Batch: A3 Roll No.: 16010121045

Experiment No.

Title: Solving discrete logarithm problem for elliptic curves

Objective:

Understating infeasibility of breaking ECC

Expected Outcome of Experiment:

CO	Outcome
CO2	Demonstrate various cryptographic algorithms for securing systems

Books/ Journals/ Websites referred:

Abstract:-

(Elliptic curves, Define prime curves and binary curves)

Related Theory: -

Define discrete logarithm problem for elliptic curves.:

Consider the equation $Q = kP$ where $Q, P \in EP(a, b)$ and $k < p$.
It is relatively easy to calculate Q given k and P, but it is hard to determine k given Q and P. This is called the discrete logarithm problem for elliptic curves.

Example:
Consider the group $E_{23}(9,17)$.
This group is defined by the equation : $y^2 \bmod 23 = (x^3 + 9x + 17) \bmod 23$.

What is the discrete logarithm k of Q = (4, 5) to the base P = (16, 5)?

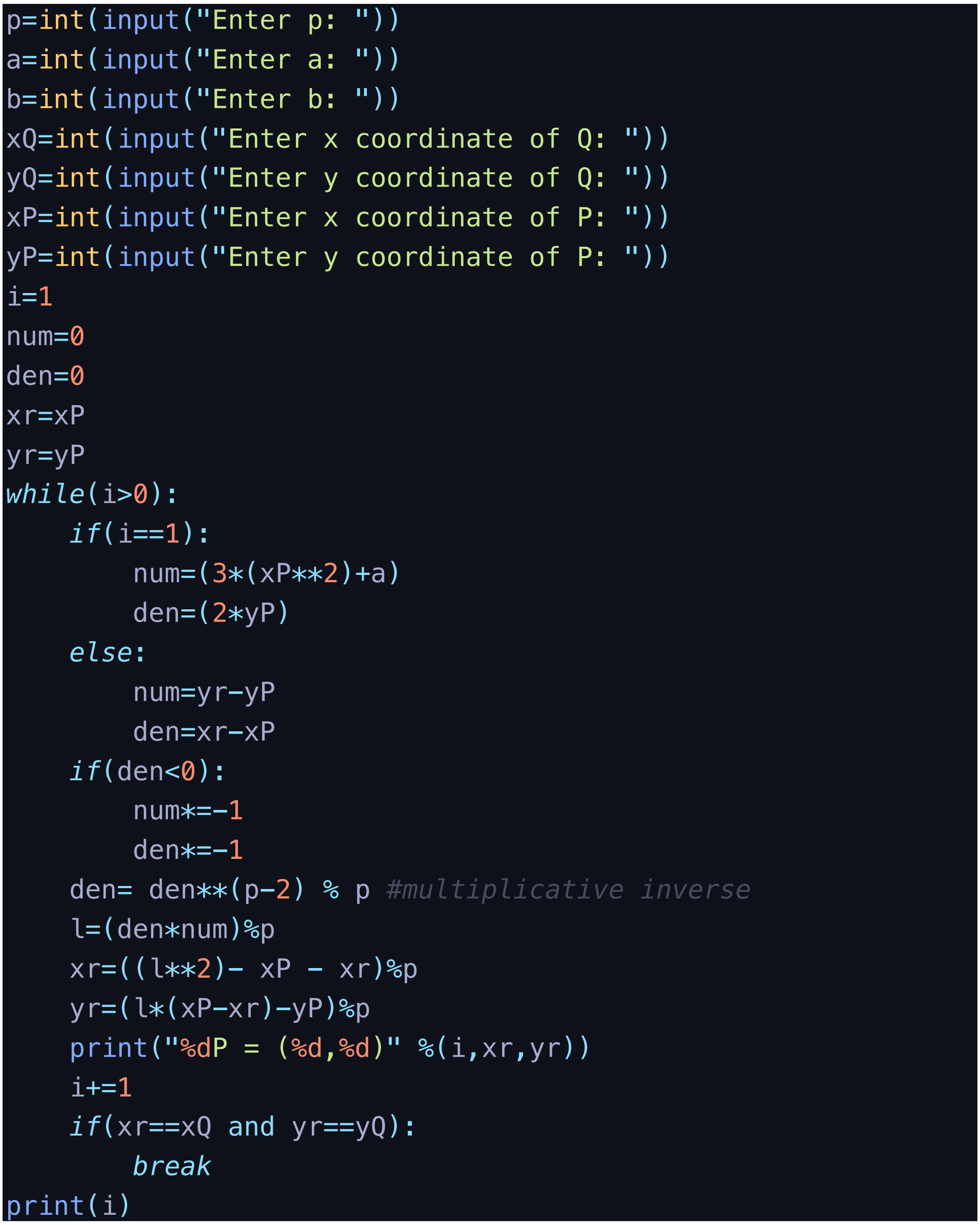
The brute-force method is to compute multiples of P until Q is found. Thus,
P = (16,5);
2P = (20, 20);
3P = (14, 14);
4P = (19, 20);
5P = (13, 10);
6P = (7, 3);
7P = (8, 7);
8P = (12, 17);
9P = (4, 5) =Q

Because 9P = (4, 5) = Q, the discrete logarithm Q = (4, 5) to the base P = (16, 5) is k = 9.
In a real application, k would be so large as to make the Brute Force approach infeasible.

Program:

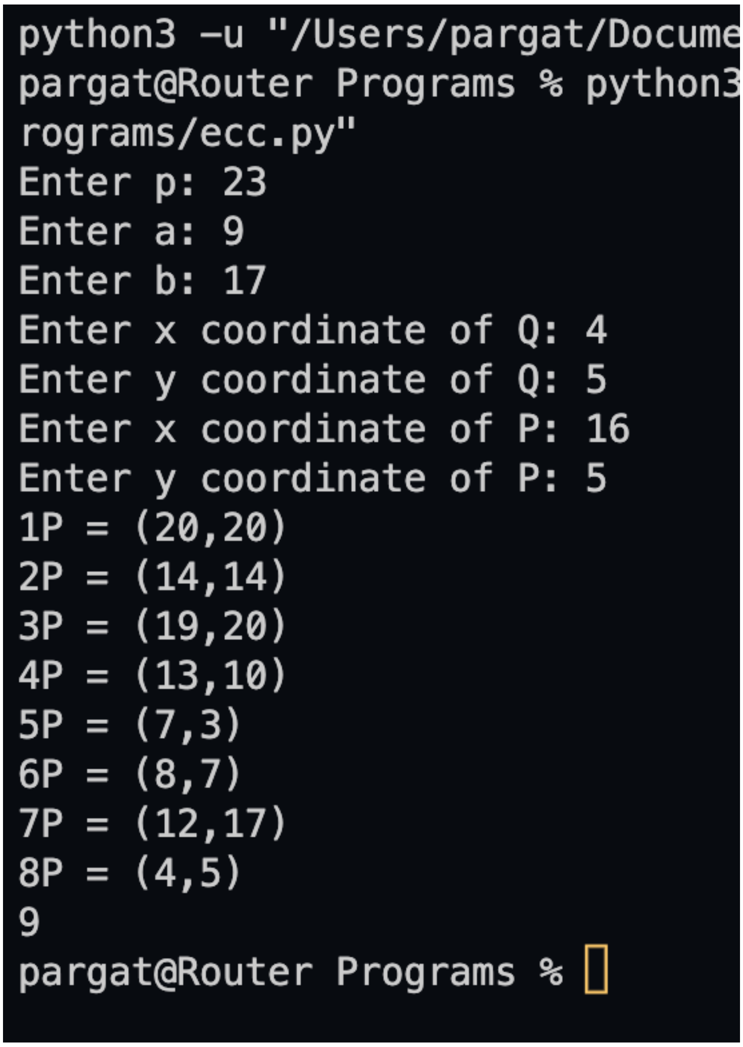
Given any points P(xp,yp) and Q(xq,yq), compute K, such that

Q=K*P.



```
p=int(input("Enter p: "))
a=int(input("Enter a: "))
b=int(input("Enter b: "))
xQ=int(input("Enter x coordinate of Q: "))
yQ=int(input("Enter y coordinate of Q: "))
xP=int(input("Enter x coordinate of P: "))
yP=int(input("Enter y coordinate of P: "))
i=1
num=0
den=0
xr=xP
yr=yP
while(i>0):
    if(i==1):
        num=(3*(xP**2)+a)
        den=(2*yP)
    else:
        num=yr-yP
        den=xr-xP
    if(den<0):
        num*=-1
        den*=-1
    den= den**(p-2) % p #multiplicative inverse
    l=(den*num)%p
    xr=((l**2)- xP - xr)%p
    yr=(l*(xP-xr)-yP)%p
    print("%dP = (%d,%d)" %(i,xr,yr))
    i+=1
    if(xr==xQ and yr==yQ):
        break
print(i)
```

Output Screenshots:



```
python3 -u "/Users/pargat/Docume
pargat@Router Programs % python3
rograms/ecc.py"
Enter p: 23
Enter a: 9
Enter b: 17
Enter x coordinate of Q: 4
Enter y coordinate of Q: 5
Enter x coordinate of P: 16
Enter y coordinate of P: 5
1P = (20,20)
2P = (14,14)
3P = (19,20)
4P = (13,10)
5P = (7,3)
6P = (8,7)
7P = (12,17)
8P = (4,5)
9
pargat@Router Programs % 
```

Conclusion:-

Successfully implemented Elliptical Curve Cyptography.

Postlab:

1. **Compare RSA and ECC.**

RSA	ECC
A well-established method of public-key cryptography.	A newer public-key cryptography method compared to RSA.
Works on the principle of the prime factorization method.	Works on the mathematical representation of elliptic curves.
RSA can run faster than ECC thanks to its simplicity.	ECC requires bit more time as it's complex in nature.
RSA has been found vulnerable and is heading towards the end of its tenure.	ECC is more secure than RSA and is in its adaptive phase. Its usage is expected to scale up in the near future.
RSA requires much bigger key lengths to implement encryption.	ECC requires much shorter key lengths compared to RSA.

2. **List various applications of ECC.**
    - ECC is among the most commonly used implementation techniques for digital signatures.
    - Cryptocurrencies apply the Elliptic Curve Digital Signature Algorithm (ECDSA) specifically in signing transactions.
    - It is used in different parts of the SSL standard utilizing signing SSL certificates with ECDSA instead of RSA.

3. **Significance of discrete logarithm problem solution in ECDH cryptanalysis process.**

    Another advantage of such a cryptosystem lies in the difficulty of solving the Elliptic Curve Discrete Log Problem (ECDLP). If an elliptic curve is chosen with some care, the ECDLP is believed to be infeasible, even with today's computational power. Using elliptic curves presents a great advantage in a few areas. For instance, compared to RSA cryptosystems, elliptic curve based systems require less memory; for example, a key size of 4096 bits for RSA gives the same level of security as 313 bits in an elliptic curve system .