

Contents

Preface — V

1 Introduction — 1

- 1.1 Users of this book — 1
- 1.2 The significance of the REXX programming language in z/OS — 1
- 1.3 Brochures for programming ISPF applications — 1
 - 1.3.1 The REXX programming literature — 2
 - 1.3.2 ISPF reference books — 3
 - 1.3.3 TSO reference books — 5
 - 1.3.4 The ISPF services — 5

2 Introduction to the REXX programming language — 7

- 2.1 What is REXX? — 7
- 2.2 Overview of REXX under TSO — 8
 - 2.2.1 Recognizing a REXX procedure by the TSO — 8
 - 2.2.2 Running REXX procedures in the TSO — 9
- 2.3 Compile REXX procedures — 11
- 2.4 Performance of REXX procedures — 11
- 2.5 The syntax of the REXX language — 12
- 2.6 Variables in REXX — 14
- 2.7 Data types of REXX variables — 15
- 2.8 Operators of the REXX language — 15
 - 2.8.1 String operators — 16
 - 2.8.2 Arithmetic Operators — 16
 - 2.8.3 Compare operators — 17
 - 2.8.4 Logical operators — 19
- 2.9 Stems in REXX — 19
 - 2.9.1 Initialize stems with null string — 21
 - 2.9.2 File processing in connection with stems — 22
 - 2.9.3 Multi-unit Stems — 22

3 The REXX commands — 25

- 3.1 ADDRESS – Connection to the subsystems — 25
 - 3.1.1 The host command environments — 25
 - 3.1.2 The active host command environment — 27
 - 3.1.3 The temporary addressing of external commands — 27
 - 3.1.4 Special case ISPEXEC and ISREDIT — 29
 - 3.1.5 The main host command environments — 30
- 3.2 ARG – Retrieve the parameter string — 31

3.3	CALL – Call other programs —	35
3.4	DO groups and DO loops —	36
3.4.1	The DO group —	36
3.4.2	The DO loop —	37
3.5	EXIT and RETURN – leaving the REXX procedure —	39
3.6	IF and WHEN – check conditions —	41
3.7	INTERPRET – generate REXX commands dynamically —	42
3.8	NUMERIC – set computing options —	44
3.9	PARSE – text fragmentation —	46
3.9.1	PARSE ARG —	48
3.9.2	PARSE VALUE WITH —	48
3.9.3	PARSE VAR —	51
3.9.4	PARSE SOURCE —	51
3.10	PROCEDURE – Option for internal subroutines —	52
3.10.1	Internal subroutines with PROCEDURE statement —	52
3.10.2	Internal subroutines without PROCEDURE statement —	53
3.11	QUEUE – Working with the TSO stack —	53
3.11.1	The TSO/E data stack —	53
3.11.2	Use options for the TSO/E Data Stack —	54
3.12	SAY – Print texts —	57
3.13	SELECT – Conditionally call alternative instructions —	57
3.14	NOP – No operation —	60
3.15	PULL – Enter data on the screen —	60
3.16	TRACE – The strong debugging aid —	61
3.17	SIGNAL – Jumping when errors —	62
4	The REXX functions —	63
4.1	General functions —	64
4.1.1	ADDRESS – Get the active host command environment —	64
4.1.2	ARG – Input parameter test or take —	65
4.1.3	DATE – Date functions —	65
4.1.4	TIME – Time functions —	68
4.1.5	QUEUED – Number of records in the data stack —	69
4.1.6	SOURCELINE – Return a program line —	69
4.1.7	USERID – Return the TSO user ID —	70
4.1.8	VALUE – Create variable names dynamically —	70
4.2	Arithmetic functions —	72
4.2.1	ABS – absolute value of a number —	72
4.2.2	DIGITS, FORM, FUZZ – Query options for arithmetic operations —	72
4.2.3	MIN, MAX – Minimum and maximum value —	73
4.2.4	RANDOM – Generate random numbers —	73
4.2.5	SIGN – Return of the sign —	74

4.3	Comparison Functions — 74
4.3.1	COMPARE – Compare texts — 74
4.3.2	DATATYPE – Determine data type — 75
4.4	Conversion functions — 77
4.4.1	C2D – Character to decimal — 77
4.4.2	C2X – Character to Hexadecimal — 79
4.4.3	D2C – Decimal to Character — 79
4.4.4	D2X – Decimal to Hexadecimal — 80
4.4.5	X2B – Hexadecimal to Binary — 80
4.4.6	X2C – Hexadecimal to Character — 81
4.4.7	X2D – Hexadecimal to Decimal — 81
4.5	Formatting functions — 82
4.5.1	CENTER – Centering a string — 82
4.5.2	COPIES – Reproduce texts — 83
4.5.3	FORMAT – Format numbers — 83
4.5.4	JUSTIFY – Formatting a string — 83
4.5.5	LEFT – Rearrange text left-justified — 84
4.5.6	RIGHT – Arrange text right-justified — 84
4.6	String functions — 85
4.6.1	DELSTR – Delete substrings — 85
4.6.2	INSERT – Insert text — 85
4.6.3	SCHANGE – Change of texts — 86
4.6.4	LENGTH – Length of a text — 86
4.6.5	OVERLAY – Superimpose text — 87
4.6.6	POS – Search for text — 87
4.6.7	STRIP – remove border characters — 87
4.6.8	SUBSTR – Extract part of a text — 88
4.6.9	TRANSLATE – translate characters — 89
4.6.10	VERIFY – verify text — 90
4.7	Word functions — 90
4.7.1	WORD – return of a word — 90
4.7.2	WORDINDEX – return the starting position of a word — 91
4.7.3	WORDLENGTH – return the length of a word — 91
4.7.4	WORDPOS – search for a word — 91
4.7.5	WORDS – number of words in a string — 92
4.8	System functions — 92
4.8.1	LISTDSI – List data set information — 93
4.8.2	MSG – control of the TSO messages — 93
4.8.3	MVSVAR – Return z/OS system information — 94
4.8.4	OUTTRAP – take TSO messages — 94
4.8.5	SYSDSN – check data set status — 96

4.8.6	SYSVAR – get system Information —	96
4.8.7	STORAGE – read and write memory contents —	98
5	The TSO/E REXX commands —	101
5.1	EXECIO – Read and write data sets —	101
5.2	DELSTACK – delete data stack contents —	103
5.3	DROPBUF – Delete data stack buffers —	103
5.4	TSO Commands —	103
6	Execute REXX programs —	105
6.1	Execution of programs in a TSO/ISPF environment —	105
6.1.1	Online execution —	106
6.1.2	Batch execution —	107
7	Introduction to ISPF programming —	109
7.1	Programming languages useable in ISPF —	109
7.2	ISPF programming objects —	109
7.3	Some typical examples —	111
7.3.1	Example for use of ISPF panels —	111
7.3.2	Example for use of skeletons —	113
7.3.3	Example for use of tables —	115
7.3.4	Example for use of ISPF variables —	116
7.3.5	Example for data processing with TSO and ISPF —	118
7.3.6	Output of messages with ISPF —	121
7.4	LIBDEF – Dynamic linking of ISPF libraries —	123
7.5	ALTLIB – Dynamic linking of EXEC libraries —	126
7.5.1	Search sequence in the procedures libraries —	126
7.5.2	The ALTLIB command in ISPF —	128
7.5.3	Stacking of the APPLICATION level ALTLIBs —	129
7.5.4	The QUIET operand of the ALTLIB DISPLAY command —	130
8	Data set processing using ISPF —	133
8.1	The LM services —	133
8.1.1	Grouping of LM services —	133
8.1.2	LMINIT – Start of the data set processing —	136
8.1.3	LMFREE – Free a data set —	137
8.1.4	LMOPEN – Open a data set for processing —	137
8.1.5	LMCLOSE – Close a data set —	138
8.1.6	LMMFIND – Localize a member —	139
8.1.7	LMMREP – Replace a member —	140
8.1.8	LMMADD – Add a member —	141

8.1.9	LMGET – Read a data records —	142
8.1.10	LMPUT – Output data records —	143
8.1.11	LMCOPY – Copy data —	145
8.1.12	LMMOVE – Move data —	146
8.1.13	LMMDEL – Delete members —	146
8.1.14	LMMREN – Rename members —	147
8.1.15	LMMSTATS – Display or change member statistics —	147
8.1.16	LMCOMP – Compress a PDS —	150
8.1.17	LMMLIST – Display a member list —	150
8.1.18	LMMDISP – Display and edit a member list —	153
8.1.19	LMDDISP – Initialize the LMDDISP service —	155
8.1.20	LMDDISP – Data set list service —	155
8.1.21	LMDLIST – List of data sets —	157
8.1.22	LMDFREE – Free a LISTID —	158
8.1.23	MEMLIST – member list dialog service —	158
8.2	Data set query services —	160
8.2.1	LMDLIST – Data set list service —	161
8.2.2	DSINFO – ISPF service which provides data set information —	162
8.2.3	LISTDSI – REXX function to list data set information —	164
8.2.4	QBASELIB – Query DSN information —	169
8.2.5	QLIBDEF – Query LIBDEF Information —	171
8.2.6	QUERYENQ – ENQs determination —	173
9	Messages – Definition, setting, output —	177
9.1	Error handling in ISPF —	177
9.1.1	Returning error messages —	178
9.1.2	Output of error messages —	178
9.1.3	SETMSG – Set next message —	180
9.1.4	Definition of ISPF messages —	180
9.1.5	Naming convention of the ISPF message IDs —	181
9.1.6	Definition of messages —	181
9.1.7	The standard message member ISRZ00 —	184
10	Panels – create and use —	189
10.1	The Dynamic Tag Language (DTL) —	189
10.2	Panel types in ISPF —	190
10.3	Definition of panels —	190
10.3.1	The structure of a panel —	191
10.3.2	Creation of panels and their call —	192
10.3.3	The panel definition sections —	192
10.3.4	Variables in panel definitions —	201

10.3.5	Panel processing —	202
10.3.6	Help panels —	204
10.3.7	Panels to display ISPF tables —	205
11	Skeletons – Design and use —	211
11.1	Creating skeletons —	211
11.2	Steps to use the file-tailoring service —	214
12	Tables – Create and edit —	215
12.1	Locations for tables —	215
12.2	Reading ISPF tables —	215
12.3	Writing ISPF tables —	215
12.4	Commands of the table services —	216
12.5	Example of working with tables —	218
13	Variables – Definition and using —	223
13.1	Variables pools —	223
13.1.1	Function pool —	223
13.1.2	Shared pool —	224
13.1.3	Profile pool —	224
13.2	Saving the profile members —	225
13.3	Creating profile members —	225
13.4	Display of actually used profile member —	226
13.5	The System Profile Pool —	227
13.5.1	Frequently used ISPF variables —	228
13.5.2	Process ISPF variables —	229
14	Edit macros – Create and apply —	233
14.1	What is an edit macro? —	233
14.1.1	Naming conventions for edit macros —	233
14.1.2	Example of two very useful edit macros —	234
14.2	Table of edit macro commands —	241
14.3	Operands and abbreviations used in the edit macro commands —	247
14.4	Test aids in the creating macros —	248
14.4.1	Prototyping —	248
14.4.2	The REXX TRACE command as developing aid —	248
14.4.3	The program ISREMSPY —	249
14.5	System variables of the ISPF editor —	251
14.6	Passing parameters to macros —	252
14.7	Examples for editing and SUBMIT batch jobs —	255
14.8	Mnemonics for macro programming —	255

15	The SMART ISPF utilities — 257
15.1	Naming conventions — 257
15.2	The dynamic panel concept — 257
15.3	List of executable programs — 258
15.4	Program descriptions — 260
15.4.1	Edit macro ## – Execute a currently edited REXX procedure — 260
15.4.2	Edit macro #ALTXT – Realign line parts — 260
15.4.3	Edit macro #EDMEM – Edit of a member — 260
15.4.4	Edit macro #IMACROA – General ISPF edit macro — 261
15.4.5	Edit macro #IMACRO1 – Initial edit macro — 261
15.4.6	Edit macro #IMACRO2 – Edit session end macro — 262
15.4.7	Edit macro #ISPFB – Submit an ISPF batch job — 263
15.4.8	Edit macro #LCH – Perform long edit change commands — 263
15.4.9	Edit macro #SPLJ – SPLIT and JOIN lines — 263
15.4.10	Edit macro #SSS – Clear SCHFOR lists — 264
15.4.11	Edit macro #SSSCH – Mass update of members — 264
15.4.12	Edit macro #SU – Submit JCL without a JOB statement — 264
15.4.13	Edit macro #TSOB – Submit a TSO batch job — 265
15.4.14	Edit macro #VERASE – Erase ISPF profile variables — 266
15.4.15	Program SCURSOR – Calling a data set from an ISPF screen — 266
15.4.16	Program SDOC – Produce documentation members — 267
15.4.17	Program SLE – Display last edited data sets — 267
15.4.18	SLOGDSN – Data member containing DSNs for logon — 269
15.4.19	Program SLOGON – Personal logon procedure — 270
15.4.20	Program SPROFEDT – Store users ISPF profile variables — 270
15.4.21	SPROFVAR – Load user ISPF variables — 270
15.4.22	Program SSC – Super clone for data sets — 270
15.4.23	Program SSS – Perform a Super-Search — 271
15.5	Programming aids — 273
15.5.1	Edit macro #C – Compile and execute a REXX program — 274
15.5.2	Edit macro #RO – Online compile of a REXX program — 274
15.5.3	Edit macro #RC – Compile a REXX procedure with a batch job — 275
15.5.4	Edit macro #RCLOAD – Produce a load or a call module — 276
15.5.5	Edit macro #IE – Insert a call to ISPF_ERROR — 277
15.5.6	Subroutine ISPF_ERROR – Display ISPF error messages — 278
15.5.7	Edit macro #PAN – Execute a panel source code — 279
15.5.8	Subroutine DAYDIFF – Calculate number of days — 280
15.5.9	Subroutine ENDDAY – Calculate a target date — 281
15.5.10	Subroutine JULDATE – Translate a date to Julian and vise versa — 282
15.5.11	Subroutine LEAPYEAR – Return the leap year information — 283
15.5.12	REXX subroutine SCHANGE – REXX change function — 283

15.5.13	Program SDYNPAN – Convert a panel source code —	284
15.6	Installation of SMART ISPF utilities —	284
15.6.1	Download and unzip —	285
15.6.2	Installation —	286
15.6.3	ALTLIB command —	286
15.6.4	Make the SMART ISPF utilities ready to run —	289

List of programs —	291
--------------------	------------

List of tables —	292
------------------	------------

List of screens —	293
-------------------	------------

Bibliography —	295
----------------	------------

Index —	297
---------	------------