**Student Number: 202135760**

**Student Name: 김현지**

**Problem 5**

* **Screenshots**

**텍스트, 스크린샷, 폰트, 디스플레이이(가) 표시된 사진

자동 생성된 설명**

* **Code**

#define \_CRT\_SECURE\_NO\_WARNINGS

#include<stdio.h>

#include<stdlib.h>

#include<string.h>

#include<ctype.h>

#define X\_MAXLEN 10001 //max 10000

#define Z\_MAXLEN 101 //max 100

int dynamic\_programming(char[], char[]);

void main() {

int num;

char x[X\_MAXLEN];

char z[Z\_MAXLEN];

scanf("%d", &num);

int\* output = (int\*)malloc(sizeof(int\*) \* num); //result

memset(output, 0, sizeof(int) \* num); //init result

int xIsLower = true; //true if x consists of lowercase, false otherwise

int zIsLower = true; //true if z consists of lowercase, false otherwise

for (int i = 0; i < num; i++) {

do {

printf("[#%d] Enter lowercase alpahbetic characters: \n", i);

scanf("%s", &x);

scanf("%s", &z);

xIsLower = true;

zIsLower = true;

for (int stringIndex = 0; stringIndex < strlen(x); stringIndex++) {

if (isupper(x[stringIndex]))//If x has an uppercase, assign false to xIsLower

xIsLower = false;

}

for (int stringIndex = 0; stringIndex < strlen(z); stringIndex++) {

if (isupper(z[stringIndex]))//If z has an uppercase, assign false to zIsLower

zIsLower = false;

}

} while (!(xIsLower && zIsLower));//If there is an uppercase in z and x, get input again

output[i] = dynamic\_programming(x, z);

}

//result print

for (int i = 0; i < num; i++) {

printf("%d\n", output[i]);

}

}

int dynamic\_programming(char x[], char z[]) {

int width = strlen(x) + 1; //Create table columns as many as x characters + add empty column at first

int height = strlen(z) + 1; //Create table columns as many as z characters + add empty column at first

int\*\* table = (int\*\*)malloc(sizeof(int\*) \* height);//Create rows in table

for (int index = 0; index < strlen(z) + 1; index++) {

table[index] = (int\*)malloc(sizeof(int) \* width);//Create columns in table

}

int i = 0;

int j = 0;

//First row initialized to 1, first column initialized to 0

for (j = 0; j < width; j++) {

table[0][j] = 1;

}

for (i = 1; i < height; i++)

table[i][0] = 0;

for (i = 1; i < height; i++) {

for (j = 1; j < width; j++) {

if (x[j - 1] == z[i - 1]) {//If they are equal, get the[Left + Left Diagonal Above] value from the table.

table[i][j] = table[i][j - 1] + table[i - 1][j - 1];

}

else//If different, get value from left side of table

table[i][j] = table[i][j - 1];

}

}

int result = table[height - 1][width - 1];

//free

for (int index = 0; index < height; index++) {

free(table[index]);

}

free(table);

return result;

}