Assignment 1

cpe 357 Fall 2010

Note

This initial assignment is intended as a warm up to get everyone started quickly in this class. It shouldn't cause anyone too much pain, but it should knock off some of the rust and reduce the shock to the system brought on by starting a new academic year.
It should not be taken as an indication of the difficulty of the other assignments in the course.

```
C Code.
C Code Run.
Run, Code, RUN!
        PLEASE!!!!
```

— /usr/games/fortune[1]

Due by 11:59:59pm, Tuesday, September 28th (Drop Day's eve).
This assignment is to be done individually.

Program: `postnet`

Your task is to write a program that maps postal delivery information to the POSTNET (Postal Numeric Encoding Technique) barcodes you find at the bottom of much of your mail.

The POSTNET barcode

The POSTNET barcode encodes delivery information in machine-readable form on the front of an envelope. All the details you could ever want, and more, are described in U.S.P.S. Pub. 63, but this section will give you all you need for this assignment.

The code consists of a series of full- and half-hight bars and consists of four segments:

1. An initial full-height *frame bar*,
2. The data payload,
3. A checksum, and
4. A final full-height frame bar.

Delivery information can be encoded in the payload in three possible forms, 5, 9, and 11 digits. The 5-digit form is the addressee's ZIP code, the 9-digit form is the addressee's ZIP+4 code, and the 11-digit form has the ZIP+4 plus a 2-digit *delivery point* to allow for automatic sorting.

[1]Berkeley Unix (BSD) distrubutions have traditionally included a collection of games stored in /usr/games. fortune(1) prints a randomly selected adage ranging from the humorous to the truly strange. Sadly, these days, many system administrators choose not to install /usr/games in order to save space or to encourage the doing of actual work.

The checksum is the number required to make the total sum evenly divisible by 10. That is, it is 10 minus the remainder when the sum of the delivery information is divided by 10.

Each number in the barcode is encoded as a sequence of full- and half-height bars. To make life simpler for this assignment we will use colon (":") and period (".") rather than "┃" and "╻". The encodings for each digit are given in Table 1.

Digit	**Binary**	**Bars**	**Punctuation**
0	11000	┃┃╻╻╻	::...
1	00011	╻╻╻┃┃	...::
2	00101	╻╻┃╻┃	..:.:
3	00110	╻╻┃┃╻	..::.
4	01001	╻┃╻╻┃	.:..:
5	01010	╻┃╻┃╻	.:.:.
6	01100	╻┃┃╻╻	.::..
7	10001	┃╻╻╻┃	:...:
8	10010	┃╻╻┃╻	:..:.
9	10100	┃╻┃╻╻	:.:..

Table 1: POSTNET digit encodings

Examples

To make everything clear, let's do an example of each form for mail addressed to the CSC Department at Cal Poly:

- 5-digit, ZIP code only:

  The payload consists of the ZIP code, 93407. The sum of the digits for this is 23 which means that to make it an even multiple of 10 our checkdigit will need to be 7.

Data only:	93407
Data + checkdigit:	934077
Encoded as bars:	┃╻┃╻╻╻╻┃┃╻╻┃╻╻┃┃┃╻╻╻┃╻╻╻┃┃╻╻╻┃
With frame bars:	┃┃╻┃╻╻╻╻┃┃╻╻┃╻╻┃┃┃╻╻╻┃╻╻╻┃┃╻╻╻┃┃

- 9-digit, ZIP+4 code:

  The department's ZIP+4 is 93407-0354. This time the sum of the digits is 35 yielding a checkdigit of 5:

Data only:	934070354
Data + checkdigit:	9340703545
Encoded as bars:	┃╻┃╻╻╻╻┃┃╻╻┃╻╻┃┃┃╻╻╻┃╻╻╻┃┃┃╻╻╻╻╻┃┃╻╻┃╻┃╻╻┃╻╻┃╻┃╻┃╻
With frame bars:	┃┃╻┃╻╻╻╻┃┃╻╻┃╻╻┃┃┃╻╻╻┃╻╻╻┃┃┃╻╻╻╻╻┃┃╻╻┃╻┃╻╻┃╻╻┃╻┃╻┃╻┃

- 11-digit, ZIP+4 and Delivery Point:

  The ZIP+4 is 93407-0354, and let's assume we're going to delivery point 42. This time our checkdigit is 9:

Data only:	93407035442
Data + checkdigit:	934070354429
Encoded as bars:	┃╻┃╻╻╻╻┃┃╻╻┃╻╻┃┃┃╻╻╻┃╻╻╻┃┃┃╻╻╻╻╻┃┃╻╻┃╻┃╻╻┃╻╻┃╻┃╻╻┃╻╻┃╻┃╻┃┃╻┃╻╻
With frame bars:	┃┃╻┃╻╻╻╻┃┃╻╻┃╻╻┃┃┃╻╻╻┃╻╻╻┃┃┃╻╻╻╻╻┃┃╻╻┃╻┃╻╻┃╻╻┃╻┃╻╻┃╻╻┃╻┃╻┃┃╻┃╻╻┃

Program specification

Finally, the specification of the program. Your program, `postnet` must:

- Read a series of delivery locations from stdin with at most one element per line,
- Ignore whitespace, including blank lines (that is, "934020354" and "93402 0354" are the same.),
- For valid inputs prodiuce a valid POSTNET barcode using colon (":") and period (".") for full- and half-height bars respectively in the format specified below,
- For invalid inputs print "Invalid delivery data" on a line by itself,
- continue processing delivery locations until it encounters end of file (EOF).

The output of `postnet` will be a table showing the mapping of delivery data to barcode. On the left will be the delivery data with spaces removed left-justified in a field 11 spaces wide. This will be followed by " `->` " and the barcode. This can easily be done using field precisions in your `printf()` format string like so: `printf("%-11s -> %s\n", info, barcode);`

Tricks and Tools

Remember, this assignment is fairly straightforward, but it there are a few edge cases, mostly having to do with malformed inputs, to watch out for. Pay close attention to these. Also, be careful not to impose artificial limits on the program's functionality. There is no reason, for example, to limit either line length or file size. There are a few library routines that may help with implementing `postnet` listed in Figure 1.

`getchar(3)`	read a character from stdin
`putchar(3)`	write a character to stdout
`getc(3)`	read a character from the specified stream
`printf(3)`	Write formatted output to stdout
`putc(3)`	write a character to the specified stream
`gets(3)`	never use this function. Not ever.
`feof(3)`	test to see if a given stream has encountered an end of file
`isspace()` `isdigit()` etc.	Two of many functions defined in `ctype.h` for text processing. Very useful.

Figure 1: Some potentially useful library functions

A note and a warning

Note, to indicate end of file to a program reading from stdin, type Ctrl-D at the beginning of a line.

And the warning: Remember, an 11-digit decimal number will fit in an integer on a 64-bit machine, but it will not fit on a 32-bit one. Be careful not to fall into that trap. This program needs to be portable.

Coding Standards

This assignment must compile cleanly with “`gcc -ansi -pedantic -Wall postnet.c`”
See also the pages on coding standards on the cpe 357 class web page.

What to turn in

Submit via `handin` to the `asgn1` directory of the `pn-cs357` account:

- Your well-documented source file(s).
- A README file that contains:
  - Your name.
  - Any special instructions for running your program.
  - Any other thing you want me to know while I am grading it.

  The README file should be **plain text,** i.e, **not a Word document**, and should be named “README”, all capitals with no extension.

Sample runs

Below are some sample runs of `postnet`, first interactively, then with stdin redirected from a file. I will also place executable versions in `~pn-cs357/demos` so you can run it yourself.

```
% postnet
12345
12345       -> :...::..:.:..::..:..:.:.:..:.:.:
93407
93407       -> ::.:....::..:..:::...:...::...::
hello?
Invalid delivery data
123456789
123456789   -> :...::..:.:..::..:..:.:.:..::..:...::..:.:.:...:.:.:
^D
% cat infile
12345
93402
934020354
93402 0354

    934020354 42
93402 0354 42
whatever
54321
% postnet < infile
12345       -> :...::..:.:..::..:..:.:.:..:.:.:
93402       -> ::.:....::..:..:::.....:.:..:.::
934020354   -> ::.:....::..:..:::.....:.:::.....::..:.:..:..:::....:
```

```
934020354   -> ::.:....::..:..:::.....:.:::.....::..:.:..:..:::...:
93402035442 -> ::.:....::..:..:::.....:.:::.....::..:.:..:..:.:..:..:.:.:..::
93402035442 -> ::.:....::..:..:::.....:.:::.....::..:.:..:..:.:..:..:.:.:..::
Invalid delivery data
54321       -> :.:.:..:..:..::...:.:...::.:.:.:
%
```