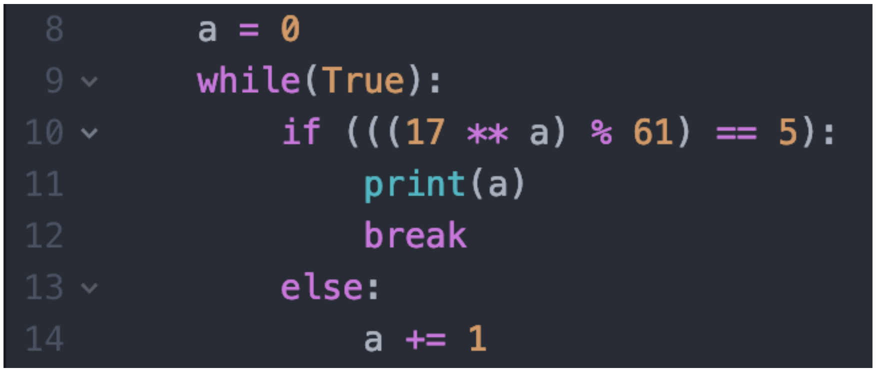
Authors: Anders Shenholm and Riaz Kelly

Diffie-Hellman:

1) K = 15

2) We can calculate the encryption key by determining a and b, the random numbers generated by Alice and Bob. We can achieve this with brute force, testing integers that could satisfy $46 = 17^{a} \% 61$ and $5 = 17^{b} \% 61$via a python program. We tested the integers in ascending order until we found a = 26 and b = 14. From here we calculate $K = 5^{26} \% 61 = 15$. Our python program would have taken much longer if the integers were much larger.

3) We would not have been able to find a and b as easily as we did if we were working with large numbers. If the a and b were many orders of magnitude larger, we wouldn't have been able to track them down efficiently by incrementing and checking numbers one at a time.



```
    a = 0
    while(True):
        if (((17 ** a) % 61) == 5):
            print(a)
            break
        else:
            a += 1
```

RSA:

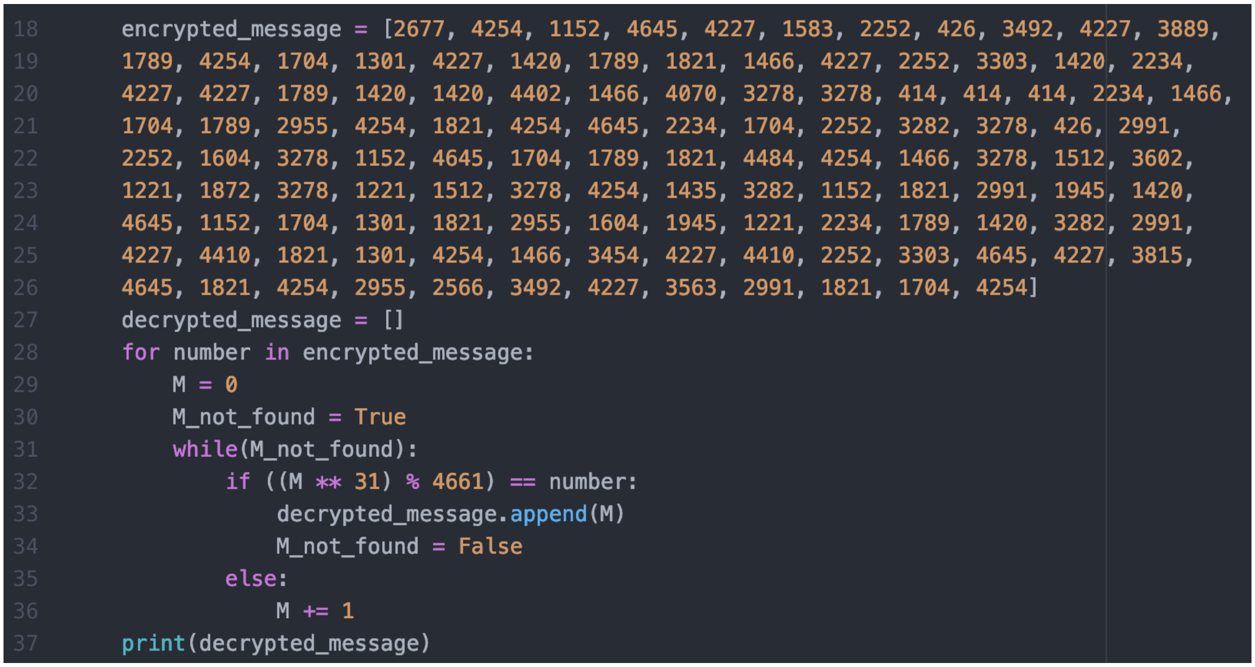
1)
[68, 101, 97, 114, 32, 66, 111, 98, 44, 32, 67, 104, 101, 99, 107, 32, 116, 104, 105, 115, 32, 111, 117, 116, 46, 32, 32, 104, 116, 116, 112, 115, 58, 47, 47, 119, 119, 119, 46, 115, 99, 104, 110, 101, 105, 101, 114, 46, 99, 111, 109, 47, 98, 108, 111, 103, 47, 97, 114, 99, 104, 105, 118, 101, 115, 47, 50, 48, 49, 55, 47, 49, 50, 47, 101, 45, 109, 97, 105, 108, 95, 116, 114, 97, 99, 107, 105, 110, 103, 95, 49, 46, 104, 116, 109, 108, 32, 89, 105, 107, 101, 115, 33, 32, 89, 111, 117, 114, 32, 102, 114, 105, 101, 110, 100, 44, 32, 65, 108, 105, 99, 101]

Translated:

Dear Bob, Check this out. https://www.schneier.com/blog/archives/2017/12/e-mail_tracking_1.html Yikes! Your friend, Alice

2) We can calculate Alice's message with brute force with the equation $E(P, M) = M^{31} \% 4661$. We applied this to each piece of the message using another python program. This gave us the decrypted message in ASCII format.

3) Similar to part one, we wouldn't have been able to do this with large integers. Our python program would have taken much longer to run and perhaps never reach M.



```
    encrypted_message = [2677, 4254, 1152, 4645, 4227, 1583, 2252, 426, 3492, 4227, 3889,
    1789, 4254, 1704, 1301, 4227, 1420, 1789, 1821, 1466, 4227, 2252, 3303, 1420, 2234,
    4227, 4227, 1789, 1420, 1420, 4402, 1466, 4070, 3278, 3278, 414, 414, 414, 2234, 1466,
    1704, 1789, 2955, 4254, 1821, 4254, 4645, 2234, 1704, 2252, 3282, 3278, 426, 2991,
    2252, 1604, 3278, 1152, 4645, 1704, 1789, 1821, 4484, 4254, 1466, 3278, 1512, 3602,
    1221, 1872, 3278, 1221, 1512, 3278, 4254, 1435, 3282, 1152, 1821, 2991, 1945, 1420,
    4645, 1152, 1704, 1301, 1821, 2955, 1604, 1945, 1221, 2234, 1789, 1420, 3282, 2991,
    4227, 4410, 1821, 1301, 4254, 1466, 3454, 4227, 4410, 2252, 3303, 4645, 4227, 3815,
    4645, 1821, 4254, 2955, 2566, 3492, 4227, 3563, 2991, 1821, 1704, 4254]
    decrypted_message = []
    for number in encrypted_message:
        M = 0
        M_not_found = True
        while(M_not_found):
            if ((M ** 31) % 4661) == number:
                decrypted_message.append(M)
                M_not_found = False
            else:
                M += 1
    print(decrypted_message)
```