

Strings in C++

Learning Objectives

- An Array Type for Strings
- Standard Class string
 - String processing

Introduction

- Two string types:
- String class
 - Uses templates
- C-strings
 - Array with base type char
 - End of string marked with null, "\0"
 - "Older" method inherited from C

Standard Class string

- Defined in library:
`#include <string>`
`using namespace std;`
- String variables and expressions
 - Treated much like simple types
- Can assign, compare, add:
`string s1, s2, s3;`
`s3 = s1 + s2; //Concatenation`
`s3 = "Hello Mom!" //Assignment`
 - Note c-string "Hello Mom!" automatically converted to string type!

Display 9.4

Program Using the Class string

Display 9.4 Program Using the Class string

```
1  //Demonstrates the standard class string.
2  #include <iostream>
3  #include <string>
4  using namespace std;

5  int main( )
6  {
7      string phrase;
8      string adjective("fried"), noun("ants");
9      string wish = "Bon appetite!";

10     phrase = "I love " + adjective + " " + noun + "!";
11     cout << phrase << endl
12          << wish << endl;

13     return 0;
14 }
```

Initialized to the empty string.

Two equivalent ways of initializing a string variable

SAMPLE DIALOGUE

I love fried ants!
Bon appetite!

I/O with Class string

- Just like other types!
- `string s1, s2;`
`cin >> s1;`
`cin >> s2;`
- Results:
User types in:
May the hair on your toes grow long and curly!
- Extraction still ignores whitespace:
s1 receives value "May"
s2 receives value "the"

getline() with Class string

- For complete lines:
string line;
cout << "Enter a line of input: ";
getline(cin, line);
cout << line << "END OF OUTPUT";
- Dialogue produced:
Enter a line of input: Do be do to you!
Do be do to you!END OF INPUT
 - Similar to c-string's usage of getline()

Other getline() Versions

- Can specify "delimiter" character:
string line;
cout << "Enter input: ";
getline(cin, line, "?");
 - Receives input until "?" encountered
- getline() actually returns reference
 - string s1, s2;
getline(cin, s1) >> s2;
 - Results in: (cin) >> s2;

Class string Processing

- Same operations available as c-strings
- And more!
 - Over 100 members of standard string class
- Some member functions:
 - .length()
 - Returns length of string variable
 - .at(i)
 - Returns reference to char at position i

Display 9.7 Member Functions of the Standard Class string (1 of 2)

Display 9.7 **Member Functions of the Standard Class string**

EXAMPLE	REMARKS
Constructors	
<code>string str;</code>	Default constructor; creates empty string object <code>str</code> .
<code>string str("string");</code>	Creates a string object with data "string".
<code>string str(aString);</code>	Creates a string object <code>str</code> that is a copy of <code>aString</code> . <code>aString</code> is an object of the class <code>string</code> .
Element access	
<code>str[i]</code>	Returns read/write reference to character in <code>str</code> at index <code>i</code> .
<code>str.at(i)</code>	Returns read/write reference to character in <code>str</code> at index <code>i</code> .
<code>str.substr(position, length)</code>	Returns the substring of the calling object starting at <code>position</code> and having <code>length</code> characters.
Assignment/Modifiers	
<code>str1 = str2;</code>	Allocates space and initializes it to <code>str2</code> 's data, releases memory allocated for <code>str1</code> , and sets <code>str1</code> 's size to that of <code>str2</code> .
<code>str1 += str2;</code>	Character data of <code>str2</code> is concatenated to the end of <code>str1</code> ; the size is set appropriately.
<code>str.empty()</code>	Returns <code>true</code> if <code>str</code> is an empty string; returns <code>false</code> otherwise.

(continued)

Display 9.7 Member Functions of the Standard Class string (2 of 2)

Display 9.7 **Member Functions of the Standard Class string**

EXAMPLE	REMARKS
<code>str1 + str2</code>	Returns a string that has <code>str2</code> 's data concatenated to the end of <code>str1</code> 's data. The size is set appropriately.
<code>str.insert(pos, str2)</code>	Inserts <code>str2</code> into <code>str</code> beginning at position <code>pos</code> .
<code>str.remove(pos, length)</code>	Removes substring of size <code>length</code> , starting at position <code>pos</code> .
Comparisons	
<code>str1 == str2</code> <code>str1 != str2</code>	Compare for equality or inequality; returns a Boolean value.
<code>str1 < str2</code> <code>str1 > str2</code>	Four comparisons. All are lexicographical comparisons.
<code>str1 <= str2</code> <code>str1 >= str2</code>	
<code>str.find(str1)</code>	Returns index of the first occurrence of <code>str1</code> in <code>str</code> .
<code>str.find(str1, pos)</code>	Returns index of the first occurrence of string <code>str1</code> in <code>str</code> ; the search starts at position <code>pos</code> .
<code>str.find_first_of(str1, pos)</code>	Returns the index of the first instance in <code>str</code> of any character in <code>str1</code> , starting the search at position <code>pos</code> .
<code>str.find_first_not_of(str1, pos)</code>	Returns the index of the first instance in <code>str</code> of any character <i>not</i> in <code>str1</code> , starting search at position <code>pos</code> .

C-string and string Object Conversions

- Automatic type conversions
 - From c-string to string object:
`char aCString[] = "My C-string";`
`string stringVar;`
`stringVar = aCString;`
 - Perfectly legal and appropriate!
 - `aCString = stringVar;`
 - ILLEGAL!
 - Cannot auto-convert to c-string
 - Must use explicit conversion:
`strcpy(aCString, stringVar.c_str());`

Converting between `string` and numbers

- In C++11 it is simply a matter of calling **`stof`**, **`stod`**, **`stoi`**, or **`stol`** to convert a string to a float, double, int, or long, respectively.

```
int i;  
double d;  
string s;  
i = stoi("35"); // Converts the string "35" to an integer 35  
d = stod("2.5"); // Converts the string "2.5" to the double 2.5
```

Converting between numbers and string objects

- In C++11 use **to_string** to convert a numeric type to a string

```
string s;  
s = to_string(d*2);    // Converts the double 5.0 to a  
                        // string "5.0000"
```