60-141 – Introduction to Programming II Winter, 2014
Assignment 3

Artificial Intelligence?

Physics teachers in high schools often think that problems given as text are more demanding than pure computations. After all, the students have to read and understand the problem first and, then solve it!

So they don't state a problem like "U = 10V, I = 5A, P = ?" but rather like "You have an electrical circuit that contains a battery with a voltage of U = 10V and a light-bulb. There's an electrical current of I = 5A through the bulb. How much power P is consumed in the bulb?"

However, half of the students just don't pay attention to the text anyway. They just extract from the text what is given: "U = 10V, I = 5A". Then they think: "Ok, which formulae do I know regarding this problem? Ah yes, "P = U * I". Therefore P = 10V * 5A = 50W. Done!"

But, this strategy may not always work well, and so, these students are usually not the top scorers. But this is usually good enough to pass the test. (Sad but true.) Today we will check if a computer can pass a high school physics test. We will concentrate on the "P – U – I" type problems. That means, the problems in which two of power, voltage and current are given and the third is to be calculated.

Your job is to write a C program (Assign3.c) that reads such a text problem and solves it according to the simple formula given as:

P = U * I,

where,

P = Power consumed or produced by a device expressed in unit Watts (W).
U = Voltage used in the system expressed in unit Volts (V).
I = Current flow through the circuit expressed in unit Amperes (A).

Further, the units of measurement must also be considered. For many devices, voltage may be expressed as millivolts (mV), kilovolts (kV), or even megavolts (MV). Similarly, for current one may modify the units as mA, kA, or MA and, for power, as mW, kW or MW. In case you are not used to these prefixes, "m" stands for milli (or 1/1000), "k" stands for kilo (1,000), while "M" stands for mega (or 1,000,000).

Input File:

You have to use input redirection technique to enter inputs from a text file.

Example: a.out < input.txt

The first line of the input file will contain the number of test cases.

Each test case will consist of one line containing exactly two data fields and some additional arbitrary words. A data field will be of the form I=xA, U=xV or P=xW, where x is a real number.

Directly before the unit (A, V or W) one of the prefixes m (milli), k (kilo) and M (Mega) may also occur. To summarize it: Data fields adhere to the following grammar:

DataField ::= Concept=RealNumber[Prefix]Unit

Concept ::= P | U | I
Prefix ::= m | k | M
Unit ::= W | V | A

Additional Assertions:
The equal sign (=) will never occur in any context other than within a data field only. There is no whitespace (tabs, blanks) inside a data field. Either P and U, P and I, or U and I will be given.

Output File:

For each test case, print three lines:

- A line saying "Problem #k" where k is the number of the test case
- A line giving the solution (voltage, power or current, depending on what was given), written without a prefix and with two decimal places as shown in the sample output
- A blank line

Contents of a sample input file:

3
If the voltage is U=200V and the current is I=4.5A, how much power is generated?
A light-bulb yields P=100W and the voltage is U=220V. Compute the current, please.
bla bla bla lightning strike I=2A bla bla bla P=2.5MW bla bla voltage?

Corresponding Output:

Problem #1
P = 900.00W
Problem #2
I = 0.45A
Problem #3
U = 1250000.00V

Assumptions:

Assume that the input file text contains valid substrings of the data fields. The remaining text can be arbitrary so long as it does not conflict with the data field character strings, or introduce ambiguities of interpretation.

REQUIREMENTS:
- Write and document a complete C program that is capable of satisfying the requirements of this assignment problem.
- UNDOCUMENTED OR IMPROPERLY DOCUMENTED code will automatically lose 50% marks.
- PLAGIARIZED work will not be graded and receive a mark of ZERO and reported according to the Senate bylaws.
- The question requires use of I/O redirection. Please review the textbook for an example on using I/O redirection from flat files.
- TO SUBMIT: No later than the submission deadline, your email must be received by: cs14101@uwindsor.ca, late submissions are not accepted and will receive a mark of ZERO.
- Email your work as an attachment, include both the source file (assign1.c) and the script file (assign1.txt) - see below how to create the script file.

To create a script file (one that logs your compilation steps and your output in a text file):
1. script assign3.txt

2. cat assign2.c
3. cat input.txt
4. cc assign3.c
5. a.out < input.txt
6. ls -l
7. exit (DO NOT FORGET THIS STEP!!)

Email both files with the mail Subject field format "**Ass #3 section 51**" (replacing 51 by your actual registered lab section number) to:

cs141@courses.cs.uwindsor.ca

NOTE: Submissions that are not received correctly by the deadline will automatically receive a ZERO mark. In the event that more than one email submission is sent, only the last one (according to the date and time stamp) will be marked.

It is your responsibility to ensure the email attachment is sent correctly (readable) and to the right mailbox by the deadline. If you omit the email subject header or fail to follow the format provided above for the attachment file, your assignment may not be graded.

Late assignment submissions are not accepted!

NOTES:

1. Your assignment must be RECEIVED by the due date and time. Late assignment submissions are NOT accepted. Keep your script file, and all your code unmodified as proof of its completion in case it is not received.

2. It is your responsibility to get an early start on the assignment, research and ask questions ahead of time from the due date.

3. You must use your uwindsor email account to submit your work. Please do not use other email accounts (hotmail, yahoo etc...) for the purpose of assignment submissions.

4. Marks will be deducted for unclear code. (improper spacing and alignment, hard to read programs and missing outputs).

5. Make sure you turn in a complete script file that clearly shows: your code, your compilation process, a listing of the directory showing your source file(s) and the a.out with the date/time stamps, and the output.

6. **PLAGIARISM:** CHEATING IS NOT TOLERATED. You must submit your own work. Students who are suspected of copying someone else's work will be reported to the department's chair and the Dean of Science and be dealt with in accordance with the University policies. You should not share your code with others. Codes that are similar to each other will BOTH be reported as potential evidence of copying. It is imperative that you write your own code.

7. Authorized/limited help on this assignment may be provided directly from your Lecture or Lab instructors and Teaching Assistants.