3.12 String comparisons

String comparison: Equality

Two strings are commonly compared for equality. Equal strings have the same number of characters, and each corresponding character is identical.

PARTICIPATION ACTIVITY 3.12.1: Equal strings.

Which strings are equal?

1) "Apple", "Apple"
   - (•) Equal
   - ( ) Unequal

   Correct
   Same length, same corresponding characters.

2) "Apple", "Apples"
   - ( ) Equal
   - (•) Unequal

   Correct
   Different lengths.

3) "Apple pie!!", "Apple pie!!"
   - (•) Equal
   - ( ) Unequal

   Correct
   Spaces and special symbols are compared just like other characters.

4) "Apple", "apple"
   - ( ) Equal
   - (•) Unequal

   Correct
   'A' and 'a' differ. Case matters.

Feedback?

A programmer can compare two strings using the equality operators == and !=.

Figure 3.12.1: String equality example: Censoring.

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

int main() {
   string userWord;

   cout << "Enter a word: ";
   cin  >> userWord;

   if (userWord == "Voldemort") {
      cout << "He who must not be named";
   }
   else {
      cout << userWord;
   }
   cout << endl;

   return 0;
}
```

```
Enter a word: Sally
Sally

...

Enter a word: Voldemort
He who must not be named

...

Enter a word: voldemort
voldemort
```

Feedback?

PARTICIPATION ACTIVITY 3.12.2: Comparing strings for equality.

To what does each expression evaluate? Assume str1 is "Apples" and str2 is "apples".

1) str1 == "Apples"

- (•) True
- ( ) False

Correct

Same length, same corresponding characters.

2) str1 == str2

- ( ) True
- (•) False

Correct

'A' and 'a' differ.

3) str2 != "oranges"

- (•) True
- ( ) False

Correct

apples does not equal oranges, so != evaluates to true.

Feedback?

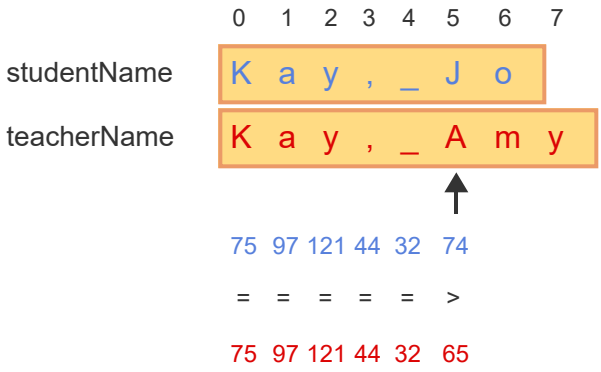
String comparison: Relational

Strings are sometimes compared relationally (less than, greater than), as when sorting words alphabetically. A comparison begins at index 0 and compares each character until the

evaluation results in false, or the end of a string is reached. 'A' is 65, 'B' is 66, etc., while 'a' is 97, 'b' is 98, etc. So "Apples" is less than "apples" because 65 is less than 97.

PARTICIPATION ACTIVITY 3.12.3: String comparison.

1 2 2x speed



'J' is greater than 'A', so studentName is greater than teacherName.

Feedback?

PARTICIPATION ACTIVITY 3.12.4: Case matters in string comparisons.

Indicate the result of comparing the first string with the second string.

1) "Apples", "Oranges"
   - (•) less than
   - ( ) equal
   - ( ) greater than

Correct
'A' is less than 'O'

2) "merry", "Merry"
   - ( ) less than
   - ( ) equal
   - (•) greater than

Correct
The coded value for 'm' is greater than the coded value for 'M'

3) "banana", "bananarama"
   - (•) less than

Correct

- equal
- greater than

If existing characters are equal, the shorter string is less than.

Feedback?

A programmer compares strings relationally using the relational operators <, <=, >, and >=.

A common error is to forget that case matters in a string comparison. A programmer can compare strings while ignoring case by first converting both strings to lowercase before comparing (discussed elsewhere).

PARTICIPATION ACTIVITY 3.12.5: Relational string comparison.

1) Write an expression that evaluates to true if myName is greater than yourName.

```
if (myName  yourName) {
    ...
}
```

Check Show answer

Answer

>

The standard relational operators work with strings.

2) Write an expression that evaluates to true if authorName1 is less than or equal to authorName2.

```
if (
    
) {
    ...
}
```

Check Show answer

Answer

authorName1 <= authorName2

A comparison begins at index 0 and compares each character until the evaluation results in false, or the end of a string is reached.

Feedback?

CHALLENGE ACTIVITY 3.12.1: String comparison: Detect word.

Write an if-else statement that prints "Goodbye" if userString is "Quit", else prints "Hello". End with newline.

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

int main() {
   string userString;

   cin >> userString;

   /* Your solution goes here  */
   if (userString == "Quit") {
      cout << "Goodbye";
   }
   else {
      cout << "Hello";
   }
   cout << endl;

   return 0;
}
```

Run ✓ All tests passed

✓ Testing for 'Quit'

Your output `Goodbye`

✓ Testing for 'Go'

Your output `Hello`

Feedback?

CHALLENGE ACTIVITY 3.12.2: Print two strings in alphabetical order.

Print the two strings in alphabetical order. Assume the strings are lowercase. End with newline. Sample output:

```
capes rabbits
```

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

int main() {
   string firstString;
```

```
7    string secondString;
8
9    cin >> firstString;
10   cin >> secondString;
11
12   /* Your solution goes here  */
13   if (firstString < secondString) {
14      cout << firstString << " " <<secondString ;
15   }
16   else {
17      cout << secondString << " " <<firstString ;
18   }
19   cout << endl;
20
```

Run ✓ All tests passed

✓ Testing input rabbits capes

Your output `capes rabbits`

✓ Testing input capes rabbits

Your output `capes rabbits`

✓ Testing input clearly clear

Your output `clear clearly`

✓ Testing input wand wand

Your output `wand wand`

Feedback?