TSO/E REXX Programming; Part I

Duration: 3-5 Days (Depending upon selected content, Part I alone is 3 days, but modules from Part II could be included with a total of 15 modules extending the duration to 5 days.)

Audience:

Delegates may be new to REXX, or have some exposure to REXX but need to underpin their experience by formal training.

Pre-requisites:

An understanding of computer concepts is assumed.

A working knowledge of TSO/ISPF is required. This can be gained from our z/OS TSO/ISPF Workshop.

Course Objectives

Each delegate will acquire a working knowledge of REXX in the z/OS TSO/E environment. Good programming practice is encouraged throughout. The course starts with the basics and furthers learning with ample hands-on exercises.

Course Content

Module 1: TSO/E REXX Environment

REXX Platforms
What is REXX?
Where is REXX code held?
How is REXX invoked?
REXX in batch; IKJEFT01 and IRXJCL

TSO/E REXX Programming; Part I

Module 2: Structure and Syntax

What constitutes REXX code?
Changing default environments; ADDRESS instruction
The coding rules described
Definition of variables
STEM variables and the DROP instruction
Operator characters; Arithmetic, Logical, Comparison and Concatenation
Operator order of precedence

Module 3: Diagnostic Aids

Checking logic flow; SAY instruction
Immediate commands; HE, HI, HT, RT, TE and TS
Trapping errors; CALL and SIGNAL instructions
Analyzing instruction execution; TRACE instruction and EXECUTIL command

Module 4: File Processing

File processing overview
Making data sets available; TSO ALLOC command
Disposing of data sets; TSO FREE command
Read or Write a data set; EXECIO command
I/O data areas; STEM and Data Stack
File Allocation using BPXWDYN

Module 5: Control Instructions

Basic decision making; IF and ELSE instructions
Case statements; SELECT, WHEN and OTHERWISE instructions
Iterative and conditional processing; DO instruction
Iterative processing logic
Bypassing iterative instructions; ITERATE instruction
Terminating iteration; LEAVE instruction

Module 6: Parsing

What is Parsing?
PARSE instruction general syntax
Read from keyboard or Data Stack; PARSE PULL
Read from a Variable; PARSE VAR
Process the results of an expression; PARSE VALUE
Handling excess data generated by PARSE
INTERPRET instruction

Module 7: Sub-routines and Functions

Naming internal sub-routines
Invoking internal sub-routines; CALL instruction
Hiding variables from a sub-routine; PROCEDURE instruction
Sharing hidden variables; EXPOSE instruction
Invoking external sub-routines; CALL instruction
Accessing passed data; PARSE ARG instruction and the ARG Built-in Function
Returning data to the caller; EXIT and RETURN instructions
The difference between a sub-routine and a Function

Module 8: REXX Built-in Functions

Conversion functions; C2D, C2X, D2C, D2X, X2B, X2C and X2D
Acquire date and time; DATE and TIME functions
Format a string; CENTRE, LEFT, RIGHT, SPACE, STRIP and TRANSLATE functions
Determine the size of a string; LENGTH, WORDLENGTH, and WORDS functions
Locate the position of something; FIND, POS, and WORD functions
Validating content; COMPARE, DATATYPE and VERIFY functions
Duplicate something; COPIES function
Shorten a string; DELSTR, DELWORD, SUBSTR and SUBWORD functions
Obtain session user identity; USERID function
Expand a string; INSERT function
Get highest/lowest value in a range; MAX and MIN functions
TSO/E functions; LISTDSI, MSG and PROMPT

Module 9: Data Stack Management

What is the Data Stack?
Add data to the stack; PUSH and QUEUE instructions
Read from the Data Stack; PARSE PULL
Stack management; DELSTACK, DROPBUF, NEWSTACK, QBUF, QELEM, and QSTACK Commands and the QUEUED Built-in Function
Some possible Data Stack uses