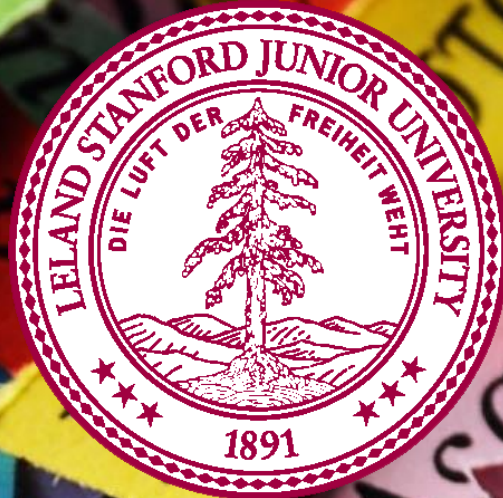
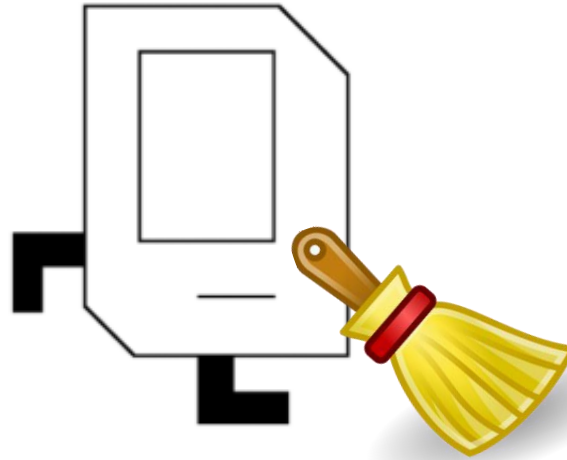




Text Processing

CS106A, Stanford University

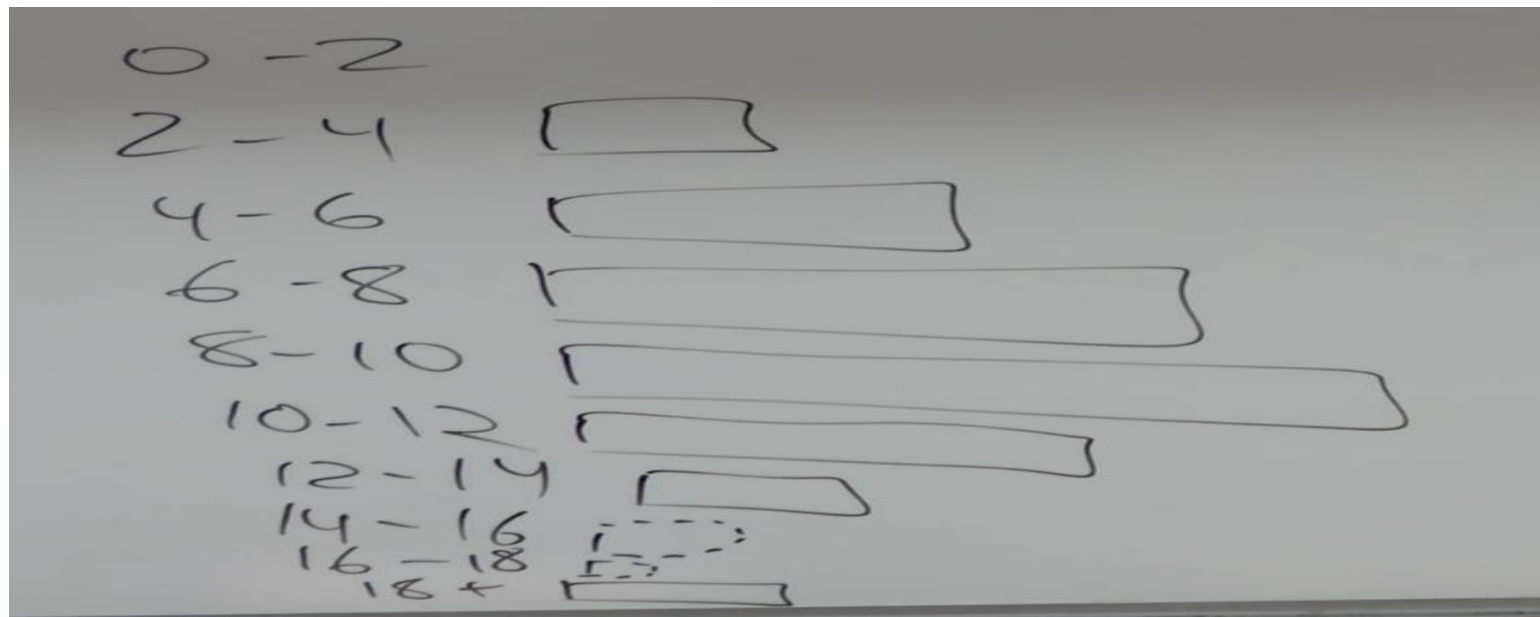
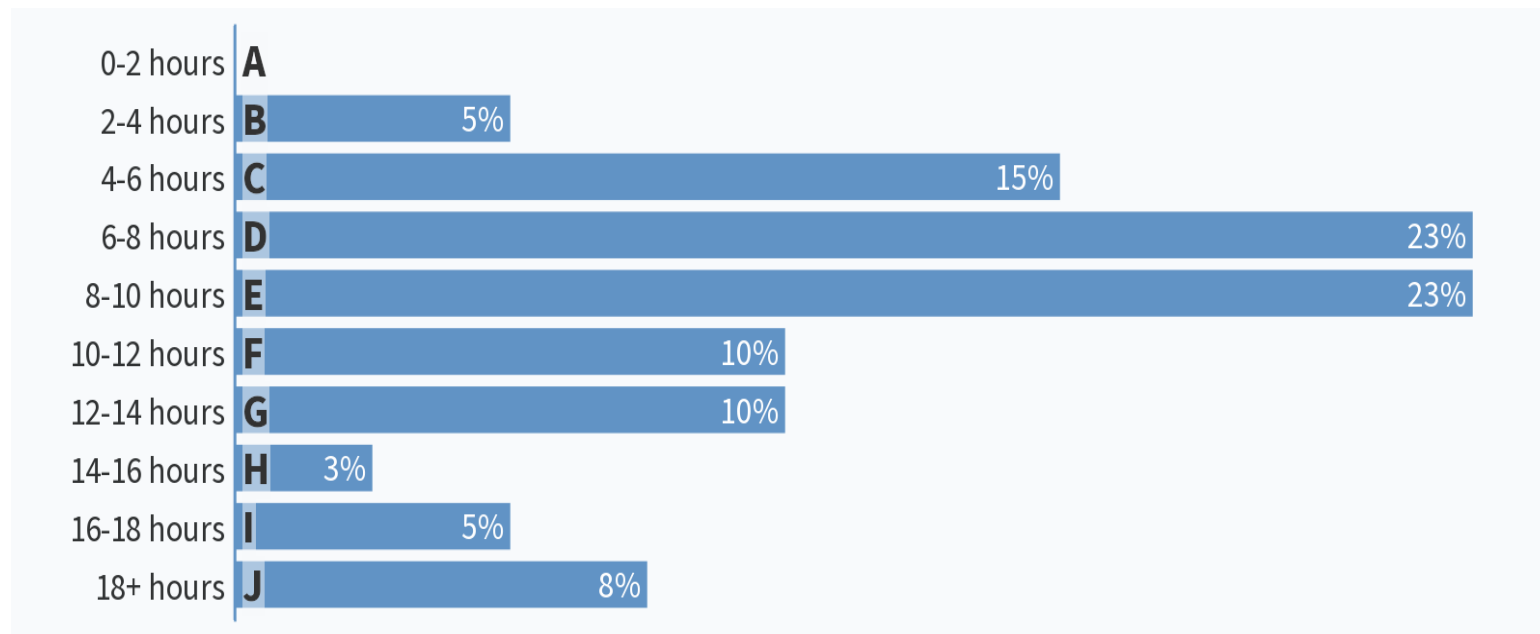
Housekeeping



- Midterm will be on Tuesday, May 3rd from 7pm-9pm
 - Midterm covers material through today's (April 27) class
 - Open book/notes, but will need to bring printed reference material
- Midterm review: Friday, 1:30-2:30pm in Nvidia Aud
- Will be using BlueBook software to take the exam
 - BlueBook will not allow you to access other applications/files on your computer during the exam
- Practice midterm available today
 - Will give you practice with topics on actual exam
 - Great way to prepare and also test your BlueBook set-up



Assignment #3 Poll

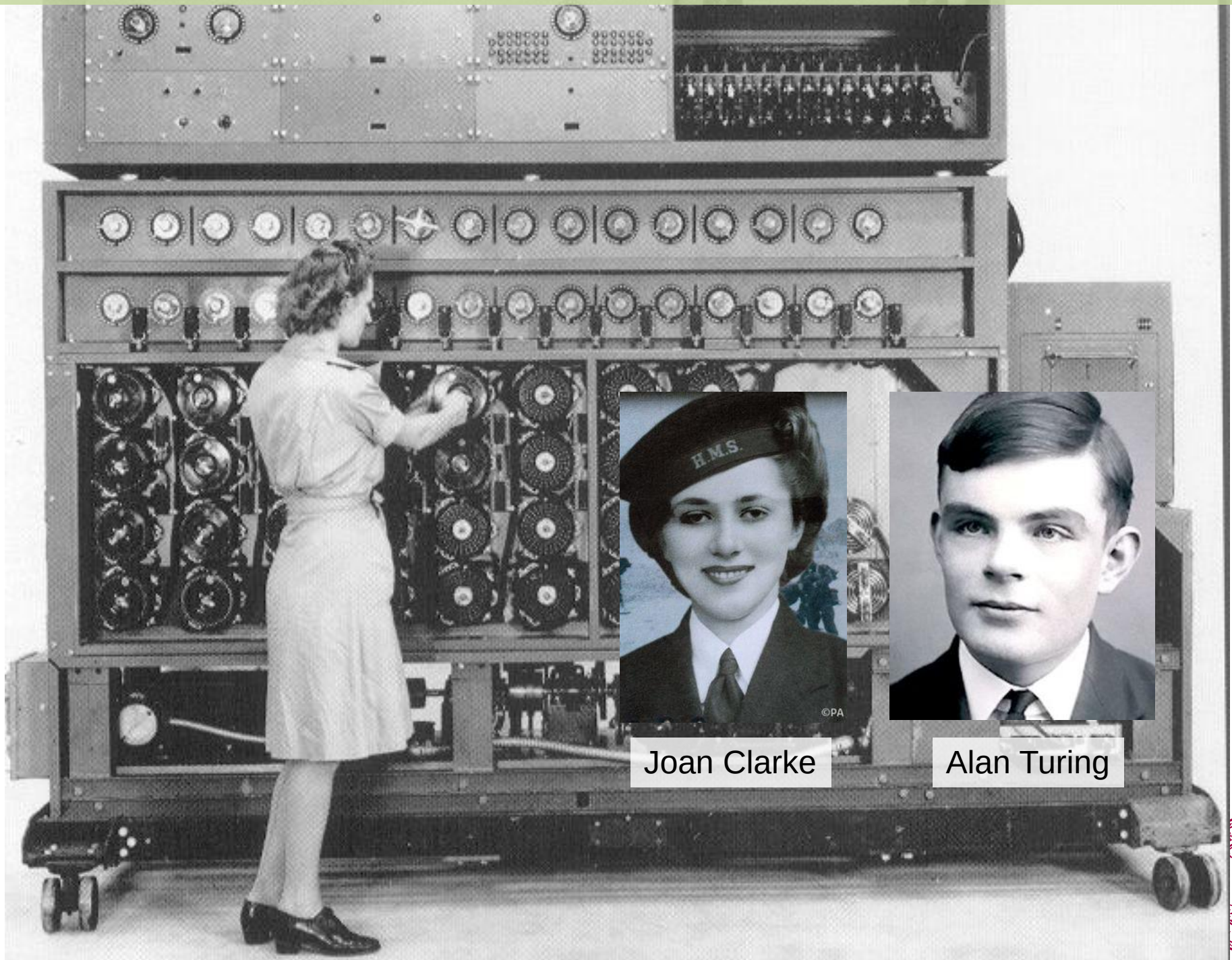


Learning Goals

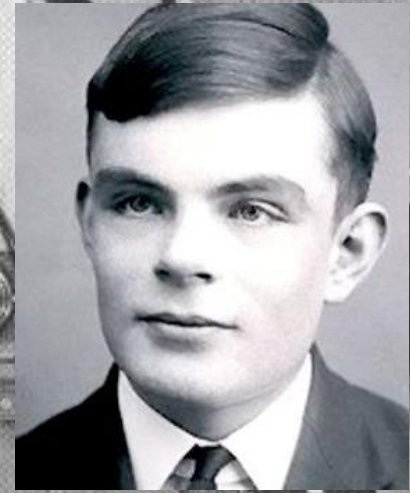
1. Learn about operations on string
2. Write code that manipulates strings



Text Problem: Decryption



Joan Clarke



Alan Turing

Text Problem: Translation

The spirit is willing but the flesh is weak.



(Russian)



The vodka is good but the meat is rotten.

**This result cost billions of dollars (adjusted for inflation)*



Text Problem: DNA Analysis



```
AGGTCAGTCAGATTTACCCTGGCTCACC  
TGTTTCGTACAACCAATTTAGGTGAGTTC.  
TTCGGAAAGACTCCCTGGTACCATCCCG  
CCGGGGGTTGGAATTTACGGGTCAGAAG  
ACCAATCGTAACATATGAGAGCCACTGC  
ATAATAGGGGAGGGTTCATTTTCGTGCT  
CTAACTTTGCTTAATACCCGACCACCAC  
CCACCCTGGCATTATAGTACCCCGAATC  
CGTAGAGCCAGATGTATGCAATGCCCCG  
GTAAGATCTCCAAAAGGTCGACGATGA  
AGTGGGTACTTTGGATAACCATCATTGGT  
ATCCGCTGATTGCTGGTTAATTCGTATG  
TCCCGGTTTCAAGTTTCAGACACTAGTT  
CCTAGGGGCGTCGACTGCCGACCATACT  
TCAATAGGGTATCGGGAGGTTTCATTAC  
TGGCACCCGTCCGTCAACGGTGTGCGCA  
GCCGCAGCTACCTCGAAAGTCATAGCGA  
CCTGATCGTCCATTACCGGGATGTGTGC  
GCGGGAGGGTCAACACAGGTAACCTCTC  
CTAACGCGTCCCCATCCCATCCGAGATT  
TTTTTTAGAAATGTTTTGTAGAATGGGAT  
TCGAGGGGTCCTTCGTTACCCATGGCGA  
AGTTTCGCAGTATTGACAAACGAGCATGG  
CCAGGAATTCCGATGCCGATCGTCTGAC  
ACACCTTTGTCCAATAACAAAGTAACG  
TGTGAAAGTTTTACCTAGATGGTCGTAG
```

How is text
represented?

The variable type `string`

Text is stored using the variable type `string`.

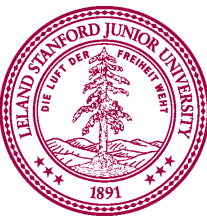
A `string` is a sequence of characters.

```
def main():  
    text = "hello!"  
    print(text)
```

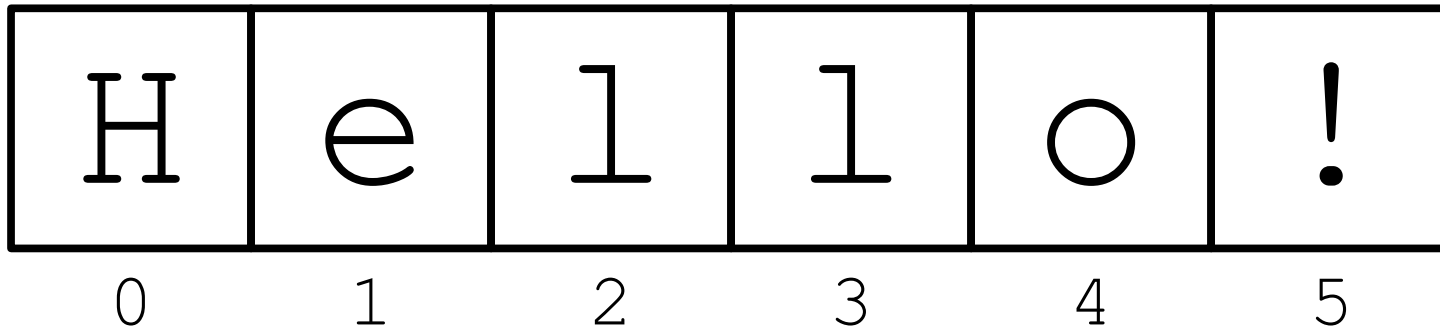


```
def main():  
    text = "hello!"
```

H e l l o !



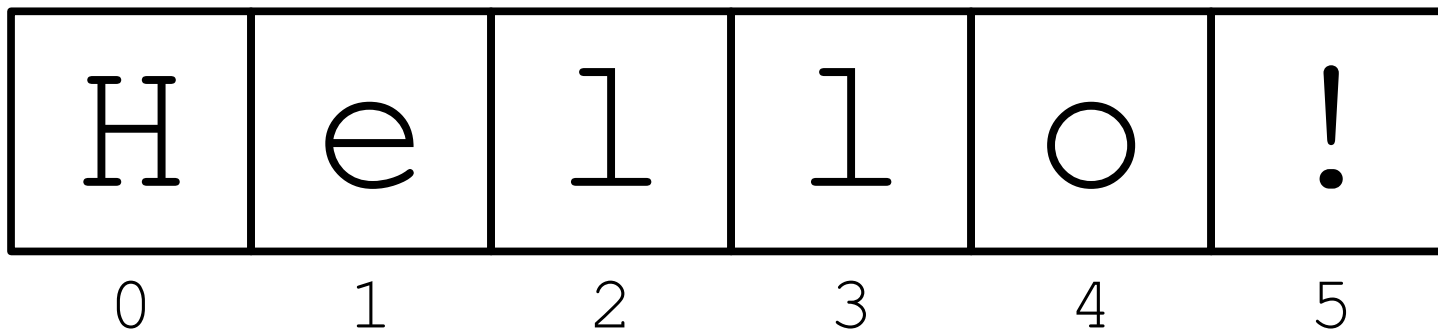
```
def main():  
    text = "hello!"
```



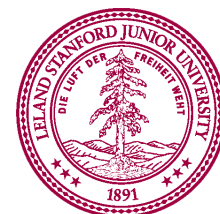
Indexed starting from 0




```
def main():  
    text = "hello!"
```



text [*index*]





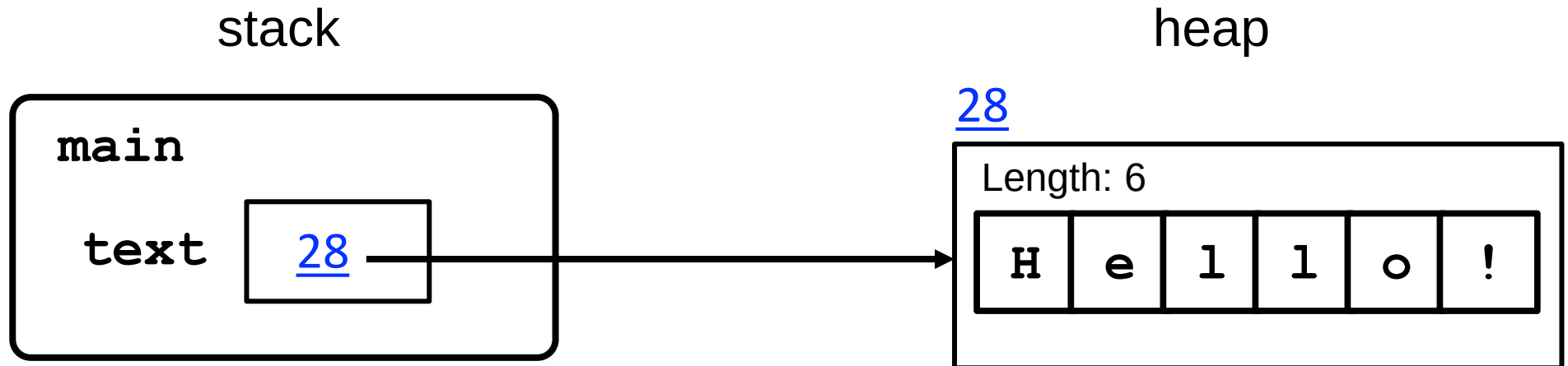
All characters in a string have
an index.

You can access a character in
the sequence via its *index*.



How it is actually stored

```
def main():  
    text = "hello!"
```



String Functions I

- Function: **len** (*string*)
 - Returns the number of characters in the string
- Function: *string* [**index**]
 - Returns the character at the given index
 - A character is really just a string of length 1

```
>>> str = "Hi mom"
```

```
>>> len(str)
```

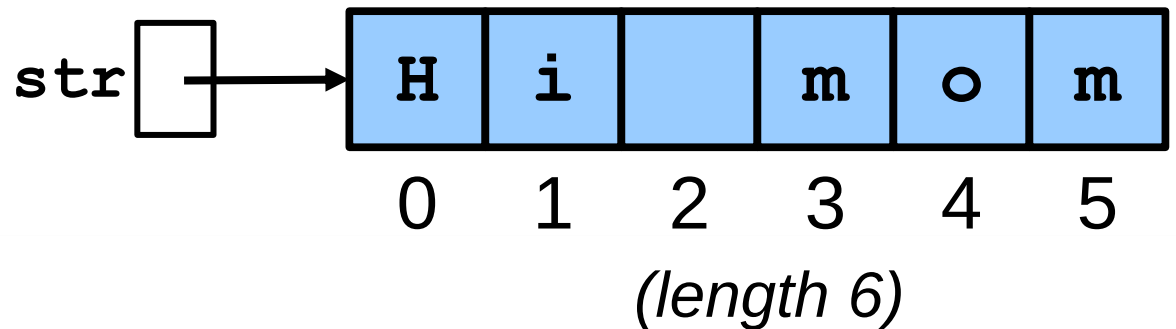
```
6
```

```
>>> str[0]
```

```
'H'
```

```
>>> str[3]
```

```
'm'
```



String Functions

```
def main():  
    example = "Hi mom"  
  
    # example of length function  
    length = len(example)  
    print(length) # prints 6  
  
    # example of getting a single character  
    first = example[0]  
    print(first) # prints 'H'  
  
    # loop that prints letters one-by-one  
    for i in range(len(example)):  
        ch = example[i]  
        print(ch)
```

String Functions

```
def main():  
    example = "Hi mom"  
  
    # example of length function  
    length = len(example)  
    print(length) # prints 6  
  
    # example of getting a single character  
    first = example[0]  
    print(first) # prints 'H'  
  
    # loop that prints letters one-by-one  
    for i in range(len(example)):  
        ch = example[i]  
        print(ch)
```

Console

String Functions

```
def main():  
    example = "Hi mom"  
  
    # example of length function  
    length = len(example)  
    print(length) # prints 6  
  
    # example of getting a single character  
    first = example[0]  
    print(first) # prints 'H'  
  
    # loop that prints letters one-by-one  
    for i in range(len(example)):  
        ch = example[i]  
        print(ch)
```

Console

example

H	i		m	o	m
0	1	2	3	4	5

String Functions

```
def main():  
    example = "Hi mom"  
  
    # example of length function  
    length = len(example)  
    print(length) # prints 6  
  
    # example of getting a single character  
    first = example[0]  
    print(first) # prints 'H'  
  
    # loop that prints letters one-by-one  
    for i in range(len(example)):  
        ch = example[i]  
        print(ch)
```

Console

example

H	i		m	o	m
---	---	--	---	---	---

0 1 2 3 4 5

length

6

String Functions

```
def main():  
    example = "Hi mom"  
  
    # example of length function  
    length = len(example)  
    print(length) # prints 6  
  
    # example of getting a single character  
    first = example[0]  
    print(first) # prints 'H'  
  
    # loop that prints letters one-by-one  
    for i in range(len(example)):  
        ch = example[i]  
        print(ch)
```

Console

6

example

H	i		m	o	m
---	---	--	---	---	---

0 1 2 3 4 5

length

6

String Functions

```
def main():  
    example = "Hi mom"  
  
    # example of length function  
    length = len(example)  
    print(length) # prints 6  
  
    # example of getting a single character  
    first = example[0]  
    print(first) # prints 'H'  
  
    # loop that prints letters one-by-one  
    for i in range(len(example)):  
        ch = example[i]  
        print(ch)
```

Console

6

example

H	i		m	o	m
---	---	--	---	---	---

0 1 2 3 4 5

length

6

first

'H'

String Functions

```
def main():  
    example = "Hi mom"  
  
    # example of length function  
    length = len(example)  
    print(length) # prints 6  
  
    # example of getting a single character  
    first = example[0]  
    print(first) # prints 'H'  
  
    # loop that prints letters one-by-one  
    for i in range(len(example)):  
        ch = example[i]  
        print(ch)
```

Console

6

H

example

H	i		m	o	m
---	---	--	---	---	---

0 1 2 3 4 5

length

6

first

'H'

String Functions

```
def main():  
    example = "Hi mom"  
  
    # example of length function  
    length = len(example)  
    print(length) # prints 6  
  
    # example of getting a single character  
    first = example[0]  
    print(first) # prints 'H'  
  
    # loop that prints letters one-by-one  
    for i in range(len(example)):  
        ch = example[i]  
        print(ch)
```

Console

6

H

example

H	i		m	o	m
---	---	--	---	---	---

0 1 2 3 4 5

length

6

first

'H'

String Functions

```
def main():  
    example = "Hi mom"  
  
    # example of length function  
    length = len(example)  
    print(length) # prints 6  
  
    # example of getting a single character  
    first = example[0]  
    print(first) # prints 'H'  
  
    # loop that prints letters one-by-one  
    for i in range(len(example)):  
        ch = example[i]  
        print(ch)
```

Console

6

H

example

H	i		m	o	m
---	---	--	---	---	---

0 1 2 3 4 5

length

6

first

'H'

i

0

String Functions

```
def main():  
    example = "Hi mom"  
  
    # example of length function  
    length = len(example)  
    print(length) # prints 6  
  
    # example of getting a single character  
    first = example[0]  
    print(first) # prints 'H'  
  
    # loop that prints letters one-by-one  
    for i in range(len(example)):  
        ch = example[i]  
        print(ch)
```

Console

6

H

example

H	i		m	o	m
---	---	--	---	---	---

0 1 2 3 4 5

length

6

first

'H'

i

0

String Functions

```
def main():  
    example = "Hi mom"  
  
    # example of length function  
    length = len(example)  
    print(length) # prints 6  
  
    # example of getting a single character  
    first = example[0]  
    print(first) # prints 'H'  
  
    # loop that prints letters one-by-one  
    for i in range(len(example)):  
        ch = example[i]  
        print(ch)
```

[0, 1, 2, 3, 4, 5]

Console

6

H

example

H	i		m	o	m
---	---	--	---	---	---

0 1 2 3 4 5

length

6

first

'H'

i

0

String Functions

```
def main():  
    example = "Hi mom"  
  
    # example of length function  
    length = len(example)  
    print(length) # prints 6  
  
    # example of getting a single character  
    first = example[0]  
    print(first) # prints 'H'  
  
    # loop that prints letters one-by-one  
    for i in range(len(example)):  
        ch = example[i]  
        print(ch)
```

Console

6

H

example

H	i		m	o	m
---	---	--	---	---	---

0 1 2 3 4 5

length

6

first

'H'

i

0

String Functions

```
def main():  
    example = "Hi mom"  
  
    # example of length function  
    length = len(example)  
    print(length) # prints 6  
  
    # example of getting a single character  
    first = example[0]  
    print(first) # prints 'H'  
  
    # loop that prints letters one-by-one  
    for i in range(len(example)):  
        ch = example[i]  
        print(ch)
```

Console

6

H

example

H	i		m	o	m
---	---	--	---	---	---

0 1 2 3 4 5

length

6

first

'H'

i

0

ch

'H'

String Functions

```
def main():  
    example = "Hi mom"  
  
    # example of length function  
    length = len(example)  
    print(length) # prints 6  
  
    # example of getting a single character  
    first = example[0]  
    print(first) # prints 'H'  
  
    # loop that prints letters one-by-one  
    for i in range(len(example)):  
        ch = example[i]  
        print(ch)
```

Console

6

H

H

example

H	i		m	o	m
---	---	--	---	---	---

0

1

2

3

4

5

length

6

first

'H'

i

0

ch

'H'

String Functions

```
def main():  
    example = "Hi mom"  
  
    # example of length function  
    length = len(example)  
    print(length) # prints 6  
  
    # example of getting a single character  
    first = example[0]  
    print(first) # prints 'H'  
  
    # loop that prints letters one-by-one  
    for i in range(len(example)):  
        ch = example[i]  
        print(ch)
```

Console

6

H

H

example

H	i		m	o	m
---	---	--	---	---	---

0

1

2

3

4

5

length

6

first

'H'

i

0

String Functions

```
def main():  
    example = "Hi mom"  
  
    # example of length function  
    length = len(example)  
    print(length) # prints 6  
  
    # example of getting a single character  
    first = example[0]  
    print(first) # prints 'H'  
  
    # loop that prints letters one-by-one  
    for i in range(len(example)):  
        ch = example[i]  
        print(ch)
```

Console

6

H

H

example

H	i		m	o	m
---	---	--	---	---	---

0

1

2

3

4

5

length

6

first

'H'

i

1

String Functions

```
def main():  
    example = "Hi mom"  
  
    # example of length function  
    length = len(example)  
    print(length) # prints 6  
  
    # example of getting a single character  
    first = example[0]  
    print(first) # prints 'H'  
  
    # loop that prints letters one-by-one  
    for i in range(len(example)):  
        ch = example[i]  
        print(ch)
```

Console

6

H

H

example

H	i		m	o	m
---	---	--	---	---	---

0 1 2 3 4 5

length

6

first

'H'

i

1

String Functions

```
def main():  
    example = "Hi mom"  
  
    # example of length function  
    length = len(example)  
    print(length) # prints 6  
  
    # example of getting a single character  
    first = example[0]  
    print(first) # prints 'H'  
  
    # loop that prints letters one-by-one  
    for i in range(len(example)):  
        ch = example[i]  
        print(ch)
```

Console

6

H

H

example

H	i		m	o	m
---	---	--	---	---	---

0 1 2 3 4 5

length

6

first

'H'

i

1

ch

'i'

String Functions

```
def main():  
    example = "Hi mom"  
  
    # example of length function  
    length = len(example)  
    print(length) # prints 6  
  
    # example of getting a single character  
    first = example[0]  
    print(first) # prints 'H'  
  
    # loop that prints letters one-by-one  
    for i in range(len(example)):  
        ch = example[i]  
        print(ch)
```

Console

6

H

H

i

example

H	i		m	o	m
---	---	--	---	---	---

0

1

2

3

4

5

length

6

first

'H'

i

1

ch

'i'

String Functions

```
def main():  
    example = "Hi mom"  
  
    # example of length function  
    length = len(example)  
    print(length) # prints 6  
  
    # example of getting a single character  
    first = example[0]  
    print(first) # prints 'H'  
  
    # loop that prints letters one-by-one  
    for i in range(len(example)):  
        ch = example[i]  
        print(ch)
```

Console

6

H

H

i

example

H	i		m	o	m
---	---	--	---	---	---

0 1 2 3 4 5

length

6

first

'H'

i

1

String Functions

```
def main():  
    example = "Hi mom"  
  
    # example of length function  
    length = len(example)  
    print(length) # prints 6  
  
    # example of getting a single character  
    first = example[0]  
    print(first) # prints 'H'  
  
    # loop that prints letters one-by-one  
    for i in range(len(example)):  
        ch = example[i]  
        print(ch)
```

Console

6

H

H

i

example

H	i		m	o	m
---	---	--	---	---	---

0 1 2 3 4 5

length

6

first

'H'

i

2

String Functions

```
def main():  
    example = "Hi mom"  
  
    # example of length function  
    length = len(example)  
    print(length) # prints 6  
  
    # example of getting a single character  
    first = example[0]  
    print(first) # prints 'H'  
  
    # loop that prints letters one-by-one  
    for i in range(len(example)):  
        ch = example[i]  
        print(ch)
```

Console

6

H

H

i

example

H	i		m	o	m
---	---	--	---	---	---

0 1 2 3 4 5

length

6

first

'H'

i

2

String Functions

```
def main():  
    example = "Hi mom"  
  
    # example of length function  
    length = len(example)  
    print(length) # prints 6  
  
    # example of getting a single character  
    first = example[0]  
    print(first) # prints 'H'  
  
    # loop that prints letters one-by-one  
    for i in range(len(example)):  
        ch = example[i]  
        print(ch)
```

Console

6

H

H

i

example

H	i		m	o	m
---	---	--	---	---	---

0 1 2 3 4 5

length

6

first

'H'

i

2

ch

! !

String Functions

```
def main():  
    example = "Hi mom"  
  
    # example of length function  
    length = len(example)  
    print(length) # prints 6  
  
    # example of getting a single character  
    first = example[0]  
    print(first) # prints 'H'  
  
    # loop that prints letters one-by-one  
    for i in range(len(example)):  
        ch = example[i]  
        print(ch)
```

Console

6

H

H

i

example

H	i		m	o	m
---	---	--	---	---	---

0

1

2

3

4

5

length

6

first

'H'

i

2

ch

' '

String Functions

```
def main():  
    example = "Hi mom"  
  
    # example of length function  
    length = len(example)  
    print(length) # prints 6  
  
    # example of getting a single character  
    first = example[0]  
    print(first) # prints 'H'  
  
    # loop that prints letters one-by-one  
    for i in range(len(example)):  
        ch = example[i]  
        print(ch)
```

Console

6

H

H

i

example

H	i		m	o	m
---	---	--	---	---	---

0 1 2 3 4 5

length

6

first

'H'

i

2

ch

' '

String Functions

```
def main():  
    example = "Hi mom"  
  
    # example of length function  
    length = len(example)  
    print(length) # prints 6  
  
    # example of getting a single character  
    first = example[0]  
    print(first) # prints 'H'  
  
    # loop that prints letters one-by-one  
    for i in range(len(example)):  
        ch = example[i]  
        print(ch)
```

Console

6

H

H

i

m

o

m

example

H	i		m	o	m
---	---	--	---	---	---

0 1 2 3 4 5

length

6

first

'H'

i

5

ch

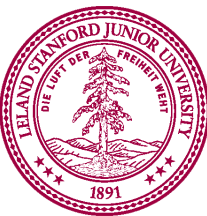
'm'



A string is indexed just
like a list!

Slices work too.

It is *almost* like it is a list
of characters.

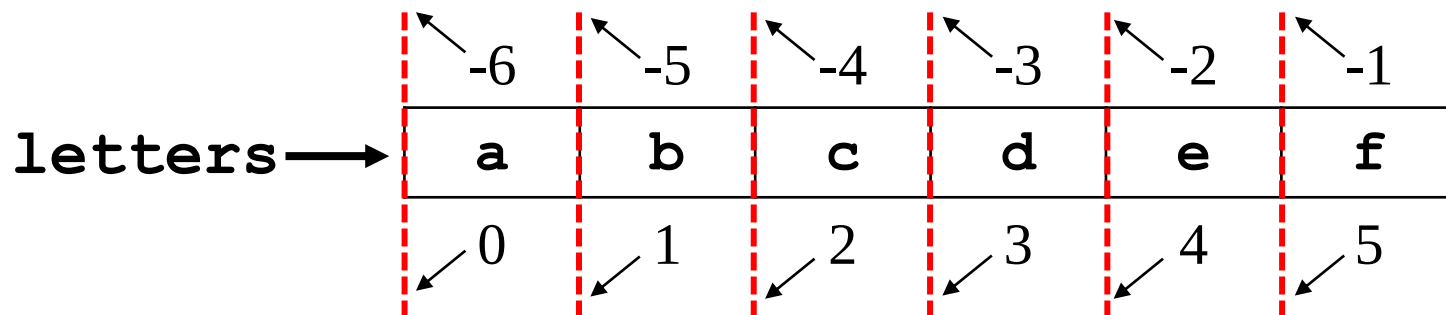


Slicing Strings

- Just like slicing a list, but with a string
string [*start* : *end* : *step*]
 - Produces a new string with characters from *string* starting at index *start* up to (but not including) index *end*

- Example:

`letters = 'abcdef'`



`letters[2:4]` → `'cd'`

`letters[1:-2]` → `'bcd'`

`letters[::-1]` → `'fedcba'`



Looping Through Strings

```
name = 'Neil'
```

- For loop using `range`:

```
for i in range(len(name)) :  
    ch = name[i]  
    print(ch)
```

- We can also use a "for-each" loop

```
for ch in name:  
    print(ch)
```

- Both loops iterate over all characters of the string
 - Variable `ch` is set to each character in string (in order)

Output:

```
N  
e  
i  
l
```

String-a-palooza (String Functions 1)

- Function: *string1* + *string2* # concatenation
 - Returns a new string that is concatenation of *string1* and *string2*

```
str1 = 'abc'  
str2 = 'def'  
str3 = str1 + str2
```

str3:

```
'abcdef'
```

- Function: *string*.strip()
 - Returns a new string with leading/trailing spaces removed

```
message = ' this is a test! '  
result = message.strip()
```

result:

```
'this is a test!'
```



String-a-palooza (String Functions 2)

- Function: string.split()

- Returns a new list based on whitespace separated terms in *string*

```
message = ' this is a test! '
```

```
my_list = message.split()
```

```
my_list:
```

```
['this', 'is', 'a', 'test!']
```

- Function: string.find(string_to_find)

- Returns index of first occurrence of *string_to_find* in *string*

- Returns -1 if *string_to_find* is not found in the original *string*

```
shout_out = 'Chris is awesome'
```

```
result1 = shout_out.find('is')
```

```
result2 = shout_out.find('bad')
```

```
result1: 3
```

```
result2: -1
```



String-a-palooza (String Functions 3)

- Function: string.upper()

- Returns a new string that is upper case version of original *string*

```
str = 'Oh, happy day!'
```

```
new_str = str.upper()
```

```
new_str:
```

```
'OH, HAPPY DAY!'
```

- Function: string.lower()

- Returns a new string that is lower case version of original *string*

```
tweet = 'I have to YELL EVERYTHING!!!'
```

```
mellow = tweet.lower()
```

```
mellow:
```

```
'i have to yell everything!!!'
```



String-a-palooza (String Functions 4)

- Function: `string.replace(old, new)`
 - Returns a new string where every occurrence of substring *old* is replaced by substring *new*

```
str = 'happy, scrappy, good time'
```

```
new_str = str.replace('ppy', 'milton')
```

```
new_str:
```

```
'hamilton, scramilton, good time'
```

- If the substring *old* does not occur in the original string, you get a new copy of the same original string

```
music = 'Rush is my favorite band'
```

```
new_music = music.replace('noise', 'Nickelback')
```

```
new_music:
```

```
'Rush is my favorite band'
```



String-a-palooza (String Functions 5)

- Function: string.isalpha()
 - Returns true if all characters in *string* are letters in alphabet

```
str1 = 'letters'  
result1 = str1.isalpha()    # result1 is True  
str2 = 'there are spaces here'  
result2 = str2.isalpha()    # result2 is False
```
- Function: string.isdigit()
 - Returns true if all characters in *string* are digits ('0' to '9')

```
str = '012345'  
result = str.isdigit()      # result is True
```
- Function: string.isspace()
 - Returns true if all characters in string are whitespace (e.g., space, tab, newline)

```
str = '   '  
result = str.isspace()      # result is True
```

Advanced version

How are characters
represented?

Single characters

- Some examples:

letter_A = 'A'

plus = '+'

zero = '0'

space = ' '

korean_ch = '호'

new_line = '\n'

tab = '\t'

backslash = '\\'

singlequote = '\''

emoji = '😊'

\n newline

\t tab

\\ (one) backslash

\' single quote

\" double quote

special

special

special

special





Advanced course

Characters are just a giant
enumeration!



Just the numbers please

ASCII

Code	Char	Code	Char	Code	Char	Code	Char	Code	Char	Code	Char
32	[space]	48	0	64	@	80	P	96	`	112	p
33	!	49	1	65	A	81	Q	97	a	113	q
34	"	50	2	66	B	82	R	98	b	114	r
35	#	51	3	67	C	83	S	99	c	115	s
36	\$	52	4	68	D	84	T	100	d	116	t
37	%	53	5	69	E	85	U	101	e	117	u
38	&	54	6	70	F	86	V	102	f	118	v
39	'	55	7	71	G	87	W	103	g	119	w
40	(	56	8	72	H	88	X	104	h	120	x
41	)	57	9	73	I	89	Y	105	i	121	y
42	*	58	:	74	J	90	Z	106	j	122	z
43	+	59	;	75	K	91	[	107	k	123	{
44	,	60	<	76	L	92	\	108	l	124	
45	-	61	=	77	M	93	]	109	m	125	}
46	.	62	>	78	N	94	^	110	n	126	~
47	/	63	?	79	O	95	_	111	o	127	[backspace]

* This is only part of the table

The letter A, for example, has the ASCII value 65



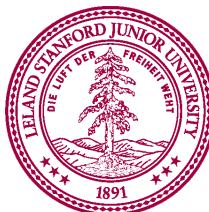
Unicode (bigger ASCII)

3200

Enclosed CJK Letters and Months

32FF

	320	321	322	323	324	325	326	327	328	329	32A	32B	32C	32D	32E	32F
0	(ㄱ)	(ㄷ)	(一)	(日)	祭	PTE	ㄱ	ㄷ	一	日	項	夜	1月	ア	チ	ム
1	(ㄴ)	(ㄹ)	(二)	株	休	21	ㄴ	ㄹ	二	株	休	36	2月	イ	ツ	メ
2	(ㄷ)	(ㅁ)	(三)	有	自	22	ㄷ	ㅁ	三	有	写	37	3月	ウ	テ	モ
3	(ㄷ)	(ㅁ)	(四)	社	至	23	ㄷ	(ㅁ)	四	社	正	38	4月	エ	ト	ヤ
4	(ㄷ)	(ㅁ)	(五)	名	問	24	ㄷ	(ㅁ)	五	名	上	39	5月	オ	ナ	ユ
5	(ㄷ)	(ㅁ)	(六)	特	幼	25	ㄷ	(ㅁ)	六	特	中	40	6月	カ	ニ	ヨ
6	(ㄷ)	(ㅁ)	(七)	財	文	26	ㄷ	(ㅁ)	七	財	下	41	7月	キ	ヌ	ラ
7	(ㄷ)	(ㅁ)	(八)	祝	筆	27	ㄷ	(ㅁ)	八	祝	左	42	8月	ク	ネ	リ
8	(ㄷ)	(ㅁ)	(九)	勞	10	28	ㄷ	(ㅁ)	九	勞	右	43	9月	ケ	ノ	ル
9	(ㄷ)	(ㅁ)	(十)	代	20	29	ㄷ	(ㅁ)	十	秘	医	44	10月	コ	ハ	レ
A	(ㄷ)	(ㅁ)	(月)	呼	30	30	ㄷ	(ㅁ)	月	男	宗	45	11月	サ	ヒ	ロ
B	(ㄷ)	(ㅁ)	(火)	学	40	31	ㄷ	(ㅁ)	火	女	学	46	12月	シ	フ	ワ
C	(ㄷ)	(ㅁ)	(水)	監	50	32	ㄷ	(ㅁ)	水	適	監	47	Hg	ス	ヘ	中
D	(ㄷ)	(ㅁ)	(木)	企	60	33	ㄷ	(ㅁ)	木	優	企	48	erg	セ	ホ	エ
E	(ㄷ)	(ㅁ)	(金)	資	70	34	ㄷ	(ㅁ)	金	印	資	49	eV	ソ	マ	ヲ
F	(ㄷ)	(ㅁ)	(土)	協	80	35	ㄷ	(ㅁ)	土	注	協	50	ITD	タ	ミ	令





'A' → 'Z' are sequential.

'a' → 'z' are sequential.

'0' → '9' are sequential.

Can get ASCII value of character **ch** with **ord(ch)**

```
>>> ord('a')
```

```
97
```

```
>>> ord('b')
```

```
98
```

```
>>> ord('1')
```

```
49
```

```
>>> ord('2')
```

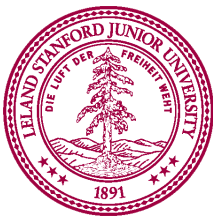
```
50
```



Math with characters

- Say we want a function that returns the position (index) of a letter in the alphabet (case insensitive)
 - Non-letters should return -1 to indicate they are not in alphabet
 - For example:
 - 'a' → 0
 - 'D' → 3
 - '!' → -1

```
def index_in_alphabet(ch):  
    if ch.isalpha():  
        lowercase = ch.lower()  
        return ord(lowercase) - ord('a')  
    else:  
        return -1
```



String Comparisons

- Can compare strings

- Test for equal strings (all characters are the same, case sensitive):

```
str = 'abc'  
if str == 'abc':      # test would be True here  
    # body
```

- Strings are compared in lexicographic (~alphabetic) order:

```
'all' < 'always'    # True  
'good' < 'bad'      # False  
'table' > 'desk'    # True
```

- Case matters (all uppercase letters come before lowercase ones)

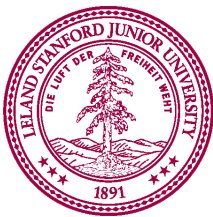
```
'ABC' < 'abc'       # True  
'Zowie' < 'abc'     # True
```

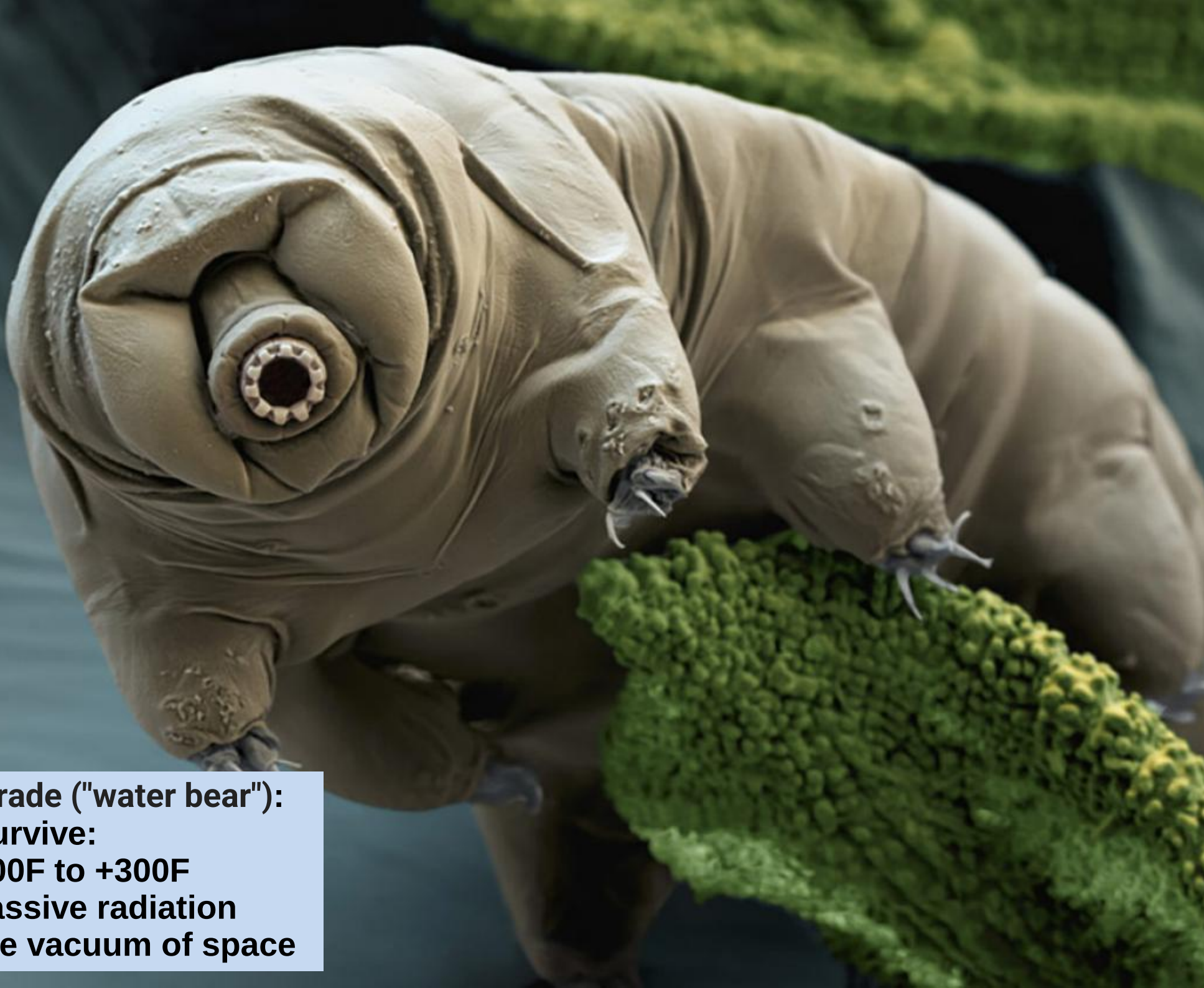


Strings have some
interesting properties

Strings are Immutable

- Python strings are *immutable*: once a string has been created **you cannot set characters**.
- To change a string:
 - *Create a new string* holding the new value you want it to have via concatenation.
 - Can reassign to the same string variable.
- *Important consequence*: if you pass a string into a function, you are guaranteed that string won't be changed.
 - Similar to behavior of int and float when passed to a function





Tardigrade ("water bear"):
Can survive:
-300F to +300F
Massive radiation
The vacuum of space

Strings are Immutable (Take 1)

```
str = 'abc'
```



```
str[1] = 'z'
```

Error!

Traceback (most recent call last):

...

TypeError: 'str' object does not support item assignment

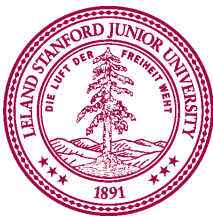
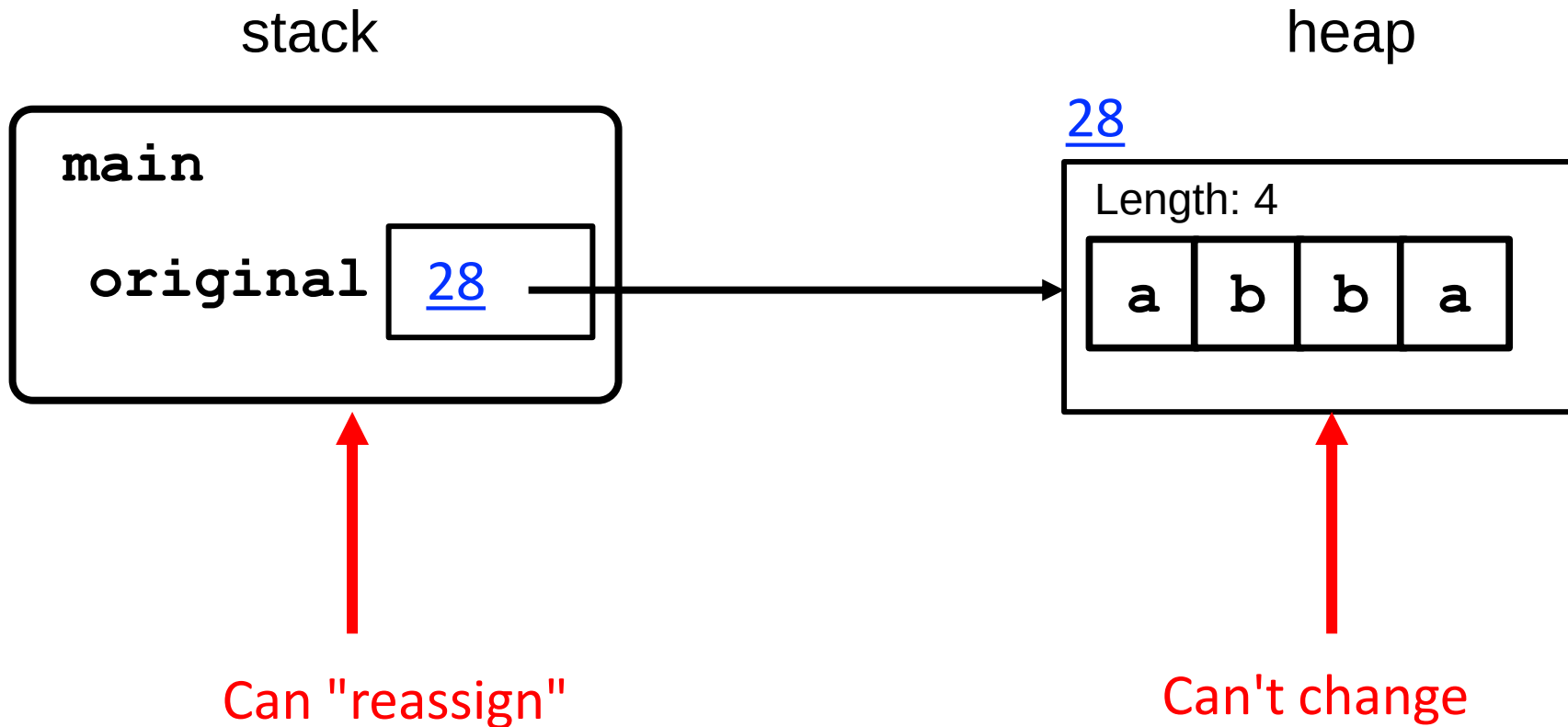


```
str = 'azc'
```

Have to assign a new string
(Set a new "binding")

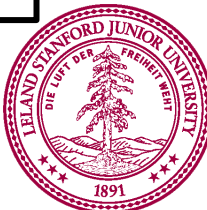
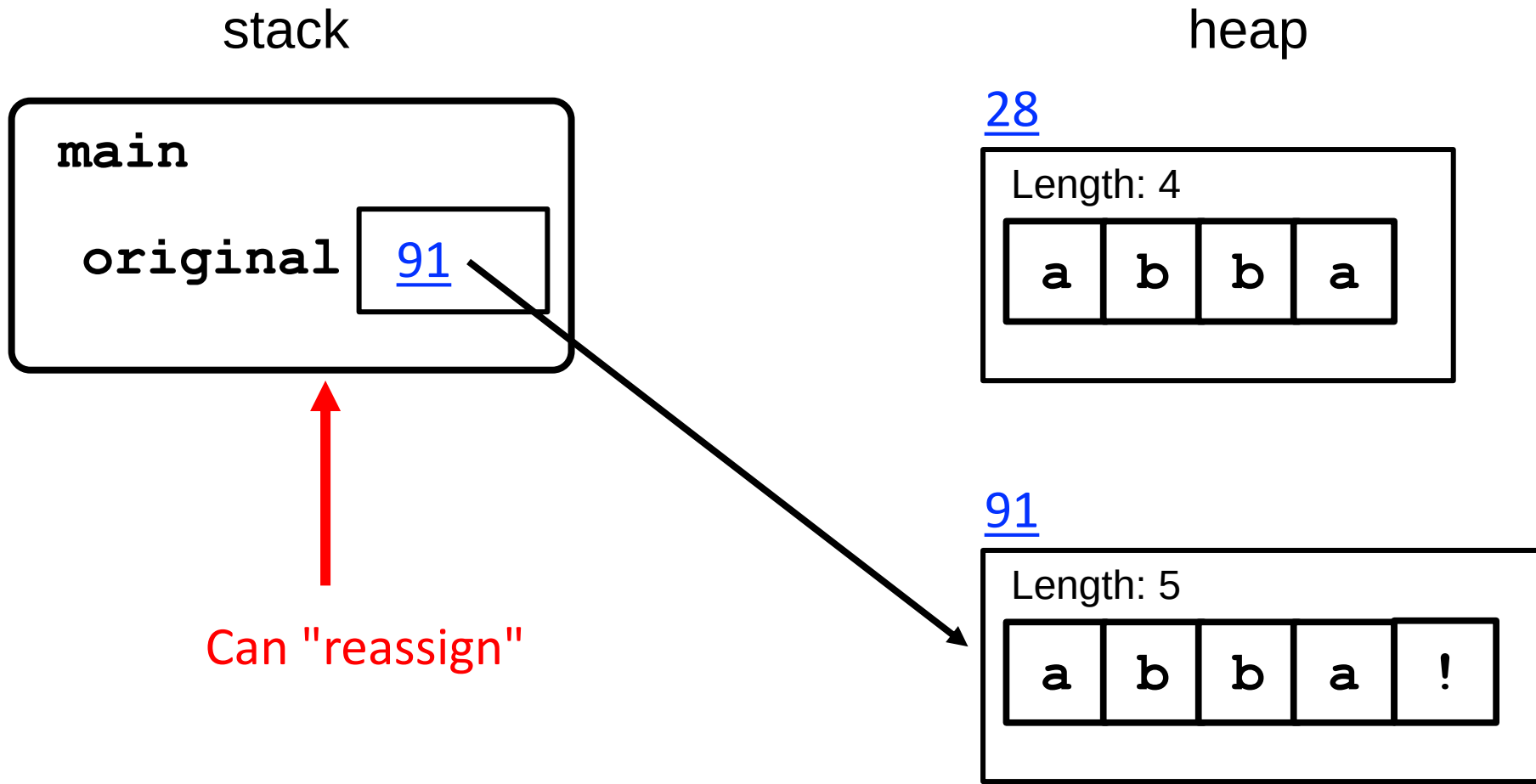
Strings are Immutable (Take 2)

```
original = 'abba'
```



Strings are Immutable (Take 2)

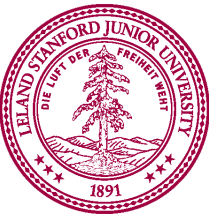
```
original = 'abba'  
original = original + '!'
```



Strings are often made through concatenation

```
def main():  
    s1 = "CS106"  
    s2 = "A"  
    s3 = "I got an " + s2 + " in " + s1 + s2  
  
    print(s3)
```

I got an A in CS106A



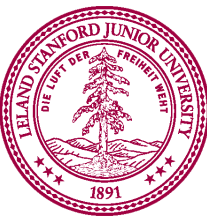


Lists are **mutable**

Strings are **immutable**

*Immutable is a guarantee that
a function won't be cheeky*

*(Immutable parameters cannot
be changed in place)*



The New Hotness: fstrings (formatted strings)

```
year = 2022
action = 'vote in the US election'
important = f'It is {year}, you gotta {action}!'
print(important)
```

It is 2022, you gotta vote in the US election!

General form:

`f 'text {variable} text {variable} text {variable} ...'`

- Were added to Python starting in version 3.6
- Values of variables are appended into string
- Can use this anywhere you use strings (print, creating strings, etc.)
- Numeric formatting to print only a certain number of decimal points

```
>>> weight = 160.123456
```

```
>>> print(f'Weight to two digits: {weight:.2f}')
```

```
Weight to two digits: 160.12
```



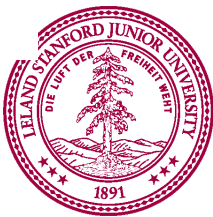
Many string algorithms use
the "loop and construct"
pattern.



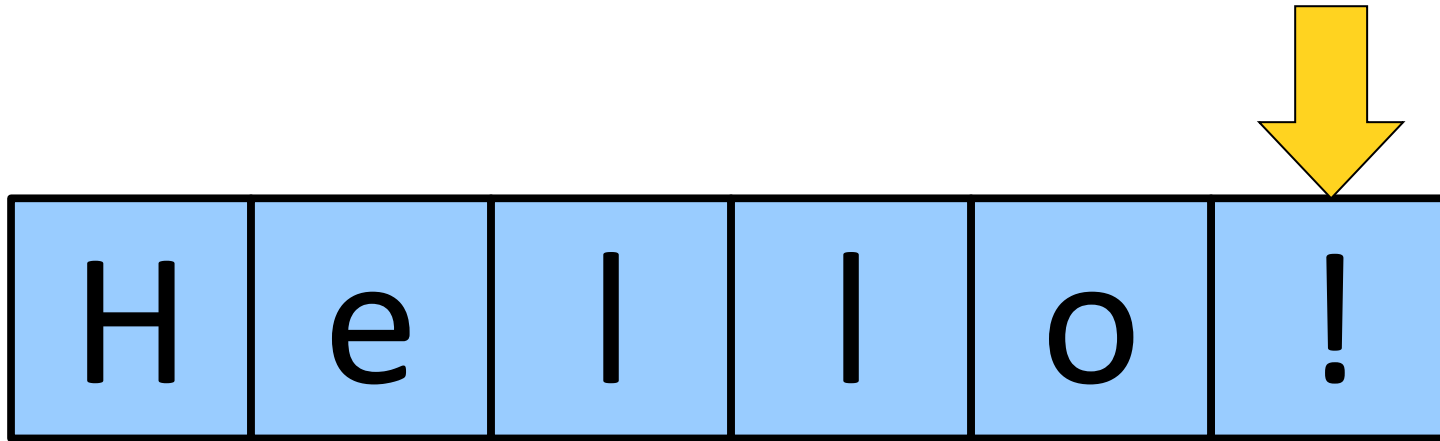
Reversing a String

H	e	l	l	o	!
---	---	---	---	---	---

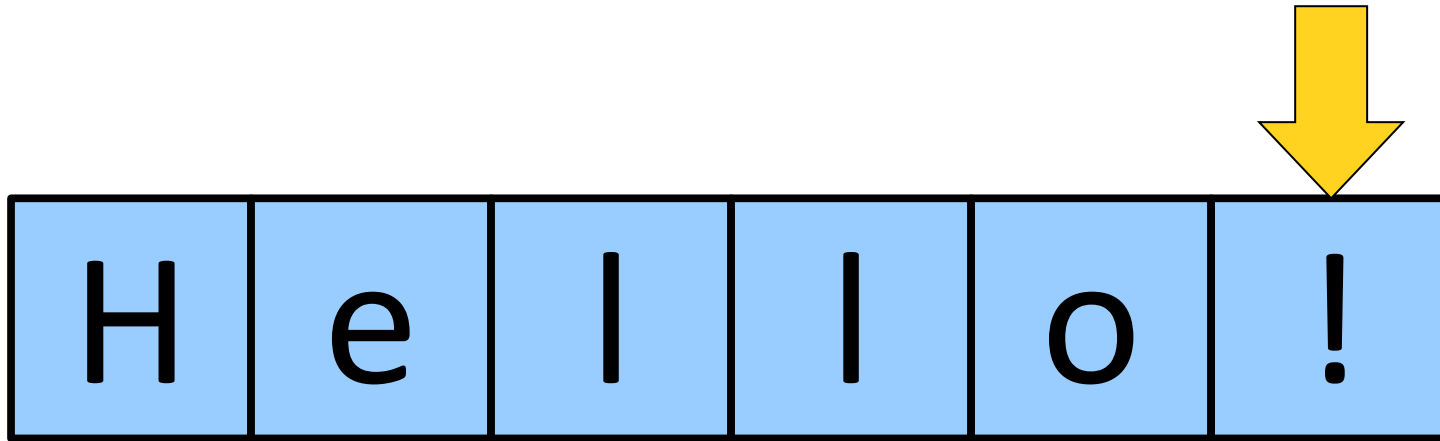
```
result = ''  
# counts from len(str)-1 down to 0 (including 0)  
for i in range(len(str)-1, -1, -1):  
    ch = str[i]  
    result += ch  
return result
```



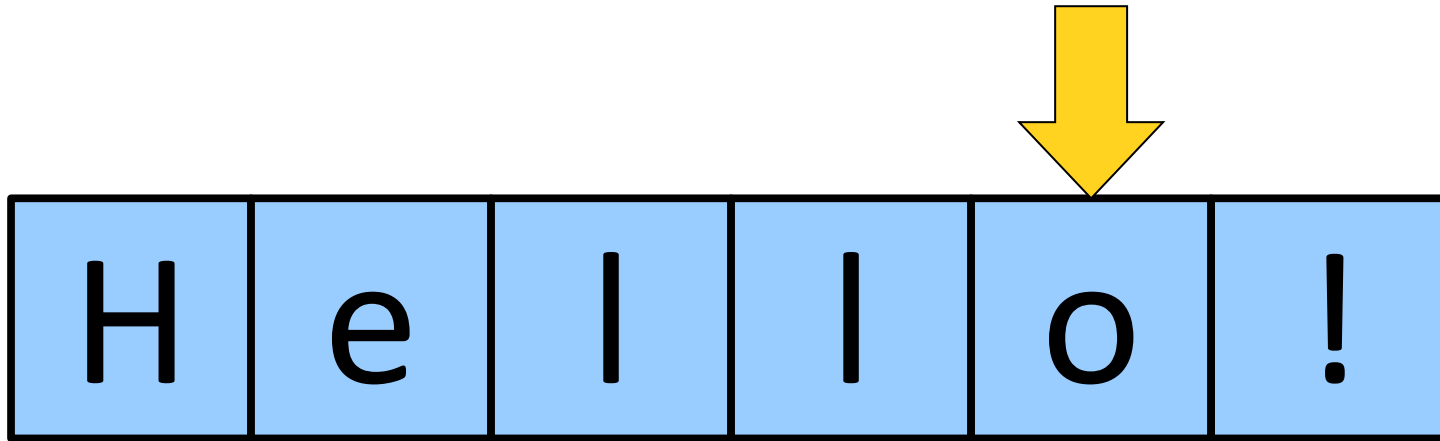
Reversing a String



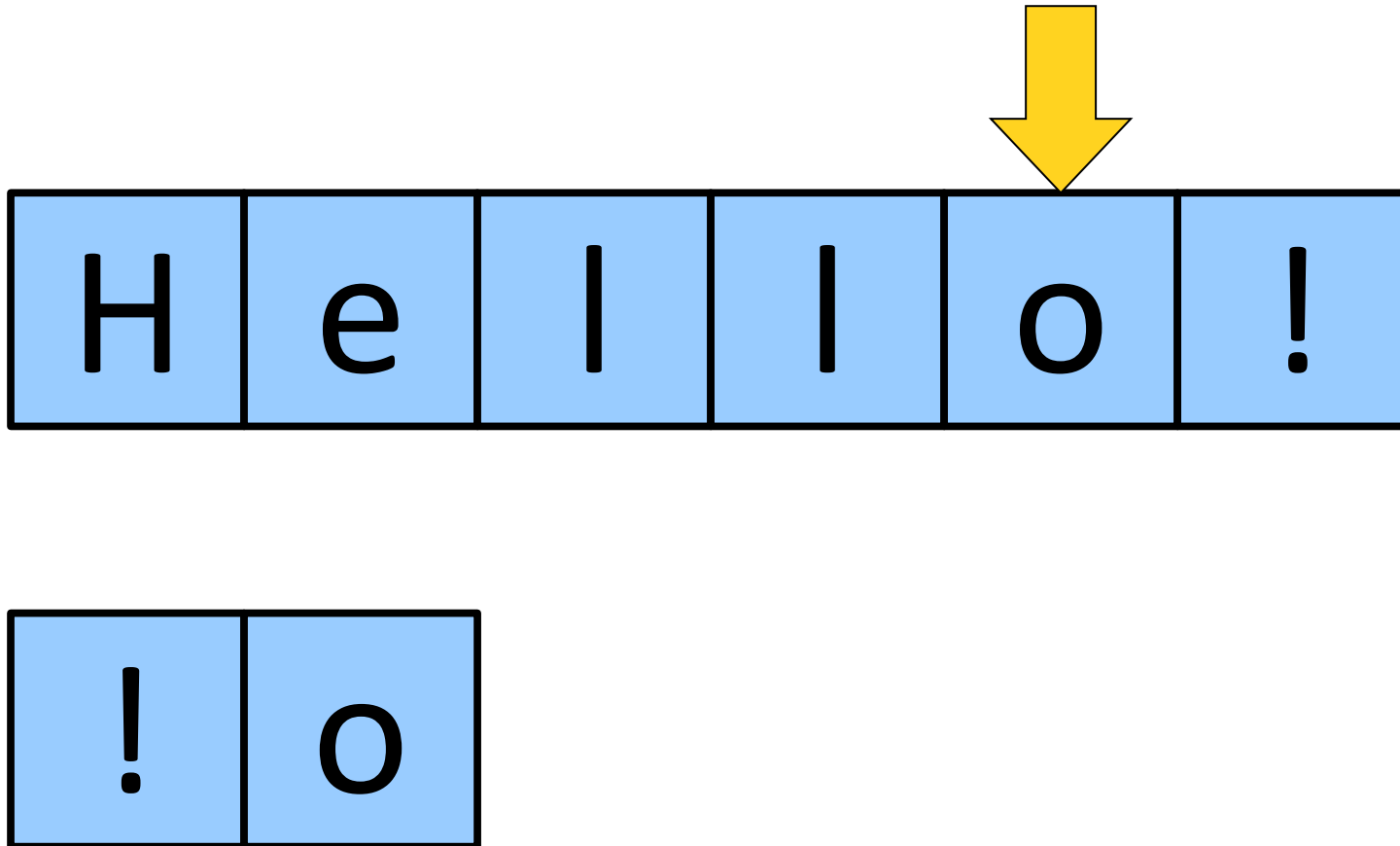
Reversing a String



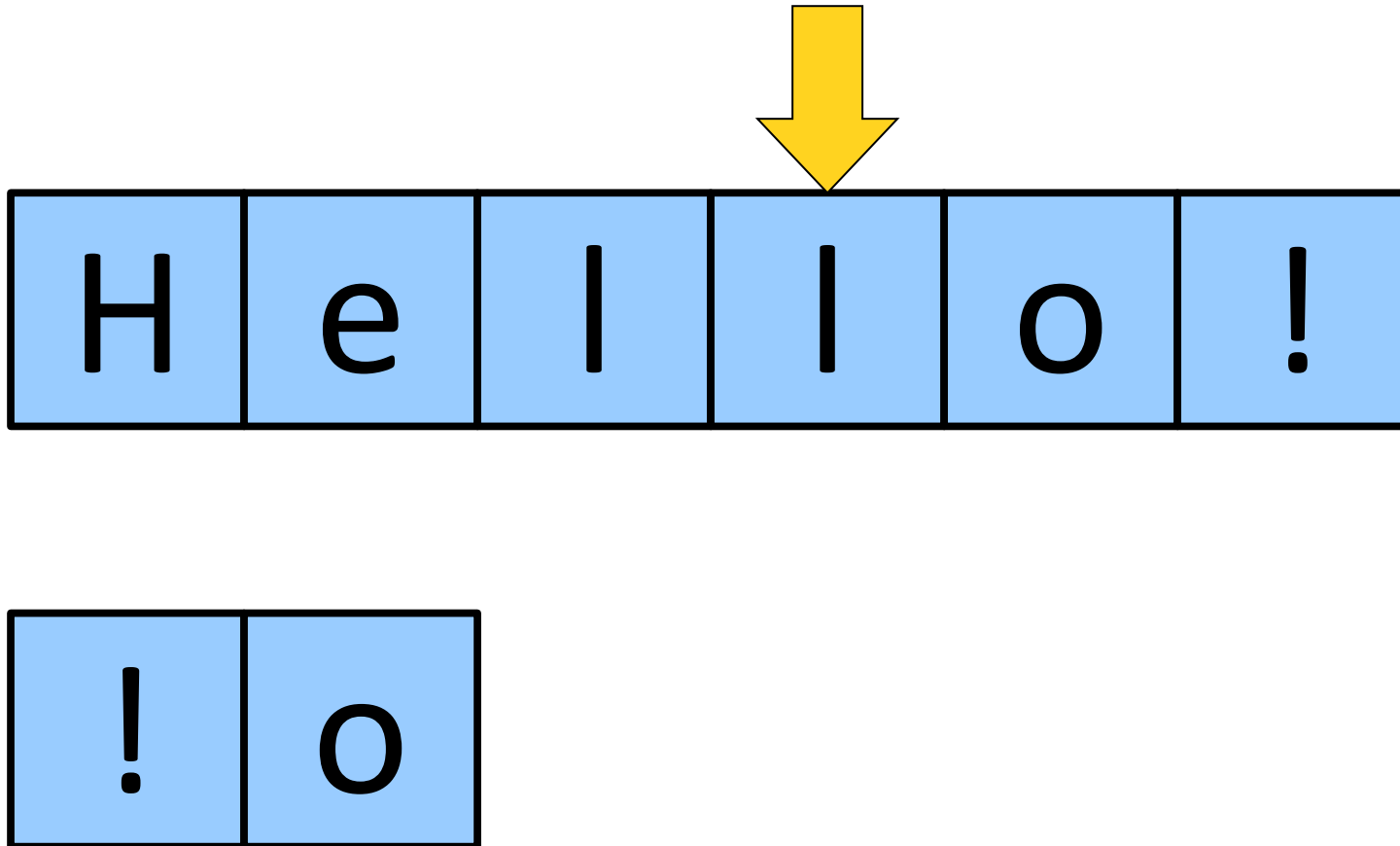
Reversing a String



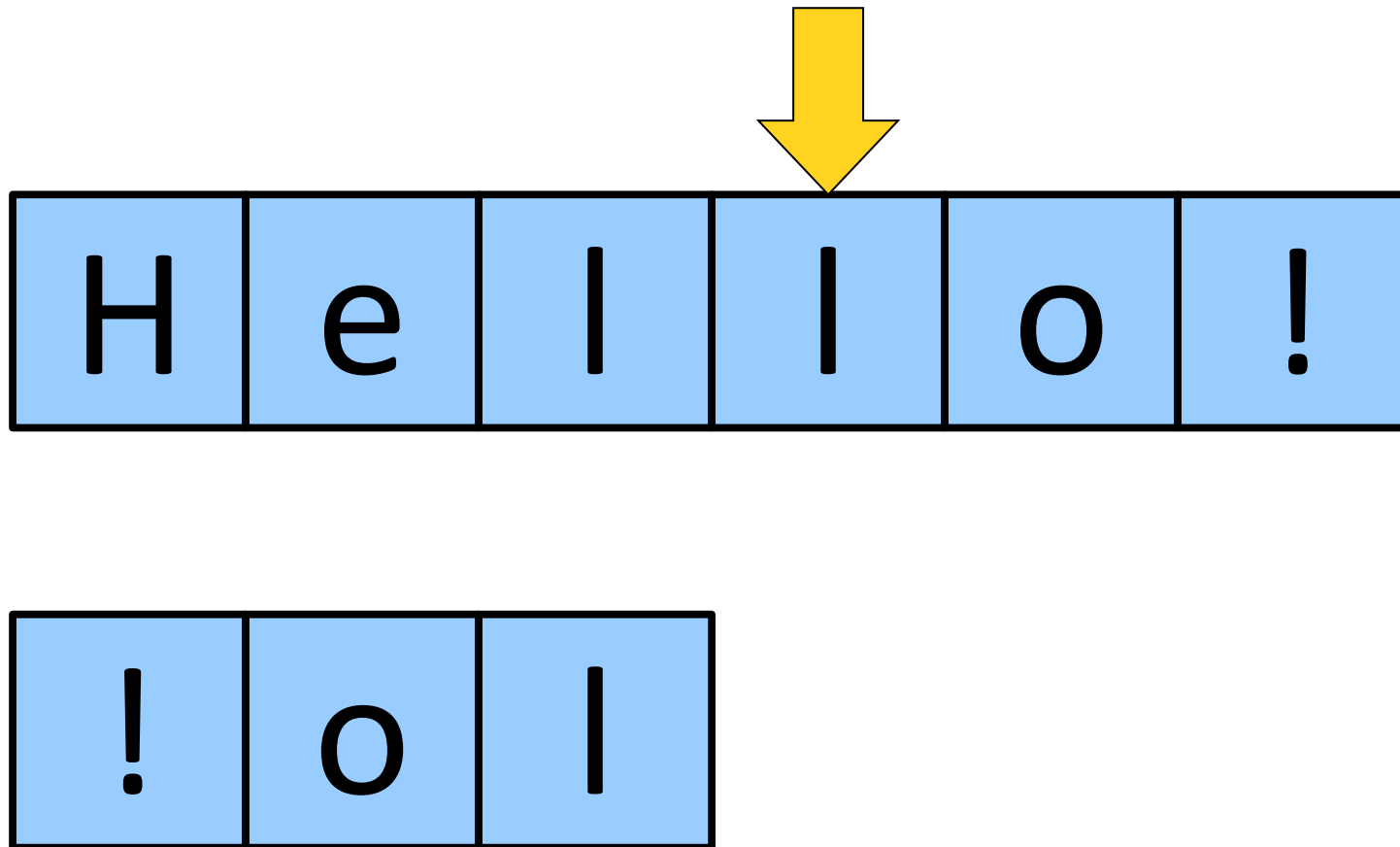
Reversing a String



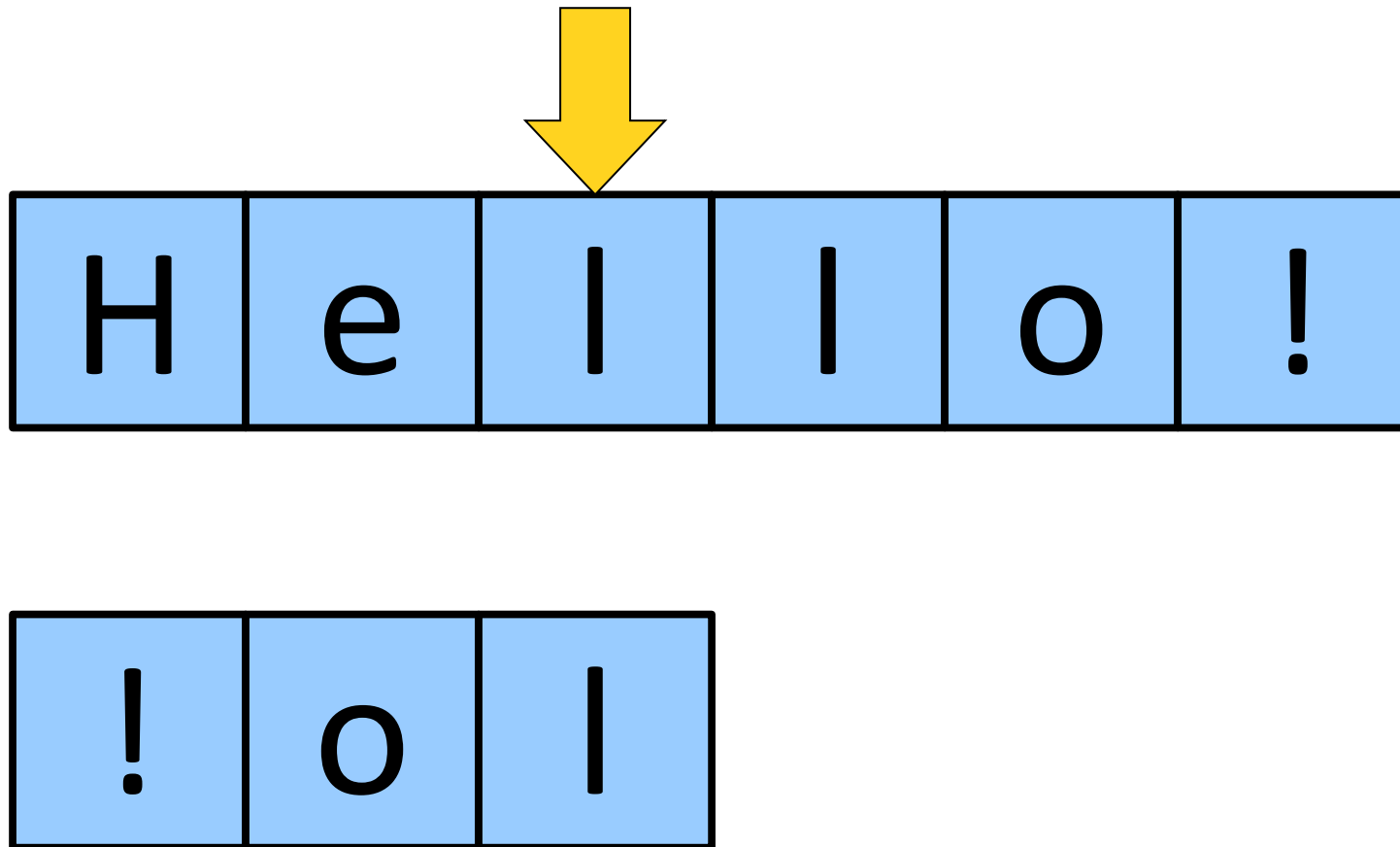
Reversing a String



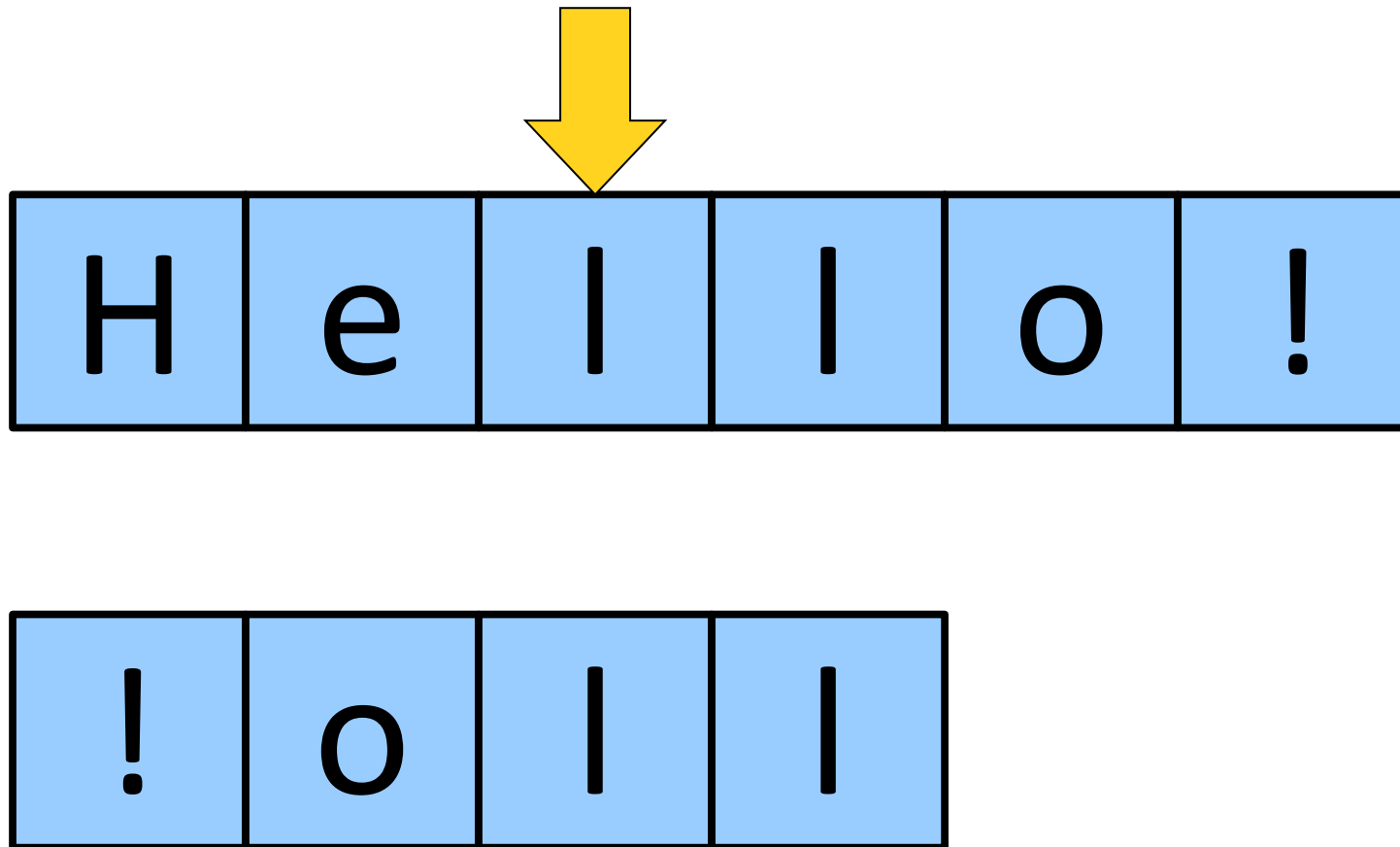
Reversing a String



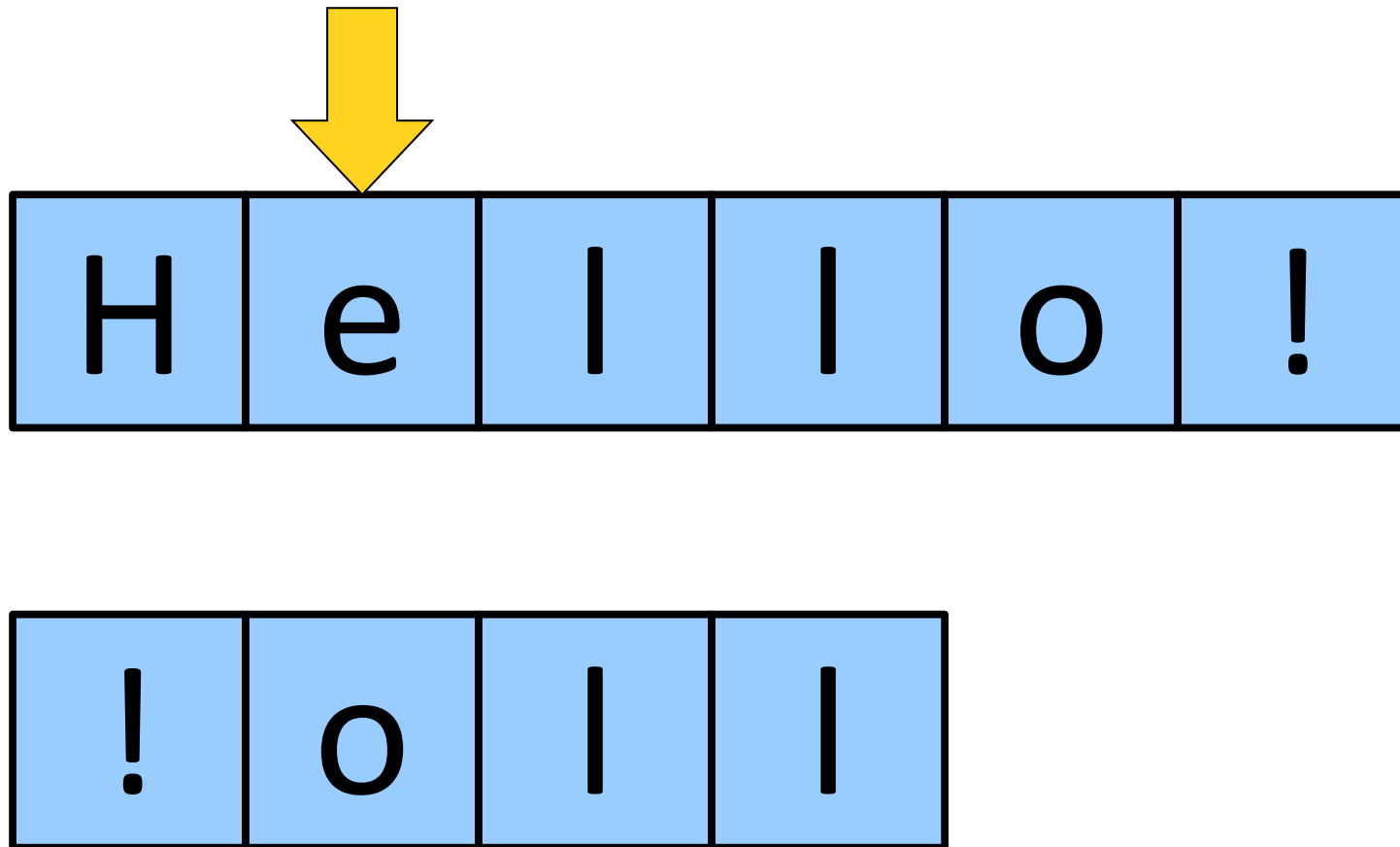
Reversing a String



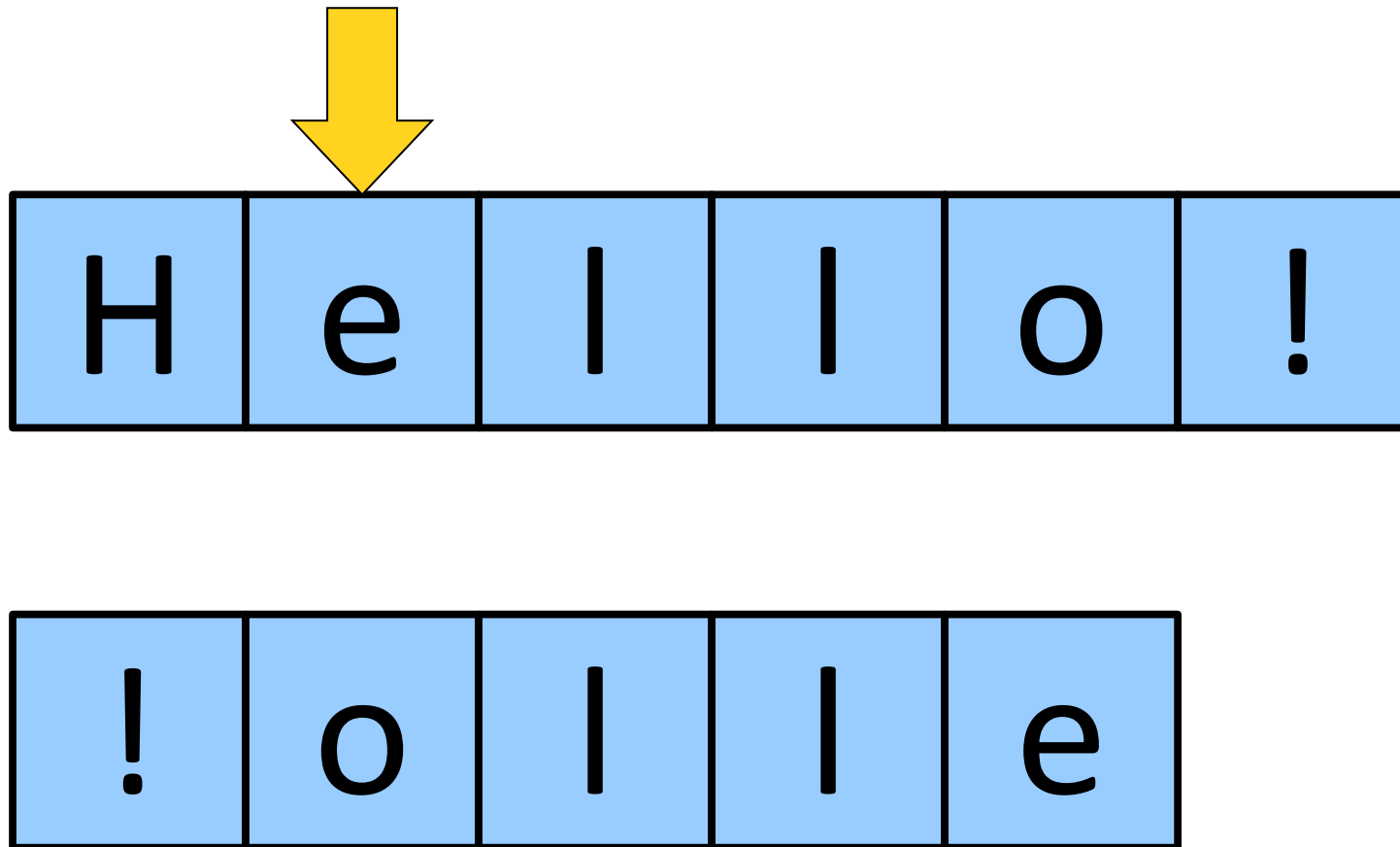
Reversing a String



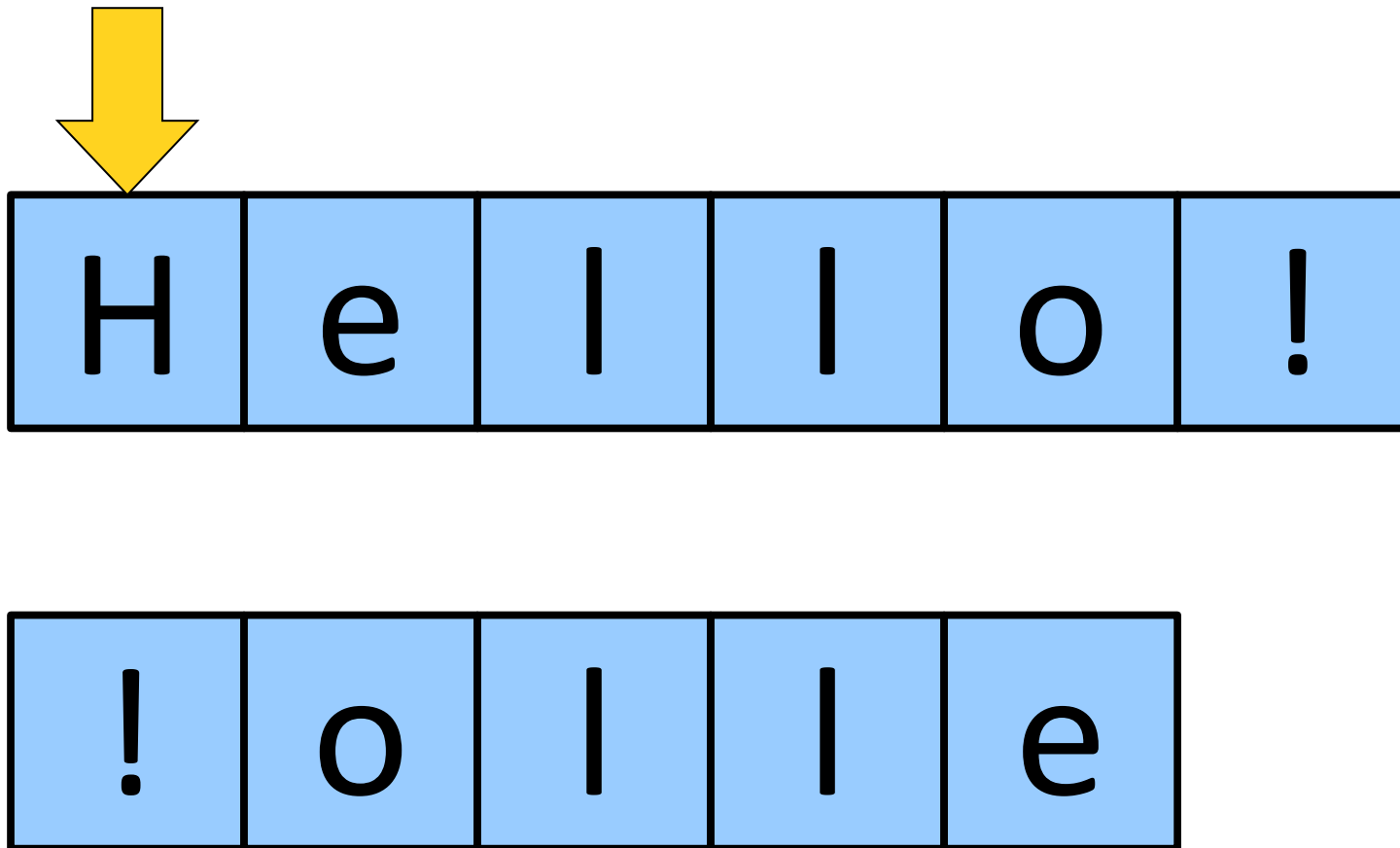
Reversing a String



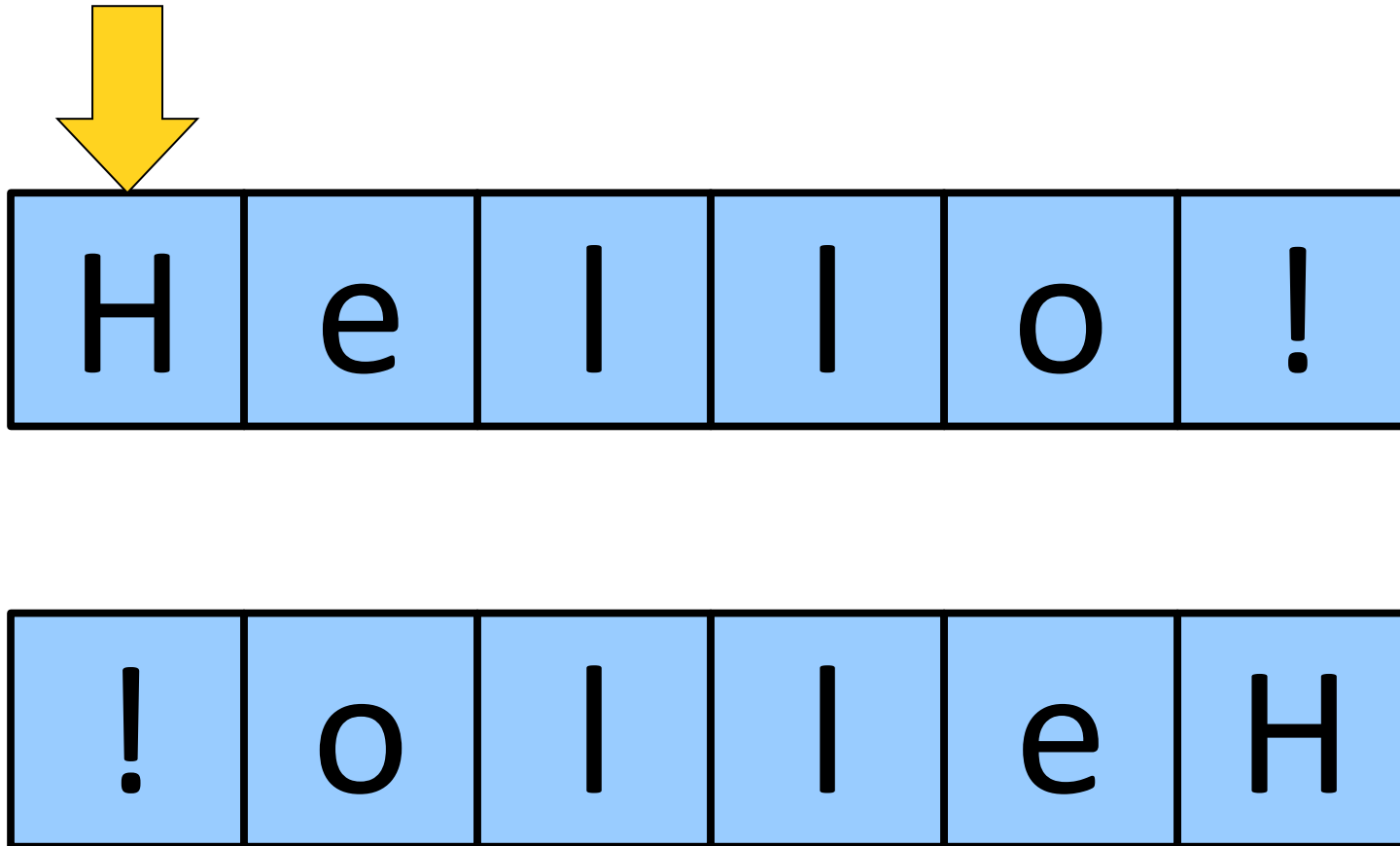
Reversing a String



Reversing a String



Reversing a String



Reversing a String

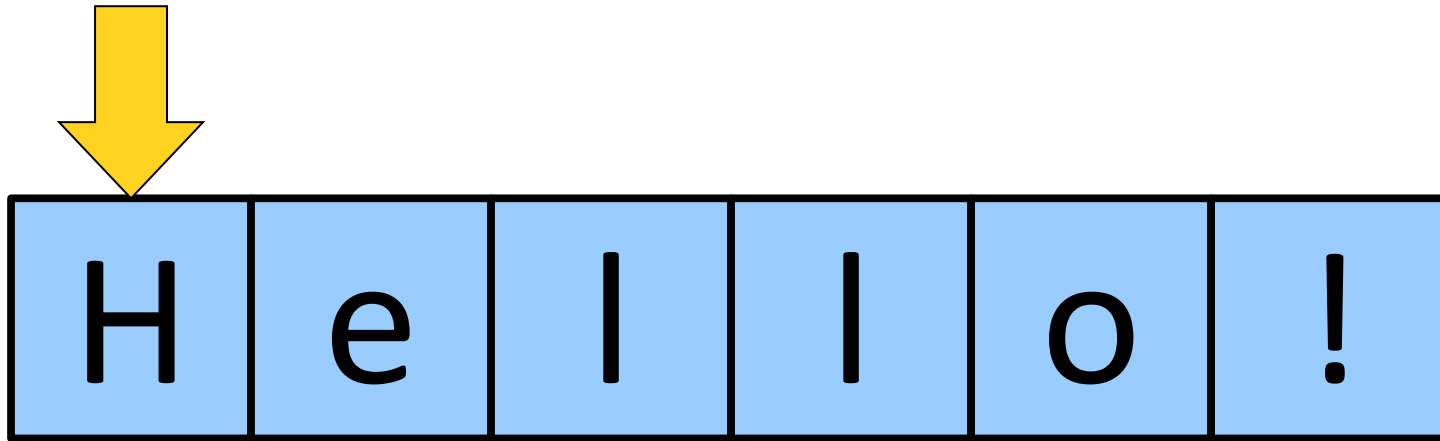
```
result = ''  
# counts from len(str)-1 down to 0 (including 0)  
for i in range(len(str)-1, -1, -1):  
    ch = str[i]  
    result += ch  
return result
```

H	e	l	l	o	!
---	---	---	---	---	---

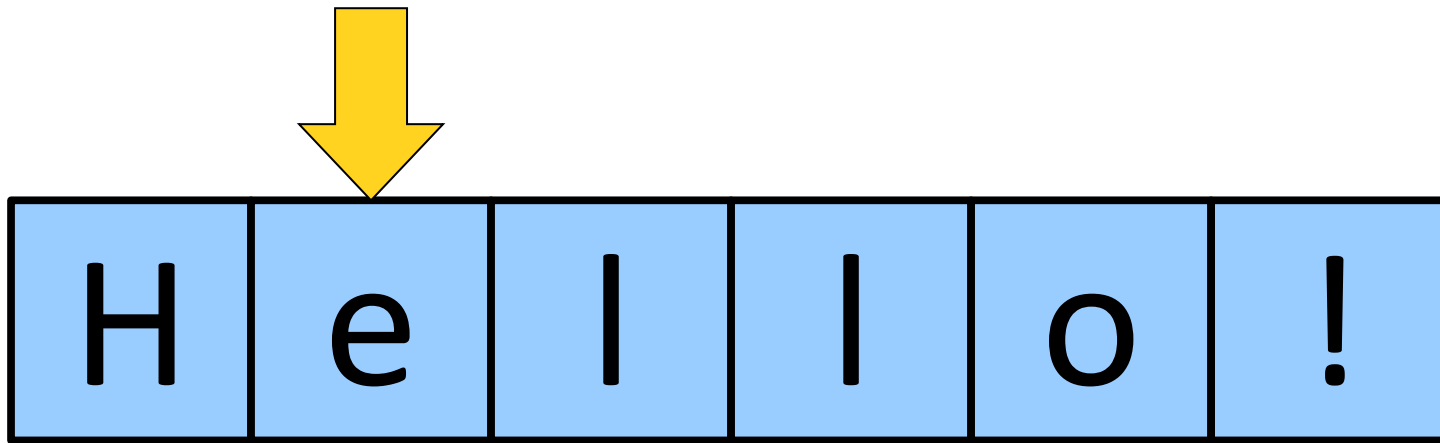
!	o	l	l	e	H
---	---	---	---	---	---



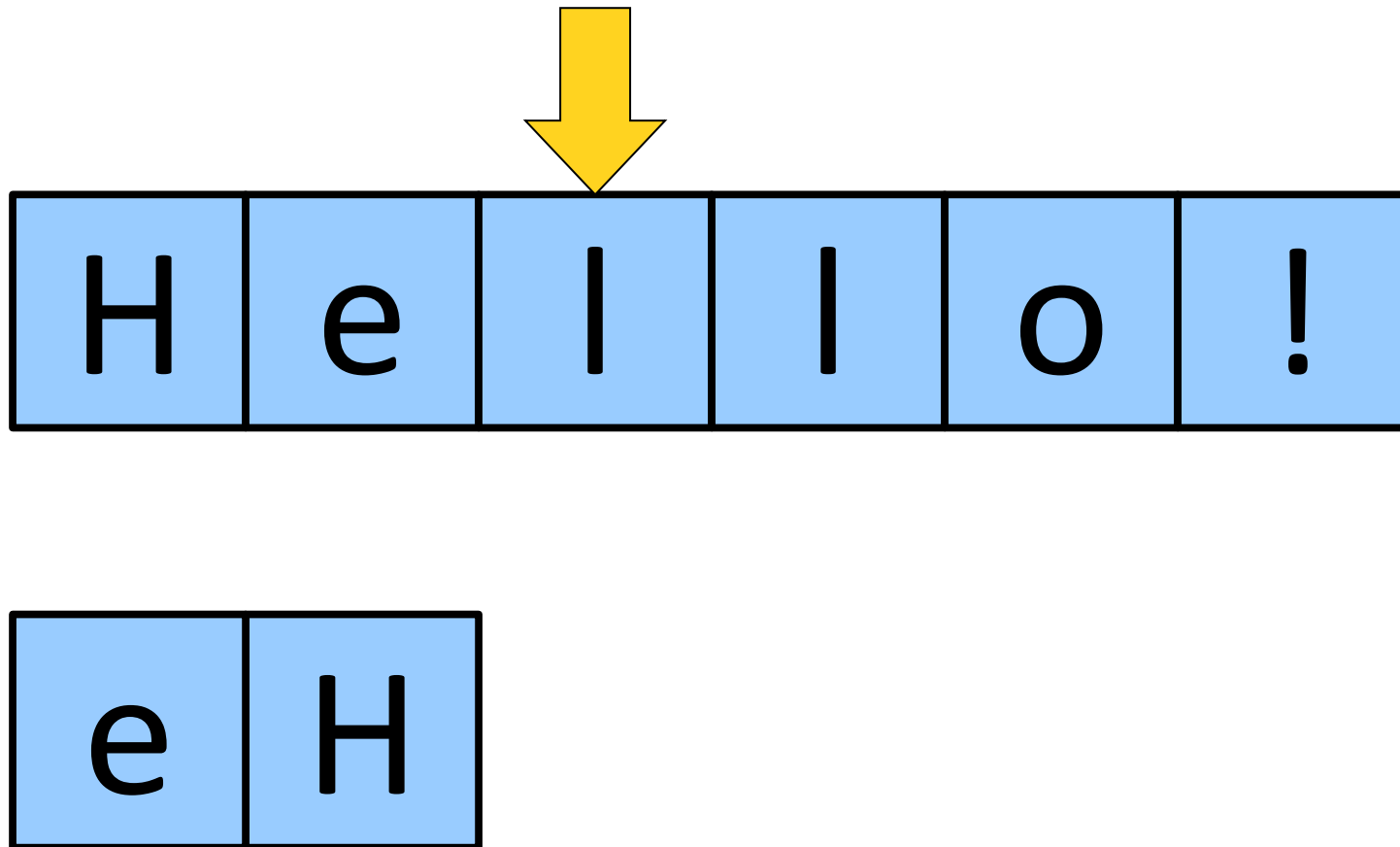
Reversing a String



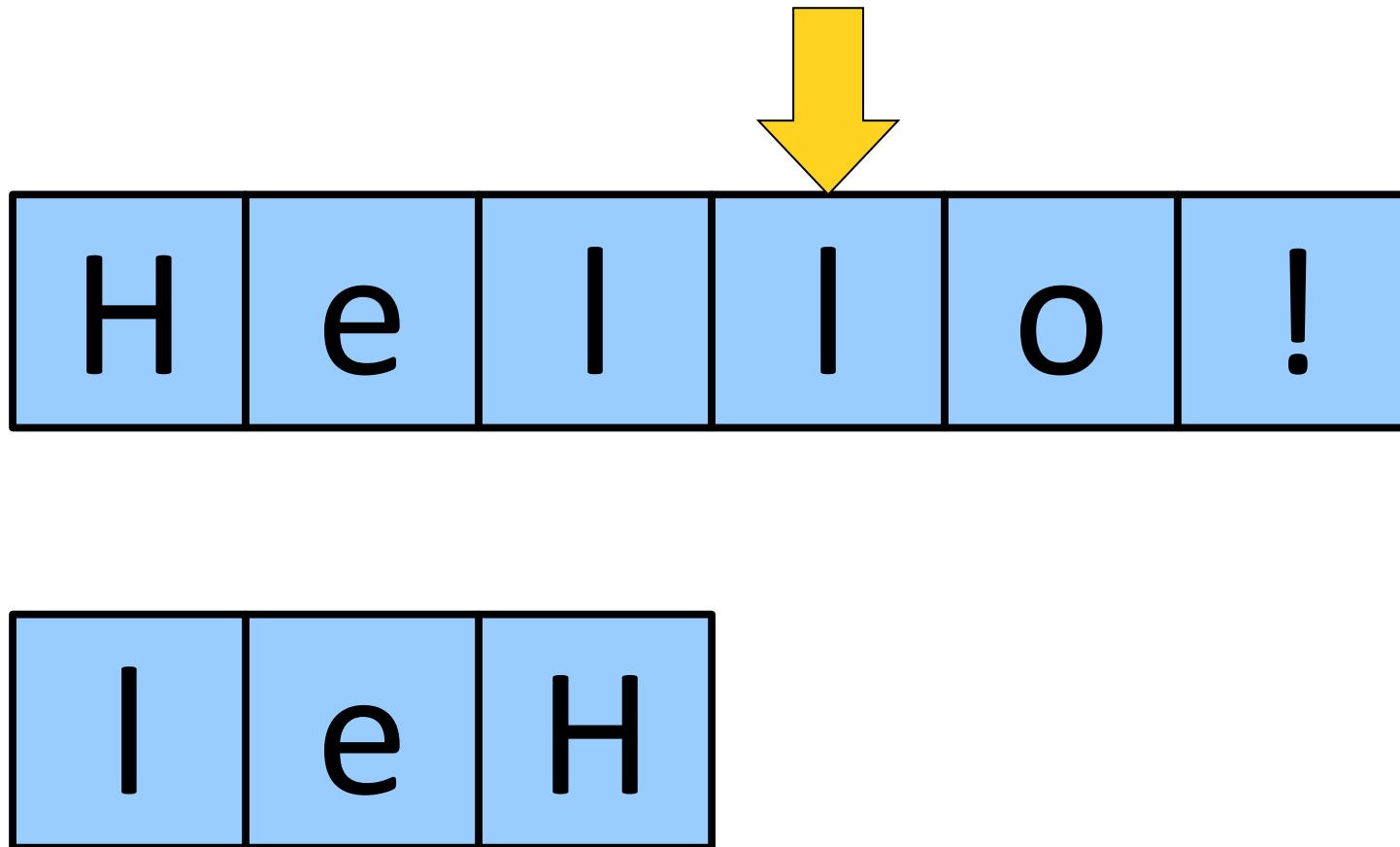
Reversing a String



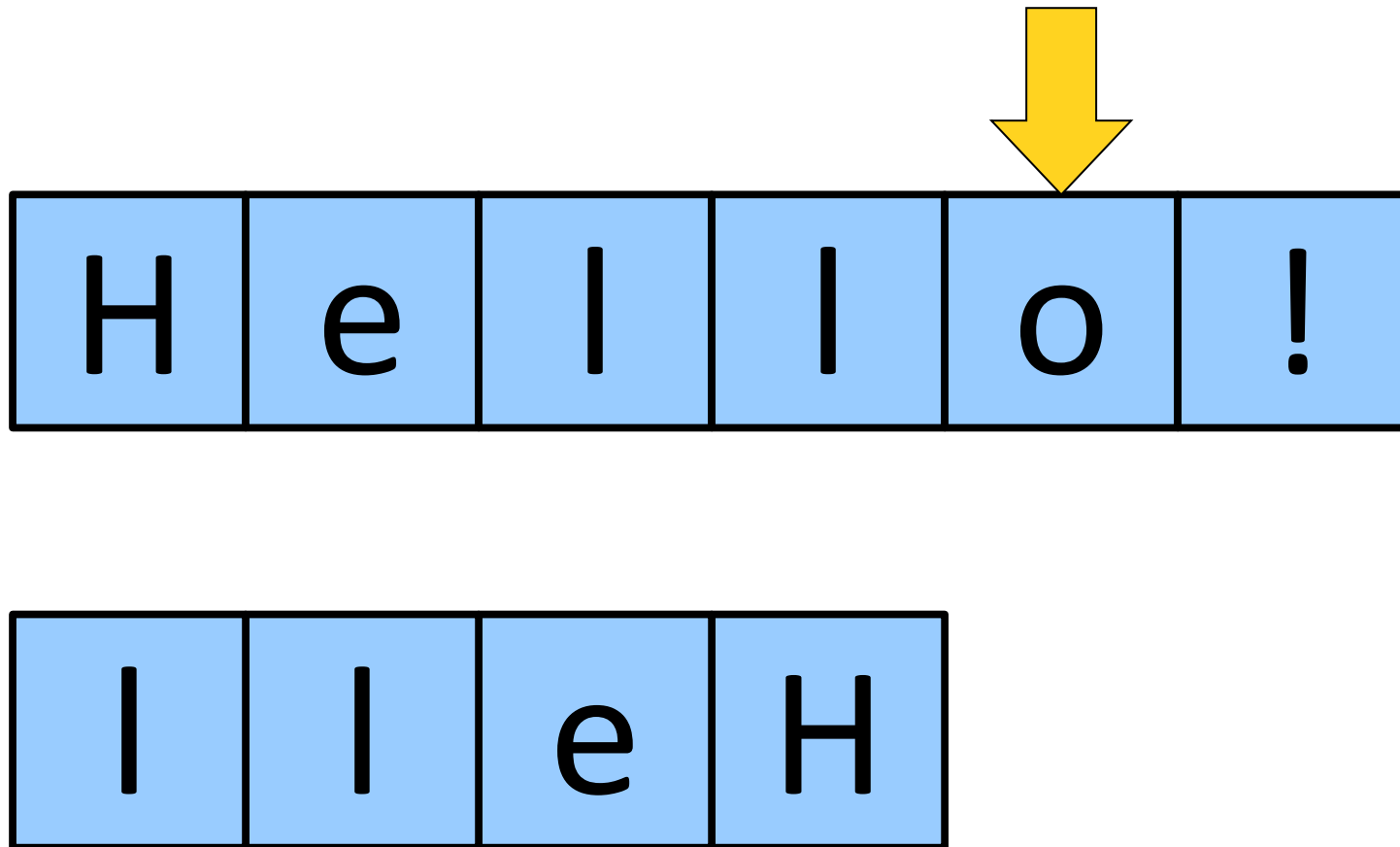
Reversing a String



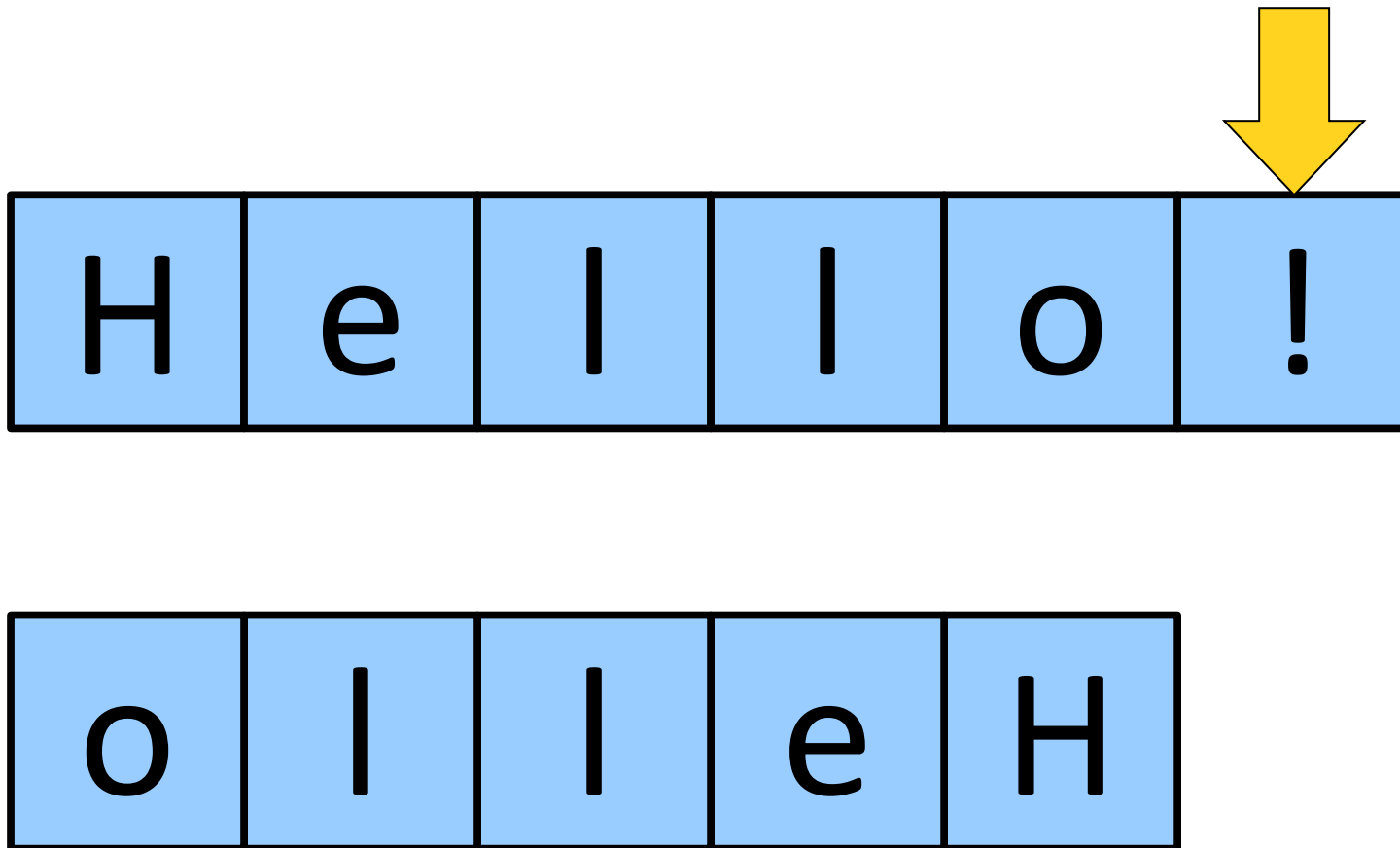
Reversing a String



Reversing a String



Reversing a String



Reversing a String

H	e	l	l	o	!
---	---	---	---	---	---

!	o	l	l	e	H
---	---	---	---	---	---

reverse_string

```
def main():
```

```
    def reverse_string(str):
```

```
        result = ""
```

```
        for i in range(len(str)):
```

```
            result = str[i] + result
```

```
        return result
```

result

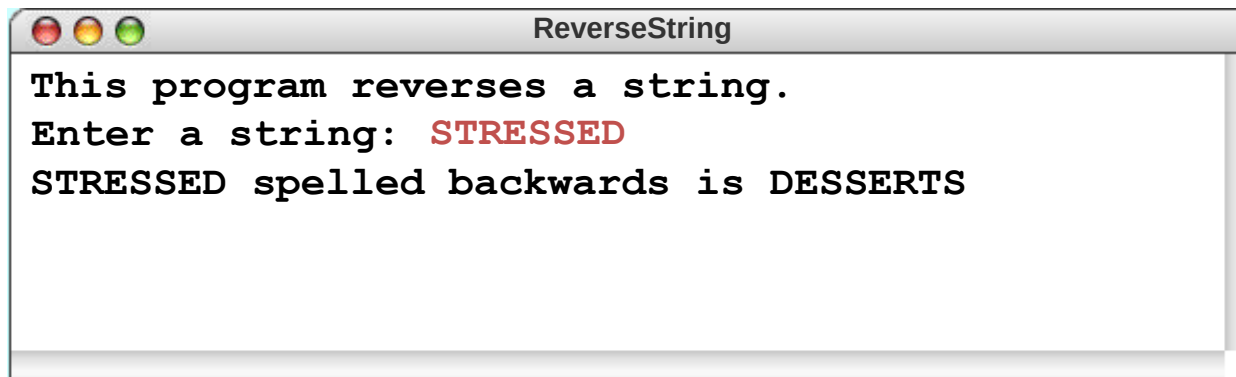
DESSERTS

str

STRESSED

i

8



reverse_string redux

```
def reverse_string(str):  
    result = ""  
    for i in range(len(str)):  
        result = str[i] + result  
    return result
```

Swiss army knife
pattern

```
def reverse_string_v2(str):  
    result = ""  
    for ch in str:  
        result = ch + result  
    return result
```

Medium bear

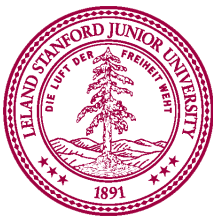
```
def reverse_string_v3(str):  
    """
```

I'll have
another slice!!

This uses the slice operator with no
start, no end, and a step of -1, so
slice reverses. It's better than pizza!

```
    """
```

```
    return str[::-1]
```



Learning Goals

1. Learn about operations on string
2. Write code that manipulates strings

