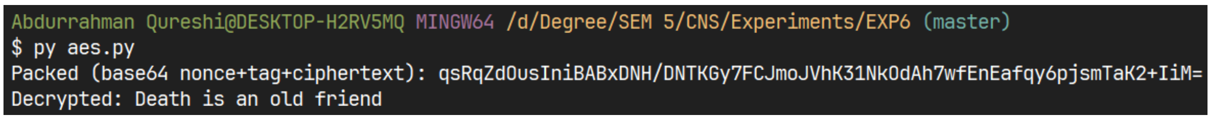
Aim: Implement AES technique to encrypt and decrypt message, Use java or python code

CODE:

```
from Crypto.Cipher import AES
from Crypto.Random import get_random_bytes
import base64

IV_LEN = 12  # 12-byte nonce for GCM
KEY_LEN = 32 # 32 bytes = 256-bit key

def generate_key():
    return get_random_bytes(KEY_LEN)

def encrypt(plaintext: str, key: bytes) -> str:
    nonce = get_random_bytes(IV_LEN)
    cipher = AES.new(key, AES.MODE_GCM, nonce=nonce)
    ciphertext, tag = cipher.encrypt_and_digest(plaintext.encode("utf-8"))
    # Pack as: nonce || tag || ciphertext (then base64)
    packed = nonce + tag + ciphertext
    return base64.b64encode(packed).decode("ascii")

def decrypt(b64_packed: str, key: bytes) -> str:
    packed = base64.b64decode(b64_packed)
    nonce = packed[:IV_LEN]
    tag = packed[IV_LEN:IV_LEN+16]
    ciphertext = packed[IV_LEN+16:]
    cipher = AES.new(key, AES.MODE_GCM, nonce=nonce)
    plaintext = cipher.decrypt_and_verify(ciphertext, tag)
    return plaintext.decode("utf-8")

if __name__ == "__main__":
    key = generate_key()
    msg = "Death is an old friend"
    b64 = encrypt(msg, key)
    print("Packed (base64 nonce+tag+ciphertext):", b64)
    print("Decrypted:", decrypt(b64, key))
```

OUPTUT:



```
Abdurrahman Qureshi@DESKTOP-H2RV5MQ MINGW64 /d/Degree/SEM 5/CNS/Experiments/EXP6 (master)
$ py aes.py
Packed (base64 nonce+tag+ciphertext): qsRqZdOusIniBABxDNH/DNTKGy7FCJmoJVhK31NkOdAh7wfEnEafqy6pjsmTaK2+IiM=
Decrypted: Death is an old friend
```