

OpenVMS Fundamentals, Version 7.2

Lab Guide

Rev. 0.30

Training



COMPAQ

OpenVMS Fundamentals, Version 7.2

Lab Guide

Rev. 0.30

Training



COMPAQ

© 2000 Compaq Computer Corporation

COMPAQ and the Compaq logo, AlphaServer, DECnet, HSJ, PDP, and VAX are registered in the U.S. Patent and Trademark Office.

Alpha, OpenVMS, and Tru64 are trademarks of Compaq Information Technologies Group, L.P.

Microsoft, Windows, Windows NT, and Visual C++ are trademarks of Microsoft Corporation.

Motif, OSF/1, UNIX, the “X” Device, IT DialTone, and The Open Group are trademarks of The Open Group.

All other product names mentioned herein may be trademarks of their respective companies.

Compaq shall not be liable for technical or editorial errors or omissions contained herein. The information in this document is subject to change without notice.

THE INFORMATION IN THIS PUBLICATION IS SUBJECT TO CHANGE WITHOUT NOTICE AND IS PROVIDED “AS IS” WITHOUT WARRANTY OF ANY KIND. THE ENTIRE RISK ARISING OUT OF THE USE OF THIS INFORMATION REMAINS WITH RECIPIENT. IN NO EVENT SHALL COMPAQ BE LIABLE FOR ANY DIRECT, CONSEQUENTIAL, INCIDENTAL, SPECIAL, PUNITIVE, OR OTHER DAMAGES WHATSOEVER (INCLUDING WITHOUT LIMITATION, DAMAGES FOR LOSS OF BUSINESS PROFITS, BUSINESS INTERRUPTION, OR LOSS OF BUSINESS INFORMATION), EVEN IF COMPAQ HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES AND WHETHER IN AN ACTION OF CONTRACT OR TORT, INCLUDING NEGLIGENCE.

The limited warranties for Compaq products are exclusively set forth in the documentation accompanying such products. Nothing herein should be construed as constituting a further or additional warranty.

OpenVMS Fundamentals, Version 7.2

Lab Guide

September 2000

Lab Module 1 — Lab Exercises

Lab Exercises	L1-1
Optional Lab Exercises	L1-8

Lab Module 2 — Lab Solutions

Lab Solutions.....	L2-1
Optional Lab Solutions	L2-45

Lab Exercises

1. Use the `$ SHOW MEMORY` command to determine how much physical memory is on your system.
2. Determine which disk devices are available on your system.
 - a. How much space is available on each disk?
 - b. How much total storage is on each disk?
3. Enter the following commands:
`$ SHOW NETWORK`
`$ TCPIP SHOW INTERFACE`
What is your system's IP address?
4. Determine your process and user name.
 - a. Can you change your process name (use `$HELP SET`)?
 - b. Can you change your user name?
5. Determine your process identification (PID).
6. Use your PID to locate your process in a `$ SHOW SYSTEM` display.
 - a. What is the scheduling state of your process?
7. Determine which privileges are assigned to your process.
 - a. Use the `$SET PROCESS/PRIVILEGE=NOTMPMBX` command to turn off TMPMBX privilege.
 - b. Enter the `$SPAWN` command.
What happens?
 - c. Turn TMPMBX privilege back on, using:
`$SET PROCESS/PRIVILEGE=TMPMBX`.
 - d. Enter the `$SPAWN` command again.
What happens now?
What is your Process Identification?
 - e. Use the `$LOGOUT` command to return to your original process.

8. Change your password to something that you can remember.
 - a. Can you change your password back to its original setting?
9. Determine who is logged in.
10. Use \$HELP Hints to determine a method for creating a file without using an editor.
 - a. Create a file called WRITER.LIS.
 - b. Enter the following commands and compare the behavior:
\$ TYPEWRITER
\$ TYPE WRITER
\$ TYPE WRITER.LIS
 - c. Create the file NAMES.DAT and enter the command \$ TYPE NAMES.
What happens?
11. Enable insert mode for command line editing.
12. Use the arrow keys to recall the command that you used to create the file called NAMES.DAT. Change the command so that it creates a file called BLAME.DAT.
13. When entering a command, what do the keys Ctrl/A, Ctrl/H, and Ctrl/E do?
14. Enter the command \$ MONITOR PROCESS
 - a. Enter Ctrl/Y. What happens?
 - b. Enter the command \$ CONTINUE
 - c. Enter Ctrl/W. What happens?
15. Use the \$ SET TERMINAL command to change your terminal width to 132 columns. Change it back to 80 columns.
16. Enter the command \$ SET TERMINAL/ECHO
 - a. Enter the command \$ SET TERMINAL/NOECHO
 - b. What happens when you enter commands?
 - c. Use the arrow keys to bring back the \$SET TERMINAL/ECHO command.
17. Use the \$ SET TERMINAL/INQUIRE command to set your terminal attributes, based on your terminal type.
18. Use the \$ SHOW SYSTEM/PAGE=SAVE command to view the processes on the system one screen at a time. Back up to a previous screen.

19. Enter the command `$ TYPE/PAGE NL:.` What happens?
20. Display one of your files using the `$ TYPE` command.
 - a. Enter the command `$SHOW SYMBOL $SEVERITY`
 - b. What value do you see?
 - c. Enter the command `$ COPY SYS$SYSTEM:SYSUAF.DAT []`
 - d. Enter the command `$SHOW SYMBOL $SEVERITY`
 - e. What value do you see?
 - f. Use `$ HELP/MESSAGE` to interpret the error message.
21. Get comfortable with using the EVE editor by creating a file with the names of five friends.
22. Enable the EDT keypad in EVE. Press the PF2 key (/ on a PC keypad) to get help. Determine how to select text, cut it, and paste it.
 - a. Does the EDT keypad allow you to copy selected text?
 - b. Define `<GOLD> C` to perform a copy command.
 - c. Define `<GOLD> M` to center a line.
 - d. Save your key definitions in an EVE section file (Hint: `SAVE EXTENDED EVE file`).
 - e. Exit the editor and use your EVE section file to recover your key definitions.
 - f. Use the command `$ DEFINE EVE$KEYPAD EDT` to automatically enable the EDT keypad when you activate the editor.
23. Use the mail utility to send a message to your instructor.
24. Send a mail message to another user.
25. Create a distribution list and use it to send messages to multiple users.
26. Create a mail subdirectory.
27. Use the PHONE utility to talk to another student.
28. Find all files in your directory that have names starting with the letter N or W.
29. Find all files in your directory that have file types of `.DAT`.
30. Find all files in your directory that have exactly five character file names with any file type.

31. Create a subdirectory.
 - a. Copy some files into the subdirectory.
 - b. Set your default directory to your subdirectory.
 - c. Create a file in the subdirectory.
 - d. Verify that the new file is in the subdirectory.
 - e. Set your default to your top-level directory (UFD).
 - f. Display the file in the subdirectory, without setting your default directory to the subdirectory.
 - g. Delete the subdirectory.
32. Create the following directories under your UFD:
[.SUB1]
[.SUB1.SUBSUBA]
[.SUB1.SUBSUBB]
 - a. Set your default directory to [.sub1.subsuba].
 - b. From this directory list the directory [.sub1.subsubb], using wildcarding.
 - c. Using wildcard characters, set your default directory back to your UFD.
33. Determine whether the student sitting next to you is in the same UIC group as you.
34. Set up your directory so users in your group can only list your directory if they know specific files in your directory by name.
35. Set up your directory so that the student sitting next to you can wildcard list your directory, but no one else in your group can.
36. Set your default protection so that new files are created with no access for group. Verify the change.
37. Use \$ HELP CREATE/DIRECTORY to determine how to set a version limit on a directory.
 - a. Create a subdirectory with a version limit of 3.
 - b. Create a file in the subdirectory.
 - c. Verify that the new file has a version limit of 3.
 - d. Make four copies of the file.
 - e. What happens?

38. Use `$ HELP SET FILE/VERSION_LIMIT` and `$ HELP SET DIRECTORY/VERSION_LIMIT` to determine how to set version limits on files in your UFD. Set a version limit of 5 for all existing and new files in your UFD. Clean up any files with more than five versions.
39. Find all occurrences of the string `PCBDEF` in the file `SYS$LIBRARY:LIB.REQ`.
40. Create a logical name identifying one of your subdirectories.
 - a. Use the logical name to list the subdirectory.
41. Create a logical name identifying one of your files.
 - a. Use the logical name to display the file.
42. Define the logical name `EVE$KEYPAD` as `EDT` in your job logical name table.
 - a. Use the `$ SPAWN` command to verify that the logical name is visible to all processes in your job.
43. Set your default directory to another location. Use the logical name `SYS$LOGIN` to get back to your login directory.
44. Create a file called `LAST.DAT` containing your last name. Create a file called `FIRST.DAT` containing your first name. Create a logical name called "NAME" defined as `LAST.DAT` in the process logical name table. Create a logical name called "NAME" defined as `FIRST.DAT` in the job logical name table.
 - a. Enter the command: `$ TYPE NAME`
 - b. Which file is displayed? Why?
45. Create a symbol called `EDIT` that causes the section file created in lab 23 to be used automatically.
46. Create a symbol called `CLEAR` that clears your screen.
47. Create a symbol called `TYPE` that uses the qualifier `/PAGE=SAVE` to display a file one screen at a time.
 - a. Can you display a file without the `/PAGE=SAVE` qualifier and without deleting the `TYPE` symbol?
48. Create a symbol called `X` with the numeric value 32 assigned to it.
 - a. Use the `$SHOW SYMBOL` command to determine the hexadecimal equivalent value for 32.
 - b. Delete the symbol.

49. View all global symbols.
50. Create a symbol called TAIL that performs a \$TYPE/TAI=10 command. The symbol should work if TA, TAI, or TAIL is entered.
 - a. Delete the symbol.
51. Create a symbol. Spawn a subprocess.
 - a. Can you see the symbol?
 - b. Can you create a symbol in the subprocess that is visible to the parent?
52. Define the key KP1 (keypad number 1) to enter a \$SHOW SYSTEM/PAGE=SAVE command.
53. Define the key KP2 to enter a \$DIRECTORY/DATE/PAGE=SAVE command.
 - a. Delete the key definition.
54. Create a LOGIN.COM command procedure that:
 - a. Creates a symbol to activate the editor with a TPU section file.
 - b. Creates symbols for:
 - 1) A MORE command that displays SYS\$INPUT one screen at a time.
 - 2) A CLEAR command that clears the screen.
 - c. Creates key definitions of your choice.
 - d. Creates a logical name pointing to a directory containing command procedures.
 - e. Enters a \$ SET TERMINAL/INQUIRE command, only if you are logging interactively.
 - f. Test your LOGIN.COM.
55. The C compiler keeps special files called *headers* in a text library. The following command extracts a header called `ssdef.h` and displays it on your screen:

```
$ libr/extract=ssdef/out=sys$output/text sys$library:sys$starlet_c
```

- a. Try the command.
- b. The output spills across the screen very quickly. Use the PIPE command and your MORE symbol to display the output one screen at a time.
- c. Write a command procedure that accepts a header file name through the P1 parameter and performs the function described in step b for any header. Try it with `iodef` and `jpidef`. (For a list of headers use: `$LIBRARY/LIST/TEXT SYS$LIBRARY:SYS$STARLET_C`).

56. Create a directory called [.backups].
 - a. Back up all files modified since midnight today to a save set in the directory. Note: exclude all files in the [.backups] directory from your backup.
 - b. List your save set.
 - c. Create a command procedure that backs up all files modified since midnight today in a file called [.backups]*dow*.BCK, where *dow* is the current day of week. The following lexical function returns the current day of week: `f$cvtime("TODAY", , "WEEKDAY")`.
57. Create the directories [.a] and [.a.b]. Put a couple of files in each directory. Use BACKUP to replicate the files in [.a] in a directory called [.x].
58. Create a command procedure that displays your name. Submit it to run in batch. Examine the log file.
59. Submit the command procedure created in problem 56, part c to run in batch.
60. Submit the command procedure created in problem 56, part c to run in batch to run after 11:30 PM today.
 - a. Verify that the job is scheduled to run.
 - b. Remove the entry.
61. Create the following command procedure:

```
$ create oops.com
$loop: goto loop
Ctrl/Z
$
```

 - a. Submit the command procedure to run in batch.
 - b. Use the \$SHOW SYSTEM/BATCH command to find the process running your batch job.
 - c. Stop the process using the \$STOP *process-name* or \$STOP/ID=*pid* command.
62. If a printer is available, print a file. Print two copies of the file with a header.

Optional Lab Exercises

1. Write a command procedure that searches files for a string and displays the names of files containing the matching string, but not the actual matching lines.
2. Write a command procedure that displays the number of images run and total page faults incurred by your process.
3. Write a command procedure that:
 - a. Saves the current default directory.
 - b. Sets default to a new directory.
 - c. Waits for one minute.
 - d. Generates an error.
 - e. Sets the default directory back to the original on errors and Ctrl/Y.
4. Write a command procedure that displays the percentage of space in use and available for all mounted disks.
5. Write a command procedure that edits, compiles, and links a program.

Lab Solutions

1. Use the \$ SHOW MEMORY command to determine how much physical memory is on your system.

\$ show memory/physical

System Memory Resources on 10-APR-1999 23:01:39.10

Physical Memory Usage (pages):	Total	Free	In Use	Modified
Private Memory (96.00Mb)	12288	4032	7522	734
Shared Memory (64.00Mb)	8192	8077	115	

Of the physical pages in use, 3497 pages are permanently allocated to OpenVMS.

\$

Total physical memory is 160Mb.

2. Determine which disk devices are available on your system.

```
$ sh dev d/mount
```

Device Name	Device Status	Error Count	Volume Label	Free Blocks	Trans Count	Mnt Cnt
ELLIS\$DKA200:	Mounted	0	AXP71FT2	238995	368	1

```
$
```

- a. How much space is available on each disk?

238995 blocks (512 bytes) are available.

- b. How much total storage is on each disk?

```
$ show device/full dka200
```

Disk ELLIS\$DKA200:, device type RZ26L, is online, mounted, file-oriented device, shareable, available to cluster, error logging is enabled.

Error count	0	Operations completed	22868
Owner process	""	Owner UIC	[SYSTEM]
Owner process ID	00000000	Dev Prot	S:RWPL,O:RWPL,G:R,W:R
Reference count	242	Default buffer size	512
Total blocks	2050860	Sectors per track	83
Total cylinders	3089	Tracks per cylinder	8
Volume label	"AXP71FT2"	Relative volume number	0
Cluster size	3	Transaction count	368
Free blocks	238995	Maximum files allowed	256357
Extend quantity	5	Mount count	1
Mount status	System	Cache name	"_ELLIS\$DKA200:XQPCACHE"
Extent cache size	64	Maximum blocks in extent cache	23899
File ID cache size	64	Blocks currently in extent cache	3567
Quota cache size	0	Maximum buffers in FCP cache	820
Volume owner UIC	[1,1]	Vol Prot	S:RWCD,O:RWCD,G:RWCD,W:RWCD

Volume Status: ODS-2, subject to mount verification, protected subsystems enabled, file high-water marking, write-through caching enabled.

```
$
```

The volume has 2050860 total blocks.

3. Enter the following commands:

\$ SHOW NETWORK

\$ TCPIP SHOW INTERFACE

\$ show network

Product: DECNET Node: ELLIS Address(es): 1.24

Product: TCP/IP Node: ellis.zzzzzzzzzz Address(es): 192.206.126.18

0

\$ tcpip show interface

Interface	IP_Addr	Network mask	Packets		MTU
			Receive	Send	
SE0	192.206.126.180	255.255.255.0	4984	4819	1500
LO0	127.0.0.1	255.0.0.0	5	5	0

\$

- a. What is your system's IP address?

This node's IP address is 192.206.126.180.

4. Determine your process and user name.

\$ show process

```
10-APR-1999 23:07:34.73  User: STUDENT7      Process ID:  00000126
                          Node: ELLIS        Process name: "STUDENT7"
```

```
Terminal:                TNA5: (Host: 192.206.126.197  Port: 1345)
User Identifier:         [BRUDEN,STUDENT7]
Base priority:          4
Default file spec:      SYS$SYSDEVICE:[STUDENT7]
Number of Kthreads: 1
```

Devices allocated: ELLIS\$TNA5:

\$

This process has a user name of STUDENT7 and a process name of STUDENT7.

a. Can you change your process name (use \$HELP SET)?

Yes, you can.

\$ set process/name="Bruce Ellis"

\$ show process

```
10-APR-1999 23:08:57.01  User: STUDENT7      Process ID:  00000126
                          Node: ELLIS        Process name: "Bruce Ellis"
```

```
Terminal:                TNA5: (Host: 192.206.126.197  Port: 1345)
User Identifier:         [BRUDEN,STUDENT7]
Base priority:          4
Default file spec:      SYS$SYSDEVICE:[STUDENT7]
Number of Kthreads: 1
```

Devices allocated: ELLIS\$TNA5:

\$

b. Can you change your user name?

No, you cannot.

5. Determine your process identification (PID).

\$ show process

```
10-APR-1999 23:08:57.01  User: STUDENT7      Process ID:  00000126
                          Node: ELLIS        Process name: "Bruce Ellis"
```

Terminal: TNA5: (Host: 192.206.126.197 Port: 1345)

User Identifier: [BRUDEN,STUDENT7]

Base priority: 4

Default file spec: SYS\$SYSDEVICE:[STUDENT7]

Number of Kthreads: 1

Devices allocated: ELLIS\$TNA5:

\$

This process has a PID of 00000126.

6. Use your PID to locate your process in a \$ SHOW SYSTEM display.

\$ show system

```
OpenVMS V7.2 on node ELLIS 10-APR-1999 23:10:52.16 Uptime 0 13:30:26
```

Pid	Process Name	State	Pri	I/O	CPU	Page flts	Pages	
00000101	SWAPPER	HIB	16	0	0 00:00:00.16	0	0	
00000106	LANACP	HIB	13	68	0 00:00:00.24	135	111	
00000108	FASTPATH_SERVER	HIB	10	9	0 00:00:00.13	69	82	
00000109	IPCACP	HIB	10	10	0 00:00:00.06	31	43	
0000010A	ERRFMT	HIB	8	1554	0 00:00:04.92	87	101	
0000010C	OPCOM	HIB	7	514	0 00:00:03.30	88	46	
0000010D	AUDIT_SERVER	HIB	10	77	0 00:00:00.35	150	163	
0000010E	JOB_CONTROL	HIB	10	63	0 00:00:00.17	55	74	
0000010F	QUEUE_MANAGER	HIB	10	273	0 00:00:00.83	135	170	
00000110	SECURITY_SERVER	HIB	10	36	0 00:00:02.39	322	326	
00000111	DNS\$ADVER	LEF	5	3257	0 00:00:08.30	348	243	
00000112	LES\$ACP_V30	HIB	9	127	0 00:00:00.46	78	90	
00000113	NET\$ACP	HIB	6	45	0 00:00:00.32	141	131	
00000114	REMACP	HIB	8	10	0 00:00:00.05	29	16	
00000115	NET\$EVD	HIB	6	20	0 00:00:00.84	244	241	
00000116	DTSS\$CLERK	LEF	12	645	0 00:00:02.60	142	171	
00000117	TP_SERVER	HIB	10	3256	0 00:00:29.88	76	94	
00000118	LATACP	HIB	14	8	0 00:00:00.79	73	58	
00000119	TCPIP\$INET_ACP	HIB	8	113	0 00:00:00.42	266	137	
0000011A	TCPIP\$PWIP_ACP	HIB	10	53	0 00:00:00.26	132	145	
0000011B	TCPIP\$NTP	LEF	9	48509	0 00:00:48.07	338	192	N
0000011D	VPM_SERVER	HIB	15	14	0 00:00:01.57	217	186	
0000011F	DECW\$SERVER_0	HIB	8	798	0 00:00:05.20	666	872	
00000120	DTLOGIN	LEF	5	143	0 00:00:01.02	358	244	
00000121	DTGREET	LEF	4	464	0 00:00:08.45	805	365	S
00000126	Bruce Ellis	CUR	0 4	432	0 00:00:02.31	1317	124	

\$

I am represented by the last line.

a. What is the scheduling state of your process?

We are in the CUR (current) scheduling state.

7. Determine which privileges are assigned to your process.

\$ show process/privilege

```
10-APR-1999 23:12:23.54  User: STUDENT7      Process ID: 00000126
                          Node: ELLIS        Process name: "Bruce Ellis"
```

Authorized privileges:

```
NETMBX      TMPMBX
```

Process privileges:

```
NETMBX      may create network device
TMPMBX      may create temporary mailbox
```

Process rights:

```
STUDENT7      resource
INTERACTIVE
REMOTE
```

System rights:

```
SY$NODE_ELLIS
```

\$

I have TMPMBX and NETMBX privilege.

- a. Use the \$SET PROCESS/PRIVILEGE=NOTTMPMBX command to turn off TMPMBX privilege.

\$ set process/priv=notmpmbx

\$

- b. Enter the \$SPAWN command. What happens?

\$ spawn

%SYSTEM-F-NOPRIV, insufficient privilege or object protection violation

\$

The command fails due to insufficient privilege.

- c. Turn TMPMBX privilege back on, using:
\$SET PROCESS/PRIVILEGE=TMPMBX.

- d. Enter the \$SPAWN command again. What happens now?

```
$ set process/priv=tmpmbx
```

```
$ spawn
```

```
%DCL-S-SPAWNED, process STUDENT7_1 spawned
```

```
%DCL-S-ATTACHED, terminal now attached to process STUDENT7_1
```

```
$
```

The SPAWN command works.

What is your PID?

```
$ sh process
```

```
10-APR-1999 23:15:08.64  User: STUDENT7      Process ID:  00000127
                        Node: ELLIS         Process name: "STUDENT7_1"
```

Terminal:

```
User Identifier:  [BRUDEN,STUDENT7]
```

```
Base priority:    4
```

```
Default file spec: SYS$SYSDEVICE:[STUDENT7]
```

```
Number of Kthreads: 1
```

```
$
```

Our process ID is 00000127, different from our login process.

- e. Use the \$LOGOUT command to return to your original process.

```
$ logout
```

```
Process STUDENT7_1 logged out at 10-APR-1999 23:16:39.66
```

```
%DCL-S-RETURNED, control returned to process Bruce Ellis
```

```
$
```

8. Change your password to something that you can remember.

```
$ set password
```

```
Old password:
```

```
New password:
```

```
Verification:
```

```
$
```

- a. Can you change your password back to its original setting?

No.

```
$ set password
```

```
Old password:
```

```
New password:
```

```
Verification:
```

```
%SYSTEM-F-PWDWEAK, password is too easy to guess; please choose another string
```

```
$
```

9. Determine who is logged in.

```
$ show user
```

```
OpenVMS User Processes at 10-APR-1999 23:19:23.49
```

```
Total number of users = 2, number of processes = 2
```

Username	Interactive	Subprocess	Batch
----------	-------------	------------	-------

BAE	1		
-----	---	--	--

STUDENT7	1		
----------	---	--	--

```
$
```

10. Use \$HELP Hints to determine a method for creating a file without using an editor.

- a. Create a file called WRITER.LIS.

```
$ create writer.lis
```

```
Paperback?
```

```
Exit
```

```
$
```

- b. Enter the following commands and compare the behavior:

```
$ TYPEWRITER
```

```
$ typewriter
```

```
_File:
```

```
Interrupt
```

```
$
```

Because there is no space between TYPE and WRITER, DCL assumes that TYPEWRITER is a command verb. DCL checks only the first four characters of the verb for uniqueness and runs TYPE, prompting for a file to type.

```
$ TYPE WRITER
```

```
$ type writer
```

```
Paperback?
```

```
$
```

The TYPE command assumes a default file type of .LIS and displays WRITER.LIS.

```
$ TYPE WRITER.LIS
```

```
$ type writer.lis
```

```
Paperback?
```

```
$
```

TYPE displays WRITER.LIS.

- c. Create the file NAMES.DAT and enter the command `$ TYPE NAMES`. What happens?

```
$ create names.dat
```

```
Bruce Ellis
```

```
Albert Einstein
```

```
Ken Olsen
```

```
Dennis O'Brien
```

```
Keith McLaughlin
```

```
Glenn Oehms
```

```
Laurel Zolfonoon
```

```
Susan Ellis
```

```
Exit
```

```
$
```

```
$ type names
```

```
%TYPE-W-SEARCHFAIL, error searching for SYS$SYSDEVICE:[STUDENT7]NAMES.LIS;
```

```
-RMS-E-FNF, file not found
```

```
$
```

DCL assumes a file type of .LIS and does not display NAMES.DAT without entering both the file name and type.

11. Enable insert mode for command line editing.

```
$ set terminal/insert
$
```

12. Use the arrow keys to recall the command that you used to create the file called NAMES.DAT. Change the command so that it creates a file called BLAME.DAT.

Use the up arrow to get the command back and the right and left arrow keys to move forward and back. Use the backspace key to delete the previous character.

```
$ create blame.dat
Big Tobacco
The President
The Liberals
The Conservatives
Anybody but me
Exit
$
```

13. When entering a command, what do the keys Ctrl/A, Ctrl/H, and Ctrl/E do?

Ctrl/A toggles between insert and overstrike mode.

Ctrl/H moves the cursor to the beginning (head) of the line.

Ctrl/E moves the cursor to the end of the line.

14. Enter the command `$ MONITOR PROCESS`

- a. Enter Ctrl/Y. What happens?
The image (program) is interrupted.
- b. Enter the command `$ CONTINUE`
The program resumes.
- c. Enter Ctrl/W. What happens?
The screen is refreshed.

15. Use the `$ SET TERMINAL` command to change your terminal width to 132 columns. Change it back to 80 columns.

```
$ set term/wid=132
```

```
$ set term/wid=80
```

16. Enter the command `$ SET TERMINAL/ECHO`

- a. Enter the command `$ SET TERMINAL/NOECHO`

```
$ set terminal/echo
```

```
$ set terminal/noecho
```

```
$
```

```
Directory SYS$SYSDEVICE:[STUDENT7]
```

```
BLAME.DAT;1
```

```
NAMES.DAT;1
```

```
WRITER.LIS;1
```

```
Total of 3 files.
```

```
$
```

```
OpenVMS User Processes at 10-APR-1999 23:38:03.74
```

```
Total number of users = 2, number of processes = 2
```

Username	Interactive	Subprocess	Batch
BAE	1		
STUDENT7	1		

```
$
```

- b. What happens when you enter commands?

You cannot see them.

- c. Use the arrow keys to bring back the `$SET TERMINAL/ECHO` command.

17. Use the \$ SET TERMINAL/INQUIRE command to set your terminal attributes, based on your terminal type.

```
$ set term/inquire
$
```

18. Use the \$ SHOW SYSTEM/PAGE=SAVE command to view the processes on the system one screen at a time. Back up to a previous screen.

```
$ show system/page=save
```

```

  Pid   Process Name   State Pri    I/O      CPU      Page flts  Pages
OpenVMS V7.2  on node ELLIS 10-APR-1999 23:40:48.56 Uptime  0 14:00:22

```

Pid	Process Name	State	Pri	I/O	CPU	Page flts	Pages
00000101	SWAPPER	HIB	16	0	0 00:00:00.16	0	0
00000106	LANACP	HIB	13	68	0 00:00:00.24	135	111
00000108	FASTPATH_SERVER	HIB	10	9	0 00:00:00.13	69	82
00000109	IPCACP	HIB	10	10	0 00:00:00.06	31	43
0000010A	ERRFMT	HIB	8	1611	0 00:00:05.10	87	101
0000010C	OPCOM	HIB	7	531	0 00:00:03.34	88	46
0000010D	AUDIT_SERVER	HIB	10	81	0 00:00:00.37	150	163
0000010E	JOB_CONTROL	HIB	10	68	0 00:00:00.18	55	74
0000010F	QUEUE_MANAGER	HIB	10	273	0 00:00:00.83	135	170
00000110	SECURITY_SERVER	HIB	10	38	0 00:00:02.50	322	326
00000111	DNS\$ADVER	LEF	4	3376	0 00:00:08.61	348	243
00000112	LESS\$ACP_V30	HIB	9	127	0 00:00:00.46	78	90
00000113	NET\$ACP	HIB	6	45	0 00:00:00.32	141	131
00000114	REMACP	HIB	8	10	0 00:00:00.05	29	16
00000115	NET\$EVD	HIB	6	20	0 00:00:00.86	244	241
00000116	DTSS\$CLERK	LEF	12	666	0 00:00:02.69	142	171
00000117	TP_SERVER	HIB	10	3376	0 00:00:31.09	76	94
00000118	LATACP	HIB	14	8	0 00:00:00.84	73	58
00000119	TCPIP\$INET_ACP	HIB	8	123	0 00:00:00.44	266	137
0000011A	TCPIP\$PWIP_ACP	HIB	10	53	0 00:00:00.26	132	145

RETURN/SPACE=More, PREV/NEXT=Scroll, INS/REM=Pan, SELECT=80/132, CTRL/Z=Quit

Pid	Process Name	State	Pri	I/O	CPU	Page flts	Pages
0000010C	OPCOM	HIB	7	531	0 00:00:03.34	88	46
0000010D	AUDIT_SERVER	HIB	10	81	0 00:00:00.37	150	163
0000010E	JOB_CONTROL	HIB	10	68	0 00:00:00.18	55	74

000010F	QUEUE_MANAGER	HIB	10	273	0 00:00:00.83	135	170
00000110	SECURITY_SERVER	HIB	10	38	0 00:00:02.50	322	326
00000111	DNS\$ADVER	LEF	4	3376	0 00:00:08.61	348	243
00000112	LESS\$ACP_V30	HIB	9	127	0 00:00:00.46	78	90
00000113	NET\$ACP	HIB	6	45	0 00:00:00.32	141	131
00000114	REMACP	HIB	8	10	0 00:00:00.05	29	16
00000115	NET\$EVD	HIB	6	20	0 00:00:00.86	244	241
00000116	DTSS\$CLERK	LEF	12	666	0 00:00:02.69	142	171
00000117	TP_SERVER	HIB	10	3376	0 00:00:31.09	76	94
00000118	LATACP	HIB	14	8	0 00:00:00.84	73	58
00000119	TCPIP\$INET_ACP	HIB	8	123	0 00:00:00.44	266	137
0000011A	TCPIP\$PWIP_ACP	HIB	10	53	0 00:00:00.26	132	145
0000011B	TCPIP\$NTP	LEF	9	50335	0 00:00:49.82	338	192 N
0000011D	VPM_SERVER	HIB	15	14	0 00:00:01.65	217	186
0000011F	DECW\$SERVER_0	HIB	8	798	0 00:00:05.20	666	872
00000120	DTLOGIN	LEF	5	143	0 00:00:01.02	358	244
00000121	DTGREET	LEF	4	464	0 00:00:08.45	805	365 S
00000126	Bruce Ellis	CUR 0	9	1008	0 00:00:05.15	2612	165
00000128	BAE	LEF	4	131	0 00:00:00.62	365	64

*** End *** | PREV/NEXT=Scroll, INS/REM=Pan, SELECT=80/132, CTRL/Z=Quit

\$

19. Enter the command \$ TYPE/PAGE NL:. What happens?

The screen is cleared.

20. Display one of your files using the \$ TYPE command.

a. Enter the command \$ SHOW SYMBOL \$SEVERITY

b. What value do you see?

```
$ type blame.dat
```

```
Big Tobacco
```

```
The President
```

```
The Liberals
```

```
The Conservatives
```

```
Anybody but me
```

```
$ show symbol $severity
```

```
$SEVERITY == "1"
```

```
$
```

1 indicates success.

c. Enter the command:

```
$ copy sys$system:sysuaf.dat []
```

d. Enter the command \$SHOW SYMBOL \$SEVERITY

e. What value do you see?

```
$ copy sys$system:sysuaf.dat []
```

```
%COPY-E-OPENIN, error opening SYSS$COMMON:[SYSEXE]SYSUAF.DAT;1 as input
```

```
-RMS-E-PRV, insufficient privilege or file protection violation
```

```
$ show symbol $severity
```

```
$SEVERITY == "2"
```

```
$
```

2 indicates an error occurred.

f. Use \$ HELP/MESSAGE to interpret the error message.

```
$ help/message
```

```
OPENIN, error opening 'input-file-spec' as input
```

```
Facility:      Shared by several facilities
```

```
Explanation:  The specified input file cannot be opened.
```

User Action: Make sure the file exists and is accessible, or take action based on an accompanying message.

\$

\$ help/message/fac=rms prv

PRV, insufficient privilege or file protection violation

Facility: RMS, OpenVMS Record Management Services

Explanation: One of the following occurred:

- o Your process does not have one or more of the privileges necessary to perform the requested file access.
- o Your process is denied access to the file because of the protection applied to the file or to the directory in which the file is listed.

User Action: Verify the protection applied to the file (using the DIRECTORY/FULL command) and change the protection if necessary (using the SET PROTECTION command). If you cannot change the protection, ask the system manager or the owner of the file to change it.

If the file's current protection should permit the desired access, verify the protection applied to the directory in

If these actions do not apply or fail to solve the problem, ask the system manager for the necessary privileges.

\$

21. Get comfortable with using the EVE editor by creating a file with the names of five friends.

```
$ edit friends.dat
```

```
Den
```

```
Susan
```

```
Max
```

```
Larry
```

```
Kim
```

```
[End of file]
```

```
Buffer: FRIENDS.DAT
```

```
| Write | Insert | Forward
```

```
5 lines written to file SYS$SYSDEVICE:[STUDENT7]FRIENDS.DAT;1
```

```
$
```

22. Enable the EDT keypad in EVE. Press the PF2 key (/ on a PC keypad) to get help. Determine how to select text, cut it, and paste it.
 - a. Does the EDT keypad allow you to copy selected text?

No.
 - b. Define <GOLD> C to perform a copy command.

```
Command: define key copy
```

```
Press the key you want to define: (( type the gold key then the letter c ))
```

```
Key defined.
```

- c. Define <GOLD> M to center a line.

```
Command: define key center
```

- d. Save your key definitions in an EVE section file (Hint: SAVE EXTENDED EVE *file*).

```
Command: save extended eve my_sec.tpu$section
```

- e. Exit the editor and use your EVE section file to recover your key definitions.

```
$ edit/section=sys$login:my_sec friends.dat
```

- f. Use the command \$ DEFINE EVE\$KEYPAD EDT to automatically enable the EDT keypad when you activate the editor.

```
$ define eve$keypad edt
```

23. Use the mail utility to send a message to your instructor.

```
$ mail
```

```
MAIL> send
```

```
To:    bae
```

```
Subj:  Volume
```

```
Enter your message below. Press CTRL/Z when complete, or CTRL/C to quit:
```

```
Could you please speak a little louder.
```

```
Exit
```

```
MAIL> exit
```

```
$
```

24. Send a mail message to another user.

```
MAIL> send
```

```
To:    max
```

```
Subj:  Que Pasa
```

```
Enter your message below. Press CTRL/Z when complete, or CTRL/C to quit:
```

```
What's Up?
```

```
Exit
```

```
MAIL>
```

25. Create a distribution list and use it to send messages to multiple users.

```
MAIL> spawn
```

```
$ create who.dis
```

```
bae
```

```
max
```

```
Exit
```

```
$ lo
```

```
Process STUDENT7_1 logged out at 11-APR-1999 00:03:02.01
```

```
You have 0 new messages.
```

```
MAIL> send
```

```
To: @who.dis
```

```
Subj: In the Pink
```

```
Enter your message below. Press CTRL/Z when complete, or CTRL/C to quit:
```

```
Is anybody out there?
```

```
Exit
```

```
MAIL>
```

26. Create a mail subdirectory.

```
MAIL> set mail_directory [.mail_bag]
```

```
%MAIL-I-CREATED, SYS$SYSDEVICE:[STUDENT7.MAIL_BAG] created
```

```
MAIL>
```

27. Use the PHONE utility to talk to another student.

```
$ phone bae
```

28. Find all files in your directory that have names starting with the letter N or W.

```
$ dir n*,w*
```

```
Directory SYS$SYSDEVICE:[STUDENT7]
```

```
NAMES.DAT;1
```

```
WHO.DIS;1
```

```
WRITER.LIS;1
```

```
Total of 3 files.
```

```
$
```

29. Find all files in your directory that have file types of .DAT.

```
$ dir *.dat
```

```
Directory SYS$SYSDEVICE:[STUDENT7]
```

```
BLAME.DAT;1      FRIENDS.DAT;3      FRIENDS.DAT;2      FRIENDS.DAT;1
NAMES.DAT;1
```

Total of 5 files.

```
$
```

30. Find all files in your directory that have exactly five character file names with any file type.

```
$ dir %%%%.*
```

```
Directory SYS$SYSDEVICE:[STUDENT7]
```

```
BLAME.DAT;1      NAMES.DAT;1
```

Total of 2 files.

```
$
```

31. Create a subdirectory.

```
$ creat/dir [.sub_dude]
```

a. Copy some files into the subdirectory.

```
$ copy *.* [.sub_dude]
```

```
%COPY-W-ACCONFLICT, file access conflict
```

```
%COPY-W-NOTCOPIED, SYS$SYSDEVICE:[STUDENT7]SUB_DUDE.DIR;1 not copied
```

```
$
```

b. Set your default directory to your subdirectory.

```
$ set default [.sub_dude]
```

c. Create a file in the subdirectory.

```
$ create x.files
```

```
Mulder
```

```
Sculley
```

```
Exit
```

```
$
```

d. Verify that the new file is in the subdirectory.

```
$ dir x
```

```
Directory SYS$SYSDEVICE:[STUDENT7.SUB_DUDE]
```

```
X.FILES;1
```

```
Total of 1 file.
```

```
$
```

e. Set your default to your top-level directory (UFD).

```
$ set def [-]
```

f. Display the file in the subdirectory, without setting your default directory to the subdirectory.

```
$ type [.sub_dude]x.files
```

```
Mulder
```

```
Sculley
```

```
$
```

- g. Delete the subdirectory.

```
$ delete [.sub_dude]*.*;*
%DELETE-W-FILNOTDEL, error deleting SYS$SYSDEVICE:[STUDENT7.SUB_DUDE]MAIL_BAG.DIR;1
-RMS-E-PRV, insufficient privilege or file protection violation
$ set file/prot=(o:rwed) [.sub_dude]mail_bag.dir;
$ delete [.sub_dude]*.*;*
$ set file/prot=(o:rwed) sub_dude.dir
$ delete sub_dude.dir;
$
```

32. Create the following directories under your UFD:

```
[.SUB1]
[.SUB1.SUBSUBA]
[.SUB1.SUBSUBB]
$ create/dir [.sub1.subsuba]
$ create/dir [.sub1.subsubb]
$
a. Set your default directory to [.sub1.subsuba].
$ set def [.sub1.subsuba]
b. From this directory list the directory [.sub1.subsubb], using wildcarding.
$ dir [-.subsubb]
%DIRECT-W-NOFILES, no files found
$
c. Using wildcard characters, set your default directory back to your UFD.
$ set def [--]
$ sh def
SYS$SYSDEVICE:[STUDENT7]
$
```

33. Determine whether the student sitting next to you is in the same UIC group as you.

```
$ write sys$output f$fa0("!%U",F$ident("student7","name_to_number"))
[42,7]
$ write sys$output f$fa0("!%U",F$ident("bae","name_to_number"))
[42,1]
$
```

Yes we are both in group 42.

34. Set up your directory so users in your group can only list your directory if they know specific files in your directory by name.

```
$ set file/prot=(g:e) [-]student7.dir
$
```

From other students account:

```
$ dir sys$sysdevice:[student7]
%DIRECT-E-OPENIN, error opening SYS$SYSDEVICE:[STUDENT7]*.*;* as input
-RMS-E-PRV, insufficient privilege or file protection violation
$ dir sys$sysdevice:[student7]names.dat
```

Directory SYS\$SYSDEVICE:[STUDENT7]

NAMES.DAT;1

Total of 1 file.

\$

35. Set up your directory so that the student sitting next to you can wildcard list your directory, but no one else in your group can.

```
$ set file/acl=(ident=bae,access=read+execute) [-]student7.dir
```

From other students account:

```
$ dir sys$sysdevice:[student7]
```

Directory SYS\$SYSDEVICE:[STUDENT7]

BLAME.DAT;1	FRIENDS.DAT;3	FRIENDS.DAT;2	FRIENDS.DAT;1
MAIL_BAG.DIR;1	MY_SEC.TPU\$SECTION;1		NAMES.DAT;1
SUB1.DIR;1	WHO.DIS;1	WRITER.LIS;1	

Total of 10 files.

\$

36. Set your default protection so that new files are created with no access for group. Verify the change.

```
$ set prot=(g)/default
```

```
$ show prot
```

```
SYSTEM=RWED, OWNER=RWED, GROUP=NO ACCESS, WORLD=NO ACCESS
```

```
$ creat new.file
```

```
Top secret
```

```
Exit
```

```
$ dir/prot new.file
```

```
Directory SYS$SYSDEVICE:[STUDENT7]
```

```
NEW.FILE;1          (RWED,RWED,,)
```

```
Total of 1 file.
```

```
$
```

37. Use \$ HELP CREATE/DIRECTORY to determine how to set a version limit on a directory.

- a. Create a subdirectory with a version limit of 3.

```
$ create/dir/vers=3 [.only3]
```

- b. Create a file in the subdirectory.

```
$ creat x.dat
```

```
This usually marks  
the spot!
```

```
Exit
```

```
$
```

- c. Verify that the new file has a version limit of 3.

```
$ dir/full x.dat
```

```
Directory SYS$SYSDEVICE:[STUDENT7.ONLY3]
```

```
X.DAT;1                      File ID: (6947,5,0)
Size:          1/3           Owner:   [BRUDEN,STUDENT7]
Created:       11-APR-1999 00:44:05.31
Revised:       11-APR-1999 00:44:16.17 (1)
Expires:       <None specified>
Backup:        <No backup recorded>
Effective:     <None specified>
Recording:     <None specified>
File organization: Sequential
Shelved state:  Online
Caching attribute: Writethrough
File attributes: Allocation: 3, Extend: 0, Global buffer count: 0
                  Version limit: 3
Record format:  Variable length, maximum 0 bytes, longest 18 bytes
Record attributes: Carriage return carriage control
RMS attributes:  None
Journaling enabled: None
File protection: System:RWED, Owner:RWED, Group:, World:
Access Cntrl List: None
Client attributes: None
```

```
Total of 1 file, 1/3 blocks.
```

```
$
```

- d. Make four copies of the file.

```
$ copy x.dat x.dat
$ copy x.dat x.dat
$ copy x.dat x.dat
$ copy x.dat x.dat
```

e. What happens?

```
$ dir
```

```
Directory SYS$SYSDEVICE:[STUDENT7.ONLY3]
```

```
X.DAT;5
```

```
X.DAT;4
```

```
X.DAT;3
```

```
Total of 3 files.
```

```
$
```

The earliest versions are purged.

38. Use `$ HELP SET FILE/VERSION_LIMIT` and `$ HELP SET DIRECTORY/VERSION_LIMIT` to determine how to set version limits on files in your UFD. Set a version limit of 5 for all existing and new files in your UFD. Clean up any files with more than five versions.

```
$ set dir/vers=5 [student7]
```

```
$ set file/version=5 *.*
```

```
$ purge/keep=5 *.*
```

```
$
```

39. Find all occurrences of the string PCBDEF in the file `SYS$LIBRARY:LIB.REQ`.

```
$ search sys$library:lib.req pcbdef
```

```
literal HWPB$S_HWPBDEF = 128;          ! Old size name - synonym
```

```
!*** MODULE $PCBDEF ***
```

```
literal PCB$S_PCBDEF = 928;            ! Old PCB size for compatibility
```

```
$
```

40. Create a logical name identifying one of your subdirectories.

- a. Use the logical name to list the subdirectory.

```
$ dir [.only3]
```

```
Directory SYS$SYSDEVICE:[STUDENT7.ONLY3]
```

```
X.DAT;5          X.DAT;4          X.DAT;3
```

```
Total of 3 files.
```

```
$ def only3 SYS$SYSDEVICE:[STUDENT7.ONLY3]
```

```
$ dir only3
```

```
Directory SYS$SYSDEVICE:[STUDENT7.ONLY3]
```

```
X.DAT;5          X.DAT;4          X.DAT;3
```

```
Total of 3 files.
```

```
$ type only3:x.dat
```

```
This usually marks  
the spot!
```

```
$
```

41. Create a logical name identifying one of your files.

- a. Use the logical name to display the file.

```
$ def blame sys$login:blame.dat
```

```
$
```

```
$ type blame
```

```
Big Tobacco
```

```
The President
```

```
The Liberals
```

```
The Conservatives
```

```
Anybody but me
```

```
$
```

42. Define the logical name EVE\$KEYPAD as EDT in your job logical name table.

- a. Use the \$ SPAWN command to verify that the logical name is visible to all processes in your job.

```
$ define/job eve$keypad edt
$ spawn
%DCL-S-SPAWNED, process STUDENT7_1 spawned
%DCL-S-ATTACHED, terminal now attached to process STUDENT7_1
$ sh log eve$keypad
  "EVE$KEYPAD" = "EDT" (LNM$JOB_80D84F00)
$
```

43. Set your default directory to another location. Use the logical name SYS\$LOGIN to get back to your login directory.

```
$ set def [.sub1.subsuba]
$ sh def
  SYS$SYSDEVICE:[STUDENT7.SUB1.SUBSUBA]
$ set def sys$login
$ sh def
  SYS$SYSDEVICE:[STUDENT7]
$
```

44. Create a file called LAST.DAT containing your last name. Create a file called FIRST.DAT containing your first name. Create a logical name called "NAME" defined as LAST.DAT in the process logical name table. Create a logical name called "NAME" defined as FIRST.DAT in the Job logical name table.

```
$ create last.dat
Ellis
Exit
$ create first.dat
Bruce
Exit
$ define name last.dat
$ define/job name first.dat
$
```

- a. Enter the command: `$ TYPE NAME`
- b. Which file is displayed?

```
$ type name
```

```
Ellis
```

```
$
```

Why?

Because the logical `LN$FILE_DEV` defines that logical names will be translated by looking in the process logical name table (`LN$PROCESS`) first, followed by the job logical name table (`LN$JOB`).

```
$ show logical/table=ln$system_directory ln$file_dev
```

```
"LN$FILE_DEV" = "LN$PROCESS" (LN$SYSTEM_DIRECTORY)
```

```
    = "LN$JOB"
```

```
    = "LN$GROUP"
```

```
    = "LN$SYSTEM"
```

```
    = "DEC$LOGICAL_NAMES"
```

```
1 "LN$SYSTEM" = "LN$SYSTEM_TABLE" (LN$SYSTEM_DIRECTORY)
```

```
    = "LN$SYSCUSTER"
```

```
2 "LN$SYSCUSTER" = "LN$SYSCUSTER_TABLE" (LN$SYSTEM_DIRECTORY)
```

```
1 "DEC$LOGICAL_NAMES" [table] = "" (LN$SYSTEM_DIRECTORY)
```

```
$
```

45. Create a symbol called `EDIT` that causes the section file created in lab 23 to be used automatically.

```
$ ed*it=="edit/section=sys$login:my_sec"
```

```
$
```

46. Create a symbol called `CLEAR` that clears your screen.

```
$ cl*ear="type/page nl:"
```

```
$
```

47. Create a symbol called TYPE that uses the qualifier /PAGE=SAVE to display a file one screen at a time.

```
$ type="type/page=save"
```

```
$
```

```
$ type friends.dat
```

```
Den
```

```
Susan
```

```
Max
```

```
Larry
```

```
Kim
```

```
...
```

```
**End-of-File** | PREV/NEXT=Scroll, INS/REM=Pan, SELECT=80/132, Q=Quit
```

- a. Can you display a file without the /PAGE=SAVE qualifier and without deleting the TYPE symbol?

Yes.

```
$ typeextra friends.dat
```

```
Den
```

```
Susan
```

```
Max
```

```
Larry
```

```
Kim
```

```
$
```

48. Create a symbol called X with the numeric value 32 assigned to it.

```
$ x=32
```

```
$
```

- a. Use the \$SHOW SYMBOL command to determine the hexadecimal equivalent value for 32.

```
$ sh sym x
```

```
X = 32    Hex = 00000020    Octal = 0000000040
```

```
$
```

- b. Delete the symbol.

```
$ delete/symbol x
```

```
$
```

49. View all global symbols.

```
$ show symbol/global/all
$RESTART == "FALSE"
$SEVERITY == "1"
$STATUS == "%X00030001"
ED*IT == "edit/section=sys$login:my_sec"
$
```

50. Create a symbol called TAIL that performs a \$TYPE/TAILO=10 command. The symbol should work if TA, TAI, or TAIL is entered.

```
$ ta*il="type/tail=10"
$
$ typex twelve.lines
line 1
line 2
line 3
line 4
line 5
line 6
line 7
line 8
line 9
line 10
line 11
line 12
$ tail twelve.lines
line 3
line 4
line 5
line 6
line 7
line 8
line 9
line 10
line 11
line 12
$
```

- a. Delete the symbol.

```
$ delete/symbol tail
$
```

51. Create a symbol. Spawn a subprocess.

- a. Can you see the symbol?

```
$ name=="Bruce"
$ spawn
%DCL-S-SPAWNED, process STUDENT7_2 spawned
%DCL-S-ATTACHED, terminal now attached to process STUDENT7_2
$ show symbol name
NAME == "Bruce"
$
```

Yes, symbols are, by default, copied to subprocesses.

- b. Can you create a symbol in the subprocess that is visible to the parent?

No.

```
$ name=="Fred"
$ lo
Process STUDENT7_2 logged out at 11-APR-1999 11:18:55.33
%DCL-S-RETURNED, control returned to process STUDENT7_1
$ show sym name
NAME == "Bruce"
$
```

52. Define the key KP1 (keypad number 1) to enter a \$SHOW SYSTEM/PAGE=SAVE command.

```
$ define/key kp1 "show system/page=save"/terminate/erase
%DCL-I-DEFKEY, DEFAULT key KP1 has been defined
$ set term/application
```

53. Define the key KP2 to enter a \$DIRECTORY/DATE/PAGE=SAVE command.

```
$ define/key kp2 "$DIRECTORY/DATE/PAGE=SAVE" /terminate/erase/noecho
%DCL-I-DEFKEY, DEFAULT key KP2 has been defined
$
```

a. Delete the key definition.

```
$ delete/key kp2
%DCL-I-DELKEY, DEFAULT key KP2 has been deleted
$
```

54. Create a LOGIN.COM command procedure that:

- a. Creates a symbol to activate the editor with a TPU section file.
- b. Creates symbols for:
 - 1) A MORE command that displays SYS\$INPUT one screen at a time.
 - 2) A CLEAR command that clears the screen.
- c. Creates key definitions of your choice.
- d. Creates a logical name pointing to a directory containing command procedures.
- e. Enters a \$ SET TERMINAL/INQUIRE command, only if you are logging interactively.

```
$ typex login.com
```

```
$!a. Creates a symbol to activate the editor with a TPU section file.
```

```
$ ED*IT="EDIT/section=sys$login:my_sec"
```

```
$
```

```
$!b. Creates symbols for:
```

```
$!1) A MORE command that will display SYS$INPUT one screen at a time.
```

```
$!2) A CLEAR command that will clear the screen.
```

```
$ MORE=="TYPE/PAGE=SAVE SYS$INPUT"
```

```
$ CL*EAR="TYPE/PAGE NL:"
```

```
$
```

```
$!c. Creates key definitions of your choice.
```

```
$ define/key kp1 "show system/page=save"/terminate/erase
```

```
$ define/key kp2 "$DIRECTORY/DATE/PAGE=SAVE" /terminate/erase/noecho
```

```
$
```

```
$!d. Creates a logical name pointing to a directory containing command
```

```
$! procedures.
```

```
$ DEFINE/JOB MY_COMS SYS$SYSDEVICE:[STUDENT7.MY_COMS]
```

```
$
```

\$!e. Issues a \$ SET TERMINAL/INQUIRE command, only if you are logging
\$! interactively.

```
$ if f$mode().EQS."INTERACTIVE"
$ THEN
$   SET TERMINAL/INQUIRE/APPLICATION
$   write sys$output "Terminal keypad set to application"
$ ENDIF
$
$
```

f. Test your LOGIN.COM.

```
$ @login
%DCL-I-DEFKEY, DEFAULT key KP1 has been defined
%DCL-I-DEFKEY, DEFAULT key KP2 has been defined
%DCL-I-SUPERSEDE, previous value of MY_COMS has been superseded
Terminal keypad set to application
$ create/dir [.my_coms]
$
```

55. The C compiler keeps special files called *headers* in a text library. The following command extracts a header called `ssdef.h` and displays it on your screen:

```
$ libr/extract=ssdef/out=sys$output/text sys$library:sys$starlet_c
```

- a. Try the command.
- b. The output spills across the screen very quickly. Use the PIPE command and your MORE symbol to display the output one screen at a time.

```
$ pipe libr/extract=ssdef/out=sys$output/text sys$library:sys$starlet_c | more
```

```

/*****
/* Created: 20-NOV-1998 19:18:21 by OpenVMS SDL EV1-60      */
/* Source:  20-NOV-1998 19:17:34 $64$DUA564:[STARLET_H.SRC]SSMSG.SDI;1 */
/*****
/*** MODULE $SSDEF ***/
#ifdef __SSDEF_LOADED
#define __SSDEF_LOADED 1

#pragma __nostandard                /* This file uses non-ANSI-Standard fea
#pragma __member_alignment __save
Rev. 0.30
```

```
#pragma __nomember_alignment
#ifdef __INITIAL_POINTER_SIZE          /* Defined whenever ptr size pr
#pragma __required_pointer_size __save /* Save the previously-defined
#pragma __required_pointer_size __short /* And set ptr size default to
#endif
```

```
#ifdef __cplusplus
    extern "C" {
#define __unknown_params ...
#define __optional_params ...
#else
#define __unknown_params
#define __optional_params ...
```

```
    RETURN/SPACE=More, PREV/NEXT=Scroll, INS/REM=Pan, SELECT=80/132, Q=Quit
```

- c. Write a command procedure that accepts a header file name through the P1 parameter and performs the function described in step b for any header. Try it with `iodef` and `jpidef`. (For a list of headers use: `$LIBRARY/LIST/TEXT SYS$LIBRARY:SYS$STARLET_C`).

```
$ typex show_vms_header.com
$if P1.EQS.""
$ then
$     write sys$error "Need header name (No ""h"")"
$     exit
$ endif$ endif
$ pipe lib/extract='P1'/out=sys$output/text sys$library:sys$starlet_c | more
$
$ @show_vms_header
Need header name (No ".h")
$
$ @show_vms_header iodef
```

```
/* *****
/* Created: 20-NOV-1998 19:18:33 by OpenVMS SDL EV1-60 */
/* Source: 20-NOV-1998 19:17:35 $64$DUA564:[STARLET_H.SRC]STARDEFFL.SDI;1 */
/* *****
/*** MODULE $IODEF ***
#endif __IODEF_LOADED
```

```

#define __IODEF_LOADED 1

#pragma __nostandard                /* This file uses non-ANSI-Standard fea
#pragma __member_alignment __save
#pragma __nomember_alignment
#ifdef __INITIAL_POINTER_SIZE        /* Defined whenever ptr size pr
#pragma __required_pointer_size __save /* Save the previously-defined
#pragma __required_pointer_size __short /* And set ptr size default to
#endif

#ifdef __cplusplus
    extern "C" {
#define __unknown_params ...
#define __optional_params ...
#else
#define __unknown_params
#define __optional_params ...
    RETURN/SPACE=More, PREV/NEXT=Scroll, INS/REM=Pan, SELECT=80/132, Q=Quit

```

56. Create a directory called [.backups].

```

$ create/dir [.my_backups]
$

```

- a. Back up all files modified since midnight today to a save set in the directory. Note: exclude all files in the [.backups] directory from your backup.

```

$ backup/since=today/mod/exclude=[.my_backups...]*.* [...]*. * -
_$ [.my_backups]thurs.bck/save
$

```

b. List your save set.

```
$ backup/list [.my_backups]thurs.bck/save
```

Listing of save set(s)

```
Save set:          THURS.BCK
Written by:        STUDENT7
UIC:              [000042,000007]
Date:             11-APR-1999 11:52:26.62
Command:          BACKUP/SINCE=TODAY/MOD/EXCLUDE=[.MY_BACKUPS...]*.* [...]*. * [
                  .MY_BACKUPS]THURS.BCK/SAVE
Operating system:  OpenVMS Alpha version V7.2
BACKUP version:    V7.2
CPU ID register:   80000000
Node name:         _ELLIS::
Written on:        _ELLIS$DKA200:
Block size:        32256
Group size:        10
Buffer count:      56
```

```
[STUDENT7]BLAME.DAT;1          1 10-APR-1999 23:31
[STUDENT7]FIRST.DAT;1         1 11-APR-1999 11:02
[STUDENT7]FRIENDS.;1         1 11-APR-1999 11:08
[STUDENT7]FRIENDS.DAT;3       1 10-APR-1999 23:58
[STUDENT7]FRIENDS.DAT;2       1 10-APR-1999 23:57
[STUDENT7]FRIENDS.DAT;1       1 10-APR-1999 23:50
[STUDENT7]LAST.DAT;1          1 11-APR-1999 11:02
[STUDENT7]LOGIN.COM;3         2 11-APR-1999 11:32
[STUDENT7]LOGIN.COM;2         2 11-APR-1999 11:31
[STUDENT7]LOGIN.COM;1         2 11-APR-1999 11:29
[STUDENT7]MAIL_BAG.DIR;1      1 11-APR-1999 00:04
[STUDENT7]MY_BACKUPS.DIR;1     1 11-APR-1999 11:48
[STUDENT7]MY_COMS.DIR;1       1 11-APR-1999 11:32
[STUDENT7]MY_SEC.TPU$SECTION;1 1208 10-APR-1999 23:56
[STUDENT7]NAMES.DAT;1         1 10-APR-1999 23:25
[STUDENT7]NEW.FILE;1          1 11-APR-1999 00:37
[STUDENT7]ONLY3.DIR;1         1 11-APR-1999 00:43
```

```
[STUDENT7.ONLY3]X.DAT;5      1  11-APR-1999  00:45
[STUDENT7.ONLY3]X.DAT;4      1  11-APR-1999  00:45
[STUDENT7.ONLY3]X.DAT;3      1  11-APR-1999  00:45
[STUDENT7]SHOW_VMS_HEADER.COM;1  1  11-APR-1999  11:36
[STUDENT7]SUB1.DIR;1          1  11-APR-1999  00:27
[STUDENT7.SUB1]SUBSUBA.DIR;1   1  11-APR-1999  00:27
[STUDENT7.SUB1]SUBSUBB.DIR;1   1  11-APR-1999  00:27
[STUDENT7]TWELVE.LINES;1      1  11-APR-1999  11:14
[STUDENT7]WHO.DIS;1           1  11-APR-1999  00:02
[STUDENT7]WRITER.LIS;1        1  10-APR-1999  23:20
```

Total of 27 files, 1237 blocks

End of save set

\$

- c. Create a command procedure that backs up all files modified since midnight today in a file called [.backups]*dow*.BCK, where *dow* is the current day of week. The following lexical function returns the current day of week: f\$cvtime("TODAY", , "WEEKDAY").

```
$ typex my_backup.com
```

```
$backup_file=f$cvtime("TODAY", , "WEEKDAY")+ ".bck"
```

```
$ backup/since=today/mod/exclude=[.my_backups...]*.* [...]*. * -
```

```
  [.my_backups]'backup_file'/save
```

\$

```
$ @my_backup
```

```
$ dir [.my_backups]
```

Directory SYS\$SYSDEVICE:[STUDENT7.MY_BACKUPS]

```
SUNDAY.BCK;1
```

```
THURS.BCK;1
```

Total of 2 files.

\$

57. Create the directories [.a] and [.a.b]. Put a couple of files in each directory. Use BACKUP to replicate the files in [.a] in a directory called [.x].

```
$ create/dir [.a.b]
$ copy name*. * [.a]
$ copy blame*. * [.a.b]
$
$ backup [.a...] *.* [.x...]
$ dir [.a...]
```

Directory SYS\$SYSDEVICE:[STUDENT7.A]

```
B.DIR;1          NAMES.DAT;1
```

Total of 2 files.

Directory SYS\$SYSDEVICE:[STUDENT7.A.B]

```
BLAME.DAT;1
```

Total of 1 file.

Grand total of 2 directories, 3 files.

```
$ dir [.x...]
```

Directory SYS\$SYSDEVICE:[STUDENT7.X]

```
B.DIR;1          NAMES.DAT;1
```

Total of 2 files.

Directory SYS\$SYSDEVICE:[STUDENT7.X.B]

BLAME.DAT;1

Total of 1 file.

Grand total of 2 directories, 3 files.

\$

58. Create a command procedure that displays your name. Submit it to run in batch. Examine the log file.

\$ create me.com

\$write sys\$output "Bruce Ellis"

Exit

\$ submit/noprint me

Job ME (queue LOW_RIDER, entry 22) started on LOW_RIDER

\$ type me.log

%DCL-I-DEFKEY, DEFAULT key KP1 has been defined

%DCL-I-DEFKEY, DEFAULT key KP2 has been defined

Bruce Ellis

STUDENT7 job terminated at 11-APR-1999 12:02:26.94

Accounting information:

Buffered I/O count:	35	Peak working set size:	1616
---------------------	----	------------------------	------

Direct I/O count:	25	Peak virtual size:	167888
-------------------	----	--------------------	--------

Page faults:	164	Mounted volumes:	0
--------------	-----	------------------	---

Charged CPU time:	0 00:00:00.32	Elapsed time:	0 00:00:00.82
-------------------	---------------	---------------	---------------

\$

59. Submit the command procedure created in problem 56, part c to run in batch.

```
$ typex my_backup.log
%DCL-I-DEFKEY, DEFAULT key KP1 has been defined
%DCL-I-DEFKEY, DEFAULT key KP2 has been defined
%BACKUP-E-OPENIN, error opening SYS$SYSDEVICE:[STUDENT7]MY_BACKUP.LOG;1 as input
-SYSTEM-W-ACCONFLICT, file access conflict
STUDENT7      job terminated at 11-APR-1999 12:03:23.24
```

Accounting information:

Buffered I/O count:	221	Peak working set size:	4672
Direct I/O count:	286	Peak virtual size:	173472
Page faults:	385	Mounted volumes:	0
Charged CPU time:	0 00:00:01.60	Elapsed time:	0 00:00:06.11

\$

```
$backup_file=f$cvtime("TODAY",,"WEEKDAY")+".bck"
```

```
$ backup/since=today/mod/exclude=( [.my_backups...]*.*,my_backup.log) [...]*. * -
[.my_backups]'backup_file'/save
```

\$

```
$ submit/noprint my_backup
```

```
Job MY_BACKUP (queue LOW_RIDER, entry 27) started on LOW_RIDER
```

```
$ typex my_backup.log
```

```
%DCL-I-DEFKEY, DEFAULT key KP1 has been defined
```

```
%DCL-I-DEFKEY, DEFAULT key KP2 has been defined
```

```
STUDENT7      job terminated at 11-APR-1999 12:05:28.61
```

Accounting information:

Buffered I/O count:	221	Peak working set size:	4624
Direct I/O count:	292	Peak virtual size:	173472
Page faults:	382	Mounted volumes:	0
Charged CPU time:	0 00:00:01.84	Elapsed time:	0 00:00:06.74

\$

60. Submit the command procedure created in problem 56, part c to run in batch to run after 11:30 PM today.

- a. Verify that the job is scheduled to run.

Charged CPU time: 0 00:00:01.84 Elapsed time: 0 00:00:06.74

\$ submit/after=23:30/noprint my_backup

Job MY_BACKUP (queue SYS\$BATCH, entry 29) holding until 11-APR-1999 23:30

\$ sh que/batch sys\$batch

Generic batch queue SYS\$BATCH

Entry	Jobname	Username	Status
-----	-----	-----	-----
29	MY_BACKUP	STUDENT7	Holding until 11-APR-1999 23:30:00

\$

- b. Remove the entry.

\$ delete/entry=29

\$ sh que/batch sys\$batch

Generic batch queue SYS\$BATCH

\$

61. Create the following command procedure:

\$ create oops.com

\$loop: goto loop

Ctrl/Z

\$

- a. Submit the command procedure to run in batch.
- b. Use the \$SHOW SYSTEM/BATCH command to find the process running your batch job.

- c. Stop the process using the \$STOP *process-name* or \$STOP/ID=*pid* command.

```
$ create oops.com
```

```
$loop: goto loop
```

```
Exit
```

```
$ submit oops
```

```
Job OOPS (queue LOW_RIDER, entry 30) started on LOW_RIDER
```

```
$ show sys/batch
```

```
OpenVMS V7.2 on node ELLIS 11-APR-1999 12:08:16.57 Uptime 0 02:01:20
```

Pid	Process Name	State	Pri	I/O	CPU	Page flts	Pages
0000012D	BATCH_30	COM	4	56	0 00:00:06.62	118	75 B

```
$ stop batch_30
```

```
$ show sys/batch
```

```
$
```

62. If a printer is available, print a file. Print two copies of the file with a header.

```
$ print oops.com
```

```
Job OOPS (queue SYS$PRINT, entry 31) pending
```

```
$
```

```
$ print oops.com/header/copies=2
```

```
Job OOPS (queue SYS$PRINT, entry 32) pending
```

```
$
```

Optional Lab Solutions

1. Write a command procedure that searches files for a string and displays the names of files containing the matching string, but not the actual matching lines.

```
$ typex search_list.com
$ if p1 .eqs."" .or. p2.eqs.""
$ then
$     write sys$error "Bad command input"
$     exit
$ endif
$!! Create a unique scratch file name, using the PID and IMAGECOUNT
$ tmp_file="sys$scratch:_slist_."f$getjpi("", "PID")'f$getjpi("", "imagecount")'
$ search/nowarn/excl=(_slist*.*)/output='tmp_file'/header 'p1' 'p2'
$ open slist 'tmp_file'
$! Cleanup if error or control y
$ on error then goto done
$ on control_y then goto done
$ got_stars="NO"
$ read_loop:
$     read/end=done slist line
$ ! If previous line was header then this is file name, so print it
$     if got_stars
$     then
$         write sys$output line
$         got_stars="NO"
$     endif
$     if line.eqs."*****" then got_stars="YES"
$     goto read_loop
$ done:
$ close slist
$ delete 'tmp_file';
$
```

```
$ @search_list *.* bruce
SYS$SYSDEVICE:[STUDENT7]FIRST.DAT;1
SYS$SYSDEVICE:[STUDENT7]ME.COM;1
SYS$SYSDEVICE:[STUDENT7]ME.LOG;1
SYS$SYSDEVICE:[STUDENT7]NAMES.DAT;1
$ dir _sl*
%DIRECT-W-NOFILES, no files found
$
```

2. Write a command procedure that displays the number of images run and total page faults incurred by your process.

```
$ typex stats.com
$ line1=f$fa0("Image count: !SL",f$getjpi("", "imagecount"))
$ line2=f$fa0("Page faults: !SL",f$getjpi("", "pageflts"))
$ write sys$output line1
$ write sys$output line2
$
$ @stats
Image count: 129
Page faults: 15099
$
```

3. Write a command procedure that:
 - a. Saves the current default directory.
 - b. Sets default to a new directory.
 - c. Waits for one minute.
 - d. Generates an error.
 - e. Sets the default directory back to the original on errors and Ctrl/Y.

```
$ typex test.com
$!a. Save the current default directory.
$!b. Set default to a new directory.
$!c. Wait for one minute.
$!d. Generate an error.
$!e. Set the default directory back to the original on errors and control Y.
$ cur_dir=f$environment("DEFAULT")
$ set default sys$manager
```

```
$ on control_y then goto outta_here
```

```
$ on error then goto outta_here
```

```
$ wait 0 :1:.
```

```
$ type nonexistent.dat
```

```
$ outta_here:
```

```
$ set default 'cur_dir'
```

```
$ exit
```

```
$
```

```
$ show default
```

```
SYSSYSDEVICE:[STUDENT7]
```

```
$ @test
```

```
Interrupt
```

```
$ show default
```

```
SYSSYSDEVICE:[STUDENT7]
```

```
$ @test
```

```
%TYPE-W-SEARCHFAIL, error searching for SYSSCOMMON:[SYSMGR]NONEXISTENT.DAT;
```

```
-RMS-E-FNF, file not found
```

```
$ show default
```

```
SYSSYSDEVICE:[STUDENT7]
```

```
$
```

4. Write a command procedure that displays the percentage of space in use and available for all mounted disks.

```
$ typex pct.com
```

```
$! Command procedure to display utilization of mounted disks, using
```

```
$! F$DEVICE
```

```
$
```

```
$! read lines and extract the disk device names
```

```
$ get_next_device:
```

```
$ disk_name = f$device("*$D*")
```

```
$ if disk_name.eqs."" then goto thats_all_folks
```

```
$ if f$getdvi(disk_name,"MNT") .eqs. "TRUE"
```

```
$ then
$     total=f$getdvi(disk_name,"MAXBLOCK")
$     free=f$getdvi(disk_name,"FREEBLOCKS")
$     pct_avail=free*100/total
$     pct_inuse=(total-free)*100/total
$     pct_stat=f$fao("!AS: !2SL% in use !2SL% available",disk_name,pct_inuse,-
                    pct_avail)
$     write sys$output pct_stat
$ endif
$ goto get_next_device
$
$! close the file and delete it
$ thats_all_folks:

$
$ @pct
_ELLIS$DKA200:: 88% in use 11% available
$
```

5. Write a command procedure that edits, compiles, and links a program.

```
$ type build.com
$! Command procedure to edit, compile, link, and run a c program
$ if p1.eqs.""
$then
$     file="temp"
$else
$     file=P1
$endif
$step=1
$ re_edit:
$ if step.eq.2 then type/page=save 'file'
$ if step.eq.3 then type/page=save 'file'.map
$ define/user sys$input sys$command
$ edit 'file'.c
$ on warning then goto re_edit
$ step=2
```

```
$ write sys$output "Compiling..."
$ cc/list 'file'
$ on warning then goto re_edit
$ step=3
$ write sys$output "Linking..."
$ link/map 'file'
$ write sys$output "Running..."
$ run 'file'
$
$ @build
#include <stdio.h>
#include <stdlib.h>
int    main(void)
{
    printf("Hello, Galaxy!");
    return(EXIT_SUCCESS);
}
[End of file]
```

Buffer: TEMP.C

| Write | Insert | Forward

Command: exit

Editing new file. Could not find: SYS\$SYSDEVICE:[STUDENT7]TEMP.C;

7 lines written to file SYS\$SYSDEVICE:[STUDENT7]TEMP.C;1

Compiling...

```

        printf("Hello, Galaxy!");
.....^
%CC-E-NLSTRING, An unexpected newline character is present in a string literal.
at line number 5 in file SYS$SYSDEVICE:[STUDENT7]TEMP.C;1

```

Source Listing

11-APR-1999 20:5

11-APR-1999 20:5

```

        1 #include <stdio.h>
      874 #include <stdlib.h>
    2188 int      main(void)
1      2189 {
1      2190         printf("Hello, Galaxy!");
        .....1
%CC-E-NLSTRING, (1) An unexpected newLine character is present in a string liter

1      2191         rturn(EXIT_SUCCESS);
1      2192 }

```

Command Line

CC/LIST TEMP

These macros are in effect at the start of the compilation.

```

#include <stdio.h>
#include <stdlib.h>
int      main(void)
{
    printf("Hello, Galaxy!");
    rturn(EXIT_SUCCESS);
}
[End of file]

```

Buffer: TEMP.C | Write | Insert | Forward

Command: exit

7 lines read from file SYS\$SYSDEVICE:[STUDENT7]TEMP.C;1

7 lines written to file SYS\$SYSDEVICE:[STUDENT7]TEMP.C;2

Compiling...

```

        rturn(EXIT_SUCCESS);
.....^
%CC-I-IMPLICITFUNC, In this statement, the identifier "rturn" is implicitly declared as a function.
at line number 6 in file SYS$SYSDEVICE:[STUDENT7]TEMP.C;2

```

Linking...

```

%LINK-W-NUDFSyms, 1 undefined symbol:
%LINK-I-UDFSYM,          RTURN
%LINK-W-USEUNDEF, undefined symbol RTURN referenced
        in psect $LINK$ offset %X00000040
        in module TEMP file SYS$SYSDEVICE:[STUDENT7]TEMP.OBJ;1

```

C

11-APR-1999 21:0

```

+-----+
! Object Module Synopsis !
+-----+

```

```

%LINK-W-NUDFSyms, 1 undefined symbol:
%LINK-I-UDFSYM,          RTURN

```

Module Name	Ident	Bytes	File
-----	-----	-----	-----
TEMP	V1.0	432	SYS\$SYSDEVICE:[STUDENT7]TEMP.OBJ;1 1

```

%LINK-W-USEUNDEF, undefined symbol RTURN referenced
        in psect $LINK$ offset %X00000040
        in module TEMP file SYS$SYSDEVICE:[STUDENT7]TEMP.OBJ;1

```

+-----+
! Program Section Synopsis !
+-----+

Psect Name	Module Name	Base	End	Length	Align
-----	-----	----	---	-----	-----
\$LINK\$		00010000	0001009F	000000A0 (160.)	OCTA
RETURN/SPACE=More, PREV/NEXT=Scroll, INS/REM=Pan, SELECT=80/132, Q=Quit					
#include <stdio.h>					
#include <stdlib.h>					
int main(void)					
{					
printf("Hello, Galaxy!");					
return(EXIT_SUCCESS);					
}					
[End of file]					

Buffer: TEMP.C

| Write | Insert | Forward

Command: exit

7 lines read from file SYS\$SYSDEVICE:[STUDENT7]TEMP.C;2

7 lines written to file SYS\$SYSDEVICE:[STUDENT7]TEMP.C;3

Compiling...

Linking...

Running...

Hello, Galaxy!

\$