COL100 Lecture 13

Note Title 10-09-2018

Review:

#include <string>

include "string.h"

Strings (C++)

member functions:

- s.at(i) — s[i]
- s.append(str)
- bool s.compare(str) — || <0 if s<str, >0 if s>str
- s.erase(index, length)
- s.find(str) — string::npos
- s.rfind(str)
- s.insert(index, str)
- s.length()
- s.size()

member function name	
replace(index , len , str)	replaces len characters starting at index with str .
substr(start , length)	
substr (start)	


```
s.replace(2, 3, "bot");
cout << s; // Hebot
```

substring

```
string s = "Hello";
string t = s.substr(2,2);
cout << t ; // ll
```

```
string s = "Hello";
string t = s.substr(2);
cout << t ;    // "llo"
```

s.substr(start) ≡ s.substr(start, s.length() - start);

```
string s = " Thomas Edison";
if (name.find("Edi") != string::npos)
{
        name.erase(name.find("Edi"), 6);
}
cout << name;  // Thomas
}
```

7

Operators

s + str → string append or string concatenation

returns a new string

eg.

```
string s = "Nik";
string t = "ola";
cout << s+t ; // Nikola
```

s + chr

s + 'a'

infix

```
s += str;
s += chr;

string s = "Nik";
s += "ola";

cout << s;      // Nikola
```

`s == str`	`s.compare(str) == 0`
`s != str`	`s.compare(str) != 0`
`s <= str`	
`s >= str`	

Strings are **mutable**

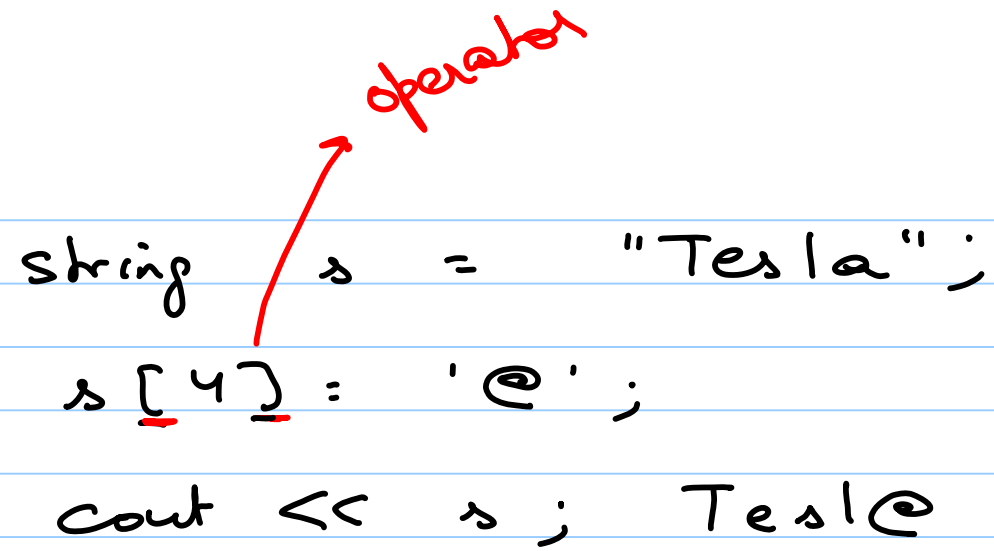
↓

their value can change over time

```
s.append("abc");

s.erase(0,2);

s[2] = '@';
```


operator
string s = "Tesla";
s[4] = '@';
cout << s; Tesl@

Methods

bool endsWith(str, suffix)	true if str ends with suffix
bool startsWith (str, prefix)	" begins "
integerToString (int) realToString (double) stringToInteger (str) stringToReal (str)	
toLowerCase (str) toUpperCase (str)	returns a lower-case / upper-case version of str

```
bool
endsWith (const string & a, const string & b)
{
    return a.length() >= b.length()
    && a.rfind(b) == a.length() - b.length();
```

```
    return a.substr(a.length() - b.length())
    == b;
}
```

```
int    i =   1123;

string  s =    integerToString ( i ); // "1123"

s. append ( " 456 ");    // "1123456"

int   j =   stringToInteger (s); // 1123456
  j =  j + 3;
 cout  <<  j ;    // 1123459
```

trim (str)

return string with surrounding whitespace removed

whitespace:

' ' , '\n' ,

'\t' ...

string s = " x x x \n Thomas Edison x "

0 1 2 3 4 5 6 7 8 9 10 11 12

```
string t = trim(s);
cout << t; // "Thomas Edison"
```

What's the output?

```
void mystery ( string a, string &b )
{
    a.erase(0, 1);      // a ← "ikola"
    b += a[0];          // b ← "teslai"
    b.insert(3, "FOO"); // b ← "tesFOOlai"
}
int main()
{
    string a = "nikola";
    string b = "tesla";
    mystery(a, b);      // a ← "nikola", b ← "tesFOOlai"
    cout << a << " " << b << endl;
    return 0;
}
```

nikola tesFOOlai

void nameDiamond (string name)

prints a pattern as follows:

if ~~name~~ = nameDiamond ("TESLA")

	0 :	1

len-1 :	spaces?	subsh
→ 0 :	1	1
→ ~~2~~ 1 :	2	2
i	i+1	i+1
len-~~2~~ 2 :	len-1	len-1

```
void nameDiamond (string s)
{
    int len = s.length();
    for (int i = 0; i < len; i++)
    {
        cout << s.substr(0, i+1) << endl;
    }
    for (int i = 0; i < len-1; i++)
    {
        for (int j; j < i+1; j++)
        {
            cout << " ";
        }
        cout << s.substr(i+1) << endl;
    }
}
```

top half

Another method to print bottom half of the diamond

```
for (int i = 0; i < len-1; i++)
{
    s.replace(i, 1, " ");
    cout << s << endl;

}
```

C-strings vs. C++ strings

"hello" : C-string

string("hello") : C++ string

"hello" + "123" ✗

string("hello") + string("123") ✓

string("hello") + "123" not recommended for this course

Next time

Vectors / Arrays