```
all: ED

ED: main.o a.o
	g++ main.o a.o -o ED -Wall -Werror -ansi -pedantic -lsfml-system -g

ED.o: main.cpp a.hpp
	g++ -c main.cpp -g -Wall -Werror -ansi -pedantic -lsfml-system -g

a.o: a.cpp a.hpp
	g++ -c a.cpp -g -Wall -Werror -ansi -pedantic -lsfml-system -g

clean:
	rm *.o ED *~ *.gch
```

```
// Angel Zheondre Calcano
// PS4

#include <SFML/System.hpp>
#include <string>
#include <iostream>
#include "a.hpp"

using namespace std;

int main( int argc, char *argv[] ) {
  int count ;
  sf::Clock clock ;
  sf::Time t ;
  string a, b ;
  cin >> a ; cin >> b ;
  //check if they are null if so cancel program.
  Edist test( a, b ) ;
  count = test.OptDistance() ;
  cout << "Edit distance " << count << endl;
  test.Alignment( 0, 0 ) ;
  t = clock.getElapsedTime() ;
  cout << "Execution time is " << t.asSeconds() << " seconds \n" ;
  cout << count << endl ;
  return 0 ;
}
```

```
// Angel Zheondre Calcano
// PS4

#include <vector>
#include <string>
#include <iostream>
#include <algorithm>
#include "a.hpp"

using namespace std;

int Edist::penalty( char temp, char tb ){
  if( temp == tb && temp == '-' ) return 0 ;
  if( temp == tb ) return 0 ;
  if( temp != tb && ((temp == '-') || (tb == '-'))) return 2 ;
  return 1 ;
}

int Edist::min( int x, int y, int z ) {

  vector< int > in; int low;

  in.push_back(x); in.push_back(y); in.push_back(z);
  low = *min_element(in.begin(), in.end());
  //cout<< low << " Lowest value found" << endl ;
  in.clear();
  return low;
}

int Edist::OptDistance(){

  int i, j, cost, hold ;

  hold = 0 ;

  for( i = maxCL - 1 ; i > -1 ; i--){
    if( i == maxCL - 1 ) cost = penalty( '-','-') ;
    else  cost =  cost + penalty( '-', b[i] ) ;
    opt[maxRL - 1][i] = cost ;
  }
  for( i = maxRL -1 ; i > -1 ; i--){
    if( i == maxRL - 1 ) cost = penalty( '-','-') ;
    else  cost =  cost + penalty( '-', b[i] ) ;
    opt[i][maxCL - 1] = cost ;
  }
  for(i = maxRL - 2 ; i > -1 ; i--)
    for( j = maxCL - 2 ; j > -1 ; j-- ) {

      hold = opt[i+1][j+1] ;
      if( a[i] != b[j] ) hold += 1 ;

      opt[i][j] = min(hold, opt[i+1][j] + 2, opt[i][j+1] + 2 ) ;
    }
  return opt[0][0] ;
}

int Edist::Alignment(int i, int j ){
  // might do this iteratively
  // need to edit code this implementation is incorrect.
  int hey = maxCL - 1 ;
  int you = maxRL - 1 ;
  if( i > maxCL - 1 || j > maxRL - 1 ) return 0 ;
  if( i < hey && j < you    ){
    if( opt[i][j] == opt[i+1][j+1] && a[i] == b[j] ){


      cout << a[i] << b[j]<< 0 << endl   ;
      i++; j++ ;
      return Alignment( i , j ) ;
    }
    if( opt[i][j] == opt[i+1][j+1] + 1 && a[i] != b[j] ){
      cout << a[i] <<  b[j] << 1 << endl ;
      i++ ; j++ ;
      return Alignment( i , j ) ;
    } ;
    if( opt[i][j] == opt[i+1][j] + 2 && a[i] != b[j] ){
      cout << a[i] << '-' << 2 << endl ;
      i++ ;
      return Alignment( i , j ) ;
    } ;
    if( opt[i][j] == opt[i][j+1] + 2 && a[i] != b[j] ){
      cout << '-' << b[j] << 2 << endl ;
      j++ ;
      return Alignment( i , j ) ;
    } ;
  }
  if( i < maxCL && j != maxRL - 1) {
    cout << b[i] << "-"<< 2<< endl ;
    i++ ;
    return Alignment(i , j ) ;
  }

  if( j < maxRL && i != maxCL ) {
    cout <<"-"<< a[j]<< 2<< endl  ;
    j++ ;
    Alignment( i , j ) ;
  }
  return 0 ;
}
```

```
// Angel Zheondre Calcano
// PS4

#ifndef _a
#define _a

#include <string>
#include <iostream>

using namespace std;

class Edist{

  string a, b ;
  int maxCL, maxRL ;
  int** opt ;

public :

  Edist( string x, string y ): a(x), b(y) {
    int i, j ;
    maxCL = a.size() + 1 ;
    maxRL = b.size() + 1 ;

    opt = new int*[maxRL] ;

    for( i = 0 ; i < maxRL ; i++ )
      opt[i] = new int[maxCL] ;

    for( i = 0 ; i < maxRL ; i++ )
      for( j = 0 ; j < maxCL ; j++ )
        opt[i][j] = -1 ;
  } ;

  ˜Edist(){
    delete opt ;
  }
  int penalty( char a, char b) ;
  int min( int x, int y, int z ) ;
  int OptDistance() ;
  int Alignment(int x, int y) ;
  // make sure to delete 2d array in deconstructor..

} ;

#endif
```