j2tham

1a) 1. Uses python random module which is only pseudorandom. I would use the secrets or sympy modules instead

2. Cipher block mode is ECB, which is less secure than other styles such as CBC. I would use CBC

3. provides n, e, c_1 and c_2 in the JSON file. I would encrypt n,e,c_1 in a second file and send it to myself

4. e was not selected specifically for RSA, (where it should be $1 < e < \phi(n)$ and that $\gcd(e, \phi(n)) = 1$). I would ensure that e is chosen within these specs and reselect e if necessary

1b)

```
# TODO
phi = totient(n)
d = mod_inverse(e,phi)
aes_key_int = pow(c_1,d,n)
aes_key = aes_key_int.to_bytes((aes_key_int.bit_length()+7)//8,byteorder='big')
cipher = Cipher(algorithms.AES(aes_key),modes.ECB())
decryptor = cipher.decryptor()
padded = decryptor.update(c_2)+decryptor.finalize()
unpadder = padding.PKCS7(128).unpadder()
plaintext = unpadder.update(padded)+unpadder.finalize()
# write the decrypted assignment to a file
with open("assignment_out.pdf", 'wb') as fh:
    fh.write(plaintext)
```

1c) 96106