

```
1: all: ED
2:
3: ED: main.o a.o
4:     g++ main.o a.o -o ED -Wall -Werror -ansi -pedantic -lsfml-system -g
5:
6: ED.o: main.cpp a.hpp
7:     g++ -c main.cpp -g -Wall -Werror -ansi -pedantic -lsfml-system -g
8:
9: a.o: a.cpp a.hpp
10:    g++ -c a.cpp -g -Wall -Werror -ansi -pedantic -lsfml-system -g
11:
12: clean:
13:    rm *.o ED *~ *.gch
```

```
1: // Angel Zheondre Calcano
2: // PS4
3:
4: #include <SFML/System.hpp>
5: #include <string>
6: #include <iostream>
7: #include "a.hpp"
8:
9: using namespace std;
10:
11: int main( int argc, char *argv[] ) {
12:     int count ;
13:     sf::Clock clock ;
14:     sf::Time t ;
15:     string a, b ;
16:     cin >> a ; cin >> b ;
17:     //check if they are null if so cancel program.
18:     Edist test( a, b ) ;
19:     count = test.OptDistance() ;
20:     cout << "Edit distance " << count << endl;
21:     test.Alignment( 0, 0 ) ;
22:     t = clock.getElapsedTime() ;
23:     cout << "Execution time is " << t.asSeconds() << " seconds \n" ;
24:     cout << count << endl ;
25:     return 0 ;
26: }
```

```
1: // Angel Zheondre Calcano
2: // PS4
3:
4: #include <vector>
5: #include <string>
6: #include <iostream>
7: #include <algorithm>
8: #include "a.hpp"
9:
10: using namespace std;
11:
12: int Edist::penalty( char temp, char tb ){
13:     if( temp == tb && temp == '-' ) return 0 ;
14:     if( temp == tb ) return 0 ;
15:     if( temp != tb && ((temp == '-') || (tb == '-')) return 2 ;
16:     return 1 ;
17: }
18:
19: int Edist::min( int x, int y, int z ) {
20:
21:     vector< int > in; int low;
22:
23:     in.push_back(x); in.push_back(y); in.push_back(z);
24:     low = *min_element(in.begin(), in.end());
25:     //cout<< low << " Lowest value found" << endl ;
26:     in.clear();
27:     return low;
28: }
29:
30: int Edist::OptDistance(){
31:
32:     int i, j, cost, hold ;
33:
34:     hold = 0 ;
35:
36:     for( i = maxCL - 1 ; i > -1 ; i--){
37:         if( i == maxCL - 1 ) cost = penalty( '-', '-' ) ;
38:         else cost = cost + penalty( '-', b[i] ) ;
39:         opt[maxRL - 1][i] = cost ;
40:     }
41:     for( i = maxRL - 1 ; i > -1 ; i--){
42:         if( i == maxRL - 1 ) cost = penalty( '-', '-' ) ;
43:         else cost = cost + penalty( '-', b[i] ) ;
44:         opt[i][maxCL - 1] = cost ;
45:     }
46:     for(i = maxRL - 2 ; i > -1 ; i-- )
47:         for( j = maxCL - 2 ; j > -1 ; j-- ) {
48:
49:             hold = opt[i+1][j+1] ;
50:             if( a[i] != b[j] ) hold += 1 ;
51:
52:             opt[i][j] = min(hold, opt[i+1][j] + 2, opt[i][j+1] + 2 ) ;
53:         }
54:     return opt[0][0] ;
55: }
56:
57: int Edist::Alignment(int i, int j ){
58:     // might do this iteratively
59:     // need to edit code this implementation is incorrect.
60:     int hey = maxCL - 1 ;
61:     int you = maxRL - 1 ;
```

```
62: if( i > maxCL - 1 || j > maxRL - 1 ) return 0 ;
63: if( i < hey && j < you ){
64:     if( opt[i][j] == opt[i+1][j+1] && a[i] == b[j] ){
65:
66:
67:         cout << a[i] << b[j]<< 0 << endl ;
68:         i++; j++ ;
69:         return Alignment( i , j ) ;
70:     }
71:     if( opt[i][j] == opt[i+1][j+1] + 1 && a[i] != b[j] ){
72:         cout << a[i] << b[j] << 1 << endl ;
73:         i++ ; j++ ;
74:         return Alignment( i , j ) ;
75:     } ;
76:     if( opt[i][j] == opt[i+1][j] + 2 && a[i] != b[j] ){
77:         cout << a[i] << '-' << 2 << endl ;
78:         i++ ;
79:         return Alignment( i , j ) ;
80:     } ;
81:     if( opt[i][j] == opt[i][j+1] + 2 && a[i] != b[j] ){
82:         cout << '-' << b[j] << 2 << endl ;
83:         j++ ;
84:         return Alignment( i , j ) ;
85:     } ;
86: }
87: if( i < maxCL && j != maxRL - 1 ) {
88:     cout << b[i] << "-"<< 2<< endl ;
89:     i++ ;
90:     return Alignment(i , j ) ;
91: }
92:
93: if( j < maxRL && i != maxCL ) {
94:     cout <<"-"<< a[j]<< 2<< endl ;
95:     j++ ;
96:     Alignment( i , j ) ;
97: }
98: return 0 ;
99: }
```

```
1: // Angel Zheondre Calcano
2: // PS4
3:
4: #ifndef _a
5: #define _a
6:
7: #include <string>
8: #include <iostream>
9:
10: using namespace std;
11:
12: class Edist{
13:
14:     string a, b ;
15:     int maxCL, maxRL ;
16:     int** opt ;
17:
18: public :
19:
20:     Edist( string x, string y ): a(x), b(y) {
21:         int i, j ;
22:         maxCL = a.size() + 1 ;
23:         maxRL = b.size() + 1 ;
24:
25:         opt = new int*[maxRL] ;
26:
27:         for( i = 0 ; i < maxRL ; i++ )
28:             opt[i] = new int[maxCL] ;
29:
30:         for( i = 0 ; i < maxRL ; i++ )
31:             for( j = 0 ; j < maxCL ; j++ )
32:                 opt[i][j] = -1 ;
33:     } ;
34:
35:     ~Edist(){
36:         delete opt ;
37:     }
38:     int penalty( char a, char b) ;
39:     int min( int x, int y, int z ) ;
40:     int OptDistance() ;
41:     int Alignment(int x, int y) ;
42:     // make sure to delete 2d array in deconstructor..
43: } ;
44:
45: #endif
```