**NAME**

switch − shell multi-way branch command

**SYNOPSIS**

**switch** arg
: label1
commands...
**breaksw**
. . .
: labeln
commands...
**breaksw**
**: default**
commands...
**endsw**

**DESCRIPTION**

*Switch* searches forward in the input file for the first one of:

1. a label that pattern-matches *arg.* The pattern-matching used is that of the Shell in generating argument lists.
2. the label *default.*
3. a matching *endsw* command.

The Shell resumes reading commands from the next line after the location where the search stopped. Thus, *switch* supplies a ‘case’ or ‘computed goto’ statement similar to that of C. Because ‘:’ is ignored by the Shell, several labels may occur in order, so that the same sequence of commands is executed for several different values of *arg.*

The *breaksw* command searches forward to the next unmatched *endsw,* and is normally used at the end of the sequence of commands following each label. It may be omitted to allow common code to be shared among label values. Several *breaksw* commands may be written on the same line to exit from that many levels of nested *switch*−*endsw* pairs.

The optional label *default* should be placed last, since *switch* always stops upon discovering it. The construct can be nested: any labels enclosed by a *switch-endsw* pair are ignored by an outer *switch.* The most common use of *switch* is to process ‘flag’ arguments in a shell procedure.

**SEE ALSO**

if(I), sh(I), while(I)

**DIAGNOSTICS**

switch: missing endsw
breaksw: missing endsw

**BUGS**

None of these commands should be hidden behind semicolons. Nested groups hidden behind *if* or *else* may also cause trouble.