CS 102
Introduction to Programming Using C++

Chapter 5-Part 3

Stepwise Refinement

Stepwise Refinement
A Tool for Program Development

- Suppose you have to write a large or complicated program
- One way to do it is to create an outline
- Then you fill in the steps
  - This is called refining the outline
- This sounds easy enough

Stepwise Refinement
A Tool for Program Development

- Yes, but what if those steps are complicated?
- You repeat the process
  - You refine those steps
- You keep doing this over and over until you have written the entire program
- Of course, this can be applied to a function too

Top-Down Design

- This is also called top-down design
- You start at the "top" of the project
  - This means you see the big picture view
- You then "look down" into the details
- This is a very common method of program design

Example 1: Getting Coffee (The Textbook's Example)

- You wake up in the morning
- You decide you want some coffee
- What should you do?

First Solution

- One solution is get coffee from someone else in your house
  - Perhaps someone else has already made coffee
- Another solution is to make instant coffee
- Yet another solution is to brew a cup of coffee
- Still another solution is to go to Starbucks

The Steps So Far

- if someone else made coffee
  - Get a cup from that person
- else if you have some money left over
  - Go to Starbucks
- else if you have instant coffee
  - Make instant coffee
- else
  - Brew a cup of coffee

Modular Design

- We could (should!) consider making each step a function
- The code is
- if someone else made coffee
  - borrow_coffee()
- else if you have some money left over
  - go_to_Starbucks()
- else if you have instant coffee
  - make_instant_coffee()
- else
  - brew_coffee()

The Comments Preceding a Function

- A good first step in the process is the comments we put at the top of a function
- I talked about JavaDoc earlier
- Comments about what code does (especially functions) should be written as you write the code
  - Why is that a good idea?
- This should guide the function's design

Focusing on the Steps

- The book's idea is that the first two functions are easy
- They don't refine them further

Unexplained Steps 1
Make Instant Coffee

- The first function to focus on is making instant coffee
- We look at the steps of that function
  - get_hot_water()
    - This is a function too
  - Add instant coffee
  - Stir

Unexplained Steps 2
Get Hot Water

- Now we refine the get_hot_water() function
- The steps are
  - If you have a microwave, you can heat up a cup of water
  - If not, you can boil a cup of water in a kettle

Unexplained Steps 3
Brewing Coffee

- The brew_coffee() function
  - Find a filter
  - Put the filter into the coffee maker
  - Add coffee grounds to filter
    - add_coffee_grounds()
  - Add water to coffee maker
  - Turn on coffee maker

Unexplained Steps 4
Adding Coffee Grounds to Filter

- What if you don't have coffee grounds?
- Then, if you have coffee beans, you grind them
- Otherwise you can go to the store and buy some
  - This might become another function

That's It!

- The text has another example of the stepwise refinement process
  - It talks about converting a number into its (string) name
    - For example, convert 13,472 to thirteen thousand, four hundred seventy-two
- You should check this example too

Another Example

- A problem: You have the choice of buying two cars
  - One is more fuel efficient than the other, but also more expensive
  - You know the price and fuel efficiency (in miles per gallon, mpg) of both cars
- Which car is the better deal?

Assumptions

- For this program, we will assume
  - The car will be used for ten years
  - The cost of a gallon of gas is $4
  - The car will be driven 15,000 miles per year
  - The car will be paid for in cash so we don't have to worry about financing costs
- These assumptions should turn into constants when writing the program

The Car Example: Understanding the task

- How should the program work?
  - It should determine which car is the better buy based on assumptions and inputs
- What should we have for input?
  - Car 1: Purchase price and fuel efficiency
  - Car 2: Purchase price and fuel efficiency
- What should we have for output?
  - Which car is the better buy?

Breaking the Problem into Smaller Tasks

- What will we do for each car?
  - We will calculate the total cost
- The total cost for a car is

  purchase price + operating cost
- Because of assumptions, the operating cost depends only on the cost of driving the car for ten years

The Functions So Far

- intialize_constants()
- read_input(car number)
- calculate_cost(car number)

Calculating the Operating Cost

- The total operating cost for ten years is

  10 x annual fuel cost

- The annual fuel cost is

  price per gallon x annual gallons of fuel used

- The annual fuel used is

  annual miles driven / fuel efficiency in mpg

Choices

- At this point we have choices,
- Some people like to keep the cost broken up into pieces
- The pseudocode would look like

Total operating cost = 10 x annual fuel cost

Annual fuel cost = price per gallon x annual fuel consumed

The annual fuel consumed = annual miles driven / fuel efficiency in mpg

Other Options

- Another option would be to do this in one big calculation
  - Operating cost for one year =

    10 x price per gallon x annual miles driven / fuel efficiency in mpg

- Which method you choose is up to you

Paying Attention to Details

- If you use the first method, you need to rearrange the steps
  - Intermediate values must be computed before they are used

Filling in the Details

- Rearrange the calculations if necessary

```
if total cost1 < total cost2
    choose_car (car_1)
otherwise
    choose_car(car_2)
```

Test Your Code by Working a Problem

- Choose sample values
  - Car 1: $25,000, 50 miles/gallon
  - Car 2: $20,000, 30 miles/gallon
- Go through the pseudocode to verify it
- Testing should always be done like this:
  - You know the expected answer or behavior that the program should produce
  - You verify that the program does what it should do

Questions?

- Are there any questions?