CL – Control Language

Vladimír Župka 2025

Obsah

Příkazy řídicího jazyka CL

Řídicí jazyk operačního systému se nazývá CL čili Control Language, někdy také Command Language, a je tvořen příkazy (povely). Každý příkaz se skládá ze *jména* a *parametrů*. Počet a náplň parametrů závisí na druhu příkazu. Některé parametry nemají žádné parametry. Příkazy jsou objekty typu ***CMD**.

Popis jazyka najdeme na stránce https://www.ibm.com/docs/en/i/7.5?topic=programming-control-language.

Platí obecné pravidlo (z něhož se vymykají některé případy), že ve jméně příkazu první tři písmena jsou zkratkou anglického slovesa vyjadřujícího činnost, kterou příkaz od operačního systému vyžaduje. Další část zkratkovitě vyjadřuje předmět, s nímž systém má pracovat. Například

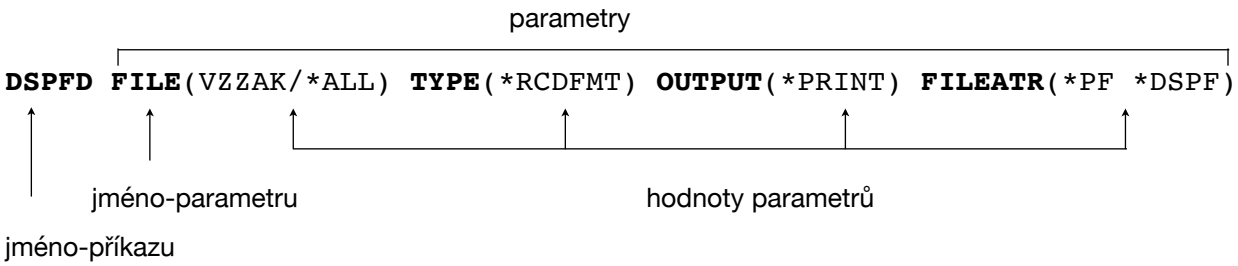
DSPJOBLOG	**DiSP**lay **JOB LOG** - zobrazit "job log" tj. záznam o průběhu úlohy
WRKSPLF	**WoRK** with **SP**ool **F**iles - pracovat s tiskovými soubory
CRTLIB	**CR**ea**T**e **LIB**rary - vytvořit knihovnu

Některé slovesné zkratky

ADD, RMV	Add - přidat, Remove – odstranit
CALL	Call - volat (program)
CHG	Change - změnit
CHK	Check - zkontrolovat
CLR	Clear - vyčistit
CPY	Copy - kopírovat
CRT, DLT	Create - vytvořit, Delete - zrušit
DCL	Declare - deklarovat (proměnnou, soubor) v CL-programu
DSP	Display - zobrazit
DUP	Duplicate - duplikovat
EDT	Edit - upravit, editovat
END, STR	End - ukončit, Start - zahájit
GO	Go - jít (na nabídku - menu)
GRT, RVK	Grant - zaručit, Revoke - odebrat
INZ	Initialize - inicializovat (pásku, disketu, fyzický soubor)
MON	Monitor - monitorovat (zprávy)
MOV	Move - přesunout (objekt)
OPN, CLO	Open - otevřít, Close - uzavřít
OVR	Override - předefinovat, přesměrovat
PRT	Print - tisknout
RCV, SND	Receive - obdržet (zprávu), Send - poslat
RGZ	Reorganize - reorganizovat (data souboru)
RNM	Rename - přejmenovat
RST, SAV	Restore - obnovit, Save - uložit (záložní objekty)
RTV	Retrieve - získat, přečíst (údaje do proměnných)
RUN	Run - spustit
QRY	Query - dotázat se
SBM	Submit - podřídit, předat k dalšímu zpracování
VRY	Vary - změnit vlastnost (zapnout, vypnout)
WRK	Work with - pracovat s (objekty, činnostmi, stavy)

Zápis CL příkazu

Příklad zápisu



Základní tvar CL příkazu

CL příkaz se skládá ze jména a parametrů. Parametry, jsou-li zadány, jsou od jména a mezi sebou odděleny aspoň jednou mezerou.

```
jméno-příkazu  [parametr  [parametr  ...]]
```

Parametry jsou tvořeny jménem a hodnotou v závorkách. Levá závorka musí těsně navazovat na jméno parametru. Má-li parametr více hodnot, jsou mezi sebou odděleny aspoň jednou mezerou.

```
jméno-parametru(hodnota  [hodnota ...])
```

Zápis v programu

V programu se před příkaz mohou doplnit ještě návěstí nebo symbol náznaku – otazník a před parametry symboly výběrového náznaku (viz dále).

```
[návěstí:][?][knihovna/]příkaz
[výběrový-náznak]parametr
```

Zápis v příkazovém řádku

V příkazovém řádku se používá jednodušší zápis.

```
[knihovna/]příkaz
```

Způsoby zápisu parametrů

Příkaz i parametry lze zapisovat velkými i malými písmeny, která lze střídat.

```
DSPFD file(vzzak/*all) type(*rcdfmt) output(*Print) FileAtr(*pf *dspf)
```

Jména parametrů i závorky je možné vynechat, ale jen v maximálním počtu prvních pozičních parametrů. Tento počet se zjistí spuštěním příkazu DSPCMD (Display Command Description). U příkazu DSPFD jsou 3, takže poslední parametr musí být pojmenován.

```
dspfd VZZAK/*All (*RcdFmt) *print FILEATR(*pf *dspf)
```

Jakmile zapíšeme parametr se jménem, další parametry musíme zapsat také se jménem, třeba v libovolném pořadí.

```
dspfd vzzak/*all fileatr(*PF *DSPF) type(*RcdFmt) output(*print)
```

Kromě povinných parametrů nemusíme zapisovat žádné další. Jejich hodnoty jsou předem zvolené (default).

```
DSPFD VZZAK/CENY
```

Získání přehledu o parametrech

Příkaz DSPCMD (Display Command Description)

Prvním příkladem k získání přehledu o parametrech je jednoduchý příkaz DSPCMD (Display Command Description) s pouhými třemi parametry zadaný z příkazového řádku. Klávesa **F4** má stejný účinek jako **otazník** před příkazem, který specifikuje jen první parametr.

```
DSPCMD CMD(DSPFFD)
```

Po stisku **F4** s kurzorem na příkazu vidíme jednu zapsanou hodnotu `DSPFFD` a předvolené hodnoty `*LIBL` a `*` s volbou možných hodnot parametrů. Zapsaná hodnota má před sebou znaménko >. Všechny jsou podtrženy na znamení toho, že do nich lze zapisovat.

```
                          Display Command (DSPCMD)

 Type choices, press Enter.

 Label  . . . . . .
 Command  . . . . . > DSPFFD_____   Name
   Library  . . . . .   *LIBL_____ Name, *LIBL, *CURLIB
 Output . . . . . .   *_____   *, *PRINT
```

Po stisku **F11** se ukážou jména parametrů s hodnotami.

```
Label  . . . . . .
Command  . . . . . CMD          > DSPFFD_____
  Library  . . . . .                *LIBL_____
Output . . . . . . OUTPUT           *_____
```

S kurzorem na hodnotě *LIBL stiskneme **F4** a dostaneme náznak možných hodnot parametru.

```
                  Specify Value for Parameter CMD

 Type choice, press Enter.

 Type . . . . . :   NAME
 Library  . . . . .   *LIBL

   *LIBL
   *CURLIB
```

Podrobný popis parametru získáme stiskem **F1** a následným stránkováním.

```
Type . . . . . :   NAME
Library  . . . . . :   *LIBL
                 .....................................................
  *LIBL          :                  Command (CMD) - Help             :
  *CURLIB        :                                                   :
                 :  Specifies the command whose information is to be  :
                 :  displayed.                                        :
                 :                                                   :
                 :  This is a required parameter.                    :
                 :                                                   :
                 :  Qualifier 1: Command                             :
                 :                                                   :
                 :  name                                             :
                 :      Specify the name of the command to be shown. :
                 :                                                   :
                 :  Qualifier 2: Library                             :
                 :                                                   :
                 :  *LIBL                                            :
                 :      All libraries in the library list for the current :
                 :      thread are searched until the first match is found. :
                 :                                                   :
                 :  *CURLIB                                          :
                 :      The current library for the job is used to locate the :
                 :      command.  If no library is specified as the current :
                 :      library for the job, QGPL is used.           :
                 :                                          More...  :
                 :  name                                             :
                 :      Specify the name of the library that contains the :
                 :      command.                                     :
                 :                                                   :
```

Když ten příkaz spustíme klávesou Enter, na obrazovce se objeví informace o příkazu DSPFFD.

```
                        Display Command Information

Command  . . . . . :   DSPFFD        Library  . . . . . :   QSYS

Program to process command . . . . . :   QWHDSFFD
  Library  . . . . . . . . . . . . . :     QSYS
  State used to call program . . . . . :     *SYSTEM
Source file  . . . . . . . . . . . . . :   S000041244
  Library  . . . . . . . . . . . . . . :     QTEMP
Source file member . . . . . . . . . . :   DSPFFD
Validity checking program  . . . . . :   *NONE
Mode(s) in which valid . . . . . . . :   *PROD
                                         *DEBUG
                                         *SERVICE
Where allowed to run . . . . . . . . :   *IMOD      *BMOD      *IREXX
                                         *BREXX     *BPGM      *IPGM
                                         *EXEC      *INTERACT  *BATCH
Allow limited user . . . . . . . . . :   *NO
Maximum positional parameters  . . . . :   2
                                                                More...
```

Příkaz DSPFD (Display File Description)

Dalším příkladem k získání přehledu o parametrech je příkaz DSPFD (Display File Description).

```
DSPFD FILE(VZZAK/*ALL) TYPE(*RCDFMT) OUTPUT(*PRINT) FILEATR(*PF *DSPF)
```

Hodnoty parametrů jsou zapsány v závorkách. Levá závorka musí následovat těsně za jménem parametru. Hodnota může být jedna, např. `(*PRINT)` nebo jich může být několik v seznamu oddělených mezerami, např. `(*PF *DSPF)`.

V editačním režimu na příkaz `DSPFD` umístíme kurzor a stiskneme **F4**. Dostaneme přehled parametrů s vysvětlivkami možných hodnot.

```
File . . . . . > *ALL          Name, generic*, *ALL
  Library  . . . . . >   VZZAK      Name, *LIBL, *CURLIB...
Type of information  . . . . . .   *RCDFMT      *ALL, *BASATR, *ATR...
              + for more values    _________
Output . . . . . > *PRINT      *, *PRINT, *OUTFILE
File attributes  . . . . . > *PF          *ALL, *DSPF, *PRTF, *DKTF...
              + for more values > *DSPF
```

Když pak stiskneme Enter, editor jej rozmístí do standardního tvaru. Znaménko + znamená pokračování na dalším řádku bez začátečních mezer a přidává se, když je příkaz příliš dlouhý.

```
            DSPFD      FILE(VZZAK/*ALL) TYPE(*RCDFMT) +
                         OUTPUT(*PRINT) FILEATR(*PF *DSPF)
```

Po stisku **F9** se ukážou všechny parametry příkazu. Stisk **F11** zobrazí jména parametrů, ale bez náznaku možných hodnot.

```
File to receive output . . . . . OUTFILE          __________
  Library  . . . . . .                              *LIBL
Output member options:           OUTMBR
  Member to receive output . . .                   *FIRST
  Replace or add records . . . .                   *REPLACE

                        Additional Parameters

System . . . . . . . . . . . . . SYSTEM          *LCL
```

Pole pro hodnoty parametrů jsou podtržena, což znamená, že do nich lze zapisovat. Ne všechny kombinace hodnot jsou povoleny.

Náznak možných hodnot parametru získáme pomocí **F4** s kurzorem umístěným v řádku zvoleného parametru:

```
Output . . . . . . . . . . . . . > *PRINT

  *
  *PRINT
  *OUTFILE
```

Podrobný popis parametru získáme stiskem **F1** a následným stránkováním.

```
Output . . . . . . . . . . . . . > *PRINT
              .....................................................................
  *           :                    Output (OUTPUT) - Help                         :
  *PRINT      :                                                                   :
  *OUTFILE    :  Specifies where the output from the command is sent.            :
              :                                                                   :
              :  *                                                               :
              :      The output is displayed (if requested by an                 :
              :      interactive job) or printed with the job's spooled          :
              :      output (if requested by a batch job).                       :
              :                                                                   :
              :  *PRINT                                                          :
              :      The output is printed with the job's spooled output.        :
              :                                                                   :
              :                                                          More...  :
              :  F2=Extended help   F10=Move to top        F12=Cancel             :
F3=Exit   F5= :  F13=Information Assistant   F20=Enlarge   F24=More keys         :
              :                                                                   :
              :...................................................................:
```

Vrátíme-li se dvojím stiskem klávesy F12, můžeme pak příkaz spustit stiskem klávesy Enter. Výsledek příkazu je zapsán v tiskovém souboru QPDSPFD. Najdeme ho pomocí příkazu WRKSPLF (Work With Spooled Files).

Příkazy a funkce používané v CL programu

`PGM`
Uvádí zdrojový program a určuje vstupní parametry. Není povinný u programu bez parametrů.

`DCL, DCLF`
Deklarace proměnných a souborů. Deklarační příkazy musí předcházet ostatním kromě PGM.

`CHGVAR`
Dosazuje do proměnné hodnotu výrazu.

`%SUBSTRING (%SST), %SWITCH, %BINARY (%BIN), %ADDRESS (%ADDR), %OFFSET (%OFS), %CHECK, %CHECKR, %SCAN, %TRIM, %TRIML, %TRIMR, %CHAR, %DEC, %INT, %UINT (%UNS), %LEN, %SIZE, %LOWER, %UPPER`
Funkce používané v aritmetických, znakových, relačních a logických výrazech.

`SNDPGMMSG, RCVMSG, MONMSG, RCVF, OVRDBF, RUNQRY, CLRPFM, CLOF, ...`
Je to většina CL příkazů z operačního systému. Přehled získáme příkazem GO MAJOR z příkazového řádku.

`IF, ELSE, DO, DOWHILE, DOUNTIL, DOFOR, ENDDO, LEAVE, ITERATE, GOTO, SELECT, WHEN, OTHERWISE, ENDSELECT, CALLSUBR, SUBR, RTNSUBR, ENDSUBR`
Řízení výpočtu v programu.

`CALL, RETURN, CALLPRC, TFRCTL`
Předání výpočtu jiným programům nebo procedurám.

`INCLUDE`
Zařazuje do programu další zdrojové příkazy z jiného zdrojového souboru.

`ENDPGM`
Nepovinný příkaz uzavírá zdrojový program.

Poznámka: Podrobné popisy CL příkazů jsou v dokumentaci https://www.ibm.com/docs/en/i/7.5?topic=language-cl-command-finder.

Náznak (prompt) pro příkaz v CL programu

Vyvolání náznaku se označuje otazníkem před jménem příkazu.

Program PROMPT1

```
PGM
? DSPLIB
ENDPGM
```

Program zkompilujeme volbou **14** v PDM nebo příkazem

```
CRTBNDCL PGM(*CURLIB/PROMPT1) SRCFILE(*LIBL/QCLSRC) SRCMBR(*PGM)
```

a spustíme volbou **C** v PDM nebo příkazem

```
CALL PGM(*LIBL/PROMPT1)
```

Výpočet se zastaví a zobrazí se následující náznak. Náznak je stejný, jako po zápisu příkazu DSPLIB na příkazový řádek a stisku klávesy **F4**:

```
Library  . . . . . . . . . . . .   *LIBL_____     Name, *LIBL, *USRLIBL...
               + for more values   __________
ASP device . . . . . . . . . . .   *__________     Name, *, *ALLAVL...
Output . . . . . . . . . . . . .   *_____         *, *PRINT
```

Náznak otazníkem není povolen v programu u následujících příkazů a nelze jej použít v dávkové úloze (batch job).

```
CALL CALLPRC CALLSUBR CHGVAR COPYRIGHT DCL DCLF DCLR DO DOFOR
DOUNTIL DOWHILE ELSE ENDDO ENDPGM ENDRCV ENDSELECT ENDSUBR GOTO
IF ITERATE LEAVE MONMSG OTHERWISE PGM RCVF RETURN RTNSUBR
SELECT SNDF SNDRCVF SUBR WAIT WHEN
```

Výběrové náznaky pro příkaz v CL programu

Výběrový náznak se označuje otazníkem a doplňkovým znakem před jménem parametru. Otazník musí být také před jménem příkazu.

Používá se pro

- výběr parametrů, které chceme zobrazit
- určení chráněných parametrů
- vynechání parametrů ze zobrazení

Dvojznaky zapisované těsně před klíčovým slovem parametru:

`??`	zobrazen, je vstupní
`?*`	zobrazen, bez vstupu, hodnoty jsou předány do CPP
`?<`	zobrazen, je vstupní, předvolená nebo změněná hodnota je předána do CPP
`?-`	nezobrazen, předvolená nebo změněná hodnota je předána do CPP
`?&`	nezobrazen, po F9 zobrazen jako vstupní
`?%`	nezobrazen, po F9 zobrazen bez vstupu, předvolená hodnota je předána do CPP

Program PROMPT2

```
PGM
 ?          DSPOBJD OBJ(*LIBL/CENY) OBJTYPE(*FILE) +
              DETAIL(*BASIC) ASPDEV(*) OUTPUT(*)
 ?          DSPOBJD ??OBJ(*LIBL/CENY) ?-OBJTYPE(*FILE) +
              ?*DETAIL(*BASIC) ?-ASPDEV(*) ?%OUTPUT()
ENDPGM
```

Po spuštění zkompilovaného programu se výpočet zastaví na prvním příkazu a zobrazí se následující náznak. V něm jsou všechny parametry bez možnosti vstupu (nepodtržené) a se zadanými hodnotami.

```
Object . . . . . . . . . . . . . OBJ          > CENY
  Library  . . . . . . . . . . .               >   *LIBL
Object type  . . . . . . . . . . OBJTYPE       > *FILE
Detail . . . . . . . . . . . . . DETAIL        > *BASIC
ASP device:                      ASPDEV
  Device . . . . . . . . . . . .               > *
  Search type  . . . . . . . . .
Output . . . . . . . . . . . . . OUTPUT        > *
```

Po stisku Enter se první příkaz provede.

Poté se výpočet zastaví na *druhém* příkazu a zobrazí další náznak. S výběrovým symbolem **??** prvním parametru jsou dvě hodnoty vstupní (podtržené) a zobrazují zadané hodnoty CENY a *LIBL. Třetí parametr se symbolem **?*** je zobrazen bez vstupu. Ostatní parametry se symbolem **?-** a **?%** nejsou zobrazeny.

```
Object . . . . . . . . . . . . . OBJ          > CENY
  Library  . . . . . . . . . . .               >   *LIBL
Detail . . . . . . . . . . . . . DETAIL        > *BASIC
```

Po stisku Enter se provede druhý příkaz.

Pro porovnání je zde uveden náznak stejného příkazu zapsaného na příkazovém řádku po stisku klávesy **F4**.

```
DSPOBJD OBJ(*LIBL/CENY) OBJTYPE(*FILE) DETAIL(*BASIC) ASPDEV(*) OUTPUT(*)
```

Náznak má všechny parametry vstupní a některé z nich s předvolenými hodnotami.

```
Object . . . . . . . . . . . . . OBJ          > CENY
  Library  . . . . . . . . . . .               >   *LIBL
Object type  . . . . . . . . . . OBJTYPE       > *FILE
                       + for more values
Detail . . . . . . . . . . . . . DETAIL        > *BASIC
ASP device:                      ASPDEV
  Device . . . . . . . . . . . .               > *
  Search type  . . . . . . . . .
Output . . . . . . . . . . . . . OUTPUT        > *
File to receive output . . . . . OUTFILE
  Library  . . . . . . . . . . .                 *LIBL
Output member options:           OUTMBR
  Member to receive output . . .                 *FIRST
  Replace or add records . . . .                 *REPLACE
```

Deklarace proměnných DCL

Plný popis příkazu DCL je uveden v dokumentaci: https://www.ibm.com/docs/en/i/7.3?topic=ssw_ibm_i_73/cl/dcl.html

Nejčastější tvar je

DCL VAR(*jméno-proměnné*) **TYPE**(*typ*) **LEN**(*délka*) **VALUE**(*počáteční-hodnota*)

Jméno proměnné je povinné. Začíná znakem **&**, dalších až 10 znaků může být alfanumerických. První z nich musí být písmeno.

Typ je povinný: `*CHAR`, `*DEC`, `*LGL`, `*INT`, `*UINT` nebo `*PTR`

Délka odpovídá typu dat. Není-li zadána, dosadí se

`32`	pro typ `*CHAR`
`1`	pro typ `*LGL`
`(15 5)`	pro typ `*DEC`
`4`	pro typ `*INT` a `*UINT`

Počáteční hodnota může a nemusí být zadána. Není-li zadána, dosadí se

mezery	pro typ `*CHAR`
`'0'`	pro typ `*LGL`
nula	pro typy `*DEC`, `*INT`, `*UINT`

Změna proměnných CHGVAR

Plný popis příkazu CHGVAR je uveden v dokumentaci: https://www.ibm.com/docs/en/i/7.5?topic=ssw_ibm_i_75/cl/chgvar.html.

Tvar příkazu je

CHGVAR VAR(*jméno-proměnné***) VALUE(***výraz***)**

Jméno proměnné v parametru VAR musí být definováno předem v příkazu DCL. Do proměnné se dosadí hodnota *výrazu* z parametru VALUE. Výrazy a příklady jsou popsány dále.

Typy dat v programu

***CHAR** znakové řetězce délky až 5000 znaků
***DEC** dekadická čísla s počtem číslic až 15 a desetinných míst až 9
***INT** celá binární čísla se znaménkem, délky 2, 4 nebo 8 bajtů
***UINT** celá binární čísla bez znaménka, délky 2, 4 nebo 8 bajtů
***LGL** logická data s hodnotami `'0'`, `'1'` (false, true)
***PTR** ukazatel na data (adresa dat)

```
DCL     &DECA        *DEC   (15 9)
DCL     &DECB        *DEC   5        /*  (5 0)    */
DCL     &INT4        *INT   4        /*  4 bajty  */
DCL     &INT2        *INT   2        /*  2 bajty  */
DCL     &UINT4       *UINT  4        /*  4 bajty  */

DCL     &CHAR_DEF    *CHAR           /*   32      */
DCL     &DEC_DEF     *DEC            /*  (15 5)   */
DCL     &LGL_DEF     *LGL
DCL     &LGL         *LGL   1  '1'

DCL     &CHAR1       *CHAR  15
DCL     &CHAR2       *CHAR  8

CHGVAR &DECA        -123456.789012345
CHGVAR &DECB        12345
CHGVAR &INT4        -2147483648     /*  min.  */
CHGVAR &INT2        -32768          /*  min.  */
CHGVAR &UINT4       4294967295      /*  max.  */

CHGVAR &DEC_DEF     1234567890.12345
CHGVAR &CHAR_DEF    'A'
CHGVAR &LGL_DEF     '1'
CHGVAR &LGL         '0'

CHGVAR &CHAR1       'Národní třída'
CHGVAR &CHAR2       Narodni         /* převede na velká písmena */

DMPCLPGM
```

Příkaz `DMPCLPGM` vytvoří tiskový soubor (spool file). V tiskovém souboru `QPPGMDMP` je výpis typů a hodnot v proměnných:

```
Variable     Type      Length       Value                          Value in Hexa
                                     *...+....1....+....2....+      * . . . + . .

&CHAR_DEF      *CHAR         32     'A                         '   C140404040404
                      +26           '       '                      4040404040404
&CHAR1         *CHAR         15     'Národní třída  '             D545999684955
&CHAR2         *CHAR          8     'NARODNI '                    D5C1D9D6C4D5C
&DEC_DEF       *DEC        15 5      1234567890.12345
&DECA          *DEC        15 9     -123456.789012345
&DECB          *DEC         5 0      12345
&INT2          *INT           2     -32768                        8000
&INT4          *INT           4     -2147483648                   80000000
&LGL           *CHAR          1     '0'                           F0
&LGL_DEF       *CHAR          1     '1'                           F1
&UINT4         *UINT          4      4294967295                   FFFFFFFF
```

Výrazy v příkazech CHGVAR, IF, DOWHILE, DOUNTIL, WHEN

Aritmetické výrazy

Operandy jsou konstanty nebo proměnné typu `*DEC, *INT, *UINT`

Operátory

`+  -  *  /`

Funkce **%BINARY, %CHECK, %CHECKR, %SCAN, %DEC, %INT, %UINT, %LEN, %SIZE, %PARMS**

Výsledek výrazu je typu `*DEC`

Znakové výrazy

Operandy jsou konstanty nebo proměnné typu `*CHAR`

Konstanty se obecně uvádějí v uvozovkách, jednoslovní konstanty smí být bez uvozovek.

Operátory

```
*CAT   *BCAT   *TCAT
!!     !>      !<
```

Funkce

`%SUBSTRING (%SST), %TRIM, %TRIML, %TRIMR, %CHAR, %LOWER, %UPPER`

Logické výrazy

Operandy jsou konstanty nebo proměnné typu `*CHAR, *DEC, *INT, *UINT`

Relační operátory

```
*EQ   *GT   *LT   *GE   *LE   *NE   *NG   *NL
=     >     <     >=    <=    ^=    ^>    ^<
```

Logické operátory

```
*AND   *OR   *NOT
&      |     ^
```

Výsledek výrazu je typu `*LGL`

`'0'` false
`'1'` true

Vybrané funkce

%CHAR(*data*)

převádí data typu *LGL, *DEC, *INT, *UINT na typ *CHAR.

%DEC(*data [počet-číslic počet-desetinných-míst]*)

převádí data typu *CHAR, *LGL, *DEC, *INT, *UINT na typ *DEC. Chybí-li údaje o počtu číslic a des. míst, dosadí se 15 a 5. U číselné konstanty se převezmou z dat. U typu *LGL se dosadí 1.

%INT(*data*) převádí data typu *CHAR, *LGL, *DEC, *UINT na typ *INT.

%UINT(*data*) převádí data typu *CHAR, *LGL, *DEC, *INT na typ *UINT.

%LEN(*proměnná*) vrací počet číslic nebo počet znaků číselné nebo znakové proměnné.

%SIZE(*proměnná*) vrací počet bajtů proměnné.

%BINARY(*znaková-proměnná [začátek délka]*) nebo
zkráceně **%BIN**(*znaková-proměnná [začátek délka]*)

vrací binární číslo se znaménkem (*INT) délky 2 nebo 4. Překročí-li součet začátku a délky velikost zadané proměnné, ohlásí chybu. Funkce se použije v příkazu CHGVAR v parametru VALUE nebo i v parametru VAR.

```
DCL         &DEC  *DEC  15
DCL         &INT  *INT  4
DCL         &HEX  *CHAR 4  X'000F0002'
DCL         &CHAR *CHAR 10

CHGVAR      &CHAR (%BINARY(&HEX 1 2))       /* '0000000015' */
CHGVAR      &INT  (%BIN(&HEX 3 2))          /* 2            */
CHGVAR      &DEC  (&INT * &INT + 15)        /* 19           */
```

```
Variable     Type       Length     Value                      Value in Hexadecimal
                                    *...+....1....+....2....+  * . . . + . . . . 1
&CHAR        *CHAR        10       '0000000015'               F0F0F0F0F0F0F0F1F5
&DEC         *DEC       15 0        19
&HEX         *CHAR        4       '    '                     000F0002
&INT         *INT         4        2                          00000002
```

```
DOWHILE (&DEC *GT 0 *AND &INT *EQ 2)       /* cyklus od 19 do 1 */
    CHGVAR    &DEC  (&DEC - 1)
ENDDO
```

```
&DEC         *DEC       15 0        0
```

```
IF (%BIN(&HEX) > 0) (CHGVAR &DEC (%BIN(&HEX))) /* &HEX je X'000F0002' */
```

```
&DEC         *DEC       15 0        983042
```

%SUBSTRING(*znaková-proměnná začátek délka*) nebo zkráceně

%SST(*znaková-proměnná začátek délka*)

vrací znakový řetězec nalezený na pozici *začátek* v délce *délka* v zadané *znakové proměnné*. Místo znakové proměnné může být zapsáno *LDA (local data area). Překročí-li součet začátku a délky velikost zadané proměnné, ohlásí chybu. Funkce se použije v příkazu CHGVAR v parametru VALUE nebo i v parametru VAR. Použije se také v podmínce příkazu IF, DOUNTIL, DOWHILE.

```
dcl &text *char 20 'ABCDEFGHIJKLMN'
dcl &pozice *int 2 4
dcl &vysledek1 *char 10
dcl &vysledek2 *char 10

CHGVAR     VAR(&VYSLEDEK1) VALUE(%SUBSTRING(&TEXT 4 3))
IF      (%SST(&TEXT 4 3) *EQ 'DEF') (CHGVAR &TEXT 'abc')
CHGVAR     VAR(%SUBSTRING(&VYSLEDEK2 &POZICE 3)) 'xyz'
```

```
Variable      Type        Length      Value                       Value in Hexadecimal
                                       *...+....1....+....2....+   * . . . + . . . . 1 .
&POZICE         *INT             2     4                           0004
&TEXT           *CHAR           20    'abc                 '       818283404040404040404040
&VYSLEDEK1      *CHAR           10    'DEF       '                 C4C5C640404040404040
&VYSLEDEK2      *CHAR           10    '   xyz    '                 404040A7A8A940404040
```

*%***SWITCH**(*8znaková maska*)

Dovoluje komunikovat mezi programy v zásobníku úlohy (jobu). V popisu úlohy (objektu typu *JOBD) je parametr SWS (Job Switches) a v něm 8místný údaj, kde v každém místě může být nula nebo jednička. Počáteční hodnota je 00000000. V průběhu úlohy ji můžeme měnit příkazem CHGJOB v parametru SWS, např.

```
CHGJOB SWS(01100001)
```

Nebo před spuštěním dávkové úlohy

```
SBMJOB CMD(CALL PGM(DQRECEIVE)) JOB(*JOBD) JOBQ(*JOBD) SWS(01100001)
```

Tuto hodnotu lze použít pomocí funkce %SWITCH s 8znakovou maskou, v níž každý znak představuje test

0	zda odpovídající místo v SWS je 0
1	zda odpovídající místo v SWS je 1
X	že odpovídající místo v SWS se nezkoumá

Jestliže se všechny hodnoty v masce – kromě X – shodují s hodnotami ve stejných pozicích v SWS, je výsledek logická hodnota `'1'` (true), jinak je `'0'` (false).

```
DCL    &IN01 *LGL

CHGVAR &IN01 %SWITCH(011XXXXX)      /* '1' */
IF (%SWITCH(11XXXXXX)) CALL PROGA
IF (%SWITCH(10XXXXXX)) CALL PROGB
IF (%SWITCH(01XXXXXX)) CALL PROGC   /* shoda, provede se */
IF (%SWITCH(00XXXXXX)) CALL PROGD
```

MONMSG – Monitor Message

Popis příkazu

Příkaz **MONMSG** se používá k zachycení zpráv typu *ESCAPE, *STATUS, *NOTIFY, které byly poslány do zásobníku volání z programu nebo procedury, kde je příkaz použit.

```
MONMSG MSGID(message-identifier) CMPDTA(comparison-data) +
     EXEC(CL-command)
```

Message-identifier (identifikátor zprávy)
První 3 znaky musí být znaky začínající písmenem a pokračující písmenem nebo číslicí. Poslední 4 znaky musí být číslice nebo znaky A až F. Jsou-li poslední dvě nebo čtyři místa nuly, je to generický identifikátor zachycující zprávy, které mají na posledních pozicích s nulami libovolné znaky. Například CPF5100 zachytí všechny zprávy začínajíví CPF, ale CPF0000 zachytí všechny zprávy začínající CPF. V parametru MSGID lze zadat více identifikátorů.

Comparison-data (porovnávací data)
Je-li zapsáno *NONE a má-li hlídaná zpráva zadaný identifikátor, provede se příkaz zadaný v parametru EXEC.
Je-li zadán znakový řetězec s nejvýše 28 znaky, je porovnán s daty z přijaté zprávy. Souhlasí-li s prvními znaky přijaté zprávy, provede se příkaz zadaný v parametru EXEC. Jako hodnotu parametru nelze zadat proměnnou, ale jen konstantu. Porovnávací data lze zobrazit příkazem DSPPGMVAR.

CL-command
Příkaz provedený při zachycení zprávy.

Úrovně použití

Úroveň programu
Jeden nebo více příkazů MONMSG je zapsáno bezprostředně za deklaračními příkazy (DCL, DCLF). Tak se stejným způsobem zachytí předepsané zprávy vyslané z kteréhokoliv příkazu v programu. Parametr EXEC je nepovinný. Není-li zadán, zprávy se ignorují. Je-li zadán, smí obsahovat jen příkaz GOTO.

Úroveň příkazu
Jeden nebo více příkazů MONMSG je zapsáno bezprostředně za zvoleným příkazem. Tím se zachytí předepsané zprávy vyslané tímto příkazem.

Zobrazování záznamů fyzického souboru na obrazovce

Referenční soubor REF

```
     A                                      CCSID(870)
     A          R REFR
     A            CENAJ          9P 2       COLHDG('Cena/j.')
     A            MATER          5          COLHDG('Číslo' 'mater.')
     A            NAZEV         30          COLHDG('Název zboží')
```

Kompilace se provede příkazem

```
CRTPF FILE(*CURLIB/REF) SRCFILE(*LIBL/QDDSSRC) MBR(*NONE)
```

Fyzický soubor CENY – Ceník zboží

```
     A                                      REF(REF)
     A                                      UNIQUE  CCSID(870)
     A          R CENYR
      *   Číslo zboží
     A            MATER     R
      *   Cena za jednotku (kus)
     A            CENAJ     R
      *   Název zboží
     A            NAZEV     R
      *   Definice klíče - Číslo zboží
     A          K MATER
```

Kompilace se provede příkazem

```
CRTPF FILE(*CURLIB/CENY) SRCFILE(*LIBL/QDDSSRC) SRCMBR(CENY)
```

Plnění souboru daty

`STRDFU` s volbou `5.` Update data using temporary program nebo

```
UPDDTA CENY
```

```
WORK WITH DATA IN A FILE                    Mode . . . . :   ENTRY
Format . . . . :   CENYR                    File . . . . :   CENY

Cislo mater.: 00001
Cena/j.:      12550
Název zboži:  Zboží 1
```

Pomocné zobrazení dat souboru ceny

```
RUNQRY *N CENY
```

nebo

```
RUNQRY QRYFILE((CENY))
```

```
Cislo        Cena/j.   Název zboži
mater.
00001         125.50   Zboží 1
00002         46899    Zboží 2
00003         468.99   Zboží 3
00004         685.99   Zboží 4
```

Obrazovkový soubor DSPFCENY - Ceník zboží

```
     A                                      CA03(03 'Konec')
     A                                      DSPSIZ(24 80 *DS3)
     A          R DSPFCENY1
     A                                   1  3'Zobrazení materiálu:'
     A                                      COLOR(RED)
     A                                      DSPATR(UL)
     A            MATER     R        O  1 24REFFLD(CENYR/MATER *LIBL/CENY)
     A                                      COLOR(TRQ)
     A                                   3  3'Název'
     A                                   3 43'Cena'
     A            NAZEV     R        O  4  3REFFLD(CENYR/NAZEV *LIBL/CENY)
     A            CENAJ     R        O  4 36REFFLD(CENYR/CENAJ *LIBL/CENY)
     A                                      EDTCDE(P)
```

Kompilace se provede příkazem

```
CRTDSPF FILE(*CURLIB/QDDSSRC) SRCFILE(*LIBL/QDDSSRC) SRCMBR(DSPFCENY)
```

Program ZOBCENY pro zobrazení záznamů souboru CENY

```
/* Zobrazování jednotlivých záznamů fyzického souboru */
/* pomocí obrazovkového souboru                       */
             DCLF       FILE(CENY) OPNID(CENY)
             DCLF       FILE(DSPFCENY) OPNID(DSPFCENY)

/* Nekonečný cyklus                                   */
             DOUNTIL    COND('0')
/* Čtu záznam z fyzického souboru                     */
             RCVF       OPNID(CENY)
/* Není-li již žádný záznam, opustím cyklus           */
             MONMSG     MSGID(CPF0864) EXEC(LEAVE)
/* Přenos dat z fyzického souboru do obrazovkového    */
             CHGVAR     VAR(&DSPFCENY_MATER) VALUE(&CENY_MATER)
             CHGVAR     VAR(&DSPFCENY_CENAJ) VALUE(&CENY_CENAJ)
             CHGVAR     VAR(&DSPFCENY_NAZEV) VALUE(&CENY_NAZEV)
/* Zápis na obrazovku a čtení z klávesnice a obrazovky */
             SNDRCVF    RCDFMT(DSPFCENY1) OPNID(DSPFCENY)
/* Po F3 končí program, jinak pokračuje v cyklu        */
             IF         COND(&DSPFCENY_IN03) THEN(RETURN)
             ENDDO
```

Program RZOBCENY vybírá a řadí záznamy, volá program ZOBCENY

```
             OVRDBF     FILE(CENY) OVRSCOPE(*JOB) SHARE(*YES)
             OPNQRYF    FILE((CENY)) +
                        QRYSLT('MATER *LE "00004"') +
                        KEYFLD((MATER *DESCEND))
             CALL       PGM(ZOBCENY)    /* volání programu zobrazení záznamů */
             CLOF       OPNID(CENY)     /* zavřu soubor */
             DLTOVR     FILE(CENY) LVL(*JOB)
```

Programy se kompilují volbou **14** v PDM nebo příkazy

```
CRTBNDCL PGM(*CURLIB/ZOBCENY) SRCFILE(*LIBL/QCLSRC) SRCMBR(ZOBCENY)
CRTBNDCL PGM(*CURLIB/RZOBCENY) SRCFILE(*LIBL/QCLSRC) SRCMBR(RZOBCENY)
```

Spouštějí se volbou **C** v PDM nebo příkazy

```
CALL ZOBCENY
CALL RZOBCENY
```

Program CYKLY – DOWHILE, DOFOR, DOUNTIL, SNDPGMMSG, SNDMSG, LEAVE

```
Dcl var(&priznak) type(*lgl) value('1')  /* příznak je zapnutý */

Dcl var(&index) type(*int) len(4)
Dcl var(&limit) type(*int) len(4) value(5)
Dcl var(&index_txt) type(*char) len(1)

Dcl var(&vypsat) type(*lgl) value('1')

/*  Vypíše do job logu text 'Hodnota je správná' jen jednou. */

DoWhile (&priznak)
    SNDPGMMSG msg('Hodnota je správná') topgmq(*prv)
    Chgvar &priznak '0'   /* vypnu příznak */
EndDo

/* Vypíše do Job Logu 'Cyklus 1' až 'Cyklus 5'. */

DoFor (&index) from(1) to(&limit) by(1)
    Chgvar &index_txt &index
    SNDPGMMSG msg('Cyklus ' *cat &index_txt) topgmq(*prv)
EndDo

/* Vypíše dvakrát text do fronty uživatele */

top:
DoWhile (&vypsat)
    SNDMSG     msg('První text') tousr(*requester)
    DoUntil cond(*not &vypsat)
        SNDMSG msg('Druhý text') tousr(*requester)
        DoWhile (&vypsat)
            LEAVE cmdlbl(top)
        EndDo
    EndDo
EndDo
```

Výpis z job logu:

```
Hodnota je správná
Cyklus 1
Cyklus 2
Cyklus 3
Cyklus 4
Cyklus 5
```

Abychom viděli dvě poslední zprávy, možná budeme muset provést příkaz

```
CHGMSGQ MSGQ(*USRPRF) DLVRY(*BREAK)
```

```
     První text
       From  . . :   VZUPKA         01/13/25   16:33:55
     Druhý text
       From  . . :   VZUPKA         01/13/25   16:30:52
```

Datová oblast (data area), zamykání objektu, podprogram

Program DARA

```
             PGM

             DCL        VAR(&JOB) TYPE(*CHAR) LEN(10)
             DCL        VAR(&USER) TYPE(*CHAR) LEN(10)
             DCL        VAR(&STRING) TYPE(*CHAR) LEN(100)
             DCL        VAR(&REPLY) TYPE(*CHAR) LEN(1)

/* Neexistuje-li objekt DATA1, vytvořím jej v podprogramu       */
             CHKOBJ     OBJ(DATA1) OBJTYPE(*DTAARA)
             MONMSG     MSGID(CPF0000) +
                        EXEC(CALLSUBR SUBR(CRTARA))
/* Získám jméno uživatele z popisu úlohy                        */
             RTVJOBA    JOB(&JOB) USER(&USER)
/* Zamknu oblast výhradně pro svou úlohu, dovolím čtení         */
             ALCOBJ     OBJ((DATA1 *DTAARA *EXCLRD))
/* Do datové oblasti zapíšu jméno úlohy a uživatele             */
             CHGDTAARA  DTAARA(DATA1 (1 10)) VALUE(&JOB)
             CHGDTAARA  DTAARA(DATA1 (12 22)) VALUE(&USER)
/* Zobrazím obsah datové oblasti                                */
             DSPDTAARA  DTAARA(DATA1)
/* Získám část obsahu datové oblasti do proměnné                */
             RTVDTAARA  DTAARA(DATA1 (1 10)) RTNVAR(&STRING)
/* Zobrazím obsah proměnné na obrazovku                         */
             SNDMSG     MSG(&STRING) TOUSR(*REQUESTER)
/* Odemknu oblast                                               */
             DLCOBJ     OBJ((DATA1 *DTAARA *EXCLRD))
             RETURN

/* Podprogram CRTARA                                            */
             SUBR       SUBR(CRTARA)
/* Vytvořím datovou oblast                                      */
             CRTDTAARA  DTAARA(DATA1) TYPE(*CHAR) LEN(100)
             ENDSUBR

             ENDPGM
```

Po spuštění se program zastaví a zobrazí se zpráva z příkazu `DSPDTAARA` – jméno úlohy a uživatele:

```
           Value
Offset      *...+....1....+....2....+....3....+....4....+....5
    0      'QPADEV0004 VZUPKA                                '
   50      '                                                 '
```

Po stisku Enter se program opět zastaví a zobrazí se zpráva z příkazu `SNDMSG`, která zobrazí část textu (jen jméno úlohy) z datové oblasti (z příkazu `RTVDTAARA`).

```
Queue . . . . . :   VZUPKA                  Program . . . . :   *DSPMSG
  Library . . . :     QUSRSYS                 Library . . . :
Severity  . . . :   00                      Delivery  . . . :   *BREAK

Type reply (if required), press Enter.
  From  . . . :   VZUPKA         12/25/24   16:46:54
  QPADEV0004
```

Spustíme li v tomto stavu stejný program v jiné úloze (session), bude tam čekat 30 sekund a pak skončí se zprávou na obrazovce:

```
CPF1002 received by procedure DARA. (C D I R)
```

Odpovím-li `C`, budou v job logu tyto zprávy:

```
3 > call dara
       900 - CHKOBJ OBJ(DATA1) OBJTYPE(*DTAARA)
      1400 - RTVJOBA JOB(&JOB) USER(&USER)
      1600 - ALCOBJ OBJ((DATA1 *DTAARA *EXCLRD))
    Cannot allocate object DATA1.
    CPF1002 received by procedure DARA. (C D I R)
  ? C
    Application error.  CPF1002 unmonitored by DARA at statement 0000001600,
      instruction X'0000'.
```

Čekání nastalo proto, že příkaz `ALCOBJ` zamkl datovou oblast výhradně pro svoji úlohu zámkem `*EXCLRD`. Druhá úloha ve stejném programu v příkazu `ALCOBJ` – číslo `0016.00` nemůže získat stejný zámek, a tak čeká předvolených 30 vteřin, než zhavaruje. Tento čas lze změnit příkazem `CHGJOB DFTWAIT(30)` s jiným číslem.

Nahradíme-li zámek `*EXCLRD` zámkem `*SHRUPD`, druhá úloha nečeká a přepisuje datovou oblast svými údaji v příkazech `CHGDTAARA`. V následující tabulce jsou pravidla zamykání objektu mezi úlohami.

Když jedna úloha získá tento zámek,		druhá může získat zámek
*EXCL	výhradní	žádný
*EXCLRD	výhradní, dovolí čtení	*SHRRD
*SHRUPD	sdílený, dovolí update	*SHRUPD, *SHRRD
*SHRNUP	sdílený, nedovolí update	*SHRNUP, *SHRRD
*SHRRD	sdílený, dovolí čtení	*EXCLRD, *SHRUPD, *SHRNUP, *SHRRD

Zasílání a přijímání zpráv mezi programy v jedné úloze

Program MSGSND

```
/*****************************************************************/
/*  Program MSGSND posílá tři zprávy do fronty volajícího programu.  */
/*****************************************************************/
dcl &text  *char 50 'Hodnota indexu je'
dcl &index *int
dcl &indch *char 10

dofor &index from(1) to(3) by(1)
    chgvar &indch &index
    sndpgmmsg  msg(&text *bcat &indch) topgmq(*PRV) msgtype(*info)
enddo
```

Parametr `*PRV` směruje zprávu do fronty volajícího programu nebo programu `QCMD`. Program `QCMD` provádí příkazy zadané z příkazového řádku.

Program MSGRCV2 volá program MSGSND

```
/*****************************************************************/
/*  Program čte všechny zprávy ze své fronty, ale nemaže je.        */
/*  Přečtené zprávy doplní a pošle výš (do *EXT nebo *PRV).         */
/*****************************************************************/
             DCL        VAR(&MSGTEXT) TYPE(*CHAR) LEN(100)
             DCL        VAR(&MSGKEY) TYPE(*CHAR) LEN(4)
             DCL        VAR(&BLANKS) TYPE(*CHAR) LEN(100) VALUE(' ')
             dcl        &key *uint 4
             dcl        &keyc *char 4

/*  Volám program, který do mé fronty pošle tři zprávy */
/*  a já je pak přečtu a pošlu výš (do *PRV nebo *EXT) */
             CALL       PGM(MSGSND)

/*  Čtu první zprávu z fronty tohoto programu a nesmažu ji */
             RCVMSG     MSGQ(*PGMQ) MSGTYPE(*FIRST) WAIT(0) RMV(*NO) +
                          KEYVAR(&MSGKEY) MSG(&MSGTEXT)
             chgvar     &key %bin(&msgkey)
             chgvar     &keyc %char(&key)
DMPCLPGM
/*  V cyklu čtu další zprávy, dokud nejsou prázdné */
             DOWHILE    COND(&MSGTEXT *NE &BLANKS)

/*  Přepošlu přečtenou zprávu do *PRV (nebo *EXT) */
/*  Zpráva poslaná do *PRV nepřeruší práci */
/*  Zpráva poslaná do *EXT přeruší práci a čeká na akci uživatele */
             SNDPGMMSG  MSG('Přeposlaná zpráva: ' *CAT &MSGTEXT +
                          *BCAT &KEYC) TOPGMQ(*EXT)
/*  Čtu další zprávu z fronty tohoto programu a nesmažu ji */
             RCVMSG     MSGQ(*PGMQ) MSGTYPE(*NEXT) MSGKEY(&MSGKEY) +
                          RMV(*NO) KEYVAR(&MSGKEY) MSG(&MSGTEXT)
             chgvar     &key %bin(&msgkey)
             chgvar     &keyc %char(&key)
```

Příkaz `DMPCLPGM` vytiskne momentální obsah paměti.

Příkaz RCVMSG čte z vlastní programové fronty MSG(*PGMQ). Nejprve přečte první zprávu (*FIRST), uloží ji do proměnné &MSGTEXT a získá **klíč zprávy** (message key) parametrem KEYVAR do proměnné &MSGKEY.

Pak v cyklu čte další zprávu (*NEXT). Přitom se řídí klíčem předchozí zprávy uloženém v proměnné &MSGKEY. a získává nový klíč paramerem KEYVAR. Parametrem RMV(*NO) ponechává zprávy ve frontě.

Klíč zprávy je v binárním tvaru, ale musí být deklarován jako typ *CHAR. Aby se klíč zobrazil (v přeposlané zprávě), byl převeden z typu *CHAR do typu *UINT a z něj zpět do typu *CHAR.

Po skončení programu v job logu zůstávají mj. tyto zprávy:

```
3 > CALL MSGRCV2
      1500 - CALL PGM(MSGSND)
      1200 - SNDPGMMSG MSG('Hodnota indexu je 0000000001') TOPGMQ(*PRV)
      MSGTYPE(*INFO)
    Hodnota indexu je 0000000001
      1200 - SNDPGMMSG MSG('Hodnota indexu je 0000000002') TOPGMQ(*PRV)
      MSGTYPE(*INFO)
    Hodnota indexu je 0000000002
      1200 - SNDPGMMSG MSG('Hodnota indexu je 0000000003') TOPGMQ(*PRV)
      MSGTYPE(*INFO)
    Hodnota indexu je 0000000003
           - RETURN        /* RETURN due to end of CL program */
      1800 - RCVMSG MSGQ(*PGMQ) MSGTYPE(*FIRST) WAIT(0) RMV(*NO)
      KEYVAR(&MSGKEY) MSG(&MSGTEXT)
      2200 - DMPCLPGM
      2900 - SNDPGMMSG MSG('Přeposlaná zpráva: Hodnota indexu je 0000000001
      441') TOPGMQ(*EXT)
    Přeposlaná zpráva: Hodnota indexu je 0000000001 441
      3200 - RCVMSG MSGQ(*PGMQ) MSGTYPE(*NEXT) MSGKEY(X'000001B9') RMV(*NO)
      KEYVAR(&MSGKEY) MSG(&MSGTEXT)
      2900 - SNDPGMMSG MSG('Přeposlaná zpráva: Hodnota indexu je 0000000002
      443') TOPGMQ(*EXT)
    Přeposlaná zpráva: Hodnota indexu je 0000000002 443
      3200 - RCVMSG MSGQ(*PGMQ) MSGTYPE(*NEXT) MSGKEY(X'000001BB') RMV(*NO)
      KEYVAR(&MSGKEY) MSG(&MSGTEXT)
      2900 - SNDPGMMSG MSG('Přeposlaná zpráva: Hodnota indexu je 0000000003
      445') TOPGMQ(*EXT)
    Přeposlaná zpráva: Hodnota indexu je 0000000003 445
      3200 - RCVMSG MSGQ(*PGMQ) MSGTYPE(*NEXT) MSGKEY(X'000001BD') RMV(*NO)
      KEYVAR(&MSGKEY) MSG(&MSGTEXT)
           - RETURN        /* RETURN due to end of CL program */
```

Této podrobnosti v job logu dosáhneme předchozí změnou parametrů příkazu CHGJOB (Change Job). Známe-li jména a hodnoty parametrů, můžeme spustit následující příkaz.

```
CHGJOB LOG(4 00 *MSG) LOGCLPGM(*YES)
```

Neznáme-li je, spustíme příkaz

```
? CHGJOB
```

a dostaneme tento náznak:

```
                              Change Job (CHGJOB)

Type choices, press Enter.

Job name . . . . . .   *_________    Name, *
  User . . . . . .     __________    Name
  Number . . . . . .   ______       000000-999999
```

Teď stiskneme postupně F11 (zobrazit jména parametrů), Enter, F10 a Page Down. Dostaneme tento náznak:

```
Type choices, press Enter.

Message logging:                 LOG
  Level  . . . . . . . . . . . .              > 4
  Severity . . . . . . . . . . .              > 00
  Text . . . . . . . . . . . . .              > *MSG
Log CL program commands  . . . . LOGCLPGM     > *YES
Job log output . . . . . . . . . LOGOUTPUT       *JOBEND
Job message queue full action  . JOBMSGQFL       *NOWRAP
Inquiry message reply  . . . . . INQMSGRPY       *RQD
Break message handling . . . . . BRKMSG          *NORMAL
Status message . . . . . . . . . STSMSG          *NORMAL
DDM conversation . . . . . . . . DDMCNV          *KEEP
Schedule date  . . . . . . . . . SCDDATE         *SAME
Schedule time  . . . . . . . . . SCDTIME         *SAME
Job date . . . . . . . . . . . . DATE            122524
Date format  . . . . . . . . . . DATFMT          *MDY
Date separator . . . . . . . . . DATSEP          '/'
                                                                    More...
F3=Exit   F4=Prompt   F5=Refresh   F12=Cancel   F13=How to use this display
F24=More keys
```

Do parameteru `LOG` (Message logging) zapíšeme **`4 00 *MSG`** a do parametru `LOGCLPGM` (Log CL program commands) zapíšeme **`*YES`**. Stiskneme Enter.

Zasílání a přijímání zpráv mezi úlohami přes datovou frontu

Program CRTDTAQ – vytvoření datové fronty

```
/*   Zkusím zrušit frontu DQFIFO typu FIFO                              */
             DLTDTAQ    DTAQ(*CURLIB/DQFIFO)
             MONMSG     MSGID(CPF0000)
/*   Vytvořím ji v *CURLIB                                              */
             CRTDTAQ    DTAQ(*CURLIB/DQFIFO) TYPE(*STD) MAXLEN(256) +
                          FORCE(*YES) SEQ(*FIFO) SENDERID(*YES) +
                          TEXT('Datová fronta (FIFO)')
```

Příkaz DQSEND k vyvolání CL programu DQSEND

```
/*   Command DQSEND                                                     */
/*   CPP:    DQSEND                                                     */
             CMD        PROMPT('Zasílání do fronty DQFIFO')
             PARM       KWD(MSG) TYPE(*CHAR) LEN(50) MIN(1) MAX(1) +
                          PROMPT('Text zasílaný do fronty DQFIFO')
```

Program DQSEND – zaslání zprávy do datové fronty

```
             PGM        PARM(&MSG)
/*   Definice proměnných                                                */
             DCL        VAR(&MSG) TYPE(*CHAR) LEN(50)
             DCL        VAR(&MSGL) TYPE(*DEC) LEN(5 0) VALUE(50)
             DCL        VAR(&WAIT) TYPE(*DEC) LEN(5 0)
             DCL        VAR(&JOBNUM) TYPE(*CHAR) LEN(6)
             DCL        VAR(&USERNAME) TYPE(*CHAR) LEN(10)

/*   Získat číslo úlohy a jméno uživatele                               */
             RTVJOBA    USER(&USERNAME) NBR(&JOBNUM)
/*   Naplnit proměnné pro volání API QSNDDTAQ                           */
             CHGVAR     VAR(&MSG) VALUE(&MSG *BCAT &USERNAME *BCAT +
                          &JOBNUM)
/*   Poslat zprávu do fronty DQFIFO                                     */
             CALL       PGM(QSNDDTAQ) PARM(DQFIFO *LIBL &MSGL &MSG)
             ENDPGM
```

Program DQRECEIVE – přijímání zpráv z datové fronty

```
/*   Definice proměnných                                                */
             DCL        VAR(&MSG) TYPE(*CHAR) LEN(256)
             DCL        VAR(&MSGL) TYPE(*DEC) LEN(5 0) VALUE(256)
             DCL        VAR(&WAIT) TYPE(*DEC) LEN(5 0) VALUE(-1)

/*   Opakuji, dokud nepřečtu zprávu začínající písmenem X              */
             DOUNTIL    COND(%SST(&MSG 1 1) *EQ 'X')

/*   Čtu z fronty DQFIFO                                                */
             CALL       PGM(QRCVDTAQ) PARM(DQFIFO *LIBL &MSGL &MSG &WAIT)
/*   Zapíšu výsledek do fronty *EXT Job Logu                            */
             SNDPGMMSG  MSG('Přečtená zpráva:' *CAT &MSG) TOPGMQ(*EXT)

             ENDDO   /* DO UNTIL */
```

Příklad – Dvě interakční úlohy

První úloha č. `115348` pošle zprávu do fronty.

```
DQSEND MSG('zpráva 1')
```

Druhá úloha přijme zprávu.

```
CALL DQRECEIVE
```

```
Přečtená zpráva:Zpráva 1 VZUPKA 115348
```

První úloha pošle zprávu `'X'` k ukončení příjmu v druhé úloze.

```
DQSEND MSG('X')
```

Druhá úloha přijme zprávu `X` a na obrazovce přibude nová zpráva.

```
Přečtená zpráva:Zpráva 1 VZUPKA 115348
Přečtená zpráva:X VZUPKA 115348

Press Enter to continue.
```

Po Enter program `DQRECEIVE` v druhé úloze skončí. V job logu druhé úlohy najdeme tyto zprávy:

```
33 > call dqreceive
      1200 - CALL PGM(QRCVDTAQ)
      1400 - SNDPGMMSG MSG('Přečtená zpráva:Zpráva 1 VZUPKA 115348')
      TOPGMQ(*EXT)
    Přečtená zpráva:Zpráva 1 VZUPKA 115348
      1200 - CALL PGM(QRCVDTAQ)
      1400 - SNDPGMMSG MSG('Přečtená zpráva:X VZUPKA 115348') TOPGMQ(*EXT)
    Přečtená zpráva:X VZUPKA 115348
           - RETURN        /* RETURN due to end of CL program */
```

Příklad – Přijímací program v dávkové úloze

```
SBMJOB CMD(CALL PGM(DQRECEIVE))
```

```
Job 115355/VZUPKA/QDFTJOBD submitted to job queue QBATCH in library QGPL.
```

Příkazem `WRKACTJOB` s následujícími volbami:

5 u jobu `QDFTJOBD` a
11. `Display call stack`

se zobrazí zásobník procedur:

```
                            Display Call Stack

Job:   QDFTJOBD       User:   VZUPKA         Number:   115355
Thread:   00000033


Type  Program                  Statement         Procedure
   1  QCMD       QSYS                      /01C8
      DQRECEIVE  VZZAK                             _CL_PEP
      DQRECEIVE  VZZAK          1200               DQRECEIVE
      QRCVDTAQ   QSYS                      /010F
```

Z interakční úlohy č. 115348 pošlu tři zprávy:

```
DQSEND MSG('Zpráva 1')
DQSEND MSG('Zpráva 2')
DQSEND MSG('Zpráva 3')
```

V job logu dávkové úlohy č. 115350 je vidět průběh přijímání zpráv:

```
Job . . :   QDFTJOBD      User . . :   VZUPKA        Number . . . :

 >> CALL PGM(DQRECEIVE)
      1200 - CALL PGM(QRCVDTAQ)          /* The CALL command contains
      parameters */
      1400 - SNDPGMMSG MSG('Přečtená zpráva:Zpráva 1 VZUPKA 115348')
      TOPGMQ(*EXT)
    Přečtená zpráva:Zpráva 1 VZUPKA 115348
      1200 - CALL PGM(QRCVDTAQ)          /* The CALL command contains
      parameters */
      1400 - SNDPGMMSG MSG('Přečtená zpráva:Zpráva 2 VZUPKA 115348')
      TOPGMQ(*EXT)
    Přečtená zpráva:Zpráva 2 VZUPKA 115348
      1200 - CALL PGM(QRCVDTAQ)          /* The CALL command contains
      parameters */
      1400 - SNDPGMMSG MSG('Přečtená zpráva:Zpráva 3 VZUPKA 115348')
      TOPGMQ(*EXT)
    Přečtená zpráva:Zpráva 3 VZUPKA 115348
      1200 - CALL PGM(QRCVDTAQ)          /* The CALL command contains
      parameters */
```

Na konec pošlu zprávu X:

```
DQSEND MSG('X')
```

```
Job 115355/VZUPKA/QDFTJOBD completed normally on 12/15/24 at 13:28:54.
```

Protože dávková úloha skončila, informaci o přijetí zprávy X uvidíme v tiskovém souboru QPJOBLOG:

```
Command                     12/15/24  13:26:07.793164  QCLCLCPR     QSYS        04CF     DQRECEIVE   VZZAK
                          To module . . . . . :   DQRECEIVE
                          To procedure  . . . . . :   DQRECEIVE
                          Statement . . . . . :   1200
                          Message . . . . :     1200 - CALL PGM(QRCVDTAQ)          /* The CALL command
                            contains parameters */
Command                     12/15/24  13:28:54.840037  QCADRV       QSYS        041C     DQRECEIVE   VZZAK
                          To module . . . . . :   DQRECEIVE
                          To procedure  . . . . . :   DQRECEIVE
                          Statement . . . . . :   1400
                          Message . . . . :     1400 - SNDPGMMSG MSG('Přečtená zpráva:X VZUPKA 115348')
                            TOPGMQ(*EXT)
Information                 12/15/24  13:28:54.840130  DQRECEIVE    VZZAK       *STMT    *EXT
                          From module . . . . . :   DQRECEIVE
                          From procedure  . . . . . :   DQRECEIVE
                          Statement . . . . . :   1400
                          Message . . . . :   Přečtená zpráva:X VZUPKA 115348
Command                     12/15/24  13:28:54.840167  QCLRTNE      QSYS        0058     DQRECEIVE   VZZAK
                          To module . . . . . :   DQRECEIVE
                          To procedure  . . . . . :   DQRECEIVE
                          Statement . . . . . :   1800
                          Message . . . . :         - RETURN        /* RETURN due to end of CL program
                            */
Completion              00  12/15/24  13:28:54.841056  QWTMCEOJ     QSYS        0161     *EXT
                          Message . . . . :   Job 115355/VZUPKA/QDFTJOBD ended on 12/15/24 at 13:28:54;
                            .004 seconds used; end code 0 .
```

Volání procedury v CL

Výpočet přepony pravoúhlého trojúhelníka v jazyku CL není možný, protože nelze vypočítat odmocninu. Použijeme tedy proceduru – funkci – zapsanou v jazyku RPG, která to umožňuje. Vyvoláme ji z CL programu příkazem CALLPRC. K tomu účelu potřebujeme dva moduly, které spojíme do programu.

RPG modul PREP_PROC

Obsahuje proceduru (funkci) `PREPONA`. Ta přijímá dva parametry a vrací výsledek příkazem `return`.

```
**free
ctl-opt  nomain;

dcl-proc PREPONA export;   // deklarace procedury

    dcl-pi prepona  packed(15: 5) ;   // rozhraní procedury
        a    packed(7: 2) value;
        b    packed(7: 2) value;
    end-pi;

    return %SQRT(a**2 + b**2);  // vrací délku přepony

end-proc;
```

CL modul PREP_CALL

Představuje hlavní program, který volá proceduru `PREPONA` v RPG modulu `PREP_PROC`. Předává dva parametry a přijímá vrácenou hodnotu.

```
DCL         VAR(&A) TYPE(*DEC) LEN(7 2) VALUE(3)
DCL         VAR(&B) TYPE(*DEC) LEN(7 2) VALUE(4)

DCL         VAR(&PREP) TYPE(*DEC) LEN(15 5)
DCL         VAR(&PREP_C) TYPE(*CHAR) LEN(16)

CALLPRC     PRC(PREPONA) PARM((&A *BYVAL) (&B *BYVAL)) +
                         RTNVAL(&PREP)

CHGVAR      VAR(&PREP_C) VALUE(&PREP)
SNDPGMMSG   MSG('přepona = ' *CAT &PREP_C) TOPGMQ(*EXT)
```

Volání funkce provádíme příkazem `CALLPRC` (Call Procedure). Parametry `&A`, `&B` předáváme hodnotou (`*BYVAL`) a vrácenou hodnotu `&PREP` zadáme v parametru `RTNVAL`.

Moduly musíme zkompilovat příkazem `CRTCLMOD` (nebo volbou 15 v PDM).

```
CRTCLMOD MODULE(PREP_CALL)
CRTRPGMOD MODULE(PREP_PROC)
```

Moduly spojíme do programu, který nazveme `VYPOC_PREP`

```
CRTPGM PGM(VYPOC_PREP) MODULE(PREP_CALL PREP_PROC)
```

Vyvoláním programu

```
CALL VYPOC_PROC
```

dostaneme výsledek

```
přepona = 0000000005.00000
```

Zjištění objektů použitých v programech

Program REFPGM – Reference podle programů

Program přijímá z příkazu jeden parametr typu *CHAR, který však má strukturu, kde první 2 bajty představují binární číslo a určuje počet skutečně zadaných parametrů v podobě dvojic 10znakových jmen (objekt, knihovna). Abychom mohli takovou strukturu zadat, použijeme k vyvolání programu CL příkaz (viz dále).

```
/*   Parametry:                                                    */
             PGM   PARM(&PLIST)
             DCL   &PLIST *CHAR 202 /* 2 + 10 x 20 */

             DCL   &PGM *CHAR 10   /* jméno programu (programů) */
             DCL   &LIB *CHAR 10   /* jméno knihovny (knihoven) */
             DCL   &N   *DEC 4 0   /* počet parametrů */
             DCL   &I   *DEC 4 0   /* čítač parametrů */
             DCL   &P   *DEC 4 0   /* ukazatel na jm. prog. */
             DCL   &L   *DEC 4 0   /* ukazatel na jm. knih. */
/*   Proměnná pro náhradu nebo přidání do souboru                  */
             DCL        VAR(&ADDREP) TYPE(*CHAR) LEN(8)
/*   Převod počtu parametrů z binárního na dekadické číslo         */
             CHGVAR     VAR(&N) VALUE(%BINARY(&PLIST 1 2))
             CHGVAR     VAR(&I) VALUE(0) /* čítač cyklů */
/*   Poprvé se obsah souboru bude nahrazovat novým seznamem        */
             CHGVAR     VAR(&ADDREP) VALUE(*REPLACE)

/*   Uzavřít soubor protože může být otevřen z předchozího volání  */
             CLOF       OPNID(QADSPPGM)
             MONMSG     MSGID(CPF0000)  /* ignorovat chybu */
 OPAKOVAT:
/*   Ukazatel (index) jména programu v seznamu parametrů           */
             CHGVAR     VAR(&P) VALUE(3 + &I * 20)
/*   Ukazatel (index) jména knihovny v seznamu parametrů           */
             CHGVAR     VAR(&L) VALUE(3 + &I * 20 + 10)

/*   Získat jméno programu a knihovny z kvalifikovaného jména      */
             CHGVAR     VAR(&PGM) VALUE(%SST(&PLIST &P 10))
             CHGVAR     VAR(&LIB) VALUE(%SST(&PLIST &L 10))

/*   Zapsat programové reference do QTEMP/QADSPPGM                 */
             DSPPGMREF  PGM(&LIB/&PGM) OUTPUT(*OUTFILE) +
                          OUTFILE(QTEMP/QADSPPGM) OUTMBR(*FIRST +
                          &ADDREP)
             CHGPF      FILE(QTEMP/QADSPPGM) CCSID(870) /* český kód */
/*   Podruhé se nový seznam bude přidávat k dosavadnímu seznamu    */
             CHGVAR     VAR(&ADDREP) VALUE(*ADD)

/*   Je-li &I menší než &N-1 - Zvýšit &I o 1 a opakovat            */
             IF (&I < &N-1) DO
                CHGVAR &I (&I + 1)
                GOTO   OPAKOVAT
             ENDDO

/*   Uzavřít soubor protože může být otevřen z předchozího volání  */
             CLOF       OPNID(QADSPPGM)
             MONMSG     MSGID(CPF0000)

             ENDPGM
```

CL příkaz k vyvolání CL programu REFPGM

Pro CL příkaz zvolíme jméno REFPGM, stejné jako jméno programu, který ho provede. Určíme to při kompilaci CL příkazu:

```
CRTCMD CMD(*CURLIB/REFPGM) PGM(*LIBL/REFPGM)
```

Prováděcímu programu se říká Command Processing Program (CPP). Definice CL příkazu obsahuje příkazy CMD, PARAM a QUAL.

```
/*   Command REFPGM                                                 */
/*   CPP:    REFPGM                                                 */

             CMD        PROMPT('Tisk referencí podle programů')
             PARM       KWD(PGM) TYPE(NAME) MIN(1) MAX(10) +
                          PROMPT('Jméno programu')
 NAME:       QUAL       TYPE(*GENERIC) SPCVAL((*ALL)) CHOICE('Jméno, +
                          generic*, *ALL')
             QUAL       TYPE(*NAME) DFT(*LIBL) SPCVAL((*LIBL) +
                          (*ALL)) CHOICE('Jméno, *LIBL, *ALL') +
                          PROMPT('Jméno knihovny')
```

Příkaz CMD má zde jen jeden parametr určující titul vytvářeného příkazu.

Příkaz PARAM má 4 parametry. KWD určuje jméno parametru vytvářeného příkazu. TYPE určuje jméno návěstí u příkazu QUAL, MIN určuje minimální počet a MAX maximální počet hodnot parametru. PROMPT vysvětluje význam hodnot parametru.

První příkaz QUAL stanoví způsob zadání programu a výběr možných hodnot. Druhý příkaz QUAL stanoví způsob zadání knihovny. Pořadí těchto příkazů (program, knihovna) je důležité pro program, který je vytvářeným příkazem vyvoláván.

Když po kompilaci na příkazový řádek napíšeme příkaz

```
REFPGM
```

a stiskneme F4, ukáže se jeden povinný parametr s naznačením možných dalších hodnot znaménkem **+**. Prázdné jméno programu musí být zapsáno, protože nebyla zadána předem zvolená hodnota. Jméno knihovny bylo předem zvoleno parametrem DFT(*LIBL), ale lze je přepsat.

```
                    Tisk referencí podle programů (REFPGM)

Type choices, press Enter.

Jméno programu . . . . . .   ____________   Jméno, generic*, *ALL
  Jméno knihovny . . . . . .     *LIBL      Jméno, *LIBL, *ALL
               + for more values   ____________
                                     *LIBL
```

Spuštění příkazu REFPGM z příkazového řádku

```
REFPGM PGM(VZZAK/DARA VZZAK/SOUBORY VZZAK/ZOBCENY)
```

nebo pomocí F4

```
Jméno programu . . . . . . . . . > DARA        Jméno, generic*, *ALL
  Jméno knihovny . . . . . . . . >   VZZAK     Jméno, *LIBL, *ALL
                                  > SOUBORY
                                  >   VZZAK
               + for more values > ZOBCENY
                                  >   VZZAK
```

Po stisku Enter získáme výsledek v souboru QADSPPGM v dočasné knihovně QTEMP a také na obrazovce.

Abychom mohli zobrazit jen vybrané položky, zjistíme jména polí souboru QADSPPGM. Zapíší se do souboru FIELDS v knihovně QTEMP.

```
DSPFFD FILE(QADSPPGM) OUTPUT(*OUTFILE) OUTFILE(QTEMP/FIELDS)
```

Tato jména polí zobrazíme

```
RUNQRY QRYFILE((QTEMP/FIELDS))
```

Redukovaný výpis jmen polí z obrazovky

```
External    Field     Number  Decimal    Field Text
Field Name  Length    Of      Positions  Description
            In Bytes  Digits
WHLIB            10       0          0  Library
WHPNAM           10       0          0  Program
WHTEXT           50       0          0  Text 'description'
WHFNUM            5       5          0  Number of objects referenced
WHDTTM           13       0          0  Retrieval date:  century/date/time
WHFNAM           11       0          0  Object referenced: 1=*EXPR
WHLNAM           11       0          0  Library referenced: 1=*EXPR
WHSNAM           11       0          0  File name in source program: 1=*EXPR
WHRFNO            3       3          0  Number of record formats referenced. -1=See WHRFNB
WHFUSG            2       2          0  1=I,2=O,3=I/O,4=U,5=I/U,6=O/U,7=I/O/U,8=N/S,0=N/A
WHRFNM           10       0          0  Record format referenced
WHRFSN           13       0          0  Format level identifier
WHRFFN            5       5          0  Number of fields
WHOBJT            1       0          0  Object type: F=File, P=Program, D=Data area
WHOTYP           10       0          0  Object type
...
```

Vybrané informace můžeme vypsat pomocí CL programu. Použijeme jména polí zjištěná ze souboru QADSPPGM a vynecháme objekty typu *SRVPGM (servisní programy).

```
DCL         VAR(&ROW) TYPE(*CHAR) LEN(120)
DCLF        FILE(QADSPPGM) OPNID(QADSPPGM)

REFPGM      PGM(VZZAK/DARA VZZAK/SOUBORY VZZAK/ZOBCENY)

OVRDBF      FILE(QADSPPGM) OVRSCOPE(*JOB) SHARE(*YES)
OPNQRYF     FILE((QTEMP/QADSPPGM)) QRYSLT('WHOTYP *NE "*SRVPGM"')

DOUNTIL     COND('0')
RCVF        OPNID(QADSPPGM)
MONMSG      MSGID(CPF0864) EXEC(LEAVE)
CHGVAR      VAR(&ROW) VALUE(&QADSPPGM_WHLIB *BCAT +
              &QADSPPGM_WHPNAM *BCAT &QADSPPGM_WHFNAM +
              *BCAT &QADSPPGM_WHLNAM *BCAT &QADSPPGM_WHOTYP)
SNDPGMMSG   MSG(&ROW) TOPGMQ(*EXT)
ENDDO

CLOF        OPNID(QADSPPGM)
DLTOVR      FILE(QADSPPGM) LVL(*JOB)
```

```
VZZAK DARA DATA1 *LIBL *DTAARA
VZZAK DARA DATA1 *CURLIB *DTAARA
VZZAK SOUBORY QADSPOBJ *FILE
VZZAK SOUBORY FILES QTEMP *FILE
VZZAK SOUBORY QAFDMBRL *FILE
VZZAK SOUBORY MEMBERS QTEMP *FILE
VZZAK SOUBORY &FILES_ODOB &LIBRARY *FILE
VZZAK SOUBORY *RUNOPT *RUNOPT *FILE
VZZAK SOUBORY QADSPOBJ QSYS *FILE
VZZAK ZOBCENY CENY VZZAK *FILE
VZZAK ZOBCENY DSPFCENY VZZAK *FILE
VZZAK DARA DATA1 *LIBL *DTAARA
```

Stejné informace dostaneme pomocí SQL v úpravnější podobě.

```
STRSQL
select WHLIB, WHPNAM, WHFNAM, WHLNAM, WHOTYP
from QTEMP/QADSPPGM
where WHOTYP <> '*SRVPGM'
```

```
Library     Program     Object       Library      Object
                        Referenced                Type
VZZAK       DARA        DATA1        *LIBL        *DTAARA
VZZAK       DARA        DATA1        *CURLIB      *DTAARA
VZZAK       SOUBORY     QADSPOBJ                  *FILE
VZZAK       SOUBORY     FILES        QTEMP        *FILE
VZZAK       SOUBORY     QAFDMBRL                  *FILE
VZZAK       SOUBORY     MEMBERS      QTEMP        *FILE
VZZAK       SOUBORY     &FILES_ODOB  &LIBRARY     *FILE
VZZAK       SOUBORY     *RUNOPT      *RUNOPT      *FILE
VZZAK       SOUBORY     QADSPOBJ     QSYS         *FILE
VZZAK       ZOBCENY     CENY         VZZAK        *FILE
VZZAK       ZOBCENY     DSPFCENY     VZZAK        *FILE
```

Program SOUBORY – Výpis jmen souborů a členů v zadané knihovně

Abychom mohli v programu vybrat žádoucí položky, zjistíme jména polí souboru QADSPOBJ. Zapíší se do souboru FIELDS v knihovně QTEMP.

```
DSPFFD FILE(QADSPOBJ) OUTPUT(*OUTFILE) OUTFILE(QTEMP/FIELDS)
RUNQRY QRYFILE((QTEMP/FIELDS))
```

Redukovaný výpis jmen polí z obrazovky

```
External   Field     Number  Decimal    Field Text
Field Name Length    Of      Positions  Description
           In Bytes  Digits
ODDCEN           1       0          0     Display century: 0=19xx, 1=20xx
ODDDAT           6       0          0     Display date (Job date format)
ODDTIM           6       0          0     Display time (HHMMSS)
ODLBNM          10       0          0     Library
ODOBNM          10       0          0     Object
ODOBTP           8       0          0     Object type
ODOBAT          10       0          0     Object attribute
ODOBFR           1       0          0     Storage freed: 0=Not freed, 1=Freed
...
```

Zde je tedy text programu SOUBORY s použitím jmen ODOBNM a ODOBAT.

```
Dcl &Library *char 10 'VZZAK'  /* zadaná knihovna */

Dclf QADSPOBJ opnid(FILES)
Dclf QAFDMBRL opnid(MEMBERS)
Ovrdbf file(QADSPOBJ) tofile(QTEMP/FILES)
Ovrdbf file(QAFDMBRL) tofile(QTEMP/MEMBERS)

/* Připravit výstup do souboru QTEMP/MEMBERS  */
Clof opnid(MEMBERS)   /* uzavřít otevřený soubor */
MonMsg CPF0000   /* ignorovat *ESCAPE chybu, když není otevřený */
ClrPfm FILE(QTEMP/MEMBERS)   /* smazat obsah souboru */
MonMsg CPF0000   /* ignorovat *ESCAPE chybu, když soubor neexistuje */

/* Vytvořit soubor se seznamem objektů typu *FILE */
DSPOBJD obj(&Library/*ALL) objtype(*FILE) output(*OUTFILE) +
        outfile(QTEMP/FILES)

DoUntil ('0') /* provedu tělo cyklu alespoň jednou */
    Rcvf opnid(FILES)   /* čtu záznam ze souboru */
    Monmsg msgid(CPF0864) exec(Leave)  /* na konci dat opustím cyklus */

    If (&FILES_ODOBAT = 'PF') Do  /* vybírám jen atribut PF – fyzický soubor */
        /* Vytvořit soubor se seznamem členů (members) tohoto souboru */
        DSPFD file(&Library/&FILES_ODOBNM) Type(*MBRLIST) +
                   output(*OUTFILE) outfile(QTEMP/MEMBERS) +
                   outmbr(*FIRST *ADD)
    EndDo   /* If */
EndDo   /* Do Until */

RunQry *n QTEMP/MEMBERS *printer formsize(255 200)   /* vytisknout výsledek */

EndPgm
```

Program zapsal kompletní výsledky do tiskového souboru QPQUPRFIL příkazem RUNQRY. Zde je redukovaný výpis výsledků:

```
File            Member     Number Of
                           Records

CENY            CENY              4
QCLSRC          ARITM            21
QCLSRC          CRTDTAQ          14
QCLSRC          CRTMENU2         21
QCLSRC          CYKLY            39
QCLSRC          DARA             36
QCLSRC          DQRECEIVE        18
...
QCLSRC          ZOBCENY          19
QCMDSRC         DQSEND            9
QCMDSRC         REFPGM           14
QDDSSRC         CENY             13
QDDSSRC         DSPFCENY         15
QDDSSRC         MENU1            33
QDDSSRC         MENU1QQ           6
QDDSSRC         MENU2            18
QDDSSRC         REF              11
QRPGLESRC       PREP_PROC        13
QRPGLESRC       RZOBCENY         14
REF             REF               0
```

Abychom mohli vypsat jen vybrané položky, zjistíme jména polí souboru MEMBERS, kam se zapsaly údaje o souborech a jejich členech.

```
DSPFFD FILE(MEMBERS) OUTPUT(*OUTFILE) OUTFILE(QTEMP/FIELDS)
RUNQRY QRYFILE((QTEMP/FIELDS))
```

Redukovaný výpis jmen polí z obrazovky

```
External   Field     Number  Decimal    Field Text
Field Name Length    Of      Positions  Description
           In Bytes  Digits
MLRCEN           1       0          0     Retrieval century:  0=19xx, 1=20xx
MLRDAT           6       0          0     Retrieval date: year/month/day
MLRTIM           6       0          0     Retrieval time: hour/minute/second
MLFILE          10       0          0     File
MLLIB           10       0          0     Library
MLFTYP           1       0          0     P=PF, L=LF, R=DDM PF, S=DDM LF
MLFILA           4       0          0     File attribute: *PHY or *LGL
MLMXD            3       0          0     Reserved
MLFATR           6       0          0     File attribute:  PF, LF, PF38, or LF38
MLSYSN           8       0          0     System Name (Source System, if file is DDM)
MLASP            2       3          0     Auxiliary storage pool ID:  1=System ASP
MLRES            4       0          0     Reserved
MLNOMB           3       5          0     Number of members
MLNAME          10       0          0     Member
MLNRCD           6      10          0     Current number of records
MLNDTR           6      10          0     Number of deleted records
MLSIZE           6      10          0     Data space and index size in bytes, -1=See MLSIZ2
...
```

Stejné informace dostaneme pomocí SQL.

```
STRSQL

select MLFILE, MLNAME, MLNRCD
from QTEMP/MEMBERS
```

```
File         Member           Number Of
                              Records
CENY         CENY                    4
QCLSRC       ARITM                  21
QCLSRC       CHGOWNER               31
QCLSRC       CRTDTAQ                14
QCLSRC       CRTMENU2               21
QCLSRC       CYKLY                  39
QCLSRC       DARA                   36
QCLSRC       DQRECEIVE              18
...
QCLSRC       ZOBCENY                19
QCMDSRC      DQSEND                  9
QCMDSRC      REFPGM                 14
QDDSSRC      CENY                   13
QDDSSRC      DSPFCENY               15
QDDSSRC      MENU1                  33
QDDSSRC      MENU1QQ                 6
QDDSSRC      MENU2                  18
QDDSSRC      REF                    11
QRPGLESRC    PREP_PROC              13
QRPGLESRC    RZOBCENY               14
REF          REF                     0
```

Změna vlastníka objektů v knihovně

Program CHGOWNER mění vlastníka všech objektů v zadané knihovně. Původní vlastník si však může vlastnictví vrátit. Pro změnu vlastníka platí určitá omezení, viz popis příkazu CHGOBJOWN.

```
     PGM      (&NEWOWN &LIB)

/*   Parametry                                                      */
     DCL      &NEWOWN *CHAR 10      /* nový vlastník */
     DCL      &LIB    *CHAR 10      /* knihovna */

/*   Deklarace souboru se seznamem objektů                          */
     DCLF       FILE(QADSPOBJ)

/*   Získání popisů všech objektů knihovny                          */
     DSPOBJD    OBJ(&LIB/*ALL) OBJTYPE(*ALL) +
                OUTPUT(*OUTFILE) OUTFILE(QTEMP/QADSPOBJ)
     OVRDBF     FILE(QADSPOBJ) TOFILE(QTEMP/QADSPOBJ)

/*   Čtu první záznam                                               */
     RCVF
     MONMSG CPF0000  EXEC(RETURN)  /* není-li žádný, končím */

/*   Není-li vlastník už shodný, změním ho                        */
     DOWHILE (&ODOBOW *NE &NEWOWN)
         CHGOBJOWN OBJ(&LIB/&ODOBNM) OBJTYPE(&ODOBTP) NEWOWN(&NEWOWN)
         RCVF   /* čtu další záznam */
         MONMSG CPF0000  EXEC(LEAVE)  /* konec dat, končím */
     ENDDO

     ENDPGM
```

Program se volá příkazem

```
CALL PGM(CHGOWNER) PARM((QUSER) (VZZAK))
```

Výsledek můžeme ověřit příkazem

```
WRKOBJ OBJ(VZZAK/*ALL)
```

a v seznamu volbou **5** u zvoleného objektu (zde u programu CHGOWNER)

```
Opt  Object      Type        Attribute
__   ARITM       *PGM        CLLE
5_   CHGOWNER    *PGM        CLLE
__   CRTDTAQ     *PGM        CLLE
     ...
```

```
                          Display Object Authority

Object . . . . . :   CHGOWNER        Owner  . . . . . :   QUSER
  Library  . . . . :     VZZAK       Primary group  . . :   *NONE
Object type  . . . :   *PGM          ASP device . . . . :   *SYSBAS

Object secured by authorization list  . . . . . . . . . . . :   *NONE

                         Object
User        Group       Authority
*PUBLIC                 *CHANGE
QUSER                   *ALL
```

Jak vytvořit nabídku (menu)

`STRSDA` s volbou `2. Design menus`

```
Type choices, press Enter.

  Source file . . . . . .   QDDSSRC___   Name, F4 for list

    Library . . . . . .     VZZAK_____   Name, *LIBL, *CURLIB

  Menu  . . . . . .         MENU1___     Name, F4 for list
```

Po stisku Enter:

```
 Work with menu image and commands  . . . . . .   Y   Y=Yes, N=No

 Work with menu help  . . . . . .   N   Y=Yes, N=No
```

Po volbě Y:

```
MENU1                           MENU1 Menu

Select one of the following:

     1.
     2.
     3.
     4.
     5.
     6.
     7.
     8.
     9.
    10.




Selection or command
F3=Exit                    F10=Work with commands         F12=Cancel
F13=Command area           F20=Reverse                    F24=More keys
```

Upravíme texty nabídek:

```
Select one of the following:

    '1. Display Object Description'
    '2. Work with Spooled Files'
    '3. Call Program'
    '4. Work with Active Jobs'



    '8. Start PDM'
```

Stiskneme F10 a zapíšeme příkazy k vybraným číslům:

```
                          Define Menu Commands

Menu . . . . . :   MENU1              Position to menu option . . . . .

Type commands, press Enter.

Option   Command
  01     ? DSPOBJD OBJ(*LIBL/CENY) OBJTYPE(*FILE) DETAIL(*BASIC)
_____________________________________
  02     ? WRKSPLF SELECT(*CURRENT)
_____________________________________
  03     ? CALL
_____________________________________
  04     ? WRKACTJOB OUTPUT(*)
_____________________________________
  05     _____________________________________________________________________
_____________________________________
  06     _____________________________________________________________________
_____________________________________
  07     _____________________________________________________________________
_____________________________________

                                                                        More...
F3=Exit       F11=Defined only options      F12=Cancel         F24=More keys
```

Po stisku F3:

```
 Work with menu image and commands  . . . . . .   N   Y=Yes, N=No

 Work with menu help  . . . . . .   N   Y=Yes, N=No
```

Po stisku Enter:

```
                            Exit SDA Menus

File  . . . . . :   QDDSSRC             DDS member  . . . . . :   MENU1
  Library . . . . :     VZZAK             Commands member . . . :   MENU1QQ

Type choices, press Enter.

  Save new or updated menu source  . . . .   Y            Y=Yes, N=No
    For choice Y=Yes:
      Source file  . . . . . .   QDDSSRC      Name,
                                               F4 for list
        Library  . . . . . .   VZZAK        Name, *LIBL, *CURLIB
      Text . . . . . .

      Replace menu members . . . . . .   Y            Y=Yes, N=No

  Create menu objects  . . . . . .   Y            Y=Yes, N=No
    For choice Y=Yes:
      Prompt for parameters  . . . . .   N            Y=Yes, N=No
      Object library . . . . . .   VZZAK        Name, *CURLIB
      Replace menu objects . . . . . .   Y            Y=Yes, N=No

F3=Exit     F4=Prompt     F12=Cancel
```

Po stisku Enter vznikne

v knihovně

- nabídka (menu) `MENU1` (objekt typu `*MENU`)
  a v souboru `QDDSSRC`
- zdrojový člen `MENU1QQ` pro soubor zpráv `MENU1` (typu `MNUCMD`)
- zdrojový člen `MENU1` pro obrazovkový soubor `MENU1` (typu `MNUDDS`)

Volání nabídky

```
GO MENU(*LIBL/MENU1)
```

nebo prostě

```
GO MENU1
```

Zrušení nabídky (všech jejích objektů)

```
DLTMNU MENU(MENU1) DLTREFOBJ(*ALL)
```

Jiný způsob vytvoření nabídky

Obrazovkový soubor MENU2 – Zdrojový text MENU2 v souboru DDSSRC.

```
     A                                      DSPSIZ(24 80 *DS3)
     A                                      CHGINPDFT
     A                                      INDARA
     A          R MENU2
     A                                      CF03
     A                                    1  2'MENU2'
     A                                       COLOR(BLU)
     A                                    1 34'MENU2 Menu'
     A                                       DSPATR(HI)
     A                                    3  2'Select one of the following:'
     A                                       COLOR(BLU)
     A                                    5  7'1. Display Object Description'
     A                                    6  7'2. Work with Spooled Files'
     A                                    7  7'3. Call Program'
     A                                    8  7'4. Work with Active Jobs'
     A                                   13  7'8. Start PDM'
     A                                  019  2'Selection or command             -
     A                                           '
```

Program CRTMENU2 - Vytvoření nabídky (objekt typu *MENU s atributem *DSPF)

```
             PGM
             DLTOBJ     OBJ(MENU2) OBJTYPE(*FILE)
             MONMSG     MSGID(CPF0000)
             CRTDSPF    FILE(*CURLIB/MENU2) SRCFILE(QDDSSRC)
/*   Vytvoření souboru zpráv (message file object) typu *MSGF        */
             DLTOBJ     OBJ(MENU2) OBJTYPE(*MSGF)
             MONMSG     MSGID(CPF0000)
             CRTMSGF    MSGF(*CURLIB/MENU2) TEXT('Message file') +
                          SIZE(20 2)
/*   Přidání popisů zpráv do souboru zpráv                           */
             ADDMSGD    MSGID(USR0001) MSGF(MENU2) MSG('?DSPOBJD')
             ADDMSGD    MSGID(USR0002) MSGF(MENU2) MSG('?wrksplf')
             ADDMSGD    MSGID(USR0003) MSGF(MENU2) MSG('?call')
             ADDMSGD    MSGID(USR0004) MSGF(MENU2) MSG('?wrkactjob')
             ADDMSGD    MSGID(USR0008) MSGF(MENU2) MSG('?STRPDM')
/*   Vytvoření nabídky MENU2 - objekt typu *MENU s atributem DSPF    */
             DLTOBJ     OBJ(MENU2) OBJTYPE(*MENU)
             MONMSG     MSGID(CPF0000)
             CRTMNU     MENU(*CURLIB/MENU2) TYPE(*DSPF) DSPKEY(*YES)
             ENDPGM
```

Volání nabídky

```
GO MENU(*LIBL/MENU2)
```

Zrušení nabídky (všech objektů)

```
DLTMNU MENU(MENU2) DLTREFOBJ(*ALL)
```