A screenshot of a computer screen

Description automatically generated

using System;

using System.Diagnostics;

namespace ECE2310\_HW06\_01

{

class Program

{

static void Main(string[] args)

{

// Initialize stopwatch for execution timing.

Stopwatch stopWatch = new Stopwatch();

stopWatch.Start();

Console.WriteLine("-----------------------------------------");

Console.WriteLine("Determining Perfect Numbers (using sqrt) ");

Console.WriteLine("-----------------------------------------");

// values for part a and part b

int aLimit = 2000;

int bStart = 5;

int bLimit = 5000;

// Part B: Display perfect numbers and divisors from {1, 2000}

Console.WriteLine($"Perfect numbers between 1 and {aLimit}:");

for (int i = 1; i < aLimit; i++)

{

if (isPerfect(i) == true)

{

Console.Write($"Perfect num: {i,-4} = ");

printDivisors(i);

Console.WriteLine();

}

}

// Part C: Display perfect numbers and divisors from {5, 5000}

Console.WriteLine("-----------------------------------------");

Console.WriteLine($"Perfect numbers between {bStart} and {bLimit}:");

int count = 0;

for (int i = bStart; i < bLimit; i++)

{

if (isPerfect(i) == true)

{

count++;

Console.Write($"{i,-4}");

}

}

Console.WriteLine($"\nCount: {count}");

Console.WriteLine("-----------------------------------------");

stopWatch.Stop();

Console.WriteLine($"Execution Time: {stopWatch.ElapsedMilliseconds} ms");

}

private static bool isPerfect(int number)

{

for (int i = 1; i < number; i++)

{

/\*

Implementation of Euclid-Euler theorem

Theorem = 2^(p-1)\*[2^(p) - 1] is an even natural number when 2^(p-1) is prime

\*/

long x = (long)(Math.Pow(2, i - 1) \* (Math.Pow(2, i) - 1));

if (x == number && isPrime(i))

{

return true;

}

}

return false;

}

private static void printDivisors(int number)

{

Console.Write("1");

for (int i = 2; i < number; i++)

{

if (number % i == 0)

{

Console.Write($" + {i}");

}

}

}

private static bool isPrime(int number)

{

/\*

Current method inefficient as it calculates the squareroot each time through loop

\*/

if (number == 1) return false;

if (number == 2) return true;

if (number % 2 == 0) return false;

var limit = Math.Floor(Math.Sqrt(number));

for (int i = 3; i <= limit; i += 2)

{

if (number % i == 0) return false;

}

return true;

}

}

}