#include <stdio.h>

#include <stdlib.h>

int myStrLen(char \*str) {

int len = 0;

while (\*str++ != '\0') {

len++;

}

return len;

}

void myStrCpy(char \*s1, char \*s2) {

while ((\*s1++ = \*s2++) != '\0');

}

void myStrCat(char \*s1, char \*s2) {

while (\*s1 != '\0') {

s1++;

}

myStrCpy(s1, s2);

}

int strReplace(char \*str, char \*pat, char \*rep) {

int i = 0, j = 0, k;

int flag = 0;

int strLen = myStrLen(str);

int patLen = myStrLen(pat);

int repLen = myStrLen(rep);

for (i = 0; i <= strLen - patLen; i++) {

k = i;

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

if (str[k++] != pat[j]) {

break;

}

}

if (j == patLen) {

flag = 1;

char \*temp = (char \*)malloc(strLen + repLen - patLen + 1);

myStrCpy(temp, str);

myStrCpy(temp + i, rep);

myStrCat(temp + i + repLen, str + i + patLen);

myStrCpy(str, temp);

free(temp);

return flag;

}

}

return flag;

}

int main() {

char STR[100], PAT[20], REP[20];

printf("Enter main string (STR): ");

fgets(STR, sizeof(STR), stdin);

printf("Enter pattern string (PAT): ");

fgets(PAT, sizeof(PAT), stdin);

printf("Enter replace string (REP): ");

fgets(REP, sizeof(REP), stdin);

STR[myStrLen(STR) - 1] = '\0'; // Remove newline at end if any

PAT[myStrLen(PAT) - 1] = '\0'; // Remove newline at end if any

REP[myStrLen(REP) - 1] = '\0'; // Remove newline at end if any

if (!strReplace(STR, PAT, REP)) {

printf("Pattern string (PAT) not found in main string (STR).\n");

} else {

printf("Modified main string (STR) is: %s\n", STR);

}

return 0;

}