

Structured Programming VI: Examples, and Strategy

by the end of this class you should be able to:

- read and learn from example loop programs
- develop modular program

Text Sections: 4.1, 4.5,

Download:

First four files for today's date from class website
(quiz1.m, A.mat, eng2pl.m and eng2plword.m)

Download: *quiz1.m* & *A.mat*

Run: `>> quiz1`

Take the quiz

Find a partner , answer questions on handout

1. How does this program step through the questions in the input file regardless of how many there are?
2. What does the while loop in this program do? Why is a while loop essential here?
3. How does the flag or indicator variable *x* work (look at all commands where it is used) ?
4. How is the score determined and maintained?

Quiz Program I/O statements

- Download *quiz1.m* and *A.mat* from website
- Run *quiz1.m* from command line (`>> quiz1`)
- Look at file (handout)
- Things to notice:
 - Use of for loop to move through questions → stored file can contain any number of questions
 - Use of while loop, if statement, and switching variable (*x*) to repeat until questions is correct
 - Use of accumulation variables
 - *qscore* - decreasing
 - *score* - increasing
 - input via, file & menu command
 - output via display with some concatenation

Try new Pig Latin Program

- Download *eng2PLword.m*
Try `>> eng2PLword('Charlie')`
- Download *eng2PL.m*
Try `>> eng2PL('Matlab is Fun')`

With your partner examine the code and answer the handout questions

Pig Latin Translator functions

1. How is each word separated out for translation
2. How are translated words assembled into a Pig Latin phrase?
3. How is *eng2plword* able to move consonant groups not just the first consonant?
4. Do you for see any cases that our translator might still have trouble with?

Loops

While

initial definition of accumulation & loop variable(s)

while logical expression

statements (do change loop variable(s))

end

Use when number of iterations cannot be determined in advance

For

initial definition of accumulation variable(s)

for loopvariable = array statements

(use but do not change the loop variable, do update accumulation variable(s))

end

Use when number of iterations can be determined in advance

Common For Loop Logic

Fill

Each loop →

- fills a value into a new location in an array

Example →

- square root a vector (e-by-e)
- 2-D array fill

Key →

- The loop index variable is used as an array index

Accumulation

Each loop →

- modifies the accumulation variable with something new

Example →

- the factorial problem
- Pig Latin sentence program

Key →

- the accumulation variable appears on both sides of the = in the loop &
- must be initialized before the loop is run.

Common While Loop Logic

Accumulation

Each loop →

- modifies the accumulation variable with something new
- Often used to count number of loops

Example →

- Inverse factorial
- Millionaire problem
- Loop counters in general

Key →

- the accumulation variable appears on both sides of the = in the loop &
- must be initialized before the loop is run.

Indicator

Each loop →

- Checks to see if indicator variable has been changed
- Indicator variable is changed by loop under certain conditions.

Example →

- Quiz program (quiz1.m)
- Pig Latin word program

Key →

- Set indicator variable before loop
- Loop checks if indicator value equals the initial value
- Have conditions in loop that can change indicator variable (with user input and/or a conditional statement)

Program Structure

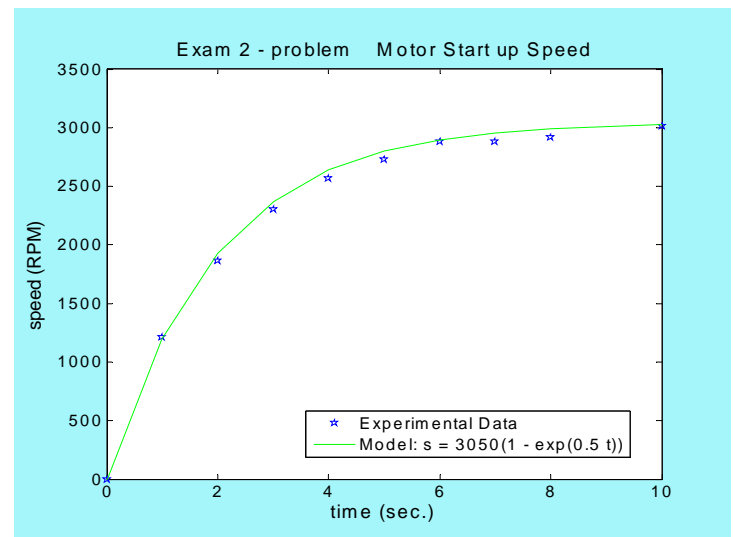
- Types of statements
 - sequential → statements in a row
 - conditional → if, else, switch
 - iterative → for & while loops
- Modular Programming
 - Break problem into separate parts
 - subroutines, subfunctions
 - write & test the parts

Progressive Development Example: Pig Latin Program

- Developed single word translator first
 - Simple move letter
 - Added consonant vowel difference
 - Added consonant groups
- Developed sentence translator using word translator
 - I/O simple command line → does not require a separate module
 - Add loops and conditionals to work through sentence.

Exam Review

- Graphing Problem
 - Resulting graph - key characteristics
 - How see solution
 - May return
- Function Problem Alternatives
 - if-elseif-else structure
 - Switch structure
 - With logical statements only
 - Try t = [
 - T <= 100
 - What do you get?
 - How might I use this?



```
% Program motor
% Solution ENGR121, Exam 2, problem 7, Spring 2009
% This script prepares a plot of the data and model provided in the
% problem
% for the start up of a motor
% Enter data
s = [0 1210 1866 2301 2564 2724 2881 2879 2915 3010];
t = [0:8,10];
% Calculate model values at experimental times
sm = 3050*(1-exp(-0.5*t));
% plot and label the data and model
plot(t,s,'p',t,sm)
xlabel 'time (sec.)', ylabel('speed (RPM)')
title('Exam 2 - problem Motor Start up Speed')
legend('Experimental Data', 'Model: s = 3050(1 - exp(0.5 t))')
```