(b)
```
strcpy(s, "0123");
i = 0;
putchar(TOUPPER(s[++i]));
```

6. (a) Write a macro `DISP(f,x)` that expands into a call of `printf` that displays the value of the function `f` when called with argument `x`. For example,

```
DISP(sqrt, 3.0);
```

should expand into

```
printf("sqrt(%g) = %g\n", 3.0, sqrt(3.0));
```

(b) Write a macro `DISP2(f,x,y)` that's similar to `DISP` but works for functions with two arguments.

Ⓦ *7. Let GENERIC_MAX be the following macro:

```
#define GENERIC_MAX(type)           \
type type##_max(type x, type y)  \
{                                   \
  return x > y ? x : y;             \
}
```

(a) Show the preprocessor's expansion of `GENERIC_MAX(long)`.

(b) Explain why GENERIC_MAX doesn't work for basic types such as `unsigned long`.

(c) Describe a technique that would allow us to use GENERIC_MAX with basic types such as `unsigned long`. *Hint:* Don't change the definition of GENERIC_MAX.

*8. Suppose we want a macro that expands into a string containing the current line number and file name. In other words, we'd like to write

```
const char *str = LINE_FILE;
```

and have it expand into

```
const char *str = "Line 10 of file foo.c";
```

where `foo.c` is the file containing the program and 10 is the line on which the invocation of `LINE_FILE` appears. *Warning:* This exercise is for experts only. Be sure to read the Q&A section carefully before attempting!

9. Write the following parameterized macros.

(a) `CHECK(x,y,n)` – Has the value 1 if both `x` and `y` fall between 0 and `n` – 1, inclusive.

(b) `MEDIAN(x,y,z)` – Finds the median of `x`, `y`, and `z`.

(c) `POLYNOMIAL(x)` – Computes the polynomial $3x^5 + 2x^4 - 5x^3 - x^2 + 7x - 6$.

10. Functions can often—but not always—be written as parameterized macros. Discuss what characteristics of a function would make it unsuitable as a macro.

11. (C99) C programmers often use the `fprintf` function to write error messages:

fprintf function ➤22.3

stderr stream ➤22.1

```
fprintf(stderr, "Range error: index = %d\n", index);
```

`stderr` is C's "standard error" stream; the remaining arguments are the same as those for `printf`, starting with the format string. Write a macro named ERROR that generates the call of `fprintf` shown above when given a format string and the items to be displayed:

```
ERROR("Range error: index = %d\n", index);
```

Section 14.4

Ⓦ 12. Suppose that the macro `M` has been defined as follows:

```
#define M 10
```