|  |  |
| --- | --- |
| Input: | **"ccc", "c"** |
| Output: | **1** |
| Expected: | **3** |

class Solution {

public:

int numDistinct(string S, string T) {

int sSize = S.size();

int tSize = T.size();

int\* combinedMatrix = new int[tSize \* sSize];

//buildup combinedMatrix;

int i,j;

for (i = 0; i < tSize; ++i) {

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

if(T[i] == S[j]) {

if (i == 0) {

if (j == 0) {

combinedMatrix[i\*sSize + j] = 1;

} else {

combinedMatrix[i\*sSize + j] = combinedMatrix[(i\*sSize + j - 1)];

}

} else {

if (j == 0) {

combinedMatrix[i\*sSize + j] = 0;

} else {

combinedMatrix[i\*sSize + j] = combinedMatrix[(i\*sSize + j - 1)] + combinedMatrix[(i - 1) \* sSize + j - 1];

}

}

} else {

if (i == 0) {

if (j == 0) {

combinedMatrix[i\*sSize + j] = 0;

} else {

combinedMatrix[i\*sSize + j] = combinedMatrix[(i\*sSize + j - 1)];

}

} else {

if (j == 0) {

combinedMatrix[i\*sSize + j] = 0;

} else {

combinedMatrix[i\*sSize + j] = combinedMatrix[(i\*sSize + j - 1)] + combinedMatrix[(i - 1) \* sSize + j - 1];

}

}

}

}

}

return combinedMatrix[tSize \* sSize - 1];

}

};

|  |  |
| --- | --- |
| Input: | **"aabb", "ab"** |
| Output: | **5** |
| Expected: | **4** |

int numDistinct(string S, string T) {

int sSize = S.size();

int tSize = T.size();

int\* combinedMatrix = new int[tSize \* sSize];

//need check parameters

if (sSize == 0) {

return 0;

}

//buildup combinedMatrix;

int i, j;

for (i = 0; i < tSize; ++i) {

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

if (T[i] == S[j]) {

if (i == 0) {

if (j == 0) {

combinedMatrix[i\*sSize + j] = 1;

}

else {

combinedMatrix[i\*sSize + j] = combinedMatrix[(i\*sSize + j - 1)] + 1;

}

}

else {

if (j == 0) {

combinedMatrix[i\*sSize + j] = 0;

}

else {

combinedMatrix[i\*sSize + j] = combinedMatrix[(i\*sSize + j - 1)] + combinedMatrix[(i - 1) \* sSize + j - 1];

}

}

}

else {

if (i == 0) {

if (j == 0) {

combinedMatrix[i\*sSize + j] = 0;

}

else {

combinedMatrix[i\*sSize + j] = combinedMatrix[(i\*sSize + j - 1)];

}

}

else {

if (j == 0) {

combinedMatrix[i\*sSize + j] = 0;

}

else {

combinedMatrix[i\*sSize + j] = combinedMatrix[(i\*sSize + j - 1)] + combinedMatrix[(i - 1) \* sSize + j - 1];

}

}

}

}

}

int iTemp = combinedMatrix[tSize \* sSize - 1];

delete[] combinedMatrix;

return iTemp;

}

int numDistinct(string S, string T) {

int sSize = S.size();

int tSize = T.size();

int\* combinedMatrix = new int[tSize \* sSize];

//need check parameters

if (sSize == 0) {

return 0;

}

//buildup combinedMatrix;

int i,j;

for (i = 0; i < tSize; ++i) {

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

if(T[i] == S[j]) {

if (i == 0) {

if (j == 0) {

combinedMatrix[i\*sSize + j] = 1;

} else {

combinedMatrix[i\*sSize + j] = combinedMatrix[(i\*sSize + j - 1)] + 1;

}

} else {

if (j == 0) {

combinedMatrix[i\*sSize + j] = 0;

} else {

combinedMatrix[i\*sSize + j] = combinedMatrix[(i\*sSize + j - 1)] + combinedMatrix[(i - 1) \* sSize + j - 1];

}

}

} else {

if (i == 0) {

if (j == 0) {

combinedMatrix[i\*sSize + j] = 0;

} else {

combinedMatrix[i\*sSize + j] = combinedMatrix[(i\*sSize + j - 1)];

}

} else {

if (j == 0) {

combinedMatrix[i\*sSize + j] = 0;

} else {

combinedMatrix[i\*sSize + j] = combinedMatrix[(i\*sSize + j - 1)];

}

}

}

}

}

return combinedMatrix[tSize \* sSize - 1];

}