string

- C++ standard library offers a `string` type to save most users from C-style manipulation of arrays of characters through pointers in <string>. There is support for C-style strings in <cstring>.
- The `string` type provides a variety of useful string operations, particularly concatenation using the + operator.

```
string str1 = "Hello";
string str2 = "World";
string str3;
unsigned long len;

// copy str1 into str3
str3 = str1;
cout << "str3 : " << str3 << endl;

// concatenates str1 and str2
str3 = str1 + str2;
cout << "str1 + str2 : " << str3 << endl;

// total lenghth of str3 after concatenation
len = str3.size();
cout << "str3.size() :  " << len << endl;
```

- An example function using string concatenation,

```
string compose(string name, string domain)
{
     return name + '@' + domain;
}
string myEmail = compose("russ", "lowkemedia");
cout << myEmail << endl;
```

- Note that the same example with arguments passed by reference would like like so, e.g.

```
string compose(const string& name, const string& domain)
{
     return name + '@' + domain;
}
```

- Here the references are passed as “read-only” (`const`) values so they can’t be modified by the function. This “by reference” example is much faster than the “by value” example, which, although easier to understand, has to copy values to and from the function.
- += may also be used when concatenating strings
  `s1 = s1 + ‘\n’;` is the same as `s1 += ‘\n’;`
- `string1 < string2` Lexicographical comparison (alphabetical order) will also work with strings. It will also work for `<=, >, and >=`
- Some useful `string` functions are,
- substr() and find()

```
string str= "We think in generalities, but we live in details.";
string str2 = str.substr(12,12); // "generalities"
long pos = str.find("live");     // position of "live" in str
string str3 = str.substr(pos);   // get from "live" to the end
```

- replace()

```
string str = "We think in generalities.";
str = str.replace(3, 8, "dream");     // "We dream generalities"
```

- insert()

```
string str = "The crowd";
str = str.insert(4, "maddening ");    // "The maddening crowd"
```

- erase()

```
string str("This is an example sentence.");
str.erase(10, 8);
cout << str << endl;    // "This is an sentence."
```

- to_string()

```
int number = 123;
string text = to_string(number);
cout << text << endl;        // “123”
```

- stoi()

```
string text = "456";
int number = stoi(text);
cout << number << endl;      // 456
```

- You may also query what letter is at a particular position in a string by addressing the string as you would an array.
- If you want to know the first letter of a string, then address the "zeroth" element, e.g.,

```
// access into string as you would an array.
string str = "Andy Pandie, pudding and pie";
cout << str[0] << endl;
```

- This will print `A` as A is considered at the `0` index of `str`. Similarly `str[8]` would print `d`.