Mathematics for Computer Science

Assignment 2
Number Theory

By: *Shimaa Kamal* *34*

Afnan mousa *15*

Problem Statement :

Problem 1 : Fast Exponentiation

Fast exponentiation , Implement it in 4 versions. The following two na ¨ıve versions, in addition to, fast exponentiation in iterative and recursive versions.

Problem 2 : Extended Euclidean Algorithm

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

Problem 3 : Chinese Remainder Theorem

Input: m1,m2,. . . ..,mn(M = m1.m2.. . . .mn), A,B ∈ZM
Output: C = A+B, D = A * B

Problem 4 : Prime Number Generation

Implement a prime number generation procedure and show its execution time in terms of the number of bits representing an integer.

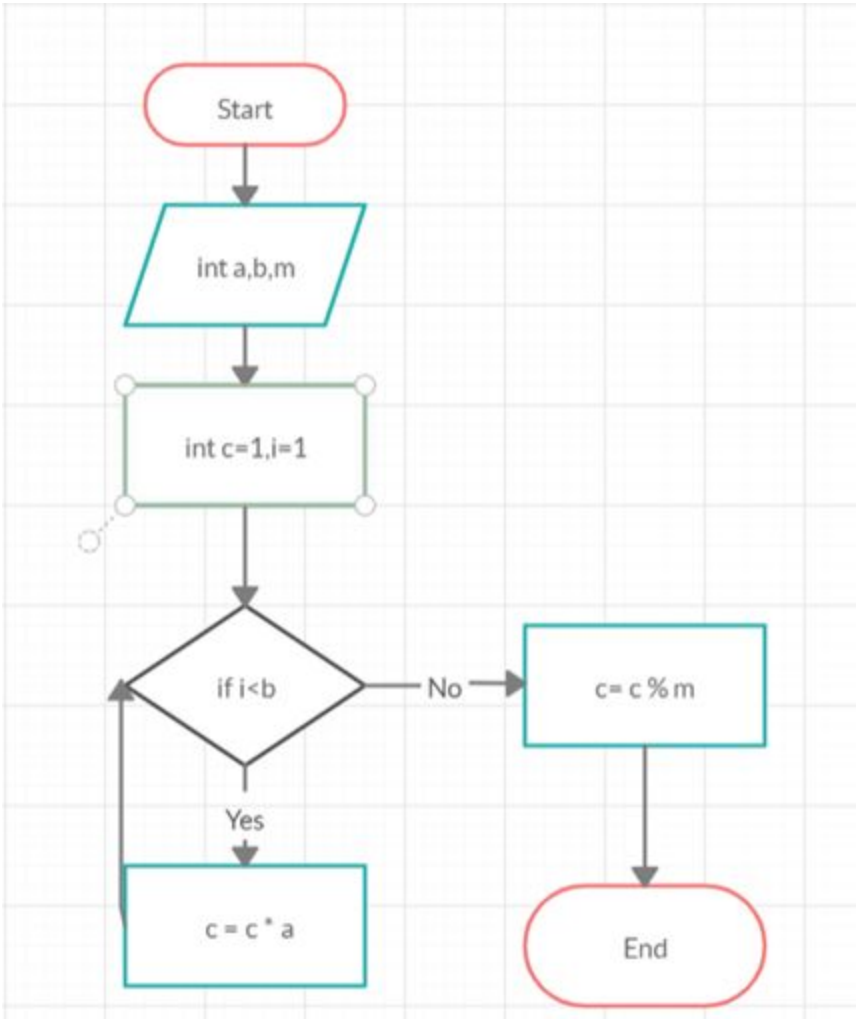
Used data structures :

- Array
- Linked List

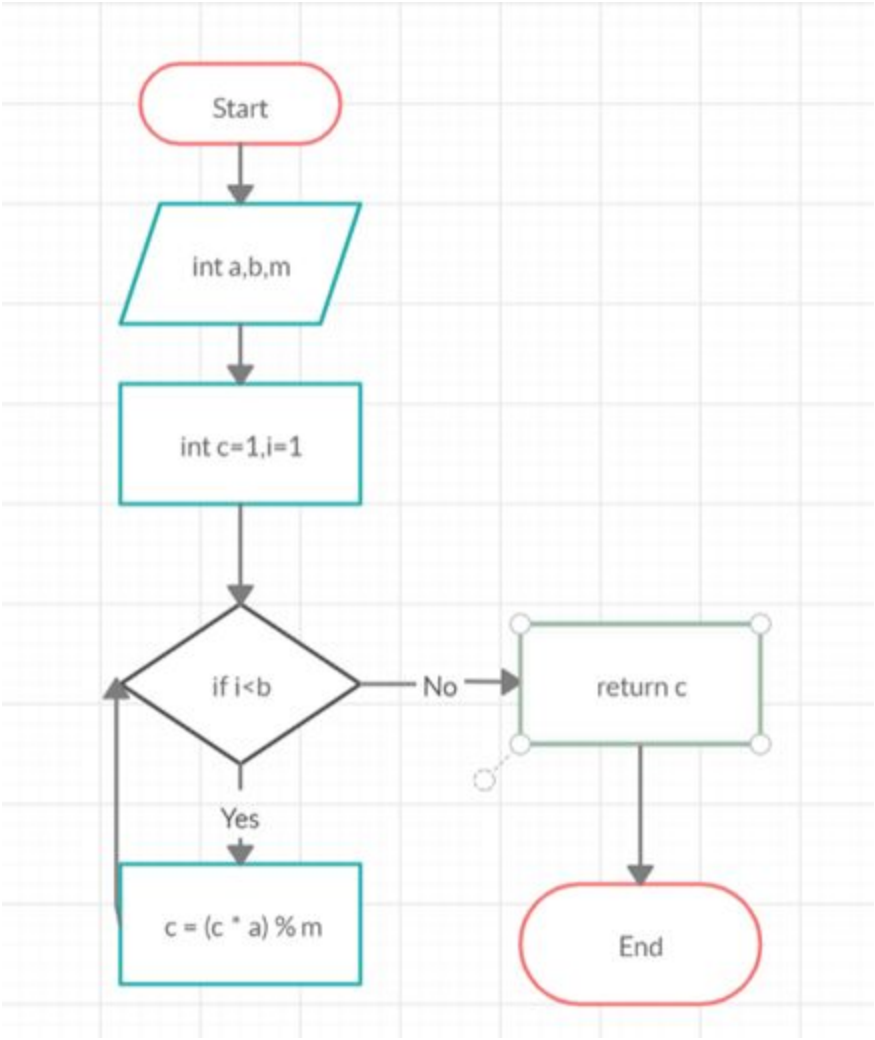
Flow charts :

Problem 1 :

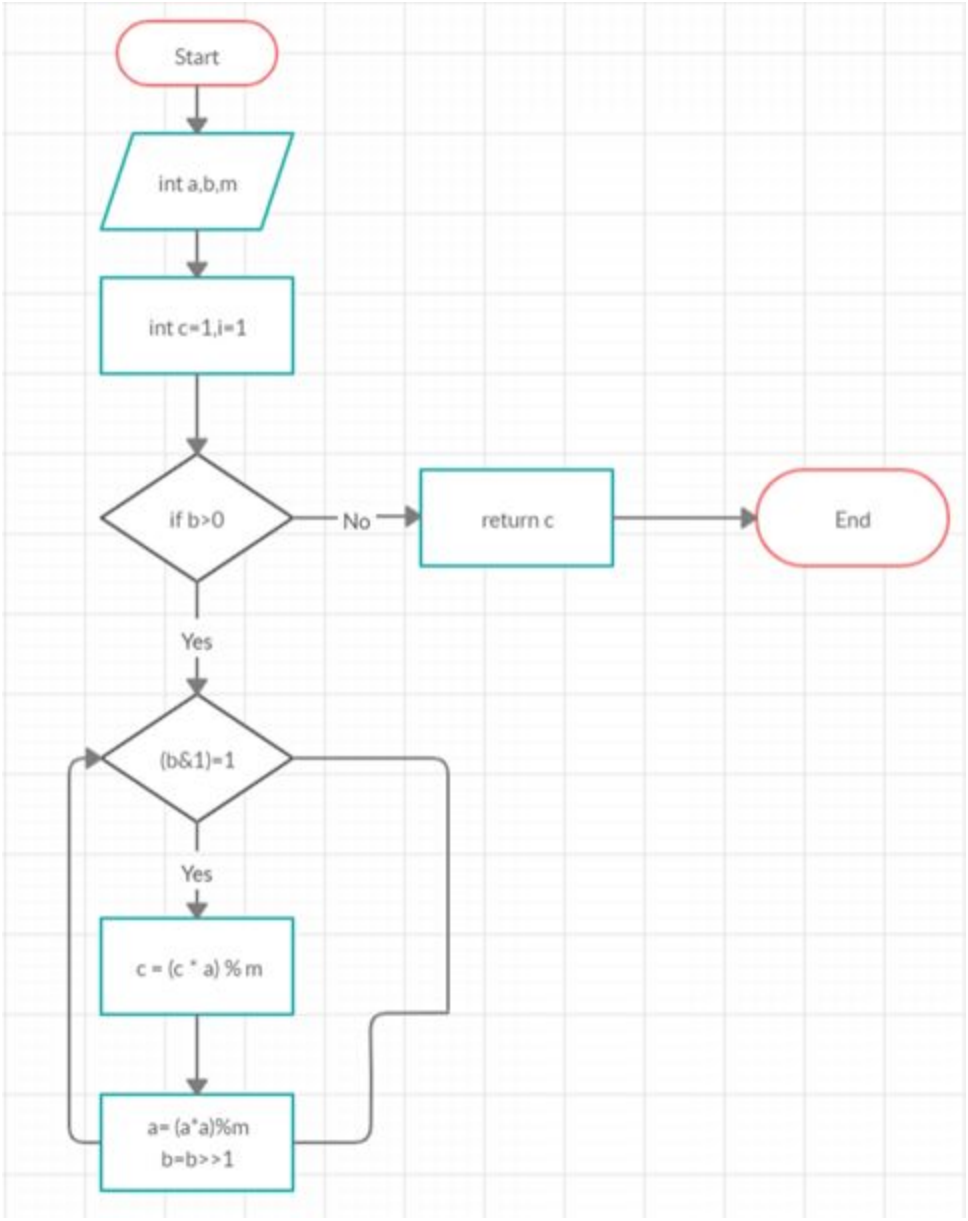
Naive1



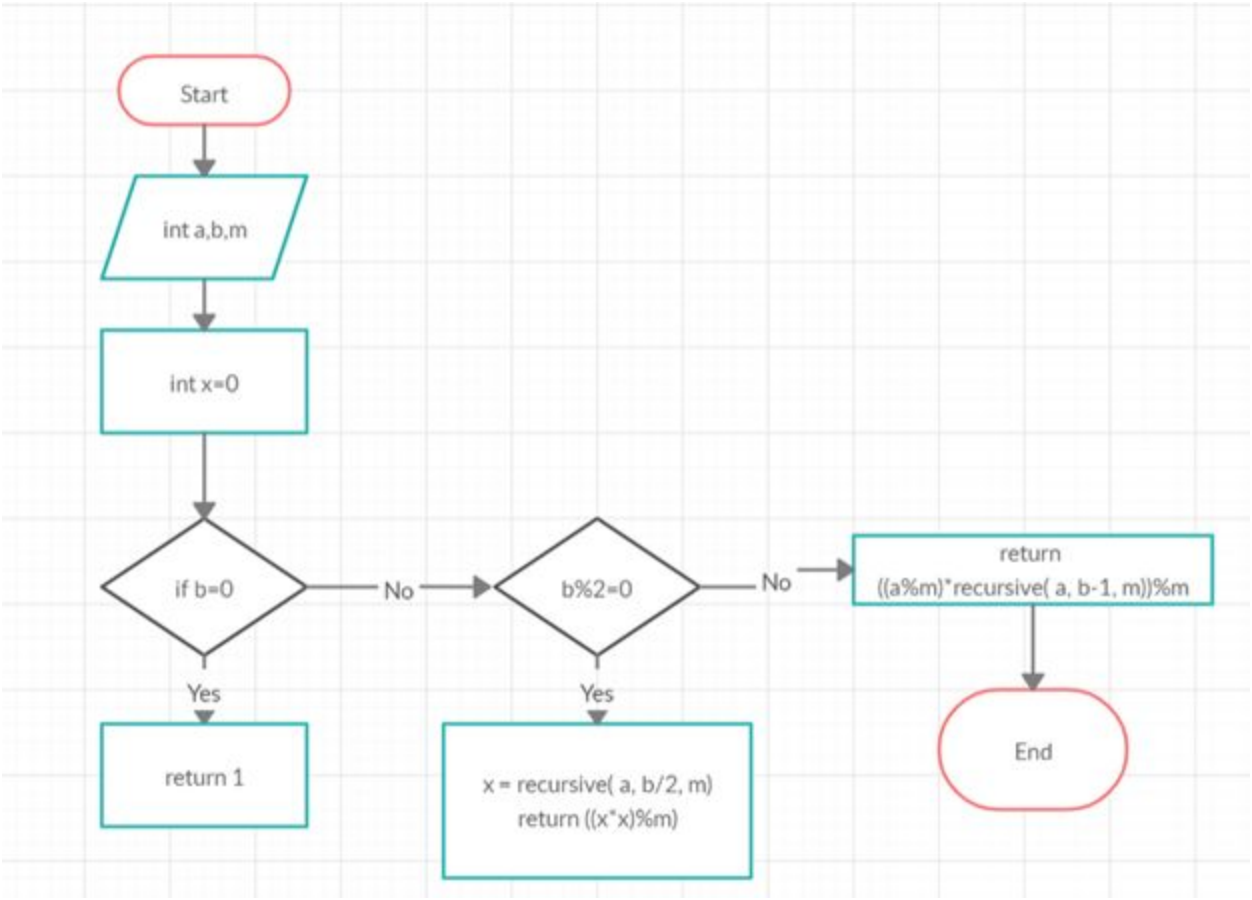
Naive2



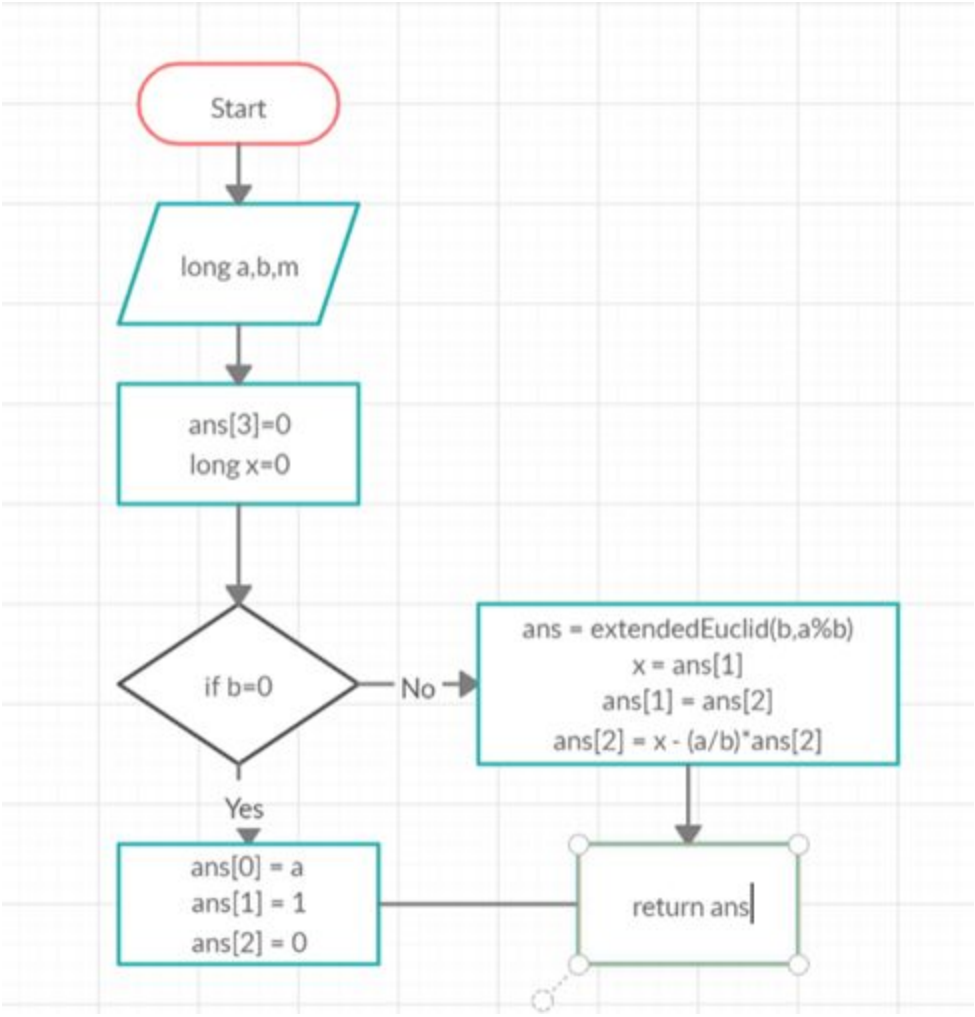
Iterative


Start
int a,b,m
int c=1,i=1
if b>0
No
return c
End
Yes
(b&1)=1
Yes
c = (c * a) % m
a= (a*a)%m
b=b>>1

Recursion

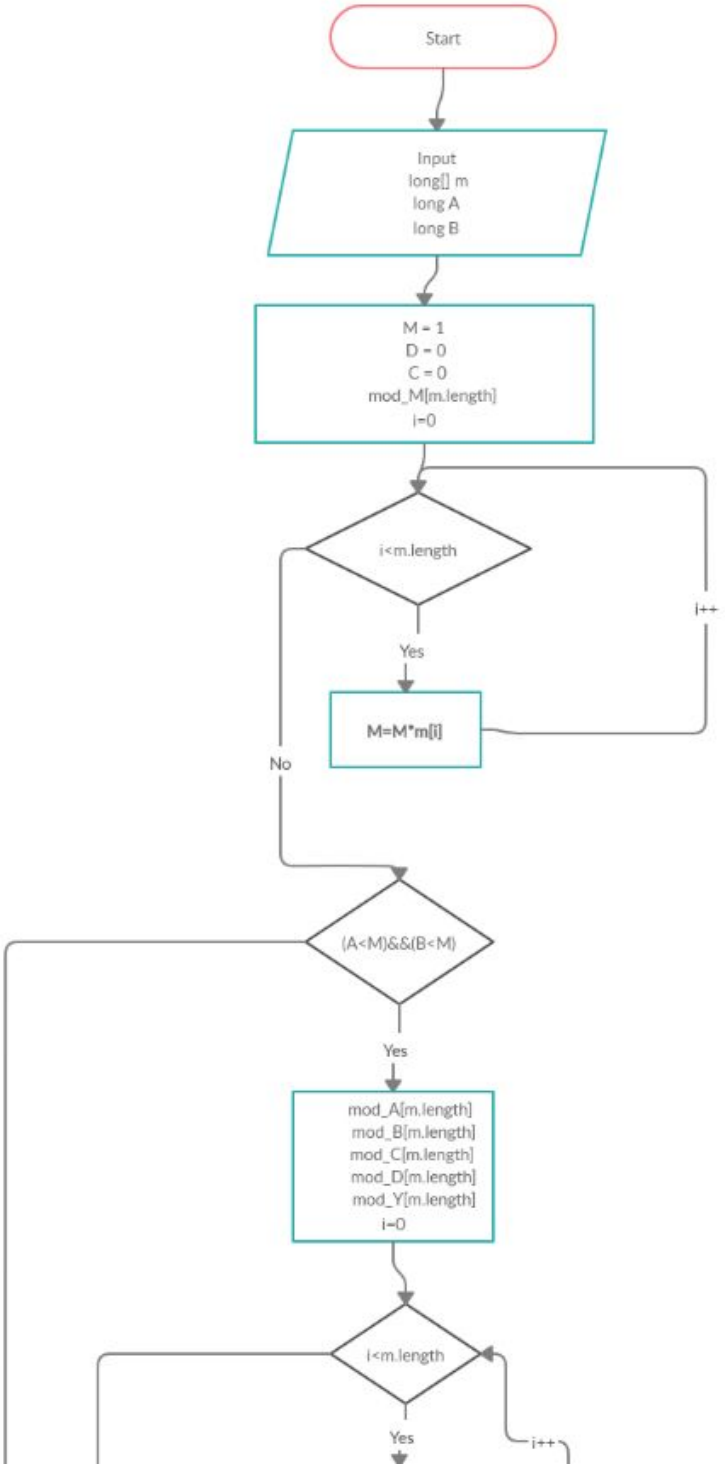

Start
int a,b,m
int x=0
if b=0
No
b%2=0
No
return
((a%m)*recursive(a, b-1, m))%m
Yes
return 1
Yes
x = recursive(a, b/2, m)
return ((x*x)%m)
End

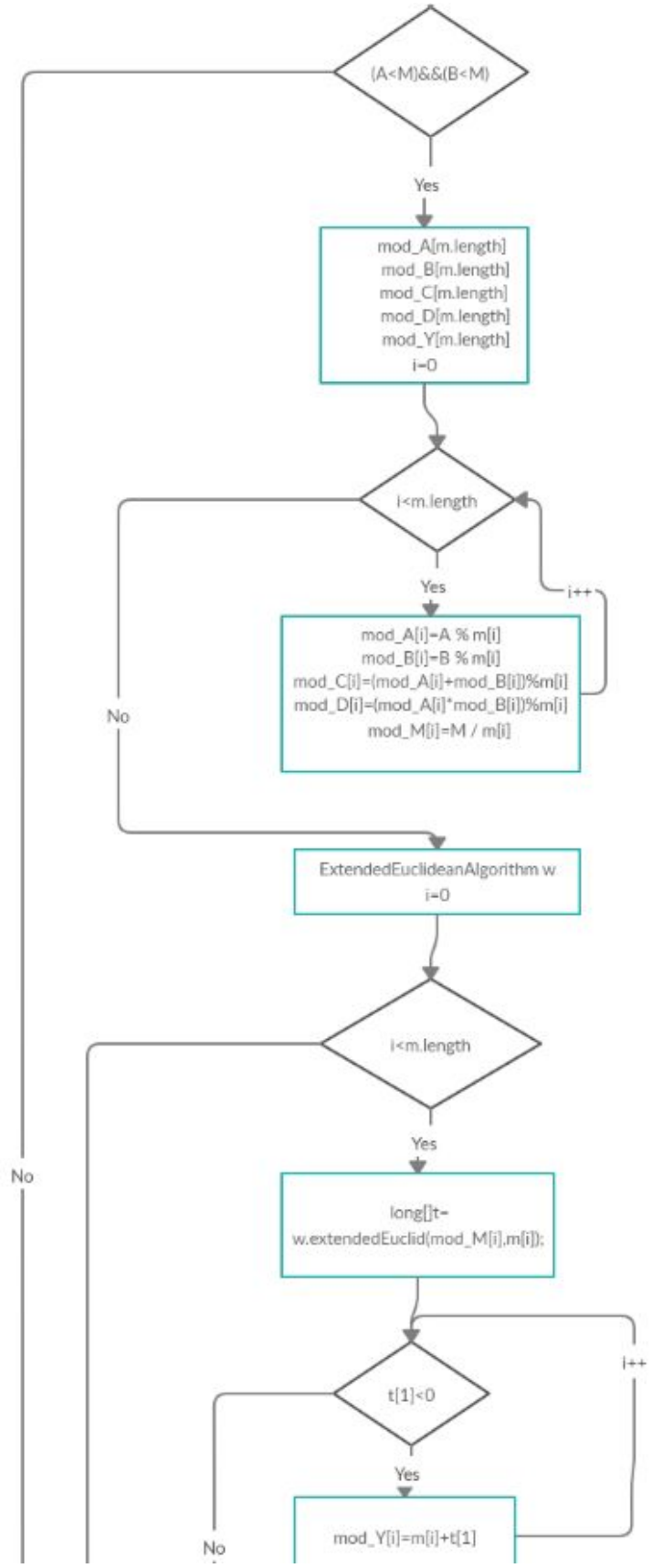
Problem 2 :

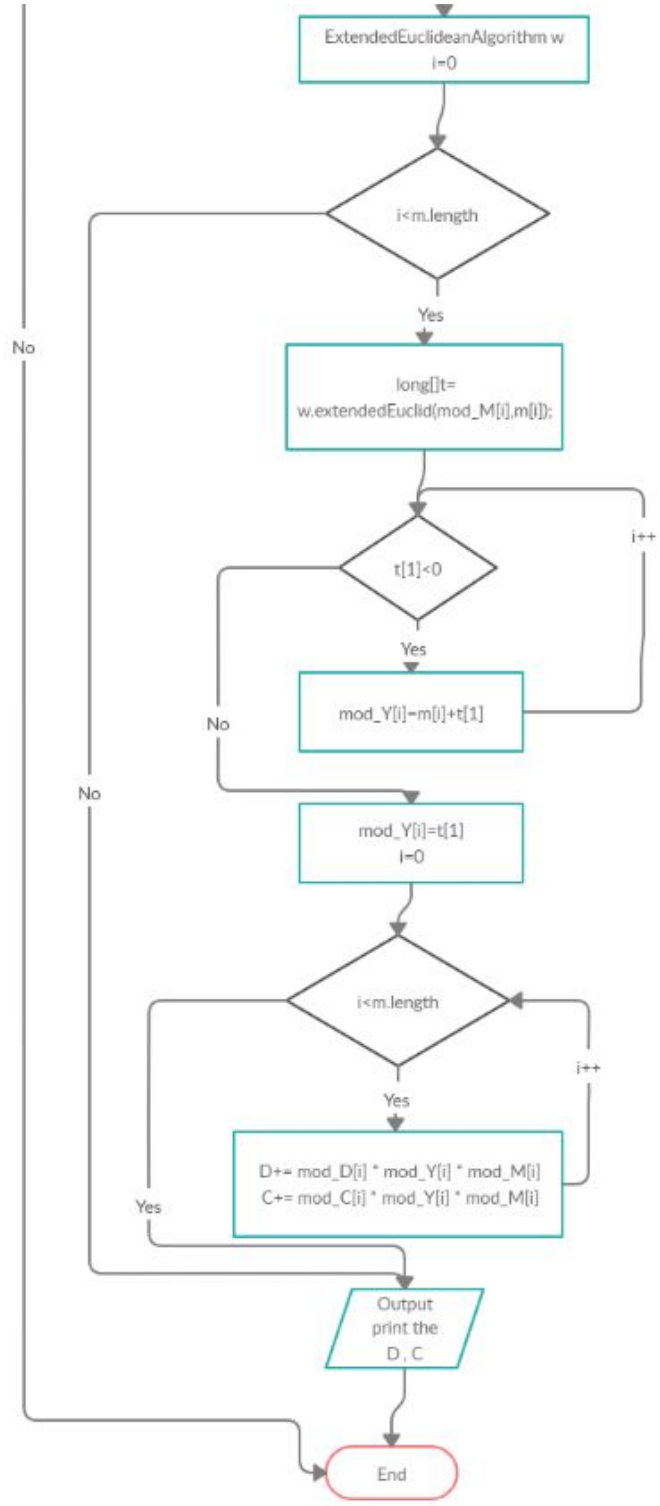

Start
long a,b,m
ans[3]=0
long x=0
if b=0
No
ans = extendedEuclid(b,a%b)
x = ans[1]
ans[1] = ans[2]
ans[2] = x - (a/b)*ans[2]
Yes
ans[0] = a
ans[1] = 1
ans[2] = 0
return ans

Problem 3 :

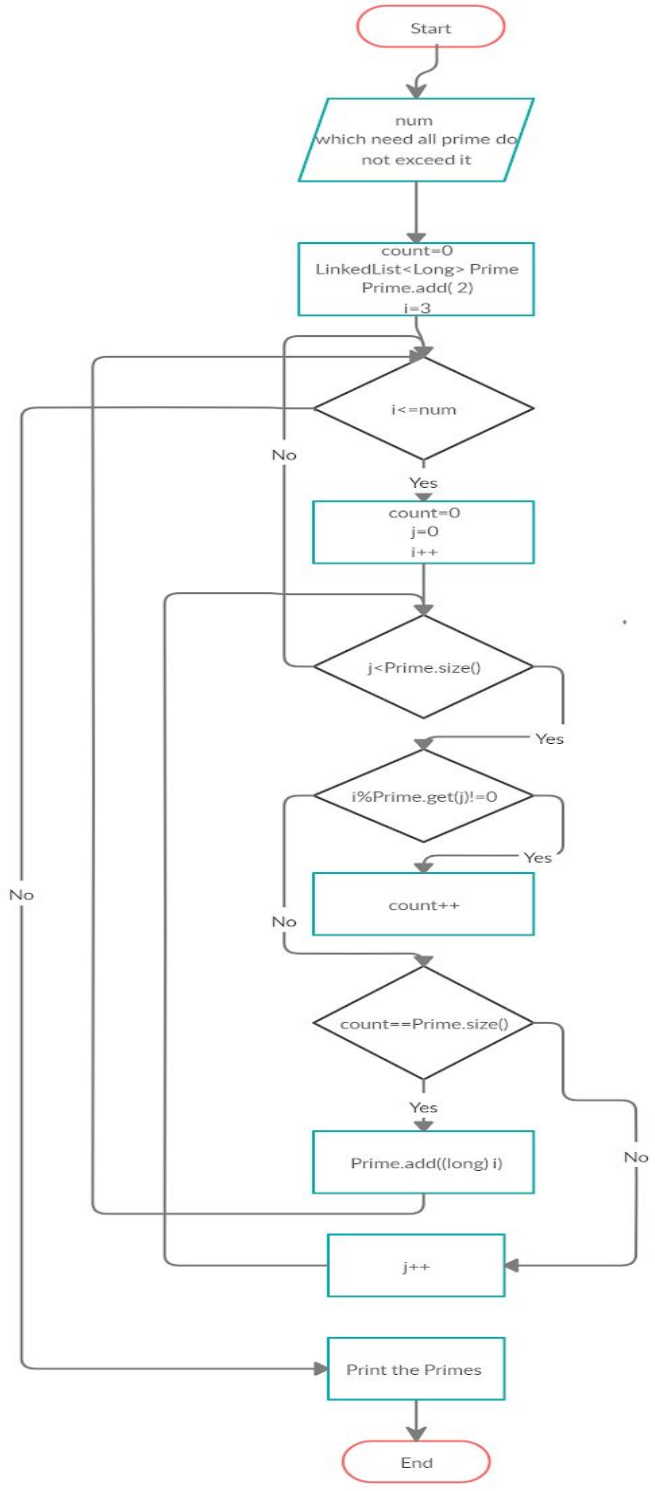
1)


Start
Input
long[] m
long A
long B
M = 1
D = 0
C = 0
mod_M[m.length]
i=0
i<m.length
Yes
M=M*m[i]
i++
No
(A<M)&&(B<M)
Yes
mod_A[m.length]
mod_B[m.length]
mod_C[m.length]
mod_D[m.length]
mod_Y[m.length]
i=0
i<m.length
Yes
i++


(A<M)&&(B<M)
Yes
mod_A[m.length]
mod_B[m.length]
mod_C[m.length]
mod_D[m.length]
mod_Y[m.length]
i=0
i<m.length
Yes
i++
mod_A[i]=A % m[i]
mod_B[i]=B % m[i]
mod_C[i]=(mod_A[i]+mod_B[i])%m[i]
mod_D[i]=(mod_A[i]*mod_B[i])%m[i]
mod_M[i]=M / m[i]
No
ExtendedEuclideanAlgorithm w
i=0
i<m.length
Yes
No
long[]t=
w.extendedEuclid(mod_M[i],m[i]);
i++
t[1]<0
Yes
No
mod_Y[i]=m[i]+t[1]


ExtendedEuclideanAlgorithm w
i=0
i<m.length
Yes
No
long[]t=
w.extendedEuclid(mod_M[i],m[i]);
t[1]<0
i++
Yes
No
mod_Y[i]=m[i]+t[1]
No
mod_Y[i]=t[1]
i=0
i<m.length
i++
Yes
D+= mod_D[i] * mod_Y[i] * mod_M[i]
C+= mod_C[i] * mod_Y[i] * mod_M[i]
Yes
Output
print the
D , C
End

Problem 4 :


Start
num
which need all prime do
not exceed it
count=0
LinkedList<Long> Prime
Prime.add(2)
i=3
i<=num
No
Yes
count=0
j=0
i++
j<Prime.size()
Yes
i%Prime.get(j)!=0
Yes
count++
No
No
count==Prime.size()
Yes
Prime.add((long) i)
No
j++
Print the Primes
End

Assumptions and details :

Problem1 :

There are 4 methods to implement Fast Exponentiation, first and second ones which are mentioned in the problem statement their complexity is **O(n)**
But Naive 2 is **better than** Naive 1 as in Naive 1 overflow may happen in large numbers. The screenshots show that in Sample run.

Problem 3 :

```
public void CRT(long [] m,long A,long B){
    long M = 1;
    long D = 0;
    long C = 0;
    long[]mod_M=new long[m.length];
    for (int i=0;i<m.length;i++) {
        M=M*m[i];
    }
    if((A<M)&&(B<M))
    {
        long[]mod_A=new long[m.length];
        long[]mod_B=new long[m.length];
        long[]mod_C=new long[m.length];
        long[]mod_D=new long[m.length];
        long[]mod_Y=new long[m.length];
        for(int i=0;i<m.length;i++) {
            mod_A[i]=A % m[i];
            mod_B[i]=B % m[i];
            mod_C[i]=(mod_A[i]+mod_B[i])%m[i];
            mod_D[i]=(mod_A[i]*mod_B[i])%m[i];
            mod_M[i]=M / m[i];
        }
        ExtendedEuclideanAlgorithm w=new ExtendedEuclideanAlgorithm();
        for (int i=0;i<m.length;i++) {
           long[]t= w.extendedEuclid(mod_M[i],m[i]);
          if(t[1]<0) {
              mod_Y[i]=m[i]+t[1];
          }
          else {
              mod_Y[i]=t[1];
          }
        }
```

```
    ExtendedEuclideanAlgorithm w=new ExtendedEuclideanAlgorithm();
    for (int i=0;i<m.length;i++) {
       long[]t= w.extendedEuclid(mod_M[i],m[i]);
      if(t[1]<0) {
           mod_Y[i]=m[i]+t[1];
       }
       else {
           mod_Y[i]=t[1];
       }
    }
    for (int i=0;i<m.length;i++) {
        D=D + mod_D[i] * mod_Y[i] * mod_M[i];
        C=C + mod_C[i] * mod_Y[i] * mod_M[i];
    }
    D=D%M;
    C=C%M;
}
System.out.println("THe Result of C = A+B is "+C+"    In Domain Zm1 *Zm2 *. . . * Zmn.");
System.out.println("THe Result of D = A*B is "+D+"    In Domain Zm1 *Zm2 *. . . * Zmn.");
System.out.println("THe Result of C = A+B is "+ (A+B) +"    In Domain ZM.");
System.out.println("THe Result of D = A*B is "+ (A*B) +"    In Domain ZM.");
```

The execution time for Domain Zm1 ∗Zm2 ∗. . . ∗ Zmn, is o(n^2) .

The execution time for Domain ZM, is o(1) .

As the above code..

Problem 4 :

Use long data type which takes " 64-bit two's complement integer" .which have Somewhat large execution time so use the second implementation which use bool data type which takes 1 byte. can optimize space to n/8 by using individual bits of an integer. We create an integer array of size n/64. the size of array is reduced to n/64 from n/2 (Assuming that integers take 32 bits)

```
public void create_prime(long num) {
    LinkedList<Long> Prime=new LinkedList<>();
    Prime.add((long) 2);
    long count=0;
    for(int i=3;i<=num;i++) {
        count=0;
       for(int j=0;j<Prime.size();j++) {
           if(i%Prime.get(j)!=0){
               count++;
           }
           if(count==Prime.size()) {
               Prime.add((long) i);
               break;
           }
       }
    }
    for (int i=0;i<Prime.size();i++) {
        System.out.println(Prime.get(i));
    }
}
```

```
public static void create_prime2(int n)
{
    boolean prime[]=new boolean[n / 2];
    Arrays.fill(prime,  val: false);
    for (int i = 3 ; i * i < n; i += 2)
    {

        if (prime[i / 2] == false)
            for (int j = i * i; j < n; j += i * 2)
                prime[j / 2] = true;
    }
    System.out.print("2 ");
    for (int i = 3; i < n ; i += 2)
        if (prime[i / 2] == false)
            System.out.print(i + " ");
}
```

Sample run:

```
Please choose a Problem
-----------------------
1- Fast Exponentiation
2- Extended Euclidean Algorithm
3- Chinese Remainder Theorem
4- Prime Number Generation

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

Problem1

The output of first way may be wrong according to **overflow**

```
------------------------------------------------------------------
1
Please choose a way
1- Naive1
2- Naive2
3- Iterative
4- Recursion

1
Please Enter a
50
Please Enter b
20
Please Enter m
3
Result = 2
------------------------------------------------------------------
```

```
1
Please choose a way
1- Naive1
2- Naive2
3- Iterative
4- Recursion

2
Please Enter a
50
Please Enter b
20
Please Enter m
3
Result = 1
```

```
------------------------------------------------------------------
1
Please choose a way
1- Naive1
2- Naive2
3- Iterative
4- Recursion

3
Please Enter a
50
Please Enter b
20
Please Enter m
3
Result = 1
```

```
1
Please choose a way
1- Naive1
2- Naive2
3- Iterative
4- Recursion

4
Please Enter a
50
Please Enter b
20
Please Enter m
3
Result = 1
```

Problem2

```
==================================================
Please choose a Problem
==================================================
2
Please Enter a
30
Please Enter b
20
GCD = 10
S = 1
T = -1
```

```
--------------------------------------------------
Please choose a Problem
==================================================
2
Please Enter a
10000
Please Enter b
237665
GCD = 5
S = -4682
T = 197
==================================================
```

Problem3

```
Please choose a Problem
----------------------
1- Fast Exponentiation
2- Extended Euclidean Algorithm
3- Chinese Remainder Theorem
4- Prime Number Generation

======================================================================
3
Please Enter number of remainders
4
Please Enter the remainders
99
98
97
95
Please Enter a
123684
Please Enter b
413456
THe Result of C = A+B is 537140    In Domain Zm1 ?Zm2 ?. . . ? Zmn.
THe Result of D = A*B is 88247874    In Domain Zm1 ?Zm2 ?. . . ? Zmn.
THe Result of C = A+B is 537140    In Domain ZM.
THe Result of D = A*B is 51137891904    In Domain ZM.
```

Problem4

```
------------------------------------
Please choose a Problem
====================================
4
Please Enter a number
100
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
====================================
```