ASSIGNMENT2: NUMBER THEORY

Seif ElDien Ehab Mostafa Ebrahim – 33

AbdelRahman Adel AbdelFattah AbdelRaouf - 37

Q1: Fast Exponentiation:

Problem Statement:

- Implement the following procedures and compare the execution time of each with the increase of number of bits representing an integer. Also report on when the procedure breaks (overflow).
- Implement it in 4 versions. The following two naive versions, in addition to, fast exponentiation in iterative and recursive versions.

Naïve 1:

```
c =1
for i = 1 to b
c = c * a
c= c mod m
return c
```

Naïve 2:

```
c= 1
for i = 1 to b
c = (c * a) mod m
return c
```

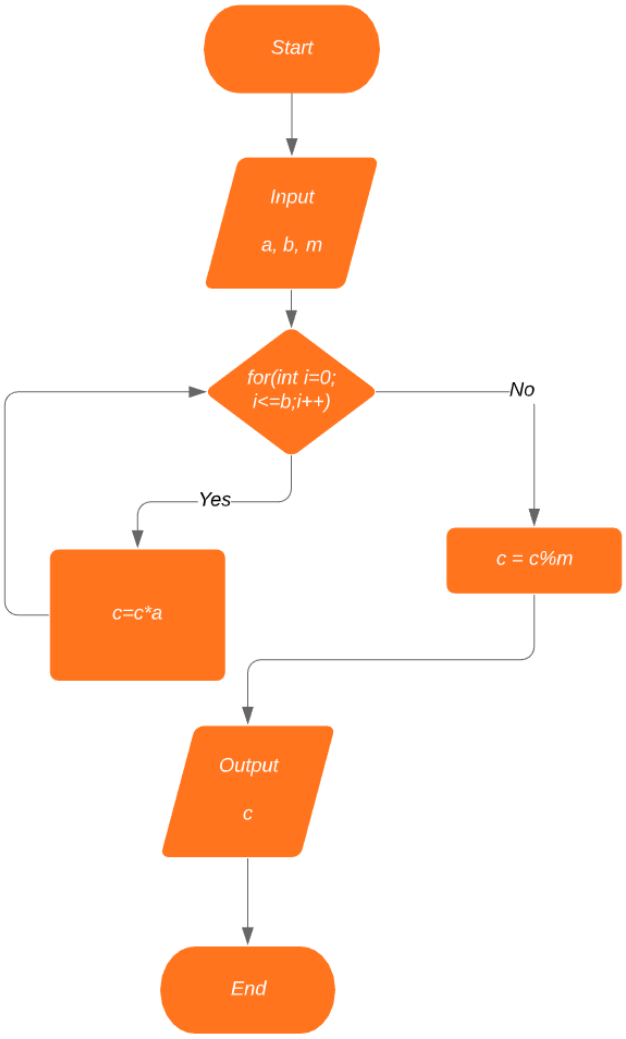
Used Data Structures:

- Int
- Long

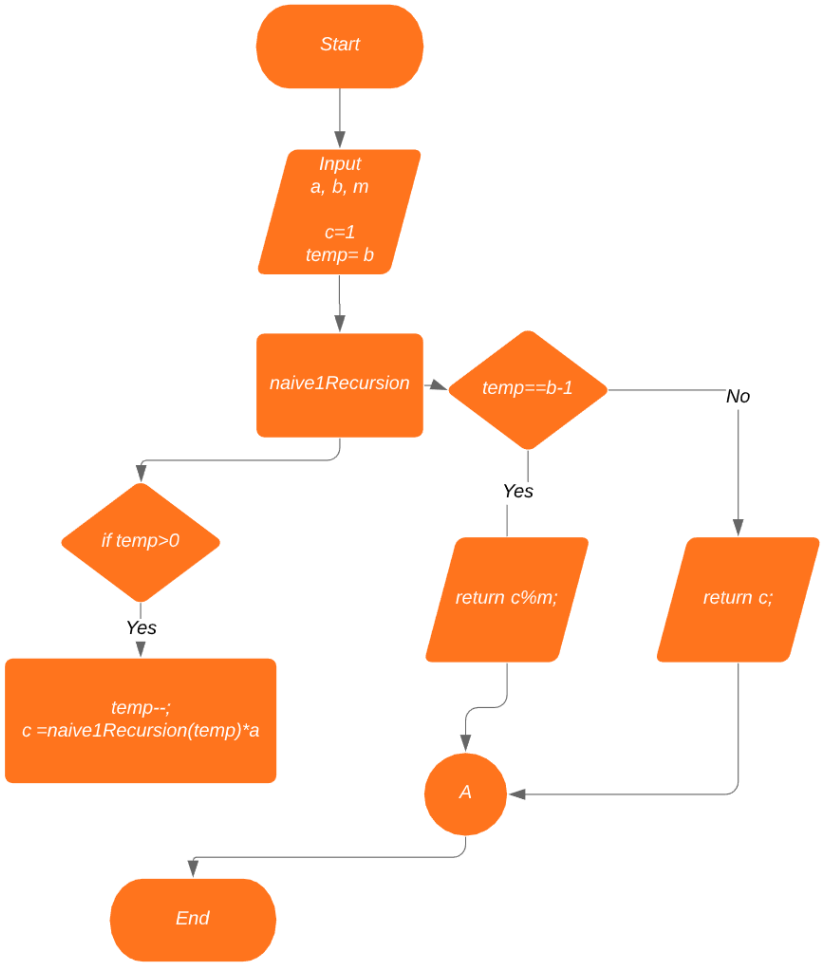
Algorithms:

Naïve 1:

Iterative:

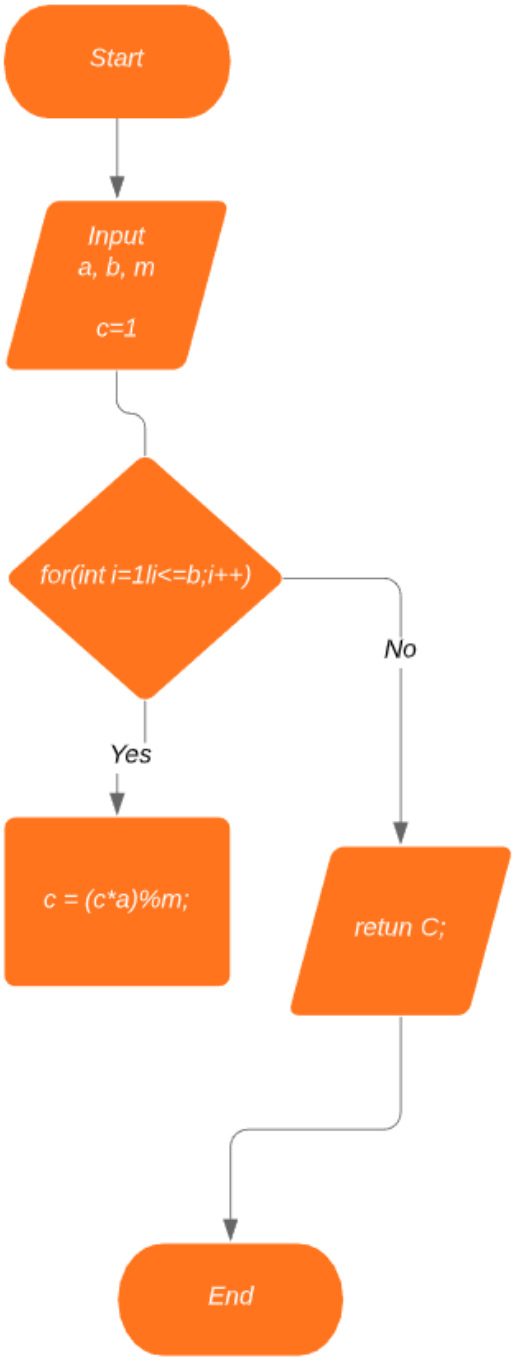


Recursive:

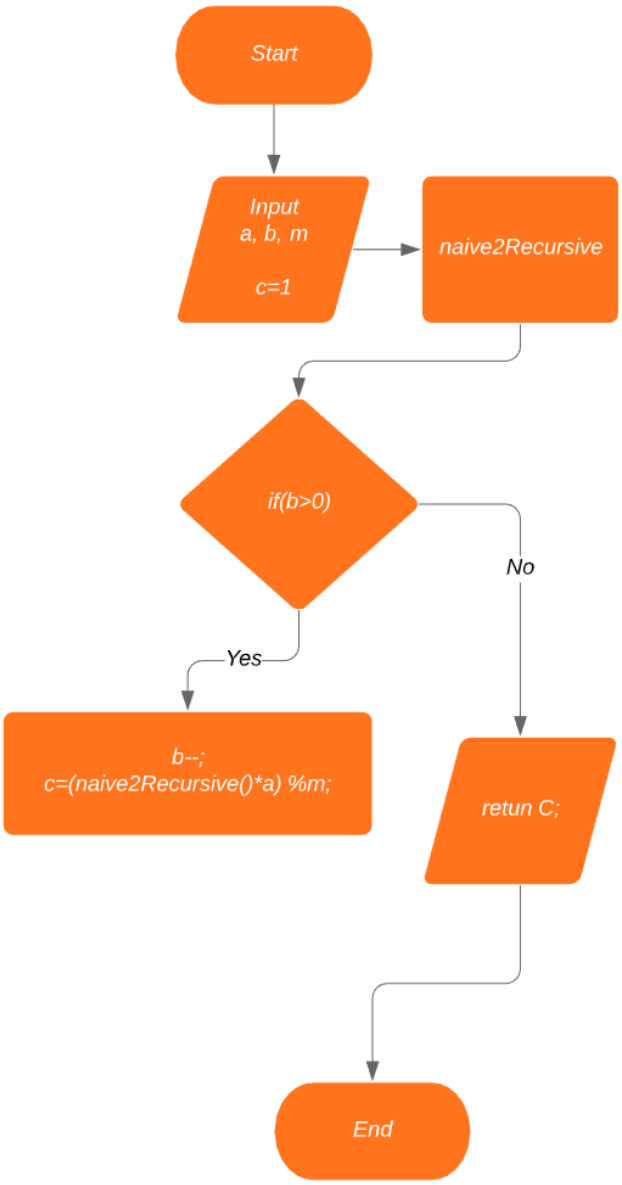

Start
Input
a, b, m
c=1
temp= b
naive1Recursion
temp==b-1
No
Yes
if temp>0
Yes
return c%m;
return c;
temp--;
c =naive1Recursion(temp)*a
A
End

Naïve 2:

Iterative:



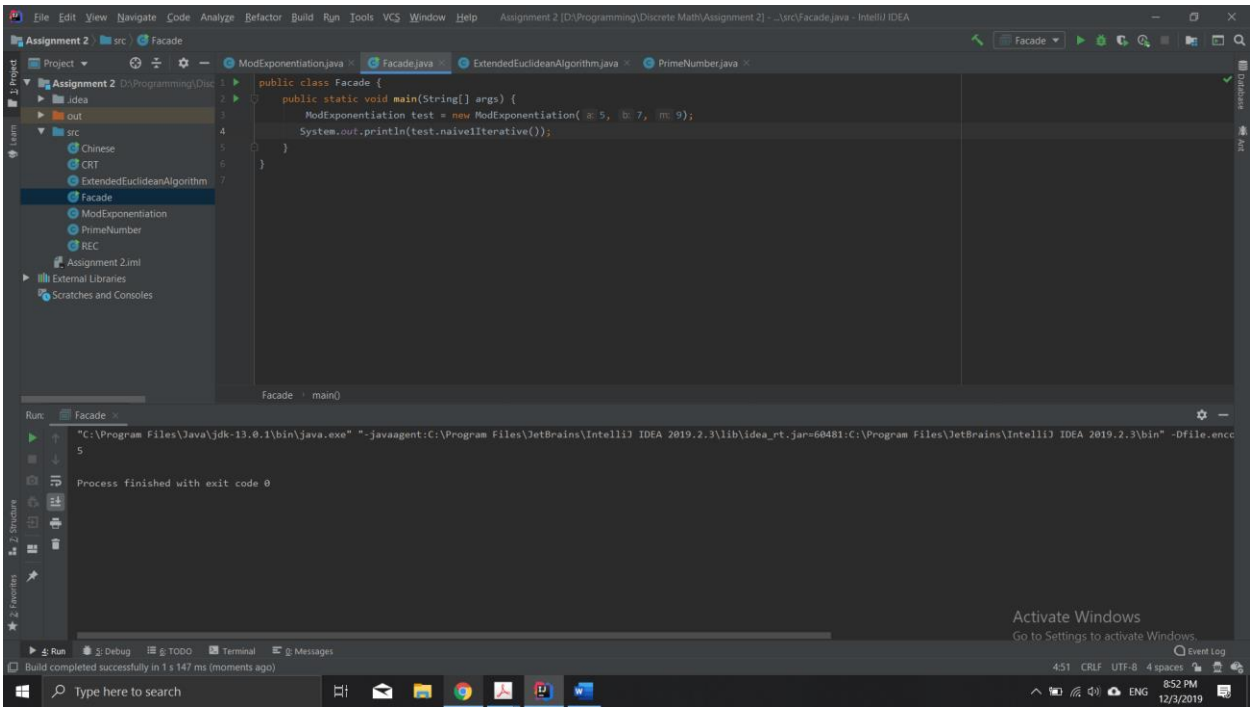
Recursive:

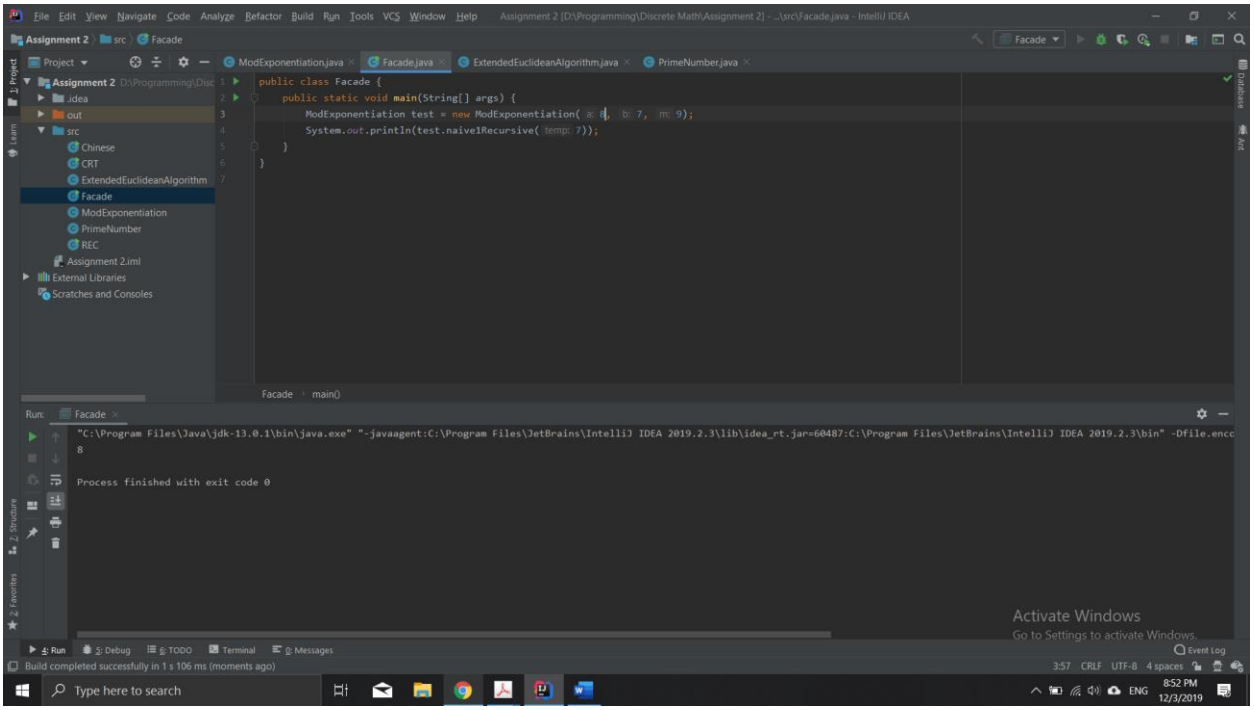


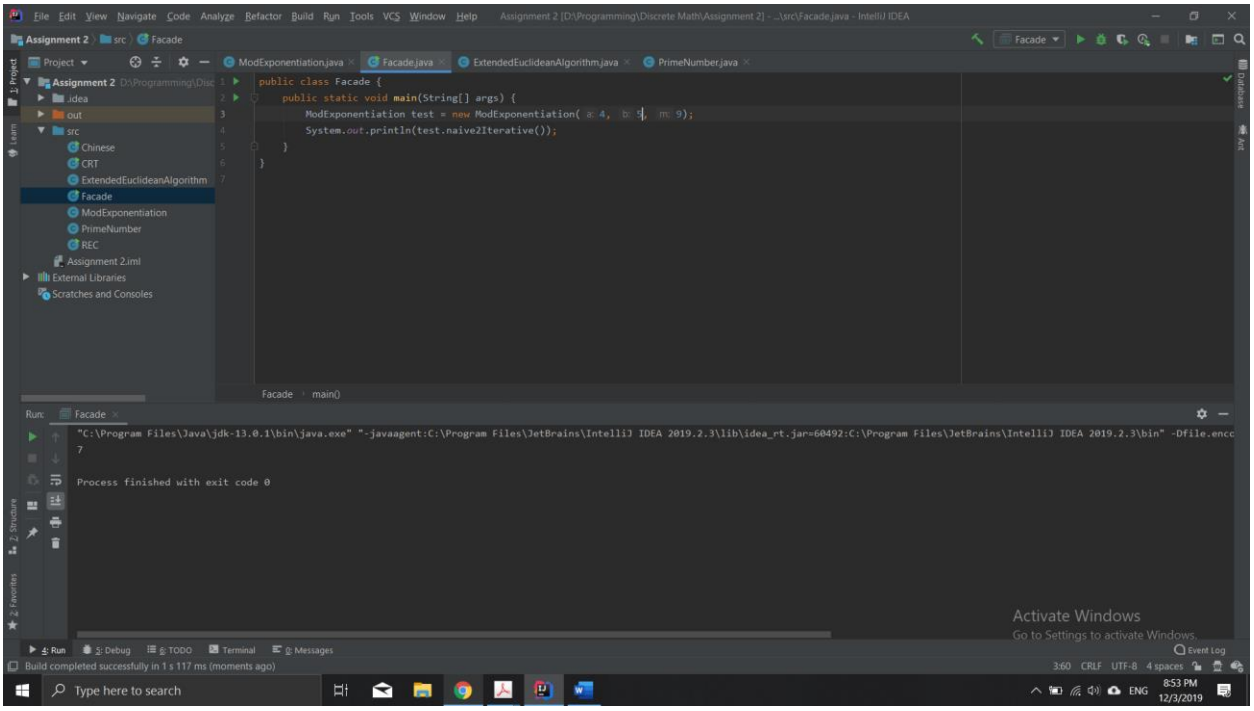
Designs Declarations:

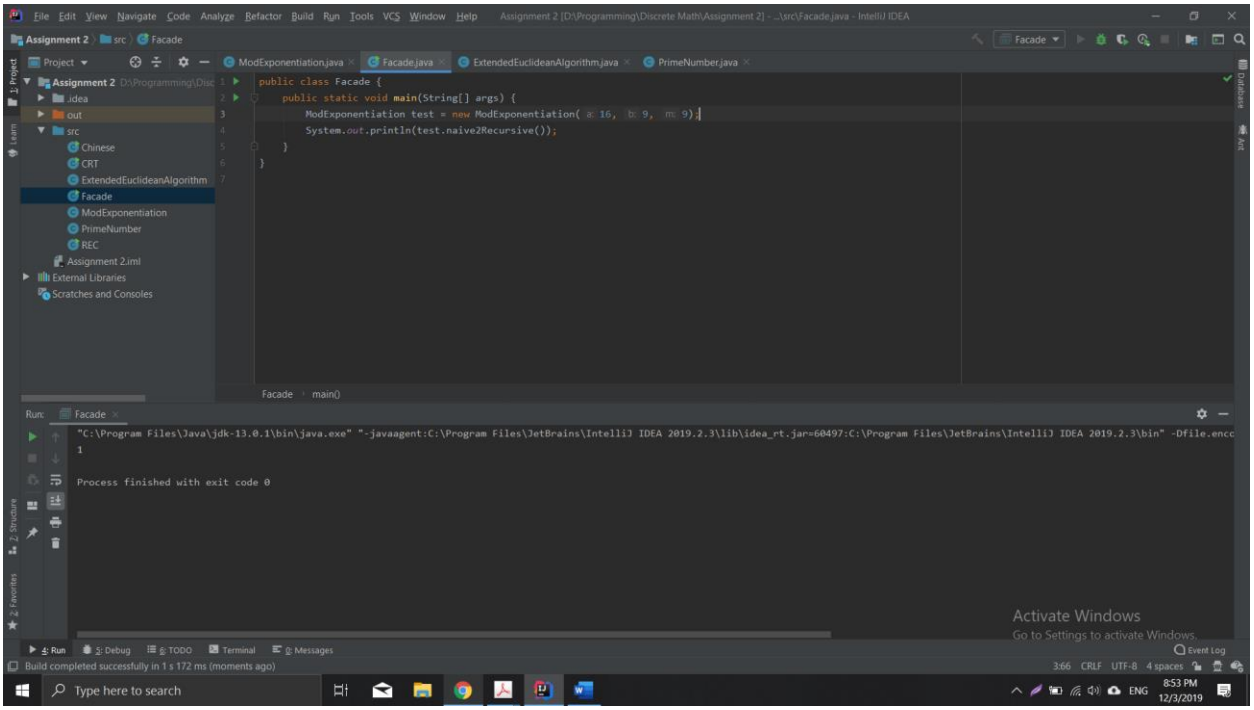
- a, b, m are global variables

Sample Runs:


public class Facade {
public static void main(String[] args) {
ModExponentiation test = new ModExponentiation(a: 5, b: 7, m: 9);
System.out.println(test.naive1Iterative());
}
}
"C:\Program Files\Java\jdk-13.0.1\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2019.2.3\lib\idea_rt.jar=60481:C:\Program Files\JetBrains\IntelliJ IDEA 2019.2.3\bin" -Dfile.encc
5
Process finished with exit code 0
Activate Windows
Go to Settings to activate Windows.
Build completed successfully in 1 s 147 ms (moments ago)
Type here to search
8:52 PM
12/3/2019


public class Facade {
public static void main(String[] args) {
ModExponentiation test = new ModExponentiation(a: 8, b: 7, m: 9);
System.out.println(test.naive1Recursive(temp: 7));
}
}
"C:\Program Files\Java\jdk-13.0.1\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2019.2.3\lib\idea_rt.jar=60487:C:\Program Files\JetBrains\IntelliJ IDEA 2019.2.3\bin" -Dfile.encc
8
Process finished with exit code 0
Activate Windows
Go to Settings to activate Windows.
Build completed successfully in 1 s 106 ms (moments ago)
Type here to search
8:52 PM
12/3/2019





Conclusions and Overflows:

Naïve 2 is better than Naïve 1 because when storing in int datatype with naïve 1 by multiplying everytime without moding makes the number exceeds the int limit so it can't handle large numbers while naïve 2 mods the number in every loop which makes it smaller and can handle larger inputs

Q2: Extended Euclidean Algorithm

Problem Statement:

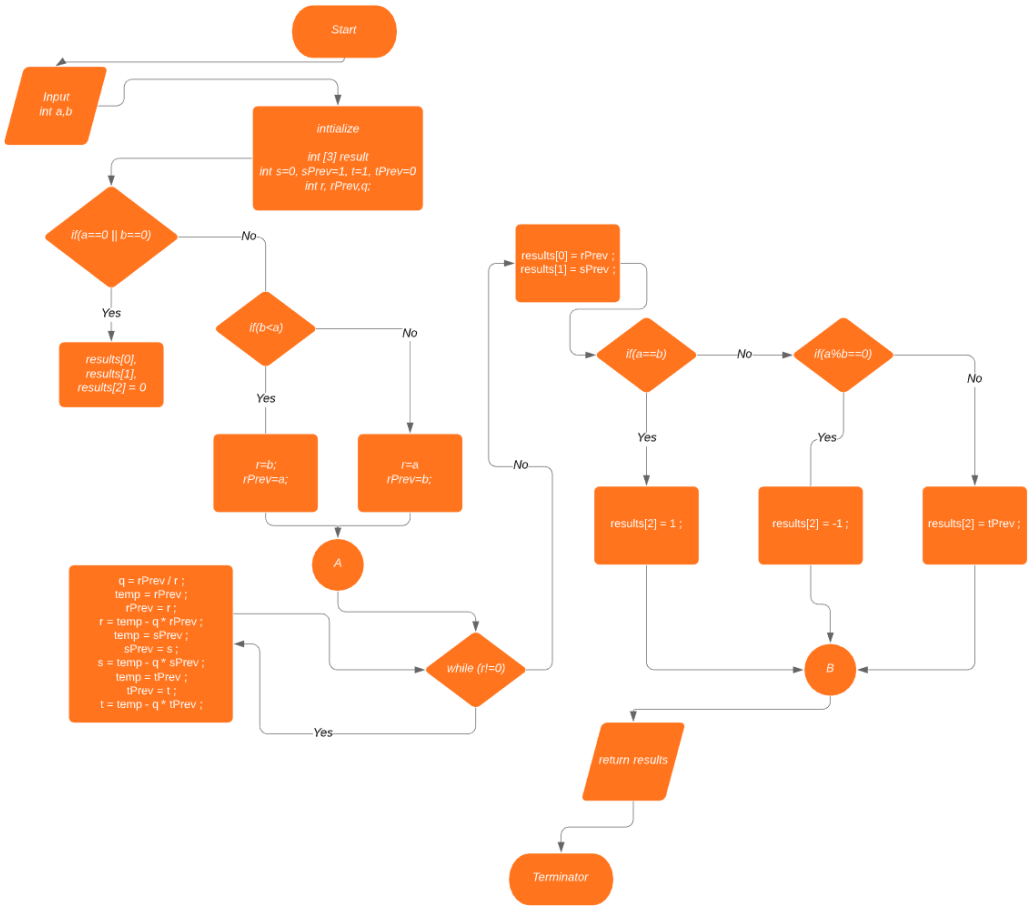
Input: a, b

Output: d = gcd(a,b) and s, t such that d = s.a + t.b

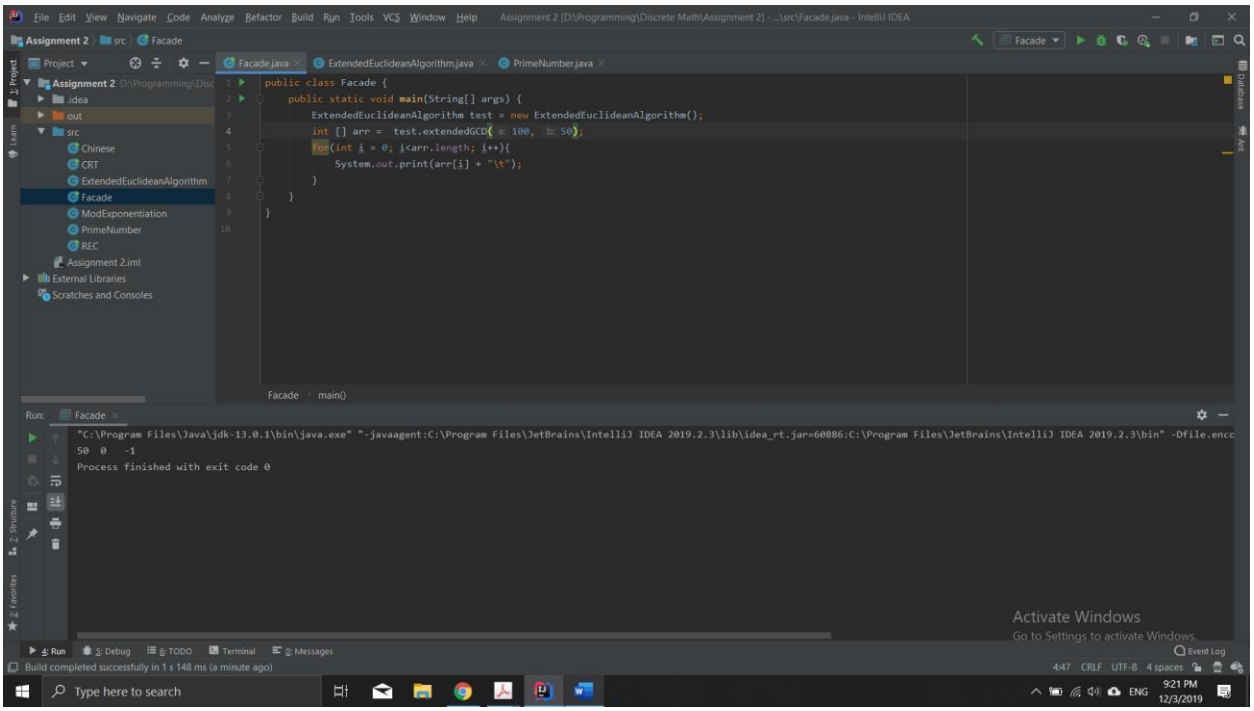
Used Data Structures:

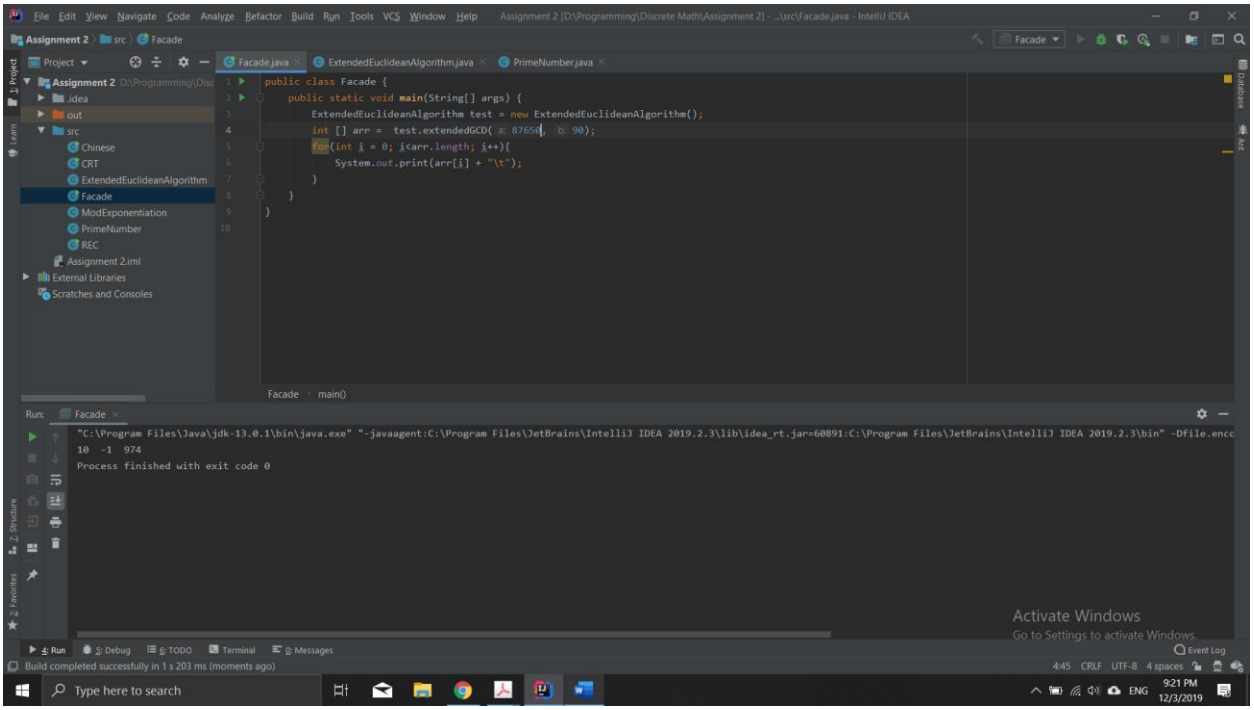
- Int
- Int [] array

Algorithms:



Sample Runs:


public class Facade {
public static void main(String[] args) {
ExtendedEuclideanAlgorithm test = new ExtendedEuclideanAlgorithm();
int [] arr = test.extendedGCD(a: 100, b: 50);
for(int i = 0; i<arr.length; i++){
System.out.print(arr[i] + "\t");
}
}
}
"C:\Program Files\Java\jdk-13.0.1\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2019.2.3\lib\idea_rt.jar=60886:C:\Program Files\JetBrains\IntelliJ IDEA 2019.2.3\bin" -Dfile.encc
50 0 -1
Process finished with exit code 0
Activate Windows
Go to Settings to activate Windows.
Build completed successfully in 1 s 148 ms (a minute ago)
Type here to search
9:21 PM
12/3/2019


public class Facade {
public static void main(String[] args) {
ExtendedEuclideanAlgorithm test = new ExtendedEuclideanAlgorithm();
int [] arr = test.extendedGCD(a: 87650, b: 90);
for(int i = 0; i<arr.length; i++){
System.out.print(arr[i] + "\t");
}
}
}
"C:\Program Files\Java\jdk-13.0.1\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2019.2.3\lib\idea_rt.jar=60891:C:\Program Files\JetBrains\IntelliJ IDEA 2019.2.3\bin" -Dfile.encc
10 -1 974
Process finished with exit code 0
Activate Windows
Go to Settings to activate Windows.
Build completed successfully in 1 s 203 ms (moments ago)
Type here to search
9:21 PM
12/3/2019

Q3: Chinese Remainder Theorem

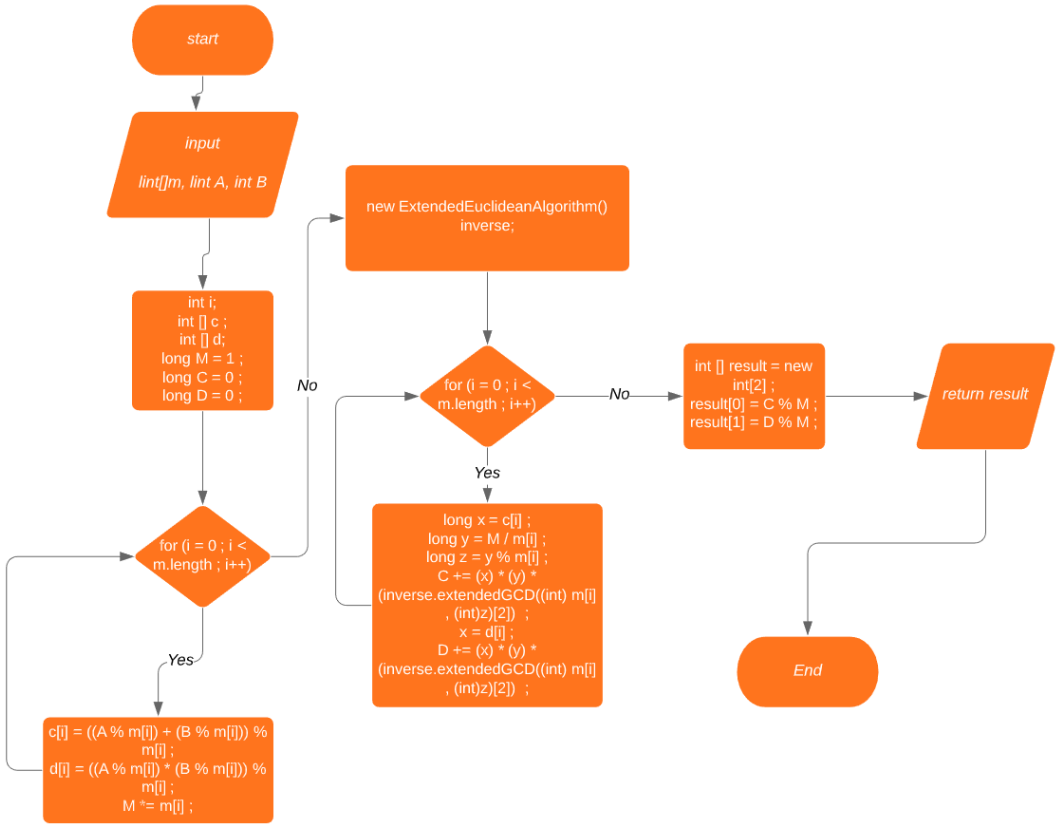
Problem Statement:

Input: m1,m2,……..,mn(M = m1.m2…….mn), A,B 2ZM
Output: C = A+B, D = A * B
Implement the addition and multiplication in both the domain ZM and the domain Zm1 *Zm2 * ….* Zmn: Compare the execution time of both version with the increase of the number of bits representing the integers in ZM.

Used Data Structures:

- Int
- int [] array
- Long

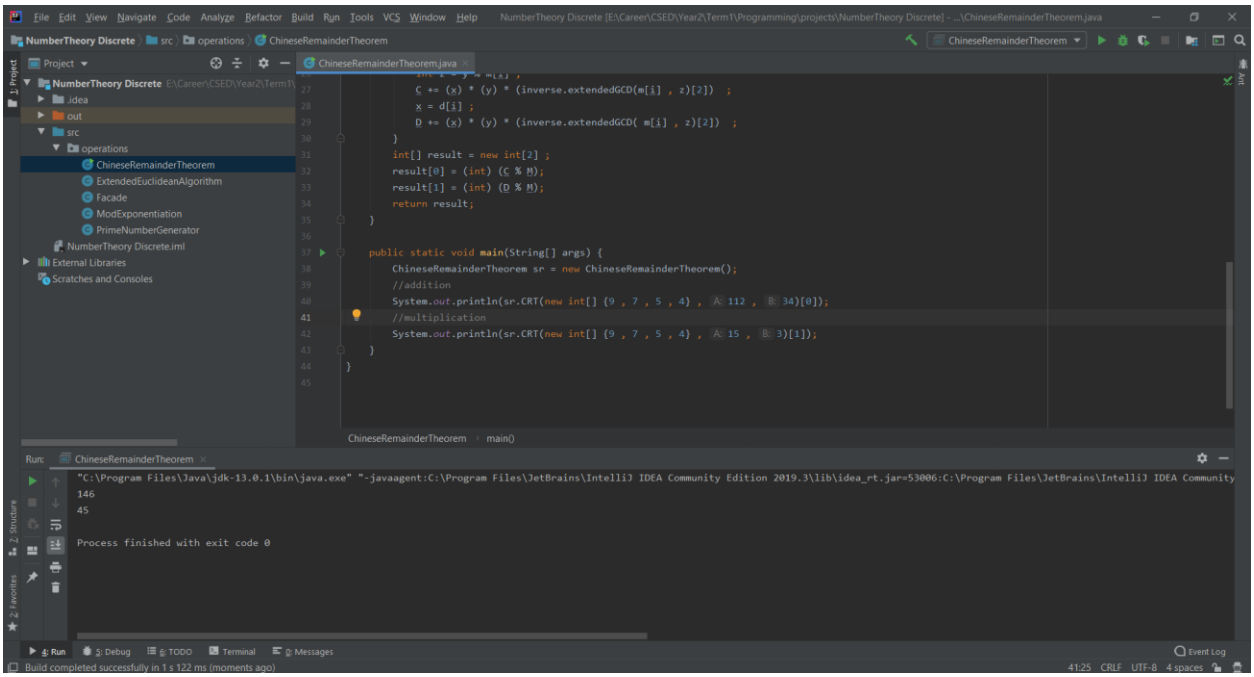
Algorithms:



Designs Declarations:

Uses the ExtendedEuclideanAlgorithm;

Sample Runs:



Q4: Prime Number Generation

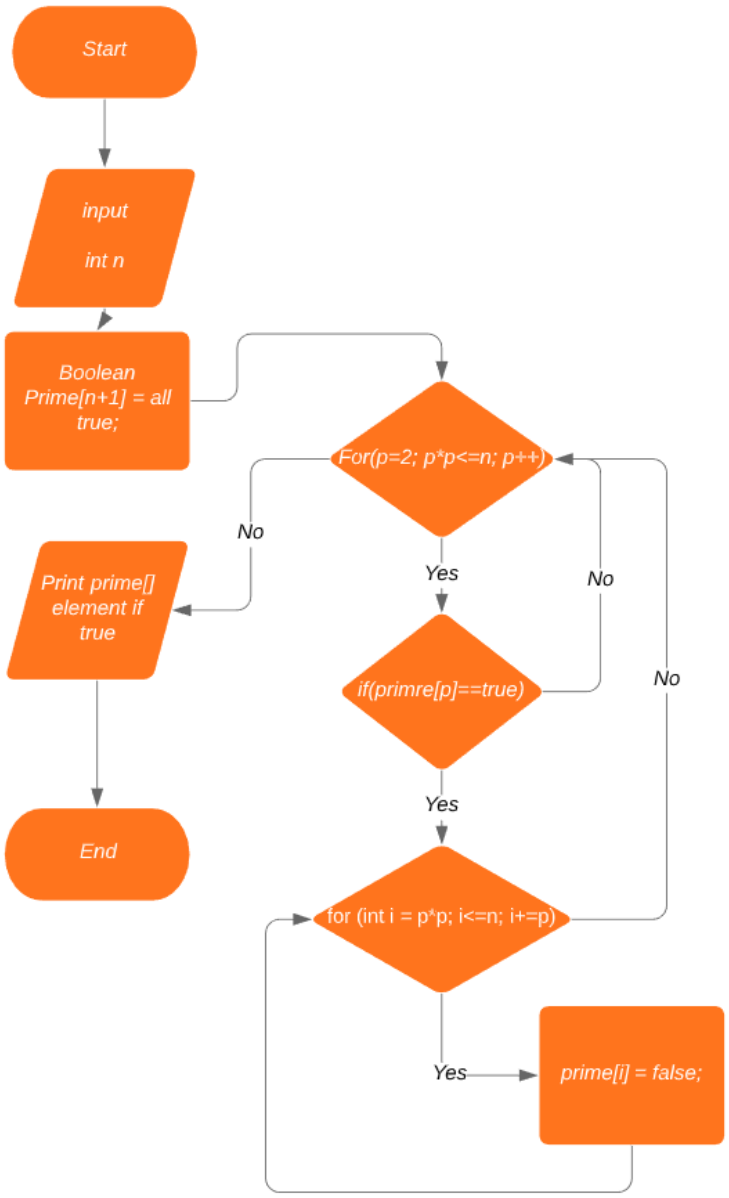
Problem Statement:

Implement a prime number generation procedure.

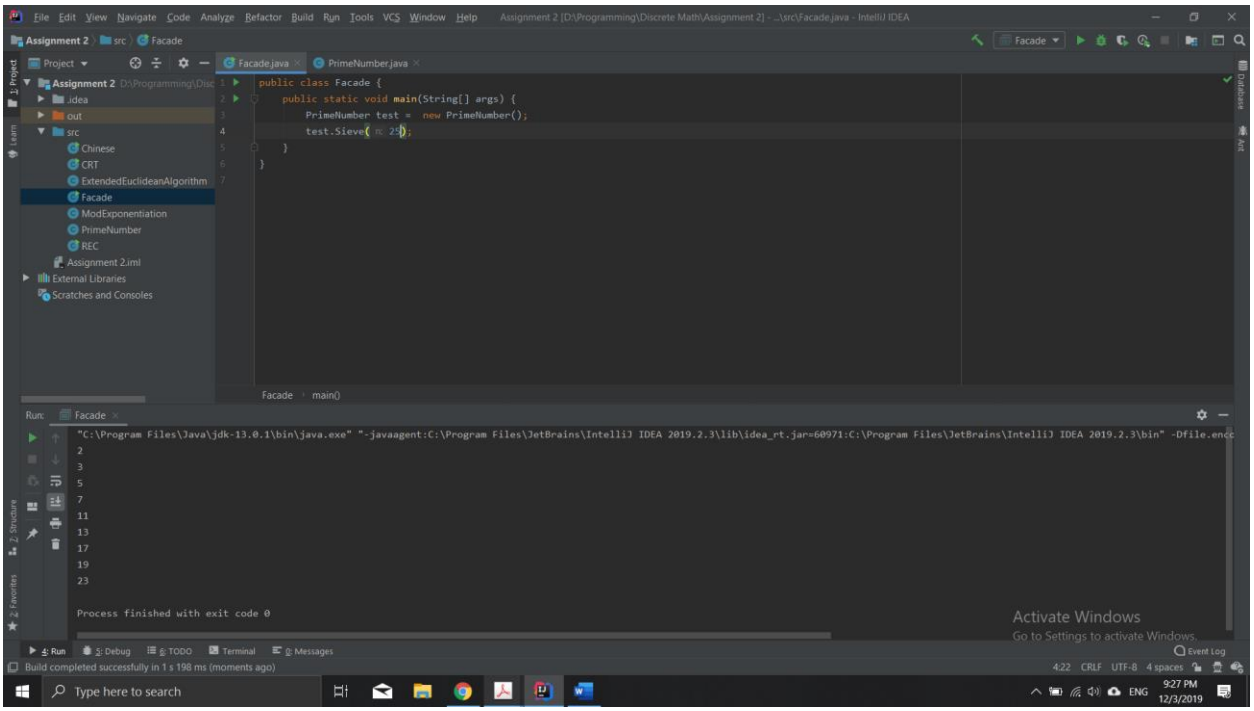
Used Data Structures:

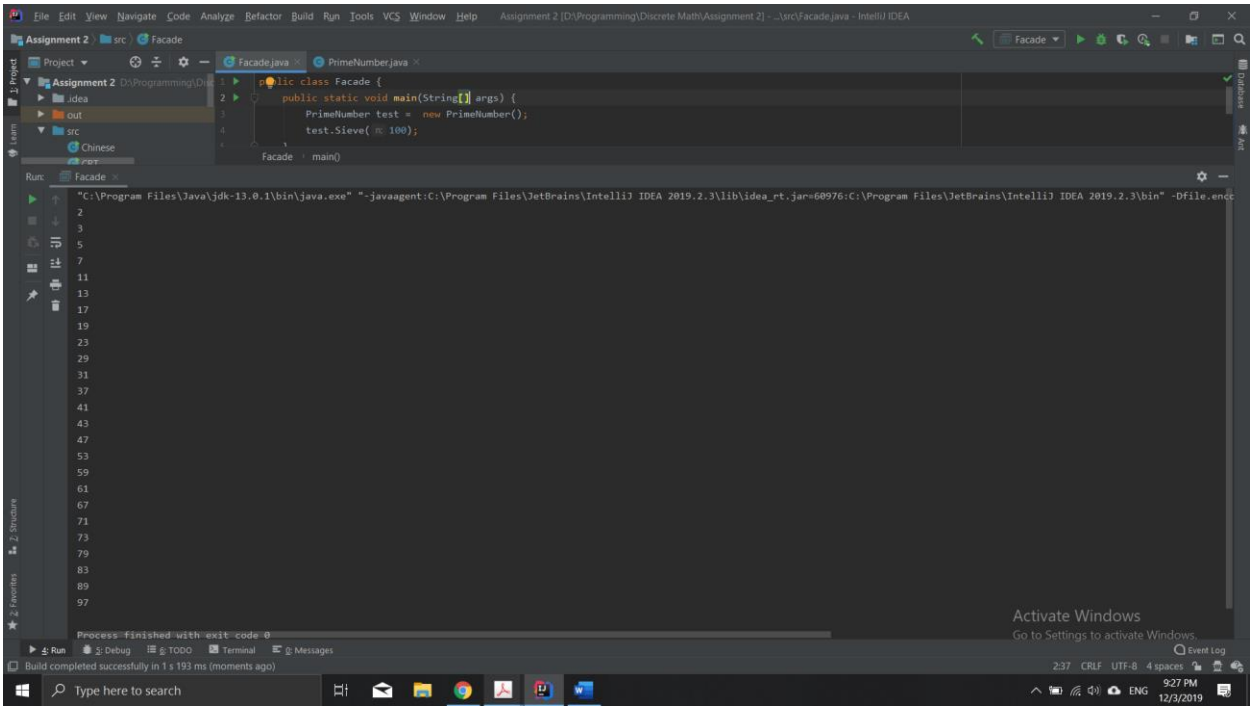
- Int
- Boolean [] array

Algorithm:


Start
input
int n
Boolean Prime[n+1] = all true;
For(p=2; p*p<=n; p++)
No
Print prime[] element if true
End
Yes
No
if(primre[p]==true)
No
Yes
for (int i = p*p; i<=n; i+=p)
Yes
prime[i] = false;

Sample Runs:


public class Facade {
public static void main(String[] args) {
PrimeNumber test = new PrimeNumber();
test.Sieve(n: 25);
}
}
"C:\Program Files\Java\jdk-13.0.1\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2019.2.3\lib\idea_rt.jar=60971:C:\Program Files\JetBrains\IntelliJ IDEA 2019.2.3\bin" -Dfile.enc
2
3
5
7
11
13
17
19
23
Process finished with exit code 0
Activate Windows
Go to Settings to activate Windows.


public class Facade {
public static void main(String[] args) {
PrimeNumber test = new PrimeNumber();
test.Sieve(n: 100);
}
"C:\Program Files\Java\jdk-13.0.1\bin\java.exe" "-javaagent:C:\Program Files\JetBrains\IntelliJ IDEA 2019.2.3\lib\idea_rt.jar=60976:C:\Program Files\JetBrains\IntelliJ IDEA 2019.2.3\bin" -Dfile.enc
2
3
5
7
11
13
17
19
23
29
31
37
41
43
47
53
59
61
67
71
73
79
83
89
97
Process finished with exit code 0
Activate Windows
Go to Settings to activate Windows.