Software Design

CSCE 110

Influenced by material developed by James Tam & Jennifer Welch
Software Development

• Activities in the lifetime of a piece of software
  • _____________________________________
  • _____________________________________
  • _____________________________________
  • _____________________________________
  • _____________________________________
  • _____________________________________

• The phases are ________________________________.
• There _________________________________ between the different activities.
• Often you have ______________________________ of a previous phase.
• Different models and approaches
  • __________________
  • __________________
  • __________________
  • __________________
  • __________________
• Identify the _______________ and what the system ______ (and ______________) do.
  • __________ and __________ requirements
• The resulting set of ________ describe the __________
  • (e.g., data must not be available to unauthorized people).
• Convert into ____________________________________
  • (e.g., 10 character passwords).
Software ________________

• Break system into _____________ ________, or __________.
• __________________________________ is key to the ability to create large systems.
• Write the __________. (most of what we do in this course)
• Test each ________________.
• Test the _______________________
  (_______________ testing).
• Make sure _______________________________ is tested.
• ______________ the software
• ________________
  • _______ bugs
  • Modify ___________________________ requirements
Software Development

• Activities in the lifetime of a piece of software
  • __________________ ___________________
  • __________________ ___________________
  • __________________ ___________________
  • __________________ ___________________
  • __________________ ___________________

What you should do in this class.

What many of you will probably end up doing that will be reflected in your grades.

J. Michael Moore
Why Use Modular Decomposition

Benefits of ____________________________
(i.e. breaking things into modules):

• __________________________ - concentrate on one task at a time
• Saves ______________ if the same (or similar) code needs to be
  ________________.
• Helps programmer see __________________ aspects of the code
  so the solution is easier to __________________.
• Easier to _________ the program
• Easier to _________ (if modules are _______________, then _________
  in one module should not ______________ other modules)

Drawback

• ______________ – understanding and setting up inter-module
  communication may appear daunting at first
• ______________ the program may appear harder as execution
  appears to “_________” around between modules.

J. Michael Moore
Start by identifying _______ ________ at the ________ level and then ___________ them to solve the main problem.

• *Disadvantages:*
  — how do you know the required _______________ initially?
  — Trying to address _______________ of a large problem all at once may prove to be ____________.

• *Advantage:*
  — favors _______________ existing __________.
design

• Start with ______ level idea of solution, and ______________ ________ it.
• Favors ___________________________ organization, with __________ inside ____________.
  • Advantage:
    —________ and __________ the problem
  • Disadvantage:
    —design _____________ made at ________ levels are _____________ for ________ levels
Top Down Design

1. Outline the ________ parts (____________)

My autobiography

Chapter 1: The humble beginnings

Chapter 2: My rise to greatness

... 

2. _____________ the solution for each part

Chapter 1: The humble beginnings

It all started seven and one score years ago with a log-shaped work station…
Top-Down Approach: 
Breaking Down A Large Problem

- General approach
  - Approach to part of problem
    - Specific steps of the solution
  - Approach to part of problem
    - Specific steps of the solution
  - Approach to part of problem
    - Specific steps of the solution

Figure extracted from Computer Science Illuminated by Dale N. and Lewis J.
Modules

• Information that is ______ between ______ module A and _____ module B with __________________ and __________________.

• Some modules we have already seen
  • writeln
  • readln
  • trunc
  • sqrt

• Types of modules that can be implemented in Pascal
  • ______________
  • ______________

• Places you’ll see ______________ and ______________ referenced
  • ___________: This is where the ________ for the ______________ are ________
  • ________: This is where the __________ is ______________ to run, i.e. where ______________ of the module ______________
Modules

- Modules can ___________ other modules:
  - Modules A includes a ___________ to _________ module B
  - When ___________ of A reaches the ___________, ___________ is ___________ to the ________________ of B
  - The _______________ of B are _______________ until the ___________ of B
  - _______________ is _______________ back to A, just _______________ the _____________ to B
Modules

- What should be made into a module?
  - A relatively ________________ piece of code that accomplishes a ____________________ task.
Design: Finding ____________ Modules

- The process of going from a ___________ _____________ (________ that ___________ what a program is ____________ to do) to writing a program that ___________ those requirements cannot be ________________ in just a series of steps that fit ______ _________________.
- The first step is to look at _________ either directly in the ___________ _____________ (indicates what _________ the program should entail) or those which can be ________ from the ___________ _____________ .
- Each _________ may be ________________ as a module but _________ _________ may have to be ________________ further into several additional modules (i.e. sub-modules).
Example Problem

• Design a program that will perform a simple interest calculation.
• The program should prompt the user for the appropriate values, perform the calculation and display the values onscreen.
Top Down Approach:
Breaking A Programming Problem Down Into Parts (Modules)
Making Change

• (Paraphrased from the book “Pascal: An introduction to the Art and Science of Programming” by Walter J. Savitch.

Problem statement:
Design a program to make change. Given an amount of money, the program will indicate how many quarters, dimes and pennies are needed. The cashier is able to determine the change needed for values of a dollar and above.

Actions that may be needed:
• Action 1: Prompting for the amount of money
• Action 2: Computing the combination of coins needed to equal this amount
  • Sub-action 2A: Compute the number of quarters to be given out.
  • Sub-action 2B: Compute the number of dimes to be given out
  • Sub-action 2C: Compute the number of pennies to be given out.
• Action 3: Output: Display the number of coins needed
Structure Diagram Of The Modules

Change program (main)

[Diagram showing the structure of the modules with arrows indicating flow and connections between the modules.]
Structure Diagram Of The Modules

Change program (main)

...