Project #1

Due Sep 4 by 11:59pm **Points** 100 **Available** Aug 21 at 12am - Sep 5 at 2am

Task

A friend wants to start a small business for floor cleaning equipment rental, similar to what Home Depot offers (https://www.homedepot.com/c/floor_cleaning_equipment_rental).

Its rental pricing is offered with 4-hours, per day, and per week options.

A rental of any number of hours less than 4-hours will be charged with the 4-hour rate. A rental of any period of less than a day will be capped (maximum charge) with the per day rate. For example, 8-hours rental of carpet blower will cost $10. A rental of any period of less than a week will be capped (maximum charge) with the per week rate.

Write a program that calculates and prints the charges for a floor cleaning equipment rental. The user should enter the selection, enter the days and hours rented for a customer, and print the charge.

	Equipment	4-hours	Per day	Per week
1	Carpet blower	$7	$10	$40
2	Carpet cleaner	$27	$39	$156
3	Carpet extractor with heater	$61	$87	$348
4	Hard flooring cleaner	$59	$84	$336

Requirements

1. The user enters the equipment selection, enter the number of days, and number of hours. The program calculates and prints the charge. Follow the format of the examples below.

2. Your program should validate the equipment selection. If the selection is not in the range of 1 to 4, print the message *"Invalid selection. Select from 1 to 4."* and exit the program.

3. Your program should validate hours (>=0 and <24 hours). If the selection is not in the range, print the message *"Invalid hours."* and exit the program.

4. Hint: use / and % operators might be useful in the calculations

Examples (your program must follow this format precisely)

Example #1:

Please select from four equipment: 1, 2, 3, and 4

Enter equipment selection: 6

Invalid selection. Select from 1 to 4.

Example #2:

Please select from four equipment: 1, 2, 3, and 4

Enter equipment selection: 3

Enter days: 0

Enter hours: 30

Invalid hours.

Example #3:

Please select from four equipment: 1, 2, 3, and 4

Enter equipment selection: 1

Enter days: 1

Enter hours: 7

Charge($): 20

Example #4:

Please select from four equipment: 1, 2, 3, and 4

Enter equipment selection: 2

Enter days: 10

Enter hours: 4

Charge($): 300

Example #5:

Please select from four equipment: 1, 2, 3, and 4

Enter equipment selection: 3

Enter days: 26

Enter hours: 8

Charge($): 1392

Submission instructions

1. Develop and test your program on the student cluster
   A. To compile your program, run: **gcc -std=c99 -Wall your_program.c**
   B. To execute your program, run: **./a.out**
2. Name your program **project1_rental.c**
   A. To rename your program, run: **mv your_program.c project1_rental.c**
3. Test your program with the shell script on Unix: try_project1_rental
   A. upload the script zip file to the student cluster in the same directory as your program.
   B. run: **$ unzip try_project1_rental.zip**
   C. move your program to the same folder as try_project1_rental file
   D. run: **chmod +x try_project1_rental**
   E. run: **./try_project1_rental**
4. Download the program from student cluster and submit it on **Canvas->Gradescope**. Make sure you submit a file with the correct name!
5. You can submit your program as many times as needed before the due date. Gradescope will indicate the test cases with incorrect output, if any exists.
6. Please note that GradeScope is used for testing some of the functionalities. It's part of the grading process. The grade you received on GradeScope only reflects the results of your program against the test cases, it is not your final project grade. Projects will be manually graded after running the test cases on GradeScope.

Grading

Total points: 100

- A program that does not compile will result in a zero.
- Runtime error and compilation warning 5%

  3 points off, if a warning is present.

  5 points off, if multiple warnings are present.

- Commenting and style 15%

  2 points off for not putting name and description at the beginning

  3 to 8 points off if the code didn't have clarifying comments.

  1 to 5 points off (depending on how much indentation is off) if the program is not indented properly.

- Functionality 80%

  For project 1, if all test cases were passed, full credit for functionality.

  - test case 1 is incorrect – 5 points off
  - test case 2 is incorrect - 5 points off
  - test case 3 is incorrect – 10 points off
  - test case 4 is incorrect – 10 points off
  - test case 5 is incorrect – 10 points off
  - test case 6 is incorrect – 10 points off
  - test case 7 is incorrect – 10 points off

Programming Style Guidelines

The major purpose of programming style guidelines is to make programs easy to read and understand. Good programming style helps make it possible for a person knowledgeable in the application area to quickly read a program and understand how it works.

- Your program should begin with a comment that briefly summarizes what it does. This comment should also include your name.
- In most cases, a function should have a brief comment above its definition describing what it does. Other than that, comments should be written only needed in order for a reader to understand what is happening.
- Variable names and function names should be sufficiently descriptive that a knowledgeable reader can easily understand what the variable means and what the function does. If this is not possible, comments should be added to make the meaning clear.
- Use consistent indentation to emphasize block structure.
- Full line comments inside function bodies should conform to the indentation of the code where they appear.
- Macro definitions (**#define**) should be used for defining symbolic names for numeric constants. For example: **#define PI 3.141592**
- Use names of moderate length for variables. Most names should be between 2 and 12 letters long.
- Use underscores to make compound names easier to read: **tot_vol** and **total_volumn** are clearer than **totalvolumn**.