

for — for (*exp1*; *exp2*; *exp3*) *stmt*

[Description](#)[Syntax](#)[Remarks and examples](#)[Also see](#)

Description

`for` is equivalent to

```

exp1
while (exp2) {
    stmt(s)
    exp3
}

```

stmt(s) is executed zero or more times. The loop continues as long as *exp2* is not equal to zero.

Syntax

```
for (exp1; exp2; exp3) stmt
```

```

for (exp1; exp2; exp3) {
    stmts
}

```

where *exp1* and *exp3* are optional, and *exp2* must evaluate to a real scalar.

Remarks and examples

stata.com

To understand `for`, enter the following program

```

function example(n)
{
    for (i=1; i<=n; i++) {
        printf("i=%g\n", i)
    }
    printf("done\n")
}

```

and run `example(3)`, `example(2)`, `example(1)`, `example(0)`, and `example(-1)`.

Common uses of `for` include

```

for (i=1; i<=rows(A); i++) {
    for (j=1; j<=cols(A); j++) {
        ...
    }
}

```

Also see

[M-2] **break** — Break out of for, while, or do loop

[M-2] **continue** — Continue with next iteration of for, while, or do loop

[M-2] **do** — do ... while (exp)

[M-2] **Semicolons** — Use of semicolons

[M-2] **while** — while (exp) stmt

[M-2] **Intro** — Language definition