Tcl/Tk and Janus
Brief introduction

11-753
Tcl/Tk

• Developed by John Ousterhout
• Tcl = Tool Command Language
  – Scripting language with simple structure
• Tk is a cross platform widget Tool Kit
  – Part that is responsible for graphical front-end
  – Available also in perl and python
  – Tk has an object oriented look and feel
• Use only 1 data type: string
• Designed to allow easy extension (C level)
How to start Tcl/Tk

• Unix:
  – `tclsh` start tcl
  – `wish` start tcl+tk
  – `janus` start tcl+tk+janus
  – `janusNX` start tcl+janus

• Important environment variables
  – `TCL_LIBRARY`, `TK_LIBRARY` if Tcl/Tk is not installed in the default directory
  – `JANUS_LIBRARY`, `JANUS_LIB`, if not compiled by yourself
• Tcl basics in 5 minutes

• Useful literature and links
  – Brent Welch “Practical Programming in Tcl and Tk”
  – ActiveState Tcl/Tk distribution and code collection
    • [http://www.activestate.com/Products/ActiveTcl/](http://www.activestate.com/Products/ActiveTcl/)
Elements of Tcl

• General structure
  – `<command> parameter1 ... parameterN ;`
  – Command name invokes an internal function that gets the parameters passed

  – Examples
    • `set myVar 1`
      – Command “set” creates a variable “myVar” and assigns “1”
    • `puts stdout “hello world”`
      – Command “puts” writes to stdout the string “hello world”
      – We have to quote the string to make it one parameter
Command substitution

• If a word contains an open bracket (``[") then Tcl performs *command substitution*.  
set myVar [expr 1 + 1]
• To do this it invokes the Tcl interpreter recursively to process the characters following the open bracket as a Tcl *script*.
• The *script* may contain any number of commands and must be terminated by a close bracket (``"]").
• The *result* of the *script* is substituted into the word in place of the brackets.
• Command substitution is not performed on words enclosed in braces “{}``. 
Variable substitution

• If a word contains a $ then Tcl performs variable substitution.
• Variable substitution may take any of the following forms:
  – $name
    • Name is the name of a scalar variable.
  – $name(index)
    • Name gives the name of an array variable and index gives the name of an element within that array. Command substitutions, variable substitutions, and backslash substitutions are performed on the characters of index.
  – ${name}
    • Name is the name of a scalar variable. It may contain any characters whatsoever except for close braces.
• There may be any number of variable substitutions in a single word.
  – set myVar $my$Var
• Variable substitution is not performed on words enclosed in braces
  – set myVar {my$Var}
Order of substitution

– Each character is processed exactly once by the Tcl interpreter as part of creating the words of a command.

– For example, if variable substitution occurs then no further substitutions are performed on the value of the variable; the value is inserted into the word verbatim.

– Substitutions take place from left to right, and each substitution is evaluated completely before attempting to evaluate the next. Thus, a sequence like

– set y [set x 0][incr x][incr x] will always set the variable y to the value, 012.
Common Tcl difficulties

set $myVar 10; # this is a valid command if myVar is defined!
Sometimes useful but often a typo.

regexpr "[ab]c$" $myVar; # it is usually better to quote the regular
expression in "{}".

set x 3; set y 2;
set myVar [expr $x / $y]; # myVar contains 1

set x 3.0; set y 2;
set myVar [expr $x / $y]; # myVar contains 1.5

Solution:
set x 3; set y 2;
set myVar [expr $x / double($y)]; # myVar contains 1.5
Janus design

• Janus is integrated in Tcl/Tk
  – All commands from Tcl/Tk available
  – Allows rapid prototyping with a nice GUI

• Janus is designed object oriented, but implemented in C
How to create Janus objects

• Type class name and instance name
  – FMatrix fm; # create instance fm of class float matrix
  – FeatureSet fs; # create instance fs of class FeatureSet
  – CodebookSet cbs fs; # instance cbs of class CodebookSet that operates on fs
  – cbs add cb1 FEAT 2 10 DIAGONAL; # creates a codebook instance in the set of codebooks cbs with 2 Gaussians of dimension 10 and diagonal covariance which use the feature FEAT stored in fs
Getting help

<class> and <instance> are names of janus objects

% <class> dummy -help
List the parameters required to create an instance

% <instance> -help
List the methods available for this instance

% <instance> method -help
List the parameters of the methods with default values
Access to sub object

- 2 types of sub objects
  - Sub-objects
    - `<instance>.[subobject]`
    - Ex. `cbs.featureSet`
    - `Cbs.featureSet -help ;# shows help of featureSet`
    - `<instance>`. List available sub objects
  - Element in set of objects
    - `<instance>:<element name>`
    - Ex. `cbs:cb1`
    - `<instance>: ;# list all available elements of set`
Homework

• Read Tcl/Tk man pages and play a little
• Write a Tcl program that saves the first 100 prime numbers into a file. Use arrays.