**Introduction to Computer Science (MAC 101)**

**Professor: Andi Toce**

**Homework 4**

**(Due Monday, November 7, via email)**

Modify the program below so that it uses a *non-recursive* solution. You will end up having to write more code. (Hint: To make the job easier, write two functions: *get\_all\_divisors* and *get\_lowest\_divisor*. The **main** function should call *get\_all\_divisors*, which in turn has a loop: *get\_all\_divisors* calls *get\_lowest\_divisor* repeatedly, each time replacing n with n/i, where i is the divisor that was found. If n itself is returned, then the number is prime, and the loop should stop.

|  |  |
| --- | --- |
| PrimeFactorizationRecursive.cpp | Output |
| #include <iostream>  #include <cmath>  using namespace std;  void get\_divisors(int n);  int main() {  int n = 0;  cout << "Enter a number and press ENTER: ";  cin >> n;  get\_divisors(n);  cout << endl;  return 0;  }// end main  // Get divisors function  // This function prints all the divisors of n,  // by finding the lowest divisor, i, and then  // rerunning itself on n/i, the remaining quotient.  void get\_divisors(int n) {  int i;  double sqrt\_of\_n = sqrt(n);  for (i = 2; i <= sqrt\_of\_n; i++)  if (n % i == 0) { // If i divides n evenly,  cout<<i<<",";  get\_divisors(n / i);  return;  }  // If no divisor is found, then n is prime;  // Print n and make no further calls.  cout << n;  }// end get\_divisors() | Enter a number and press ENTER: 140  2,2,5,7 |

**Note:** The output should look exactly as shown the example above

**How to submit**

Save the file as LastName.FirstName.HW4.cpp

**E-mail** me the file by the due date. Late submissions will **NOT** be accepted

**Grading**

Correctness – 50%

Efficiency – 30%

Code style and appearance – 20%