Discrete Assignment 3

Name & ID:-

1- Abdelrahman ibrahem fawzy

ID: 18010893

2-Abdelrhman Abdelftah Kassem

ID: 18010948

Question 1

1.1-Problem statement:

Implement sieve of Eratosthenes algorithm for finding all prime numbers up to any given limit**.**

1.2-Used data structures:

We used in our code arrays and built in functions such as (`Math.sqrt()`).

1.3-pseudo code:

```
    boolean[] sieveOfEratosthenes( limit)  {
              boolean[limit+1] isPrime

              //assume all numbers up to the limit are prime
              for (i=0 to limit) {
                 isPrime[i] = true
              }

              //start at 2 and up to Square root of the limit
              for (i=2 to √limit) {

                 if(isPrime[i] = true) {
                        //starting i^2 and by adding i on each iteration
                        // mark the digit at j as non prime
                        //adding i to j on each iteration means that if i=2
                        //then all digits divisible by 2 are marked as non prime
                        //note we start at 4 so values 1,2,3 will always be marked as
                       // prime
                        for(j=i^2 to limit) {
                               isPrime[j] = false
                        }
                 }
              }

              return isPrime
          }
}
```

1.4-Sample runs:

Problems @ Javadoc Declaration Console ☒ Coverage

Main [Java Application] C:\Program Files\Java\jre1.8.0_111\bin\javaw.exe (Jan 7, 2021, 4:26:56 PM)

```
Enter the letter corresponding to your choice
A: Sieve of Eratosthenes algorithm
B: Trial Division algorithm
C: Extended Euclidean algorithm
D: Chinese remainder theorem
E:  Miller’s test

a
Enter the limit:
20
all prime numbers up to 20 :
2, 3, 5, 7, 11, 13, 17, 19,


=======================================================================
```

Problems @ Javadoc Declaration Console ☒ Coverage

Main [Java Application] C:\Program Files\Java\jre1.8.0_111\bin\javaw.exe (Jan 7, 2021, 4:32:56 PM)

```
Enter the letter corresponding to your choice
A: Sieve of Eratosthenes algorithm
B: Trial Division algorithm
C: Extended Euclidean algorithm
D: Chinese remainder theorem
E:  Miller’s test

a
Enter the limit:
70
all prime numbers up to 70 :
2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61, 67,


=======================================================================
```

Question 2

2.1-Problem statement:

Implement Trial Division algorithm for integer factorization.

2.2-Used data structures:

We used in our code arrays and built in functions such as (`Math.sqrt()`).

2.3-pseudo code:

```
    boolean trialDivision( n) {

      boolean [] primes = SieveOfEratosthenes.sieveOfEratosthenes( (int) Math.sqrt(n))
              boolean isPrime = true;
                for (i=2 to primesLenght) {
                if(primes[i]=true) {
                       if (n%i=0)
                              isPrime = false
                 }
              }

                return isPrime;

          }
   }
```

2.4-Assumptions and details:

In this function I called the previous function which I code in Q1 by passing to it (√n) and store the return value in array after that I cheek if n is divisible by (i) .

2.5-Sample runs:

```
Problems  @ Javadoc  Declaration  Console ☒  Coverage
Main [Java Application] C:\Program Files\Java\jre1.8.0_111\bin\javaw.exe (Jan 7, 2021, 5:05:22 PM)
Enter the letter corresponding to your choice
A: Sieve of Eratosthenes algorithm
B: Trial Division algorithm
C: Extended Euclidean algorithm
D: Chinese remainder theorem
E:  Miller’s test

b
Enter the number to test:
10
10 is not Prime
```

```
Problems  @ Javadoc  Declaration  Console ☒  Coverage
Main [Java Application] C:\Program Files\Java\jre1.8.0_111\bin\javaw.exe (Jan 7, 2021, 5:05:22 PM)
=======================================================================


Enter the letter corresponding to your choice
A: Sieve of Eratosthenes algorithm
B: Trial Division algorithm
C: Extended Euclidean algorithm
D: Chinese remainder theorem
E:  Miller’s test

b
Enter the number to test:
23
23 is Prime
```

Question 3

3.1-Problem statement:

Implement the extended Euclidean algorithm that finds the greatest common divisor d of two positive integers a and b. In addition, it outputs Bezout's coefficients s and t such that d = s a + t b

3.2-Used data structures:

We used in our code arrays only.

3.3-pseudo code:

```
int[] extendedEuclidean ( a,  b) {
             if (b = 0)
                    return new int [] {a , 1 , 0}
             int [] arr = extendedEuclidean(b, a%b)
              d = arr[0]
              s = arr[2]
              t = arr[1] - (a/b) * arr[2]
             return new int [] {d , s , t}
}
```

3.4-Sample runs:

```
Problems  @ Javadoc  Declaration  Console  Coverage
Main [Java Application] C:\Program Files\Java\jre1.8.0_111\bin\javaw.exe (Jan 7, 2021, 5:24:49 PM)
Enter the letter corresponding to your choice
A: Sieve of Eratosthenes algorithm
B: Trial Division algorithm
C: Extended Euclidean algorithm
D: Chinese remainder theorem
E:  Miller's test

c
Enter the first number:
20
Enter the second number:
30
GCD = 10
Bezout's coefficients : s = -1,  t = 1
-1(20) + 1(30) = 10



========================================================================
```

Question 4

4.1-Problem statement:

Implement Chinese remainder theorem that takes as input m1, m2, m3, …., mn that are pairwise relatively prime and (a1, a2, …., an) and calculates x such that

x = a1 (mod m1)

x = a2 (mod m2)

…

x= an (mod mn)

4.2-Used data structures:

We used in our code arrays only.

4.3-pseudo code:

```
int chineseRemainder (int[] mn , int[] an ,  k ) {
             m = 1;
            for (int n to mn)
                   m*=n
             x = 0;
            for (i=0 to k) {
                  M = m/mn[i]
                  x += an[i]*M*modInverse(M, mn[i])
            }

            return x%m

}

int modInverse( a,  m)
       {
           int [] gcd = ExtendedEuclidean. extendedEuclidean(a, m) // I called this
                                              //static function to calculate the gsd
            g = gcd[0]
            x = gcd[1]
           if (g != 1)
```

```
            //inverse doesn't exist
             return -1
          else
          {
               res = (x % m + m) % m
             return res
          }
   }
```

4.4-Assumptions and details:

In this method I called the static function from `(ExtendedEuclidean)class` to evaluate the gcd and Bezout's coefficients.

4.5-Sample run:

```
Problems  Javadoc  Declaration  Console  Coverage
Main [Java Application] C:\Program Files\Java\jre1.8.0_111\bin\javaw.exe (Jan 7, 2021, 5:45:02 PM)
Enter the letter corresponding to your choice
A: Sieve of Eratosthenes algorithm
B: Trial Division algorithm
C: Extended Euclidean algorithm
D: Chinese remainder theorem
E:  Miller's test

d
Enter the number of integers:
3
Enter the array of the pairwire relativily primes:
Space separated values works
3 5 7
Enter the array of the remainders:
Space separated values works
2 3 2
23 is the smallest positive integer
```

Question 5

5.1-Problem statement:

Implement Miller's test (a probabilistic primality test).

5.2-Used data structures:

I used only the built in function `Math.random().`

5.3-pseudo code:

```
    // It returns false if n is composite
     // and returns true if n is probably
     // prime. k is an input parameter that
     // determines accuracy level. Higher
     // value of k indicates more accuracy.
     boolean isPrime( n,  k) {

         //base case
         if (n = 0 || n = 1)
             return false
         // base case - 2 is prime
         if (n == 2)
             return true
         // an even number other than 2 is composite
         if (n % 2 == 0)
             return false

         d = n - 1;

         while (d % 2 = 0)
             d /= 2;

         // Iterate given number of 'k' times
         for (i to k)
             if (!miillerTest(d, n))
                 return false

         return true
  }
```

```
// exponentiation. It returns (x^y) % p
     int power( x,  y,  p) {

          res = 1 // Initialize result

         //Update x if it is more than or
         // equal to p
         x = x % p

         while (y > 0) {

             // If y is odd, multiply x with result
             if ((y & 1) = 1)
                 res = (res * x) % p;

             // y must be even now
             y = y >> 1  // y = y/2
             x = (x * x) % p
         }

         return res
 }

    // This function is called for all k trials.
    // It returns false if n is composite and
    // returns false if n is probably prime.
    // d is an odd number such that d*2<sup>r</sup>
    // = n-1 for some r >= 1
     boolean miillerTest( d, n) {

         // Pick a random number in [2..n-2]
         // Corner cases make sure that n > 4
         a = 2 + (Math.random() % (n - 4))

         // Compute a^d % n
         x = power(a, d, n) // exponentiation. It returns (a^d) % n


         if (x = 1 || x = n - 1)
             return true;

         // Keep squaring x while one of the
         // following doesn't happen
         // (i) d does not reach n-1
         // (ii) (x^2) % n is not 1
         // (iii) (x^2) % n is not n-1
         while (d != n - 1) {
             x = (x * x) % n
             d *= 2

             if (x = 1)
                 return false;
             if (x = n - 1)
                 return true
         }
```

```
            // Return composite
            return false
        }

}
```

5.4-Sample runs:

```
Problems  @ Javadoc  Declaration  Console ☒  Coverage
Main [Java Application] C:\Program Files\Java\jre1.8.0_111\bin\javaw.exe (Jan 7, 2021, 6:19:20 PM)
Enter the letter corresponding to your choice
A: Sieve of Eratosthenes algorithm
B: Trial Division algorithm
C: Extended Euclidean algorithm
D: Chinese remainder theorem
E:  Miller’s test

e
Enter the number which you want to test:
15
Enter the number of iterations:
4
'15' is not prime


=======================================================================
```

```
Problems  @ Javadoc  Declaration  Console ☒  Coverage
Main [Java Application] C:\Program Files\Java\jre1.8.0_111\bin\javaw.exe (Jan 7, 2021, 6:20:47 PM)
Enter the letter corresponding to your choice
A: Sieve of Eratosthenes algorithm
B: Trial Division algorithm
C: Extended Euclidean algorithm
D: Chinese remainder theorem
E:  Miller’s test

e
Enter the number which you want to test:
13
Enter the number of iterations:
3
'13' is prime


=======================================================================
```