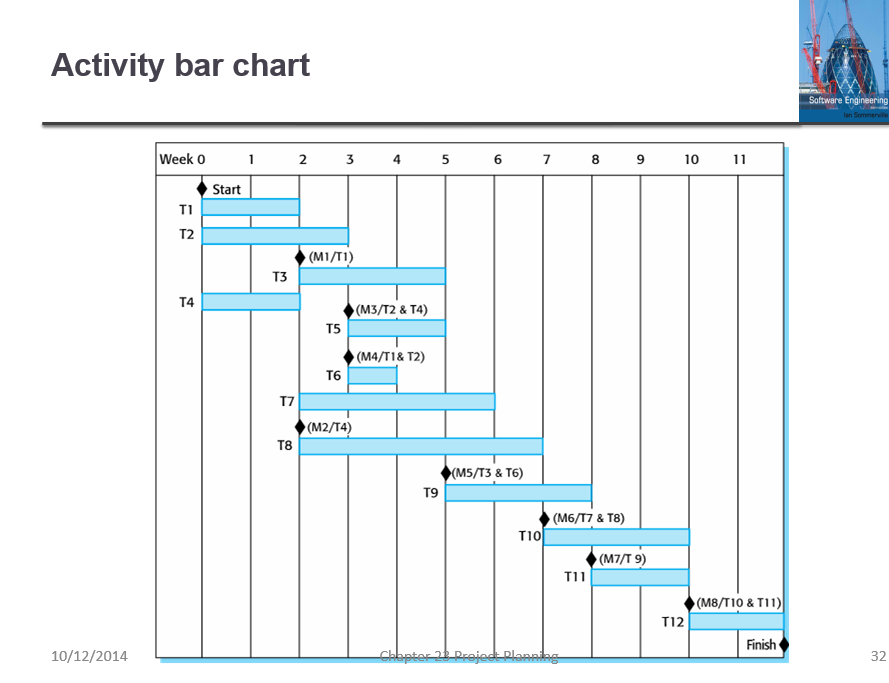
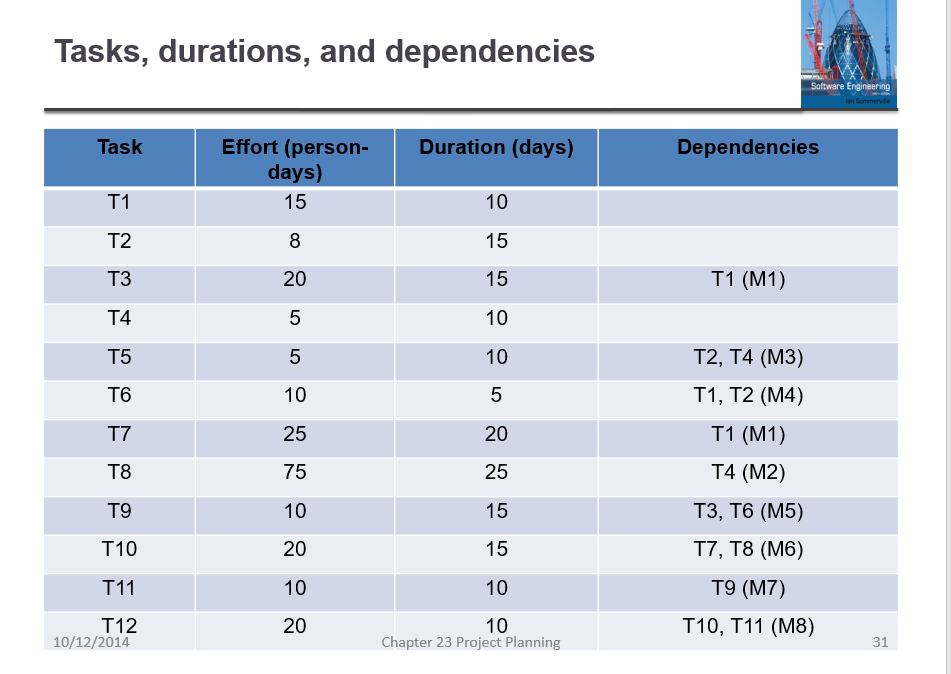
You will work alone (take home).

**Submit to Canvas (**remember to put your name inside the file like JohnDoeECE4318Final)

**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 of doing part (c) and (d) by software.

**Bad Smells in Program Code (15%)**

1. (15%) Bad smells. In chapter 9 on evolution, there are two slides (#57 and #58) talking about bad programming practice like follows: in slide #57, duplicate code, long methods, switch statement; in slide #58, data clumping and speculative generality. A total of 5 different kinds of problems were explained in slides 57 and 58.

You had written code in report 2 (several programs), report 5 (email).

1. (7%) No such problems in your code (or if there is, how would you fix). **Explain** briefly that you do NOT think your code in any of report 2, 4, 5, or 6 has such “bad smells”. It is impossible to include all your code in the answer here, but try to have one or two code snippets (of 5 to 10 lines) showing how you code and that you do NOT have such problems in the code snippet you explain here
2. (8 + %) **Examples of code** with “bad smells”. You can either search internet to find some programs that commit at least one of the bad smells problem or you can “make” one program that has at least one out of the 5 problems. Note the problematic source code should compile and run without logic errors. The only problem as mentioned in #57 and #58 should be called **style problem**. You can earn more than 8 points for this part if you find or create program with more than one problem
3. (5%+ \*). Discuss if there may be other “bad smells” that Sommerville did NOT mention in slides #9.57 and #9.58.

**Programming Questions (45%)**

There are 3 programming questions here (of the level of ECE 1310 or C programming) from Q3 to Q5. For the programming questions, you need to turn in the source code (*separately*) and the outputs.

Note: Source code alone (without output) will earn 0 points. Outputs alone without source code also will earn 0 points.

In addition to source code and outputs, include a few lines of source code snippet in the Word (or PDF) of your answer, and explain.

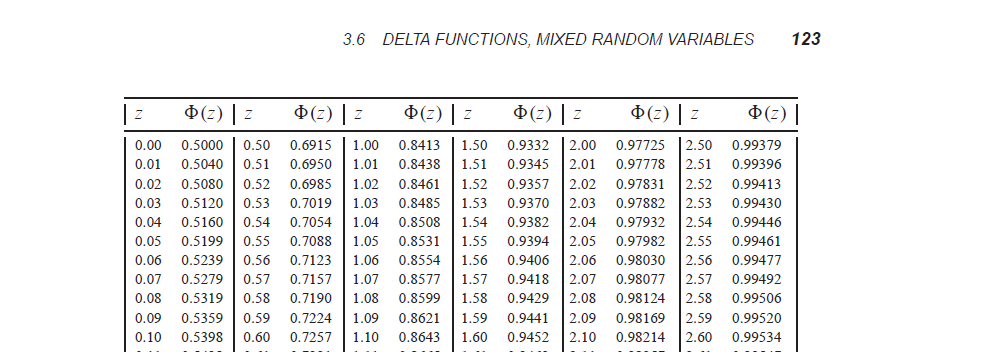
Warning: do NOT turn in the huge stupid C / C++ big folder under .vs hidden folder or ipch folder of 20M bytes, 30 M bytes.

**Generate Gaussian distribution table: 15% (ECE 3715)**

1. (15%) Gaussian Distribution table *(please turn in the source code as well as showing outputs in the Word file). File for this question should be at most 2M bytes.*

Gaussian distribution is used a lot (normal distribution with the bell curve). Here is a table used in ECE 3715,

with the top of table captured as follows:



Please see the separate PDF file (on Canvas, in the new module called Final Supplements) called Probability and Stochaistic Processes.pdf, chapter 3.5 from page 118 till page 122 of explanation of Gaussian distribution and table 3.1 on page 124 that presents the data of the normalized Gaussian distribution (with mean  = 0 and standard deviation  = 1, also labelled as Gaussian (0, 1)) for z = 0 until z = 2.99 in increments of 0.01.

1. (12%) You have learned Trapezoidal rule (or Simpson’s rule) in integral calculus (Mat 115). Using that rule plus C programming (or Python etc.) to **generate some table similar to table 3.1 from z = 0 till z = 4.99** (> 2.99) (table 3.1 in the book is till z = 2.99).
2. (3%) **Verify** your table agrees with table 3.1 from z = 0 till z = 2.99.

**Compute the roots of function: (15%)** *(please turn in the source code as well as showing outputs in the Word file). File for this question should be at most 2M bytes.*

1. (15%) Consider the cubic polynomial function f(x) = x3 + x + 1. We know that this function is strictly increasing (i.e. if x1 < x2, then f(x1) < f(x2)).
2. (3%) **Prove** that this function is strictly increasing using differential calculus (Mat 114)
3. (12%) We know f(-1) = -1 – 1 + 1 = -1 < 0 and f(0) = 1 > 0. So from calculus, we know that there is a root (x0) such that f(x0) = 0 with -1 < x0 < 0.

**Write** a computer program in C / C++ / C# / Python to calculate this root to at least 5 digits precision (do NOT use MATLAB’s root command). You may use algorithm like bisection, Newton etc.

(you may use the PDF from Burden and Faires chapter 2. Pdf from EGR 5110 I instructed. The PDF is also in the module Final Supplements).

1. (15%) Exponential Function: *(please turn in the source code as well as showing outputs in the Word file). File for this question should be at most 2M bytes.*
   1. (9%) Write a function myexp(x) that computes

 . . **Use precision**, not the number of iterations to stop the loop. **Print** out the values for myexp (1), myexp (2), myexp (5), myexp (0), and myexp (-1).

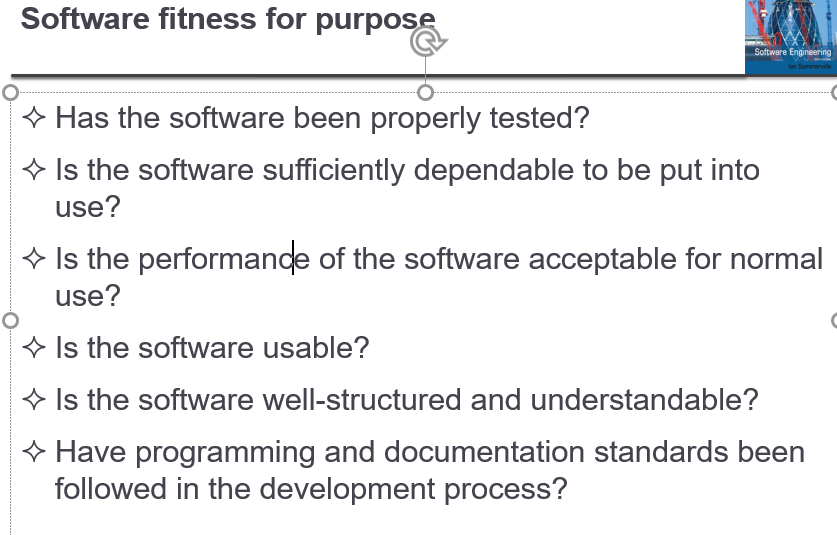
* 1. (3%) Compare *programmatically* the values of myexp functions for the values 1, 2, 5, 0, and –1 with the math function exp (x) in <math.h>
  2. (3%) Compare with the values e2, e5, and e-1 computed using a calculator or using *pow* function. We assume e = 2.71828

**Testing** (9%)

1. (9% + ) In chapter 8 you learned testing. **Explain** what kind of test data you used to verify that your answers for Q3, Q4, and Q5 above are correct. **Why you think** your set of test data (or the ways you used to verify) is enough or adequate? You can earn more than 9 points if your answer is good / convincing.

**Software Fitness** (10%+)

1. (10%+) Slide 24.11 Software Fitness for Purpose (in chapter 24 on Quality Management) is interesting like below (with 6 bullets)



**Elaborate** on 1 or 2 bullets here (or more if you can). You can use software you know (such as Microsoft Word), your term project or senior project software, or some software you plan to develop etc. You may earn more than 10 points if your answer is great.

**Software Reuse** (10%+)

1. (10%+) Chapter 15 on software reuse that I covered recently seems to contain a lot of good stuff, worth of studying, expanding etc.

**Write and elaborate** at least two (2) things on software reuse you know (from chapter 15’s slides or from your own experience etc.). You may earn more than 10 points if your answer is great.