CSE 20212 Fundamentals of Computing II
Spring 2016

Lab handout for weeks of Feb 29 and March 14 (Labs #7 and 8)
Due 30 minutes before your next lab after break (week of 3/21), no exceptions!

Objectives

• Develop a plan and initial deliverables for your final project with help of the TAs
• Finish the current lab if needed
• Start to work on your project

In-lab activities

1. (1 point) Report to lab on time. Attendance will be taken at the scheduled lab time this week (#7) and the lab after break (#8).

2. This week and the week after break will be open-ended by design: I want to provide you the opportunity to learn OOP/C++ through your projects, but we also recognize some of you may need help with Sudoku.

3. At a bare minimum, you are required to develop and deposit “lab 8 milestones” into your drop box before leaving lab. This, in short, will be a simple initial set of requirements that you can reflect on for Part 3 of the actual rubric (see below). Please generate a rubric for 40 points with weighting of your choice.

4. Relatedly, it is normal if you have difficulty with accurately weighting tasks or determining what is easy or hard. The important aspect of this thought experiment, historically, is as a highly efficient exercise to help you and your group prioritize and get started as a team.

5. Committing to doing SDL tutorials can be sufficient, but we will require something that can be compiled for full credit this lab and at least one non-tutorial deliverable per group member. Example are moving a shape on the screen or embedding an image on a surface like the brief example we’ll discuss from 3/2.

6. You are welcome to collaborate with your group but you must:
   a. Put in the required time (see next page)
   b. At least one compilable task following the tutorials must be assigned and committed by you.
   c. You must all use the same Git/Bitbucket repository. If need help with Git/Bitbucket for group development, please ask a TA for help.

7. Discuss your proposed plan to TAs before leaving lab for participation credit. If this was completed prior to lab you can leave lab early if you also can demonstrate a successful commit to your group’s repository.
Preparing for the grading rubric

In addition to your normal report, we will request the following for grading (see Piazza for official rubric):

1. A time log of what you worked on for the project, which will be included in your final report to Prof. Emrich. We expect roughly >= 7 hrs of work including (a little) non-Sudoku time this week and the in lab portion following spring break.

2. Evidence of a repository and code that you have checked in that matches your assigned task(s).

3. A rubric that continues your development effort. If you decide to move a task from in-lab to a “next step”, please detail in your report why and how we might be able to help overcome technical issues. Examples include but are not limited to “task much more challenging than expected” and an attempt to explain why. This rubric is solely for planning and feedback purposes; it will not relate to the next lab assignment.

Handing it in

Please read and follow the general lab guidelines available on the course website:
http://www.cse.nd.edu/courses/cse20212/www/labs.html

We will require a longer report.txt this week (see above and the rubric). If you have any questions about this lab, please use Piazza.

Coder challenge! Supermarket sweep!

The knapsack problem (or shopping cart problem in this example) is as follows: Given a set of items, each with a volume and a value, determine the number of each item to include so that the total volume is less than or equal to a given limit and the total value is as large as possible. Note that this is a computationally hard problem to find the best solution but simple solutions also exist. You are encouraged to use your ingenuity to find the fastest/best solution for full extra credit. The TAs will run your solution for at most 30 minutes so you should report one solution and ideally improve it over time.

For this problem lets imagine we have a huge cart that is a cube with a dimension of 7ft (so 343 sq ft for our shopping frenzy). For simplicity, you can assume that items can be mushed as needed and therefore optimally crammed in (but you must take 100% of a given item). Your input is therefore simply pairs of occupied space and value per item (two floats per line). The best answer so far should be sent to a file as follows: report a net value on the first line and how many of each item to include from the input file list (to be provided shortly). An example output file will also be posted to the course website. This will aid in grading.
Any valid solution (which can be done in ~30 minutes by everyone in class) will receive $5 coder dollars. This could be the “naïve” (but fun) solution of filling the cart with the first k items greedily until there is no more space. For an analogy, this is akin to filling the cart with cereal or large packs of cheap instant noodles if they are in the first aisle even though the value per volume of expensive meat is higher in aisle 5.

$15 additional credit points will be awarded to the highest value per volume calculated in at most 30 minutes. In the event of ties (same calculated value) we will pick two winners: one based on efficiency (time to solution) and another “TA choice” based on subjective analysis of the code for the solution.