# TCL/Tk Introduction

- About TCL
- About Tk
- Basic TCL Syntax
- TCL Command Overview
- Tk Basic Commands

# TCL/Tk Introduction – About TCL and Tk

## About TCL

- Simple and programmable syntax
- Can be used as standalone application or embedded in programs
- Open source
- Interpreted language
- New TCL commands can be implemented using C language

## About Tk

- Graphical user interface (GUI) toolkit
- Tk adds about 35 TCL commands
  - Create and manipulate widgets
- Widget is a window in GUI with particular appearance and behavior

# TCL/Tk Introduction – Basic Tcl Syntax

- First item of a command line is a command name.
- The words in the command line are separated by one or more spaces.
- Words can be grouped with double quotes or curly braces.
- Commands are terminated with new line or semi-colon.
- A word starting with a dollar sign ($) must be a variable name. The string will be replaced by the value of the variable.
- Words enclosed within square brackets must be a legal Tcl command. The strings would be replaced by the results of evaluating the command.

# TCL/Tk Introduction – TCL Command Overview

- Text Output
- Variables and Variable Substitution
- Expressions
- Command Substitution
- Comparisons and Loops
- Lists
- Arrays
- Strings
- Reading and Writing Data to Files
- Procedures
- Namespaces

# TCL Command Overview – Text Output

- Print a string using `puts` command

Input:

```
puts HyperWorks
```

Output: ***HyperWorks***

- If string is more than one word, the string must be enclosed in either double quotes (“ “) or braces ({ })

Input:

```
puts “This is an example with quotes”
puts {This is an example with braces}
```

Output:

***This is an example with quotes***
***This is an example with braces***

# TCL Command Overview – Text Output

- TCL command terminated with newline or semicolon ;
- Comments are designated with **#** at beginning of line or after semicolon

Input:

```
# An example using a semicolon
puts “This is line 1”; puts {This is line 2}; #Note after the ;
```

Output:

```
This is line 1
This is line 2
```

# TCL Command Overview – Variables

- Variables do not need to be declared before using
- Create variables using **`set`** command
- Delete variables using **`unset`** command
- Variable substitution done using $ to access value stored in variable

Input:

```
set software "HyperWorks"
puts "The software we are using is $software"
```

Output: ***The software we are using is HyperWorks***

# TCL Command Overview – Variables

- Variable substitution inside strings
  - Difference between using double quotes and braces

- Variable substitution with double quotes
  - Using \ before variable results in literal value being printed

Input:

```
set Z Albany
set Z_LABEL “This Capitol of New York is: “
puts “$Z_LABEL $Z” ; # Prints the value of Z
puts “$Z_LABEL \$Z”; #Prints a literal $Z instead of the value
                      of Z
```

Output

***Albany***

***The Capitol of New York is:***

***The Capitol of New York is:  Albany***

***The Capitol of New York is:  $Z***