Math 495R Homework 16

(1) The equation $y^2 = x^3 + x + 1$ gives an *elliptic curve*. For any prime p, we can count the points on the elliptic curve over $\mathbb{Z}_p$ by finding all of the pairs $(x, y) \in \mathbb{Z}_p \times \mathbb{Z}_p$ satisfying $y^2 \equiv x^3 + x + 1 \pmod{p}$. Write code that will compute how many points are on the curve for $p = 5, 7, 11$, and 13.

(2) In Homework 3, you built a function `string2int26` that, given a string of capital letters, returned an integer using the 26 character alphabet `A = 00, B = 01`, etc., as well as a function `int2string26` that reversed the process.

The *affine cipher* interprets each letter as an element of $\mathbb{Z}_{26}$. It has two parameters a and b, with $a, b \in \mathbb{Z}_{26}$, and encrypts a letter X by replacing X with $aX+b \pmod{26}$.

Write two functions `affineencrypt(a, b, plaintext)` and `affinedecrypt(a, b, ciphertext)` that encrypt and decrypt an arbitrary string of capital letters using the affine cipher with parameters `a, b`. Note that we must have $\gcd(a, 26) = 1$ for decryption to work, because decrypting is the same as solving $ax + b \equiv c \pmod{26}$ for x when we know a, b, c, and we can't divide by $a \pmod{26}$ if $\gcd(a, 26) > 1$.

(3) The next alphabet that we will use, which is more suited to mathematical cryptography, has one hundred characters. They are (in order)

00	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19
	!	"	#	$	%	&	’	(	)	*	+	,	-	.	/	0	1	2	3
20	21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39
4	5	6	7	8	9	:	;	<	=	>	?	@	A	B	C	D	E	F	G
40	41	42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59
H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	[
60	61	62	63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79
\	]	^	_	‘	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
80	81	82	83	84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99
p	q	r	s	t	u	v	w	x	y	z	{	|	}	~	¢	£	≪	≫	±

Notice that with this alphabet, together with the convention that leading zeros are written so that every character corresponds to a two-digit number, we can convert any number into a string of characters, and any string of characters into a number. If the number has an odd number of digits (for example, `12345`), we assume that there is a leading zero (that is, we interpret it as `012345`). Thus "Brigham Young University" corresponds to the number

348273717265770057798578710053787386698283738489

and the number

3388621811348811352916

corresponds to the string of characters "`Ax^2+Bx+C=0`". Note that the character corresponding to the number 00 is a space. The first 95 characters in this alphabet are standard (in fact, they are `ASCII` encodings after subtracting 32), and can be typed on any American computer keyboard. The last five are not standard.

Write two functions `string2int100` and `int2string100` that translate strings to integers and integers to strings using this alphabet. Be careful with special characters such as `", ’`, and the backslash.