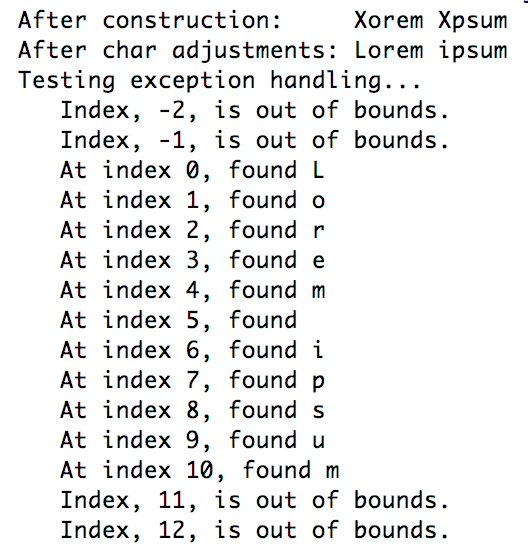
CSC 122 001 Computer Science II  
Julius Ranoa

Chapter 16 Programming Challenge 1 String Bound Exceptions.

Write a class *BCheckString* that is derived from the STL *string* class. This new class will have two member functions:

1. A *BCheckString(string s)* constructor that receives a string object passed by value and passes it on to the base class constructor.
2. A *char operator[](int k)* function that throws a *BoundsException* object if *k* is negative or is greater than or equal to the length of the string. If *k* is within the bounds of the string, this function will return the character at position *k* in the string.

Screenshot of runtime.



Files included: (1) main.cpp, (2) BCheckString.h, (3) BCheckString.cpp

**main.cpp**

#include **<iostream>**#include **"BCheckString.h"  
  
int** main() {  
 BCheckString b(**"Xorem Xpsum"**);  
  
 std::cout << **"After construction: "** << b << **"\n"**;  
  
 *// This still works!* b[0] = **'L'**;  
 b[6] = **'i'**;  
  
 std::cout << **"After char adjustments: "** << b << **"\n"**;  
  
 std::cout << **"Testing exception handling... \n"**;  
  
 *// I don't know why just b.length() without casting  
 // doesn't work.* **for** (**int** i = -2; i < (**int**)(b.length()) + 2; i++) {  
 **static char** t;  
 **try** {  
 t = b[i];  
 std::cout << **" At index "** << i << **", found "** << t << **"\n"**;  
 } **catch**(BCheckString::BoundsException e) {  
 std::cout << **" Index, "** << e.getErrorIndex() << **", is out of bounds. \n"**;  
 }  
 }  
  
 **return** 0;  
}

**BCheckString.h**

#ifndef **CH11\_PR1\_STRING\_BOUND\_EXCEPTIONS\_BCHECKSTRING\_H**#define **CH11\_PR1\_STRING\_BOUND\_EXCEPTIONS\_BCHECKSTRING\_H**#include **<string>  
  
class** BCheckString : **public** std::string {  
  
**private**:  
 *// This is a pointer to the first character,  
 // since after overloading my own [] operator,  
 // the string original [] operator gets  
 // overwritten and is lost.* **char** \* firstChar;  
 *// Note that this only works after construction.  
 // I haven't overwritten the = operators.***public**:  
 *// Exception Class* **class** BoundsException {  
 **private**:  
 **int** attemptedIndex;  
 **public**:  
 BoundsException(**int** idx) {  
 attemptedIndex = idx;  
 }  
 **int** getErrorIndex() **const** {  
 **return** attemptedIndex;  
 };  
 };  
  
 BCheckString();  
 BCheckString(std::string);  
 ~BCheckString() {  
 firstChar = **nullptr**;  
 }  
 **char**& **operator**[](**int**);  
};  
  
  
#endif *//CH11\_PR1\_STRING\_BOUND\_EXCEPTIONS\_BCHECKSTRING\_H*

**BCheckString.cpp**

#include **"BCheckString.h"**#include **<iostream>**BCheckString::BCheckString() : std::string(**""**) { }  
BCheckString::BCheckString(std::string s) : std::string(s) {  
 firstChar = **const\_cast**<**char** \*>( **this**->data() );  
}  
  
**char**& BCheckString::**operator**[](**int** k) {  
 **if** (k < 0 || k >= **this**->size()) {  
 **throw** BoundsException(k);  
 } **else** {  
 **return** \*(firstChar + k);  
 }  
}