The critical portion of a Perl script that accomplishes this task is as follows. The only part you need to understand is the regular expression that appears inside m/ / in line 117.

```perl
if (m/\(\s*\d+\s\w+\),\s\w+\s\D*)\(\d+\)\(\s\)(\s*\d+)/) {
    print "$1$4 |";
}
```

The regular expression looks for one or more leading digits (perhaps preceded by some whitespace), then a comma separated name. It calls that part of the match $1. Then it tries to also match another digit (the student ID, which will be discarded), and then some more whitespace, followed by the PO number (named $4). If a successful match is made, the name (with leading number) and the PO are printed (with a trailing “|” to make things look nice).

### 9.4.3 Exercises

The exercises marked with ✡ have detailed solutions in Appendix G.

1. Show that the + operator (metacharacter) is not necessary. That is, show that the regular expression \+ can be replaced by an equivalent regular expression that only uses concatenation and the | and * operators.

2. Write a regular expression that matches telephone numbers in the form ###-###-####.

3. ✡ Write a regular expression that matches words that start and end in vowels (the standard English vowels aeiou). Assume that only lowercase letters are being used. Assume that the word must have at least two letters.

4. Write a regular expression that matches any line of characters that does not contain any periods, commas, colons, or semicolons.

5. Create regular expressions that match single words with the following characteristics:
   (a) Containing at least one q.
   (b) ✡ Containing a double lowercase vowel. A double lowercase vowel would be the same vowel twice, as in book.
   (c) Containing at least two double lowercase vowels. They can be the same double lowercase vowel, as in beekeeper.

6. Write a regular expression that will match only the eight phrases: “X the Y from the Z,” where X is either “separate” or “separate,” and Y and Z are both either “novices” or “experienced,” with \( Y \neq Z \).

7. Write a regular expression that matches any non-null binary string.

8. Write a regular expression that matches a valid Pascal identifier. A Pascal identifier consists of a letter, followed by 0 or more letters and/or digits.

9. Write a regular expression that matches a standard C identifier. A standard C identifier consists of a letter, followed by 0 or more letters, digits, and/or underscores.

10. A hexadecimal constant is usually written as a 0, followed by either an \( x \) or an \( X \) and then a mixture of digits and letters from the set \{a, b, c, d, e, f, A, B, C, D, E, F\} (for example, 0X3B6). Write a regular expression that matches any hexadecimal constant.

11. Write a regular expression that matches the string “\textit{\em \begin{definition}}”.

12. Write a regular expression that matches a complete sentence that is a question. The sentence should start with a capital letter and end with a question mark. Assume that the only permissible internal punctuation characters are commas.

13. Write a regular expression that matches a formal name. The name may have an optional title or honorific (such as Doctor, Dr., Miss, Mr.), a first name, an optional middle name or initial (initials will always have a period), a last name, followed by an optional period-free designation (such as Junior, III). The designation should be preceded by a comma. All parts of the name should begin with uppercase letters. You may assume that all words, except a middle initial, contain at least two letters.

14. ✡ Write a regular expression that matches a credit card number that contains an expiration date. The number may either be in the form “dadddddddddd mm/yy” or the form “daddddddd-dddd mm/yy” or the form “dadddddddddddddddd mm/yy,” where “d,” “m,” and “y” represent digits.

15. Write a regular expression that matches a letter grade. Assume that letter grades must be in the following set: \{A, A−, B+, B, B−, C+, C, C−, D+, D, F, I, W\}. You may not use the regular expression “(A|A−|B+|B−|C+|C|C−|D+|D|F|I|W)”, even if you fix the error.

16. Write a regular expression that matches any single lowercase word. For this problem, a word must contain one or more consonants and at least one vowel. (Assume that a, e, i, o, and u are the only vowels, so the word why will not be matched.)

---

17 The mi/ characters tell Perl to match the regular expression inside the /’s. The variables $1, $2, and so on match the subregular expressions inside the parentheses.