CS31 Discussion 1E

Jie(Jay) Wang
Week5 Oct.27
Outline

• Project 3
• Debugging
• Array
Project 3

• Read the spec and FAQ carefully.

• Test your code on SEASnet Linux server using the command
  • `curl -s -L http://cs.ucla.edu/classes/fall16/cs31/Utilities/p3tester | bash`

• Go through two notes carefully.
  • A technique for processing strings
  • A note about characters and integers.

• Incremental development

• No assumption about the `pollData` parameter. It can be anything!

• DDL: **Monday, October 31**
About spec

• Poll data string format
• Return value
• Pseudocode
# String processing

```c++
#include <string>
```

<table>
<thead>
<tr>
<th>Operation</th>
<th>What it does</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>string s = &quot;hello&quot;; string s = &quot;!!!&quot;;</td>
<td>Declare strings s and s2</td>
<td></td>
</tr>
<tr>
<td>s.length() or s.size()</td>
<td>Return the length of s</td>
<td>cout &lt;&lt; s.size(); // prints 5</td>
</tr>
<tr>
<td>s[i] or s.at[i]</td>
<td>Return i-th character. (i should be integer between 0 and size-1 (inclusive))</td>
<td>cout &lt;&lt; s[1]; // prints ‘e’</td>
</tr>
<tr>
<td>s + s2</td>
<td>Concatenate two strings</td>
<td>cout &lt;&lt; s + s2; // prints “hello!!!”</td>
</tr>
</tbody>
</table>

# String processing

```c++
#include <cctype>
```

<table>
<thead>
<tr>
<th>Operation</th>
<th>What it does</th>
</tr>
</thead>
<tbody>
<tr>
<td>char c;</td>
<td>Declare a character c</td>
</tr>
<tr>
<td>isspace(c)</td>
<td>True if c is a whitespace character</td>
</tr>
<tr>
<td>isalpha(c)</td>
<td>True if c is a letter</td>
</tr>
<tr>
<td>isdigit(c)</td>
<td>True if c is a digit</td>
</tr>
<tr>
<td>islower(c)</td>
<td>True is c is a lowercase letter</td>
</tr>
<tr>
<td>isupper(c)</td>
<td>True if c is a uppercase letter</td>
</tr>
</tbody>
</table>
String processing

• In order to process characters in a string,
• E.g., string str = “123AFb32#@sd”;

```cpp
for (int i = 0; i < str.size(); i++) {
    char ch = str[i]; // do something to ch
}
```

for loop

```cpp
int i = 0;
while (i < str.size()) {
    char ch = str[i]; // do something to ch
    i++;
}
```

while loop
String processing

• **Question:** count the number of digits and letters in the string str.

<table>
<thead>
<tr>
<th>str</th>
<th>#digit</th>
<th>#letter</th>
</tr>
</thead>
<tbody>
<tr>
<td>“ABC12@cd”</td>
<td>2</td>
<td>5</td>
</tr>
<tr>
<td>“sd#12#12”</td>
<td>4</td>
<td>2</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Operation</th>
<th>What it does</th>
</tr>
</thead>
<tbody>
<tr>
<td>char c;</td>
<td>Declare a character c</td>
</tr>
<tr>
<td>isspace(c)</td>
<td>True if c is a whitespace character</td>
</tr>
<tr>
<td>isalpha(c)</td>
<td>True if c is a letter</td>
</tr>
<tr>
<td>isdigit(c)</td>
<td>True if c is a digit</td>
</tr>
<tr>
<td>islower(c)</td>
<td>True is c is a lowercase letter</td>
</tr>
<tr>
<td>isupper(c)</td>
<td>True if c is a uppercase letter</td>
</tr>
</tbody>
</table>
String processing

• **Question**: given a string, filter out all non-letter characters, and print out the new string which is concatenated by all the letters left.

• E.g., `string str = “123AFb32#@sd”;` → “Afbsd”

• `string concatLetter(string str);`

```
#include <string>

<table>
<thead>
<tr>
<th>Operation</th>
<th>What it does</th>
<th>Example</th>
</tr>
</thead>
<tbody>
<tr>
<td>s + s2</td>
<td>Concatenate two strings</td>
<td><code>cout &lt;&lt; s + s2; // prints “hello!!!”</code></td>
</tr>
</tbody>
</table>
```

```
#include <ctype>

<table>
<thead>
<tr>
<th>Operation</th>
<th>What it does</th>
</tr>
</thead>
<tbody>
<tr>
<td>isalpha(c)</td>
<td>True if c is a letter</td>
</tr>
<tr>
<td>isdigit(c)</td>
<td>True if c is a digit</td>
</tr>
</tbody>
</table>
```
String processing

- **Question:** You are writing a program to filter out the illegal date records in the database, and return the number of legal records in December.

- The legal date string:
  - year (4 digits) month (3 letters, all UPPERCASE) day (1/2 digits).

- The month is guaranteed to be all uppercase letters.

- Only care about number of characters!

- `int filterCount(string str);`

<table>
<thead>
<tr>
<th>str</th>
<th>Y/N</th>
</tr>
</thead>
<tbody>
<tr>
<td>1993DEC3</td>
<td>Y</td>
</tr>
<tr>
<td>2004DEC52</td>
<td>Y</td>
</tr>
<tr>
<td>12MAR3</td>
<td>N</td>
</tr>
<tr>
<td>2012AU15</td>
<td>N</td>
</tr>
<tr>
<td>2016OCT2</td>
<td>N</td>
</tr>
</tbody>
</table>

2
Characters and Integers

<table>
<thead>
<tr>
<th>char</th>
<th>int</th>
</tr>
</thead>
<tbody>
<tr>
<td>‘0’</td>
<td>48</td>
</tr>
<tr>
<td>‘1’</td>
<td>49</td>
</tr>
<tr>
<td>‘2’</td>
<td>50</td>
</tr>
<tr>
<td>‘3’</td>
<td>51</td>
</tr>
<tr>
<td>‘4’</td>
<td>52</td>
</tr>
<tr>
<td>‘5’</td>
<td>53</td>
</tr>
<tr>
<td>‘6’</td>
<td>54</td>
</tr>
<tr>
<td>‘7’</td>
<td>55</td>
</tr>
<tr>
<td>‘8’</td>
<td>56</td>
</tr>
<tr>
<td>‘9’</td>
<td>57</td>
</tr>
</tbody>
</table>

‘0’ is not mapped to 0! However, the integer code for chars ‘0’ through ‘9’ are contiguous.

ASCII code
## Characters and Integers

<table>
<thead>
<tr>
<th>char</th>
<th>int</th>
</tr>
</thead>
<tbody>
<tr>
<td>'0'</td>
<td>48</td>
</tr>
<tr>
<td>'1'</td>
<td>49</td>
</tr>
<tr>
<td>'2'</td>
<td>50</td>
</tr>
<tr>
<td>'3'</td>
<td>51</td>
</tr>
<tr>
<td>'4'</td>
<td>52</td>
</tr>
<tr>
<td>'5'</td>
<td>53</td>
</tr>
<tr>
<td>'6'</td>
<td>54</td>
</tr>
<tr>
<td>'7'</td>
<td>55</td>
</tr>
<tr>
<td>'8'</td>
<td>56</td>
</tr>
<tr>
<td>'9'</td>
<td>57</td>
</tr>
</tbody>
</table>

```c
char ch = '0';
ch++;  // ch is '1'
int a = ch - '0';  // a is 1
ch += 7;  // ch is '8'
a = ch - '0';  // a is 8
```
Characters and Integers

• **Question:** Given a string `str` which contains several digits (0-9), how do you derive the integer value that it represents?

• `int cast(string str)`

<table>
<thead>
<tr>
<th>str</th>
<th>return value</th>
</tr>
</thead>
<tbody>
<tr>
<td>“123”</td>
<td>123</td>
</tr>
<tr>
<td>“45”</td>
<td>45</td>
</tr>
</tbody>
</table>

• **Hint:**
  • ‘4’ -> `int a = ‘4’ – ‘0’; // a is 4`
  • ‘5’ -> `int a = ‘5’ – ‘0’; // a is 5`
  • `45 = 4*10 + 5`
Debugging

• Demo on VC++ and XCode
Array

• Declare an array
  - `<type> <name>[size]`
    ```
    int a[4];
    ```
  - `a[i]` is an i-th variable in the array `a`.
  - `size` should must be a positive integer constant.
    ```
    int a[4];
    const int N = 10;
    int a[N];
    ```

• You can treat each element of the array as a variable.
  ```
  x[3] = 5;
  x[1]++;
  cout << x[i] << endl;
  ```
Initialization of an Array

```c
int a[5] = {1, 2, 3, 5, 7};
int a[] = {1, 2, 3, 5, 7};

• You cannot set the size to what’s less than the number of elements in the initialization statement.
• However, it is okay to set the size to what’s more than the number of elements in the initialization statement.

int a[3] = {1, 2, 3, 5, 7};  // wrong
int a[10] = {1, 2, 3, 5, 7};  // okay
```
Common mistakes

• No `size()` function is defined for arrays.

```cpp
int a[10];
for (int i = 0; i < a.size(); i++) {
  ...
}
```

```cpp
const int SIZE = 10;
int a[SIZE];
for (int i = 0; i < SIZE; i++) {
  ...
}
```
Common mistakes

- Out-of-boundary access not allowed!

```c
int a[10];
a[15] = 5;  // error
a[-10] = 4; // error
a[9] = 10;  // okay
```
Arrays in a Function

```c
void fibo10(int fib[]);
```

Note that the size of `fib` is not specified, you can explicitly pass the size in the function.

```c
void fibo10(int fib[], int n);
```
Quick tests

• **Question:** Will the code compile? If so, what’s the output?

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

int main () {
    int flimsySize = 3;
    int i[flimsySize];
    cout << i[1] << endl;
}
```

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

int main () {
    const int FLIMSY_SIZE = 3;
    int i[FLIMSY_SIZE];
    cout << i[1] << endl;
}
```
Quick tests

• **Question:** Will the code compile? If so, what’s the output?

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

int main () {
    int i[3];
    i = {104, 101, 121};
    cout << i[0] << endl;
    cout << i[1] << endl;
    cout << i[2] << endl;
}
```

The initialization syntax is special at declaration.
• We will cover more details about array next week.
Credit to 2 previous CS31 TAs

• This slide is finished with reference from:
  • Andrew Forney
  • http://web.cs.ucla.edu/~forns/
  • Brian Choi
  • http://netlab.cs.ucla.edu/~schoi/cs31/
Thanks!