레포트 #4

알고리즘 3분반, 32170578, 김산

소스코드

E_DIST.java

```
public class E_DIST {
    int m, n;
    int cost[][];
    char edit[][];
    String str1, str2;

    E_DIST(String _str1,String _str2) {
        str1 = _str1;
        str2 = _str2;
        m = _str1.length();
        n = _str2.length();
        cost = new int [m+1][n+1];      // Minimum Cost
        edit = new char [m+1][n+1];     // Edit Order
    }

    private int minEdit(int del, int chg, int ins,int i, int j) {
        if(del < chg && del < ins) {
            edit[i][j] = 'D';
            return del;
        } else if(chg < ins && chg < del) {
            edit[i][j] = 'C';
            return chg;
        } else {
            edit[i][j] = 'I';
            return ins;
        }
    }

    public int editDistDP() {
        for (int i = 0; i <= m; i++) {
            for (int j = 0; j <= n; j++) {
                if (i == 0) {
                    cost[i][j] = j;
                    edit[i][j] = 'I';
                }
                else if(j == 0) {
                    cost[i][j] = i;
                    edit[i][j] = 'D';
                }
                else if (str1.charAt(i - 1) == str2.charAt(j - 1)) {
                    cost[i][j] = cost[i - 1][j - 1];
                    edit[i][j] = 'C';
                }

                else
```

```
                    cost[i][j] = minEdit(cost[i-1][j] + 1,
                                         cost[i-1][j-1] + 2,
                                         cost[i][j - 1] + 1,
                                         i,j);
            }
        }
        edit[0][0] = '-';
        return cost[m][n];
    }
    public char[] optOrder() {
        int i = m;
        int j = n;
        int count = 0;
        char c_tmp = edit[i][j];
        char temp[] = new char[m+n];
        while(i != 0 || j != 0) {
            if (edit[i][j] == 'I'){
                c_tmp = 'I';
                j--;
            } else if (edit[i][j] == 'D') {
                c_tmp = 'D';
                i--;
            } else {
                c_tmp = 'C';
                i--;
                j--;
            }
            temp[count++] = c_tmp;
        }

        char order[] = new char[count];
        for(int k = 0; k < count; k++) {
            order[k] = temp[count - k - 1];
        }
        return order;
    }

    public String[] orderApply(char order[]) {
        int i = 0, j = 0;
        String[] orderApply = new String[order.length];
        for (int k = 0; k < order.length; k++) {
            String tmp = "";
            switch (order[k]) {
                case 'I':
                    j++;
                    break;
                case 'D':
                    i++;
                    break;
                case 'C':
                    i++;
                    j++;
                    break;
            }
            tmp = str2.substring(0,j) + str1.substring(i,str1.length());
            orderApply[k] = tmp;
        }
        return orderApply;
```

```
    }

}
```

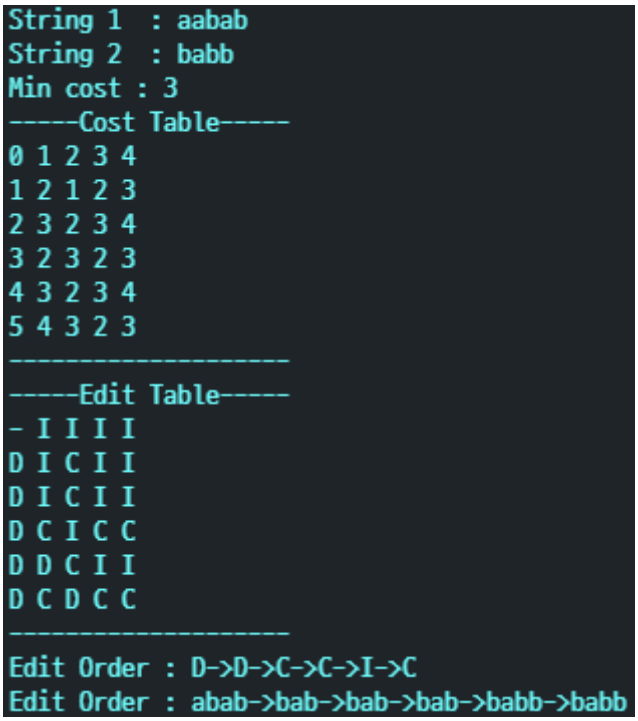
E_DIST_TEST.java(Main함수)

```
import java.util.Scanner;

public class E_DIST_TEST {
    public static void main(String[] args) {
        Scanner sc = new Scanner(System.in);
        System.out.print("String 1  : ");
        String str1 = sc.nextLine();
        System.out.print("String 2  : ");
        String str2 = sc.nextLine();
        sc.close();

        /*2개의 문자열 입력을 받아 최적 경로를 구하기 위한 객체를 생성합니다. */
        E_DIST E = new E_DIST(str1,str2);
        System.out.println("Min cost : " + E.editDistDP());
        /*비용 배열을 출력합니다.*/
        System.out.println("-----Cost Table-----");
        for (int i = 0; i < E.cost.length; i++) {
            for (int j = 0; j < E.cost[i].length; j++) {
                System.out.print(E.cost[i][j] + " ");
            }
            System.out.println();
        }
        System.out.println("--------------------");
        /*편집 배열을 출력합니다.*/
        System.out.println("-----Edit Table-----");
        for (int i = 0; i < E.edit.length; i++) {
            for (int j = 0; j < E.edit[i].length; j++) {
                System.out.print(E.edit[i][j] + " ");
            }
            System.out.println();
        }
        System.out.println("--------------------");

        /*최적의 편집 순서열을 구합니다*/
        char order[] = E.optOrder();
        System.out.print("Edit Order : ");
        for (int i = 0; i < order.length - 1; i++) {
            System.out.print(order[i] + "->");
        }
        System.out.println(order[order.length-1]);

        /*최적의 편집 순서열을 적용하여 문자열 편집 순서를 나타냅니다.*/
        String s_order[] = E.orderApply(order);
        System.out.print("Edit Order : ");
        for (int i = 0; i < s_order.length - 1; i++) {
            System.out.print(s_order[i] + "->");
        }
        System.out.println(s_order[s_order.length-1]);
    }
}
```

실행결과

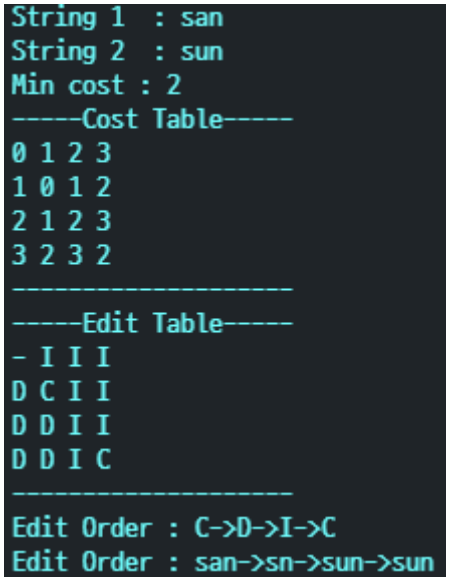
Case 1

String 1	String 2
aabab	babb



Case 2

String 1	String 2
san	sun



Case 3

String 1	String 2
sunday	saturday

```
String 1  : sunday
String 2  : saturday
Min cost : 4
-----Cost Table-----
0 1 2 3 4 5 6 7 8
1 0 1 2 3 4 5 6 7
2 1 2 3 2 3 4 5 6
3 2 3 4 3 4 5 6 7
4 3 4 5 4 5 4 5 6
5 4 3 4 5 6 5 4 5
6 5 4 5 6 7 6 5 4
--------------------
-----Edit Table-----
- I I I I I I I
D C I I I I I I
D D I I C I I I
D D I I D I I I
D D I I D I C I
D D C I I I D C I
D D D I I I D D C
--------------------
Edit Order : C->I->I->C->D->I->C->C->C
Edit Order : sunday->saunday->satunday->satunday->satuday->saturday->saturday->saturday->saturday
```

Case 4

String 1	String 2
food	money

```
String 1  : food
String 2  : money
Min cost : 7
-----Cost Table-----
0 1 2 3 4 5
1 2 3 4 5 6
2 3 2 3 4 5
3 4 3 4 5 6
4 5 4 5 6 7
--------------------
-----Edit Table-----
- I I I I I
D I I I I I
D I C I I I
D I C I I I
D I D I I I
--------------------
Edit Order : D->D->I->C->D->I->I->I
Edit Order : ood->od->mod->mod->mo->mon->mone->money
```

Case 5

String 1	String 2
voldemort	dumbledore

```
String 1  : voldemort
String 2  : dumbledore
Min cost : 11
-----Cost Table-----
0 1 2 3 4 5 6 7 8 9 10
1 2 3 4 5 6 7 8 9 10 11
2 3 4 5 6 7 8 9 8 9 10
3 4 5 6 7 6 7 8 9 10 11
4 3 4 5 6 7 8 7 8 9 10
5 4 5 6 7 8 7 8 9 10 9
6 5 6 5 6 7 8 9 10 11 10
7 6 7 6 7 8 9 10 9 10 11
8 7 8 7 8 9 10 11 10 9 10
9 8 9 8 9 10 11 12 11 10 11
--------------------
-----Edit Table-----
- I I I I I I I I I I
D I I I I I I I I I I
D I I I I I I I C I I
D I I I I C I I I I I
D C I I I I I C I I I
D D I I I I C I I I C
D D I C I I I I I I D
D D I D I I I I C I I
D D I D I I I I D C I
D D I D I I I I D D I
--------------------
Edit Order : D->D->D->C->D->I->C->I->I->I->C->C->D->I
Edit Order : oldemort->ldemort->demort->demort->dmort->dumort->dumort->dumbort->dumblort->dumbleort->dumbledort->dumbledort->dumbledort->dumbledor->dumbledore
```