



BMC Mainframe: Advanced Assembler and Macro Usage

COURSE ABSTRACT

COURSE CODE

- » MGRS-AAMU-2021

APPLICABLE VERSIONS

- » Not Applicable

DELIVERY METHOD

- » Instructor-led Training (ILT)

COURSE DURATION

- » 4 Days

PREREQUISITES

- » BMC Mainframe: Using z/OS Assembler

RECOMMENDED TRAININGS

- » BMC Mainframe: z/OS System Anatomy Part 1 - z Architecture
- » BMC Mainframe: z/OS System Anatomy Part 2 - z/OS Infrastructure & Services

Course Overview

The course is developed and delivered by © RSM Technology.

This four-day course is designed to build on the skills taught in RSM's definitive Using z/OS Assembler course. It starts with a short review of the basic Assembler instructions before introducing a number of more advanced subjects, such as general addressability, 31-bits and 64-bits addressing, re-entrancy, macro usage, debugging and recovery.

The course also covers the Assembler macro language fundamentals and z/OS Control Block usage - all essential knowledge required for maintaining and writing z/OS exits.

This course is also available for exclusive one-company presentations and for live presentation over the Internet, via the Virtual Classroom Environment service.

Target Audience

- » Programmers
- » Administrators

Learner Objectives

- » Use advanced addressability techniques
- » Use data management macros
- » Receive and pass subroutine parameters and process return codes
- » Utilize the linkage stack
- » Apply 31-bits & 64-bits addressing concepts
- » Write re-entrant code
- » Use the basic Assembler macro language
- » Code Binder control statements
- » Use MVS control blocks in exit routines
- » Use supervisor services macros
- » Debug dumps
- » Write ESTAE routines
- » Write re-entrant programs

BMC, BMC Software, and the BMC Software logo are the exclusive properties of BMC Software, Inc., are registered with the U.S. Patent and Trademark Office, and may be registered or pending registration in other countries. All other BMC trademarks, service marks, and logos may be registered or pending registration in the U.S. or in other countries. All other trademarks or registered trademarks are the property of their respective owners. ©2021 BMC Software, Inc. All rights reserved.

For more information about BMC Education Services, visit www.bmc.com/education.





BMC Mainframe: Advanced Assembler and Macro Usage

COURSE ABSTRACT

COURSE ACTIVITIES

- » Classroom Presentations
- » Demonstration


BMC MAINFRAME INFRASTRUCTURE AND PLATFORMS LEARNING PATH

- » <https://www.bmc.com/education/courses/find-courses.html#filter/%7B%22type%22%3A%22edu-specific-types-159150236%22%7D>

CERTIFICATION PATHS

- » This course is not part of a BMC Certification Path

DISCOUNT OPTIONS

- » Have multiple students? Contact us to discuss hosting a private class for your organization
- » [Contact us for additional information](#) 

Course Modules

Addressability

- » Control Sections - CSECTS
- » CSECT addressability - methods and examples
- » DROP directive
- » CSECTS in excess of 4K
- » Multiple Base Registers
- » Reference by explicit address
- » Reference by Dummy Section
- » DSECT mapping and external storage
- » MVS control block DSECTS
- » Explicit reference and internal control blocks
- » DSECT reference and internal control blocks
- » DSECT reference and external control blocks

I/O and Data Management

- » Sequential files:
 - Fixed Length Records, Variable Length Record
 - Data Control Block (DCB)
 - QSAM DCB
 - OPEN macro

- CLOSE macro
- Using the GET macro in Move mode
- Using the PUT macro in Move mode
- Using the GET macro in Locate mode
- Using the PUT macro in Locate mode
- Execute

Intermodule Communication

- » Static Load Module structure and the VCON
- » Static Load Module linkage - 'CALL' macro
- » Secondary entry points
- » Establishing addressability for secondary entry points
- » External references using EXTRN
- » Save Area conventions
- » Standard Module linkage
- » SAVE and RETURN macros
- » Parameter passing standard convention
- » PARM keyword on JCL EXEC statement
- » Parameter passing using Explicit Interface
- » CALL macro:
 - Explicit Parameter Interface
 - Implicit Parameter Interface
 - Implicit Variable Parameter Interface

- Return code setting
- Return code testing
- Manipulating the Linkage Stack with BAKR and PR
- Status saving using the Linkage Stack

Extended Addressability

- » Virtual Storage
- » Establishing AMODE and RMODE values
- » AMODE and RMODE combination rules
- » Addressing mode sensitivity
- » Mode setting instructions - BASSM and BSM
- » Calling and returning using BASSM and BSM (Basic Mode)
- » Mode switching to retrieve data from above 16 MB
- » Using pointer-defined linkage
- » BASSM & BSM in 64-bit mode
- » Using linkage assist routines
- » Using capping to assist linkage to 24-bit program
- » Using capping to assist linkage to 31-bit program

BMC, BMC Software, and the BMC Software logo are the exclusive properties of BMC Software, Inc., are registered with the U.S. Patent and Trademark Office, and may be registered or pending registration in other countries. All other BMC trademarks, service marks, and logos may be registered or pending registration in the U.S. or in other countries. All other trademarks or registered trademarks are the property of their respective owners. ©2021 BMC Software, Inc. All rights reserved.



BMC Mainframe: Advanced Assembler and Macro Usage

COURSE ABSTRACT

Binder (Linkage editor)

- » Binder overview
- » Resolving external references
- » Object and LOAD module records
- » External Symbol Dictionary
- » ReLocation Dictionary
- » Automatic library call feature
- » Weak external reference
- » Binder JCL
- » Binder module attribute options
- » Binder module processing and output options
- » Binder control statements
- » Binder codes
- » AMBLIST utility

Debugging

- » Problem program dump facilities
- » ABEND
- » SNAP
- » ABEND dump files
- » ABEND dump default contents
- » Summary dump
- » Symptom dump
- » ABEND macro
- » Parameter descriptions
- » SNAP macro
- » SNAP SDATA options
- » SNAP PDATA options

- » Save Area traces
- » Save Area trace and linkage stack entry

System Services

- » Event synchronization
- » WAIT macro
- » POST macro
- » Write to log macro (WTL)
- » Write to operator macro (WTO)
- » Write to operator with reply macro (WTOR)
- » Delete operator message macro (DOM)
- » TIME macro
- » STIMER macro
- » TTIMER macro
- » CPUTIMER macro

Recovery

- » Recovery flow
- » ESTAE macro
- » System Diagnostic Work Area (SDWA)
- » Recovery routine processing
- » SETRP macro
- » Retry options
- » Lab exercise (write an ESTAE recovery routine)

Advanced Instructions

- » Move long (MVCL)
- » Compare logical long (CLCL)
- » Translate (TR)

- » Translate & test (TRT)
- » Edit (ED)
- » Edit & mark (EDMK)

Macro Language Fundamentals

- » Macro language introduction
- » Macro language feature
- » Defining a macro - basic rules
- » Passing positional parameters
- » Passing keyword parameters
- » Parameter and keyword sub lists
- » Common system variables
- » Testing how a field was defined
- » Lab exercise

Task Management

- » LINK
- » LOAD ATTACH
- » XCTL

Re-entrant & Re-usable Code

- » Re-entrant and re-usable code - objectives
- » Re-usable code (REUS)
- » Re-entrant code (RENT) - the rules
- » RENT - how do you do it?
- » Re-entrant code example
- » List and Execute forms of macros
- » Getting the re-entrant benefit
- » Refreshable
- » Lab exercise (make a routine re-entrant)