Figure C.7 Print the time and date in a format similar to `date`(1)

Running this program gives us

```
$ ./a.out                              author's default is US/Eastern
Wed Jul 25 22:58:32 EDT 2012
$ TZ=US/Mountain ./a.out               U.S. Mountain time zone
Wed Jul 25 20:58:32 MDT 2012
$ TZ=Japan ./a.out                     Japan
Thu Jul 26 11:58:32 JST 2012
```

Chapter 7

7.1 It appears that the return value from `printf` (the number of characters output) becomes the return value of `main`. To verify this theory, change the length of the string printed and see if the new length matches the return value. Note that not all systems exhibit this property. Also note that if you enable the ISO C extensions in `gcc`, then the return value is always 0, as required by the standard.

7.2 When the program is run interactively, standard output is usually line buffered, so the actual output occurs when each newline is output. If standard output were directed to a file, however, it would probably be fully buffered, and the actual output wouldn't occur until the standard I/O cleanup is performed.

7.3 On most UNIX systems, there is no way to do this. Copies of `argc` and `argv` are not kept in global variables like `environ` is.

7.4 This provides a way to terminate the process when it tries to dereference a null pointer, a common C programming error.

7.5 The definitions are

```
typedef void    Exitfunc(void);

int atexit(Exitfunc *func);
```

7.6 `calloc` initializes the memory that it allocates to all zero bits. ISO C does not guarantee that this is the same as either a floating-point 0 or a null pointer.

7.7 The heap and the stack aren't allocated until a program is executed by one of the `exec` functions (described in Section 8.10).

7.8 The executable file (`a.out`) contains symbol table information that can be helpful in debugging a `core` file. To remove this information, use the `strip`(1) command. Stripping the two `a.out` files reduces their size to 798,760 and 6,200 bytes.

7.9 When shared libraries are not used, a large portion of the executable file is occupied by the standard I/O library.

7.10 The code is incorrect, since it references the automatic integer `val` through a pointer after the automatic variable is no longer in existence. Automatic variables declared after the left brace that starts a compound statement disappear after the matching right brace.