

```
#####  
Script started on Sun Aug 23 16:32:23 1998
```

```
#####  
newton 21> more Makefile
```

```
.IGNORE:
```

```
F77          = f77  
F77FLAGS    = -g -n32  
F77CFLAGS   = -c  
F77LFLAGS   = -L/usr/localn32/lib -n32
```

```
F77_COMPILE = $(F77) $(F77FLAGS) $(F77CFLAGS)  
F77_LOAD    = $(F77) $(F77FLAGS) $(F77LFLAGS)
```

```
.f.o:  
    $(F77_COMPILE) $*.f
```

```
EXECUTABLES = fdemo1
```

```
all: $(EXECUTABLES)
```

```
fdemo1: fdemo1.o  
    $(F77_LOAD) fdemo1.o -o fdemo1
```

```
clean:  
    rm *.o  
    rm $(EXECUTABLES)
```

```
#####  
newton 22> make  
    make -f Makefile  
    f77 -g -n32 -c fdemo1.f  
    f77 -g -n32 -L/usr/localn32/lib -n32 fdemo1.o -o fdemo1
```

```
#####  
# I encourage you to download 'fdemo1.f', compile it,  
# and run it INTERACTIVELY yourself. You should see  
# output essentially identical to that shown below.  
# Note, however, that both because I'm lazy, as well  
# as to illustrate the use of I/O re-direction, I have  
# previously prepared a file called 'INPUT', which  
# contains many lines consisting of a single character  
# These lines will be read by the 'prompt' subroutine  
# which, when run interactively, writes a prompt to  
# stdout and then waits for input from stdin.
```

```
#####
```

```
newton 23> head -10 INPUT
```

```
q  
q  
q  
q  
q  
q  
q  
q  
q  
q
```

```

newton 24> fdemo1 < INPUT
a = 2.5000000000000001E-02 b = -1.2339999999999999E-16
c = 1.0000000000000000 i = 3000 switch = T
Through scalar assignment
#####
# Note: For readability, all other instances of the
# following output from the 'prompting' routine have been
# converted to blank lines with a text editor command.
#####
Enter any non-blank character & enter to continue
res1 = 5.0000000000000000 res2 = 13.0000000000000000
res3 = 3.605551275463989
Through real*8 arithmetic expressions

ires1 = 5 ires2 = 0
ires3 = 512 ires4 = 64
Through integer arithmetic expressions

res1 = 5.0000000000000000 res2 = 0.0000000000000000E+00
res3 = 0.7500000000000000
Through mixed-mode arithmetic

Loop 1: i = 1
Loop 1: i = 2
Loop 1: i = 3
Through loop 1

Loop 2: i = 1
Loop 2: i = 2
Loop 2: i = 3
Through loop 2

Loop 3: i = 1
Loop 3: i = 3

```

Loop 3: i = 5
Loop 3: i = 7
Through loop 3

Loop 4: i = 3
Loop 4: i = 2
Loop 4: i = 1
Through loop 4

Loop 5: i, j = 1 1
Loop 5: i, j = 1 2
Loop 5: i, j = 2 1
Loop 5: i, j = 2 2
Loop 5: i, j = 3 1
Loop 5: i, j = 3 2
Through loop 5

Loop 6: i = 2
Loop 6: i = 4
Loop 6: i = 6
Through loop 6

lres1 = T lres2 = T lres3 = F
Through basic conditionals

25.000000000000000 > 12.000000000000000
Through if 1

25.000000000000000 > 12.000000000000000
Through if 2

25.000000000000000 > 24.000000000000000
Through nested if

Case 1
Case 2
Case 3
Default case
Through case via if

Do while loop: b = 0.0000000000000000E+00
Do while loop: b = 0.1000000000000000
Do while loop: b = 0.2000000000000000
Do while loop: b = 0.3000000000000000
Do while loop: b = 0.4000000000000000
Do while loop: b = 0.5000000000000000
Do while loop: b = 0.6000000000000000
Do while loop: b = 0.7000000000000000
Do while loop: b = 0.7999999999999999
Do while loop: b = 0.8999999999999999
Do while loop: b = 0.9999999999999999
Through while loop

res1 = 0.8090169943749473 res2 = 0.5877852522924732
res3 = 1.0000000000000000 res4 = 1.0000000000000000
Through built-in fcn 1

res1 = 0.7853981633974483
Through built-in fcn 2

min(3.0d0,2.0d0) = 2.0000000000000000
min(1,-3,5,0) = -3
Through built-in fcn 3

i = 0
i = 100
i = 200
i = 300

```
i =      400
i =      500
i =      600
i =      700
i =      800
i =      900
i =     1000
```

```
Through built-in fcn 4
```

```
Through fdemo1
```

```
newton 25> exit
```

```
newton 26>
```

```
script done on Sun Aug 23 16:33:26 1998
```