**13**

**Exporting**

**Objectives**

· Define the scope of a Shell variable

· Distinguish between local variables and exported variables

·Pass Shell variables to subshells



Notes

**Variables in Processes**

**A variable can be defined within a shell such that its existence is known only** **to that shell.**

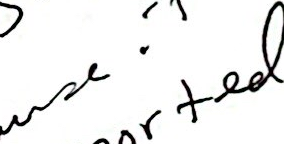
**· Example 1:**

$ cat vartest1

echo :$x:

**Assign a value to x**

x=100

**Run** vartest1

vartest1

#vartest1 does

#not know

#the value of x.

$ echo $x

100

**Notes**

·Use the export command to pass variable values to subshells.

Once a variable is exportd, it remains exported. You cannot unexport a variable.

These files are used in this chapter. You should create them for Walkthrough demonstrations.

*vartest1*

echo :$x:

vartest2

echo :$x:

vartest3



echo :$x:

echo :$y:

vartest4

echo :$x:

echo :$y:

echo :$y:

Notice the use of the colons.Values echoed from within vartest [1-4]are enclosed in colons.

Assign a value to x

$ x=100

Run vartest1

$vartest1

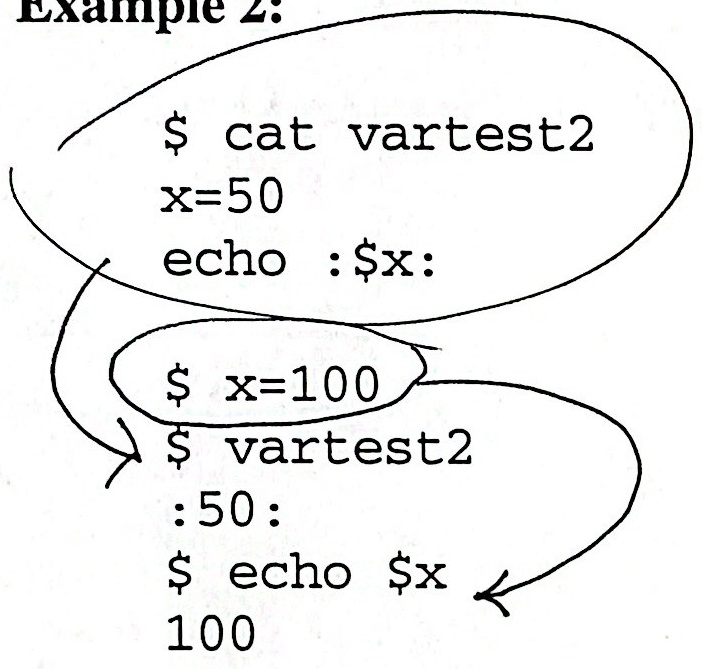
::

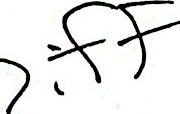
Echo the value of x

$ echo $x

100

**Variables in Processes(Cont'd)**

**Example 2:**



|  |  |  |
| --- | --- | --- |
|  | Variable in Shell Script  X | Variable in Parent Shell  X |
| vartest1 | nu11 | 100 |
| vartest2 | 50 | 100 |

**Notes**

vartest1 and vartest2 are subshells. They can assign values to local variables,but they have no knowledge of local variables that were assigned by the parent Shell.

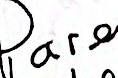
If one process calls another, the variable must be redefined or exported if it is to have an assigned value.

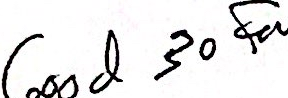
exported will be explained in detail later.

**Variables in Processes (Cont'd)**

**· Example 3:**

**Can a subshell change the values of the variables in the parent Shell?**

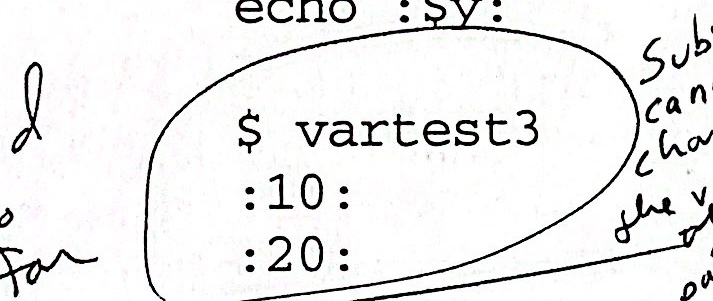
$ x=100;y=200

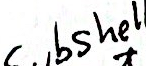
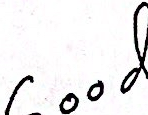
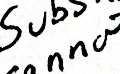
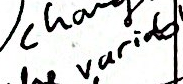
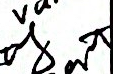
 100 200$ echo $x $y

$ cat vartest3

x=10

echo :$x:

echo :Sy:

|  |  |  |
| --- | --- | --- |
| Comment | Command | Result |
| define x | echo $0 | 100 |
| define y | y=200  echo $y | 200 |
| run vartest3 | vartest3 | :10:  :20: |
| determine the value  of x, y in the original Shell | echo $x  echo $y | 100  200 |



$ echo $x

100

$ echo $y

200

**Notes**

The values of the variables are modified as the subshell is executed, but upon completion of the execution the variables appear to revert to their original value because they were never changed in the parent shell! Hence, a subshell cannot change the values of the variables of a parent shell.

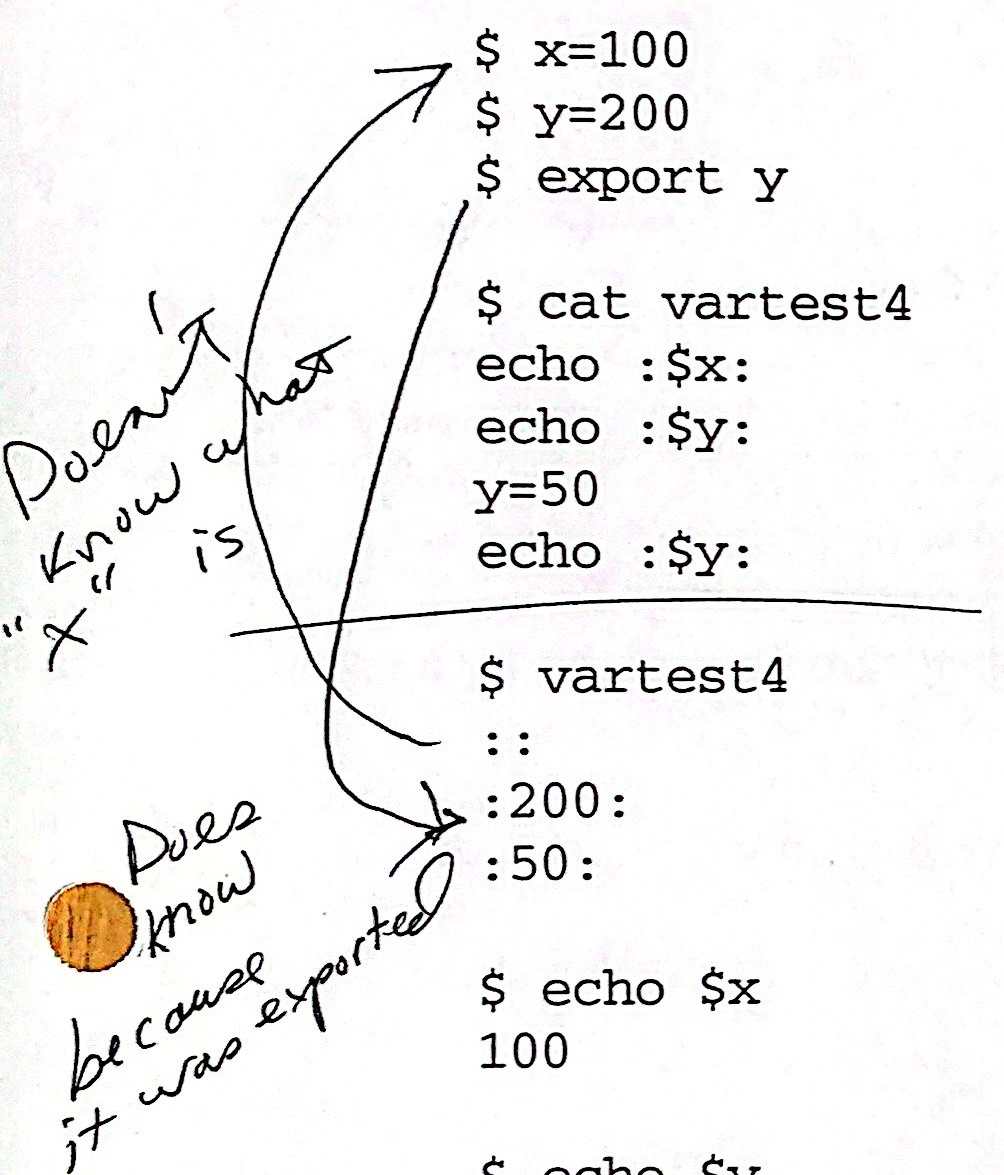
**Exporting Variables**

|  |  |  |
| --- | --- | --- |
| Comment | Command | Result |
| xis 100  y is 200 | echo $x  echo $y | 100  200 |
| export y | export y |  |
| run vartest4 | vartest4 | ::  :200:  :50: |
| x was not defined  with vartest4 |  |  |
| determine the value of x,y in the original Shell | echo $x  echo $y | 100  200 |

**Example:**

200

**Notes**

It is possible to pass a variable to a child process using the export command.When a variable is exported, all subsequent subshells know the value of the exported variable.

The value of the variable can be changed in the subshell, but when the subshell dies, the value returns to the value defined by the parent Shell.



**Exporting Notes**

·Values that are to remain constant can be declared with the readonly command.

The Shell maintainsthe user environment.

A subshell has no knowledge of local variables that may have been assigned by a parent Shell. must

When a subshell dies,the values of any locally assigned variables disappear.

To make a variable known to a subshell, it must be exported.



The value of a variable defined by a parent Shell cannot be globally changed from within a subshell.

Once a variable is exported, it remains exported to all subsequent subshells.

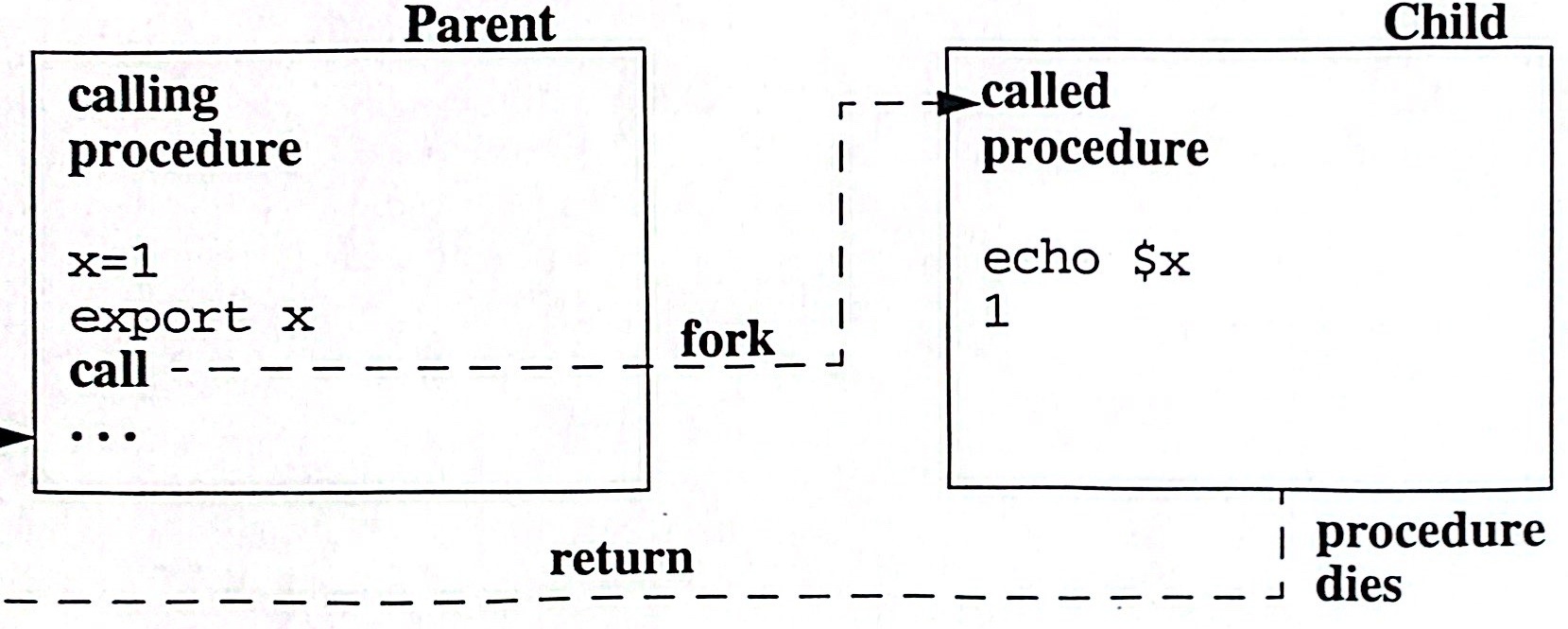
**Notes**

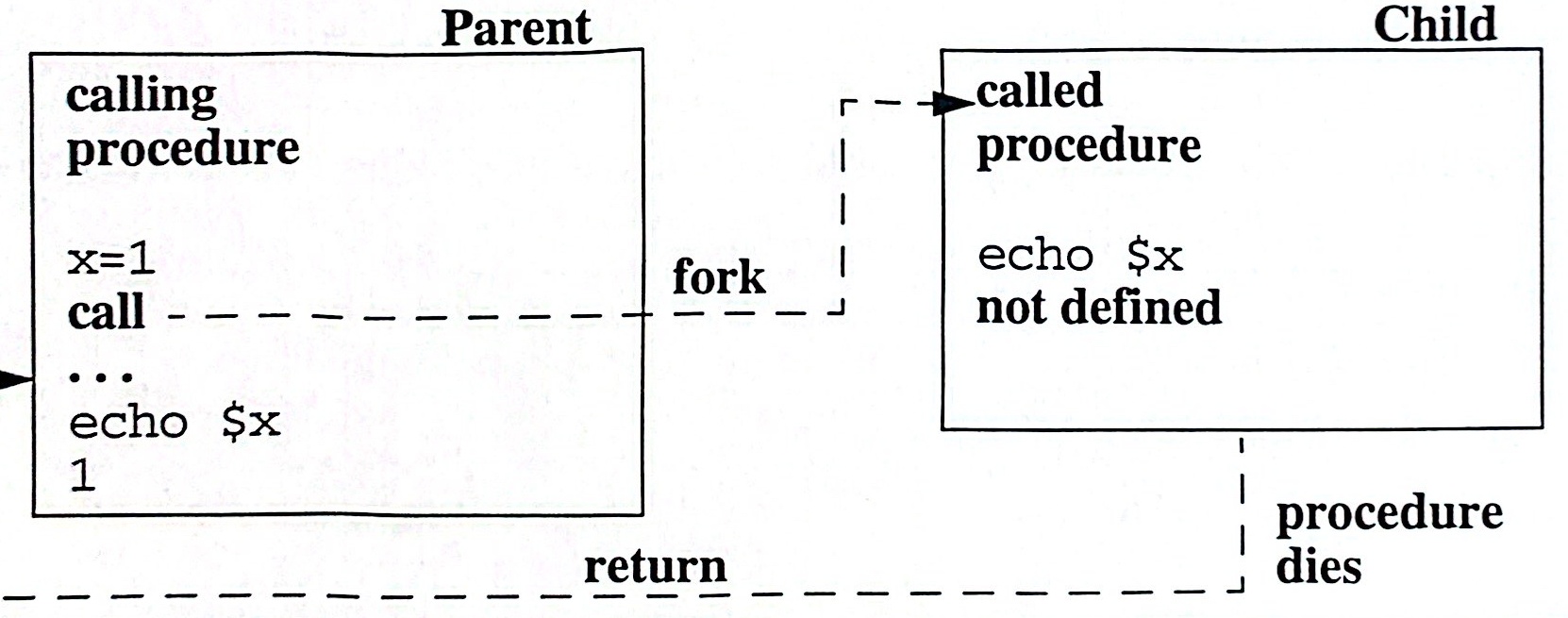
To pass a defined variable to another Shell, you must first export the variable so it will be known in your new environment.

When a Shell procedure is called, copies are made of all exported variables. Modification of the variables within the called procedure does not affect the values in the calling procedure.



**Exporting**





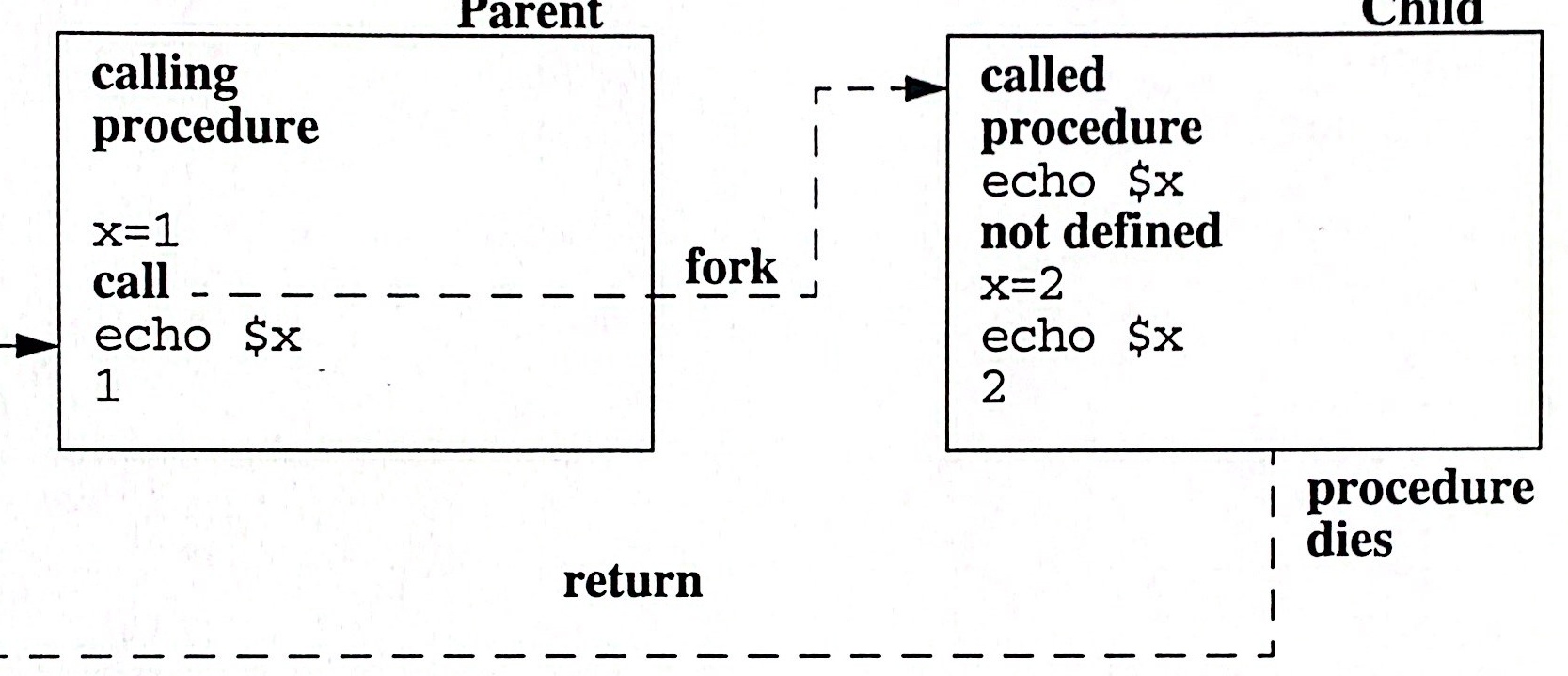
**Notes**

A variable exported in the current Shell is known in all child processes spawned by the current Shell.

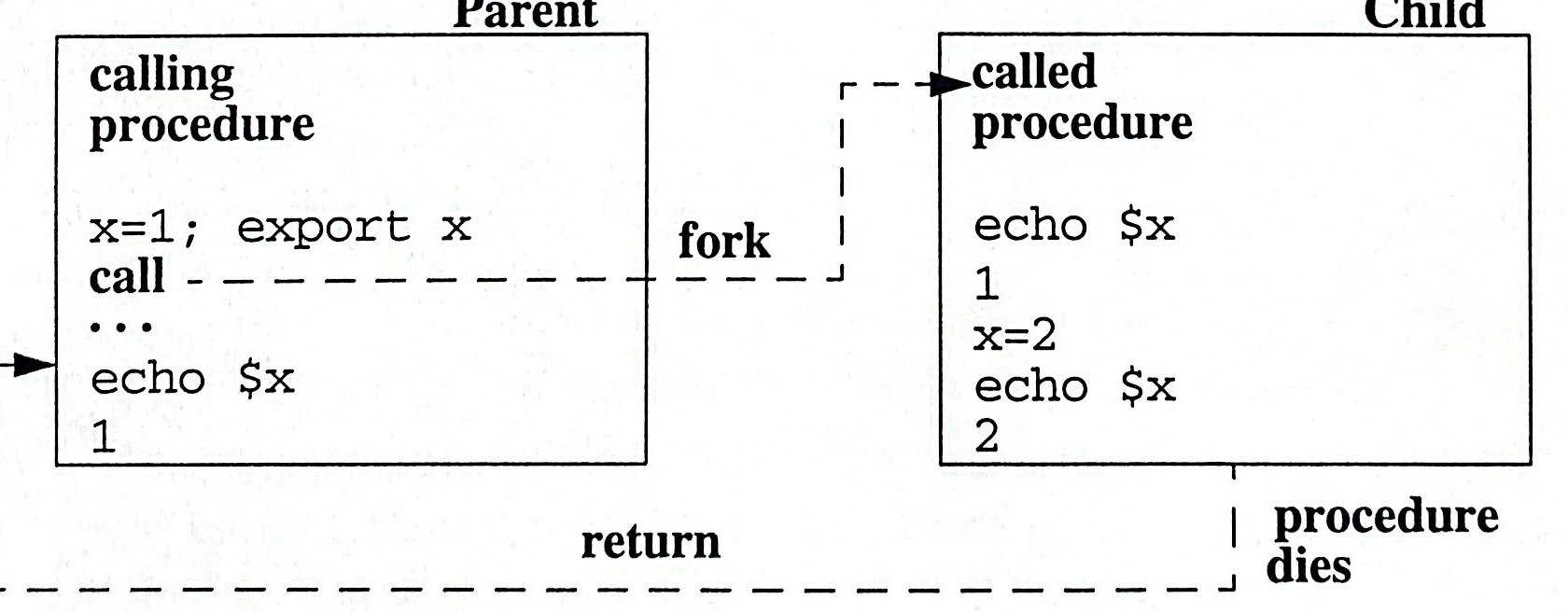
To determine what variables are exported from the current Shell type: export

The readonly variables will also be listed.

**Exporting(Cont'd)**



**After the called procedure terminates, the original value of x is displayed.**



**Notes**