```
//package hogehoge.com;

import java.math.BigInteger;

//厳密解を求めるためのクラス
public class ExactSolution {
static int NumLen = 10;
static final int NumLenA = 5;
static int UpLimit =99999;
static boolean[] sieve ;
static int[] sosuList ;

public static void main(String[] args){
NumLen = Integer.parseInt(args[0]);
exactSol(NumLen);
}

public static long[] exactSol(int num){
//素数を探す範囲を指定
if(num % 2 == 1){ //奇数なら
UpLimit=(int)Math.pow(10, ( (num+1)/2) ) / 3 ;
}else{ //偶数なら
UpLimit=(int)Math.pow(10, num/2 );
}

sieve = new boolean[UpLimit];
sosuList = new int[UpLimit/5];

System.out.println("UpLimit = " + UpLimit );

//実行時間計測
long startTime = System.currentTimeMillis();

//素数探索用マスク作成
```

```
//package hogehoge.com;

import java.math.BigInteger;

//厳密解を求めるためのクラス
public class ExactSolution {
static int NumLen = 10;
static final int NumLenA = 5;
static int UpLimit =99999;
static boolean[] sieve ;
static int[] sosuList ;

public static void main(String[] args){
NumLen = Integer.parseInt(args[0]);
exactSol(NumLen);
}

public static long[] exactSol(int num){
//素数を探す範囲を指定
if(num % 2 == 1){ //奇数なら
UpLimit=(int)Math.pow(10, ( (num+1)/2) ) / 3 ;
}else{ //偶数なら
UpLimit=(int)Math.pow(10, num/2 );
}

sieve = new boolean[UpLimit];
sosuList = new int[UpLimit/5];

System.out.println("UpLimit = " + UpLimit );

//実行時間計測
long startTime = System.currentTimeMillis();

//素数探索用マスク作成
```

```
BigNumber.initPrimeList();
BigNumber.makePrimeMask();

//素数リストの作成
sieve();
sosuList();
//実行時間計測
long stopTime = System.currentTimeMillis();

//実行時間を出力
System.out.println("Run Time(Prime Number search) = " +
(stopTime - startTime) + " ms " );

int n=0;
for (int i=0; i<sosuList.length ; i++){
if(sosuList[i] != 0){
// System.out.print(sosuList[i]);
n++;
// if(n%100 ==0){
// System.out.println();
// }else{
// System.out.print(" ");
// }
}
}
System.out.println();
System.out.println("The number of Prime Number = " + n);

//厳密解を算出
long candidate[] = {0,0};
long max=(long)Math.pow(10, num)-1; //999999999
long min=(long)Math.pow(10, NumLenA-1); //10000
```

```
BigNumber.initPrimeList();
BigNumber.makePrimeMask();

//素数リストの作成
sieve();
sosuList();
//実行時間計測
long stopTime = System.currentTimeMillis();

//実行時間を出力
System.out.println("Run Time(Prime Number search) = " +
(stopTime - startTime) + " ms " );

int n=0;
for (int i=0; i<sosuList.length ; i++){
if(sosuList[i] != 0){
// System.out.print(sosuList[i]);
n++;
// if(n%100 ==0){
// System.out.println();
// }else{
// System.out.print(" ");
// }
}
}
System.out.println();
System.out.println("The number of Prime Number = " + n);

//厳密解を算出
long candidate[] = {0,0};
long max=(long)Math.pow(10, num)-1; //999999999
long min=(long)Math.pow(10, NumLenA-1); //10000
```

```
int count=0;

System.out.println("The number of max = " + max);
System.out.println("The number of min = " + min);

A:for (long N=max; N>=min ; N -= 1){
/* /マスクを使って判定
int maskoffset = (int)(max % (long)BigNumber.maskLen) ;
if(!BigNumber.primeMask[maskoffset]){
int maskoffset = (int)(max % (long)BigNumber.maskLen) ;
maskoffset = (maskoffset == 0 ? BigNumber.maskLen-1 : maskoffset -1 );
continue;
}
maskoffset = (maskoffset == 0 ? BigNumber.maskLen-1 : maskoffset-1 );
*/
//System.out.println(" candidate = N :" + N);
B:for (int i=0; i<sosuList.length ; i++){
if(sosuList[i] != 0){
if( N % sosuList[i] == 0 ){
count++;
//System.out.println(" candidate = " + count + ":" + sosuList[i]);
if(count>=3){
count=0;
break B;
}

if(sosuList[i] < min){
count=0;
break B;
}
```

```
int count=0;

System.out.println("The number of max = " + max);
System.out.println("The number of min = " + min);

A:for (long N=max; N>=min ; N -= 1){
/* /マスクを使って判定
int maskoffset = (int)(max % (long)BigNumber.maskLen) ;
if(!BigNumber.primeMask[maskoffset]){
int maskoffset = (int)(max % (long)BigNumber.maskLen) ;
maskoffset = (maskoffset == 0 ? BigNumber.maskLen-1 : maskoffset -1 );
continue;
}
maskoffset = (maskoffset == 0 ? BigNumber.maskLen-1 : maskoffset-1 );
*/
//System.out.println(" candidate = N :" + N);
B:for (int i=0; i<sosuList.length ; i++){
if(sosuList[i] != 0){
if( N % sosuList[i] == 0 ){
count++;
//System.out.println(" candidate = " + count + ":" + sosuList[i]);
if(count>=3){
count=0;
break B;
}

if(sosuList[i] < min){
count=0;
break B;
}
candidate[count-1]=sosuList[i];

if(count==1){
if( N % candidate[0] != 0){
count=0;
break B;
}
candidate[1] = N / candidate[0];
if(isPrimeNum(candidate[1],0)==1){
break A;
}

}
/*
if(count==2){
System.out.println(" candidate = N :" + candidate[0] *
candidate[1] +
" A=" + candidate[0] +
" B=" + candidate[1]);
if(candidate[0] * candidate[1] == N){

break A;
}else{
count=0;
break B;
}
}
*/
}
}else if(sosuList[i] == 0){
System.out.println(" N is Prime = " + N);
```

```
candidate[count-1]=sosuList[i];

if(count==1){
if( N % candidate[0] != 0){
count=0;
break B;
}
candidate[1] = N / candidate[0];
if(isPrimeNum(candidate[1],0)==1){
break A;
}

}
/*
if(count==2){
System.out.println(" candidate = N :" + candidate[0] *
candidate[1] +
" A=" + candidate[0] +
" B=" + candidate[1]);
if(candidate[0] * candidate[1] == N){

break A;
}else{
count=0;
break B;
}
}
*/
}
}else if(sosuList[i] == 0){
System.out.println(" N is Prime = " + N);
```

```
count=0;
break B;
}
}
}

//結果を出力
if(count==1){
System.out.println("Exact Solution exsits : "+
"N=" + formatNumber(candidate[0]*candidate[1]) +
" A=" + formatNumber(candidate[0]) +
" B=" + formatNumber(candidate[1]));
//System.out.println(formatNumber(candidate[0]*candidate[1]));
//System.out.println(formatNumber(candidate[0]));
//System.out.println(formatNumber(candidate[1]));

}else{
System.out.println("Exact Solution does not exists");
}
//実行時間計測
stopTime = System.currentTimeMillis();

//実行時間を出力
System.out.println("Run Time = " + (stopTime - startTime) + " ms
" );

long result[] = new long[3];

result[0]=candidate[0];
result[1]=candidate[1];
result[2]=candidate[0] * candidate[1];

return result;
```

```
count=0;
break B;
}
}
}

//結果を出力
if(count==1){
System.out.println("Exact Solution exsits : "+
"N=" + formatNumber(candidate[0]*candidate[1]) +
" A=" + formatNumber(candidate[0]) +
" B=" + formatNumber(candidate[1]));
//System.out.println(formatNumber(candidate[0]*candidate[1]));
//System.out.println(formatNumber(candidate[0]));
//System.out.println(formatNumber(candidate[1]));

}else{
System.out.println("Exact Solution does not exists");
}
//実行時間計測
stopTime = System.currentTimeMillis();

//実行時間を出力
System.out.println("Run Time = " + (stopTime - startTime) + " ms
" );

long result[] = new long[3];

result[0]=candidate[0];
result[1]=candidate[1];
result[2]=candidate[0] * candidate[1];

return result;
```

```java
}

private static void sosuList(){
//ゼロで初期化
for (int i=0; i<sosuList.length ; i++){
sosuList[i] = 0;
}

//素数のみセット
int count=0;

for (int i=2; i<sieve.length ; i++){
if(sieve[i]){
sosuList[count] = i;
count++;
}
}
System.out.println("Last Prime Number = " + sosuList[count-1]);
System.out.println(" count = " + Integer.toString(count));
}

private static String formatNumber(long convNum){
String tempStr=Long.toString(convNum);
int len=tempStr.length();
//System.out.println("String len " + convNum + " = " + len);

StringBuffer buf = new StringBuffer();
//buf.append("1");
for (int i = 0; i < len-1; i++) {
buf.append(tempStr.substring(i,i+1));
if((len-i-1)%5==0){
buf.append(" ");
}
```

```java
}

private static void sosuList(){
//ゼロで初期化
for (int i=0; i<sosuList.length ; i++){
sosuList[i] = 0;
}

//素数のみセット
int count=0;

for (int i=2; i<sieve.length ; i++){
if(sieve[i]){
sosuList[count] = i;
count++;
}
}
System.out.println("Last Prime Number = " + sosuList[count-1]);
System.out.println(" count = " + Integer.toString(count));
}

private static String formatNumber(long convNum){
String tempStr=Long.toString(convNum);
int len=tempStr.length();
//System.out.println("String len " + convNum + " = " + len);

StringBuffer buf = new StringBuffer();
//buf.append("1");
for (int i = 0; i < len-1; i++) {
buf.append(tempStr.substring(i,i+1));
if((len-i-1)%5==0){
buf.append(" ");
}
}
buf.append(tempStr.substring(len-1,len));
buf.append("(" + len + ")");

return buf.toString();
}

private static int isPrimeNum(long PrimeNumber,int mode){
System.out.println("Prime Number check start : " + PrimeNumber);

//素数マスクで仮判定
//int maskoffset = (int)(PrimeNumber % BigNumber.maskLen) ;
//if(!BigNumber.primeMask[maskoffset]) return 0;

//与えられた整数の平方根を探索の上限とする
long rootPrimeNumber = (long)Math.sqrt(PrimeNumber);
for (int i=0; i<sosuList.length ; i++){
if(sosuList[i] != 0){
if( PrimeNumber % sosuList[i] == 0 ){
//System.out.println("Prime Number check : not prime");
return 0;
}
if(sosuList[i]>rootPrimeNumber){

System.out.println("Prime Number check : prime");
return 1;
}
//return 0; // それ以上の繰返しは不要
}else{
return 1;
}

}
```

```
}
buf.append(tempStr.substring(len-1,len));
buf.append("(" + len + ")");

return buf.toString();
}

private static int isPrimeNum(long PrimeNumber,int mode){
System.out.println("Prime Number check start : " + PrimeNumber);

//素数マスクで仮判定
//int maskoffset = (int)(PrimeNumber % BigNumber.maskLen) ;
//if(!BigNumber.primeMask[maskoffset]) return 0;

//与えられた整数の平方根を探索の上限とする
long rootPrimeNumber = (long)Math.sqrt(PrimeNumber);
for (int i=0; i<sosuList.length ; i++){
if(sosuList[i] != 0){
if( PrimeNumber % sosuList[i] == 0 ){
//System.out.println("Prime Number check : not prime");
return 0;
}
if(sosuList[i]>rootPrimeNumber){

System.out.println("Prime Number check : prime");
return 1;
}
//return 0; // それ以上の繰返しは不要
}else{
return 1;
}

}
```

```
return -1;

}

private static void sieve(){
//trueで初期化
for (int i=0; i<sieve.length ; i++){
sieve[i] = true;
}

//0と1は除外
sieve[0] = false;
sieve[1] = false;

int max = (int)Math.sqrt(sieve.length);

for (int p=2; p<=max ; p++){
if(sieve[p]){
for (int i=p*2; i<sieve.length ; i += p ){
sieve[i] = false;
}
}
}
}
}
```

文字数: 4323
空白数: 1387 空白込み文字数: 5710
改行数: 252 改行込み文字数: 5962
単語数: 636

```
return -1;

}

private static void sieve(){
//trueで初期化
for (int i=0; i<sieve.length ; i++){
sieve[i] = true;
}

//0と1は除外
sieve[0] = false;
sieve[1] = false;

int max = (int)Math.sqrt(sieve.length);

for (int p=2; p<=max ; p++){
if(sieve[p]){
for (int i=p*2; i<sieve.length ; i += p ){
sieve[i] = false;
}
}
}
}
}
```

文字数: 4323
空白数: 1387 空白込み文字数: 5710
改行数: 252 改行込み文字数: 5962
単語数: 636

全体を表示 | カラー1 カラー2 [✓ モノクロ]