Hewlett Packard Enterprise

TACL programming U4199S

Master the art of writing functions in the Tandem Advanced Command Language (TACL) program on HP NonStop servers in this 5-day course. Through student projects and hands-on labs, you will gain valuable experience with TACL programming. After completing this course, you will be able to write macros and routines, perform file I/O, use structured data, and write server functions.

TACL programming

Price	USD $4,000
Links to local schedules, pricing and	**US/Canada** **Mexico/Latin America** **Brazil**
HP course #	U4199S
Category	NonStop
Duration	5 days

Audience

- System programmers
- System and network managers
- Application designers
- Application programmers
- System analysts
- Data communications programmers and analysts

Prerequisites

- Concepts and facilities course
- Knowledge of at least one other programming language
- At least six months of programming experience

Benefits to you

- Segment files
- Define process
- MACRO and ROUTINE functions
- Variable editing
- Server functions
- Exception handling
- Debugging

Course outline

Overview of TACL features
- Productivity aids provided by TACL: HISTORY, FC, ? , ! HELP facility
- Function key, custom prompts, file name templates, and macro files
- TACL features as a programming language

TACL variables
- Obtaining information about variables using either commands or built-in functions
- Using commands or built-in functions to create, initialize, modify, and eliminate variables
- Concept of a "frame" and how it relates to managing variables
- Variable stacks and their levels: what they are and how to create, reference, and eliminate them
- Syntax rules for writing TACL functions
- Lab Exercise (20 minutes):
  - Learn and understand how to logon and use TACL function keys

Directories and segments
- Creating a segment file containing a library function
- Using the existing segment file by attaching it to a directory
- Getting information on the segment file
- Syntax rules for writing TACL functions
- Lab Exercise (30 minutes):
  - Learn to create and use a segment file

Editing variables
- Performing variable file I/O
- Performing global editing of a variable
- Performing line editing of a variable
- Performing character editing of a variable
- Locating the position of a string in a variable
- Extracting lines and characters from a variable

Writing functions - macros
- Syntax required to write macro functions
- TACL's handling of arguments to macro functions
- TACL's expansion of macro functions
- Writing macro functions

Writing functions - #IF statements
- Write functions that use the TACL #IF |THEN| |ELSE| construct
- Lab Exercise (1 hour):
  - Describe the syntax required to write functions in general and macro type functions in particular
  - Describe the different forms of the "control" built-in #IF and contrast when to use one form or the other (#IF or #IF NOT)
  - Write a macro type function that accepts one or more arguments and ensures that the arguments are correct by making use of the "control" built-in #IF

Writing functions - #LOOP statements

- Writing functions that use the TACL #LOOP |DO| |UNTIL| construct
- Writing functions that use the TACL #LOOP |WHILE| |DO| construct
- Lab Exercise (1 hour):
  - Describe the syntax required to write general functions, with particular focus on macro type functions
  - Describe the two forms of the "control" built-in #LOOP and determine when to use #LOOP | DO | | UNTIL | or #LOOP | WHILE | | DO |
  - Write a macro type function that outputs all of the volume names on the system

Writing functions - #CASE statements

- Writing functions that use the TACL #CASE construct

Writing functions - debugging

- Using the TACL debugging facility provided by TACL to aid in getting functions to work
- Lab Exercise (2 hours):
- Start and stop the Debugger
- Set and clear breakpoints
- Display and modify the contents of a variable
- Single step through your function and resume execution of your function
- Describe the syntax for #IF, #LOOP, and #CASE constructs
- Write a function that employs the #CASE built-in

Writing functions - file I/O

- How TACL is able to do device independent I/O
- Using #REQUESTER and #WAIT to perform either "waited" or "no-waited" I/O to files and devices

Writing functions - routines

- Writing "Routine" type functions and use #ARGUMENT, #MORE, and #REST
- Lab Exercise (3 hours):
  - Modify and write routine functions
  - Describe the syntax and usage of #ARGUMENT and #MORE
  - Describe additional capabilities that routines offer that macros do not
  - Describe the use of the built-ins: #MYSYSTEM, #PROCESSORSTATUS, and #PROCESSORTYPE, #LOOP, and #CASE
  - Using structures
  - Using a STRUCT to access data

Inline processing

- Performing process I/O using the INLINE facility
- Controlling the display of the process output
- Logging the process output to a variable debugger
- Lab Exercise (30 minutes):
  - Describe the syntax required to write INLINE functions in general
  - Use the INLINE facility for interfacing with the PERUSE utility
  - Practice coding techniques using the variable editing built-ins
  - Review the usage of #INPUTV, #LOOP, and #IF
  - Describe the use of #INLINEPREFIX, INLPREFIX, #INLINETO, and INLTO
  - Write a macro-type function that purges jobs from the spooler and prompts the user for permission to purge each job

Writing functions - server files

- How the server file facility provides for communication between a TACL function and a process it has activated
- Situations in which it is appropriate to use implicit server files
- Writing functions that use implicit server files
- Lab Exercise (45 minutes):
  - Describe the syntax and usage of functions that employ implicit servers
  - Describe the usage of the RUN-options:
  - INV DYNAMIC PROMPT

Learn more at

hpe.com/us/training/nonstop

U4199S Ver J.00

c04706962 June 2015

Hewlett Packard Enterprise