

# CS 161 Lab Session: Functional Decomposition Problem

## Functional Decomposition

Functional decomposition is the process of breaking down a large problem into small sub-problems, and then breaking down the sub-problems, and repeating this process until each sub-problem is described in such simple and clear terms that it is obvious how to express it in program code. The sub-problems become functions, and the initial outline describes the code for the main function.

This process makes programming easier. First, we don't try tackling a large complicated problem all at once—we break into “bite-size” pieces. Second, it helps us make sure that we are getting the steps in the right order by describing them in familiar language. Third, the process identifies the parts of the solution that make logical functions. Finally, we can ignore the details of program syntax while we solve the problem, and when we reach the point of writing code we can focus on syntax without worrying too much about anything else.

Your task today will involve applying this process to a problem presented at varying levels of complexity, and discuss your results with other members of the class after each step.

## Scenario

This program prepares estimates for the Acme Custom Window Company. As with other examples used in this class, it will be somewhat contrived.

The user enters information for a custom window order, and the program prints an estimate showing the the frame cost, the glass cost, a 1/3 down payment due immediately, the balance due on delivery, and the total cost.

## Tasks

### *Step One*

We'll approach this starting with the simplest possible case, and then add more details one step at a time.

For the first stage, identify the information that the user must input, the values that will be output, and then identify any variables (input, output, or intermediate) that the program will need. In this simple version, frames cost \$1.50 per lineal foot and glazing costs \$1.15 per square foot.

Next, create an outline describing the general steps necessary to get from start to finish. Don't worry about specifics, such as computations. One way to outline this basic task would be:

- get window dimensions
- determine glass cost
- determine frame cost
- create estimate

This actually is a decent outline for the top level. However, we aren't finished. Outline the three sub-tasks of “determine glass cost,” “determine frame cost,” and “create estimate.” Work with someone else if you must, but don't let someone else do your work.

Allow 8 minutes for this, and then take a few minutes to compare notes with some other people.

### *Step Two*

Now we'll make this a bit more realistic. Frames may be aluminum, steel, or vinyl. The cost of a frame depends on the material and the amount of the material needed. Aluminum is \$2.00 per lineal foot, steel is \$1.35 per lineal foot, and vinyl is \$1.85 per lineal foot.

Glass may be single glazed or double glazed. The cost depends on the glazing and the area. Especially large windows (over 24 square feet) cost extra because thicker glass is needed. Single glazed windows are \$1.15 per square foot; double glazed windows are \$2.75 per square foot, and windows over 24 square feet are an extra 25%.

Modify or re-write the outline you created for step one to account for these details. Allow 10 minutes for this activity, and then compare notes again.

Did you have to change the top-level outline? Did you need to add functions, or merely alter the functions you had?

### *Step Three*

It is unusual for people to buy only one window at a time. Modify your outline to allow multiple windows. If the estimate is to report the cost of each window, some changes will have to occur other than just adding a while or for loop.

If you think you have a solution that works, compare notes with some others and see if you all agree.

### *Step Four*

Cost estimating programs often include the estimate date and delivery date. Since humans often make mistakes, it's a good idea to do something to assure that the entered dates are valid. Obvious checks include making sure that the month is in the range of 1–12, and that the day is in the correct range for the month. What makes a valid year is dependent on the context.

Design (in plain English, using functional decomposition, the necessary solution(s) to get valid estimate and delivery dates.

Do we need a different function to validate the two dates?

How much change is necessary to the outline you already had?

Once again, compare notes when you are finished with this. We'll talk about this more tomorrow.