Sprawozdanie z Laboratorium 2 – DH

1. Screenshot z aplikacji

```python
def main():  1 usage  new *
    prime_list = find_primes()
    n = random.choice(prime_list)
    g = find_primitive_root(n)

    Y = 0
    X = 0

    print(f"Liczba n: {n}")
    print(f"liczba g: {g}")

    x, X = A(g,n)
    y, Y = B(g,n)

    kA = A_k(x,Y,n)
    kB = B_k(x,Y,n)

    print(f"Klucz k: {kA} oraz {kB}")
```

```python
def find_primes():  1 usage  new *
    primes = [p for p in range(1001, 10000, 2) if isprime(p)]
    return primes

def find_primitive_root(n):  1 usage  new *
    factors = list(primerange( a: 2, n-1))
    for g in range(2, n):
        if all(pow(g, (n-1)//factor, n) != 1 for factor in factors):
            return g
    return None

def A(g,n):  1 usage  new *
    x = random.randint( a: 1,  b: 9999)
    X = pow(g, x, n)
    return x, X

def A_k(x,Y,n):  1 usage  new *
    k = pow(Y, x, n)
    return k

def B(g,n):  1 usage  new *
    y = random.randint( a: 1,  b: 9999)
    Y = pow(g, y, n)
    return y, Y
def B_k(x,Y,n):  1 usage  new *
    k = pow(Y, x, n)
    return k
```

2. Przeanalizuj, czy są jakieś ograniczenia dla użytych parametrów?

Liczba n powinna być wystarczająco duża, liczba g musi być większa od 1 i mniejsza od n

3. Jakie dane można podsłuchać i czy możesz zaproponować jakiś schemat ataku?

Algorytm DH jest podatny na ataki pośrodku (ang. intruder-in-the-middle/man-in-the-middle attack). Atakujący ma możliwość przechwytywania komunikatów przesyłanych pomiędzy A i B. Schemat ataku wyglądałby następująco:

1. A wysyła B wiadomość X
2. Atakujący przechwytuje komunikat i przesyła do B wartość Xi (wartość obliczona przez atakującego według wzoru $Xi=g^{xi} \bmod n$)
3. B wysyła do A wiadomość Y
4. Atakujący przechwytuje komunikat i przesyła do B wartość Yi (wyliczoną tak samo jak Xi)
5. A i B obliczają tajny klucz otrzymując różne wyniki. Te same wyniki mogą być wyliczone przez atakującego