COMP2401 - Assignment #1

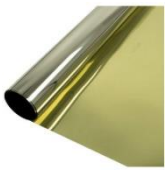
(Due: **Sun. Jan 26, 2020 @ 6pm**)

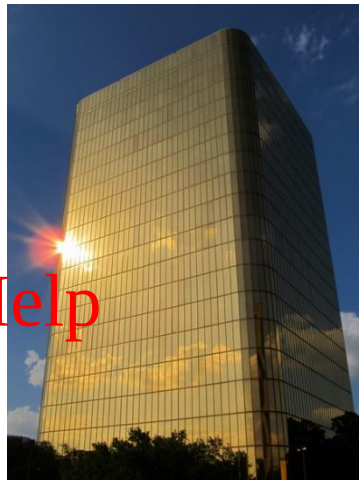
In this assignment, you will create some simple programs in C to get used to the language (i.e., control structs, variables, arrays), the Linux environment, and the compiling/running process.

ALL of your code for both questions must be written neatly with proper indentation and have a reasonable amount of comments. If you do not, you will be losing multiple marks.

---

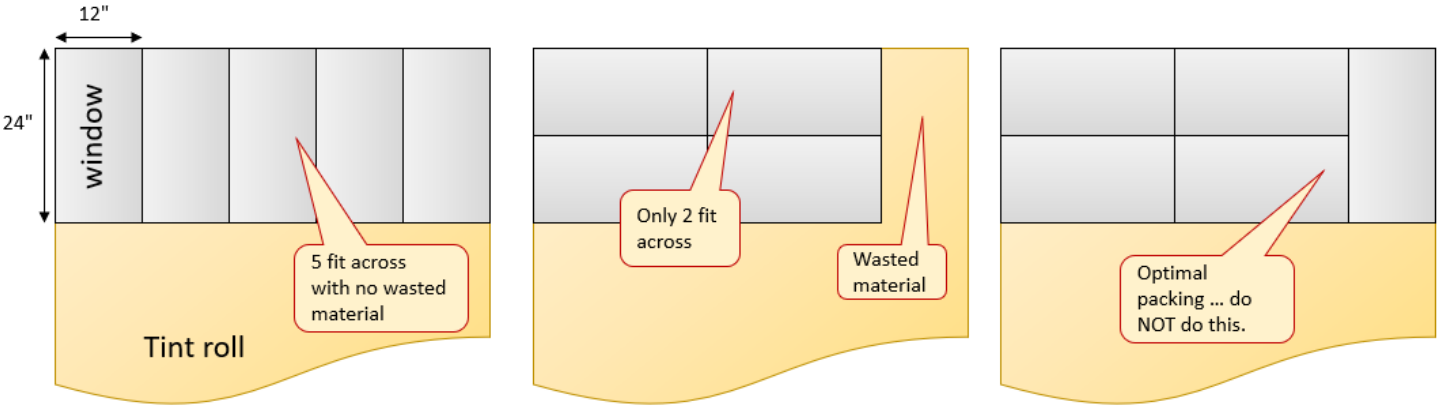
(1) Window Tinting





Assume that you have a company that tints building windows. You have found a good price on nice gold tinting film that comes in rolls that are **5** feet wide and **92** feet long.

You want to write a program that determines the number of rolls that will be needed to be purchased to cover a building completely. Your program should assume that each window is the exact same size in the building and that the windows are always less than or equal to **5** feet wide. The program should ask the user for the number of floors in the building, the number of windows on each floor (includes all sides of the building) and the dimensions of the window (in inches). The program should then calculate the number of rolls needed. Note that when tinting a window, it must be done with one entire piece of film. That is, you CANNOT cut pieces of film to fit on the windows. You should determine whether the 5 foot width of tint roll is able to cover two or more windows. For example, if the windows are 12" x 24", then you can cover five windows with just two feet of roll … but if placed sideways, you can only cover 4 windows, with a little waste left over (see picture below). You should choose the best option in this case. However, you must not apply any other "packing/fitting" algorithm … for the purpose of marking consistency. Just choose which direction to put the tint on … up and down … or left to right.



The program should output the correct number of tinting rolls that need to be purchased. Make sure to define appropriate constants in your code, as this represents proper coding style. Also, make sure to test adequately.

(2) Scheduling Reservations

Write a program called **reservations.c** that simulates some people trying to reserve seats at a restaurant for dinner. The program **MUST** work as follows:

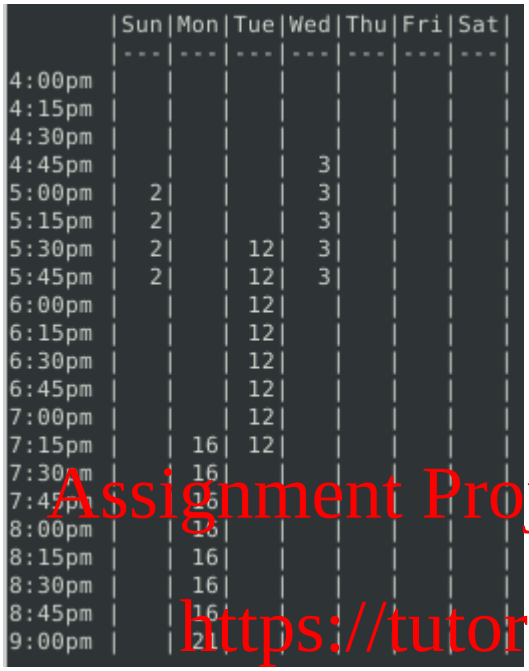
- The restaurant takes reservations anywhere from 4pm until 9pm. That means … someone can phone in and ask to reserve a table for a certain number of people starting at some specific time in the range of 4pm until 9pm. The people are allowed to stay past 9pm, but no more reservations can be made after 9pm.
- The restaurant can seat at most **80** people at any time, on any day of the week. Reservations may be made at **15**-minute intervals starting at 4pm. Hence, 4:00pm, 4:15pm, 4:30pm, 4:45pm, 5:00pm, 5:15pm, …, 8:45pm, 9:00pm.

- The program MUST keep track of reservations by using a two dimensional array to indicate the number of patrons in the restaurant at any given timeslot.

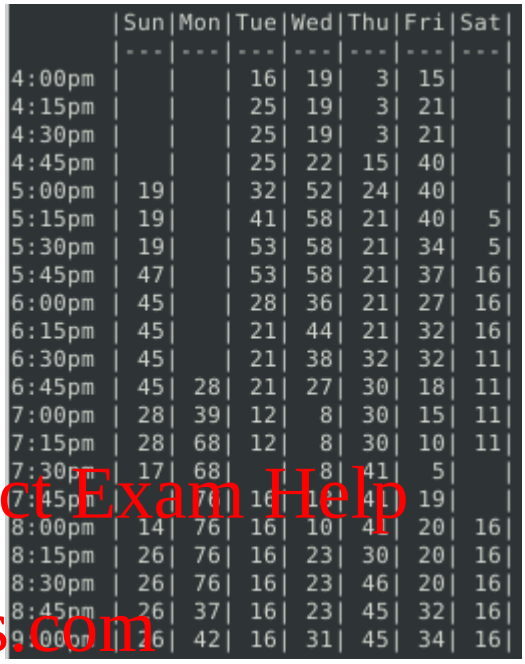
- The program must attempt to make 100 random (must be unique each time you run) reservations by choosing
    - A random day of the week

    - A random timeslot (i.e., there are 21 options from 4pm-9pm)
    - A random size (i.e., number of people) for that reservation which can be at most **40**
- For each attempted reservation, you should show (nicely on he terminal window) the number of people for the reservation, the day of the week and the start time of the reservation.
- If the reservation is able to be made, it should indicate that the reservation was accepted, otherwise it should indicate that it was denied.
- In order to accept a reservation … we need to estimate the length of stay for each reservation. Assume the following:
    - If the reservation is for 2 people, the estimate length of stay is 60 minutes.
    - If the reservation is between 3-7 people, the estimate is for 75 minutes.
    - If the reservation is between 8-11 people, the estimate is for 105 minutes.
    - Otherwise the estimate should be for 120 minutes.
- The program should accept the reservation if the restaurant can hold that many people for the duration of the estimated length of stay. For example, if a 5pm reservation for 4 people is made, you need to ensure that there are at most 76 people (i.e., restaurant capacity minus the reservation size) currently reserved for the following timeslots: 5:00pm, 5:15pm, 5:30pm, 5:45pm and 6:00pm. If any are over the value of 76, then the reservation should not be made, it should be denied.

- After each random reservation is made (or denied), the program should display a nicely-formatted table showing the current reservation counts for each timeslot. This displaying must be done by means of a function that takes the 2D array as a parameter. The table should look as show below on the left. Note that it indicates the number of people currently reserved for that timeslot. You can see that on Tuesday, a reservation for 12 people has been reserved starting at 5:30pm and they are estimated to be gone by 7:30pm. The smaller group of 2 people on Sunday at 5:00pm are estimated to only be there for 1 hour. The image on the right shows the results after quite a few reservations have been made … many of them overlapping.



	Sun	Mon	Tue	Wed	Thu	Fri	Sat
4:00pm							
4:15pm							
4:30pm							
4:45pm				3			
5:00pm	2			3			
5:15pm	2			3			
5:30pm	2		12	3			
5:45pm	2		12	3			
6:00pm			12				
6:15pm			12				
6:30pm			12				
6:45pm			12				
7:00pm			12				
7:15pm		16	12				
7:30pm		16					
7:45pm		16					
8:00pm		16					
8:15pm		16					
8:30pm		16					
8:45pm		16					
9:00pm		21					



	Sun	Mon	Tue	Wed	Thu	Fri	Sat
4:00pm			16	19	3	15	
4:15pm			25	19	3	21	
4:30pm			25	19	3	21	
4:45pm			25	22	15	40	
5:00pm	19		32	52	24	40	
5:15pm	19		41	58	21	40	5
5:30pm	19		53	58	21	34	5
5:45pm	47		53	58	21	37	16
6:00pm	45		28	36	21	27	16
6:15pm	45		21	44	21	32	16
6:30pm	45		21	38	32	32	11
6:45pm	45	28	21	27	30	18	11
7:00pm	28	39	12	8	30	15	11
7:15pm	28	68	12	8	30	10	11
7:30pm	17	68		8	41	5	
7:45pm		76	16	10	41	19	
8:00pm	14	76	16	10	41	20	16
8:15pm	26	76	16	23	30	20	16
8:30pm	26	76	16	23	46	20	16
8:45pm	26	37	16	23	45	32	16
9:00pm	26	42	16	31	45	34	16

IMPORTANT SUBMISSION INSTRUCTIONS:

Submit all of your **c source code** files as a single **tar** file containing:

1. A **Readme** text file containing
   - your name and studentNumber
   - a list of source files submitted
   - any specific instructions for compiling and/or running your code
2. All of your **.c source** files and all other files needed for testing/running your programs.
3. Any output files required, if there are any.

The code **MUST** compile and run on the course VM, which is **COMP2401-W20**.

- If your internet connection at home is down or does not work, we will not accept this as a reason for handing in an assignment late ... so make sure to submit the assignment WELL BEFORE it is due !

- You WILL lose marks on this assignment if any of your files are missing. So, make sure that you hand in the correct files and version of your assignment. **You will also lose marks if your code is not written neatly with proper indentation and containing a reasonable number of comments.** See course notes for examples of what is proper indentation, writing style and reasonable commenting).
