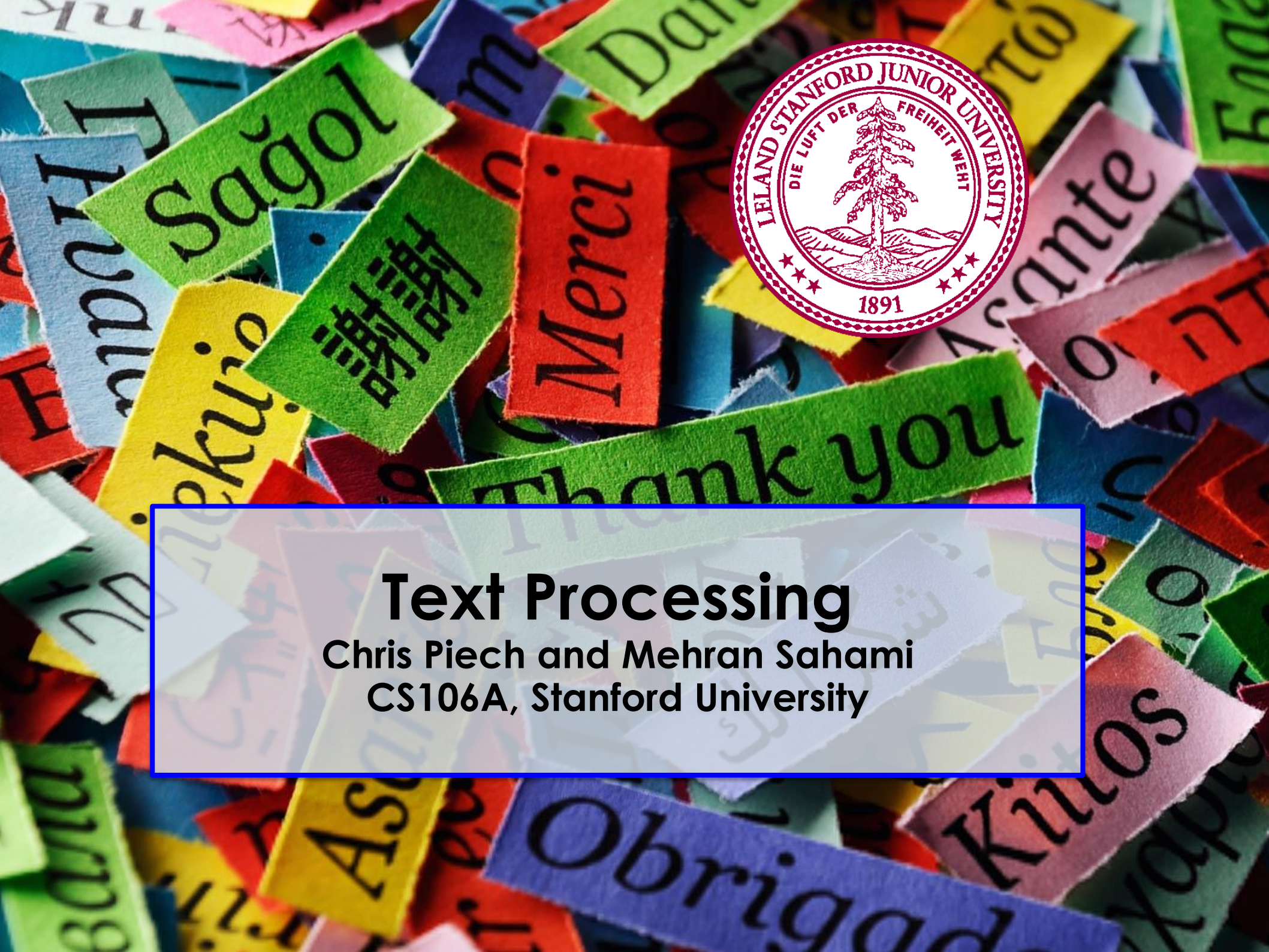
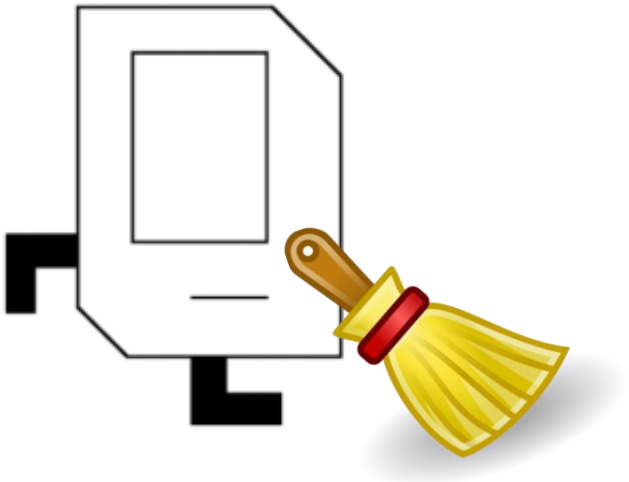



# Text Processing

Chris Piech and Mehran Sahami
CS106A, Stanford University

# Housekeeping

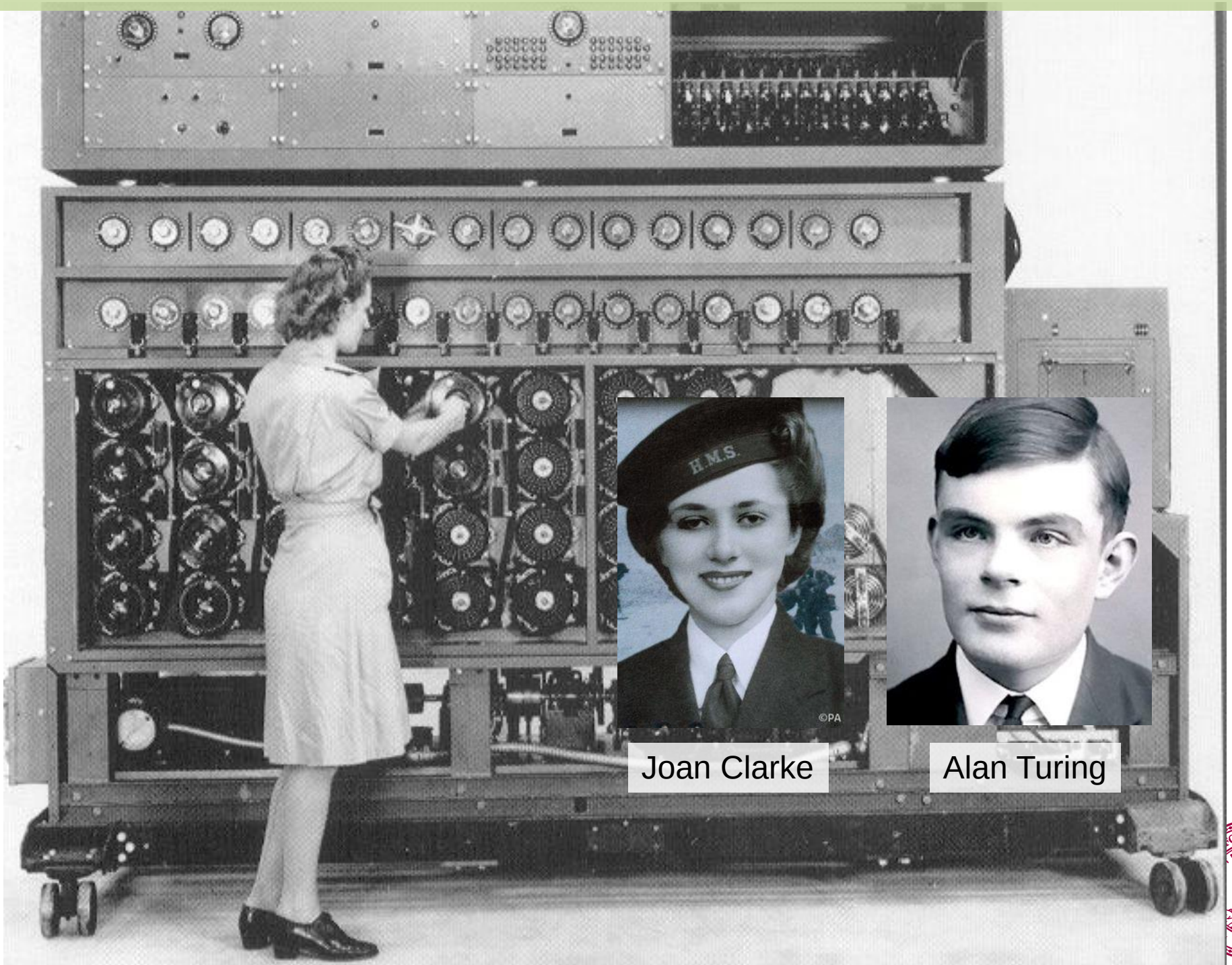


- Happy Friday!
- Diagnostic regrade requests due by Saturday (Oct. 17) at 1pm
  - Fill out this form: https://forms.gle/S2L2vSS4UtJCx4Uu7
- Mid-quarter evaluations
  - Part of Assignment #4
  - We read all the feedback!
- Katie Creel will give a guest mini-lecture on ethical issues in character representations at the end of class

# Learning Goals

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

# Text Problem: Decryption



# Text Problem: Translation

*The spirit is willing but the flesh is weak.*

↓

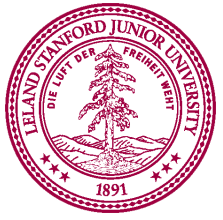
(Russian)

↓

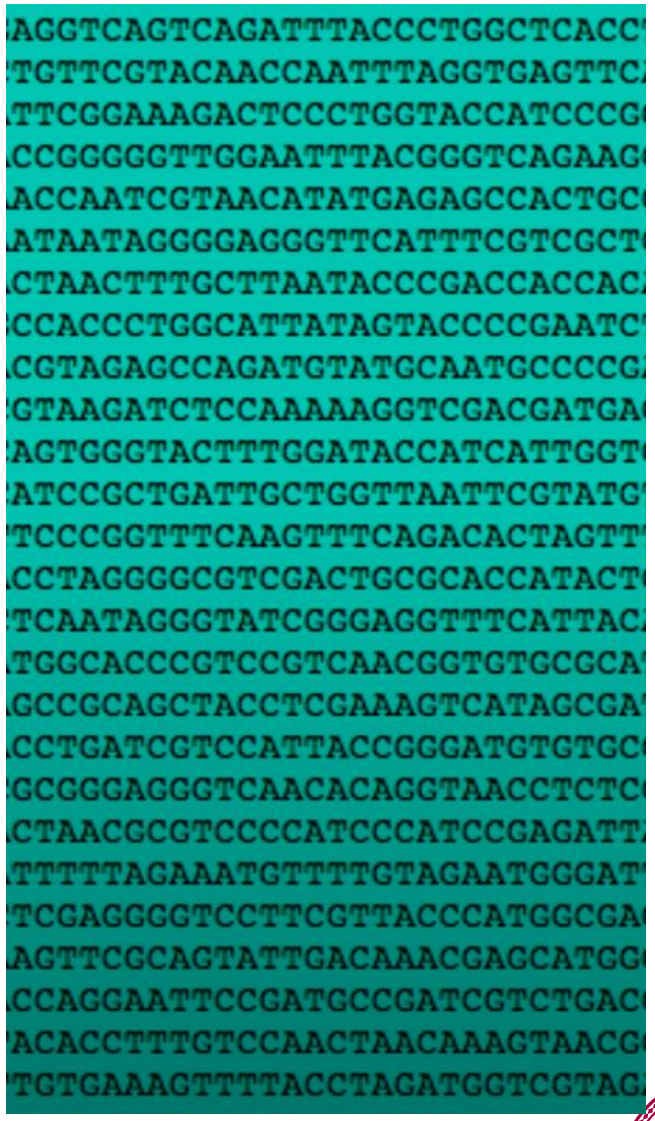
*The vodka is good but the meat is rotten.*

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





# Text Problem: DNA Analysis

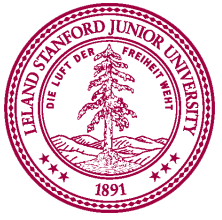
How is text represented?

# 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)
```

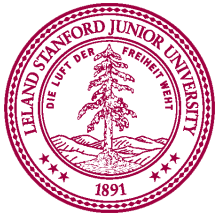




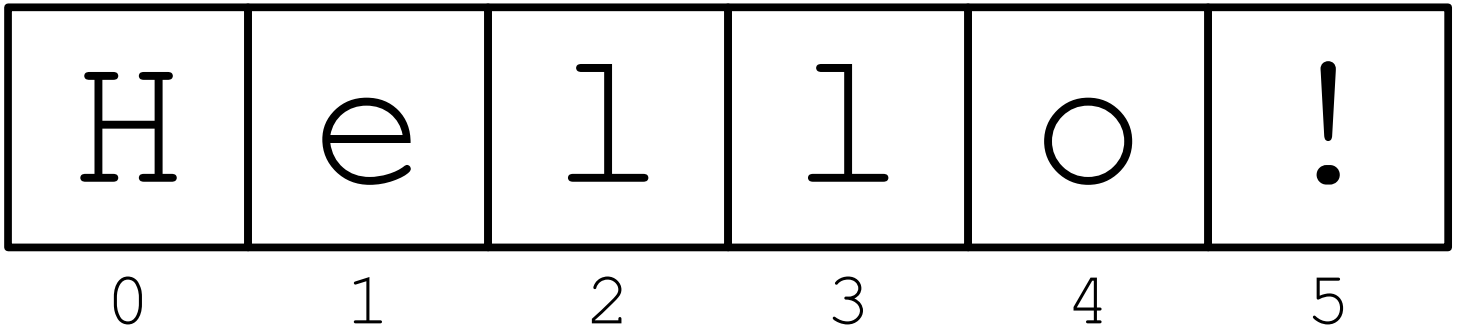
```
def main():
    text = "hello!"
```

H e l l o !



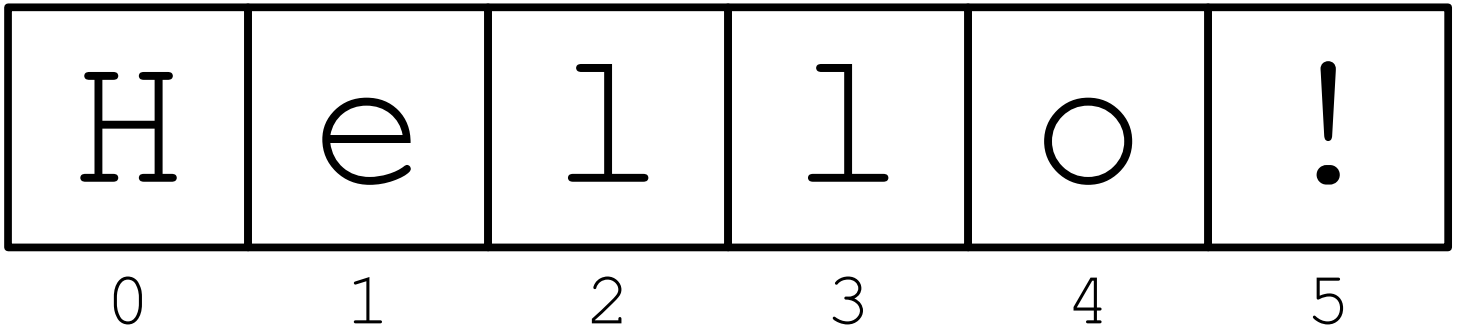


```
def main():
    text = "hello!"
```




Indexed starting from 0

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



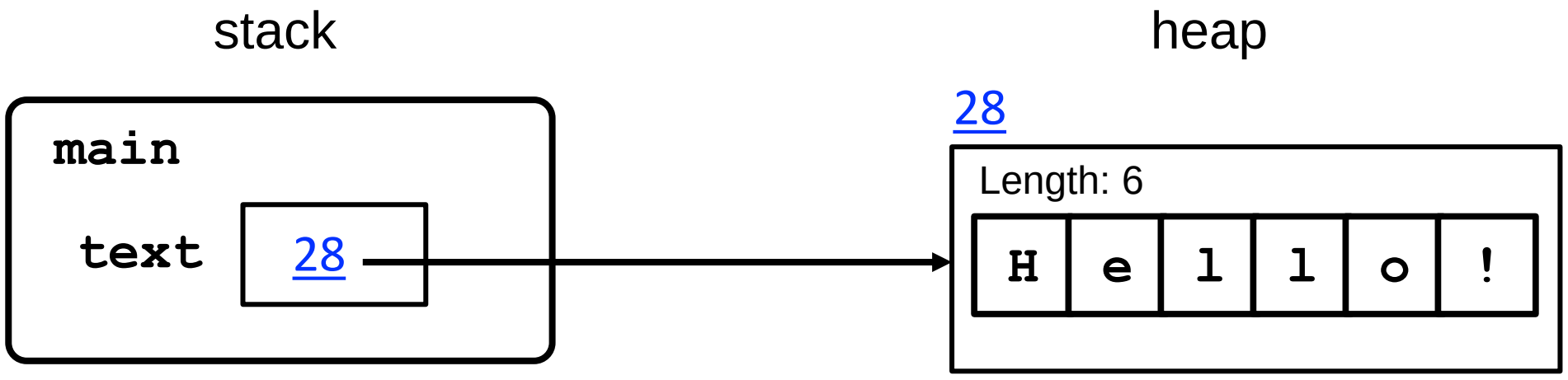

***text*[*index*]**

All characters in a string have an index.

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

# How it is actually stored

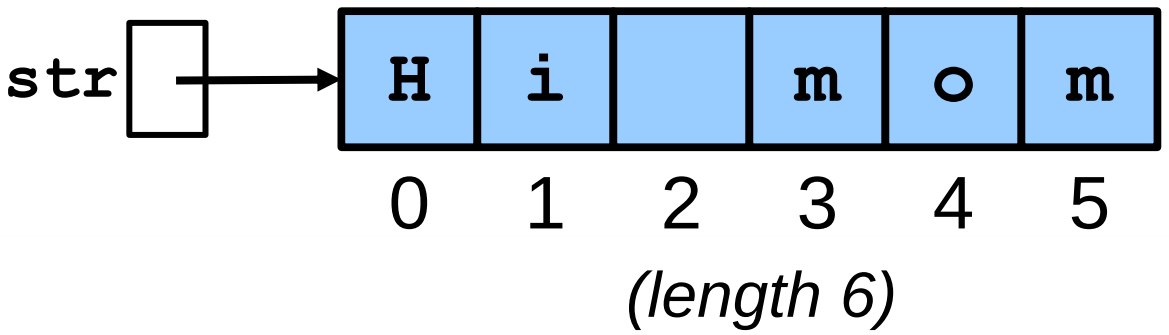
```
def main():
    text = "hello!"
```



# 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)
```

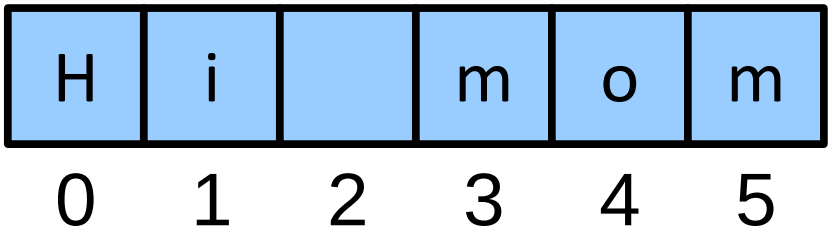
Console

# 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

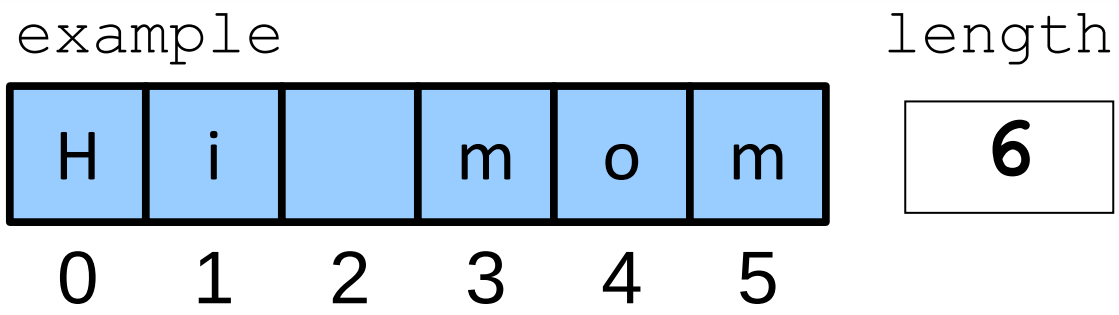
example



# 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

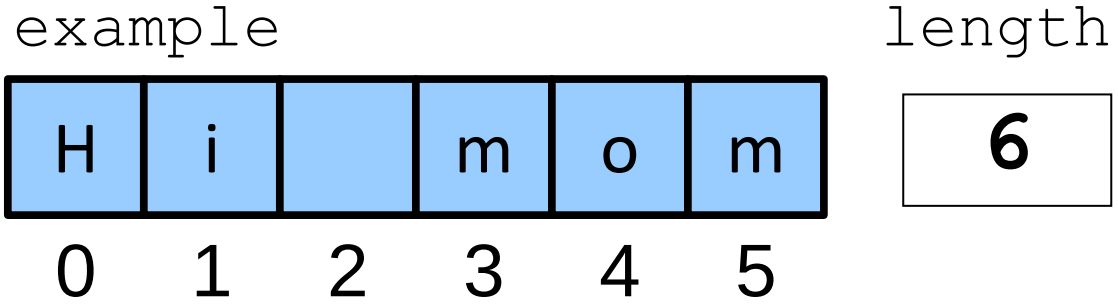


# String Functions

```
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

```
6
```

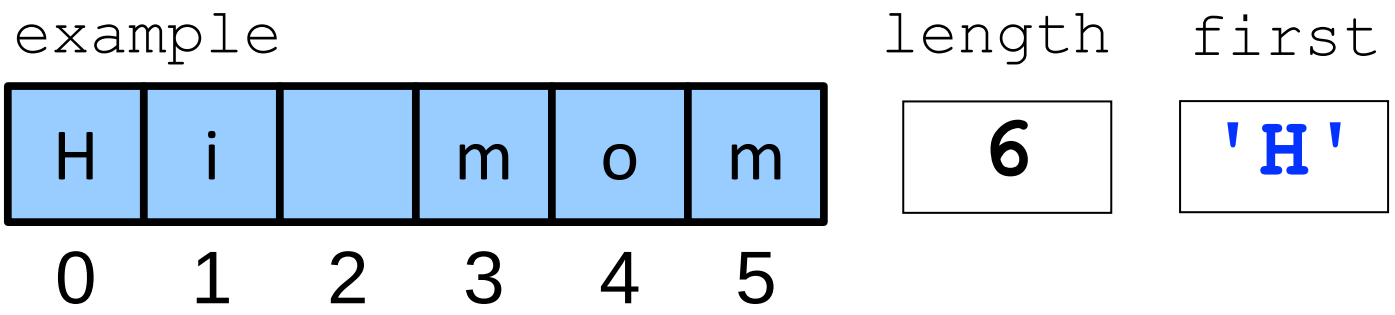


# String Functions

```
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
```

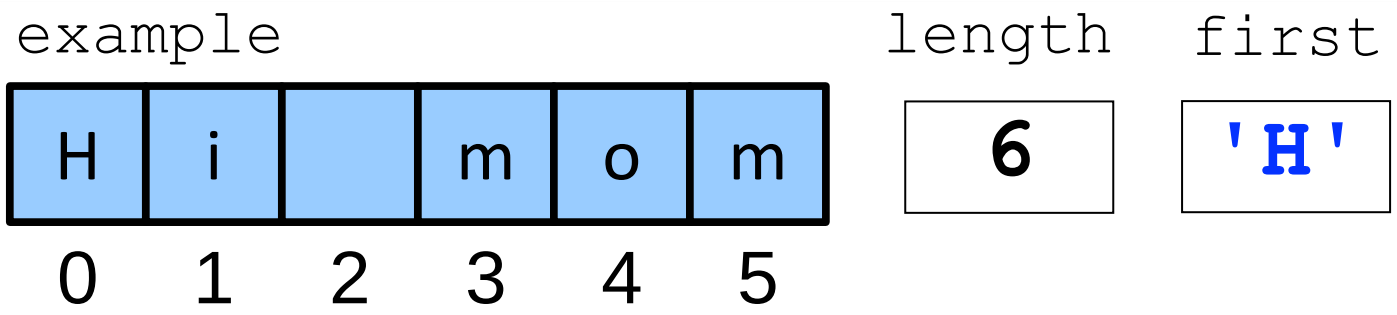


# String Functions

```
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
```

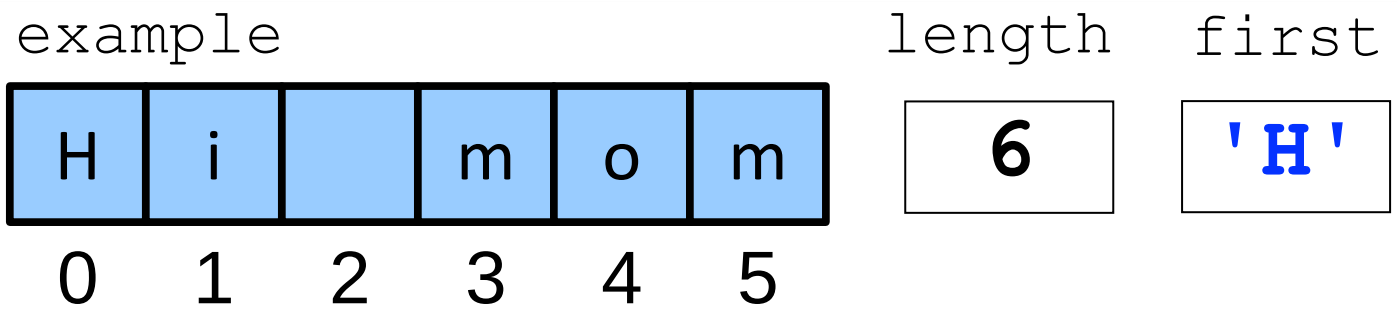


# String Functions

```
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
```

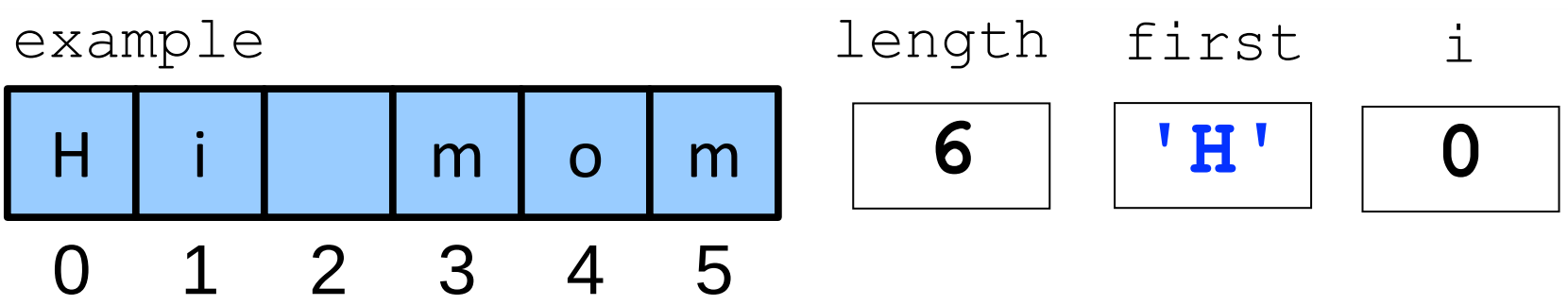


# String Functions

```
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
```

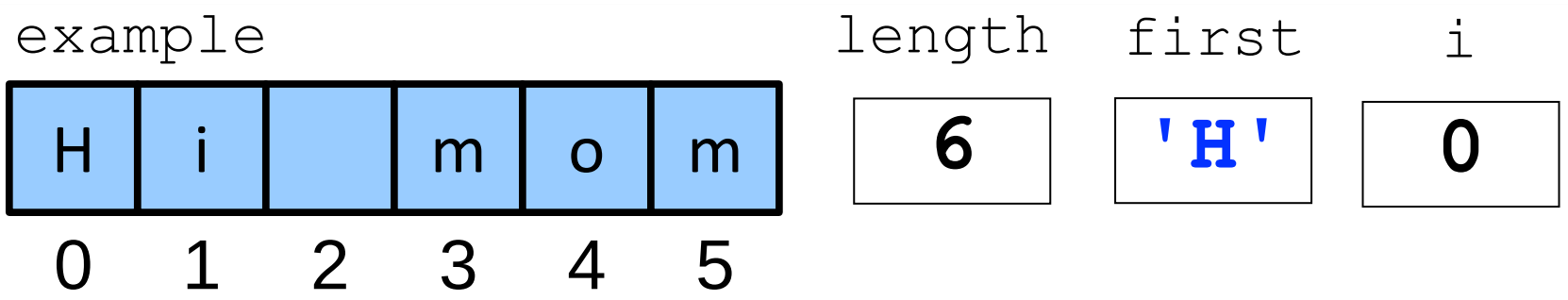


# String Functions

```
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
```



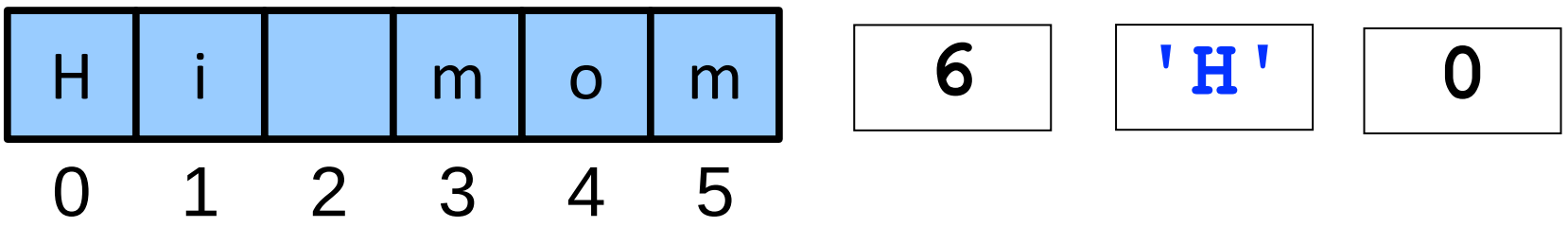
# String Functions

```
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
```

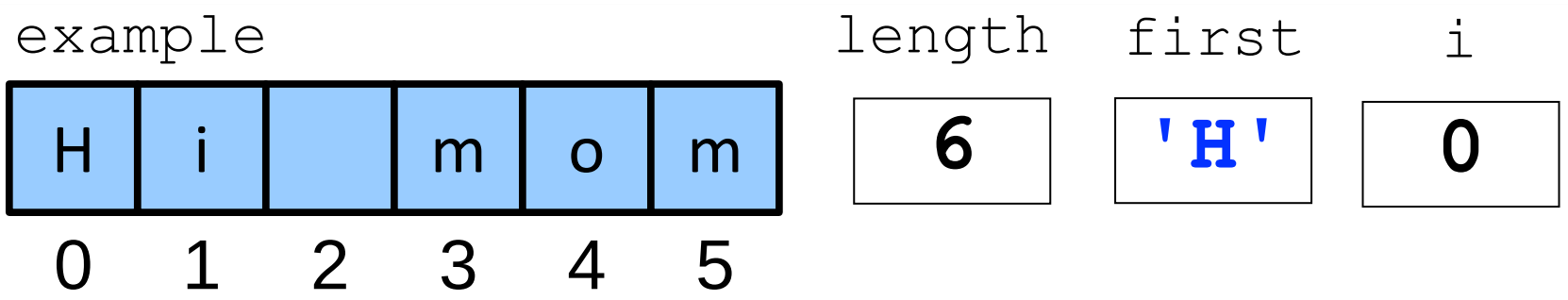


# String Functions

```
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
```

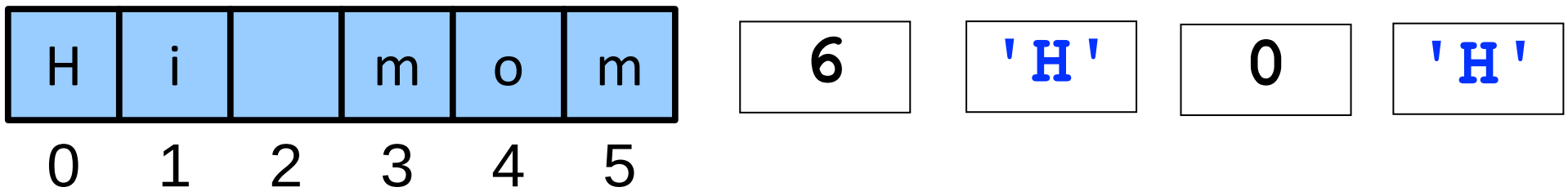


# String Functions

```
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
```

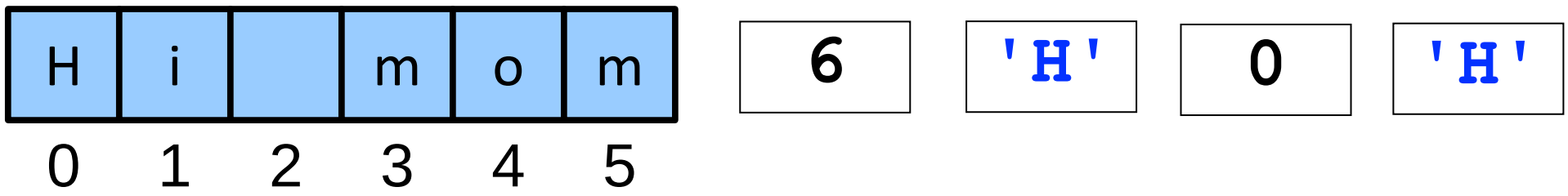


# String Functions

```
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
```

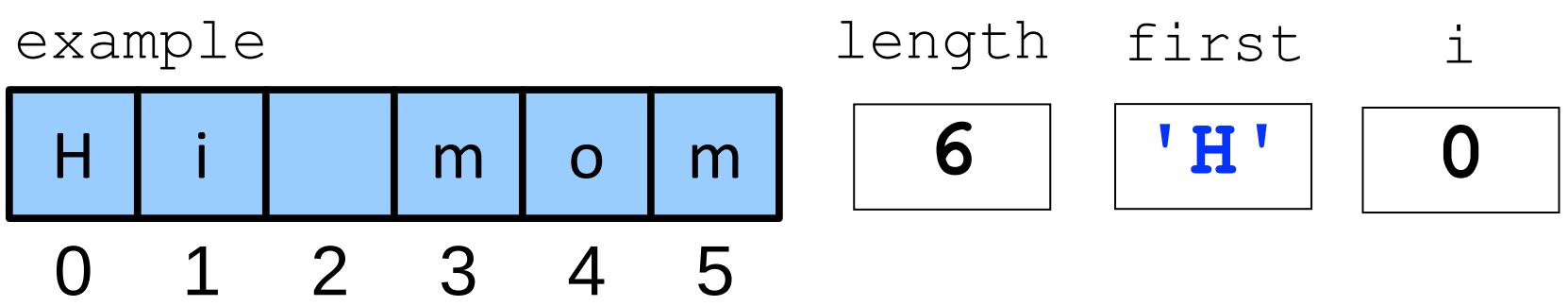


# String Functions

```
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
```

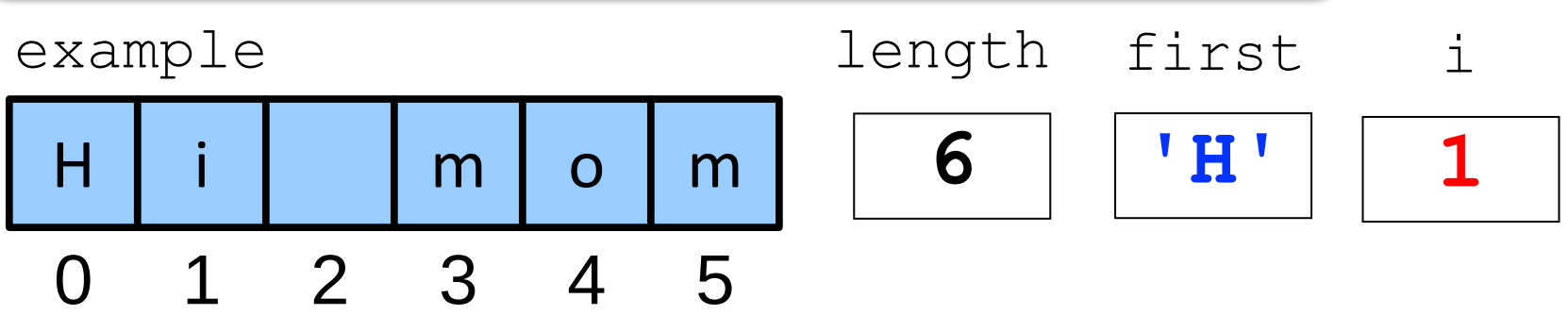


# String Functions

```
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
```

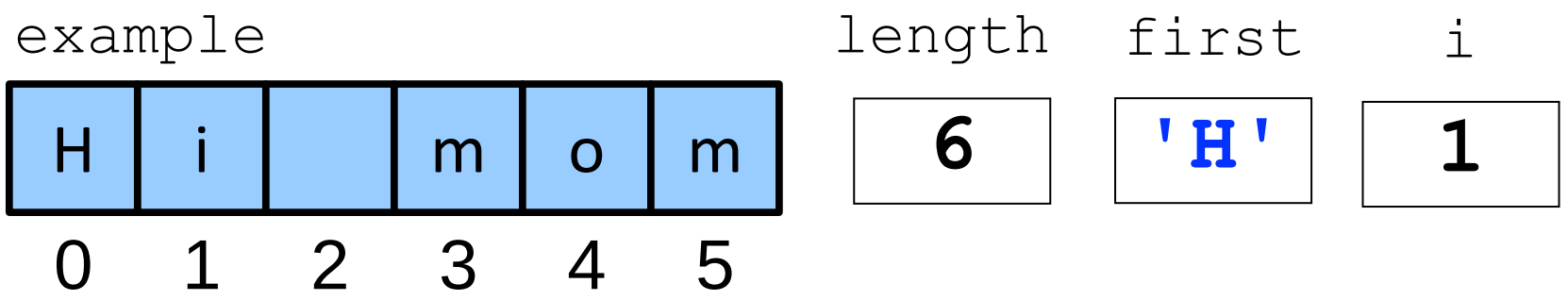


# String Functions

```
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
```

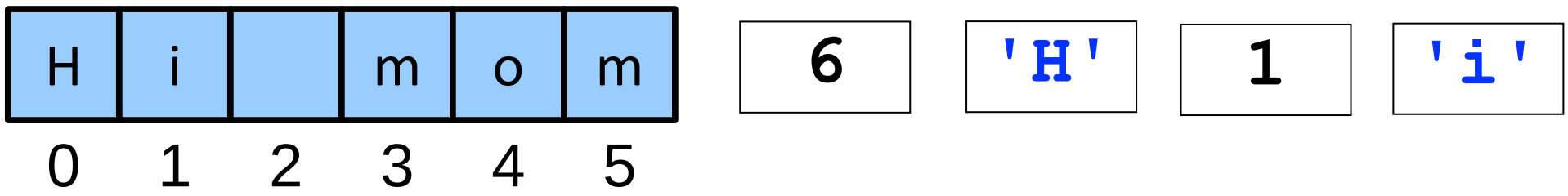


# String Functions

```
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
```

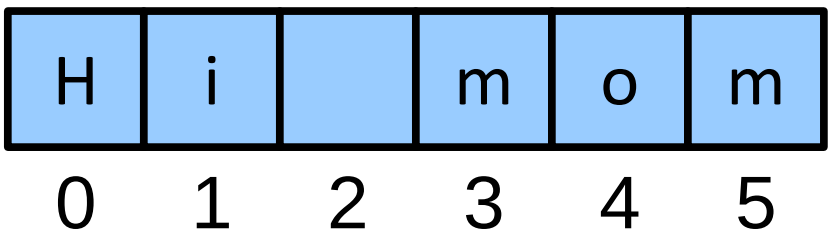


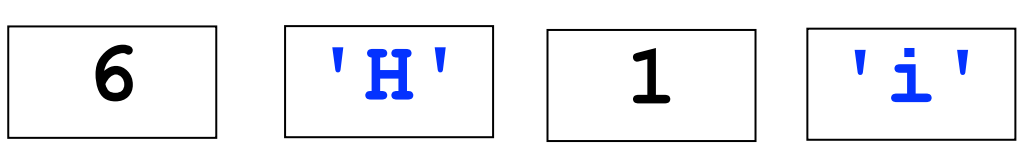
# String Functions

```
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
```



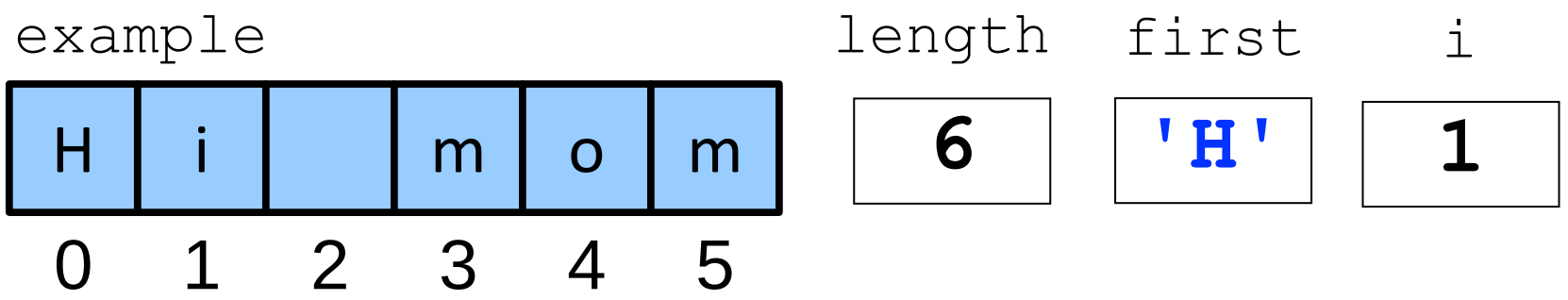



# String Functions

```
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
```

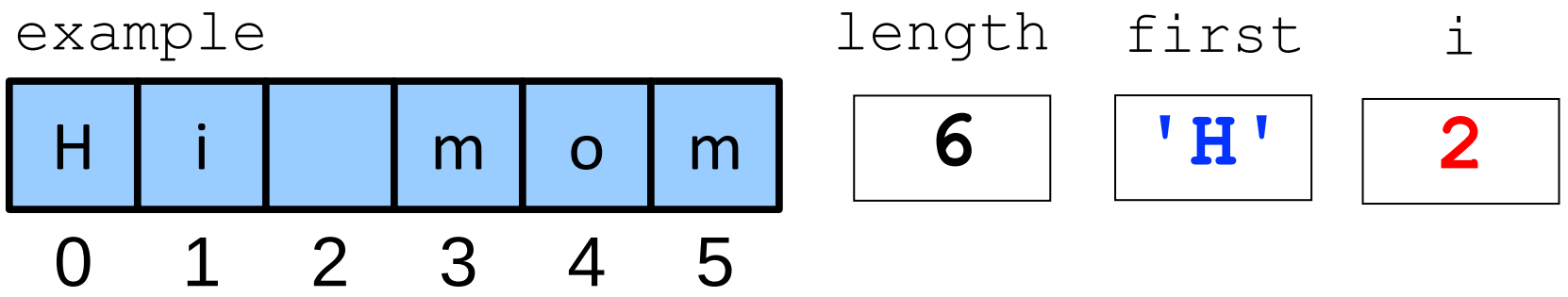


# String Functions

```
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
```

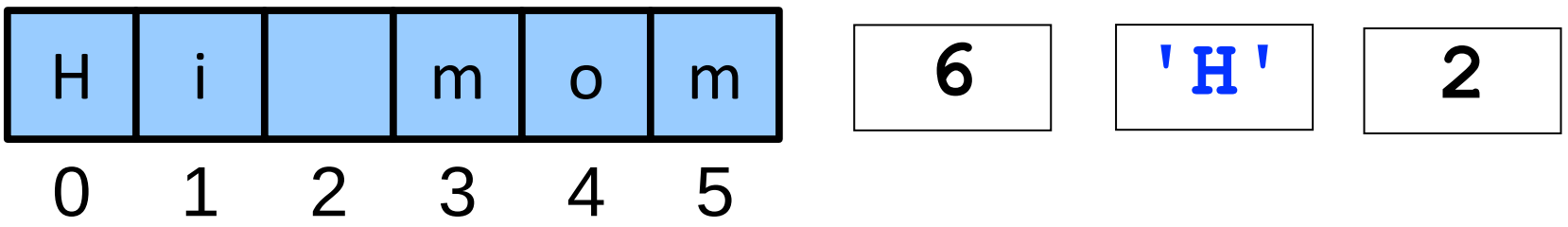


# String Functions

```
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
```

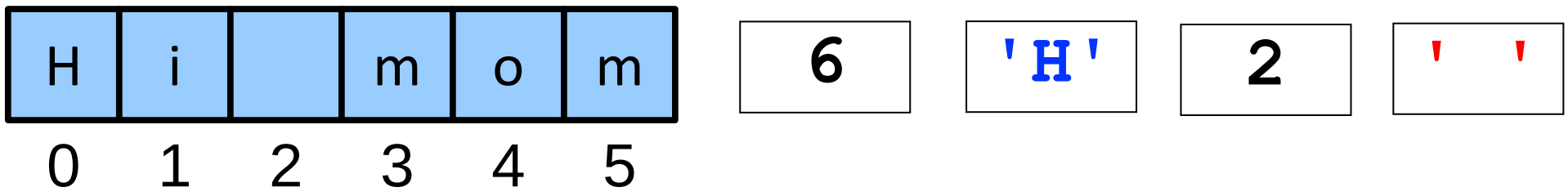


# String Functions

```
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
```

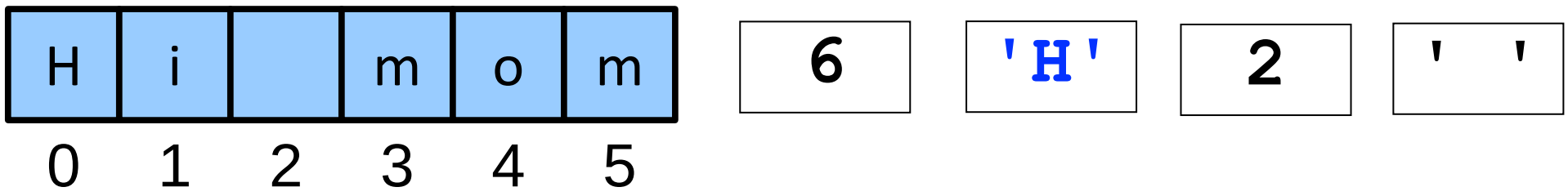


# String Functions

```
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
```

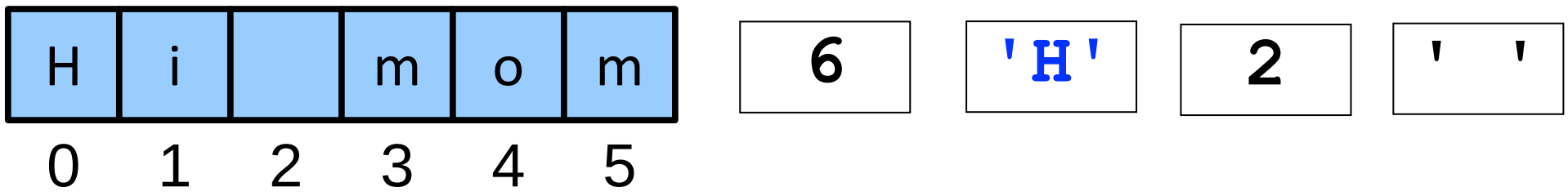


# String Functions

```
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
```

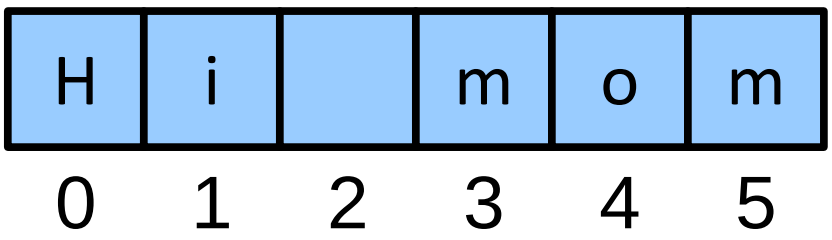


# String Functions

```
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
```




| length | first | i | ch |
|---|---|---|---|
| 6 | 'H' | 5 | 'm' |

A string is indexed just like a list!

Slices work too.

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

# Slicing Strings

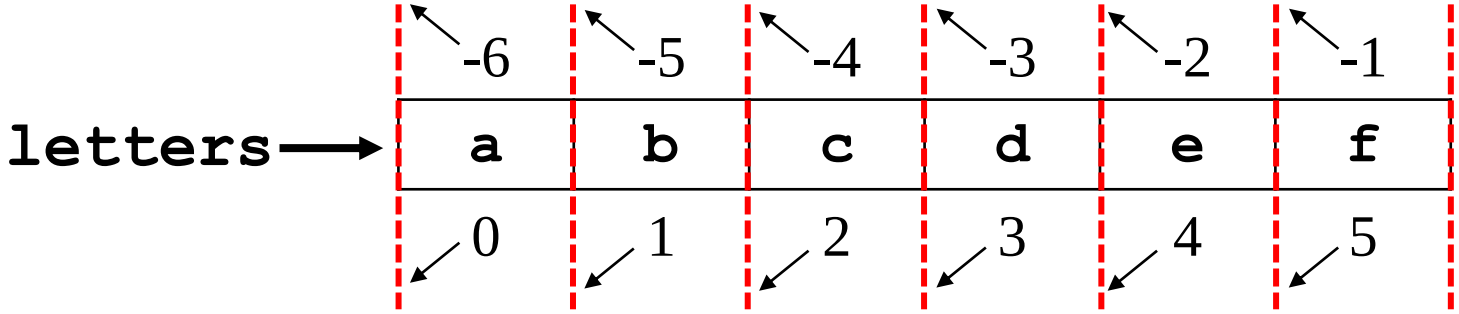
- Just like slicing a list, but with a string

  ***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 List Elements

```
name = 'Neil'
```

- For loop using **`range`**:

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

Output:

```
N
e
i
l
```

- 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 value in list (in order)

# String-a-palooza (String Functions 1)

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

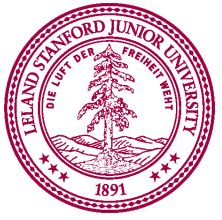
```
str1 = 'abc'
str2 = 'def'
str3 = 'abc' + 'def'
```

```
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:

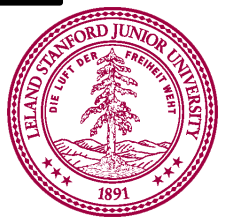
```
['this', 'is', 'a', 'test!']
```

- 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()
```

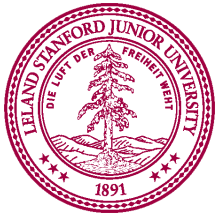
```
new_str:
'OH, HAPPY DAY!'
```

- 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')
```

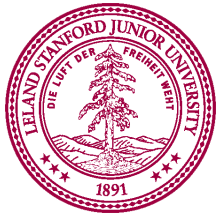
```
new_str:
'hamilton, scramilton, good time'
```

  - If the substring ***old*** is 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 = '보'
new_line  = '\n'    # special
tab       = '\t'    # special
backslash = '\\'    # special
backslash = '\''    # special
emoji     = '😀'
```

| | |
|---|---|
| `\n` | newline |
| `\t` | tab |
| `\\` | (one) backslash |
| `\'` | single quote |
| `\"` | double quote |

Advanced course

Characters are just a giant enumeration!

Just the number please

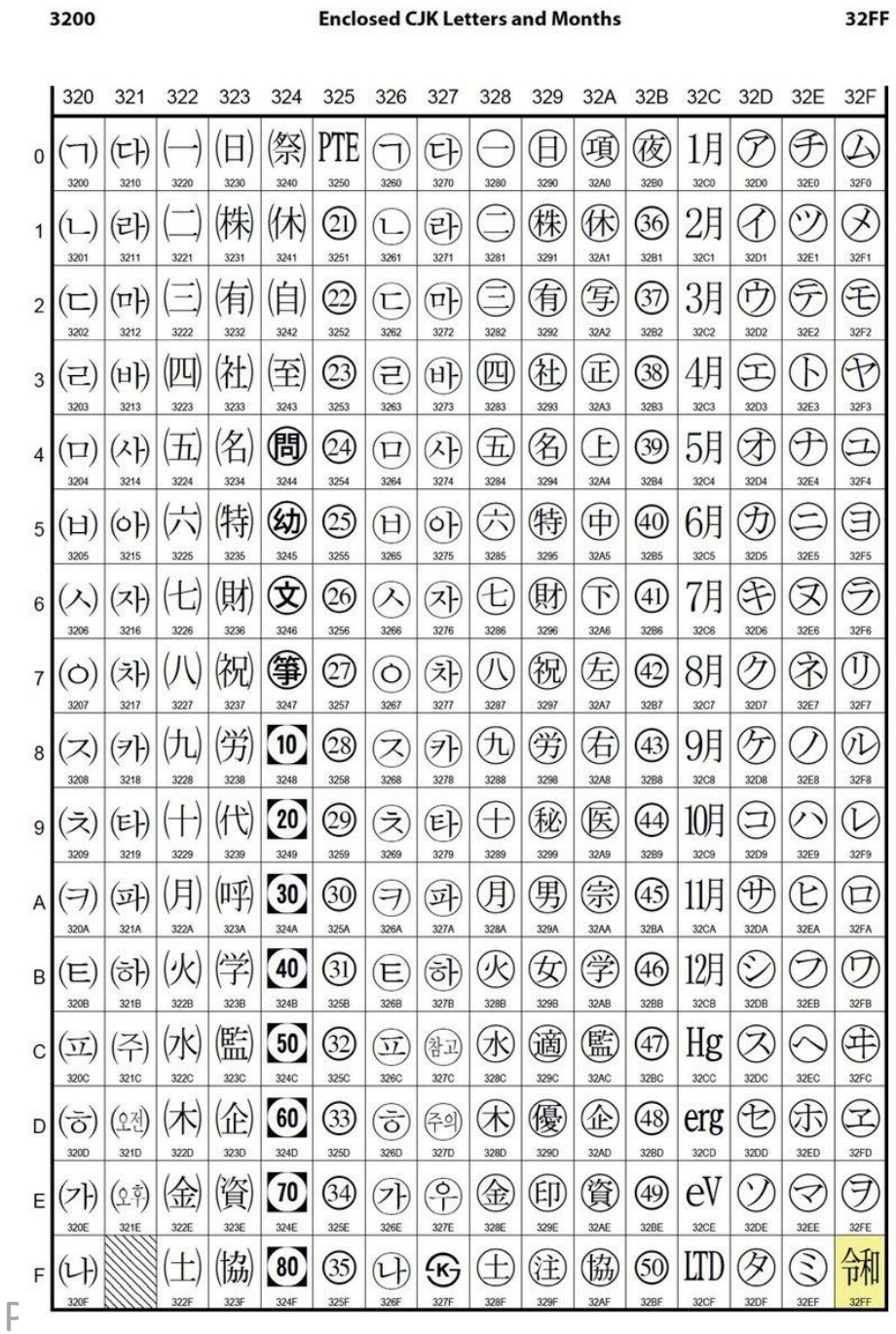
# ASCII

| 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 the first half of the table

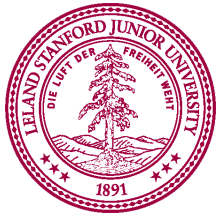
The letter A, for example, has the ASCII value 65

# 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 | ㈀ 3200 | ㈁ 3210 | ㈂ 3220 | ㈃ 3230 | ㈄ 3240 | PTE 3250 | ㉠ 3260 | ㉡ 3270 | ㉢ 3280 | ㉣ 3290 | ㉤ 32A0 | ㉥ 32B0 | 1月 32C0 | ㋀ 32D0 | ㋐ 32E0 | ㋠ 32F0 |
| 1 | ㈁ 3201 | ㈂ 3211 | ㈃ 3221 | ㈄ 3231 | ㈅ 3241 | ㉑ 3251 | ㉡ 3261 | ㉢ 3271 | ㉣ 3281 | ㉤ 3291 | ㉥ 32A1 | ㉯ 32B1 | 2月 32C1 | ㋁ 32D1 | ㋑ 32E1 | ㋡ 32F1 |
| 2 | ㈂ 3202 | ㈃ 3212 | ㈄ 3222 | ㈅ 3232 | ㈆ 3242 | ㉒ 3252 | ㉢ 3262 | ㉣ 3272 | ㉤ 3282 | ㉥ 3292 | ㉦ 32A2 | ㉰ 32B2 | 3月 32C2 | ㋂ 32D2 | ㋒ 32E2 | ㋢ 32F2 |
| 3 | ㈃ 3203 | ㈄ 3213 | ㈅ 3223 | ㈆ 3233 | ㈇ 3243 | ㉓ 3253 | ㉣ 3263 | ㉤ 3273 | ㉥ 3283 | ㉦ 3293 | ㉧ 32A3 | ㉱ 32B3 | 4月 32C3 | ㋃ 32D3 | ㋓ 32E3 | ㋣ 32F3 |
| 4 | ㈄ 3204 | ㈅ 3214 | ㈆ 3224 | ㈇ 3234 | ㈈ 3244 | ㉔ 3254 | ㉤ 3264 | ㉦ 3274 | ㉧ 3284 | ㉨ 3294 | ㉨ 32A4 | ㉲ 32B4 | 5月 32C4 | ㋄ 32D4 | ㋔ 32E4 | ㋤ 32F4 |
| 5 | ㈅ 3205 | ㈆ 3215 | ㈇ 3225 | ㈈ 3235 | ㈉ 3245 | ㉕ 3255 | ㉥ 3265 | ㉧ 3275 | ㉨ 3285 | ㉩ 3295 | ㉩ 32A5 | ㉳ 32B5 | 6月 32C5 | ㋅ 32D5 | ㋕ 32E5 | ㋥ 32F5 |
| 6 | ㈆ 3206 | ㈇ 3216 | ㈈ 3226 | ㈉ 3236 | ㈊ 3246 | ㉖ 3256 | ㉦ 3266 | ㉧ 3276 | ㉨ 3286 | ㉩ 3296 | ㉪ 32A6 | ㉴ 32B6 | 7月 32C6 | ㋆ 32D6 | ㋖ 32E6 | ㋦ 32F6 |
| 7 | ㈇ 3207 | ㈈ 3217 | ㈉ 3227 | ㈊ 3237 | ㈋ 3247 | ㉗ 3257 | ㉧ 3267 | ㉨ 3277 | ㉩ 3287 | ㉪ 3297 | ㉫ 32A7 | ㉵ 32B7 | 8月 32C7 | ㋇ 32D7 | ㋗ 32E7 | ㋧ 32F7 |
| 8 | ㈈ 3208 | ㈉ 3218 | ㈊ 3228 | ㈋ 3238 | ㈌ 3248 | ㉘ 3258 | ㉨ 3268 | ㉩ 3278 | ㉪ 3288 | ㉫ 3298 | ㉬ 32A8 | ㉶ 32B8 | 9月 32C8 | ㋈ 32D8 | ㋘ 32E8 | ㋨ 32F8 |
| 9 | ㈉ 3209 | ㈊ 3219 | ㈋ 3229 | ㈌ 3239 | ㈍ 3249 | ㉙ 3259 | ㉩ 3269 | ㉪ 3279 | ㉫ 3289 | ㉬ 3299 | ㉭ 32A9 | ㉷ 32B9 | 10月 32C9 | ㋉ 32D9 | ㋙ 32E9 | ㋩ 32F9 |
| A | ㈊ 320A | ㈋ 321A | ㈌ 322A | ㈍ 323A | ㈎ 324A | ㉚ 325A | ㉪ 326A | ㉫ 327A | ㉬ 328A | ㉭ 329A | ㉮ 32AA | ㉸ 32BA | 11月 32CA | ㋊ 32DA | ㋚ 32EA | ㋪ 32FA |
| B | ㈋ 320B | ㈌ 321B | ㈍ 322B | ㈎ 323B | ㈏ 324B | ㉛ 325B | ㉫ 326B | ㉬ 327B | ㉭ 328B | ㉮ 329B | ㉯ 32AB | ㉹ 32BB | 12月 32CB | ㋋ 32DB | ㋛ 32EB | ㋫ 32FB |
| C | ㈌ 320C | ㈍ 321C | ㈎ 322C | ㈏ 323C | ㈐ 324C | ㉜ 325C | ㉬ 326C | 참고 327C | ㉭ 328C | ㉮ 329C | ㉯ 32AC | ㉺ 32BC | Hg 32CC | ㋌ 32DC | ㋜ 32EC | ㋬ 32FC |
| D | ㈍ 320D | (오전) 321D | ㈎ 322D | ㈏ 323D | ㈐ 324D | ㉝ 325D | ㉭ 326D | 주의 327D | ㉮ 328D | ㉯ 329D | ㉰ 32AD | ㉻ 32BD | erg 32CD | ㋍ 32DD | ㋝ 32ED | ㋭ 32FD |
| E | ㈎ 320E | (오후) 321E | ㈏ 322E | ㈐ 323E | ㈑ 324E | ㉞ 325E | ㉮ 326E | ㉯ 327E | ㉰ 328E | ㉱ 329E | ㉲ 32AE | ㉼ 32BE | eV 32CE | ㋎ 32DE | ㋞ 32EE | ㋮ 32FE |
| F | ㈏ 320F | | ㈐ 322F | ㈑ 323F | ㈒ 324F | ㉟ 325F | ㉰ 326F | ㉱ 327F | ㉲ 328F | ㉳ 329F | ㉴ 32AF | ㉽ 32BF | LTD 32CF | ㋏ 32DF | ㋟ 32EF | 令和 32FF |



'A' -> 'Z' are sequential.
'a' -> 'z' are sequential.
'0' -> '9' are sequential.

```
ord(ch)
```

# 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
```

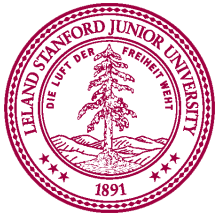
  - Strings are compared in lexicographic (~alphabetic) order:

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

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

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



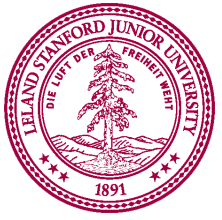


Strings have some
interesting properties

# 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.
  - Reassigning the String variable (that's allowed).
- ***Important consequence:*** if you pass a String into a function, you are guaranteed your 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)

```
x = 'abc'
```

```
x[1] = 'z'
```

**Error!**

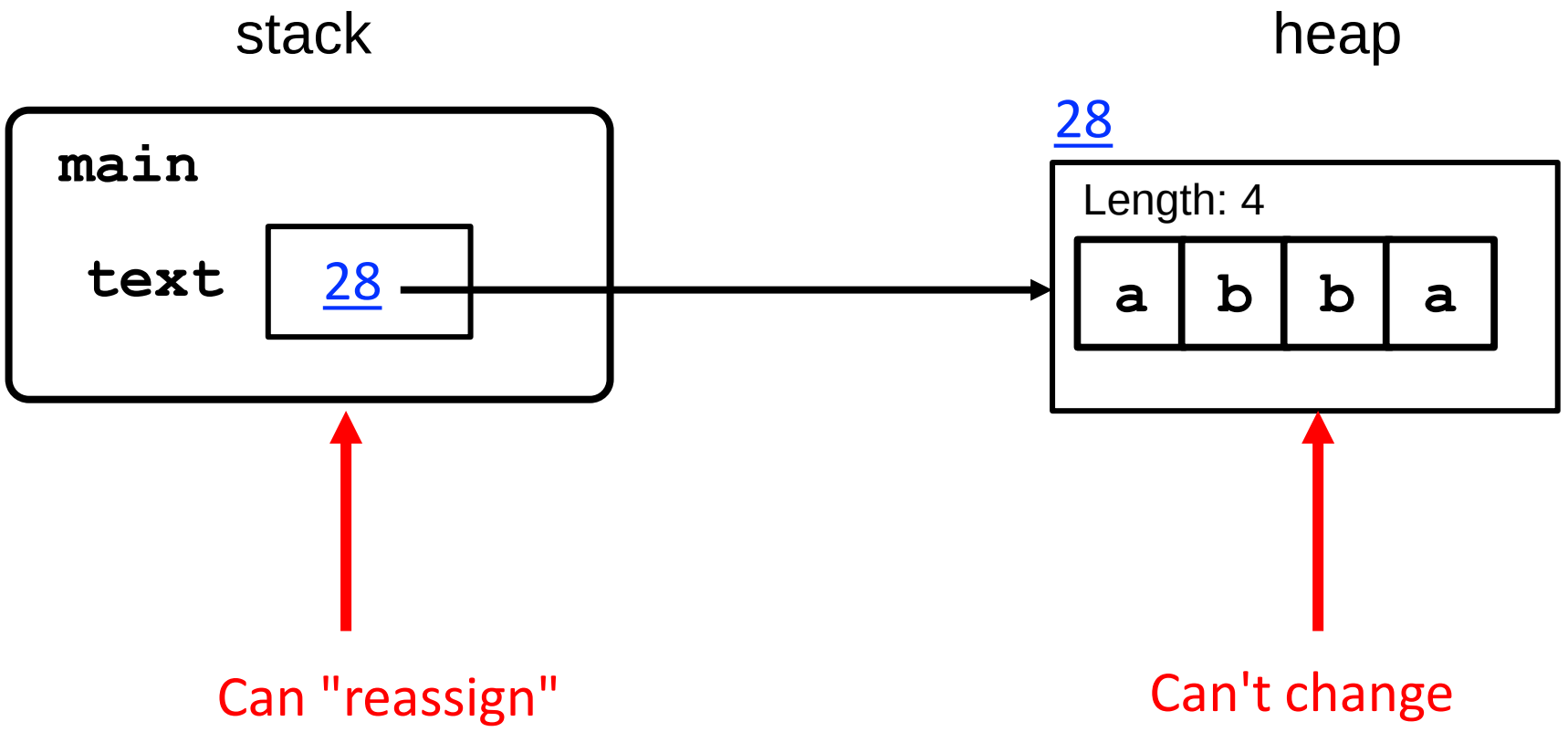
```
Traceback (most recent call last):
...
TypeError: 'str' object does not support item assignment
```

```
x = 'azc'
```

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

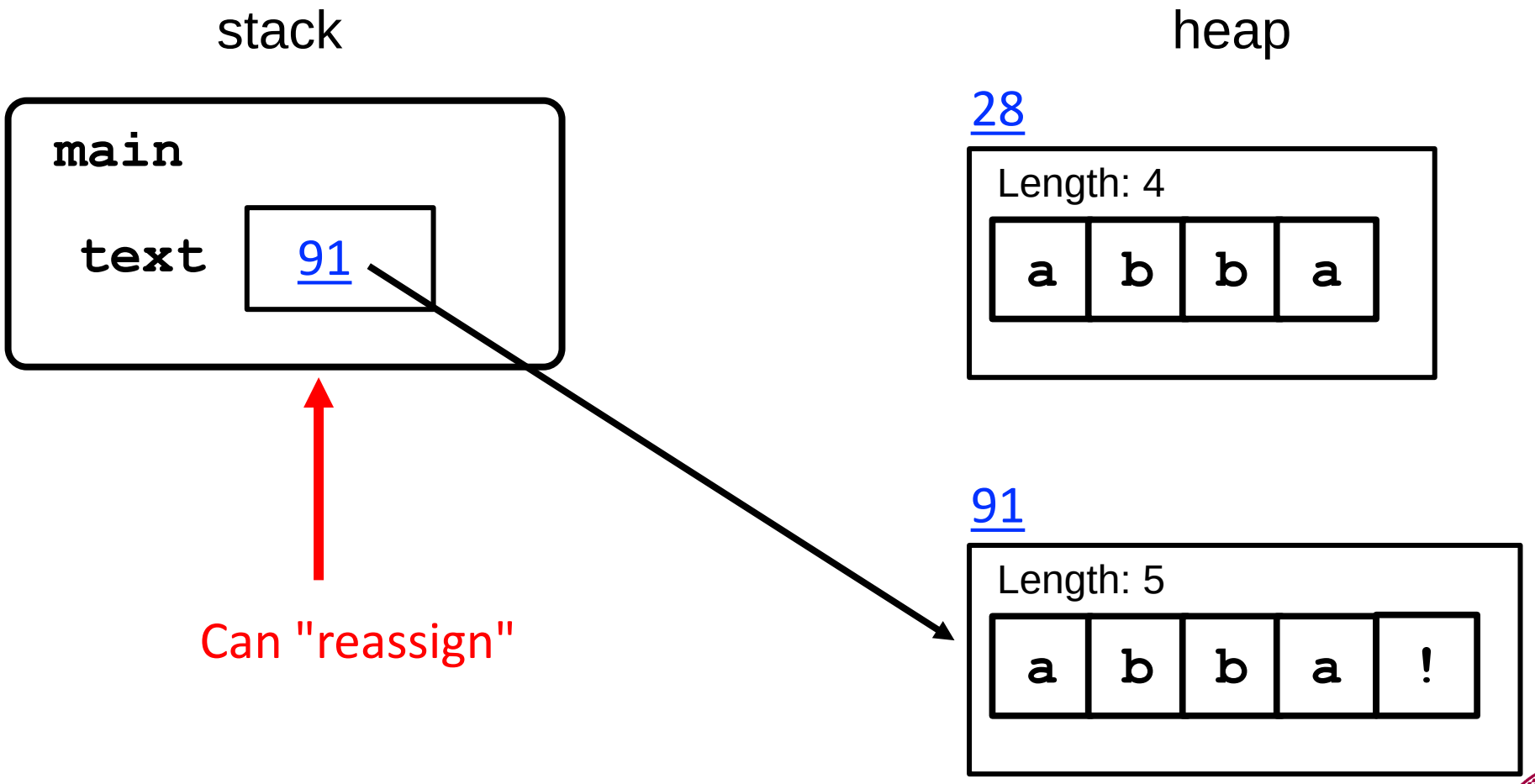
# Strings are Immutable (Take 2)

```
original = 'abba'
```



# Strings are Immutable (Take 2)

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

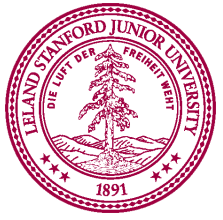


# Strings are often made through concatenation

```
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 = 2020
action = 'vote in the US election'
important = f'It is {year}, you gotta {action}!'
print(important)
```

```
It is 2020, you gotta vote in the US election!
```

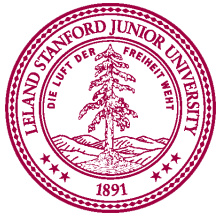
**General form:**

**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

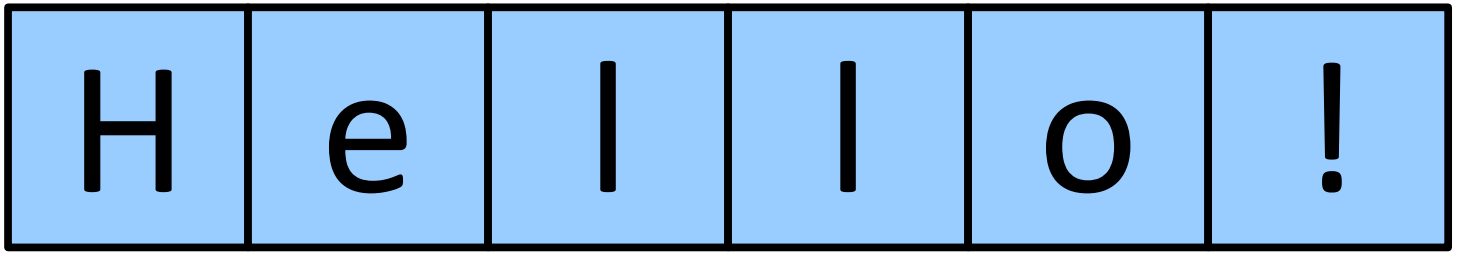
```
>>> weight = 160.123456
>>> print(f'Weight to two digits: {weight:.2f}')
Weight to two digits: 160.12
```





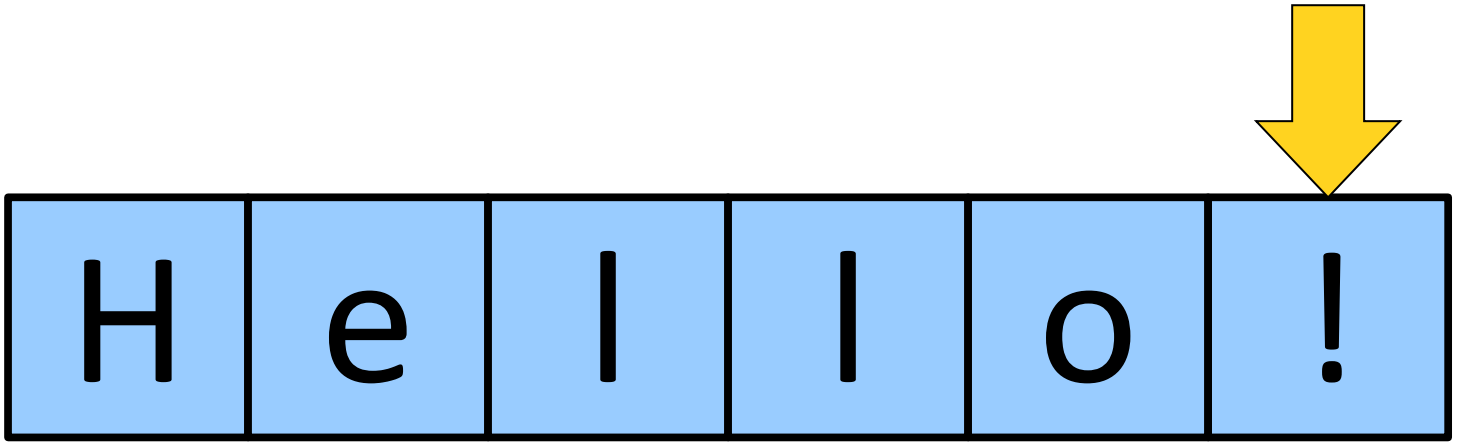
# Many string algorithms use the "loop and construct" pattern.

# Reversing a String

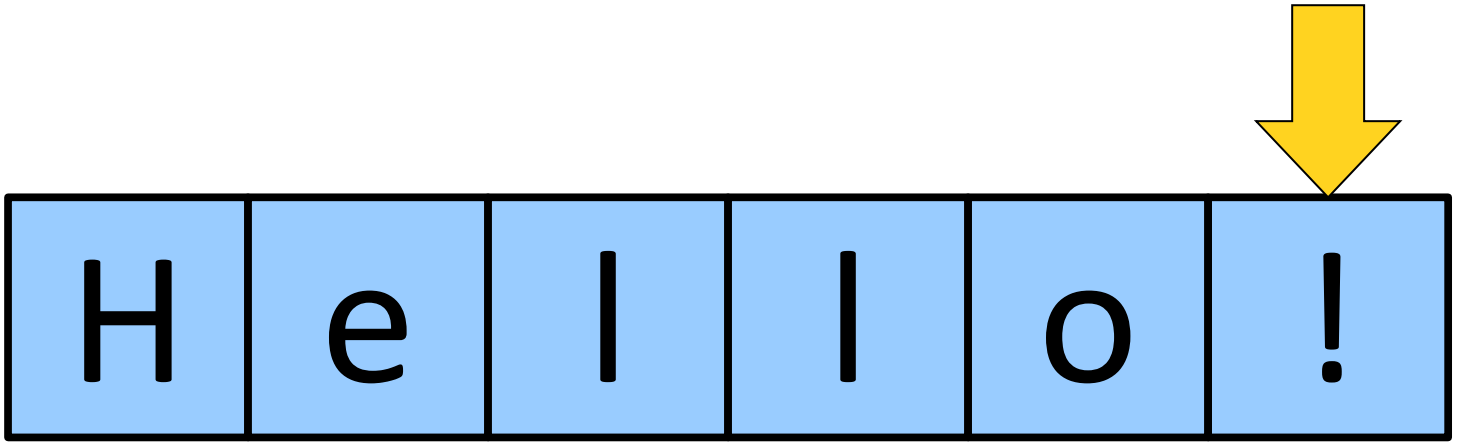



```
result = ''
# counts from len(str)-1 down to 0 (including 0)
for i in range(len(str)-1, -1, -1):
    ch = str[i]
    result += ch
return result
```

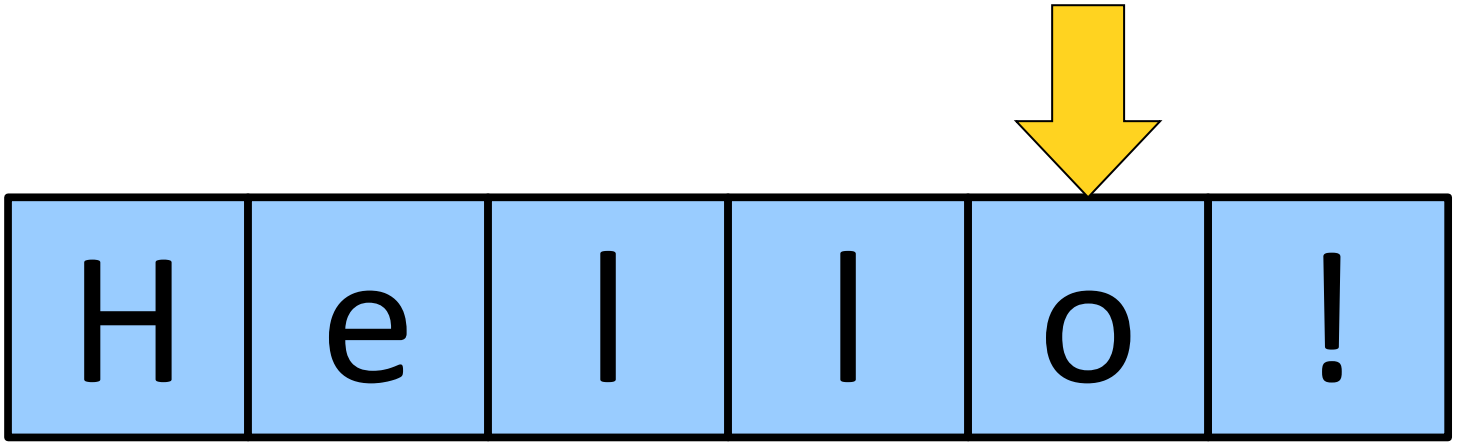
# Reversing a String



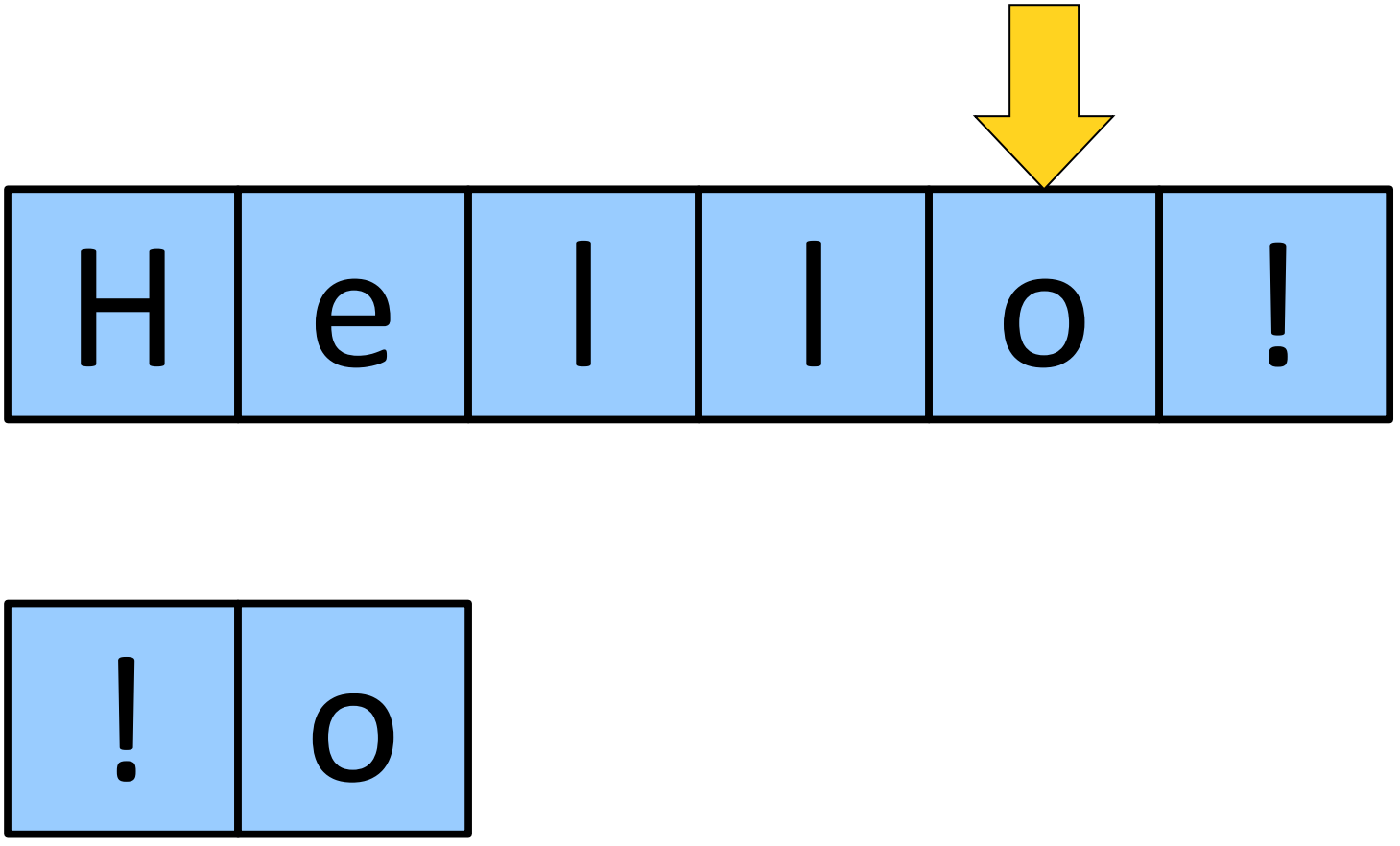
# Reversing a String



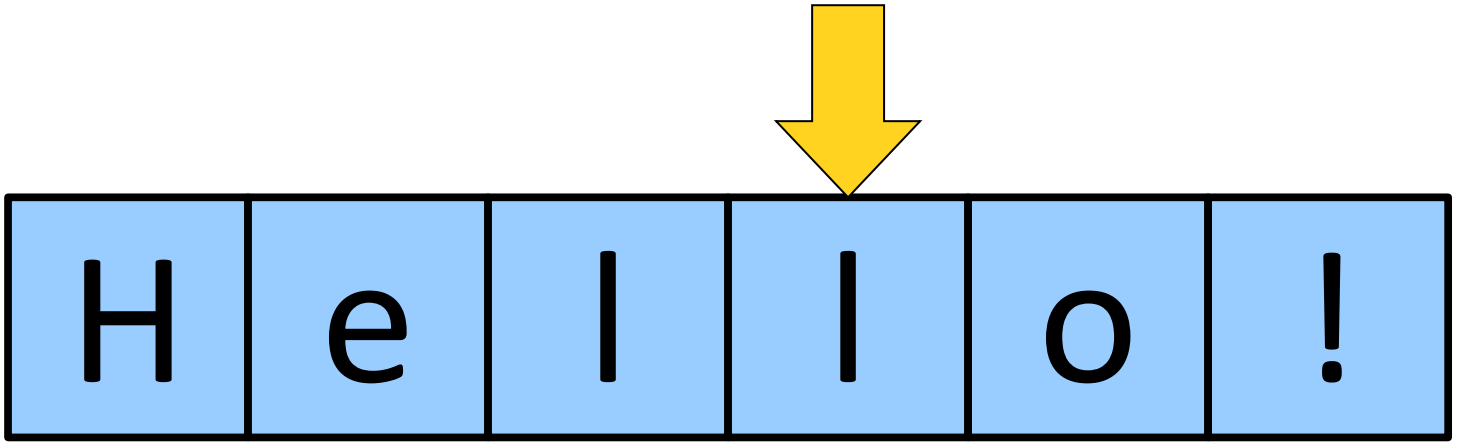

# Reversing a String

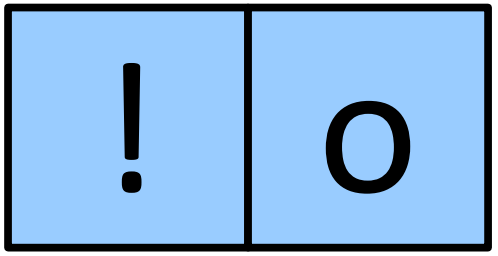



# Reversing a String

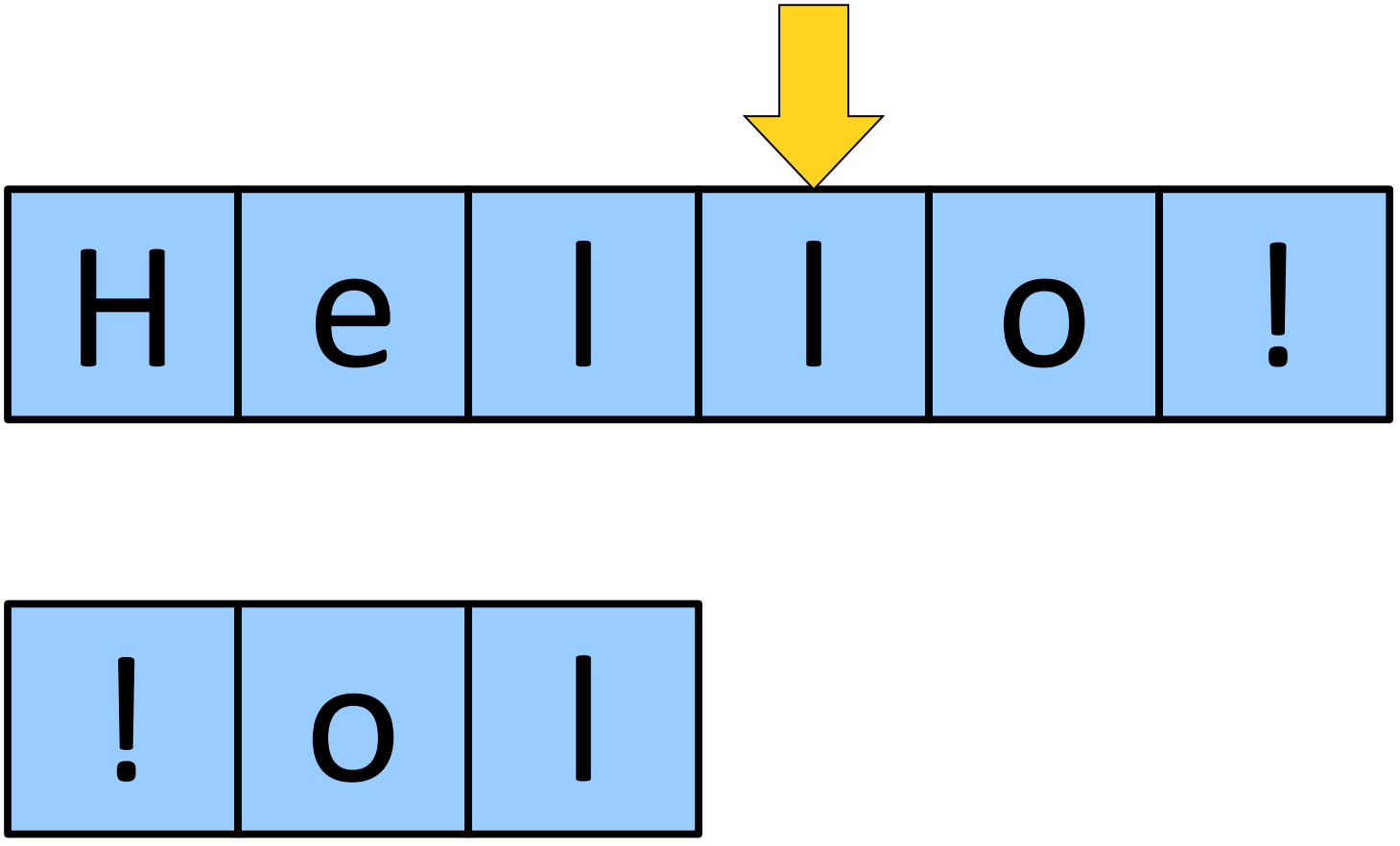


# Reversing a String

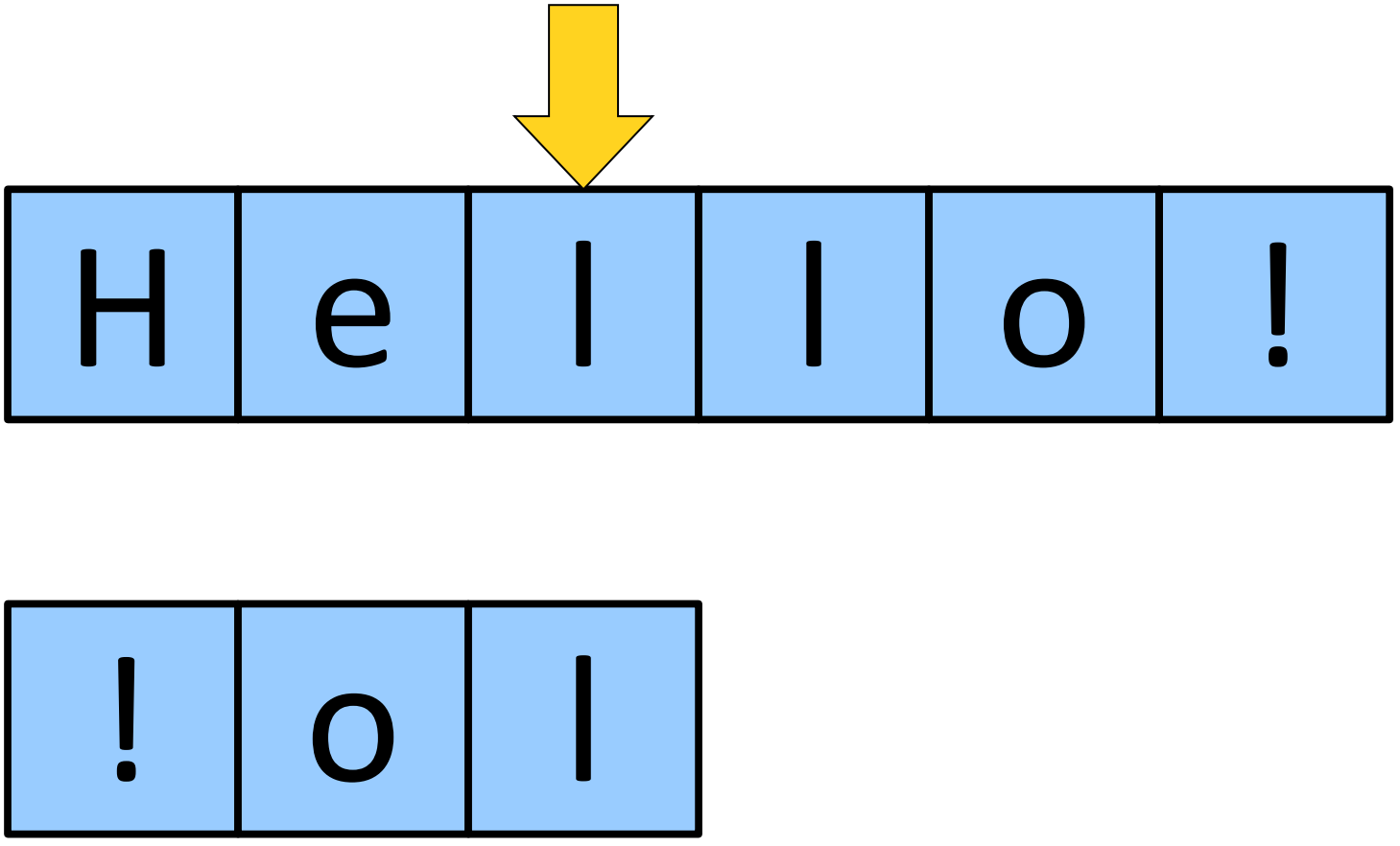


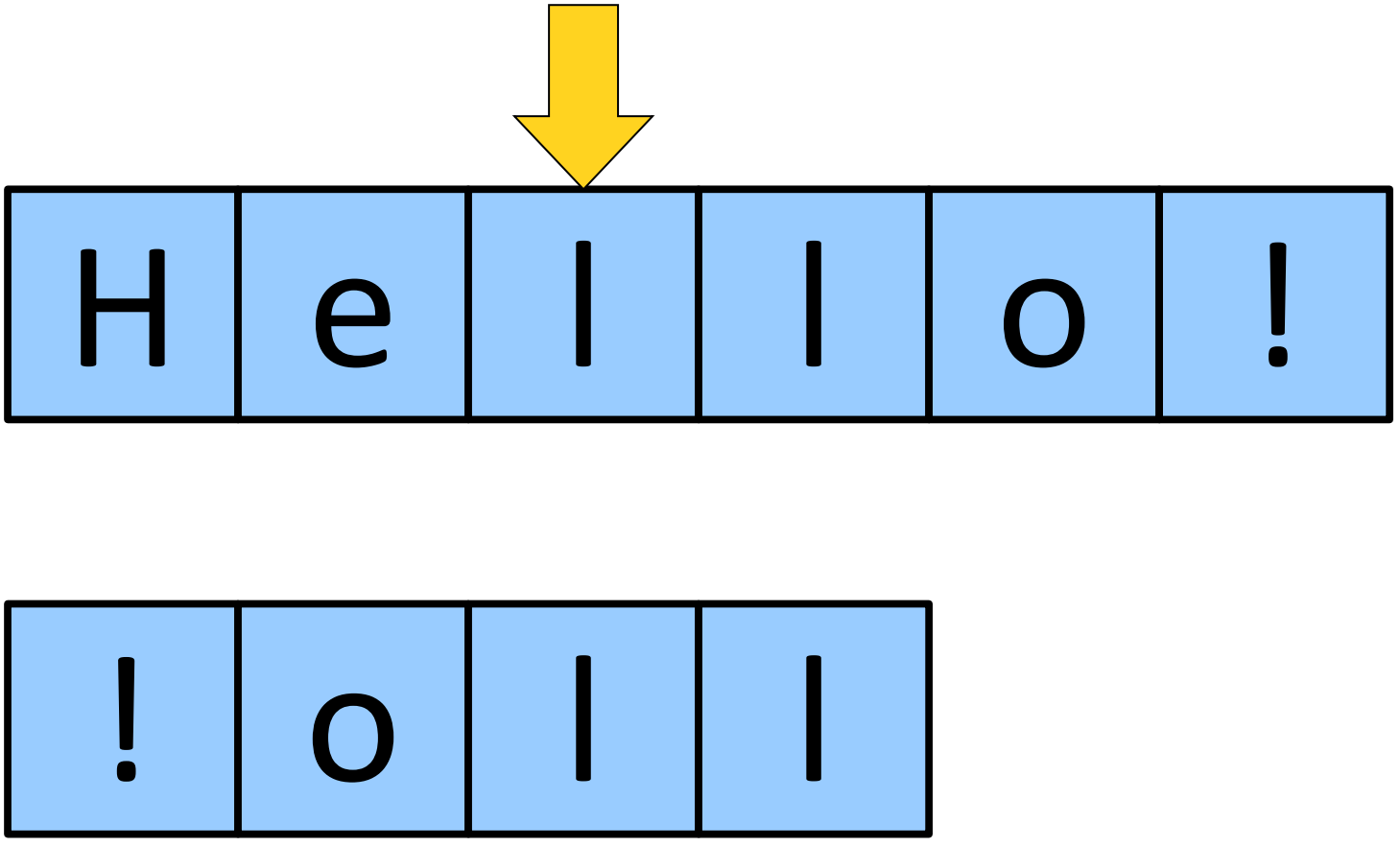
# Reversing a String



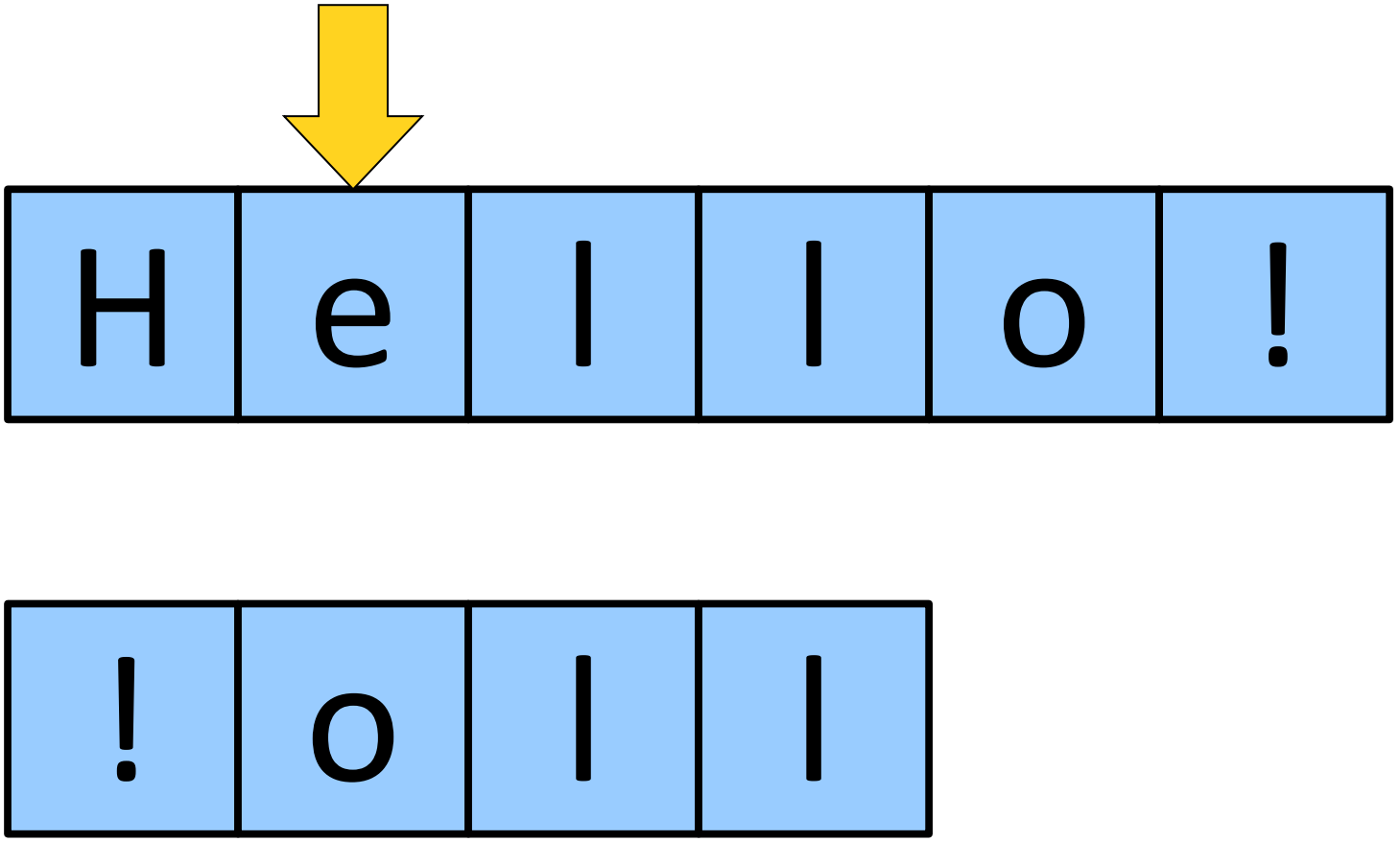
# Reversing a String



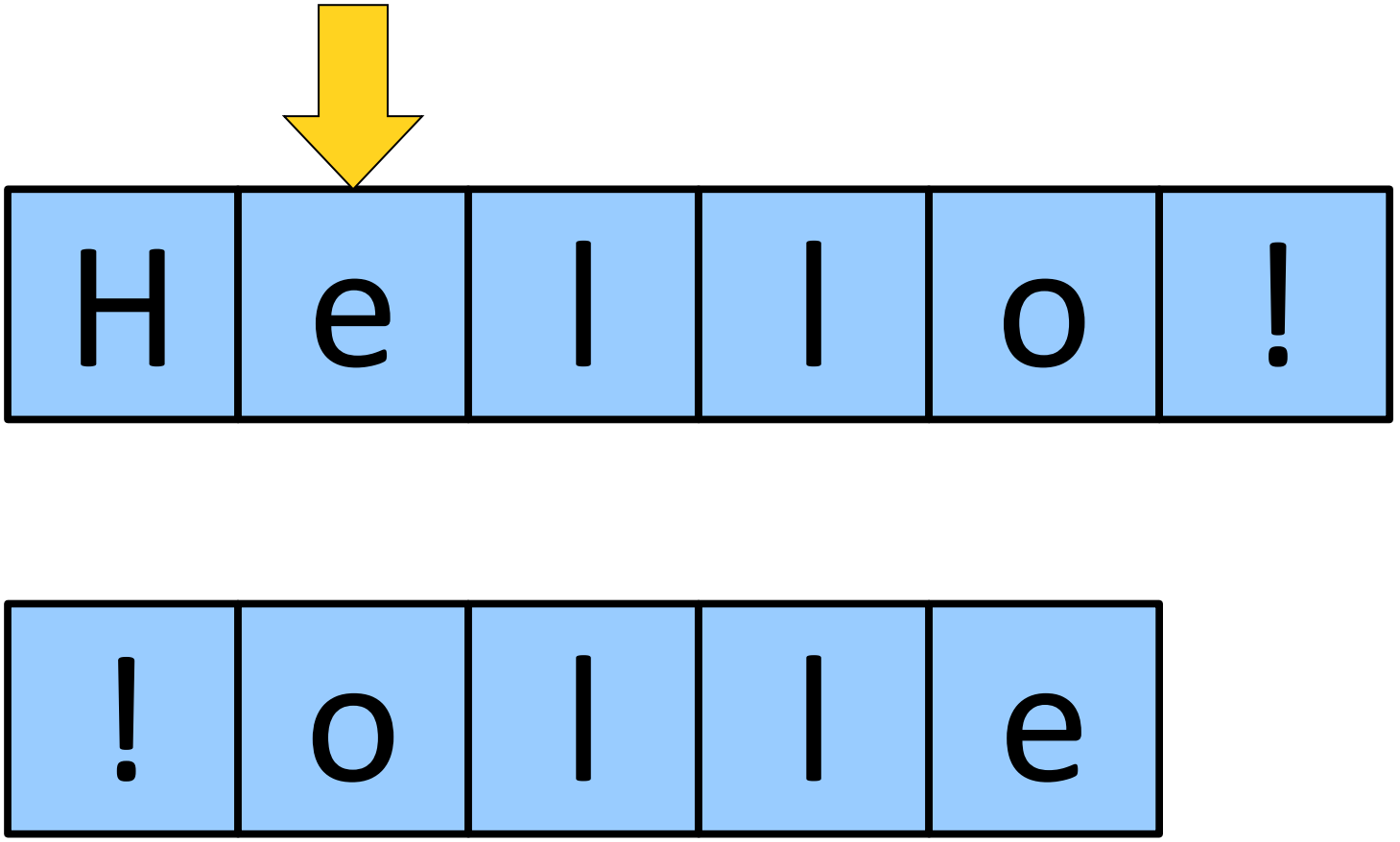
# Reversing a String



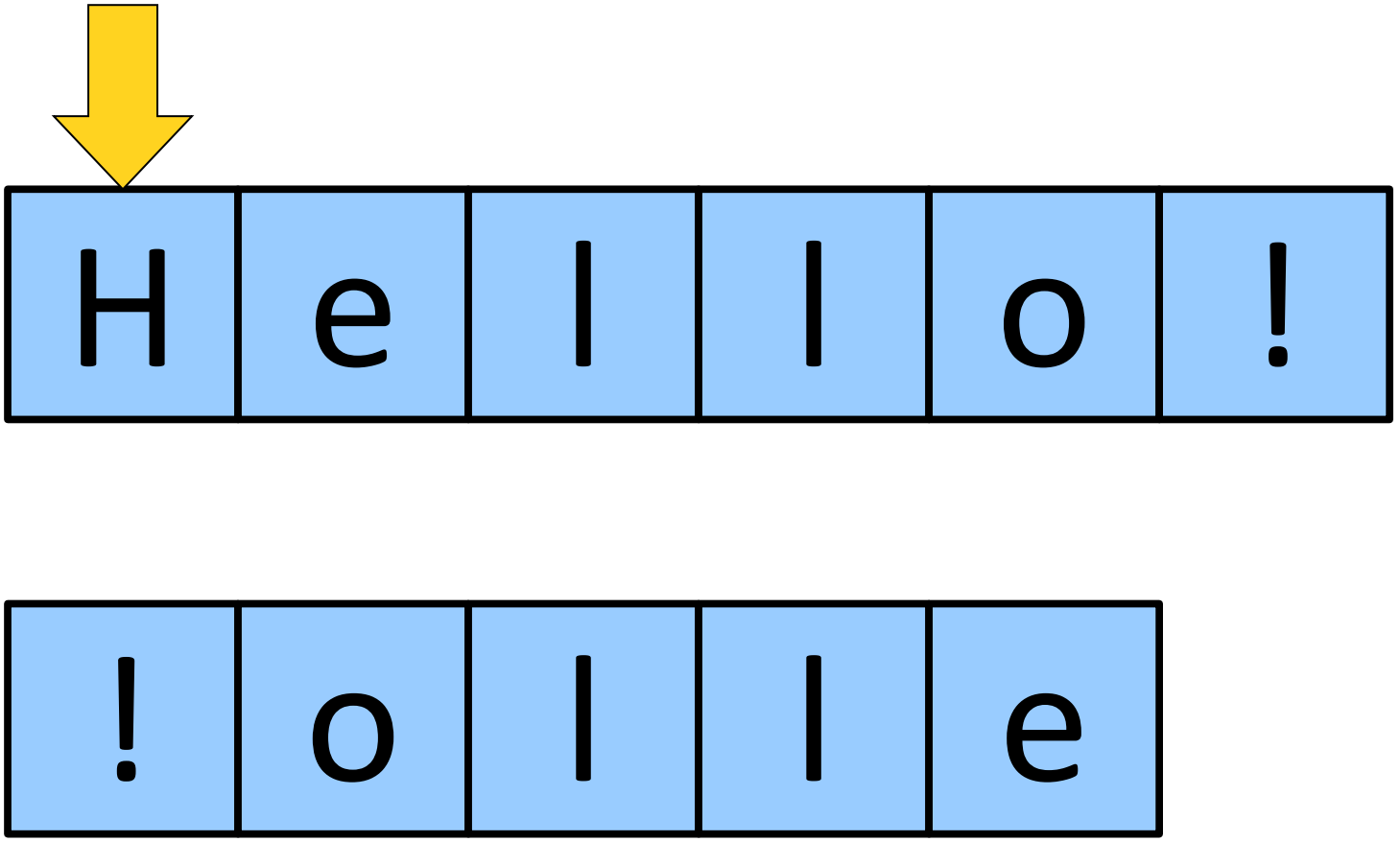
# Reversing a String



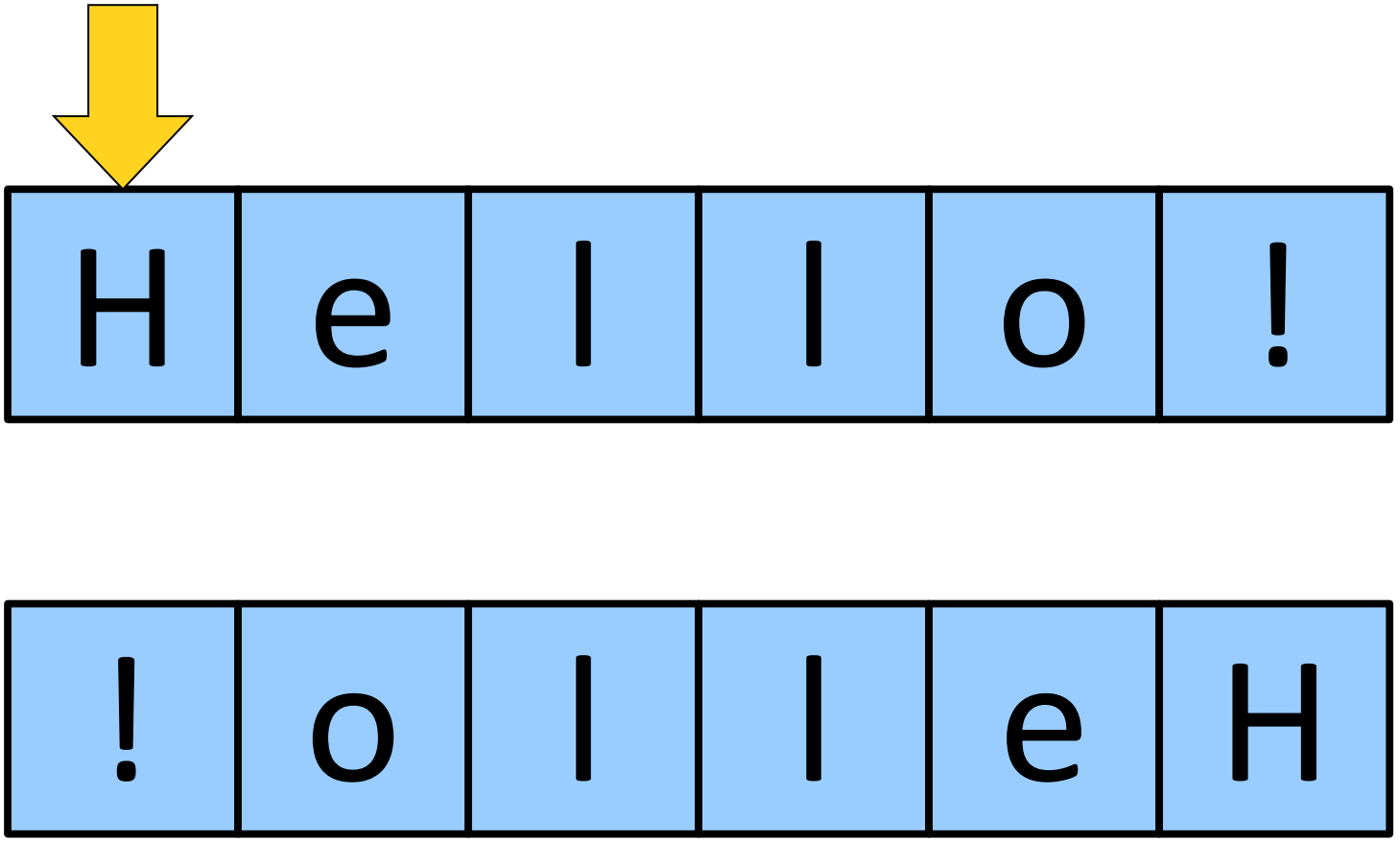
# Reversing a String



# Reversing a String

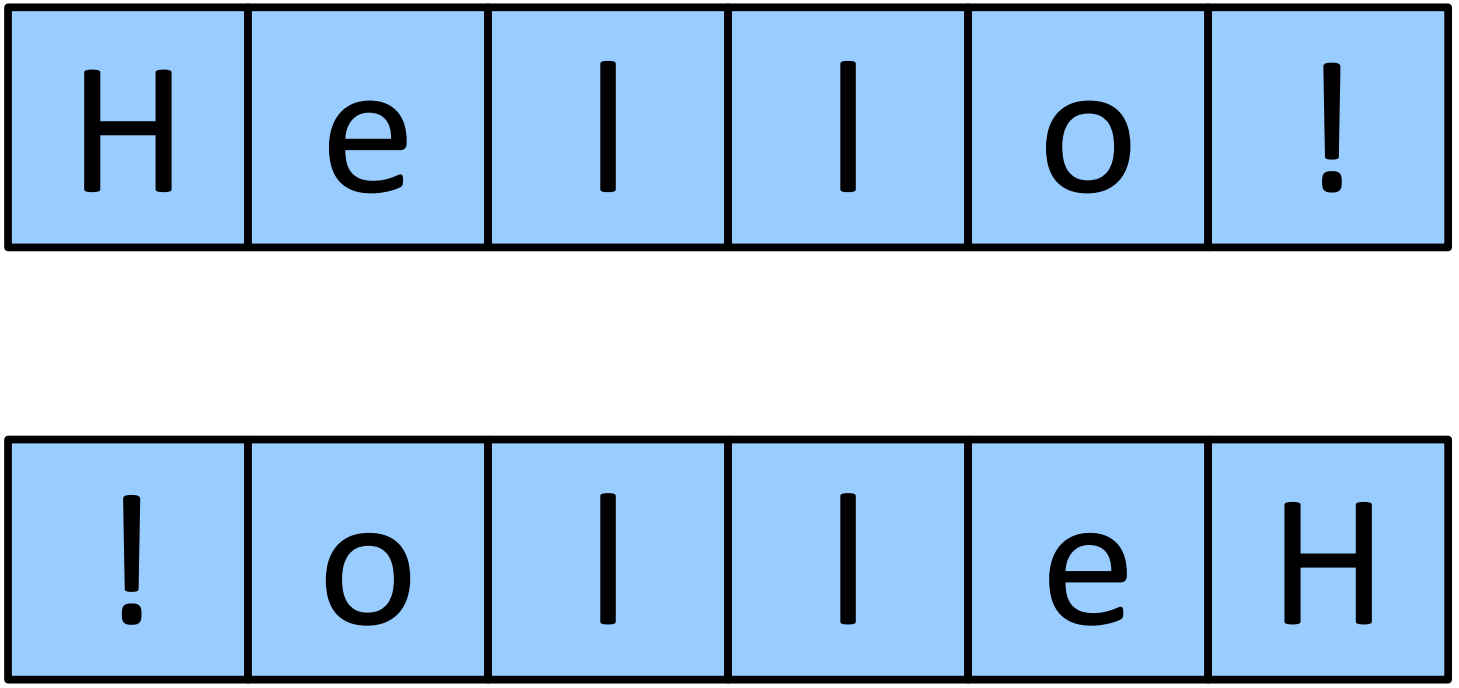


# Reversing a String

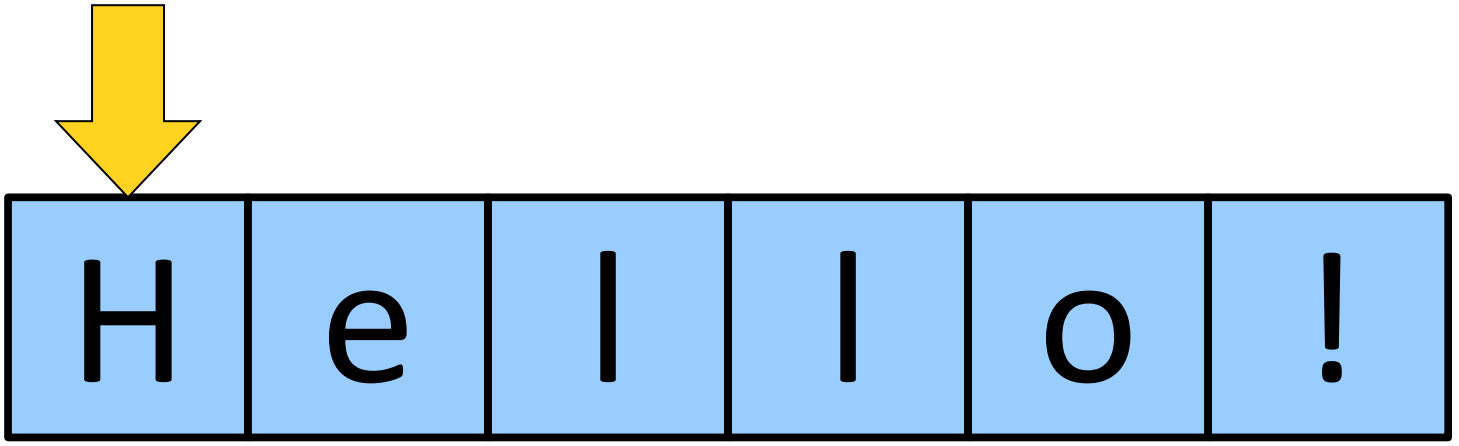


# Reversing a String

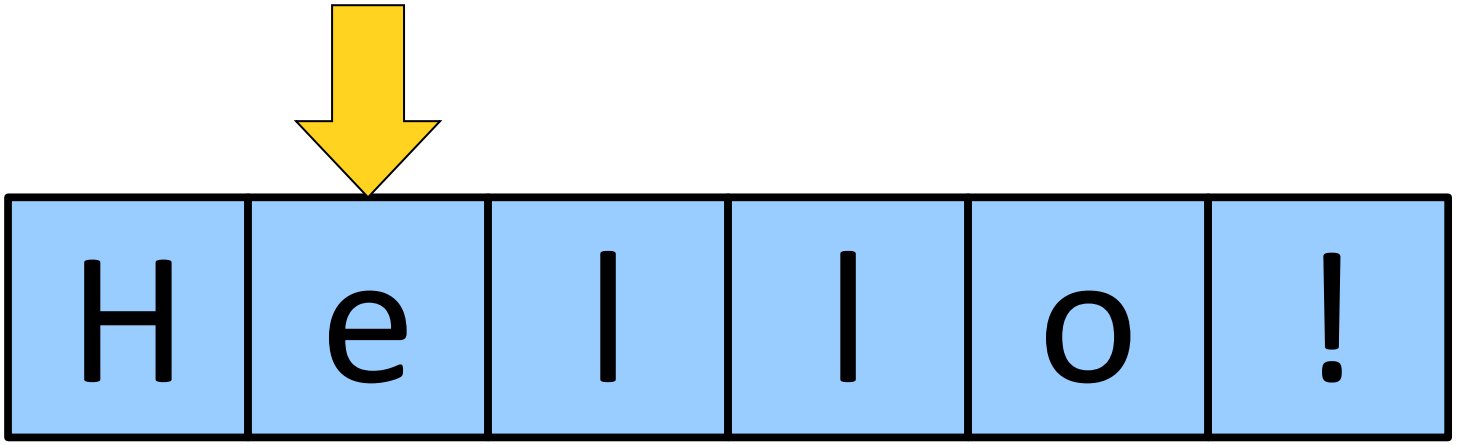
```
result = ''
# counts from len(str)-1 down to 0 (including 0)
for i in range(len(str)-1, -1, -1):
    ch = str[i]
    result += ch
return result
```

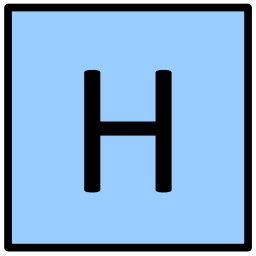


# Reversing a String

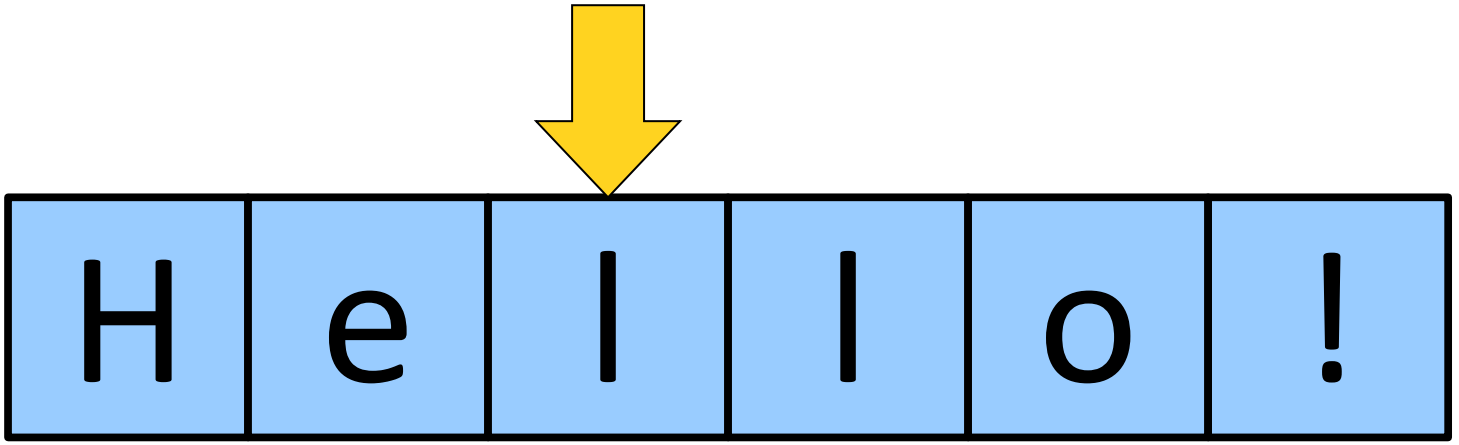


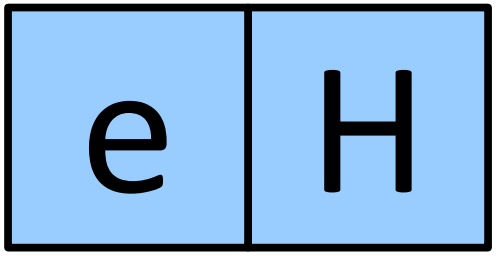
# Reversing a String



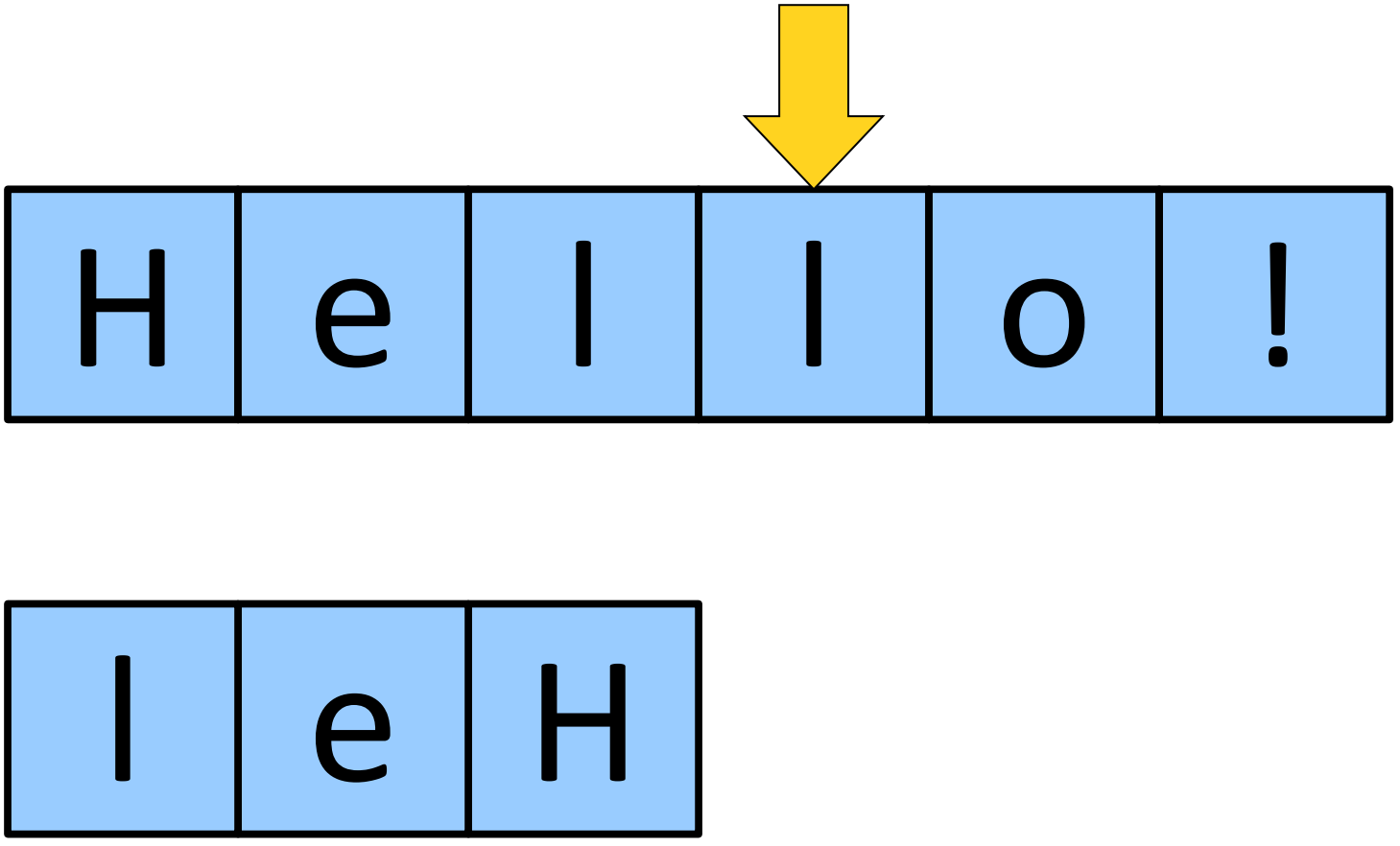



# Reversing a String

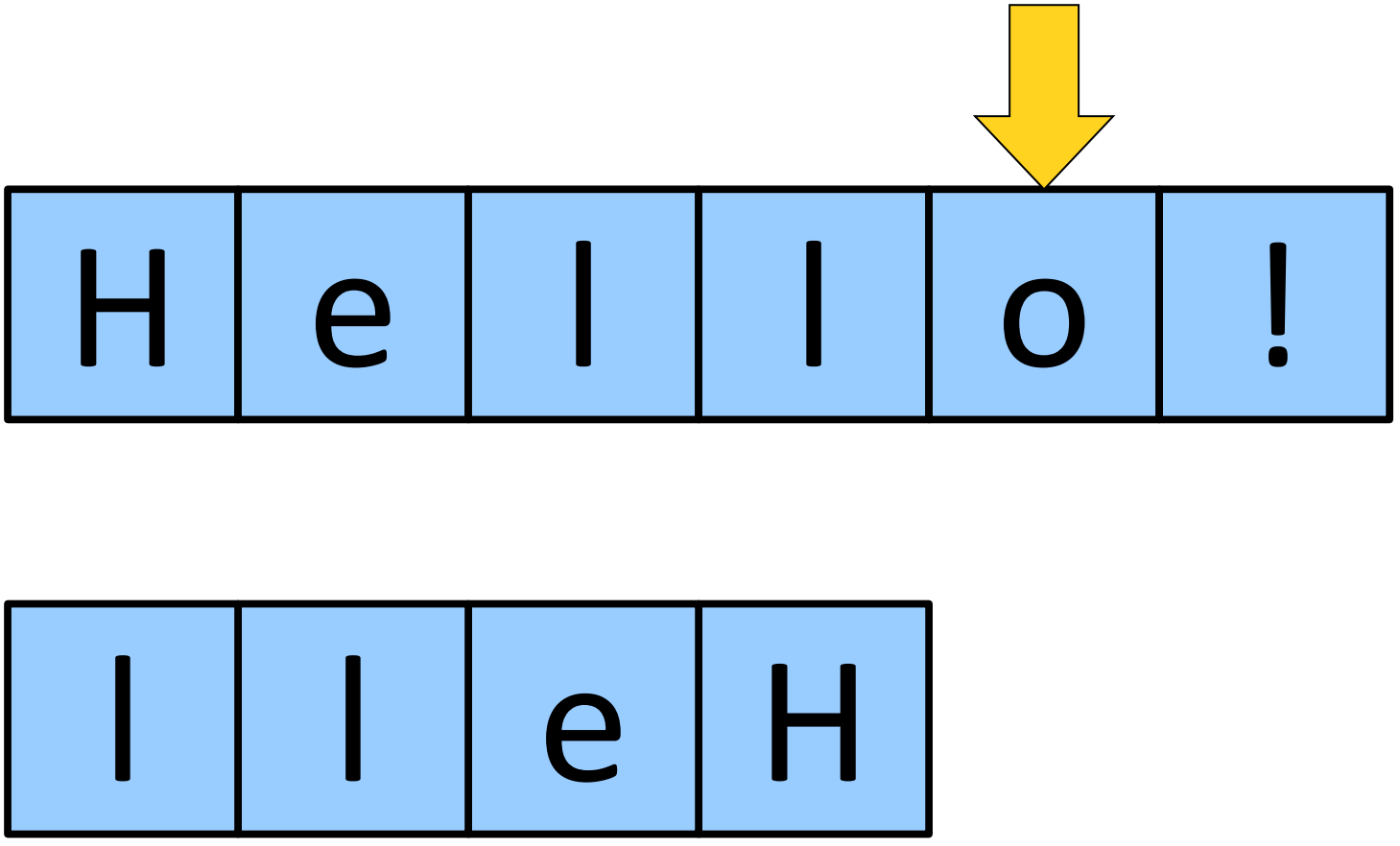


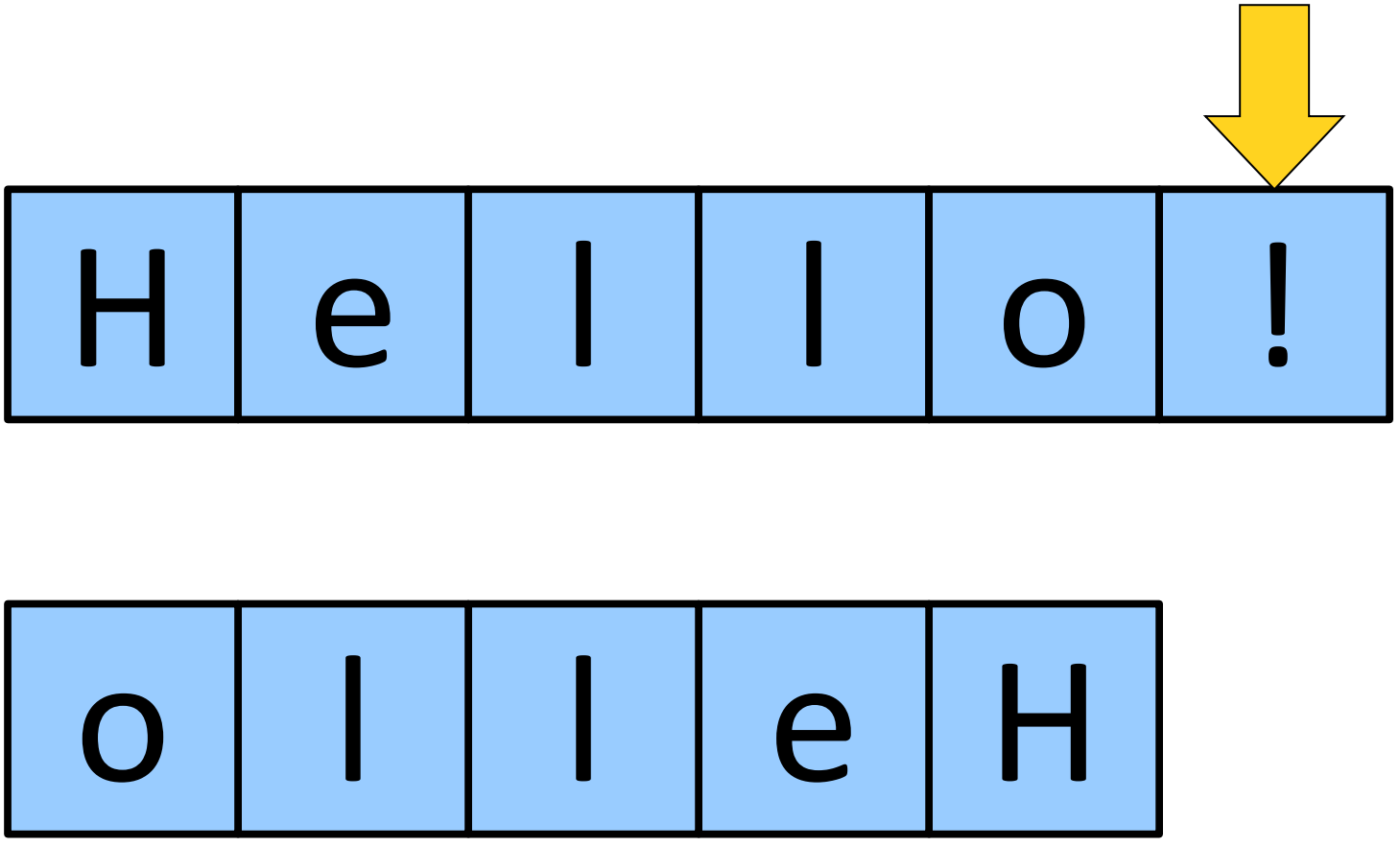
# Reversing a String



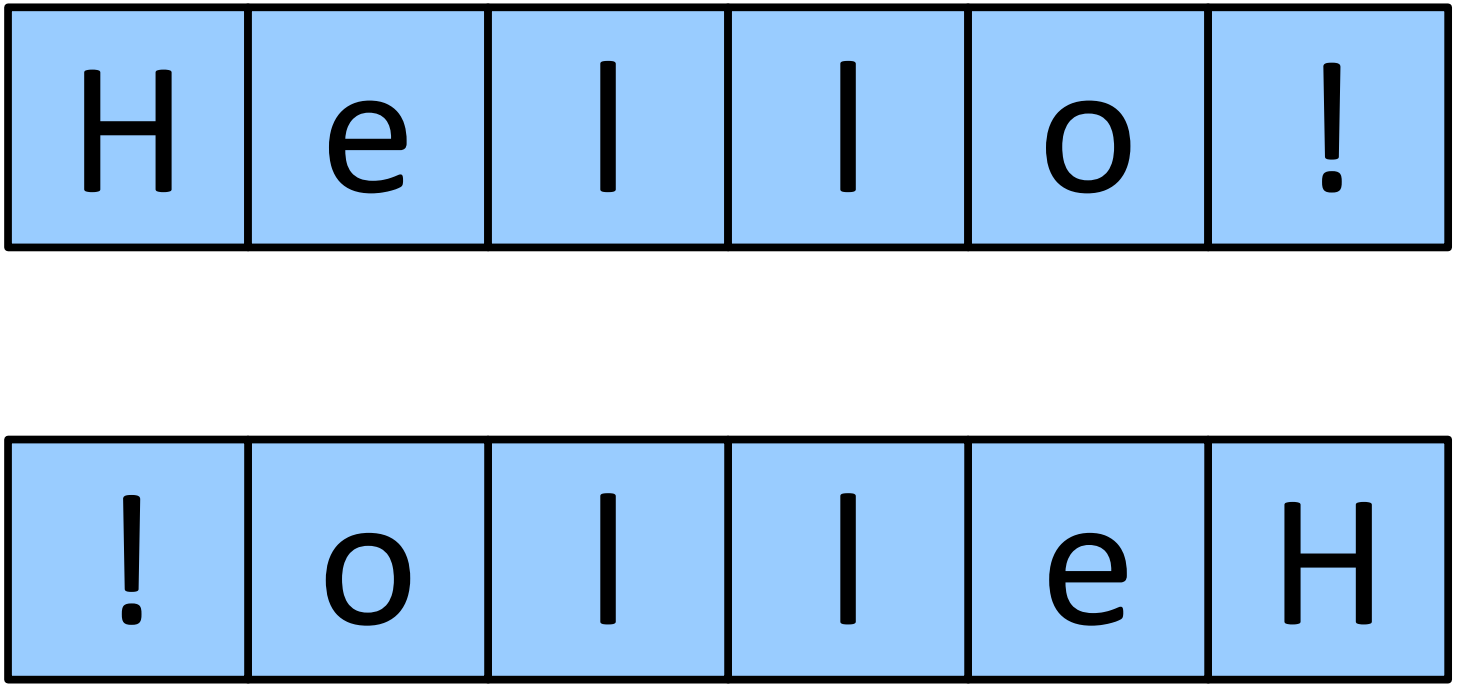
# Reversing a String



# Reversing a String



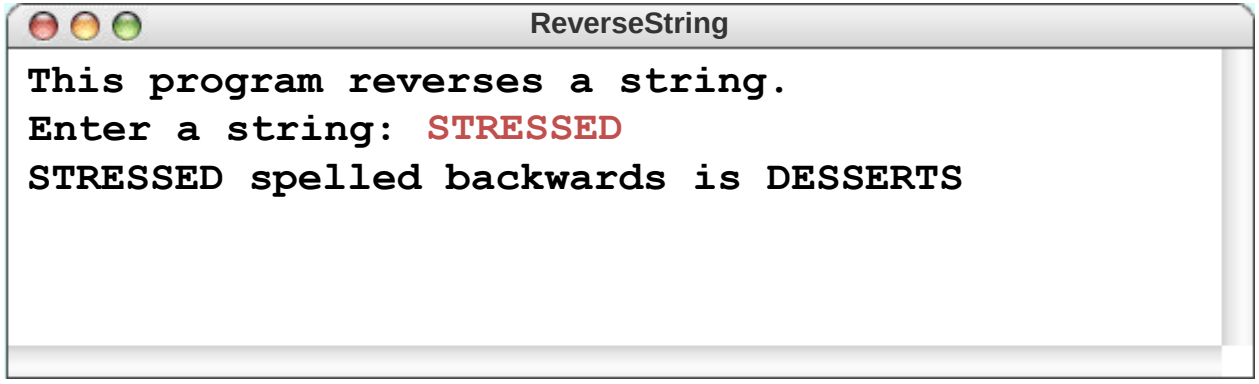
# Reversing a String



# reverse_string

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

        return result
```

| result | str | i |
|---|---|---|
| DESSERTS | STRESSED | 8 |



# reverse_string redux

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

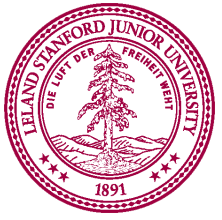
Swiss army knife pattern

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

Medium bear

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

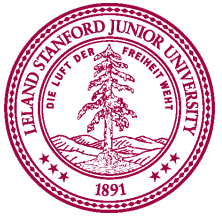
I'll have another slice!!



# Palindrome

- A ***palindrome*** is a string that reads the same forwards and backwards.
- For example:
  - Abba
  - Racecar
  - Kayak
  - Mr. Owl ate my metal worm.
  - Go hang a salami!  I'm a lasagna hog.
  - Elu par cette crapule

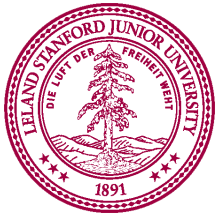




# Some test cases

- Let's test our program on some examples:
  - Racecar
  - Kayak
  - Mr. Owl ate my metal worm.
  - A man, a plan, a canal, Panama!
  - Go hang a salami!  I'm a lasagna hog.
- Will it work?





# More Palindromes

Here are some palindromes in other languages:

- بلح تعلق تحت قلعة حلب (Dates hang underneath a castle in Halab)
- 여보, 안경 안보여 (Honey, I can't see my glasses)
- कड़क (a loud thunderous sound)
- 上海自來水來自海上 (Shanghai tap water originates from "above" the ocean)

The comedian Dmitri Martin also has a routine about palindromes check it out at https://www.youtube.com/watch?v=0hUHDIOazIU

Let's take it out for a spin!

palindrome.py

# Learning Goals

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