Queens College, CUNY, Department of Computer Science
Object-Oriented Programming in C++
CSCI 211/611
Summer 2018
Instructor: Dr. Sateesh Mane

due date Tuesday, July 3, 2018, 11.59 pm

Homework: Introduction

- **Experience with other classes has demonstrated that in many cases the source of difficulty is not the mathematics or the programming.**
- **The source of difficulty is the English (understanding the text).**
- **If you do not understand the words in the lectures or homework, THEN ASK.**
- **If you do not understand the concepts in the lectures or homework, THEN ASK.**
- **Send me an email, explain what you do not understand.**
- **Do not just keep quiet and then produce nonsense in exams.**

- **Consult your lab instructor for assistance.**
- **You may also contact me directly, but I cannot promise a prompt response.**
- Please submit your inquiry via email, as a file attachment, to `Sateesh.Mane@qc.cuny.edu`.
- Please submit one zip archive with all your files in it.
  1. The zip archive should have either of the names (CS211 or CS611):

     `StudentId_first_last_CS211_hw_intro.zip`
     `StudentId_first_last_CS611_hw_intro.zip`
  2. The archive should contain one "text file" named "hw_intro.[txt/docx/pdf]" and one cpp file per question named "Q1.cpp" and "Q2.cpp" etc.
  3. Note that not all questions may require a cpp file.

General information

- You should include the following header files and namespace, to run the programs below.

  ```
  #include <iostream>
  #include <iomanip>
  #include <string>
  #include <cmath>

  using namespace std;
  ```

- You may require additional header files for individual questions.
- If you require additional header files to do your work, feel free to include them.
- **Include the list of all header files you use, in your solution for each question.**
- The questions below mainly use strings.
- The questions below do not require complicated mathematical calculations.
- If for any reason you require help with mathematical calculations, **ask the lab instructor or the lecturer.**

Q1 Concatenate strings

- **This question is to write a function to concatenate an input array of strings.**
- Suppose the input is an array of four strings
  `str[0] = "first"`,
  `str[1] = "Second"`,
  `str[2] = "THIRD"`,
  `str[3] = "fourth"`.
- The output should be the string `"firstSecondTHIRDfourth"`.
- The array length is n.
- **Write three functions as follows.**
  1. The first function signature is

     ```
     void concat1(int n, string str[], string &s_out);  // output is in s_out
     ```
  2. The second function signature is

     ```
     string concat2(int n, string str[]);
     ```
  3. The third function signature returns a reference. **Explain why this is unsafe.**

     ```
     string& concat3(int n, string str[]);
     ```
- Test your functions with the following main program.

```
// header files and function declarations concat1, etc

int main()
{
  int n = 4;
  string str[] = { "first", "Second", "THIRD", "fourth" };

  string s1;
  concat1(n, str, s1);
  cout << "s1 = " << s1 << endl;

  string s2 = concat2(n, str);
  cout << "s2 = " << s2 << endl;

  string s3 = concat3(n, str);    // EXPLAIN WHY THIS IS UNSAFE
  cout << "s3 = " << s3 << endl;
  return 0;
}
```

Q2 Swap

- **The following main program calls functions `swap1` and `swap2` to swap two integers.**

```
void swap1(int a, int b)
{
  int c = a;
  a = b;
  b = c;
}

void swap2(int &a, int &b)
{
  int c = a;
  a = b;
  b = c;
}

int main()
{
  int a = 3, b = 4, c = 5, d = 6;
  cout << "original:  "  << a << "   " << b  << "   " << c  << "   " << d << endl;

  swap1(a,b);
  cout << "swap1:  "  << a << "   " << b << endl;

  swap2(c,d);
  cout << "swap2:  "  << c << "   " << d << endl;
  return 0;
}
```

- **Explain what the function `swap1(...)` does.**
- **Explain what the function `swap2(...)` does.**
- **Explain what the main program will print.**
- **Explain if the function `swap3(...)` below will work correctly to swap the values of `a[0]` and `a[1]`.** (You may assume the array a has length ≥ 2, so do not worry about "array out of bounds" errors.)

```
void swap3(int a[])
{
  int c = a[0];
  a[0] = a[1];
  a[1] = c;
}
```

Q3 Russian peasant multiplication

- **Russian peasant multiplication** is an algorithm to multiply two (positive) integers.
- It is actually an old algorithm. There is evidence it was known by the ancient Egyptians.
- It is simplest to explain with an example. Suppose we wish to multiply 89×21.
  1. Let $a = 89$ and $b = 21$. Form a table of three columns of numbers as follows.

a	b	
89	21	21
44	42	
22	84	
11	168	168
5	336	336
2	672	
1	1344	1344
		1869 (sum)

  2. At each step, **<u>if a is odd</u>**, we copy the value of b into the third column.
  3. Then we divide a by 2 (integer division) and multiply b by 2.
  4. We stop when the value of a reaches 0.
  5. **The value of $a \times b$ is the sum of the numbers in the third column.**
- Hence the algorithm breaks down the multiplication of two (possibly large) numbers into a set of additions and integer multiplications and divisions by 2.
  1. Integer multiplication and division by 2 are easy operations in binary.
  2. Integer multiplication by 2 is a left shift of the binary digits of a number.
  3. Integer division by 2 is a right shift of the binary digits of a number (and loss of the "least significant bit").
  4. Addition is also a simpler operation than multiplication, in general.
- **Implement a function for Russian peasant multiplication of two positive integers.**

```
int RPM(int a, int b);
```

  1. Declare and initialize a temporary variable `int sum = 0`.
  2. Begin a loop.
     (a) If a is odd then increment `sum = sum + b`.
     (b) Perform integer division `a = a/2` and integer multiplication `b = b*2`.
     (c) Repeat the loop. Stop when $a = 0$.
  3. Return the value of `sum`.
- Set a and b to the first and last four digits of your student id.
- **If `id = 23054611`, then $a = 2305$ and $b = 4611$. Use your function to multiply $a \times b$.**

Q4 Word match

- **This function will be employed in questions in exams/projects later in this course.**
- **Write a function `word_match` to return `true` for a match, `false` otherwise.**

  ```
  bool word_match(string s1, string s2);
  ```

- The definition of a "match" is as follows.
  1. You may assume the strings `s1` and `s2` contain only one word each.
  2. However, they may contain leading and/or trailing white space (blanks).
  3. The match is **case insensitive.**
- The following pairs are all positive matches and should all return `true`.
  1. `"Alice", "Alice"`
  2. `"Alice", "ALICE"`
  3. `"Alice", "alice"`
  4. `" alice ", "AlicE "`
- The following pairs are all negative matches and should all return `false`.
  1. `"Alice", "allice"`
  2. `" alice ", " alicee "`
  3. `"Alice", "Bob"`
- Here is one possible way to implement the function body.
  1. Use the class `istringstream`.

     ```
     istringstream is1(s1);
     is1 >> s1;
     istringstream is2(s2);
     is2 >> s2;
     ```

  2. Technically, we have not learned about C++ classes yet.
  3. However, note that `string` itself is a C++ class.
  4. For now, *we use classes* and later we shall learn how to *write our own C++ classes.*
  5. The above code removes the leading and trailing whitespace from `s1` and `s2`.
  6. Next, force all the characters in `s1` and `s2` to uppercase.
  7. This is accomplished by using the function `toupper`.
  8. **Look up online how to use `toupper`, else ask your lab instructor for assistance.**
  9. Finally, compare `s1` and `s2` and return.

     ```
     return (s1 == s2);
     ```

- Here is a simple main program to illustrate the use of your function.
- You need to include `<sstream>` as well, if you employ `istringstream`.

```
#include <iostream>
#include <iomanip>
#include <sstream>
#include <string>

using namespace std;

// your code for "word_match"

int main()
{
  string s1, s2;
  bool b;

  s1 = ...
  s2 = ...
  b = word_match(s1, s2);
  if (b == true)
    cout << "true:   [" << s1 << "]    [" << s2 << "]" << endl;
  else
    cout << "false:  [" << s1 << "]    [" << s2 << "]" << endl;

  return 0;
}
```

Q5 Function declarations

- You are given two functions, which call each other.
- The function names are `print_even` and `print_odd`.
- The function bodies and a main program are given below.
- **Write the function declarations so that the code compiles and runs correctly.**

```
//  include relevant header files and all function declarations

int main()
{
  for (int i = 10; i < 20; ++i) {
    print_even(i);
  }
  return 0;
}

void print_even(int n)
{
  if (n%2 == 0)
    cout << "n is even: " << n << endl;
  else
    print_odd(n);
}

void print_odd(int n)
{
  if (n%2 != 0)
    cout << "n is odd: " << n << endl;
  else
    print_even(n);
}
```