CRYPTOGRAPHY AND NETWORK SECURITY

LAB 7: DIFFIE HELMAN KEY EXCHANGE ALGORITHM IMPLEMENTATION

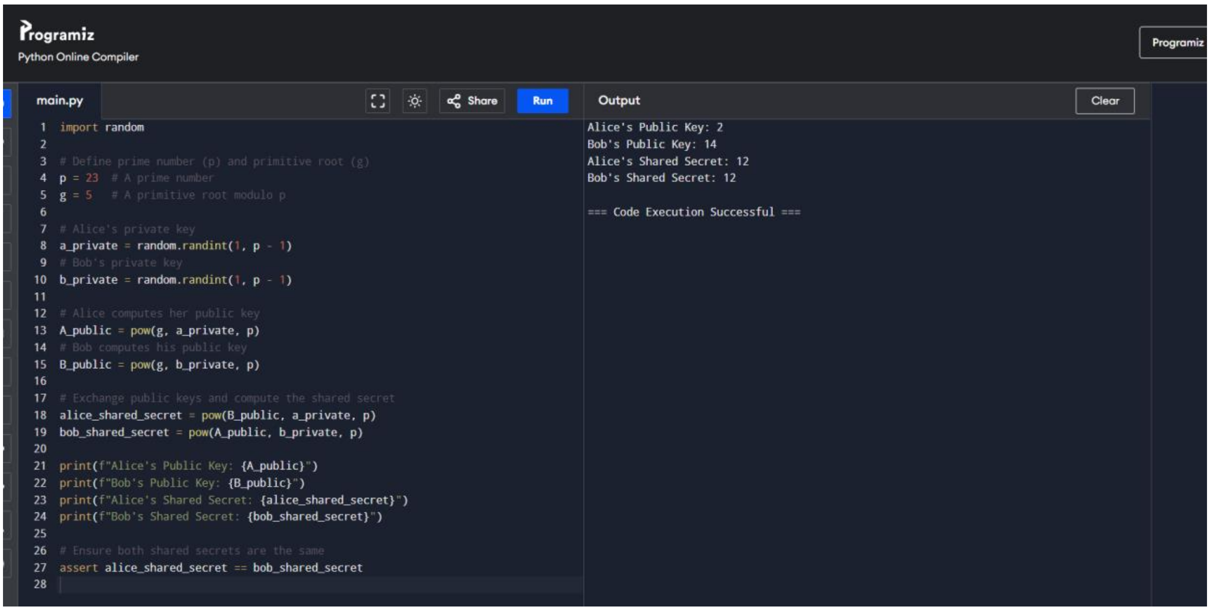
NAME : OM SUBRATO DEY

REGISTER NUMBER : 21BAI1876

CODE:

```
import random
# Define prime number (p) and primitive root (g)
p = 23  # A prime number
g = 5   # A primitive root modulo p

# Alice's private key
a_private = random.randint(1, p - 1)
# Bob's private key
b_private = random.randint(1, p - 1)

# Alice computes her public key
A_public = pow(g, a_private, p)
# Bob computes his public key
B_public = pow(g, b_private, p)

# Exchange public keys and compute the shared secret
alice_shared_secret = pow(B_public, a_private, p)
bob_shared_secret = pow(A_public, b_private, p)

print(f"Alice's Public Key: {A_public}")
print(f"Bob's Public Key: {B_public}")
print(f"Alice's Shared Secret: {alice_shared_secret}")
print(f"Bob's Shared Secret: {bob_shared_secret}")

# Ensure both shared secrets are the same
assert alice_shared_secret == bob_shared_secret
```

OUTPUT:



```python
import random

# Define prime number (p) and primitive root (g)
p = 23  # A prime number
g = 5   # A primitive root modulo p

# Alice's private key
a_private = random.randint(1, p - 1)
# Bob's private key
b_private = random.randint(1, p - 1)

# Alice computes her public key
A_public = pow(g, a_private, p)
# Bob computes his public key
B_public = pow(g, b_private, p)

# Exchange public keys and compute the shared secret
alice_shared_secret = pow(B_public, a_private, p)
bob_shared_secret = pow(A_public, b_private, p)

print(f"Alice's Public Key: {A_public}")
print(f"Bob's Public Key: {B_public}")
print(f"Alice's Shared Secret: {alice_shared_secret}")
print(f"Bob's Shared Secret: {bob_shared_secret}")

# Ensure both shared secrets are the same
assert alice_shared_secret == bob_shared_secret

```

```
Alice's Public Key: 2
Bob's Public Key: 14
Alice's Shared Secret: 12
Bob's Shared Secret: 12

=== Code Execution Successful ===
```