PDS Lab - Section 9

Assignment 4

February 3rd, 2020

Fine Naming: If you roll no. is 18CS30012, the file name would be 18CS30012_a4p1.c, 18CS30012_a4p2.c and 18CS30012_a4p3.c.

1. Goldbach's conjecture states that any even integer $n \geq 6$ can be written as the sum of two (odd) primes. For example, 34 has exactly four such decompositions: $34 = 3 + 31 = 5 + 29 = 11 + 23 = 17 + 17$. Here, $3 + 31$ is not treated different from $31 + 3$.

   (a) Write a function **int isprime (int n)** which returns 1 if n is prime, 0 otherwise.

   (b) Write a function **int gold_decomposition (int n, int x)** where x and n are positive numbers, $x < n$, which returns 1 if x and n-x are both prime, 0 otherwise.

   (c) Write a main() to find an integer n less than or equal to a bound B such that n has the maximum number of decompositions into sums of two primes (among all positive candidates $\leq$ B). The value of B needs to be input from the user.

   Sample Output: (only see for format, not for correctness!!)

   ```
   Enter the bound B: 200
   34 has the largest number of gold decomposions: 4
   ```

2. One of the methods to find a cube root of a number N is to find the root of the function $f(x) = x^3 - N$. One can use Newton's method to approximate the root of $f(x)$. Start with an initial guess x_0. The first approximation is given as

   $$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}$$

   where $f'(x) = 3x^2$ denotes the derivative of $f(x)$. Similarly, k^{th} approximation is generated as:

   $$x_k = x_{k-1} - \frac{f(x_{k-1})}{f'(x_{k-1})}$$

   You can stop at the k^{th} iteration if $|x_k - x_{k-1}| < \epsilon$. x_k will be the cube root for N, approximated using Newton's method.

(a) Write a function **float newtonRaphson (int n)** that takes as input an integer n and uses Newton's method to find the cube root of the integer. Use $x_0 = 0.5$ and $\epsilon = 0.001$

(b) Write a main() that

i. reads a positive integer $num \geq 10$ from the user

ii. for each integer i starting from 1 up to num, it calls the function newtonRapson(i) and prints i as well as the approximated cube root.

Sample Output: See for format

```
Enter a number >=10: 12
Cube root of 1: 1.0
Cube root of 2: ....
Cube root of 3: ...
.....
Cube root of 12:
```

3. Any prime (other than 2) must be of the form $4k + 1$ or $4k + 3$. For a positive integer n, let $s_1(n)$ denote the number of primes of the form $4k + 1$ less than or equal to n, and let $s_3(n)$ denote the number of primes of the form $4k+3$ less than or equal to n. You need to find out the smallest value of n for which $s_1(n) > s_3(n)$. You may proceed as follows.

(a) Write a function that accepts a positive integer as input and returns the decision whether the input integer is prime.

(b) Inside the main() function, keep on checking odd integers $3, 5, 7, 9, \ldots$ for primality until the condition $s_1(n) > s_3(n)$ is satisfied.

(c) Print the value of n, $s_1(n)$ and $s_3(n)$ after the loop terminates.