# Project 2 (Global Sequence Alignment)

Clarifications and Hints

## Prologue

Project goal: write a program to compute the optimal sequence alignment of two DNA strings

The zip file (`https://www.cs.umb.edu/~msolah/cs110_f18/project2.zip`) for the project contains

- project specification (`project2.pdf`)
- starter files (`edit_distance.py`, `alignment.py`)
- test script (`run_tests.py`)
- test data (`data/`)
- report template (`report.txt`)

This checklist will help only if you have read the writeup for the project and have a general understanding of the problems involved. So, please read the project writeup ↗ before you continue with this checklist.

## Problems

Problem 1 (*Calculating Edit Distance Using Dynamic Programming*) Write a program `edit_distance.py` that reads strings `x` and `y` from standard input and computes the edit-distance matrix `opt`. The program should output `x`, `y`, the dimensions (number of rows and columns) of `opt`, and `opt` itself.

Hints

- Read the sequences `x` and `y` from standard input, as strings
- Create an `(M + 1) x (N + 1)` edit-distance matrix `opt` with all elements initialized to `0`, where `M` and `N` are the lengths of `x` and `y` respectively
- Set the bottom row of `opt` to `2 * (N - j)` and its right column to `2 * (M - i)`, where `0 <= j <= N - 1` and `0 <= i <= M - 1`
- For example, if `x = 'HAM'` (`M = 3`) and `y = 'SPAM'` (`N = 4`), then the corresponding `opt` matrix after the above step is

```
        |  0  1  2  3  4
   x\y  |  S  P  A  M  -
-------------------------
 0  H   |  0  0  0  0  6
 1  A   |  0  0  0  0  4
 2  M   |  0  0  0  0  2
 3  -   |  8  6  4  2  0
```

## Problems

- Fill in the rest of the `opt` matrix, starting at `opt[M - 1][N - 1]` and ending at `opt[0][0]`, as follows: if `x[i]` and `y[j]` are the same, where `0 <= i <= M - 1` and `0 <= j <= N - 1`, then

  ```
  opt[i][j] = min(opt[i + 1][j + 1], opt[i + 1][j] + 2, opt[i][j + 1] + 2)
  ```

  and

  ```
  opt[i][j] = min(opt[i + 1][j + 1] + 1, opt[i + 1][j] + 2, opt[i][j + 1] + 2)
  ```

  otherwise

- The `opt` matrix for the above example after the preceding step is

  ```
          |  0  1  2  3  4
      x\y |  S  P  A  M  -
  -------------------------
   0  H   |  3  1  2  4  6
   1  A   |  4  2  0  2  4
   2  M   |  6  4  2  0  2
   3  -   |  8  6  4  2  0
  ```

- Write the following output, each starting on a new line
  - String `x`
  - String `y`
  - Dimensions of the `opt` matrix separated by a space
  - Elements of `opt`; use format string `'%3d '` for elements *not* on the last column, and `'%3d\n'` for the last-column elements

## Problems

Problem 2 (*Recovering the Alignment*) Write a program `alignment.py` that reads from standard input, the output produced by `edit_distance.py`, ie, input strings `x` and `y`, and the `opt` matrix. The program should then recover an optimal alignment, and write to standard output the edit distance between `x` and `y` and the alignment itself.

Hints

- Read from standard input the sequences `x` and `y` as strings, and the matrix `opt` as a 2D array of integers
- Write the edit distance between `x` and `y`, ie, the value of `opt[0][0]`
- Recover and output the alignment, starting at `opt[0][0]` and ending at `opt[M - 1][N - 1]`, as follows: if `opt[i][j]` equals `opt[i + 1][j] + 2`, then align `x[i]` with a gap and penalty of 2, and increment `i` by 1; if `opt[i][j]` equals `opt[i][j + 1] + 2`, then align a gap with `y[j]` and penalty of 2, and increment `j` by 1; otherwise, align `x[i]` with `y[j]` and penalty of 0/1 based on whether `x[i]` and `y[j]` match or not, and increment both `i` and `j` by 1

  Note: if one of the sequences is exhausted before the other, align a character from the other with a gap and penalty of 2
- For our running example, the optimal alignment produced by the previous step is

```
- H A M
S P A M
2 1 0 0 (edit distance = 3)
```

# Epilogue

Be sure to test your programs thoroughly using the short test data files and actual genomic data files under the `data` directory; here are the optimal edit distances of several of the supplied files:

| | |
|---|---|
| ecoli2500.txt | 118 |
| ecoli5000.txt | 160 |
| fli8.txt | 6 |
| fli9.txt | 4 |
| fli10.txt | 2 |
| ftsa1272.txt | 758 |
| gene57.txt | 8 |
| stx1230.txt | 521 |
| stx19.txt | 10 |
| stx26.txt | 17 |
| stx27.txt | 19 |

## Epilogue

Your project report (use the given template, `report.txt`) must include

- time (in hours) spent on the project
- short description of how you approached each problem, issues you encountered, and how you resolved those issues
- acknowledgement of any help you received
- other comments (what you learned from the project, whether or not you enjoyed working on it, etc.)

Before you submit your files

- make sure your programs meet the input and output specifications by running the following command on the terminal

  ```
  $ python3 run_tests.py -v [<problems>]
  ```

  where the optional argument `<problems>` lists the problems (`Problem1`, `Problem2`, etc.) you want to test, separated by spaces; all the problems are tested if no argument is given

- make sure your programs meet the style requirements by running the following command on the terminal

  ```
  $ pycodestyle <program>
  ```

- make sure your report isn't too verbose, doesn't contain lines that exceed 80 characters, and doesn't contain spelling/grammatical mistakes