```cpp
#include <iostream>
#include <stdlib.h>

using namespace std;

void printValue(int v[], int l) {
    int i;
    std::cout<<"list: {";
    for (i=0; i<l; i++){
        std::cout<<v[i] << " ";
    }
    std::cout<<"}"<<std::endl;
}

int quickSort_Compare(int A[], int q, int p) {
    int i=q;
    int pivot=A[q];
    int j;
    int temp;
    for (j=q+1;j<p;j++) { // start with the one after the pivot
        if (A[j]<pivot) {
            i=i+1;
            temp=A[j]; // switch between A[i] and A[j]
            A[j]=A[i];
            A[i]=temp;
        }
    }
    temp=A[i]; // switch the pivot to the correct location
    A[i]=A[q];
    A[q]=temp;
    return i;
}

int randomized_partition(int A[],int q, int p){

    int temp;
    int randomNumber = q+(rand() % (p-q)) ; // generate a random number between q and
p not include p
    temp = A[q];
    A[q] = A[randomNumber];
    A[randomNumber] = temp;
    int r = quickSort_Compare(A,q,p);
    return r;
}

int RandomSelect(int A[], int p, int q, int i){
    int r;
    if (p==q){
```

```
        return A[p];
    }
    r = randomized_partition(A,p,q);
    int k = r-p;
    if (k==i){
        return A[r];
    }else if (i < k){
        RandomSelect(A,p,r,i);
    }else if (i > k){
        RandomSelect(A,r+1,q,i-k-1);
    }
}

int main(){
    int t=100;       // length of A
    int A[t];
    int i=1;         // the ith order value of the given list A
    int s=0;
    for (int i=0;i<t;i++){
        A[i] = i+1;
    }

    srand((unsigned) time(0));
    for (int i=0;i<t;i++){ // shuffle the list
        int randomNumber = (rand() % t);
        int temp = A[i];
        A[i] = A[randomNumber];
        A[randomNumber] = temp;
    }
    // random select
    // cout<<"The original ";
    // printValue(A,t);
    int value = RandomSelect(A,s,t,i);
    cout<<"The value of index "<<i<<" is "<< value<<endl;


}
```

```cpp
#include <iostream>
#include <stdlib.h>
using namespace std;

void printValue(int v[], int l) {
    int i;
    std::cout<<"list: {";
    for (i=0; i<l; i++){
        std::cout<<v[i] << " ";
    }
    std::cout<<"}"<<std::endl;
}

int quickSort_Compare(int A[], int q, int p) {
    int i=q;
    int pivot=A[q];
    int j;
    int temp;
    for (j=q+1;j<p;j++) { // start with the one after the pivot
        if (A[j]<pivot) {
            i=i+1;
            temp=A[j]; // switch between A[i] and A[j]
            A[j]=A[i];
            A[i]=temp;
        }
    }
    temp=A[i]; // switch the pivot to the correct location
    A[i]=A[q];
    A[q]=temp;
    return i;
}

void quickSort_main(int A[],int q,int p) {
    if (p > q) {
        int r = quickSort_Compare(A,q,p);
        quickSort_main(A,q,r);
        quickSort_main(A,r+1,p);
    }
}

int Select(int A[], int q, int p, int i){
    int g = 5; // g per group
    int group = (p-q)/g; // # of group
    // sort each group of g
    for (int itt=0;itt<group;itt++){
        //cout<<"before and after the quick sort ...\n";
        //printValue(A,p);
        quickSort_main(A,q+itt*g,q+itt*g+5); // each g-group would be sorted
```

```
        //printValue(A,p);
    }
    //find the median of all the group
    for (int itt=1;itt<group;itt++){
        int j=(itt-1)*g+2;
        int key=A[itt*g+2]; // median
        while (j>=2 && A[j]>key){
            A[j+g]=A[j];
            j = j-g;
        }
        A[j+5]=key;
    }
    // switch median with index 0
    if (group>=1){
        int temp = A[q];
        A[q] = A[q+g*(group/2)+2]; // upper median
        A[q+g*(group/2)+2]=temp;
    }
    //cout<<"before after pivot selection... \n";
    //printValue(A,p);
    // compare
    int r = quickSort_Compare(A,q,p);
    //printValue(A,p);
    return r;
}

int Select_main(int A[], int p, int q, int i){
    int r;
    if (p==q){
        return A[p];
    }
    r = Select(A,p,q,i);
    cout<<"inside main p: "<<p<<" ,q: "<<q<<" ,r: "<<r<<"\n";
    int k = r-p;
    if (k==i){
        return A[r];
    }else if (i < k){
        Select_main(A,p,r,i);
    }else if (i > k){
        Select_main(A,r+1,q,i-k-1);
    }
}

int main(){
    int t=100; int s=0;
    int i=81;
    int A[t];
```

```
    for (int i=0;i<t;i++){
        A[i] = i+1;
    }

    srand((unsigned) time(0));
    for (int i=0;i<t;i++){
        int randomNumber = (rand() % t);
        int temp = A[i];
        A[i] = A[randomNumber];
        A[randomNumber] = temp;
    }

    //select
    //cout<<"The original ";
    //printValue(A,t);
    int value = Select_main(A,s,t,i);
    cout<<"The value of index "<<i<<" is "<< value<<endl;


}
```

```cpp
#include <iostream>
#include <cstring>

using namespace std;

void dp(char *A, char *B,int m, int n){
    int map[m+1][n+1];
    char val[m+1][n+1];

    for (int i=0;i<=m;i++){
        map[i][0] = 0;
    }
    for (int j=0;j<=n;j++){
        map[0][j] = 0;
    }
    for (int i=1;i<=m;i++){
        for (int j=1;j<=n;j++){
            if (A[i-1]==B[j-1]){
                map[i][j] = map[i-1][j-1]+1;
                val[i][j] = 'c';
            } else if (map[i-1][j]>map[i][j-1]){
                map[i][j] = map[i-1][j];
                val[i][j] = 'l';
            } else {
                map[i][j] = map[i][j-1];
                val[i][j] = 'u';
            }
        }
    }
    cout<<"the length of LCS is: "<<map[m][n]<<"\n";
}

int main() {
    char A[] = "ACGFDTAT";
    char B[] = "BCDRF";
    int m = strlen(A);
    int n = strlen(B);
    dp(A,B,m,n);
}
```