Question 1

By listing the first six prime numbers: 2, 3, 5, 7, 11, and 13, we can see that the 6th prime is 13.

What is the 10 001st prime number?

**PROGRAM:**

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

namespace Assignment\_2

{

class Program

{

static void Main(string[] args)

{

int end = 10001;

long[] primes = new long[end];

// we know 6 primes: 2,3,5,7,11,13

//Ihave given the elements of primes because it is required foreach loop

primes[0] = 2;

primes[1] = 3;

primes[2] = 5;

primes[3] = 7;

primes[4] = 11;

primes[5] = 13;

// prime after number 13 will have position 6 when counting from 0

int position = 6;

for (long x = primes[5]; position < end; x += 2)

{

if (IsPrime(x, primes))

primes[position++] = x;

}

Console.WriteLine("The " + end + " th prime is " + primes[end - 1]);

Console.ReadKey();

}

//IsPrime checks if the int is Prime or not

static bool IsPrime(Int64 p, long[] primes)

{

Int64 max = (Int64)Math.Ceiling(Math.Sqrt(p));

//all elements are iterated in foreach of primes to get the next prime

foreach (long element in primes)

{

//avoiding all divisible by old prime

if (p % element == 0)

return false;

//only the max(square gives the number) greater than element to avoid its multiple

if (element > max)

break;

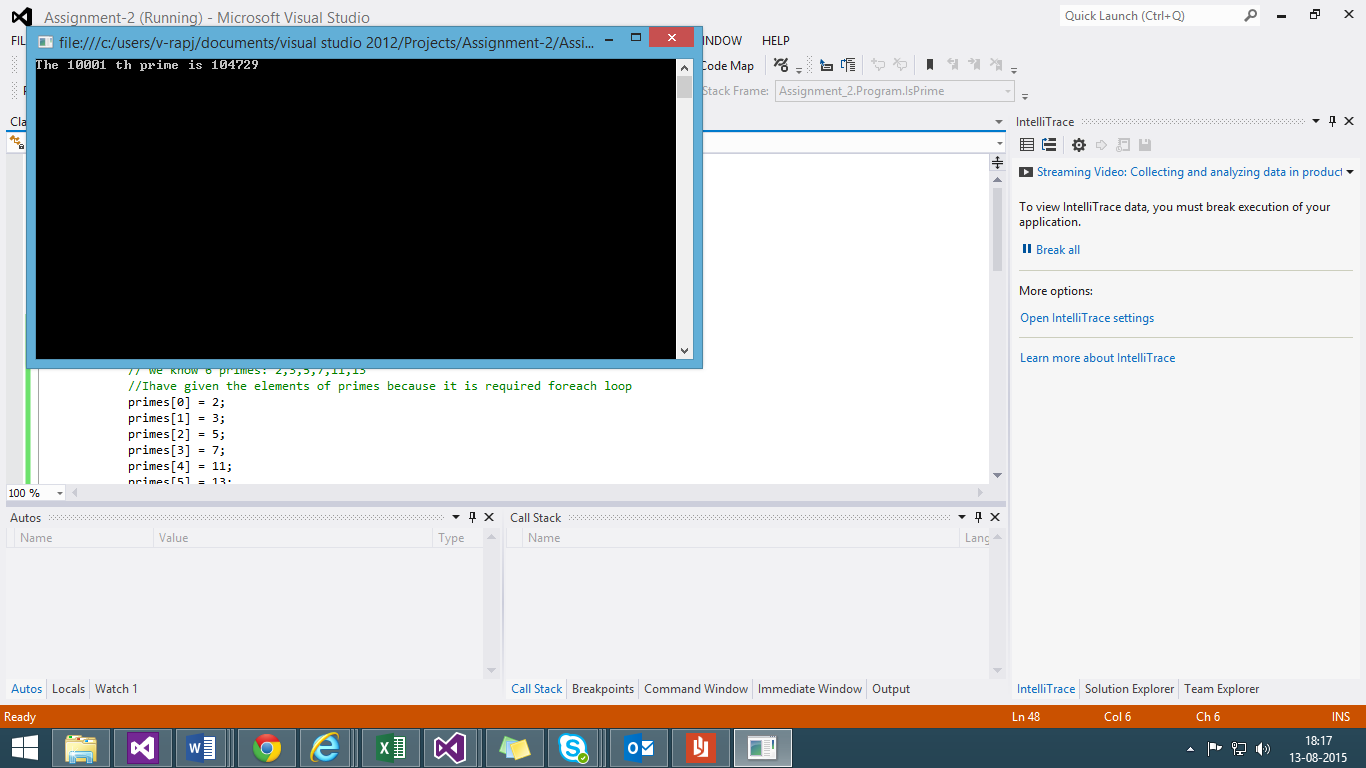
}

return true;

}

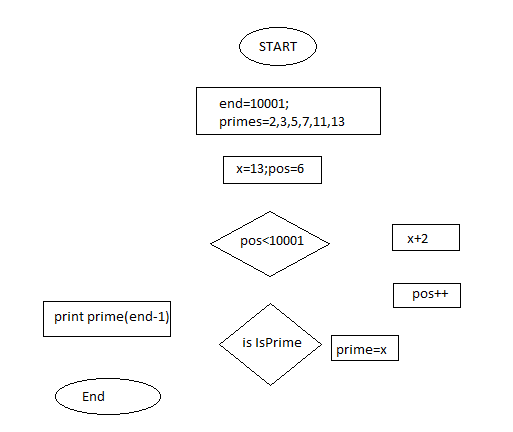
}

}



Algorithm:

1. Declare an End point wiz., 10001 in our case
2. We take an array of type long with size of end point
3. We take the first 6 elements for the primes array so that it can be used for the foreach loop for traversal
4. For loop having the last element of primes and incremented by 2 to avoid even numbers
5. It has a IsPrime method with integers and the array passed to it
6. IsPrime method takes and stores the last elements square root and compare it to the element for eliminating the square of the primes
7. IsPrime return true in case of prime and elements get added to array till the for loop achieves the End point.
8. The last before element in the array gives the 10001st Prime Number.



Question 12

n! means n × (n − 1) × ... × 3 × 2 × 1

For example, 10! = 10 × 9 × ... × 3 × 2 × 1 = 3628800, and the sum of the

digits in the number 10! is 3 + 6 + 2 + 8 + 8 + 0 + 0 = 27.

Find the sum of the digits in the number 100!

PROGRAM:

using System;

using System.Collections.Generic;

using System.Linq;

using System.Text;

using System.Numerics;

namespace Assignment\_2

{

class Class1

{

static void Main(string[] args)

{

int sum = 0;

//BigInteger is used for storing each digit in the 100!

BigInteger fact = 1;

//use of foor loop can reduce complexity

for (int i = 2; i <= 100; i++)

fact \*= i;

Console.WriteLine("The factorial of 100 is {0}",fact);

//each number is seperated using this while loop and added

while (fact > 0)

{

sum += (int)(fact % 10);

fact /= 10;

}

Console.WriteLine("The sum of digits in 100! is {0}", sum);

Console.ReadKey();

}

}

}

ALGORITH:

1. Sum and Fact is set to 0 and 1 respectively.
2. Use for loop to get the factorial of 100 starting from 2.
3. In while loop if fact is greater than 0 get the sum of each digit using modulo operator

