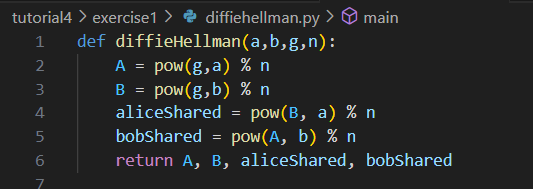
Exercise 1

* Task 1

  
  
The intermediate values can be computed by making a python function.   
  
A black screen with blue text

Description automatically generated

* Task 2

For task 2 we have g = 5, p = 47 X= 99 (not modulo) and Y=23 (has been moduloed)

99 mod 47 = 5 so use X as 5.

A screen shot of a computer program

Description automatically generated

* Task 3, using Key as 23.

A computer screen shot of text

Description automatically generated

A black screen with blue text

Description automatically generated

Exercise 2

To design a protocol that allows Alice (A), Bob (B), and Charlie (C) to exchange a single symmetric key *K* using an extended Diffie-Hellman (DH) key exchange, we aim to meet the following conditions:

Key Features:

Only A, B, and C access the key.

Key integrity is verified by all.

Minimize the number of messages.

Proposed protocol:

Public variables – Just like DH we keep prime p and generator g. These values are public and can be reused.

Private variables – All 3, A, B and C select a random private a,b and c value. This value is not shared and kept only with them.

Intermediate values –

Alice computes – A = g^a mod p and sends it to Bob

Bob computes – B = g^b mod p and sends it to Charlie

Charlie computes – C = g^c mod p and sends it to Alice

Key Exchange –

Alice computes Kca = C^a mod p and sends it to Bob

Bob computes Kab = A^b mod p and sends it to Charlie

Charlie computes Kbc = B^c mod p and sends it to Alice

Again

Alice computes Kcab = (C^a)^b mod p which is C^ab mod p which is g^cab mod p

Bob computes Kabc = g^abc mod p

Charlie computes Kbca = g^bca mod p

This way every one gets the same key computed. The total number of messages is 6 this can be reduced by involving a trusted party like ‘Trent’ who would create a common key and send it to everyone.