

The program will indicate whether or not each string is a planet name; if it is, the program will also display the planet's number (with planet 1 being the one closest to the Sun):

```
Jupiter is planet 5
venus is not a planet
Earth is planet 3
fred is not a planet
```

Notice that the program doesn't recognize a string as a planet name unless its first letter is upper-case and its remaining letters are lower-case.

```
planet.c  /* Checks planet names */

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

#define NUM_PLANETS 9

int main(int argc, char *argv[])
{
    char *planets[] = {"Mercury", "Venus", "Earth",
                       "Mars", "Jupiter", "Saturn",
                       "Uranus", "Neptune", "Pluto"};

    int i, j;

    for (i = 1; i < argc; i++) {
        for (j = 0; j < NUM_PLANETS; j++)
            if (strcmp(argv[i], planets[j]) == 0) {
                printf("%s is planet %d\n", argv[i], j + 1);
                break;
            }
        if (j == NUM_PLANETS)
            printf("%s is not a planet\n", argv[i]);
    }

    return 0;
}
```

The program visits each command-line argument in turn, comparing it with the strings in the `planets` array until it finds a match or reaches the end of the array. The most interesting part of the program is the call of `strcmp`, in which the arguments are `argv[i]` (a pointer to a command-line argument) and `planets[j]` (a pointer to a planet name).

Q & A

Q: How long can a string literal be?

A: According to the C89 standard, compilers must allow string literals to be at least