Kde hledat informace:

- http://cs.felk.cvut.cz/vms
- http://cs.felk.cvut.cz/vms/cz.html
- příkaz `HELP`
  - "?" pro zopakování
  - "*" všechna podtémata
  - podrobnosti viz `help help`
- tištěné manuály na chodbě 2. patra
- elektronická konference `vms@cs.felk.cvut.cz`

Jak se přihlásit

- terminál (terminálový server, LAT protocol)
- ssh, telnet
- grafický režim DEC windows
  - `xhost cs.felk.cvut.cz` (`xhost +`)
  - `telnet/ssh cs.felk.cvut.cz`
  - `xwin, xwin my.host.cvut.cz 1`
- XDM

Textové editory

- EDT a EVE, klasické OpenVMS editory (Ctrl/Z, ex)
- pico
- vi (vile)

Příkazový jazyk DCL (Digital Command Language)

Příkazy: interní, externí, cizí

Interní a externí příkazy

Tabulka příkazů v interpretu, definice parametrů a kvalifikátorů.
=> interpret zná jména, počty a typy parametrů a kvalifikátorů
=> zkratky jmen příkazů a kvalifikátorů

Jednotný formát příkazu:

command parameter/qualifier parameter/qualifier
command/qualifier parameter
parametry povinné/nepovinné

kvalifikátory poziční/globální:

```
print a.txt,b.txt/copies=2
print/copies=2 a.txt,b.txt
```

Řídicí klávesy

Ctrl/S	pozastavení výstupu na terminál
Ctrl/Q	obnovení výstupu
Ctrl/O	potlačení/obnovení výstupu na terminál (dočasné zahazování výstupu)
Ctrl/A	přepne režim vkládání a přepisování znaků
Ctrl/H	home, přesun kurzoru na začátek řádku
Ctrl/E	end, přesun kurzoru na konec řádku
Ctrl/T	stav procesu - jméno procesu a programu, čas CPU, io operace, výpadky stránek, pracovní sada
Ctrl/Y	přerušení běhu programu Ctrl/Y spawn (vytvoření podprocesu) příkazy prováděné v kontextu podprocesu logout continue

Proces - adresní prostor je rozdělen na část pro interpret příkazů a pro aplikační program (image).

Chybové zprávy

```
%DCL-W-IVVERB, unrecognized command verb
```

help/message ivverb

DCL	kdo chybu hlásí, označení programu nebo části operačního systému
W	stupeň závažnosti: W warning, E error, I information
IVVERB	kód chyby

Specifikace souboru

```
node::device:[directory]filename.type;version
```

Příklady:

cs::dka100:[student.xnovak]dopis.txt;31
student$device:[student.xnovak]dopis.txt
[student.xnovak]dopis.txt
dopis.txt

node	jméno počítače v rámci sítě DECNET
device	označení zařízení/disku
directory	adresář je soubor typu `.dir`
version	každým zapsáním souboru na disk vzniká nová verze

Verze souboru

- při počátečním vytvoření vznikne verze 1
- při otevírání souboru pro čtení se implicitně vyhledá nejvyšší existující
- při zápisu (vytvoření nové verze) vznikne verze o 1 vyšší, než je nejvyšší existující
- znakem "*" se vyberou/označí všechny verze
- verze 0 je symbolický odkaz na aktuální (nejvyšší) verzi
- verze -1 je odkaz na předchozí verzi (záložní)

filename.type

Klasický OpenVMS:	tečka není součástí jména souboru, ale oddělovačem mezi jménem a typem
filesystém ODS-2	jméno i typ max 39 znaků (39.39)
	povolena písmena, číslice, dolar, pomlčka, podtržítko
	velikost písmen nerozhoduje, konvertuje se na velká
Nový OpenVMS	delší jména, celkem až 255 znaků
filesystém ODS-5	podpora Unicode
	stále je tečka oddělovačem jména a typu. Ve jménu a typu může být "druhotná" tečka
	"neobvyklé" znaky se prefixují pomocí ^
	soubor.tar^.gz
	tady^20mezera.txt Unicode ^U012F
	case preserving
	=> kompatibilita s Win32, lepší podpora pro Javu

Adresáře

show default	aktuální adresář
create/directory [.texty]	vytvoření podadresáře texty
create/directory [texty]	vytvoření adresáře texty v hlavním adresáři disku
set default [.texty]	změna aktuálního adresáře
cd [.texty] cd texty	místní program pro změnu adresáře, rozumí VMS i Unixu

Soubory

directory *.%a	libovolné jméno, typ dlouhý dva znaky, končící na A
dir/page/size/date/owner/protection/acl	některé často používané kvalifikátory
dir [-]*.txt	soubory *.txt v nadřízeném adresáři
dir [-.data]	sousední adresář `data` (o jednu úroveň vzhůru, pak do podadresáře data)
dir [--.xyz]	o dvě úrovně vzhůru, pak podadresář `xyz`
dir [...]	aktuální adresář a všechny jeho podadresáře
delete [□]*.vms*.dat;*	smaže z aktuáního adresáře a všech jeho podadresářů soubory, která mají ve svém jméně `vms`, jsou typu dat, všechny verze
dir [...source]	všechny podadresáře source, jen tyto podadresáře
dir [...source...]	všechny podstromy aktuálního adresáře, které začínají podadresářem source
copy alfa::[zdroje□]*.c beta::[src...]	kopírování stromu obsahujícího soubory *.c z počítače alfa a adresáře zdroje na počítač beta, adresář src (a odpovídající podadresáře)

copy alfa::[zdroje□]*.c beta::[src]	totéž, bez vzniku podadresářů
copy/log [aaa.bbb]*.* []	kopíruj do aktuálního adresáře, /log informuje o průběhu

Symboly

- obecně: proměnná v příkazovém jazyce, užívá se při tvorbě příkazových procedur
- alias □ jiné jméno příkazu, lze i s parametry a kvalifikátory => tvorba zkratek
- cizí příkaz □ alias pro spouštění programu, který není integrální součástí VMS (není definován v tabulkách

  příkazového jazyka)

První identifikátor na příkazovém řádku se zkouší nahradit hodnotou symbolu.

Alias:

```
du:=dir [...]/siz=all/tot
ss:=show symbol
ss du
```

Cizí příkaz:

```
vi*le:=$other_sw:vile
```

Zkrácení jména aliasu a cizího příkazu:

Hvězdička ukazuje, kam až lze při používání alias nebo cizí příkaz zkracovat.

Vznik symbolu

přiřazovací příkaz	= nebo ==	vpravo je výraz
přiřazení řetězu znaků	:= nebo :==	vpravo jsou jen znaky

Vedlejší efekt: obsah příkazového řádku se konvertuje na velká písmena. Uzavření do uvozovek tomu zabrání.

```
ss:=show symbol
sh sym ss => SS = "SHOW SYMBOL"
ss="show symbol"  (stringová konstanta)
sh sym ss => SS = "show symbol"
NE ss=show symbol      (toto není platný výraz)
```

lokální symbol	= nebo :=	Platí jen na dané úrovni příkazového jazyka, to jest: Vznikne-li symbol v příkazové proceduře, po jejím ukončení symbol zanikne.
globální symbol	== nebo :==	Existuje i po skončení příkazové procedury. V login scriptu `login.com` lze definovat vlastní aliasy, definice musí být globální.

Zrušení symbolu

```
delete/symbol  ss
del/sym/global ss     pro globální symbol
```

Typy symbolů

Symboly obsahují celá čísla (4 byty, kladná i záporná) nebo řetěz znaků (max 255)

```
a=17
sh sym a     => A = 17   Hex = 00000011  Octal =
00000000021
a=a+1
sh sym a     => A = 18   Hex = 00000012  Octal =
00000000022
ALE
a:=a+1
sh sym a     =>  A = "A+1"
```

Symbol má i booleovskou hodnotu true/false

True	lichá čísla (vazba na návratové/stavové kódy)
	řetěz znaků začínající na T nebo Y
False	sudá čísla včetně nuly
	řetěz znaků nezačínající T ani Y

Příklady na práci se symboly

```
I=11     J=%x15    ! desitkova a sestnactkova konstanta
J = (I+2)*3
SO = "sobota"
NE = "nedele"
Spojení řetězů
WK = SO+NE
    => "sobotanedele"
WK = SO + " a " + NE
```

Toto není výraz, zjednodušené přiřazení řetězu znaků

```
WK := SO + NE
    => WK = "SO+NE"
```

Totéž, navíc substituce

```
WK := 'SO' a 'NE'
    => WK = "SOBOTA A NEDELE"
WK := SO'NE'
    => WK = "SONEDELE"
```

Substituci lze provést i uvnitř stringové "konstanty", úvodní apostrof musí být zdvojený

```
WK := "''SO' a ' 'NE'"
    => WK = "sobota a nedele"
```

Substituci lze provést kdekoliv jakkoliv, pak teprve se analyzuje správnost příkazu.

```
ITEM'I' = "''I'. den"
    => ITEM11 = "11. den"
```

Odčítání řetězů - vyhledej první výskyt zleva

```
X = WK - "a"
    => X = "sobot a nedele"
```

Náhrada, toto není typický způsob práce. Na práci s textem máme lexikální služby

```
X[0,1] := r   (offset,size)
    => X = "Robot a nedele"
```

Pro číselné symboly se počítá po jednotlivých bitech, pole 32 bitů.