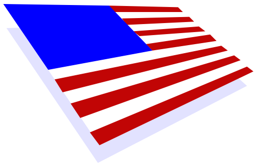
PowerPoint foils for lecture #5

KJC001 (9/23/99)

Lecture #5 for *Computer Tools for Engineers*



College of Engineering

University of South Florida

KJC002

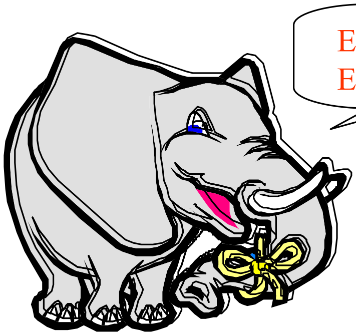
Today’s agenda:

- Miscellaneous and review from last lecture
- What’s under the hood (review and conclusion)
  - Components of a computer
  - Overview of client/server
  - How memory works
  - Machine language to assembly language to high-level language
- Design methods
  - The four steps
  - Flow charting
  - Divide-and-conquer
  - Successive refinement
  - Phases of a programming project
- Review and strategy for Exam #1

College of Engineering

KJC003

Miscellaneous:



Excel quiz is this week and Exam #1 is next week.

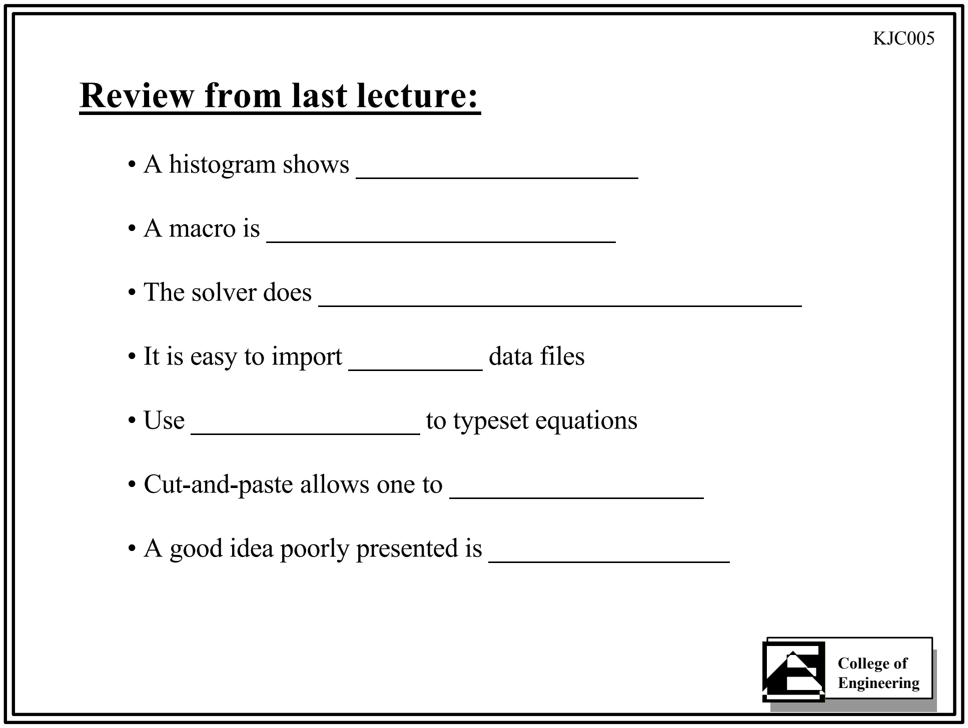
College of Engineering

KJC004

Miscellaneous: (continued)

- Grading errors...
  - If there is a grading error on your quiz or exam, ***we want to fix it!***
  - Please see me or a TA if too many, ***or too few***, points were deducted for a problem
  - You need to do this within ***one week*** of receving your quiz back

College of Engineering

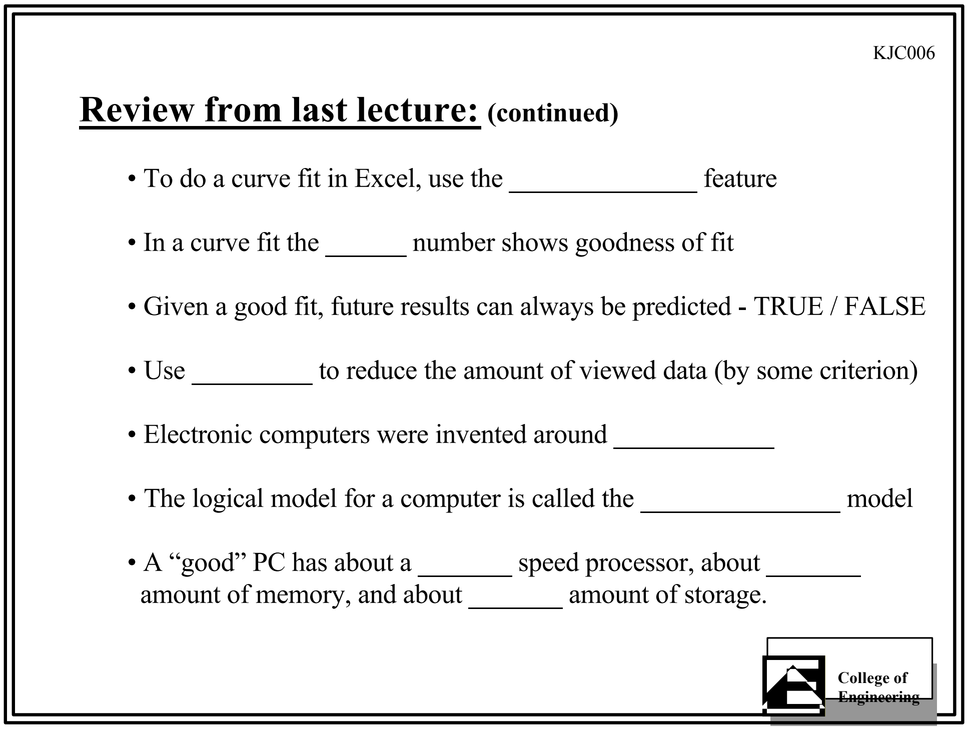


KJC005

Review from last lecture:

- A histogram shows _____________________
- A macro is __________________________
- The solver does ____________________________________
- It is easy to import __________ data files
- Use _________________ to typeset equations
- Cut-and-paste allows one to ___________________
- A good idea poorly presented is __________________

College of Engineering

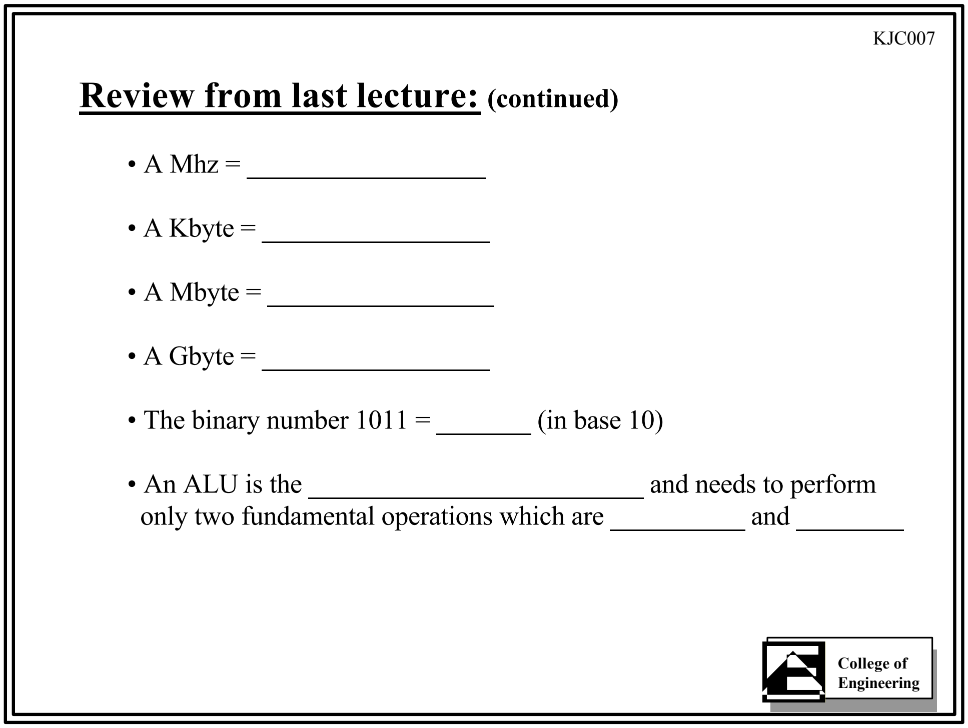


KJC006

Review from last lecture: (continued)

- To do a curve fit in Excel, use the ______________ feature
- In a curve fit the ______ number shows goodness of fit
- Given a good fit, future results can always be predicted - TRUE / FALSE
- Use _________ to reduce the amount of viewed data (by some criterion)
- Electronic computers were invented around ____________
- The logical model for a computer is called the _______________ model
- A “good” PC has about a _______ speed processor, about _______ amount of memory, and about _______ amount of storage.

College of Engineering

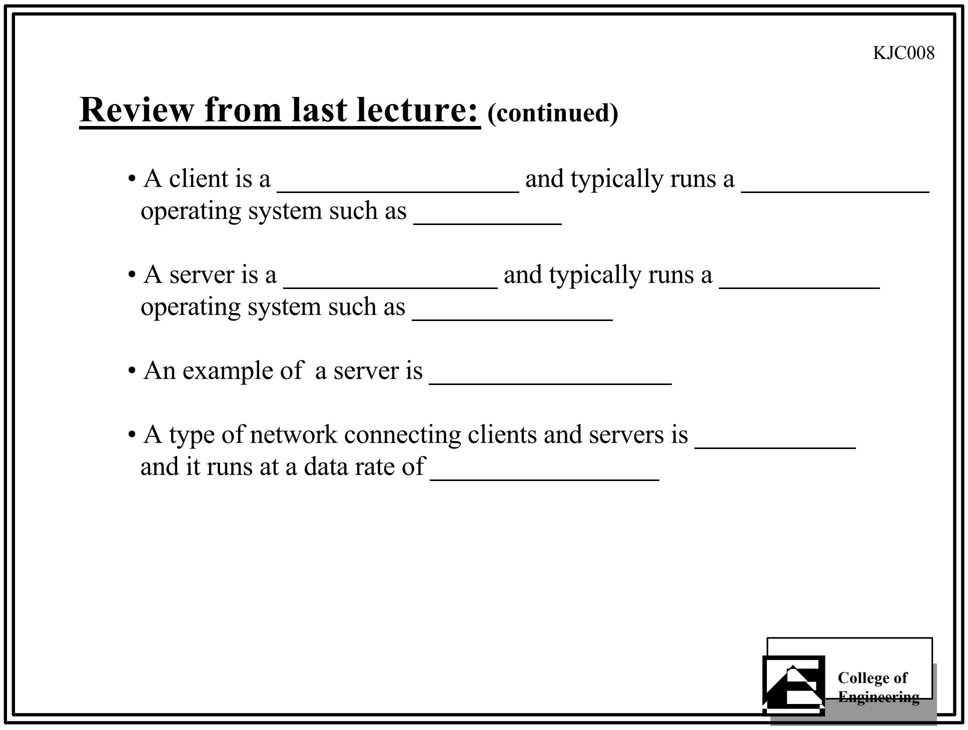


KJC007

Review from last lecture: (continued)

- A Mhz = ___________________
- A Kbyte = __________________
- A Mbyte = __________________
- A Gbyte = __________________
- The binary number 1011 = _______ (in base 10)
- An ALU is the __________________________ and needs to perform only two fundamental operations which are __________ and ________

College of Engineering



KJC008

Review from last lecture: (continued)

- A client is a ___________________ and typically runs a _______________ operating system such as ___________
- A server is a ________________ and typically runs a ____________ operating system such as _______________
- An example of a server is ___________________
- A type of network connecting clients and servers is ____________ and it runs at a data rate of __________________

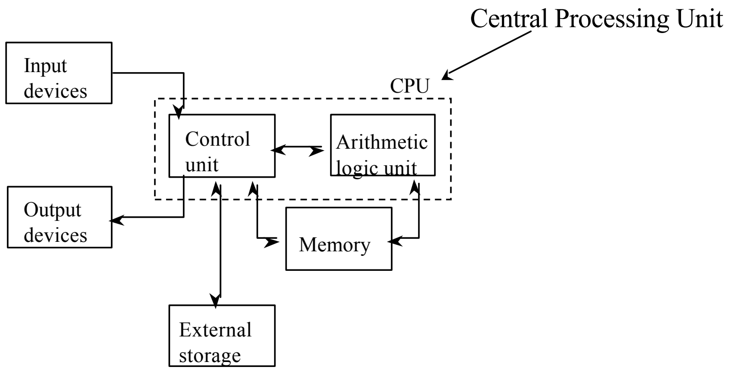
College of Engineering

KJC009

The Von Neumann model:

Review

- The organization of components in a computer
  - A logical view



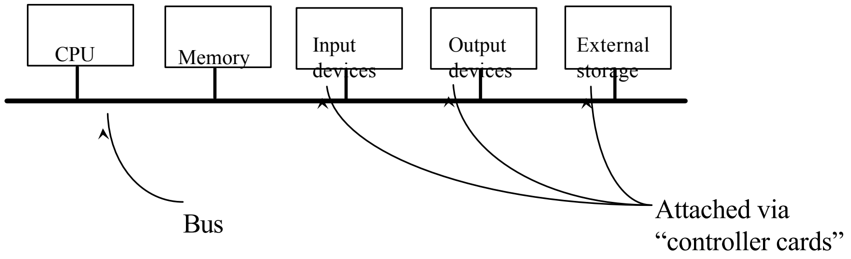
College of Engineering

KJC010

Components of a computer: (continued)

Review

- A typical implementation has all components attached to a bus
  - A physical view



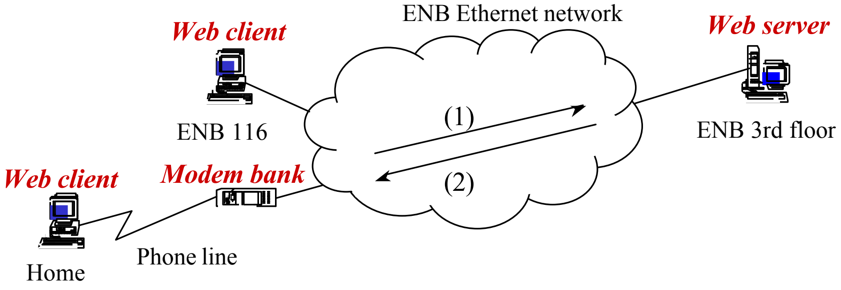
College of Engineering

KJC011

Client/server: (continued)

Review

- Typically, a network connects clients and servers
  - An example is the Web



ENB Ethernet network

Web client

ENB 116

Web server

ENB 3rd floor

(1)

(2)

Web client

Home

Modem bank

Phone line

(1) = *Request* from client to server for a Web page
(2) = *Response* from server (Web page, inline images, download files, etc.)

College of Engineering

KJC012

How memory works:

Review

- Everything in a computer is binary
  - Binary means two states, 0 or 1
  - A bit = one state
  - A byte equal 8 bits
  - A byte can represent 256 values (00000000, 00000001, … 11111111)

- Each digit in a binary number represents a power of 2 (base 2)
  - N bits can represent 2^N values
  - 011 = 3
  - 101 = 5
  - 1001 = 9
  - 10000011 = 131

- Standard “codes” exist to represent letters of alphabet, etc.
  - The ASCII code is standard for characters

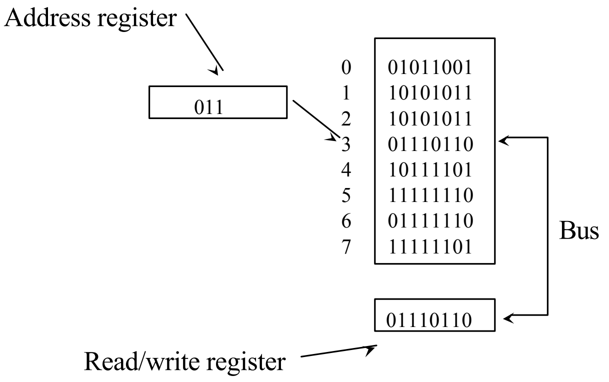
College of Engineering

KJC013

How memory works: (continued)

Review

- Memory is a grid (analogous to a spreadsheet)
  - Each grid location has an address
  - At each address 8, 16 , or 32-bits are stored
  - A register is a one word memory (found in the CPU)



0	01011001
1	10101011
2	10101011
3	01110110
4	10111101
5	11111110
6	01111110
7	11111101

College of Engineering

KJC014

How memory works: (continued)

Review

- Memory contains *instructions* and *data*
  - Instructions tell the CPU what to do
  - Data is what the CPU operates on

- Random Access Memory (RAM)
  - Can both read and write
  - But, when power is removed the contents are deleted

- Read Only Memory (ROM)
  - Can only read
  - But, contents are retained even when power is removed
  - Typically used for “boot-up” programs

College of Engineering

KJC015

Machine language:

- *Machine language* is defined as bit patterns that control the CPU
  - Each bit pattern causes a specific action to occur
- CPU operations include
  - Moving data from memory to CPU
  - Move data from CPU to memory
  - Arithmetic operations on data when in the CPU
  - Also, decision and branch instructions
- Every brand of CPU has its own machine language
  - RISC = Reduced Instruction Set Computer (e.g., UNIX workstation)
  - CISC = Complex Instruction Set Computer (e.g., PC)

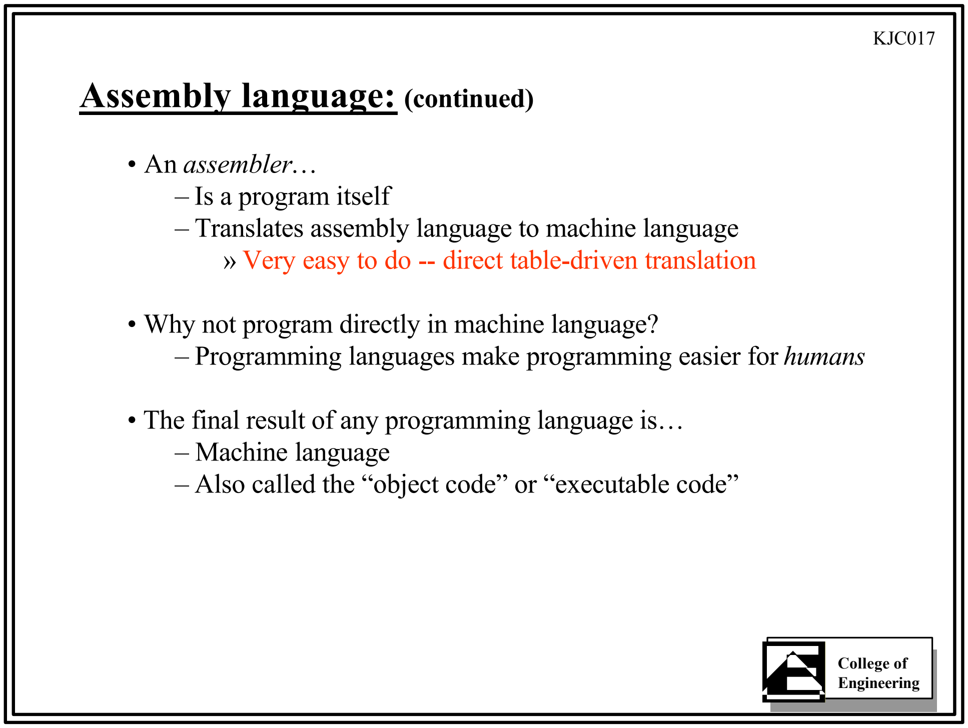
KJC016

Assembly language:

- In the old days (1950’s through 1960’s)…
  - Programmed directly in machine language
  - Entered 1’s and 0’s into computer memory via a panel of switches
- First step was to invent *assembly language*
  - Assembly language gives each bit pattern a mnemonic
  - Mnemonic was easier to remember than a bit pattern

Example:

```
mov a, b                     ; Move contents of register b to a
mov a, addr = 11000110       ; Adds memory to a register
add a, data = 01101111       ; Adds data value to a register
```



KJC017

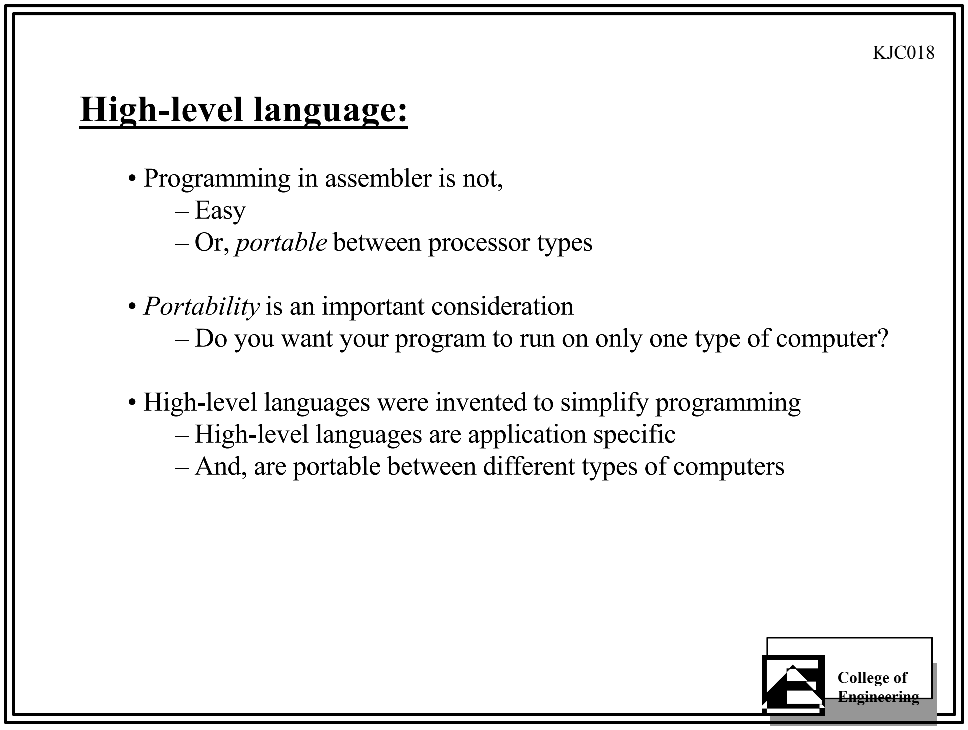
Assembly language: (continued)

- An *assembler…*
  - Is a program itself
  - Translates assembly language to machine language
    - Very easy to do -- direct table-driven translation

- Why not program directly in machine language?
  - Programming languages make programming easier for *humans*

- The final result of any programming language is…
  - Machine language
  - Also called the "object code" or "executable code"

College of Engineering



KJC018

High-level language:

- Programming in assembler is not,
  - Easy
  - Or, *portable* between processor types

- *Portability* is an important consideration
  - Do you want your program to run on only one type of computer?

- High-level languages were invented to simplify programming
  - High-level languages are application specific
  - And, are portable between different types of computers

College of Engineering

KJC019

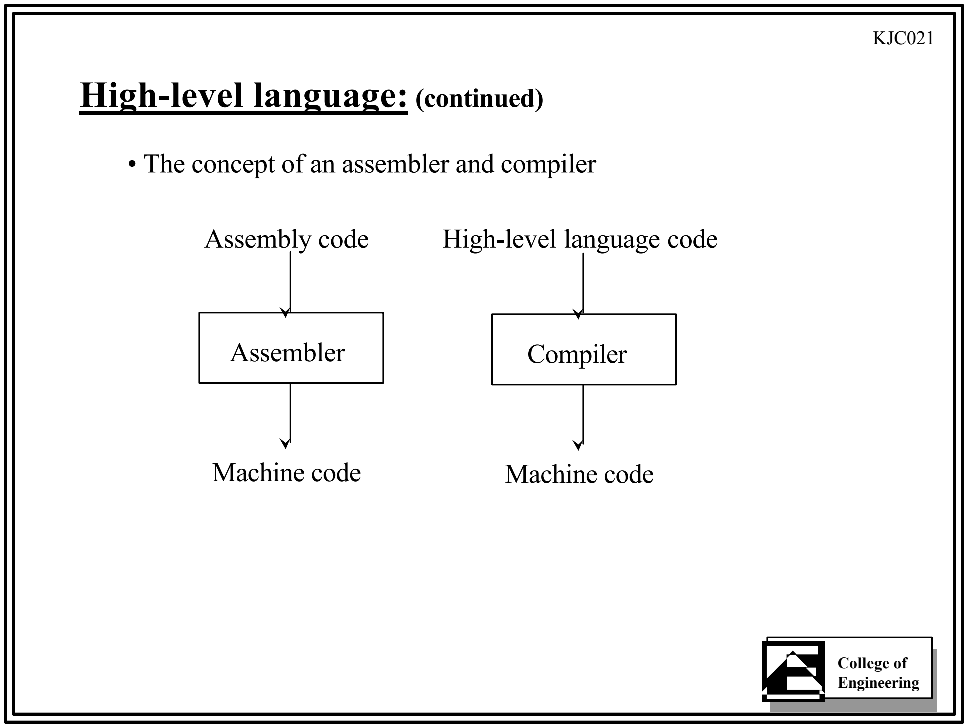
High-level language: (continued)

- Procedural languages
  - – FORTRAN
  - – COBOL
  - – Pascal
  - – Ada
  - – C
- Objected Oriented Programming
  - – A new way of thinking about data and procedures together
    - » C++
    - » Smalltalk
    - » Java

KJC020

High-level language: (continued)

- FORTRAN
  - – Formula Translation - for engineering and scientific applications
- COBOL
  - – Common Business Oriented Language - for business applications
- C
  - – For writing operating systems and other “systems programs”
- Ada
  - – Mandated by Department of Defense
- C++
  - – An object oriented flavor of C for easier team programming
- Java
  - – Very similar to C++ for ??? application



KJC021

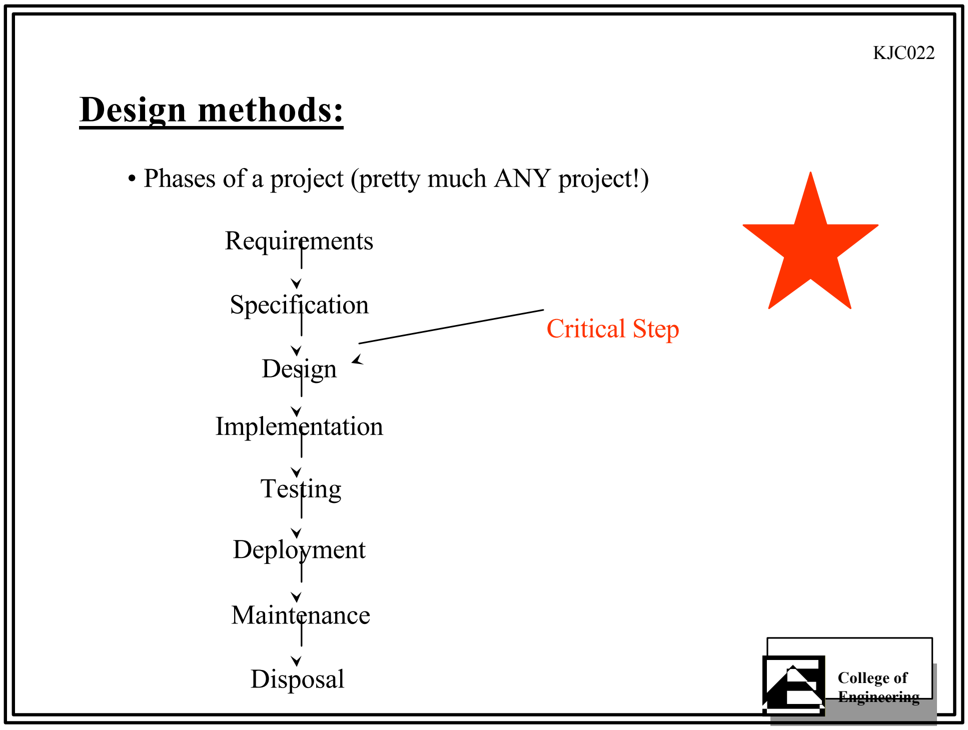
High-level language: (continued)

- The concept of an assembler and compiler

Assembly code → Assembler → Machine code

High-level language code → Compiler → Machine code

College of Engineering



KJC022

Design methods:

- Phases of a project (pretty much ANY project!)

Requirements → Specification → Design → Implementation → Testing → Deployment → Maintenance → Disposal

Critical Step

College of Engineering

KJC023

Design methods: (continued)

- Flow charting
  - Simple way to describe an *algorithm* for a small task
- Divide-and-conquer
  - Method of breaking a big problem into small tasks
- Successive refinement
  - Method of adding detail to a small, but ambiguous task

College of Engineering

KJC024

What is an algorithm?

- An algorithm is simply a set of steps to accomplish something
  - A cooking recipe
  - Instructions to assemble a tricycle
  - Procedure to overhaul an engine

Formally, an algorithm is defined as...
1) Described in a finite sequence of instructions
2) Each instruction is executable
3) Execution always terminates

Hint: If it is in red (like the above is), then it is important!

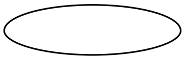
College of Engineering

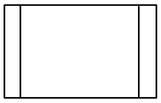
KJC025

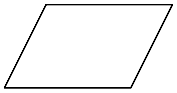
Flow charting:

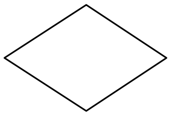
Memorize these!

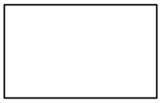
- Flow charting symbols

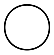

- Begin/End


- Subprogram (i.e., another flowchart)


- Input/output


- Decision


- Assignment or computation


- Continuation

College of Engineering

KJC026

Flow charting:

- Flow charting symbols (have you memorized ‘em yet???)

- Begin/End
- Subprogram (i.e., another flowchart)
- Input/output
- Decision
- Assignment or computation
- Continuation

College of Engineering

KJC027

Flow charting:

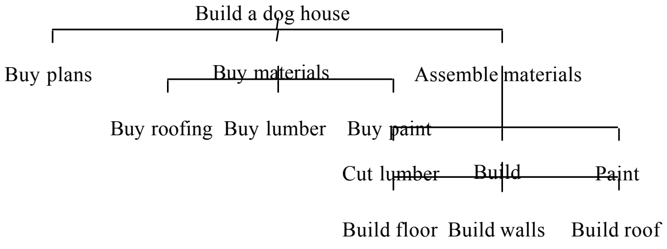
- Do some examples on the board

College of Engineering

KJC028

Divide-and-conquer:

- Results in a *structure diagram*
  - Useful for breaking a large project into manageable tasks
  - Procedure - repetitively partition a job into small tasks



College of Engineering

KJC029

Divide-and-conquer: (continued)

- Divide-and-conquer example
  - See pages 27 and 28 in book

College of Engineering

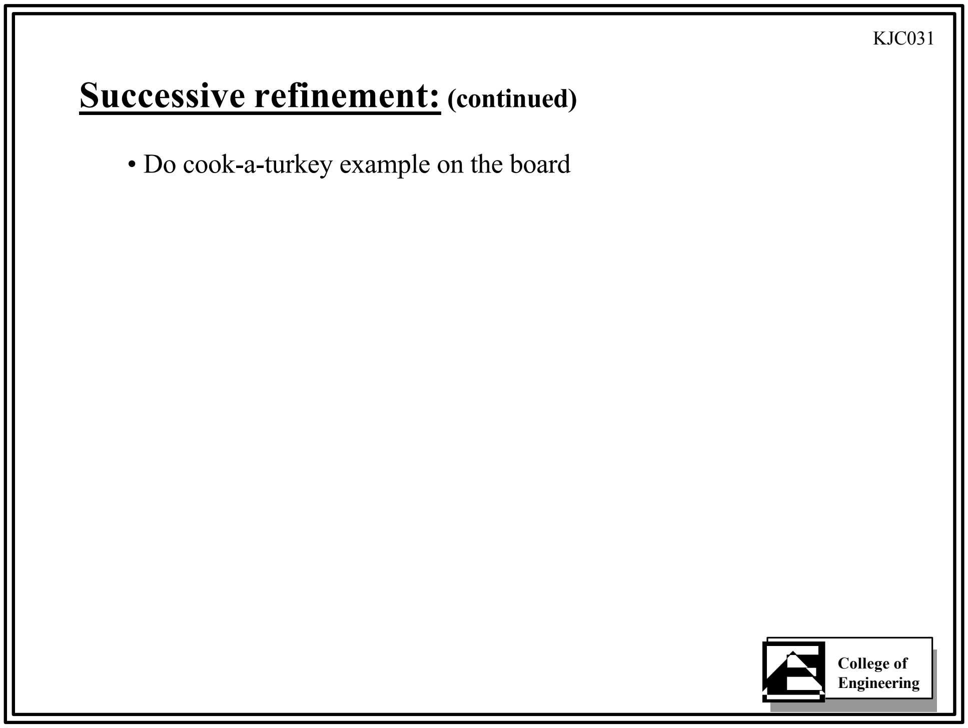
KJC030

Successive refinement:

- Go from ambiguous to precise
  - Start with a general, albeit small, task
  - Can initially "hide" details
  - Then, add detail until we have a flowchart that can be implemented

You will use this method a lot for your programming problems.

College of Engineering

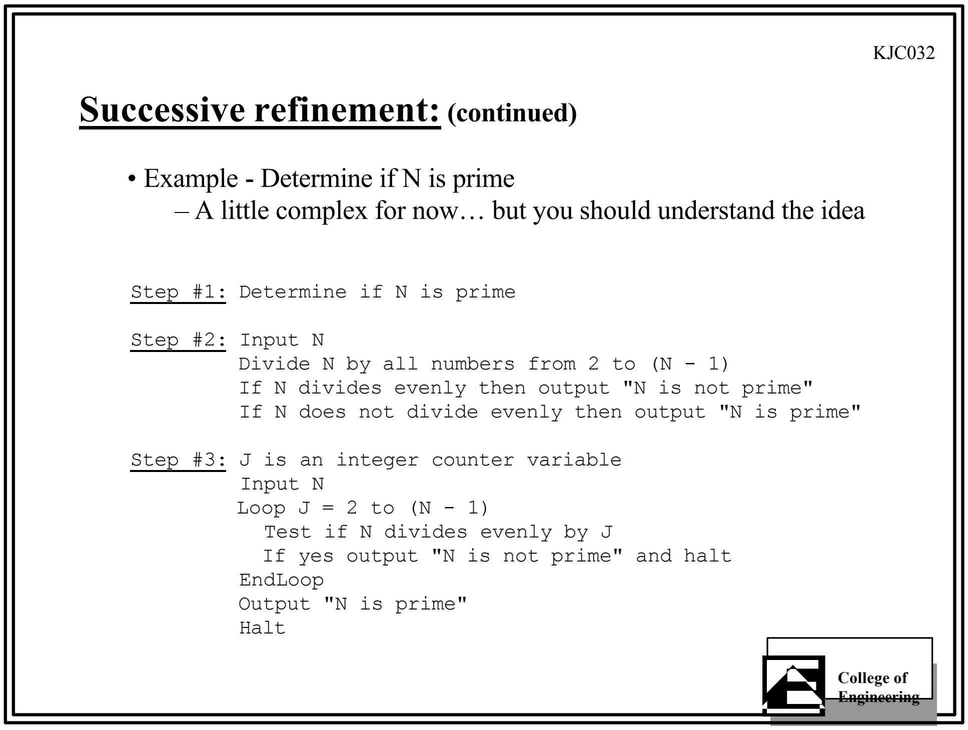


KJC031

Successive refinement: (continued)

- Do cook-a-turkey example on the board

College of Engineering



KJC032

Successive refinement: (continued)

- Example - Determine if N is prime
  - A little complex for now… but you should understand the idea

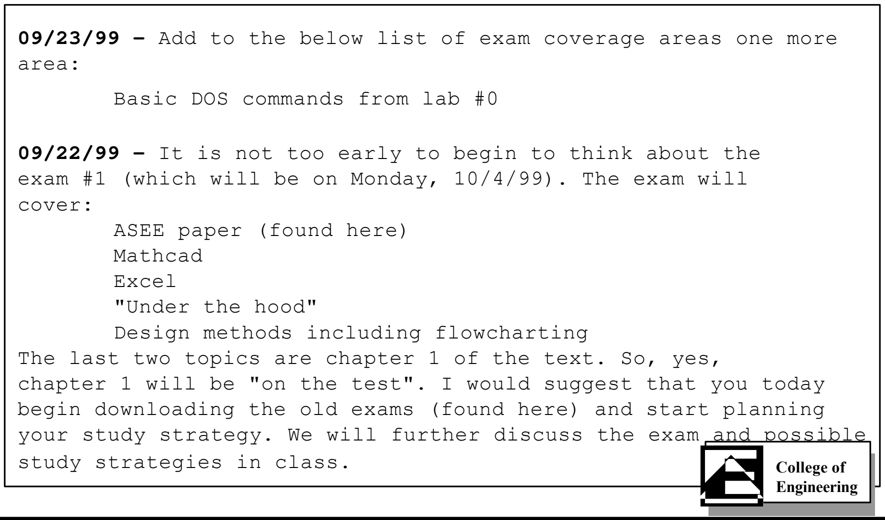
```
Step #1: Determine if N is prime

Step #2: Input N
         Divide N by all numbers from 2 to (N - 1)
         If N divides evenly then output "N is not prime"
         If N does not divide evenly then output "N is prime"

Step #3: J is an integer counter variable
         Input N
         Loop J = 2 to (N - 1)
           Test if N divides evenly by J
           If yes output "N is not prime" and halt
         EndLoop
         Output "N is prime"
         Halt
```

College of Engineering

KJC033

Coverage for Exam #1:

From Notices page



09/23/99 - Add to the below list of exam coverage areas one more area:

Basic DOS commands from lab #0

09/22/99 - It is not too early to begin to think about the exam #1 (which will be on Monday, 10/4/99). The exam will cover:

ASEE paper (found here)
Mathcad
Excel
"Under the hood"
Design methods including flowcharting

The last two topics are chapter 1 of the text. So, yes, chapter 1 will be "on the test". I would suggest that you today begin downloading the old exams (found here) and start planning your study strategy. We will further discuss the exam and possible study strategies in class.

College of Engineering

KJC034

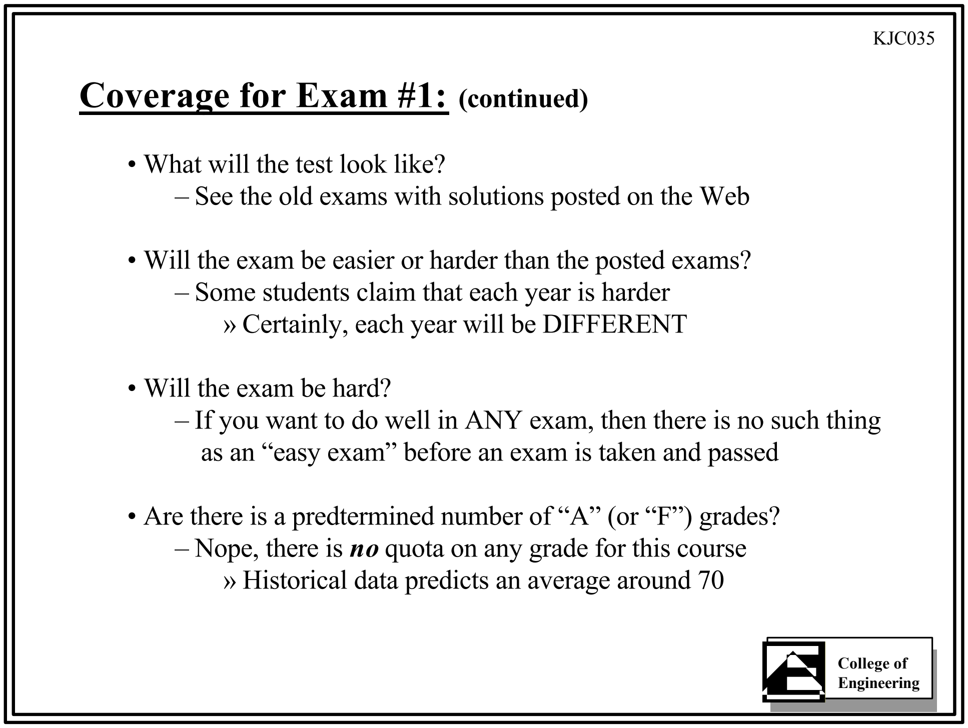
Coverage for Exam #1: (continued)

- Approximate problem breakdown (*no lawyers, please*!)
  - 1 short answer problem with about 12 fill-ins
  - 1 DOS problem
  - 1 multipart Mathcad problem
  - 2 or 3 Excel problems
  - 2 or 3 “under the hood” problems
  - 2 design problems
  - 1 “anything goes” extra credit problem

10 problems total

- So, what am I responsible for?
  - Everything covered in the ASEE paper
  - Everything covered in lab
  - Everything covered in class
  - Everything in chapter 1 except the FORTRAN code
    - » But, certainly the flowcharts and design stuff are “in”

College of Engineering

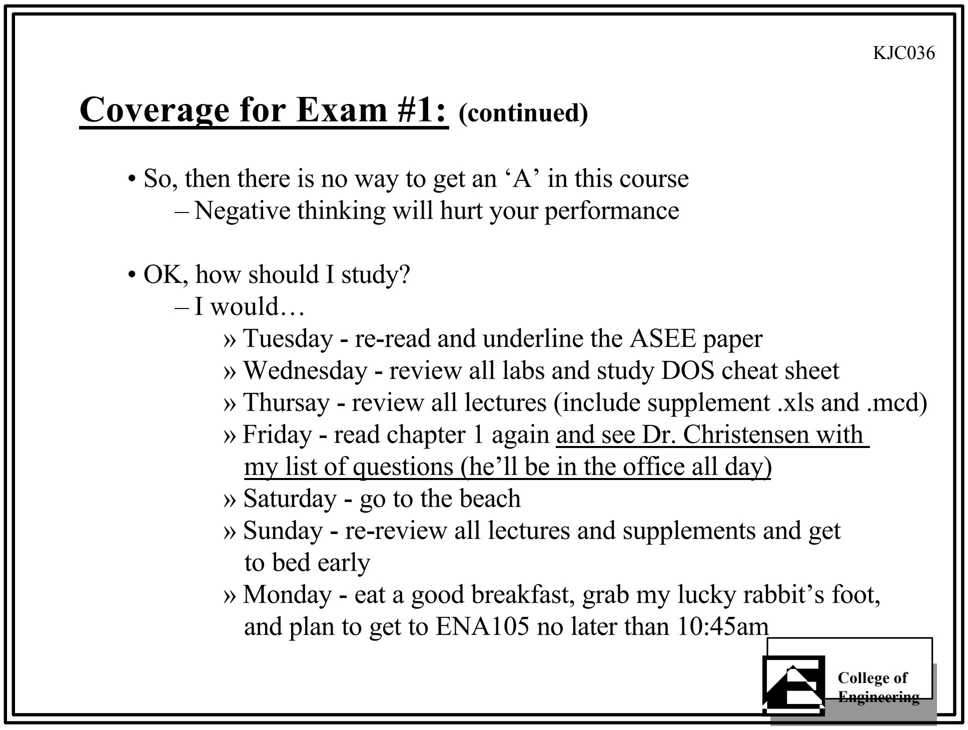


KJC035

Coverage for Exam #1: (continued)

- What will the test look like?
  - See the old exams with solutions posted on the Web
- Will the exam be easier or harder than the posted exams?
  - Some students claim that each year is harder
    - » Certainly, each year will be DIFFERENT
- Will the exam be hard?
  - If you want to do well in ANY exam, then there is no such thing as an “easy exam” before an exam is taken and passed
- Are there is a predtermined number of “A” (or “F”) grades?
  - Nope, there is ***no*** quota on any grade for this course
    - » Historical data predicts an average around 70

College of Engineering

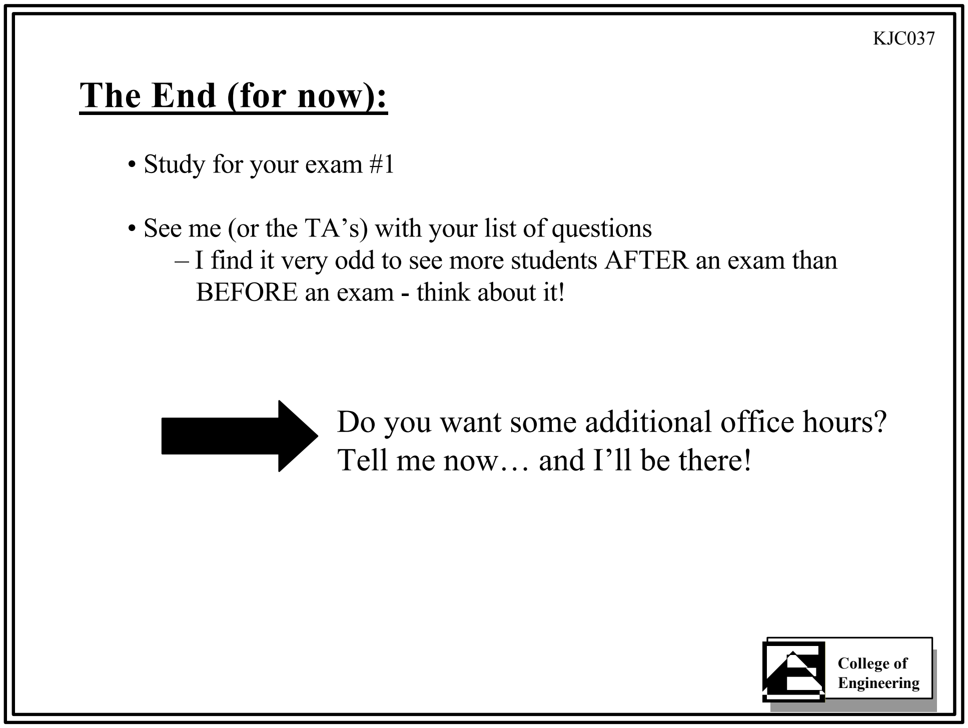


KJC036

Coverage for Exam #1: (continued)

- So, then there is no way to get an ‘A’ in this course
  - Negative thinking will hurt your performance
- OK, how should I study?
  - I would…
    - » Tuesday - re-read and underline the ASEE paper
    - » Wednesday - review all labs and study DOS cheat sheet
    - » Thursay - review all lectures (include supplement .xls and .mcd)
    - » Friday - read chapter 1 again and see Dr. Christensen with my list of questions (he’ll be in the office all day)
    - » Saturday - go to the beach
    - » Sunday - re-review all lectures and supplements and get to bed early
    - » Monday - eat a good breakfast, grab my lucky rabbit’s foot, and plan to get to ENA105 no later than 10:45am

College of Engineering



KJC037

The End (for now):

- Study for your exam #1
- See me (or the TA’s) with your list of questions
  - I find it very odd to see more students AFTER an exam than BEFORE an exam - think about it!

Do you want some additional office hours?
Tell me now… and I’ll be there!

College of Engineering