LCS problem 01:

```
#include <stdio.h>
#include <string.h>
int i, j, m, n, LCS_t[50][50];
char s1[50] = "abcbdab", s2[50] = "dbcaba", b[50][50];
void lcsAlgo(){
m = strlen(s1);
n = strlen(s2);
for(i=0; i<=m; i++)
LCS_t[i][0] = 0;
for(i=0; i<=n; i++)
LCS_t[0][i] = 0;
for(i=1; i<=m; i++)
for(j=1; j<=n; j++)
   if(s1[i-1] == s2[j-1]){
      LCS_t[i][j] = LCS_t[i-1][j-1]+1;
   } else if(LCS_t[i-1][j] >= LCS_t[i][j-1]){
      LCS_t[i][j] = LCS_t[i-1][j];
   }else{
      LCS_t[i][j] = LCS_t[i][j-1];
      }
   }
}
```

```
int index = LCS_1[m][n];
char lcsAlgo[index+1];
lcsAlgo[index] = '\0';

int i=m, j=n;
while (i>0 && j>0){
    if (S1[i-1] == S2[j-1]){
        lcsAlgo[index-1] = S1[i-1];
    i--;
    j--;
    index--;
    }
    else if (LCS_1[i-1][j] > LCS_1[i][j-1])
    i--;
    else
    j--;
}
printf("S1: %s\nS2: %s\n", S1, S2);
printf("DNA Most Match: %s", lcsAlgo);
}
int main(){
    lcsAlgo();
}
```

LCS Problem 02:

```
#include <stdio.h>
#include <string.h>
int i, j, m, n, LCS_t[50][50];
char s1[50] = "catcat", s2[50] = "tgcat"; b[50][50]
void lcsAlgo(){
m = strlen(s1);
n = strlen(s2);
for (i=0; i<=m; i++)
LCS_t[i][0] =0;
for (i=0; i<=n; i++)
LCS_t[0][] =0;
for (i=1; i<=m; i++)
for (j=1; j<n; i++){
   if(s1[i-1] ==s2[j-1]){
  LCS_t[i][j] =LCS_t[i-1][j-1]+1;
}else if(LCS_t[i-1][j]>=LCS_[i][j-1]){
  LCS_t[i][j] =LCS_t[i-1][j];
}else{
  LCS_t[i][j] =LCS_t[i][j-1];
}
}
```

```
int index = LCS_t[m][n];
char lcsAlgo[index+1];
lcsAlgo[index] = '\0';
int i=m, j=n;
while(i>0 && j>0){
  if(S1[i-1] == S2[j-1]){
   lcsAlgo[index-1] = S1[i-1];
i--;
j--;
 index--;
}
else if(LCS_t[i-1][j] > LCS_t[i][j-1])
i--;
else
j--;
}
printf("S1: %s\nS2: %s\n", S1, S2);
printf("DNA Most match: %s", lcsAlgo);
}
int main(){
  lcsAlgo();
}
```