SPEECH PROCESSING

HOMEWORK 2

In this homework, you will write a computer program to compute the minimum edit distance between two strings. Edit distance can be used to correct spelling errors in a word editor or to compute the recognition accuracy of a speech recognition system. Suppose a speech recognition system produces the following output as the best hypothesis for a given speech sample.

REF *Ses işleme dersini almam iyi oldu*

HYP *Ses işaretinde almam iyimser oldu*

In the above example, *REF* represents the correct string and *HYP* represents the recognized string. When the reference and hypotheses strings are aligned as shown below, we see that there are actually 3 matches, 2 substitution errors and 1 deletion error in the recognition output. The accuracy of the recognition system can be computed using these numbers.

	M	S	D	M	S	M
REF	Ses	işleme	dersini	almam	iyi	oldu
HYP	Ses	işaretinde		almam	iyimser	oldu

Edit distance is also used in computational biology to compute the similarity between two DNA sequences.

The edit distance between two strings is defined as the minimum number of edit operations required to transform one string into another. Let’s assume that the first string is named as target string and second string is named as source string. We want to convert source string to target. An edit is one of three operations: a delete (a character from the source

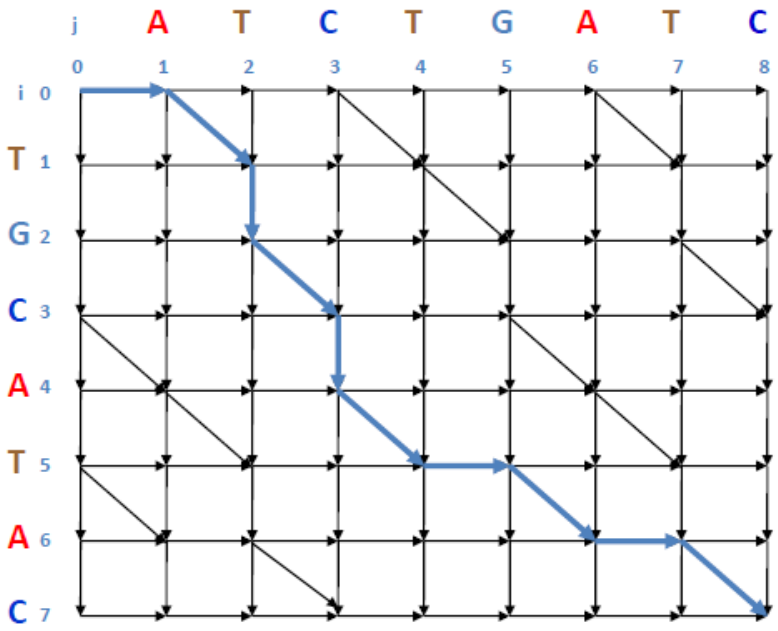
string), an insert (a character from the target string), and a substitute (a character from the source string with a character from the target string). There is a fourth operation, named as match, which does not count as an edit. Consider two input strings *"activate"* and *"caveat"*. Below you can see one possible transformation. In the example, a transformation is represented by a string consisting of the characters *M* for match, *S* for substitute, *D* for delete and *I* for insert.

```
D M D S M I M M D
a c t i v  a t e
  c  a v e a t
```

If we assume that each operation's cost is equal to 1, the edit cost between the strings is 5, since 3 delete, 1 insert and 1 substitute operations are performed to convert one string to the other. Observe that there are two other possible transformations with the same cost, *DMSDMIMMD* and *DMSSSMMD*.

You can use the following instructions to implement your code:

1. Place one of the string to the column of a matrix and the other string to the row as shown in the figure.



2. As you can see from the figure, you can reach to each node of the matrix (except the nodes in the first column and row) using three paths:
    1) From the same row but previous column
    2) From the same column but previous row
    3) From the previous row and previous column

    The first two paths stand for the delete or insert operations. The type of operation is dependent on the placement of your strings. The last path stands for a match if the character's in the strings in that row and column match, or a substitute if the characters do not match.
3. For each node, you need to keep the path to arrive that node and the cost. The cost arriving to the end node corresponds to the edit distance. The total best can be computed with back-tracing.
4. In your code, define the cost for each operation as follows:

    Delete cost = 0.7

    Insert cost = 0.7

    Substitute cost = 1.0

    Match does not have a cost

Now write a computer program to compute the minimum edit distance between two strings. Compute the edit distance between the following strings using your code. Additionally, determine the best path. In your homework, include your code with your comments.

AHMET	-	*MUHAMMET*
INTENTION	-	*EXECUTION*
KERAMET	-	*MERHAMET*