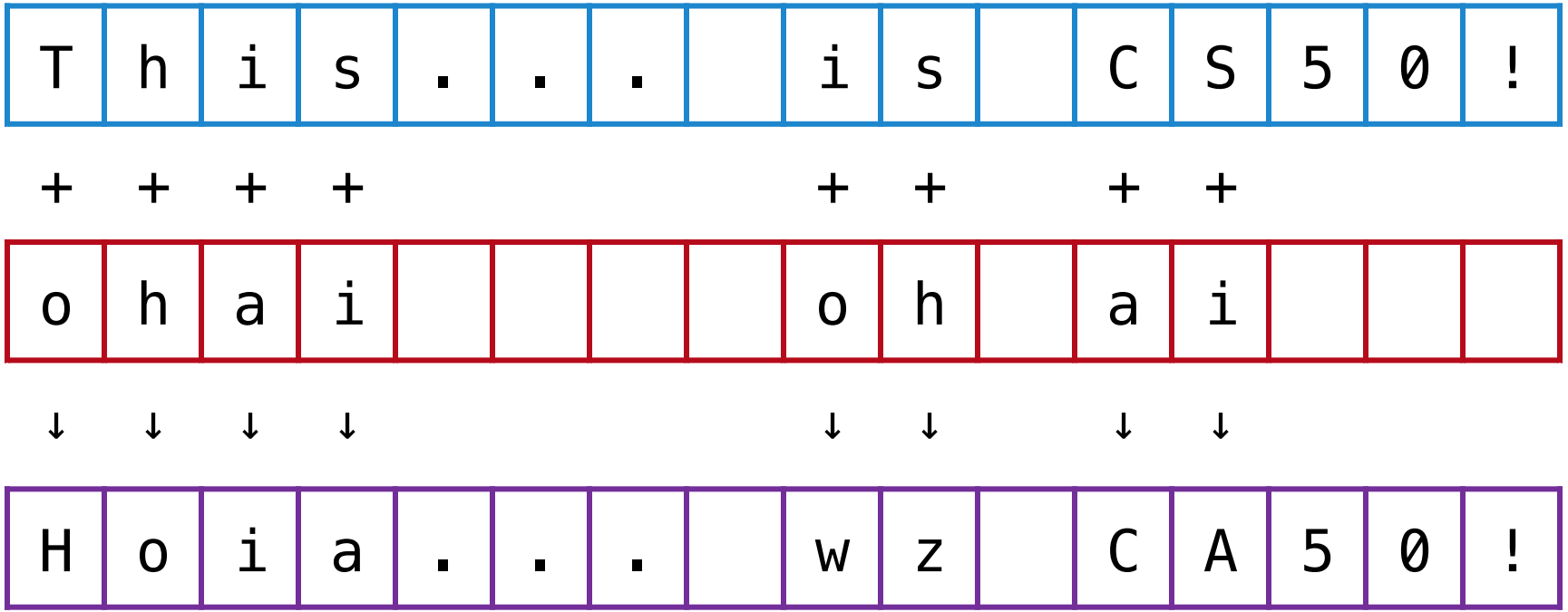
vigenère

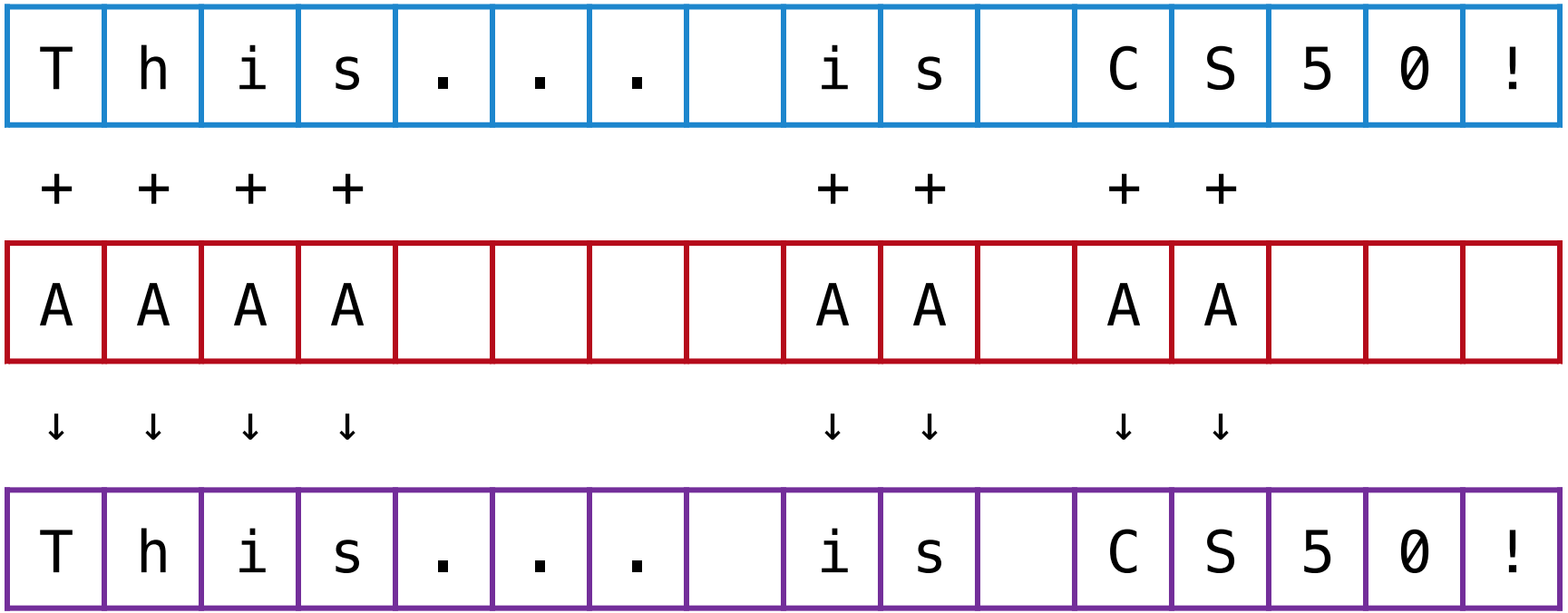
./vigenere ohai



The keyword is a string

- 'O' and 'o': key 14
- 'H' and 'h': key 7
- 'A' and 'a': key 0
- 'I' and 'i': key 8

./vigenere A



TODO

- [ ] get the key
- [ ] get the plaintext
- [ ] encipher
- [ ] print ciphertext

TODO

- get the key
  - 2nd command line argument: `argv[1]`
  - must be alphabetical: `isalpha`
- get the plaintext
- encipher
- print ciphertext

TODO

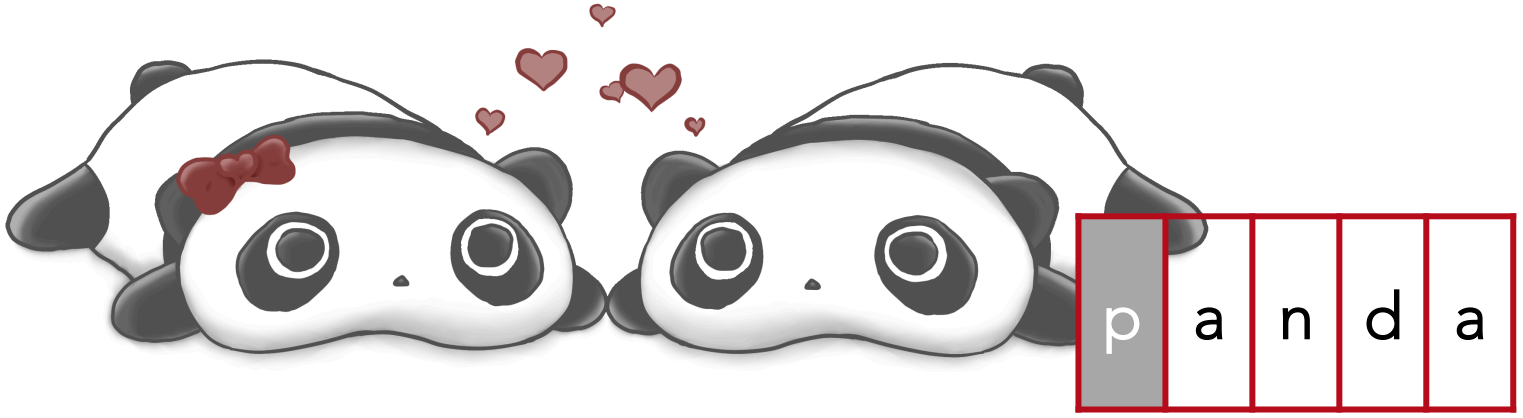
- [x] get the key
- [ ] get the plaintext
  - [ ] `get_string`
- [ ] encipher
- [ ] print ciphertext

TODO

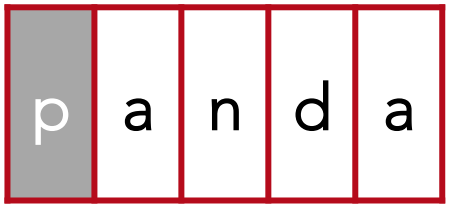
- [x] get the key
- [x] get the plaintext
- [ ] encipher
  - [ ] one character
  - [ ] entire plaintext
- [ ] print ciphertext

$$c_i = (p_i + k_j) \% 26$$

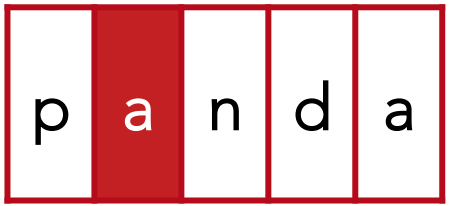
- c_i : i^{th} letter of ciphertext
- p_i : i^{th} letter of plaintext
- k_j : j^{th} letter of key
- % 26: remainder after dividing by 26



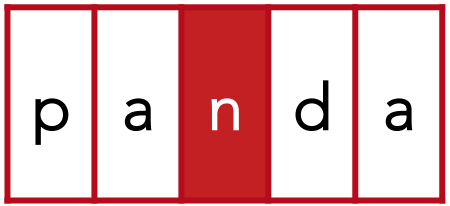
plaintext	I		l	i	k	e		y	o	u
ciphertext	X									



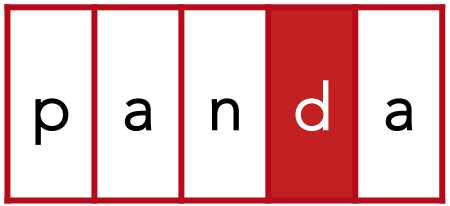
plaintext	I		l	i	k	e		y	o	u
ciphertext	X									



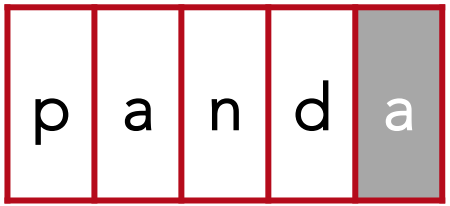
plaintext	I		l	i	k	e		y	o	u
ciphertext	X		l							



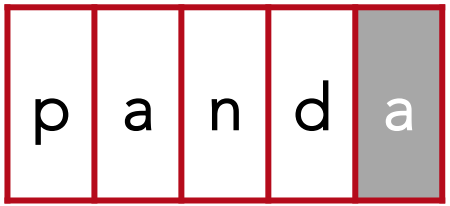
plaintext	I		l	i	k	e		y	o	u
ciphertext	X		l	v						



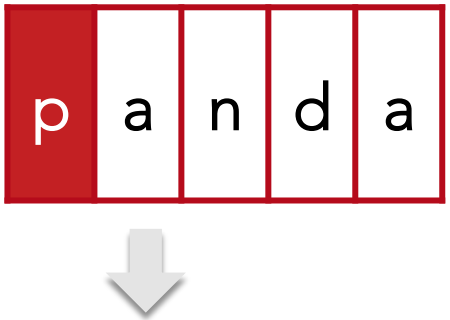
plaintext	I		l	i	k	e		y	o	u
ciphertext	X		l	v	n					



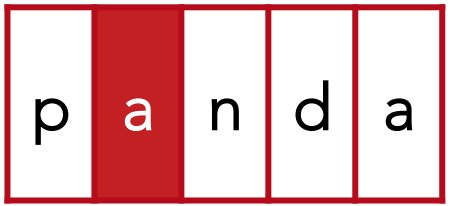
plaintext	I		l	i	k	e		y	o	u
ciphertext	X		l	v	n	e				



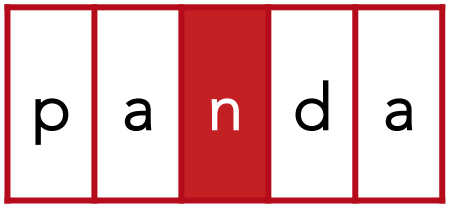
plaintext	I		l	i	k	e		y	o	u
ciphertext	X		l	v	n	e				



plaintext	I		l	i	k	e		y	o	u
ciphertext	X		l	v	n	e		n		



plaintext	I		l	i	k	e		y	o	u
ciphertext	X		l	v	n	e		n	o	



plaintext	I		l	i	k	e		y	o	u
ciphertext	X		l	v	n	e		n	o	h

enciphering

- advance to the next letter in the keyword only if the character in plaintext is a letter
  - `isalpha`
- preserve case
  - `isupper`, `islower`

letter	A	B	C	...	W	X	Y	Z
ASCII	65	66	67	...	87	88	89	90
shift	0	1	2	...	22	23	24	25

letter	a	b	c	...	w	x	y	z
ASCII	97	98	99	...	119	120	121	122
shift	0	1	2	...	22	23	24	25

$$c_i = (p_i + k_j) \% 26$$

- keep track of:
  - position in plaintext
  - position in keyword

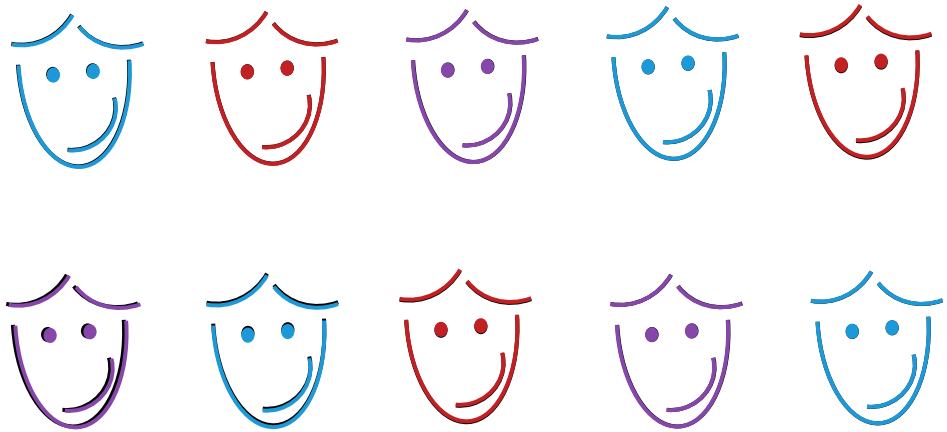
→ two separate variables

using modulo for wraparound

using modulo for wraparound

using modulo for wraparound

```
1 % 3 = 1;
2 % 3 = 2;
3 % 3 = 0;
4 % 3 = 1;
5 % 3 = 2;
```



TODO

- ☑ get the key
- ☑ get the plaintext
- ☑ encipher
- ☑ print ciphertext

this was vigenère