Q&A

Always store the return value of `fgetc`, `getc`, or `getchar` in an `int` variable, not a `char` variable. Testing a `char` variable against `EOF` may give the wrong result.

ungetc

There's one other character input function, `ungetc`, which "pushes back" a character read from a stream and clears the stream's end-of-file indicator. This capability can be handy if we need a "lookahead" character during input. For instance, to read a series of digits, stopping at the first nondigit, we could write

`isdigit` function ➤23.5

```
while (isdigit(ch = getc(fp))) {
  ...
}
ungetc(ch, fp);    /* pushes back last character read */
```

The number of characters that can be pushed back by consecutive calls of `ungetc`—with no intervening read operations—depends on the implementation and the type of stream involved; only the first call is guaranteed to succeed. Calling a file-positioning function (`fseek`, `fsetpos`, or `rewind`) causes the pushed-back characters to be lost.

file-positioning functions ➤22.7

`ungetc` returns the character it was asked to push back. However, it returns `EOF` if an attempt is made to push back `EOF` or to push back more characters than the implementation allows.

PROGRAM

Copying a File

The following program makes a copy of a file. The names of the original file and the new file will be specified on the command line when the program is executed. For example, to copy the file `f1.c` to `f2.c`, we'd use the command

```
fcopy f1.c f2.c
```

`fcopy` will issue an error message if there aren't exactly two file names on the command line or if either file can't be opened.

fcopy.c

```
/* Copies a file */

#include <stdio.h>
#include <stdlib.h>

int main(int argc, char *argv[])
{
  FILE *source_fp, *dest_fp;
  int ch;
```