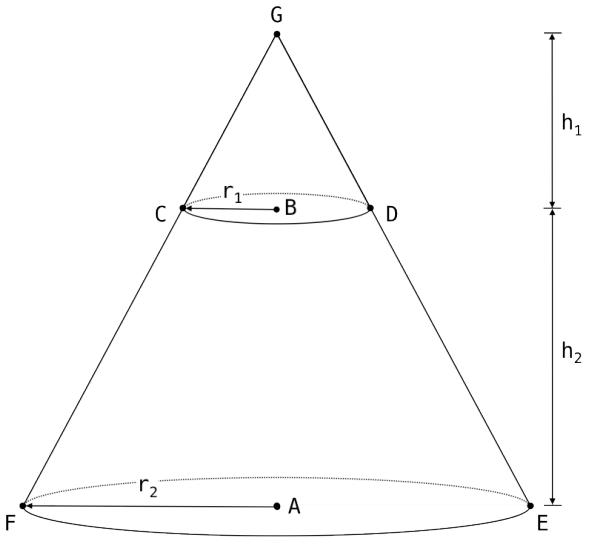
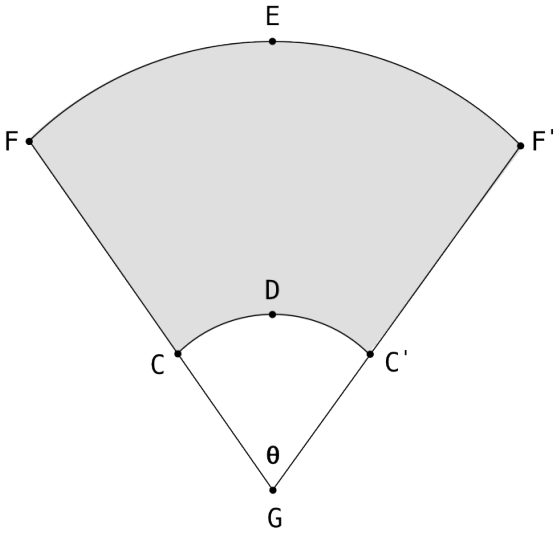
Allegheny International Manufacturing

Background

You are a new member of the Allegheny International Manufacturing (AIM) Information Technology (IT) Department. The company manufactures unique containers of various shapes and sizes from sheet metal. Historically cubes, rectangular prisms, and cylinders have been AIMs primary product lines. AIM sales representatives recently noticed demand for containers in the shape of a cone or frustum of a cone. The IT department has been tasked to develop software to support manufacturing of these new products

Right circular cones are familiar shapes. A frustum of a cone is basically a cone with a portion of its top end removed. The figure below left represents a cone annotated with the measurements required from a customer. If h_1 is zero, the shape is a cone; otherwise, the shape is a frustum of a cone. The figure below right is the pattern we need to cut from a piece of sheet metal to fabricate the shape. The shaded portion of the pattern is the final piece that will form a frustum of a cone. Of course the entire pattern would be shaded to form a cone.





Our initial challenge is to determine the key measurements necessary to create the pattern shown on the right from the dimensions shown on the shape's image to the left. Specifically, we must know the length of line segment $\overline{GC}$, the length of the line segment $\overline{GF}$, and the angle θ.

Next, we need to compute the total surface area of the finished shape. A cone shape may have an open or closed base. A frustum of a cone may have an open top, open base, or any combination thereof.

Calculations

Pattern	$GC = \sqrt{\left(\frac{r_1 h_2}{r_2 - r_1}\right)^2 + r_1^2}$
	$GF = \sqrt{\left(\frac{r_1 h_2}{r_2 - r_1}\right)^2 + r_1^2} + \sqrt{h_2^2 + (r_2 - r_1)^2}$
	$\theta = 180 \left(\frac{2r_2 - 2r_1}{GF - GC}\right)$

Lateral Surface Area, Right Circular Cone	$S = \pi r_2 \sqrt{r_2^2 + h_2^2}$

Lateral Surface Area, Frustum of Right Circular Cone	$S = \pi(r_1 + r_2)\sqrt{h_2^2 + (r_2 - r_1)^2}$

Base / Top Surface Area	$S = \pi r^2$ (where r is either r_1 or r_2)

Program Input

- The shape code (C for cone, F for frustum of a cone)
- The radius of the shape top, r_1 (minimum 0 for a cone, or 0.5(r_2); maximum 0.75(r_2))
- The radius of the shape base, r_2 (minimum 4 inches; maximum 20 inches)
- The height of the shape, h_2 (minimum 5 inches, and must be at least equal to r_2; maximum 25 inches)
- Flag to indicate if the top is opened or closed (Y indicates closed; N indicates open)
- Flag to indicate if the base is opened or closed (Y indicates closed; N indicates open)
- Color code
  - R (Red)
  - O (Orange)
  - Y (Yellow)
  - G (Green)
  - B (Blue)
  - I (Indigo)
  - V (Violet)

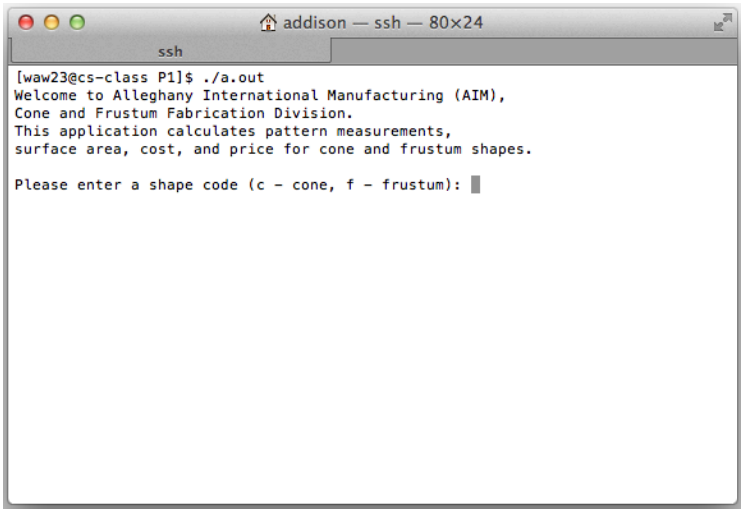
Note: For single character input either upper case or lower case shall be accepted as valid. Additionally, the full word represented by the character shall also be accepted. For example G, g, Green, or green would all be accepted as the entry for color while only the single character G would actually be stored. (**Note: To simplify the software, any word beginning with a letter corresponding to one of the color codes will be accepted.)**

Program Output

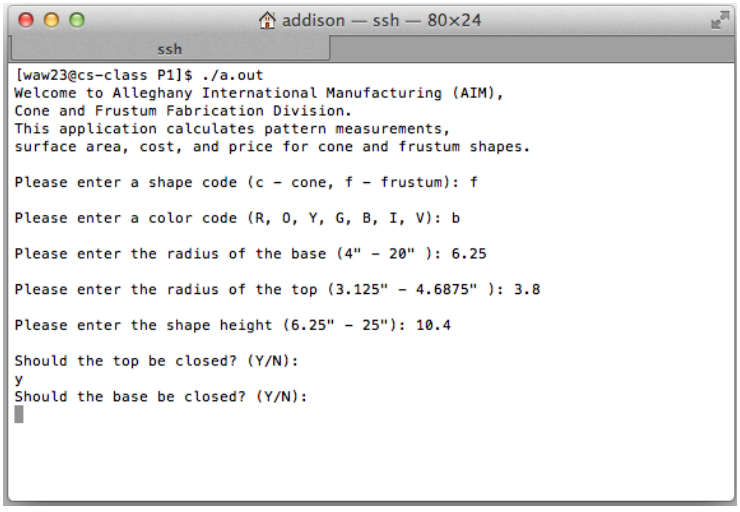
Program output should consist of a brief report that lists:

- The values that were input
- Message(s) that describe any input data in violation of rules listed above
- If all input data are valid:
    - The calculations for the pattern dimensions
    - The total surface area of the finished shape
- Cost of raw material based on $4.79/sq. ft. for sheet metal
- Retail price of the final manufactured shape, the retail price is the total cost of raw materials plus 26%

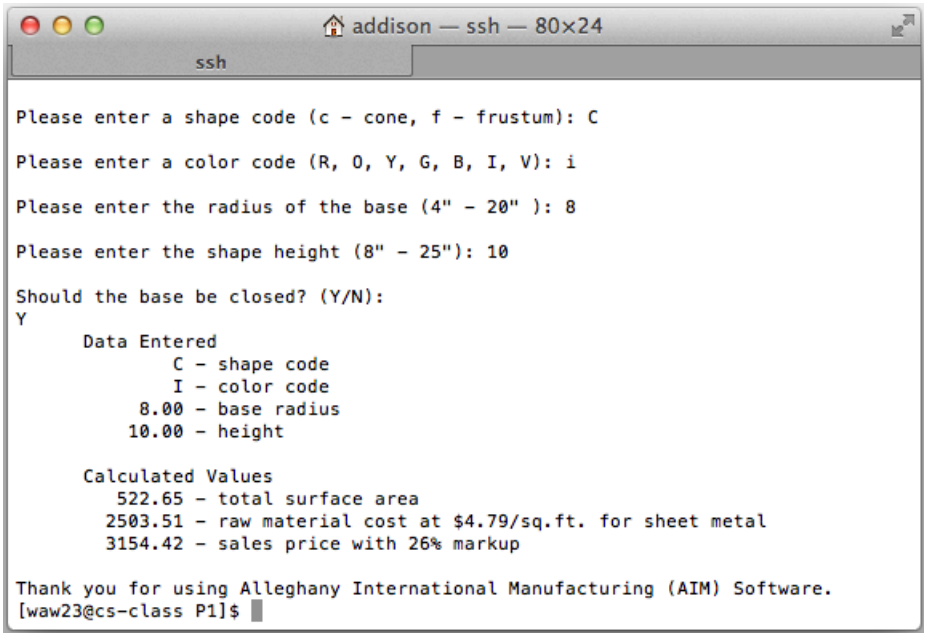
Shown below is output from several different executions of the program:



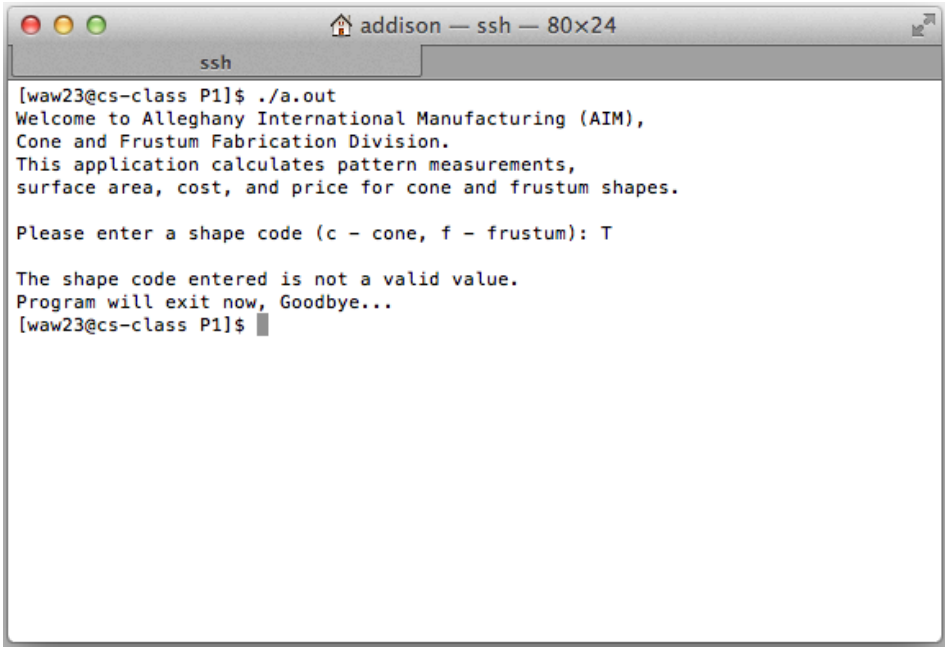
Your software shall have "smart prompts" that inform the user of acceptable entries:



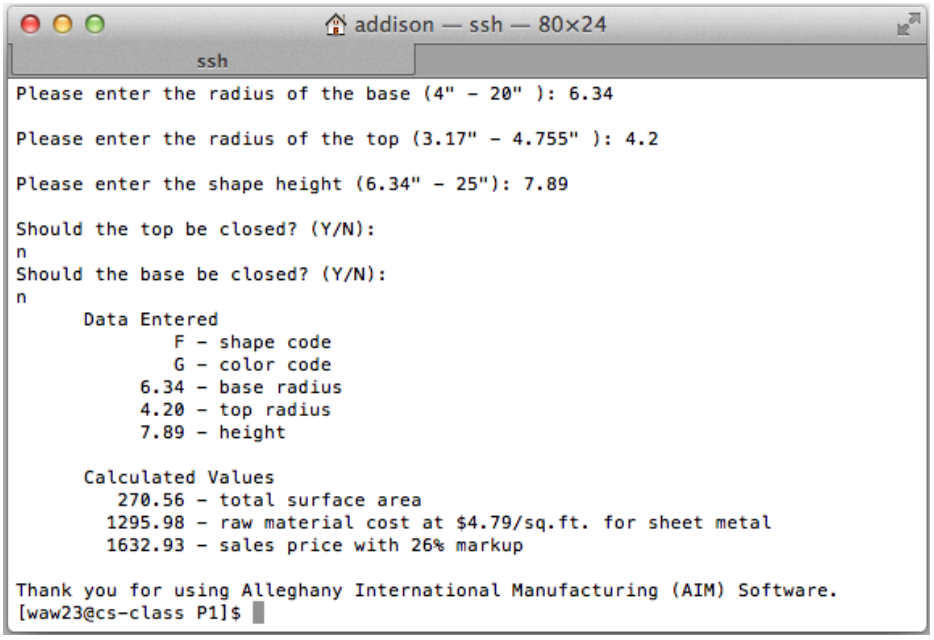
Your software shall accept upper case and lower case for character data entry. It shall also ignore any extraneous characters entered after a valid value is entered. As mentioned in the earlier note, any word beginning with a letter matching one of the color codes shall be accepted. For example if the user entered `Granite` for the color code, that would be accepted as meaning Green and the character `G` would be stored as the user's entry. The remaining characters are ignored. This is a bit lax with respect to data validation, but it is fine for our purposes.

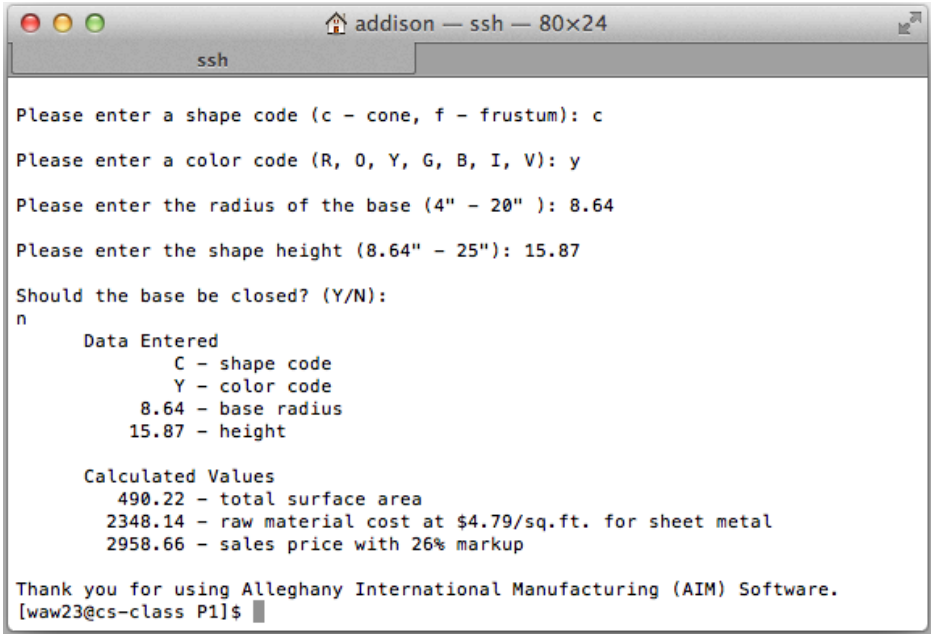


Your software shall display clear error messages for any invalid entries. If an invalid value is entered, the program should exit after displaying the error message.



Your software shall display neatly formatted, accurately calculated output.





Academic Integrity

This is an individual project and all work must be your own. Refer to the guidelines specified in the *Academic Honesty* section of this course syllabus or contact me if you have any questions.

Part A - Design Document

Include the following comments at the start of your design document file:

```
/*
 * <FileName>.<file extension>
 *
 *  COSC 051 Summer 2016
 *  Project #1
 *
 *  Due on: JUL 14, 2016
 *  Author: <your name>
 *
 *
 *  In accordance with the class policies and Georgetown's
 *  Honor Code, I certify that, with the exception of the
 *  class resources and those items noted below, I have neither
 *  given nor received any assistance on this project.
 *
 *  References not otherwise commented within the program source code.
 *  Note that you should not mention any help from the TAs, the professor,
 *  or any code taken from the class textbooks.
 */
```

For the first part of this project you must submit a pseudocode design document showing the algorithm(s) you plan to implement. Our pseudocode language for the project contains the following terms:

START
INPUT
OUTPUT
CALCULATE
IF *condition*, **THEN** *statement*
IF *condition*, **THEN** *statement*; **OTHERWISE**, *statement*
STOP

If you need to group multiple statements together, say in an **IF** statement, use

BEGIN
statement
...
statement
END

Part A - Submission Details

Post to Blackboard a .pdf file containing your design using the language described above. Locate the assignment `Project 1a` on Blackboard and attach/upload your file. Use the following file name for your file: `<netID>P1.pdf` (replace `<netID>` with your netID and remove the angle brackets). Due date for Part A is no later than end-of-day (11:59pm) on July 14th. Late submissions will be penalized 2.5% for each 15 minutes late. If you are over 10 hours late you may turn in the project to receive feedback but the grade will be zero. In general requests for extensions will not be considered. The value for Part A is 100 points. Your Part A grade is 35% of the overall Project #1 grade.

Part B - Program Source Code

Important: Your output and input should be very similar to that shown in the sample output. Please ask for the input in **exactly** the same order shown and only request the same items shown - do not ask for any other input. This will assist in grading your program. Some content must also be included in your program **exactly** as specified.

Include the following comments at the start of your source code file:

```
/*
 * <FileName>.<file extension>
 *
 *  COSC 051 Summer 2016
 *  Project #1
 *
 *  Due on: JUL 18, 2016
 *  Author: <your name>
 *
 *
 *  In accordance with the class policies and Georgetown's
 *  Honor Code, I certify that, with the exception of the
 *  class resources and those items noted below, I have neither
 *  given nor received any assistance on this project.
 *
 *  References not otherwise commented within the program source code.
 *  Note that you should not mention any help from the TAs, the professor,
 *  or any code taken from the class textbooks.
 */
```

These comments must appear **exactly** as shown above. The only difference will be values that you replace where there are "place holders" within angle brackets such as `<netID>` which should be replaced by your own `netID`. For example, I would replace `<netID>P1.cpp` with `waw23P1.cpp`.

Part B - Submission Details

Post to Blackboard a `.cpp` file containing your source code. Locate the assignment `Project 1b` on Blackboard and attach/upload your file. Do **not** post your executable file. You should ensure that your source file compiles on the server and that the executable file runs and produces the correct output. Use the following file name for your file: `<netID>P1.cpp`. Due date for Part B is no later than end-of-day (11:59pm) on July 18th. Late submissions will be penalized 2.5% for each 15 minutes late. If you are over 10 hours late you may turn in the project to receive feedback but the grade will be zero. In general requests for extensions will not be considered. The value for Part B is 100 points. Your Part B grade is 65% of the overall Project #1 grade.

Part A - Grading

	Grade Standards - Missing: 0%, Poor: up to 50%, Fair: up to 67%, Good: up to 82%, Excellent: up to 99%, Perfect: 100%		
	Detailed Rubric (Design)		
		100.00	**<-- TOTAL**
1	**Constants and variables (See Note 1)**	***15.00***	**<--sub total**
	good use of constants (be thorough)		
	good constant names		
	good variable names		
2	**User interface / data input**	***18.00***	**<--sub total**
	outputs a brief greeting message		
	outputs prompt for shape code		
	outputs prompt for color code		
	outputs prompt for base radius		
	outputs prompt for top radius (if shape is frustum)		
	outputs prompt for height		
	outputs prompt for top closed or open (if shape is frustum)		
	outputs prompt for base closed or open		
	for character input, both uppercase and lowercase are accepted as valid		
3	**Data validation algorithms (See Note 2)**	***15.00***	**<--sub total**
	all input data are validated to ensure they are valid and/or within limits		
	prompts for data input are in reasonable order, test for errors and exit as soon as possible (don't make the user keep entering data if there has already been a fatal error)		
	if any input data fail validation error message(s) are displayed		
	processing terminates if any data fail validation, "abnormal" exits are allowed for Project #1, but will eventually be prohibited		
4	**Calculation algorithms**	**40.00**	**<--sub total**
	total surface area of finished shape is accurately calculated		
	cost of raw material (sheet metal) is accurately calculated		
	sales price (raw material cost plus mark-up percentage) is accruately calculated		
5	**Output**	***12.00***	**<--sub total**
	outputs values that user entered		
	outputs calculated values		
	Note 1: For the Design Part, you will not explicitly specify data types, but you will be using names for "things". These named things will become constants or variables in your code. Make the names clear and understandable, and consider using the same names that you will declare in the coding part of the project.		
	Note 2: Advanced error handling is not required for this project. However, you must test data entered to ensure that values are "reasonable" (refer to the project description for specific validation rules and what constitutes reasonable values). For data that do not meet this criteria, you should output a message to the user explaining the nature of the issue and that the program will exit.		
	Common Deductions (Design)		
	Filename does not follow conventions specified	-20.00	
	Deviates from the specified pseudocode terms (-20 for each occurance up to the max deduction listed at the right)	-100.00	
	Uses any C++ specific code (-10 for each occurance up to the max deduction listed at the right)	-100.00	
	Required comments and honor statement not included at start of file exactly as specified	-40.00	
	Late penalty for each 15 minutes late	-2.50	

Part B - Grading

	Grade Standards - Missing: 0%, Poor: up to 50%, Fair: up to 67%, Good: up to 82%, Excellent: up to 99%, Perfect: 100%		
	Detailed Rubric (Code)	**100.00**	**<-- TOTAL**
1	**Code Quality and Formatting**	***15.00***	**<--sub total**
	proper indentation		
	good variable and constant names		
	good use of constants (no "magic numbers" in calculations)		
	good use of comments		
	good use of vertical white space to separate code		
	good use of horizontal white space to improve readability		
	line length less than 100 characters		
2	**User interface / data input**	***15.00***	**<--sub total**
	outputs a brief greeting message		
	outputs prompt for shape code		
	outputs prompt for color code		
	outputs prompt for base radius		
	outputs prompt for top radiius (if shape is frustum)		
	outputs prompt for height		
	outputs prompt for top closed or open (if shape is frustum)		
	outputs prompt for base opened or closed		
	values entered by user are input into named variables of appropriate data type		
	any extraneous characters entered after a valid entry are ignored		
	error messages are clear and descriptive		
	for character input, both uppercase and lowercase are accepted as valid		
3	**Achieves Program Intent**	***15.00***	**<--sub total**
	Submission correctly, and in good faith, implements code to achieve the intent and requirements of the program as specified in the project description and clarified during in-class discussions and forum posts.		
4	**Data validation algorithms**	***15.00***	**<--sub total**
	all input data are validated to ensure they are valid and/or within limits		
	prompts for data input are in reasonable order, test for errors and exit as soon as possible (don't make the user keep entering data if there has already been a fatal error)		
	if any input data fail validation error message(s) are displayed		
	processing terminates if any data fail validation, "abnormal" exits are allowed for Project #1, but will eventually be prohibited		
5	**Calculation algorithms**	**20.00**	**<--sub total**
	total surface area of finished shape is accurately calculated		
	cost of raw material (sheet metal) is accurately calculated		
	sales price (raw material cost plus mark-up percentage) is accurately calculated		
6	**Output**	**20.00**	**<--sub total**
	outputs values that the user entered		
	outputs calculated values		
	output is neatly arranged on screen and is consistent with the output shown in the example program		
	Common Deductions (Code)		
	Program does not compile ON THE CLASS SERVER (deduction varies depending on how bad, value listed is max)	-45.00	
	Program compiles but has warnings ON CLASS SERVER (deduction varies depending on how bad, value listed is max)	-30.00	
	Program crashes during execution ON CLASS SERVER (deduction varies depending on how bad, value listed is max)	-30.00	
	Code uses any global variables	-40.00	
	Filename does not follow conventions specified	-20.00	
	Required comments and honor statement not included at start of file exactly as specified	-30.00	
	Late penalty for each 15 minutes late	-2.50	

(This page intentionally left blank)