EECE.3220: Data Structures

Key Questions

Functions; Strings (Lectures 3 & 4)

QUESTIONS

1. Describe the basics of using functions.
2. Describe the differences between pass by value, pass by address, and pass by reference for function arguments.
3. Explain how comparison operators (==, !=, <, >, <=, >=) can be used to compare strings.
4. Explain how string concatenation works in C++.
5. Explain the operation of the `substr()` and `find()` functions.
6. Explain how to access individual characters within a string.

EXAMPLE: Show the output of the following short program.

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

double f1(int v1, int v2);
void f2(int *ptr1, int *ptr2);
void f3(int &ref1, int &ref2);

int main() {
   int foo = 10;
   int bar = 57;
   double baz;

   baz = f1(foo, bar);
   cout << "After f1(), foo = " << foo << ", bar = "
      << bar << ", baz = " << baz << "\n";

   f2(&foo, &bar);
   cout << "After f2(), foo = " << foo << ", bar = " << bar << "\n";

   f3(foo, bar);
   cout << "After f3(), foo = " << foo << ", bar = "<< bar << "\n";
   return 0;
}

double f1(int v1, int v2) {
   return (v1 + v2) / 2.0;
}

void f2(int *ptr1, int *ptr2) {
   while (*ptr1 > 5) {
      *ptr2 -= 3;
      (*ptr1)--;
   }
}

void f3(int &ref1, int &ref2) {
   if (ref1 == 5 && ref2 >= 45) {
      ref1++;
      ref2--;
   }
   else if (ref1 == 5) {
      ref1--;
      ref2++;
   }
   else {
      ref1 = ref2 - 10;
      ref2 = ref1 + 10;
   }
}
```

1. List the output for each of the following code snippets from the same program.

```
int main()
{
   string s1( "happy" );
   string s2( " birthday" );
   string s3;
   // test overloaded equality and relational operators
   cout << "s1 is \"" << s1 << "\"; s2 is \"" << s2
      << "\"; s3 is \"" << s3 << '\"'
      << "\n\nThe results of comparing s2 and s1:"
      << "\ns2 == s1 yields " << ( s2 == s1 ? "true" : "false" )
      << "\ns2 != s1 yields " << ( s2 != s1 ? "true" : "false" )
      << "\ns2 >  s1 yields " << ( s2 > s1 ? "true" : "false" )
      << "\ns2 <  s1 yields " << ( s2 < s1 ? "true" : "false" )
      << "\ns2 >= s1 yields " << ( s2 >= s1 ? "true" : "false" )
      << "\ns2 <= s1 yields " << ( s2 <= s1 ? "true" : "false" );
```

OUTPUT:

```
// test string member function empty

   cout << "\n\nTesting s3.empty():" << endl;
   if ( s3.empty() )
   {
      cout << "s3 is empty; assigning s1 to s3;" << endl;
      s3 = s1; // assign s1 to s3
      cout << "s3 is \"" << s3 << "\"";
   } // end if
   // test overloaded string concatenation operator
   cout << "\n\ns1 += s2 yields s1 = ";
   s1 += s2; // test overloaded concatenation
   cout << s1;
   // test concatenation operator with C-style string
   cout << "\n\ns1 += \" to you\" yields" << endl;
   s1 += " to you";
   cout << "s1 = " << s1 << "\n\n";
```

OUTPUT:

```
// test string member function substr

   cout << "The substring of s1 starting at location 0 for\n"
      << "14 characters, s1.substr(0, 14), is:\n"
      << s1.substr( 0, 14 ) << "\n\n";
   // test substr "to-end-of-string" option
   cout << "The substring of s1 starting at\n"
      << "location 15, s1.substr(15), is:\n"
      << s1.substr( 15 ) << endl;
   // test using subscript operator to create lvalue
   s1[ 0 ] = 'H';
   s1[ 6 ] = 'B';
   cout << "\ns1 after s1[0] = 'H' and s1[6] = 'B' is: "
      << s1 << "\n\n";
   // test subscript out of range with string member function "at"
   cout << "Attempt to assign 'd' to s1.at( 30 ) yields:" << endl;
   s1.at( 30 ) = 'd'; // ERROR: subscript out of range
   return 0;
} // end main
```

OUTPUT: