Revision 1

```
/* text formatter version 1; textv1.c */

#include <stdio.h>

extern void filecopy( FILE * fp );

main( int argc, char *argv[] )
  {
    FILE *fp;

    if( argc == 1 )
      {
        printf( "no file specified\n" );
      }
    else
      {
        fp = fopen( *++argv, "r" );
        if( fp == NULL )
          {
            printf( "unable to open %s\n", *argv );
          }
        else
          {
            filecopy( fp );
            fclose( fp );
          }
      }
  }

/* copy file to standard output one character per line */

extern void filecopy( FILE *fp )
  {
    int c;

    for( ; ; )
      {
        c = getc( fp );
        if( c == EOF )
          {
            break;
          }
        printf( "here is a character...  %c\n", c );
      }
  }
```

Revision 2

```
/* text formatter v. 2; textv2.c */

#include <stdio.h>
#include "textdef.h"

extern void Openword( void );
extern void Closewrd( void );
extern void Makeword( const char c );

static void format( FILE *fp );

main( int argc, char *argv[] )
  {
    FILE *fp;

    if( argc == 1 )
      {
        printf( "no file specified\n" );
      }
    else
      {
        fp = fopen( *++argv, "r" );
        if( fp == NULL )
          {
            printf( "unable to open %s\n", *argv );
          }
        else
          {
            Openword();
            format( fp );
            Closewrd();
            fclose( fp );
          }
      }
  }

/*  copy file to standard output one character per line */

static void format( FILE *fp )
  {
    int c;

    for ( ; ; )
      {
        c = getc( fp );
        if( c == EOF )
          {
            break;
          }
        Makeword( c );
      }
  }
```

```
/* file: wordpv1.c */

#include <stdio.h>
#include "textdef.h"

static char Word[ MAXWORD ]; /* word buffer */
static int Noofchar;         /* no of chars in word */

extern void Openword( void )
  {
    Noofchar = 0;
  }

extern void Closewrd( void )
  {
  }

/* compose characters into words */

extern void Makeword( const char c )
  {
    if( c == ' ' || c == '\n' )
      {
        Word[ Noofchar ] = NULL;
        printf( "Here comes a word... %s\n", Word );
        Noofchar = 0;
      }
    else
      {
        Word[ Noofchar++ ] = c;
      }
  }
```

Revision 3

```
/* text formatter v. 2; textv2.c */

#include <stdio.h>
#include "textdef.h"

extern void Openword( void );
extern void Closewrd( void );
extern void Makeword( const char c );

static void format( FILE *fp );

main( int argc, char *argv[] )
  {
    FILE *fp;

    if( argc == 1 )
      {
        printf( "no file specified\n" );
      }
    else
      {
        fp = fopen( *++argv, "r" );
        if( fp == NULL )
          {
            printf( "unable to open %s\n", *argv );
          }
        else
          {
            Openword();
            format( fp );
            Closewrd();
            fclose( fp );
          }
      }
  }

/*  copy file to standard output one character per line */

static void format( FILE *fp )
  {
    int c;

    for ( ; ; )
      {
        c = getc( fp );
        if( c == EOF )
          {
            break;
          }
        Makeword( c );
      }
  }
```

```
/* file: wordpv2.c */

#include <stdio.h>
#include "textdef.h"

extern void Openline( void );
extern void Closelin( void );
extern void Makeline( char word[], int length );

static char Word[ MAXWORD ]; /* word buffer */
static int Noofchar;         /* no of chars in word */

extern void Openword( void )
  {
    Noofchar = 0;
    Openline();
  }

extern void Closewrd( void )
  {
    Closelin();
  }

extern void Makeword( const char c )
/*
   compose characters into words
*/
  {
    if( c == ' ' || c == '\n' )
      {
        Word[ Noofchar ] = NULL;
        Makeline( Word, Noofchar );
        Noofchar = 0;
      }
    else
      {
        Word[ Noofchar++ ] = c;
      }
  }
```

```
/* file: linepv1.c */

#include <stdio.h>
#include <string.h>
#include "textdef.h"

static char Line[ MAXLINE+1 ]; /* word buffer */
static int Noofchar;           /* chars per line */

static void copyword( char word[], int length );

extern void Openline( void )
  {
    Noofchar = 0;
  }

extern void Closelin( void )
  {
    Line[ Noofchar-1 ] = NULL;
    printf( "Here comes a line... %s\n", Line );
  }

extern void Makeline( char word[], int length )
/*
   compose words into lines
*/
  {
    if( length > RIGHT_MARGIN - Noofchar )
      {
        Line[ Noofchar-1 ] = NULL;
        printf( "Here comes a line... %s\n", Line );
        Noofchar = 0;
        copyword( word, length );
      }
    else
      {
        copyword( word, length );
      }
  }

/*
   copy word into line
*/
static void copyword( char word[], int length )
  {
    strcpy( &Line[ Noofchar ], word );
    Noofchar = Noofchar + length;
    if( length != 0 )
      {
        Line[ Noofchar ] = ' ';
        Noofchar++;
      }
  }
```

Revision 4

```
/* text formatter v. 2; textv2.c */

#include <stdio.h>
#include "textdef.h"

extern void Openword( void );
extern void Closewrd( void );
extern void Makeword( const char c );

static void format( FILE *fp );

main( int argc, char *argv[] )
  {
    FILE *fp;

    if( argc == 1 )
      {
        printf( "no file specified\n" );
      }
    else
      {
        fp = fopen( *++argv, "r" );
        if( fp == NULL )
          {
            printf( "unable to open %s\n", *argv );
          }
        else
          {
            Openword();
            format( fp );
            Closewrd();
            fclose( fp );
          }
      }
  }

/* copy file to standard output one character per line */

static void format( FILE *fp )
  {
    int c;

    for ( ; ; )
      {
        c = getc( fp );
        if( c == EOF )
          {
            break;
          }
        Makeword( c );
      }
  }
```

```
/* file: wordpv3.c */
#include <stdio.h>
#include "textdef.h"

extern void Openline( void );
extern void OpenRecord( void );
extern void Closelin( void );
extern void CloseRecord( void );
extern void Sendword( char word[], int length );

static char Word[ MAXWORD ]; /* word buffer */
static int Noofchar;         /* no of chars in word */

extern void Openword( void )
  {
    Noofchar = 0;
    Openline();
    OpenRecord();
  }

extern void Closewrd( void )
  {
    Closelin();
    CloseRecord();
  }

/* compose characters into words */
extern void Makeword( const char c )
  {
    if( c == ' ' )
      {
        Word[ Noofchar ] = NULL;
        if( Noofchar != 0 )
          {
            Sendword( Word, Noofchar );
          }
        Noofchar = 0;
      }
    else if( c == '\n' )
      {
        Word[ Noofchar ] = NULL;
        CloseRecord();
        if( Noofchar != 0 )
          {
            Sendword( Word, Noofchar );
          }
        Noofchar = 0;
        OpenRecord();
      }
    else
      {
        Word[ Noofchar++ ] = c;
      }
  }
```

```
/* file: editpv1.c */

#include <stdio.h>
#include <string.h>
#include "textdef.h"

extern void Makeline( char word[], int length );
extern Boolean IsCommnd( char word[] );
extern void FlushLine( void );
extern void SetStRec( Boolean newvalue );
extern void SetEndRc( Boolean newvalue );
extern Boolean IsStRec( void );
extern Boolean IsEndRec( void );

static void MakeCommandline( char word[],
                             int length );
static void copyword( char word[], int length );

static char CommandLine[ MAXLINE+1 ];
static int Noofchar;
static Boolean Command;

extern void OpenRecord( void )
  {
    Noofchar = 0;
    SetStRec( TRUE );
    SetEndRc( FALSE );
    Command = FALSE;
  }

extern void CloseRecord( void )
  {
    SetEndRc( TRUE );
  }
```

```
extern void Sendword( char word[], int length )
  {
    if( IsStRec() == TRUE )
      {
        SetStRec( FALSE );
        Command = IsCommnd( word );
        if( Command == FALSE )
          {
            Makeline( word, length );
          }
        else
          {
            FlushLine();
            MakeCommandline( word, length );
          }
      }
    else
      {
        if( Command == FALSE )
          {
            Makeline( word, length );
          }
        else
          {
            MakeCommandline( word, length );
          }
      }
  }

static void MakeCommandline( char word[],
                             int length )
/*
   make lines from words
*/
  {
    if( (length > MAXRECORD - Noofchar ) ||
        (IsEndRec() ==  TRUE) )
      {
        copyword( word, length );
        CommandLine[ Noofchar-1 ] = NULL;
        printf( "Here is a command line... %s\n",
                CommandLine );
        Noofchar = 0;
      }
    else
      {
        copyword( word, length );
      }
  }
```

```
static void copyword( char word[], int length )
/*
   copy word into line
*/
  {
    strcpy( &CommandLine[ Noofchar ], word );
    Noofchar = Noofchar + length;
    if( length != 0 )
      {
        CommandLine[ Noofchar ] = ' ';
        Noofchar++;
      }
  }
```

```
/* file: linepv2.c */

#include <stdio.h>
#include <string.h>
#include "textdef.h"

static void copyword( char word[], int length );

static char Line[ MAXLINE+1 ]; /* word buffer */
static int Noofchar;           /* chars per line */

extern void Openline( void )
  {
    Noofchar = 0;
  }

extern void Closelin( void )
  {
    Line[ Noofchar-1 ] = NULL;
    printf( "Here comes a line... %s\n", Line );
    Noofchar = 0;
  }

extern void FlushLine( void )
  {
    if( Noofchar > 0 )
      {
        Closelin();
      }
  }

extern void Makeline( char word[], int length )
/* compose words into lines */

{
    if( length > RIGHT_MARGIN - Noofchar )
      {
        Line[ Noofchar-1 ] = NULL;
        printf( "Here comes a line... %s\n", Line );
        Noofchar = 0;
        copyword( word, length );
      }
    else
      {
        copyword( word, length );
      }
  }
```

```
static void copyword( char word[], int length )
/*
   copy word into line
*/
  {
    strcpy( &Line[ Noofchar ], word );
    Noofchar = Noofchar + length;
    if( length != 0 )
      {
        Line[ Noofchar ] = ' ';
        Noofchar++;
      }
  }
```

```
/* file: commpv1.c */

#include <stdio.h>
#include "textdef.h"

static Boolean StartRec, EndRec;

extern Boolean IsCommnd( char word[] )
/*
   command processor for imbedded commands
*/
  {
    Boolean value;
    if( word[ 0 ] == ':' )
      {
        printf( "here is a command...%s\n", word );
        value = TRUE;
      }
    else
      {
        value = FALSE;
      }
    return( value );
  }

extern Boolean IsStRec( void )
  {
    return( StartRec );
  }

extern Boolean IsEndRec( void )
  {
    return( EndRec );
  }

extern void SetStRec( Boolean incoming )
  {
    StartRec = incoming;
  }

extern void SetEndRc( Boolean incoming )
  {
    EndRec = incoming;
  }
```

Shared header file

```
/* file name textdef.h */

#define Boolean int   /* for Boolean Variables */

#define TRUE  (0==0)  /* TRUE */
#define FALSE (1==0)  /* FALSE */

#define RIGHT_MARGIN 40 /* right margin */
#define MAXWORD      80 /* length of longest word */
#define MAXLINE     132 /* length of longest line */
#define MAXRECORD    80 /* length of longest record */
```