Outline

- Feedback
- Regular Expressions
- Graphics
- Library Modules
- Creating a library

- Reading: Chapter 4,5 (pg-167)
Feedback from last class

• More code
• Slides online
• Pace of course

Announcement

• Bill gates will be on campus October 13, tickets will be available fcfs basis
• Keep eyes open
Homework 1

• Download from webpage www.columbia.edu/~cs3157

• use perl in a practical project
• Learn about computer security

Computer Security

• System and theory of ensuring the confidentiality, integrity, availability, and control of electronic information and systems.
  – Network
  – Host
  – Data
For host based security

- Want to ensure permission system
  - X should only be allowed to do A, B, and C
- Want to ensure accountability
  - If Y does something not allowed, should be noted
- Want to be able to track
  - If something has been tampered with, how can we locate it
  - Both preventative and reactionary

Project

- Assuming you are a system administrator or just paranoid
- Take chronological snapshots of your system to compare and find changes
  - Many changes by system
  - Many changes by valid user
  - Might locate malicious user/system changes
Useful tips

• Can turn on warning to help prevent errors
• Run in strict mode to catch potential mistypes
• Create debugging statements to help chart progress throughout program…
• Better yet, learn to use the perl debugger (next week).

Doing the work

• Find a good perl environment
• Read up on perl
• Can work
  – Home
  – Clic lab
  – Home, remote on clic machine
TOOLS: VNC

• www.realvnc.com

• Start server on a clic machine:
  – vncserver

  – Run client on your side

  – demo

Regular Expression

• Review
• new examples
Regular Expressions

- simplest regular expression is a literal string
- complex regular expressions use *metacharacters* to describe various options in building a pattern.

<table>
<thead>
<tr>
<th>Character</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>\</td>
<td>escapes the character immediately following it</td>
</tr>
<tr>
<td>.</td>
<td>matches any single character except newline</td>
</tr>
<tr>
<td>^</td>
<td>matches at the beginning of a string</td>
</tr>
<tr>
<td>$</td>
<td>matches at the end of a string</td>
</tr>
<tr>
<td>*</td>
<td>matches the preceding element 0 or more times</td>
</tr>
<tr>
<td>+</td>
<td>matches the preceding element 1 or more times</td>
</tr>
<tr>
<td>?</td>
<td>matches the preceding element 0 or 1 times</td>
</tr>
<tr>
<td>(...)</td>
<td>specifies a range of occurrences for the element preceding it</td>
</tr>
<tr>
<td>[...]</td>
<td>matches any one of the class of characters in the brackets</td>
</tr>
<tr>
<td>( ... )</td>
<td>groups expressions</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Basic

- The most basic match is:
  - `$string =~ m/sought_text/;`
  - Will return true if `sought_text` is part of `string`, false otherwise
  - Perl assume `m/??/?` when use `/??/`

1) if (/shlomo/) {....}
2) if (not /shlomo/) {...}
   if($_ !~ /shlomo/) {...}
Basic II

1. if ($a =~ /s|h|m/) {.....}
2. if ($a =~ /[a-z]/) {...}

3. How would we look for a phone number?

4. What about a social security?

Pattern attributes

• operators:
  – m/pattern/gimosx : match
• g = match globally (all instances)
• i = do case insensitive matching
• e = evaluate right side as an expression
• s = let . match newlines
• m = $ and ^ can refer to inside newlines
groups

To allow groups of alternative choices

if($string =~ /(A|E|I|O|U|Y)/i)
{ print "String contains a vowel!n"; }

Alternatively we can also specify character choices:
if( $string =~ /[AEIOUY]/i )
{ print "String contains a vowel!n"; }

Can also specify ranges
if( $string =~ /^[a-e]/I )
{ something }

Groups II

• To allow us to reference for selection and substitution
• Each group can be referred to by scalar $1, $2, $3 ….

Example
• “From s@aol.com Wed Jun 3 12:12:12 2005”
• If(/^[A-Z]* (.* (\d+) (\d+) (\d+) (\d+)$/)
shortcuts

$name = "advanced programming class"

if($name =~ /programming/){
    print $` ;
    print $& ;
    print $' ;
}

subsitations

• s/pattern/pattern/
transliteration

- tr/search_list/replacement_list/
- -c all characters not in the search list
- -d anything without replacement …delete
- -s squash dublicates

quantifiers

- ba*b
- ba{3,5}b
- ba{1}b
Buffer Overflow

• What is it?

Security

• Should use pattern matches as a security check on input

• Example:

  unless ( $year =~ /^\d\d$/ ) {
    die ("problem with year input!");
  }


hashes

• A hash function is a function that converts an input from a (typically) large domain into an output in a (typically) smaller range

• Example:
  – Map each name in the class to a somewhat unique number

• Collision = when different keys map to the same output.

Use of hashes

• Hash tables
  – Data structure
  – Unordered list, fast lookup

• Cryptography

• Data processing
MD5 Sum

- MD5 – uses a 128 bit hash value
- Designed in 1991
- Known problems with collision attacks

Using Perl Libraries
**Digested MD5 Perl - Perl implementation of RFC 1321 MD5 Algorithm**

**DISCLAIMER**

This is not an interface (like Digest::MD5) but a Perl implementation of MD5. It is written in perl only and because of this it is slow but works without C-Code. You should use Digest::MD5 instead of this module if it is available. This module is only useful for:

- computer where you cannot install C-modules (e.g. lack of a C-Compiler)
- encrypting only small amounts of data (less than one meter byte), I use it to hash passwords.
- educational purposes

**SYNOPSIS**

```perl
# Functional style
use Digest::MD5 qw(digest); # digest with hash function
digest = digest($hash, $password);

# or
digest = Digest::MD5::new->digest($hash, $password);

# with config
digest = Digest::MD5::new(config => {
    Algorithm => 'MD5',
    Style     => 'lowcase',
    Verbose   => 0,
    Signature => 0,
    Signature => 0,
})->hexdigest($hash, $password);
```

**DESCRIPTION**

This modules has the same interface as the much faster Digest::MD5. Be you can easily exchange them, e.g.
Digests

- The 128-bit (16-byte) MD5 hashes (also termed message digests) are typically represented as 32-digit hexadecimal numbers.
- Even small change can result in a totally different hash digest
- MD5("The quick brown fox jumps over the lazy dog") = 9e107d9d372bb6826bd81d3542a419d6
- MD5("The quick brown fox jumps over the lazy cog") = 1055d3e698d289f2af8663725127bd4b
- MD5(""") = d41d8cd98f00b204e9800998ecf8427e

MD5 Attacks

- Recent work has found flaws with the MD5 sum.
- Will not consider this in our class.
scope

• Default scope is main
• $name can also be referred to as $main::name

• package NAMESPACE
  – Within any block of code, can declare that the rest of the code will belong to a specific namespace

Scope II

• my declares the variable and value local to the current scope
• our confines the name to local scope
• local confines the value to local scope
• More than one variable in parenthesis!!
What exactly is a module

• Collection of useful subroutines or objects for a specific task

Creating a simple library

• Will do it in next weeks lab
Graphics

```perl
#!c:\perl\bin
use Tk;

my $mwin = MainWindow->new;

$mwin->Button(-text => "Hello World!", -command => sub{exit})-pack;
MainLoop;
```

• Will not cover in depth
• Good to know about
• Might need to one day debug someone else’s code (GASP!)
For next time

- Reading
- Make sure you have cs account for next week lab
- Start sketching the homework