Phần I

1. Xây dựng máy Turing M2 thực hiện phép trừ 1 của số nhị phân.

Máy Turing M2:

M2 =(K_2 , $\sum_2$, ɛ, s)

Trong đó:

K_2 = {s,h,q}, s là bắt đầu, "h" là trạng thái dừng

$\sum_2$ = {0, 1,□, ■}

Bảng hàm chuyển ɛ:

STT	t	k	ɛ(t,k)=(k1,t1,{←/→/— })
1	s	0	(0, s, →)
2	s	1	(1, s, →)
3	s	□	(□, s, →)
4	s	■	(■, q, ←)
5	q	0	(0, q, ←)
6	q	1	(0, h, →)
7	q	□	(□,h, ←)

2. Xây dựng máy Turing M3 thực hiện việc thay tất cả các số 0 trong một dãy nhị phân thành các số 1 và ngược lại. Ví dụ: 01001 10110.

Máy Turing M3:

M3 =(K_3 , $\sum_3$, ɛ, s)

Trong đó:

K_3 = {s,h}, s là bắt đầu, "h" là trạng thái dừng

$\sum_3$ = {0, 1,□, ■}

Bảng hàm chuyển ɛ:

STT	t	k	ɛ(t,k)=(k1,t1,{←/→/— })
1	s	0	(1,s,→)
2	s	1	(0,s,→)
3	s	□	(0,□,→)
4	s	■	(■,h,→)

Phần II

1. Multiplication: axb

   mul(0, a) = a = $P_1^1(a)$ (1.1)

   mul(S(a), b) = mul(a, b) + a

   = add(mul(a, b), a) = add(P_1^3(mul(a,b),a,b), a) (1.2)

   Từ (1.1) & (1.2) => đpcm

2. Exponentiation: a^b

   exp(0, a) = 1 = const (2.1)

   exp(S(b), a) = mul(exp(b,a), a)

   = mul(P_1^3(mul(exp(b,a), a), a, b) (2.1)

   Từ (2.1) & (2.2) => đpcm

3. Factorial a! : 0! = 1, a'! = a!xa'

   fac(0) = 1 = const (3.1)

   fac(x+1) = (x+1)*fac(x)

   = mul(S(x), fac(x)) (3.2)

   Từ (3.1) & (3.2) => đpcm

Phần III

1.

input : Hai số int a,b

output : Ước chung lớn nhất

```
int ucln(int a, int b) {
   if (b == 0) return a;
   return ucln(b, a % b);
}
```

2.

input : Số int n,

output: Các số int nguyên tố từ 2 đến n

```
Nhập N
khởi tạo mảng bool check[N + 1]

lặp (i = 2; i <= N; i++){
        check[i] = tru
}

lặp (i = 2; i < N; i++){
        Nếu check[i] == true
                lặp (j = 2*i; j <= N; j+=i){
                        check[j] = flase
                }
        }
}
```

```
lặp (i = 2; i <=N; i++){
        Nếu check[i] == true{
                In ra i
        }
}
```

3.

input :
        list   : danh sách đỉnh
        matrix   : danh sách kề
output đỉnh có nhiều cạnh nhất

```
list[]                          // danh sách đỉnh
matrix[]                        // danh sách kề : matrix[u] = {v, x, y, z}  x, y, z là các đỉnh kề với u
res = 0                         // result
for u in list:
        res = len(matrix[u]) > len(matrix[res]) ? u : res
output res
lặp (i = 0; i < list.size; i++){
        Nếu độ dài(matrix[ list [ i ] ] ) > độ dài (matrix [ res ] ){
                res = list[i]
        }
}
```

in ra : res

```
################################################
code phần III
                            bài 1
def gcd(a, b) :
    if b == 0:
        return a
    return gcd(b, a%b)


a = int(input("Enter a : "))
b = int(input("Enter b : "))

print("GCD : ", gcd(a, b))
                            bài 2

prime = {}

def era(n):
    for i in range(n+1):
        prime[i] = 1
    prime[0] = prime[1] = 0

    for i in range(n+1):
        if prime[i] == 1:
            print(i)
            for j in range(i+i, n+1, i):
                prime[j] = 0
n = int(input("Enter n : "))
era(n)
                            baif 3
lists = [1, 2, 3, 4, 5]
matrix = [[2,3], [1], [4], [1,2,3,5], []]

res = lists[0]
for u in lists:
    if len(matrix[u-1]) > len(matrix[res-1]):
        res = u
print(res)
```