Chapter 10:

Characters, C-Strings, and More About the string Class

Kafi Rahman
Assistant Professor@CS
Truman State University

Not going to be used in the exam

- 10.2, 10.3, 10.4, 10.6

10.5

- C-String/Numeric Conversion Functions

C-String/Numeric Conversion Functions

- Requires <cstdlib> header file

FUNCTION	PARAMETER	ACTION
atoi	C-string	converts C-string to an int value, returns the value
atol	C-string	converts C-string to a long value, returns the value
atof	C-string	converts C-string to a double value, returns the value

string to Number Conversion

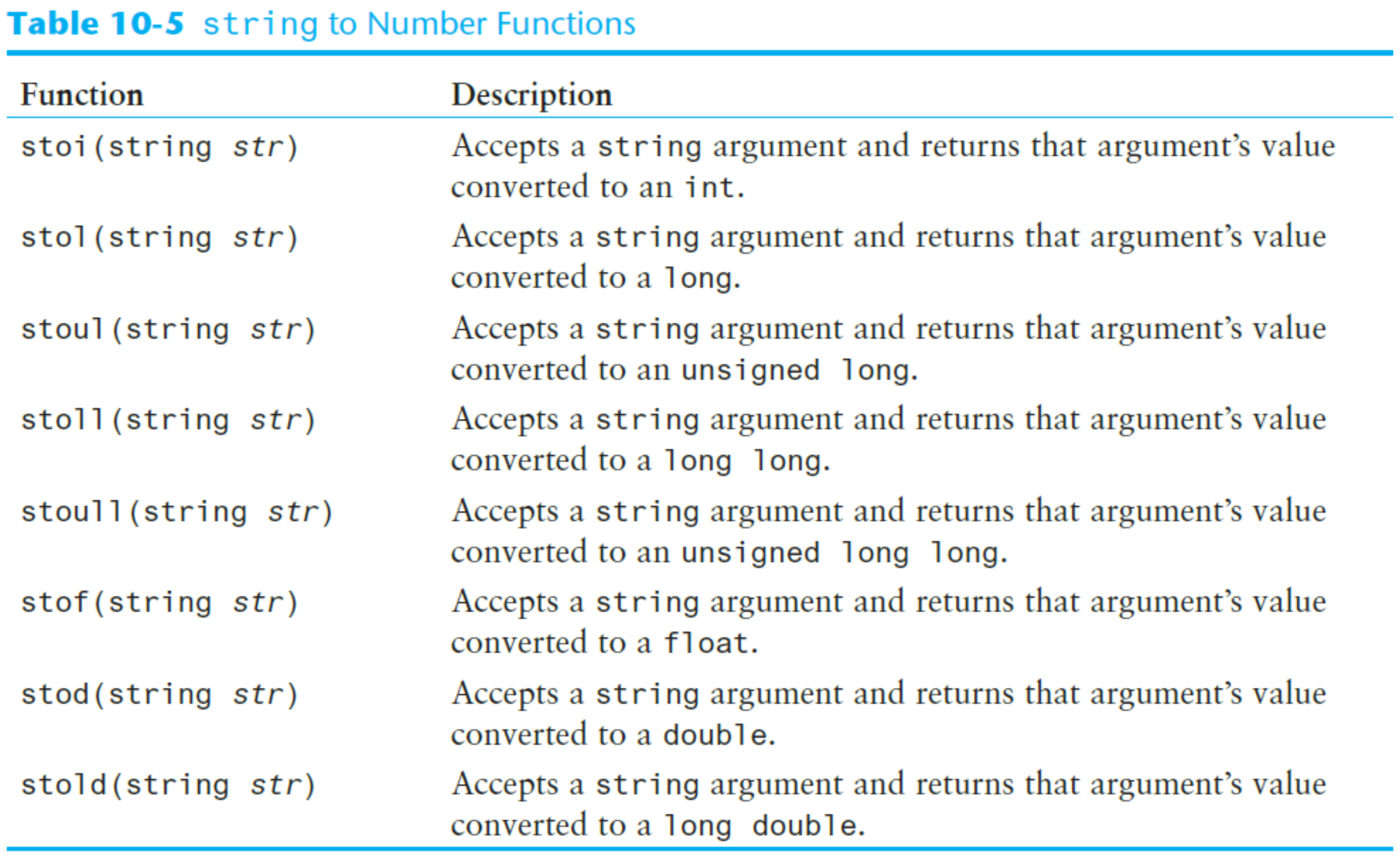

Table 10-5 `string` to Number Functions

Function	Description
`stoi(string` *`str`*`)`	Accepts a `string` argument and returns that argument's value converted to an `int`.
`stol(string` *`str`*`)`	Accepts a `string` argument and returns that argument's value converted to a `long`.
`stoul(string` *`str`*`)`	Accepts a `string` argument and returns that argument's value converted to an `unsigned long`.
`stoll(string` *`str`*`)`	Accepts a `string` argument and returns that argument's value converted to a `long long`.
`stoull(string` *`str`*`)`	Accepts a `string` argument and returns that argument's value converted to an `unsigned long long`.
`stof(string` *`str`*`)`	Accepts a `string` argument and returns that argument's value converted to a `float`.
`stod(string` *`str`*`)`	Accepts a `string` argument and returns that argument's value converted to a `double`.
`stold(string` *`str`*`)`	Accepts a `string` argument and returns that argument's value converted to a `long double`.

C-String/Numeric Conversion Functions

```
int iNum;
long lNum;
double dNum;

iNum = stoi("1234"); // puts 1234 in iNum
lNum = stol("5678"); // puts 5678 in lNum
dNum = stof("35.7"); // puts 35.7 in dNum


cout << "\n"
        << dNum << " " << iNum;
```

The to_string Function

Table 10-6 Overloaded Versions of the `to_string` Function

Function	Description
`to_string(int value);`	Accepts an `int` argument and returns that argument converted to a `string` object.
`to_string(long value);`	Accepts a `long` argument and returns that argument converted to a `string` object.
`to_string(long long value);`	Accepts a `long long` argument and returns that argument converted to a `string` object.
`to_string(unsigned value);`	Accepts an `unsigned` argument and returns that argument converted to a `string` object.
`to_string(unsigned long value);`	Accepts an `unsigned long` argument and returns that argument converted to a `string` object.
`to_string(unsigned long long value);`	Accepts an `unsigned long long` argument and returns that argument converted to a `string` object.
`to_string(float value);`	Accepts a `float` argument and returns that argument converted to a `string` object.
`to_string(double value);`	Accepts a `double` argument and returns that argument converted to a `string` object.
`to_string(long double value);`	Accepts a `long double` argument and returns that argument converted to a `string` object.

C-String/Numeric Conversion Functions

```
int iNum = 100;
long lNum = 55000;
double dNum = 5595.950425;

string textInt = std::to_string(iNum);
string textDouble = std::to_string(dNum);

cout << "\nConverted int value: " << textInt;
cout << "\nConverted double value: " << textDouble;
```

10.7

More About the C++ string Class

The C++ string Class

- Special data type supports working with strings
  - `#include <string>`
- Can define string variables in programs:

```
string firstName, lastName;
```

- Can receive values with assignment operator:

```
firstName = "George";
lastName = "Washington";
```

- Can be displayed via cout

```
cout << firstName << " " << lastName;
```

Input into a string Object

- Use cin >> to read an item into a string:

```
string firstName;
cout << "Enter your first name: ";

cin >> firstName;

cout << "\nYour name is: " << firstName;
```

Input into a string Object

- Use getline function to put a line of input, possibly including spaces, into a string:

```
string address;

cout << "Enter your address: ";
getline(cin, address);

cout << "Your address is: "<< address;
// 100 Normal Ave E
```

Initializing C++ strings

Definition	Meaning
string name;	defines an empty string object
string myname("Chris");	defines a string and initializes it
string yourname(myname);	defines a string and initializes it
string aname(myname, 3);	defines a string and initializes it with first 3 characters of myname
string verb(myname,3,2);	defines a string and initializes it with 2 characters from myname starting at position 3
string noname('A', 5);	defines string and initializes it to 5 'A's

string Operators

OPERATOR	MEANING
=	assigns string on right to string object on left
+=	appends string on right to end of contents on left
+	concatenates two strings
[]	references character in string using array notation
>, >=, <, <=, ==, !=	relational operators for string comparison. Return true or false

string Comparison: relational operator

- Can use relational operators directly to compare string objects:

```
string strx = "George", stry = "Georgia";

if (strx == stry)
     cout << strx << " is the same as " << stry;

 if (strx < stry)
     cout << strx << " is less than " << stry;

 if (strx > stry)
     cout << strx << " is greater than " << stry;
```

string Operators: addition

```
string wordx, phrase;
string wordy = " Dog";

cin >> wordx; // user enters "Hot Tomato"
              // wordx has "Hot"
phrase = wordx + wordy;
// phrase is now, "Hot Dog"

phrase += " on a bun";
// phrase is now, "Hot Dog on a bun"

for (unsigned i = 0; i < 16; i++)
    cout << phrase[i];
// output: "Hot Dog on a bun"
```

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
H	o	t		D	o	g		o	n		a		b	u	n	\0

string Member Functions

- Are behind many overloaded operators
- Categories:
  - assignment: assign, copy, data
  - modification: append, clear, erase, insert, replace, swap
  - space management: capacity, empty, length, resize, size
  - substrings: find, front, back, at, substr
  - comparison: compare
- See Table 10-8 for a list of functions.

string Member Functions: assign, append, insert

```
string wordx, wordy, phrase;
cin >> wordx; // wordx is "Hot"

wordy.assign(" Dog");
phrase.append(wordx);
phrase.append(wordy); // phrase has "Hot Dog"

phrase.append(" with mustard relish", 13);
// phrase is now, "Hot Dog with mustard"

phrase.insert(8, "on a bun ");
cout << phrase << endl;
// phrase is now, "Hot Dog on a bun with mustard"
```

string Member Function: find

```
string word = "Ottawa is the most beautiful city in the world!";

// find the position of a character in the string
cout << "\nFirst o is found at: "
     << word.find('o') << " index";

cout << "\nLast o is found at: "
      << word.find_last_of('o') << " index";

// find the position of a string in the string
unsigned pos = word.find("city"); // position of "city" in word
cout << "\nPosition of city is: " << pos;
```

string Member Function: substr

```
string word = "Ottawa is the most beautiful city in the world!";


// create a substring: slicing
string subStrOne = word.substr(19, 9); // "beautiful"
string subStrTwo = word.substr(29);    // get from "city" to the end

cout << "\n" << subStrOne  // will display: beautiful
     << "\n" << subStrTwo; // will display: city in the world
```

string Member Function: length

```
string word = "Welcome to the Foundation of Computer Science";

cout << "\nThe length of the string is: "
     << word.length() << endl;

// looping through all the elements of the string
for (unsigned index = 0; index < word.length(); index++)
{
    cout << word[index];
}

// output: Welcome to the Foundation of Computer Science
```

string Operators: accumulator

```
string word = "Hello, today is a beautiful day!";

string upperWord = ""; // empty, accumulator variable

// navigate through all the elements of the word string
for (unsigned i = 0; i < word.length(); i++)
    // is this character lower case
    if (islower(word[i]) == true)
    { // change it to uppercase and add it
        upperWord += toupper(word[i]);
    }
    else
    { // add it without making any changes
        upperWord += word[i];
    }

cout << "\nAll uppercase word: " << upperWord;
// Output: HELLO, TODAY IS A BEAUTIFUL DAY!
```

Character Testing

- Requires cctype header file

FUNCTION	MEANING
isalpha	true if arg. is a letter, false otherwise
isalnum	true if arg. is a letter or digit, false otherwise
isdigit	true if arg. is a digit 0-9, false otherwise
islower	true if arg. is lowercase letter, false otherwise
isprint	true if arg. is a printable character, false otherwise
ispunct	true if arg. is a punctuation character, false otherwise
isupper	true if arg. is an uppercase letter, false otherwise
isspace	true if arg. is a whitespace character, false otherwise