```
#include    <stdio.h>

#define     MAXLINELENGTH     80

int main ()       {
      char  line[MAXLINELENGTH];
      int   ch, length;

      while ( (ch = getchar ()) != EOF ) {
            length = 0;                           /* Read a line */
            while ( length < MAXLINELENGTH ) {
                  if (ch == '\n') break;
                  line[length++] = ch;
                  if ( (ch = getchar ()) == EOF ) break;
            }
            while ( length > 0 ) {        /* Print the line reversed */
                  putchar ( line[--length] );
            }
            putchar ( '\n' );
      }
      return 0;
}

/*
This is a test
tset a si sihT
and so is this
siht si os dna
What about EOF
FOE tuoba tahW

Under Unix an EOF can be supplied by entering Cntrl-D
Under Unix a program can be terminated by entering Cntrl-C
Under Unix a program can be suspended by entering Cntrl-Z
*/

/*---------------------------------------------------------------------------*/
```

```
all:     file.dvi file.ps twopage.ps
         ls -lpg *ps

twopage.ps:     file.ps
         psnup  -2 -pletter file.ps >twopage.ps

file.ps:   file.dvi
         dvips -t letter file.dvi

file.dvi: file.tex myreport.cls
         latex file.tex
         latex file.tex
```

```
/*--------------------------------------------------------------------------*/
```

```
#include <stdio.h>
int main ()
{
      char c;
      short int p;
      int q;
      unsigned int v;
      long int r;
      float s;
      double t;
      long double u;

      printf ("Size of Variables:\n %d %d %d %d %d %d %d %d \n", sizeof c,
            sizeof p, sizeof q, sizeof r, sizeof s,
            sizeof t, sizeof u, sizeof v );
      printf ("Size of (Types):\n %d %d %d %d %d %d %d %d \n", sizeof (char),
            sizeof(short int), sizeof(int), sizeof(long int), sizeof(float),
            sizeof(double), sizeof(long double), sizeof(unsigned int));
      return 0;
}

/*
Size of Variables:
 1 2 4 4 4 8 8 4
Size of (Types):
 1 2 4 4 4 8 8 4
*/
/*  Advice:  Just write sizeof(something) */

/*------------------------------------------------------------------------*/
# To reduce errors of repetition the make facility provides a macro
#    (or define) facility.
#    Thus in practice the make file might really look like

CREATOR   =   gcc -Wall -ansi

phone :    main.o phone.o
           $(CREATOR) -o phone main.o phone.o

phone.o :  phone.c
           $(CREATOR) -c phone.c

main.o :   main.c phone.h
           $(CREATOR) -c main.c

#    In the above the macro CREATOR is defined to be "gcc -Wall -ansi"
#    and the macro is expanded whenever $(CREATOR) appears.
#    The names you choose for macros are entirely your own.
#    See the assignment 1 statement for a different example of a makefile:
#    Namely to build and run a series of experiments, recompiling only those
#    that you modify.

#    Two useful tests on makefiles are:
#    make -vet makefile
#    checks that each command line of the makefile begins with a TAB character
#    make -n phone
#    shows what commands will be executed IF make phone is requested.
```