Introduction to Computing

Lectured by: Dr. Pham Tran Vu

t.v.pham@cse.hcmut.edu.vn

Programming

- Programming languages
- **Program design, testing, debugging and documenting**
- Data structures

Program Design

- Computer programs need to be error free and robust
- Programs need to deal with not only correct input data but also non-ideal data which is subject to certain level of errors
- Computer design and production is a very skilled activity

Problem Solving

- Understand the problem
- Design a solution
- Test the solution
- Document the solution

Understand the Problem

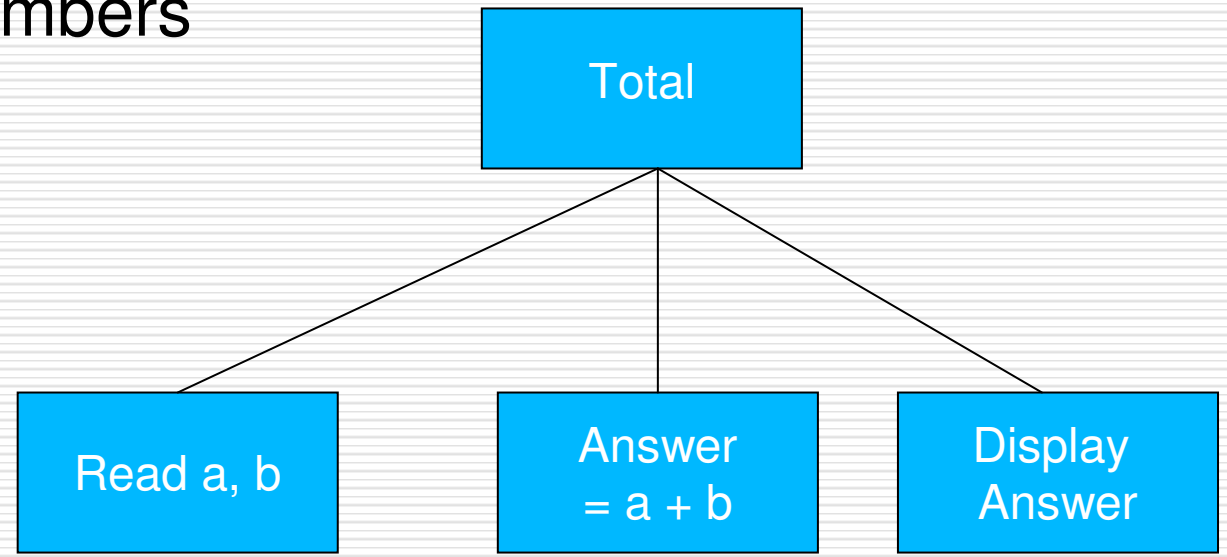
- Before a solution can be design, it is necessary to thorough understand the problem
- E.g.
  - Write a program to read in a date and convert it to the number of days from the start of the calendar year

Design a Solution

- Top-down stepwise refinement
  - Design the main module and the reduce it to smaller, simpler and more manageable components
- Tools:
  - Structure chart
  - Pseudocode

Structure Chart

- To provide a graphical representation of logical structure of a program
- E.g.: a program to calculate the sum of two numbers



Pseudocode

- Code-like instructions to help program coding and testing
- Usually independent from programming languages
- E.g.: a program to calculate the sum of two numbers

1. *Read a, b*
2. *Answer = a + b*
3. *Display answer*

Data Table

- Define the purpose and type of variables used in the solution

Name	**Description**	**Type**
Answer	Holds the sum of the two numbers	Real number
a	First number entered	Real number
b	Second number entered	Real number

Test the Solution

- Use test data to step through the solution to check if the solution is correctly design

Document the Solution

- Documentation is important
- Design document needs to include
  - The problem statement
  - The top-level program design
  - The final detail program design
  - The data table

Structured Programming

- Most program design methodologies are based on structured programming
- Basic principles
  - Restricted use of control structures
  - Modularity
  - Top-down, stepwise refinement
  - Clear program format
  - Comments
  - Simplicity

Example

- Design a program that reads in student examination marks. Each mar is to be displayed as a grade as follows:

Mark	**Grade**
80 or over	Distinction
60 or over	Merit
40 or over	Pass
Less than 40	Fail

Program Testing and Debugging

- Errors always present when someone write a piece of code
- Two types of error:
  - Syntax errors
  - Logic errors
- Debugging is the process of finding errors or bugs
- Testing is the process of checking if the program is working as it is designed to do

Debugging

- Detecting syntax errors
  - Most of the developing environments have tool to check for syntax errors
- Detecting logic errors
  - This is a more challenging task
  - Approaches:
    - Talking with other about your solution
    - Using debugging code (print out variables' values)
    - Using debugging facilities

Testing

- To check for less obvious logic errors
- Test data should be carefully chosen so that
  - Every statement is executed at least once
  - The effectiveness of every section of coding devoted to detecting invalid input is verified
  - Every route in the program is tried at least once
  - The accuracy of the processing is verified
  - The program operates to according to its original design specification

Test Data

- Normal data
  - Most general data the program is designed to handle
- Extreme values
  - Boundary values of variables and inputs
- Exceptional data
  - Unexpected data

Black Box Testing

- To test functions of program in terms of inputs and outputs
- The whole function being tested is considered as a black box
- Usually used for boundary value analysis

White Box Testing (1)

- The program code is thorough tested
- White box testing focus
  - Covering multiple conditions in a program
    - Test data should cover all the possible conditions in a program

White Box Testing (2)

- Covering loops within program
  - Test data should cover the following
    - Going the loop at a minimum number of times
    - Executing the loop once
    - Executing the loop maximum number of times
    - Executing the loop one less than the maximum number of times
    - Executing the loop several times

Validation

- One the programmer is confident that the program has sufficient testing and it will behave according to the specification without error, the program is then transferred to expected users for validation before final release
- If the validation is failed, the program needs to be modified and re-tested

Program Documentation

- Design of a program needs to be carefully documented so that other programmers can follow the documentation to build or modify the program
- Document checklist
  - Identification
  - General specification
  - User information
  - Program specification

Identification

- Title of program
- Short statement of its function
- Author
- Date written
- Language used and its version
- Hardware requirements

General Specification

- Main actions of the program
- File specifications
- Restrictions and limitations
- Equations used or text to explain complex procedures/techniques

Program Specification

- Structure charts, flowcharts, decision tables
- Annotated listing
- Testing procedure and test data

User Guide

- Installation instructions
- Detailed explanation of the program operations
- Tutorial
- Screen shot
- Trouble shooting guide