COMP161 - Lab 10

Spring 2019

Abstract

For these problems you should simply cram each loop into a single main procedures. No libraries. No tests. Just a one file program containing a main.

1. Warm-up: Counting the interval $[0, n)$
   (a) Write a *for* loop that counts out the interval $[0, n)$ while and computing and finally printing the sum of all the numbers in that interval.
   (b) Write a *do-while* loop that counts and computes the sum of the interval $[0, n)$. Will this loop work exactly the same as your for loop for every value of n?
   (c) Write a *while* loop that counts backwards through the interval. Have it print the numbers, separated by spaces, as it counts.

2. In plain English, describe what this loop does. (Hint: Step through it for some small value of n and see exactly what it does for that n. Then describe that in more general terms. *Do not translate the code verbatim to English.*). Check yourself by copying and running the code.

```
for(int i{0}; i < n; i++ ){
  std::cout << n-1-i;
  if( i % 5 == 4 ){
    std::cout << '\n';
  }
  else{
    std::cout << ' ';
  }
}
if( n % 5 != 0 ){
  std::cout << '\n';
}
```

3. Write a validation loop that is suitable for getting a double from the interval $(0, 1)$.

4. Write a loop that counts down through the first n multiples of 3 and computes their product. When it's done, print that product.

5. Write a loop that prints out every other string in a vector of strings. Print one string per line. Your loop should continue to work if you change the vector's size, i.e. don't hard-code 10 into the loop if the test vector contains 10 strings.

6. Write a loop that works with a vector of integers and prints out the sum of adjacent, non-overlapping pairs. Assume the vector contains an even number of integers. For example, if the vector contains $\{1, 2, 3, 4, 5, 6\}$ then it should print 3, then 7, and finally 11 by adding 1 and 2, then 3 and 4, and finally 5 and 6. (Hint: You need to count through the pairs, not the individual vector elements).