///In the name of ALLAH

#include<bits/stdc++.h>

using namespace std;

int main ()

{

/// Declare string

string s;

/// Assign string

s = "abcdf";

/// Printing size of string

cout << s.size() << endl; /// 5

/// Printing string

cout << s << endl; /// abcdf

/// Pushing char back to a string

s += 'b';

s += 'c';

cout << s << endl; /// abcdfbc

/// Taking input string

cin >> s;

cout << s << endl;

s = "asdfgg";

/// Checking is a string empty or not

string s1;

cout << s.empty() << endl; /// 0

cout << s1.empty() << endl; /// 1

/// Assigning an string in another string variable

s1 = s;

s.clear();

cout << s.empty() << endl; /// 1

cout << s1.empty() << endl; /// 0

/// assigning 'k' in 0-th index

s = "asdfg";

s[0] = 'k';

cout << s << endl; ///kasdfg;

s = "abc";

s1 = "def";

/// String concatenation

string tmp = s + s1;

cout << tmp << endl; /// abcdef

/// String iterator

string::iterator it;

for ( it = s.begin(); it != s.end(); it++ ) cout << \*it; /// abc

cout << endl;

/// For each loop

for ( auto c : s ) cout << c; /// abc

cout << endl;

s = "asd";

tmp = s;

/// Comparing two strings

if ( tmp == s ) cout << "Yes Match\n";

else "No Match\n";

/// String reverse and checking is a string is palindrome or not

s = "asddsa";

tmp = s;

reverse( tmp.begin(), tmp.end() );

if ( tmp == s ) cout << "Yes Palindrome" << endl;

else cout << "Not Palindrome" << endl;

/// String sorting in non-decreasing order

s = "gfds";

sort ( s.begin(), s.end() );

cout << s << endl; /// dfgs

/// String sorting in non-increasing order

sort ( s.rbegin(), s.rend() );

cout << s << endl; /// sgfd

/// Getting all unique elements of a string. Be care full, string should be sorted.

s = "aaadddsss";

int n = unique( s.begin(), s.end() ) - s.begin();

for ( int i = 0; i < n; i++ ) cout << s[i];/// ads

cout << endl;

/// Getting maximum element of string

cout << \*max\_element( s.begin(), s.end() ) << endl; /// s

/// Getting minimum element of string

cout << \*min\_element( s.begin(), s.end() ) << endl; /// a

/// When we want to take input with space

/// input : Muhammad Shahriar Alam

char c;

cin >> c;

getline( cin, s );

s = c + s;

cout << s << endl; /// Muhammad Shahriar Alam

/// If we need to sort some string on lexicographical order :

vector<string> v;

v.push\_back( "Muhammad" );

v.push\_back( "Nova" );

v.push\_back( "Maslenia Mubarrat" );

v.push\_back( "CPS Academy" );

v.push\_back( "Rashedul Alam Anik" );

v.push\_back( "Farhan sadik Sakib" );

v.push\_back( "Gazi Mohaimin Iqbal" );

sort ( v.begin(), v.end() );

for ( auto u : v ) cout << u << endl;

/\*\*

Out put :

CPS Academy

Farhan sadik Sakib

Gazi Mohaimin Iqbal

Maslenia Mubarrat

Muhammad

Nova

Rashedul Alam Anik

\*/

s = "asdf";

s.pop\_back(); /// removes last char of string

cout << s.back() << endl; /// print last char of string

v.clear();

v = { "Shahriar", "Shahriar", "Momo", "Momo", "Sharif", "Sharif" };

int Sz = unique ( v.begin(), v.end() ) - v.begin();

cout << Sz << endl; /// Number of unique strings in vector v;

for ( int i = 0; i < Sz; i++ ) cout << v[i] << endl; /// Prints all unique strings in vector v

/// Converting int to string

int a = 123;

s = to\_string (a);

cout << s << endl; /// 123

s[0] = '3';

cout << s << endl; /// 323

/// Converting string to integer

s = "123";

a = stoi ( s );

cout << a << endl; /// 123

a++;

cout << a << endl; /// 124;

/// Deleting a substring from string

s = "ShaKAKAhriar";

s.erase ( s.begin()+3, s.begin()+7 ); /// erase substring "KAKA" from string s

cout << s << endl;

/// Copying a substring of a string to a string

tmp = "Gagha Alam Gadha";

s = "Shahriar ";

copy ( tmp.begin()+6, tmp.begin()+10, back\_inserter ( s ) ); /// copying "Alam substring to string s back.

cout << s << endl; /// Shahriar Alam

/// Erasing all occurrence of a specific char from string.

s = "aaassdddaaasdd";

s.erase ( remove ( s.begin(), s.end(), 'a' ), s.end() ); /// removes all 'a' from s

cout << s << endl;

/// Checking is a string is substring of another string in O(n\*m)

s = "ashshasdakks";

if ( s.find( "asd" ) != -1 ) cout << "Substring found";

else cout << "Not found";

return 0;

}