CS236 HW6 – Teng Xu

1) (2 pts) Solve 3p+11≡23 (mod 26) using subtraction and division of congruence by a number. Why are you allowed to do this?

Hw 6

1). $3p + 11 \equiv 23 \pmod{26}$

$3p \equiv 12 \pmod{26}$

Reason: If $a \equiv b \pmod{m}$ holds then $c+a \equiv c+b \pmod{m}$

Inverse of 3 mod 26:

$26 = 8 \times 3 + 2$

$3 = 2 + 1$

$2 = 2 \times 1 + 0$

$3 - 2 = 1$

$3 - (26 - 8 \times 3) = 1$

$-26 + 9 \times 3 = 1$

$\therefore$ inverse $= 9$

$3p \cdot 9 \equiv 12 \cdot 9 \pmod{26}$

$p \equiv 108 \pmod{26}$

$p \equiv 4 \pmod{26}$

$p = 4$

Reason: If $a \equiv b \pmod{m}$, then $c \cdot a \equiv c \cdot b \pmod{m}$

$a \cdot \bar{a} \equiv 1 \pmod{m}$

2) (2 pts) Write a function extendedGcd(a,b,) returning gcd and Bézout coefficients of a and b, using extended Euclidian algorithm outlined on slides 31, 34 of Lecture 5. Test it on gcd(5,26), gcd(10,26), gcd(13,26), gcd(26,36), gcd(24,36).

```
In [22]: extendedGcd(5,26)
Out[22]: (1, -5, 1)

In [23]: extendedGcd(10,26)
Out[23]: (2, -5, 2)

In [24]: extendedGcd(13,26)
Out[24]: (13, 1, 2)

In [25]: extendedGcd(26,36)
Out[25]: (2, 7, -5)

In [26]: extendedGcd(24,36)
Out[26]: (12, -1, 1)
```

3) (2 pts) Is it possible for a multiplicative inverse modulo m to be 0? Under what condition does multiplicative inverse of *a* modulo m exist? Write a function returning multInverse(a,m) that checks existence of a multiplicative inverse of *a* modulo m, and if it exists - returns it (otherwise return 0 and print the message: “multiplicative inverse does not exist”).

```
In [57]: multInverse(3,7)
Out[57]: -2

In [58]: multInverse(7,26)
Out[58]: -11
```

4) (2 pts) Write a function encryptAffine(letter, a,b) that uses encryption function f(p)= ap+b (mod 26) and decryptAffine(letter, a,b) that uses corresponding decryption function. Check that encryption function is a bijection, otherwise – output error message. Test your functions on the results of problem 3 for HW5, and encrypting and decrypting back messages: “cryptography is based on modular arithmetic”

```
In [71]: encryptAffine('a', 5,3)
Out[71]: 'd'

In [72]: encryptAffine('b', 5,3)
Out[72]: 'i'

In [73]: encryptAffine('c', 5,3)
Out[73]: 'n'

In [19]: encryptAffine('cryptography is based on modular arithmetic',5,3)
Out[19]: 'nktauvhkdamt rp idpxs vq lvszgdk dkrumlxurn'

In [20]: decryptAffine('nktauvhkdamt rp idpxs vq lvszgdk dkrumlxurn',5,3)
Out[20]: 'cryptography is based on modular arithmetic'
```

5.6.7 :