Problem Set #2

Due Tuesday, January 19, 2010, at the **beginning** of class. Assignments turned in more than 10 minutes after the beginning of class will be penalized.

The first four questions pertain to the following DP matrix:

$$\begin{tikzcd}
 & & A & C & D & E & F \\
 & 0 \arrow[r] \arrow[d] \arrow[dr] & -2 \arrow[r] \arrow[dr] & -4 \arrow[r] \arrow[dr] & -6 \arrow[r] \arrow[dr] & -8 \arrow[r] & -10 \\
G & -2 \arrow[d] & 7 \arrow[r] \arrow[d] \arrow[dr] & 5 \arrow[dr] & 12 \arrow[r] & 10 \arrow[r] & 8 \\
H & -4 \arrow[d] & 5 \arrow[d] & 9 \arrow[d] \arrow[dr] & 14 \arrow[r] \arrow[dr] & 12 \arrow[r] \arrow[dr] & 10 \\
I & -6 \arrow[d] & 3 \arrow[d] \arrow[dr] & 7 & 20 \arrow[r] \arrow[d] & 18 \arrow[d] \arrow[dr] & 21 \\
K & -8 & 1 & 12 & 18 \arrow[r] & 16 & 24
\end{tikzcd}$$

1. (10 points) Write down all maximal scoring alignments for the dynamic programming matrix shown above.
2. (5 points) Was this DP matrix generated by the Smith-Waterman or Needleman-Wunsch algorithm? How do you know?
3. (5 points) For this DP matrix, is the gap penalty linear or affine? Give the value(s).
4. (10 points) Draw an empty amino acid substitution matrix, and fill in as many values as you can, based on the above DP matrix.
5. (15 points) Use the BLOSUM62 matrix (found at ftp://ftp.ncbi.nih.gov/blast/matrices/BLOSUM62) and a gap penalty of -4 to find the optimal local alignment of `VKR` and `LQCTAS`. Show your work.
6. (10 points) Write a program `copy-file.py` that copies a given file. For example, if you have a file called `seq1.txt` that contains one line ("GATCCAT"), then you could create a copy of this file called `seq2.txt` as follows:

   ```
   > python copy-file.py seq1.txt seq2.txt
   > cat seq2.txt
   GATCCAT
   ```

7. (10 points) Write a program `reverse-lines.py` that reads in the contents of a file, and prints out the lines in reverse order. For example, say that your file is called `three-lines.txt` and consists of these three lines:

   ```
   This is the first line.
   This is the second line.
   This is the third line.
   ```

   Your program should do this:

   ```
   > python reverse-lines.py three-lines.txt
   This is the third line.
   This is the second line.
   This is the first line.
   ```

8. (10 points) Write a program `split-number.py` that reads a real number from the command line and prints its integer part on one line, followed by its decimal part (i.e., the digits after the decimal point) on a second line. For the decimal

part, print no more than 6 digits after the decimal, but do not print trailing zeroes.

```
> python split-number.py 1.234567
1
0.234567
> python split-number.py 1.23456711
1
0.234567
> python split-number.py 1.23
1
0.23
```

9. (15 points) Write a program `format-number.py` that takes as input two arguments: a number and a format, where the format is either `integer`, `real` or `scientific`. Print the given number in the requested format, and print an error if an invalid format string is provided.

```
> python format-number.py 3.14159 integer
3
> python format-number.py 3.14159 real
3.14159
> python format-number.py 3.14159 scientific
3.141590e+00
> python format-number.py 3.14159 foo
Invalid format: foo
```

10. (10 points) Write a program merge-lines.py that reads a file and prints the contents of the file all on one line (no white space, no newlines). If the file content is:
    GATGTACC
    ATCGGACT

    > python merge-lines.py
    GATGTACCATCGGACT

11. Challenge problems 1. Read a sequence from a file (where the sequence may be on one or many lines) and randomize (shuffle) the order of residues in the sequence. Do the same thing but make it work on a fasta format sequence (this is where there is a sequence name line that starts with '>' followed by any number of lines that represent the sequence). Do the same thing but make it work on a command-line specified range within the entire sequence (e.g. 'python shuf.py seq.fasta 17 23' would shuffle residues 7 to 23).

12. Challenge problems 2. Read a file of tab-delimited text and print the Nth and Mth fields from each line, where N and M are command-line specified. (A tab-delimited text file is one in which each line contains blocks of text separated by tab characters, a common way of storing large biological data sets). e.g. if file.txt contains (meaning gene1 is on chromosome 17 and starts at 128553 and ends at 129233):

    gene1 <tab> 17 <tab> 128553 <tab> 129233
    gene2 <tab> 17 <tab> 126003 <tab> 126512

    and your program was told to print the 1st and 3rd fields:

    >python tabdel.py file.txt 1 3
    gene1 <tab> 128553
    gene2 <tab> 126003

    (here I am representing the tab character with <tab> so that you can see it, but in the real input and output use the actual tab character)

    Do the same thing but give an error message if the field is missing from a line, e.g.

    > python tabdel.py file.txt 1 7
    Error: line 1 doesn't have 7 fields
    Error: line 2 doesn't have 7 fields