String operations

(Run the examples below to check how they behave and try some simple variations)

Given a C++ string like

string str = "Hello there";

You can find the **SIZE** of the string

using *str.size().*
For example:

```
#include <iostream> #include  <string>
using
namespace std;

void change( int [][3], int);//prototype


int main(){  string str = "Hello
there"; cout << str.size() << endl;
return
0;  }
```

CONCATENATE (join) two strings using the + operator:

```
#include <iostream> #include  <string>  using
namespace std;

int main(){

string s1 = "This is s1";    string
s2 = "And this one is s2";

string s3 = s1+s2;

cout << s3 << endl;
```

```
return 0;
}
```

ERASE the characters starting at the initial position (counting from 0), given the number of chars to erase:

```
#include <iostream>
 #include <string>

 using namespace std;


int main(){


string st = "we will erase four characters starting at position 5";

st.erase(5, 4);

cout  << st << endl;

return 0;
}
```

This will **erase** the characters “ll e”
Keep in mind that the space is also a character.

You can also **INSERT** a string of characters into another, **right before** the given position.

```
#include <iostream>
 #include <string>

  using namespace std;

int main(){

//insert the string “OR NOT” into st1
```

```
string st1 = "we will erase some characters";

string toInsert= "OR NOT ";

st1.insert(14, toInsert);    cout
<< st1 << endl;

return 0;
}
```

As a result, the string *toInsert* will be inserted before the 'i' in the original string.

SUBSTRING returns the string at a given position in the original string and spanning a number of characters

```
  #include <iostream>
#include <string>    using
namespace std;

int main() {

string st2 = "this is another string";
cout << st2.substr(5, 7) << endl;

return 0;
}
```

Returns the string formed by 7 characters from st2 starting including the character in position 5
(as usual, starting to count from position 0)