Computer Science Fundamentals

Seminar 5: Inside your computer| Python: String

Lecturer: Olga Yugay

Agenda

Part A

- Lecture based Q&A
- Inside your computer
- Chipset architecture
- How CPU works

Part B: Python

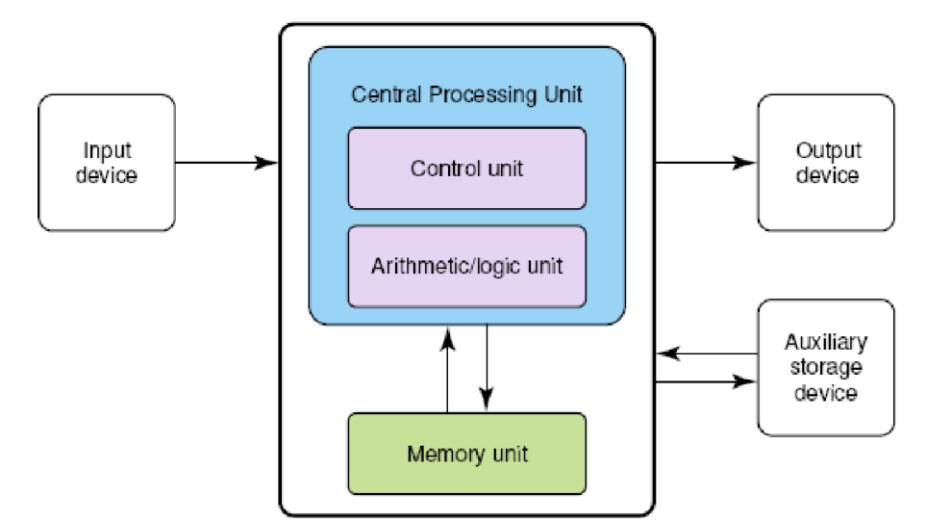
- String with for loop and break
- Value returning functions a.k.a. fruitful functions

Part A:

Lecture recap

What is main characteristic of von Neumann architecture?

Explain it on the diagram



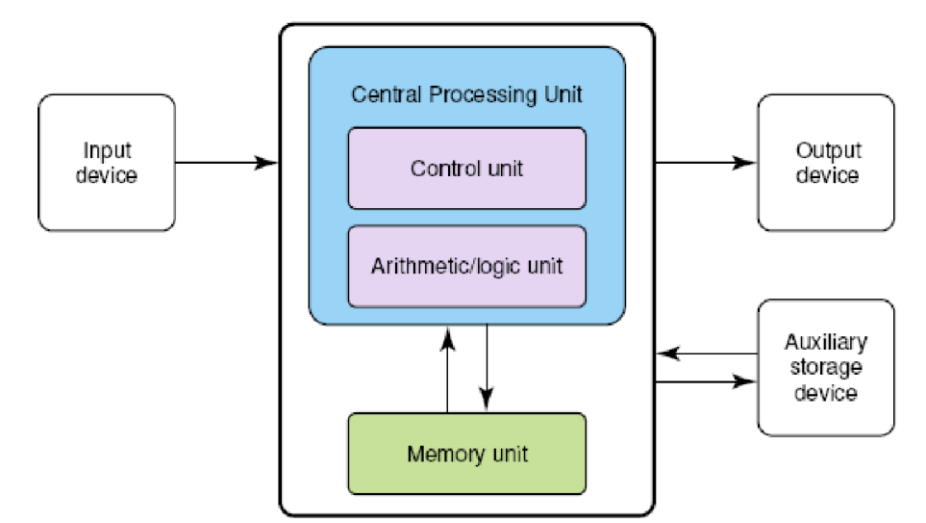
Lecture recap

1. What does it mean to say that a processor is 1.4 GHz?
2. What does it mean to say that memory is 133MHz?
3. Compare and contrast RAM and ROM memory.
4. What is a secondary storage device, and why are such devices important?

Inside your computer

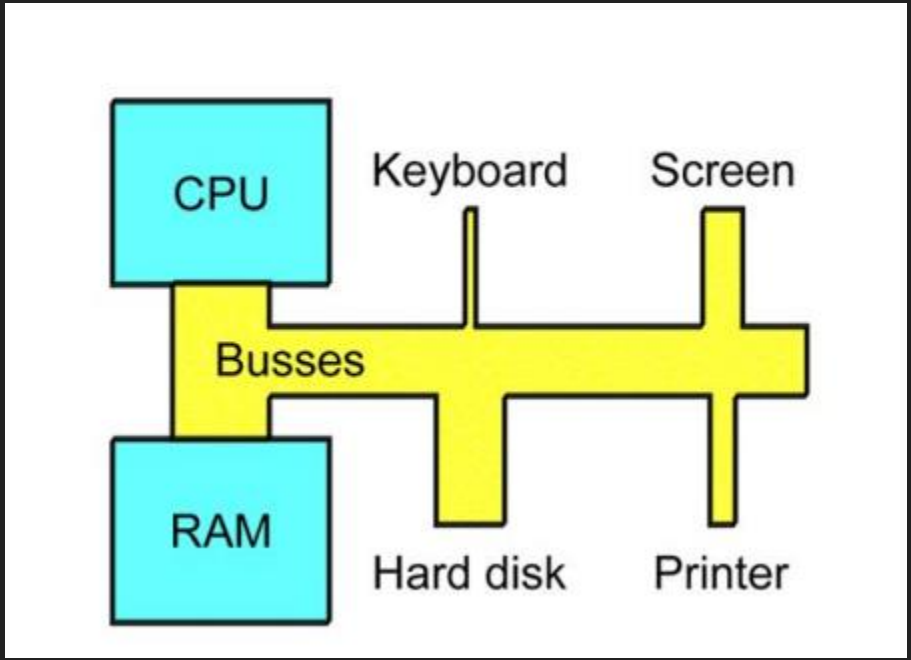
Watch Computer Parts

Map the computer parts from the video to the Neumann architecture

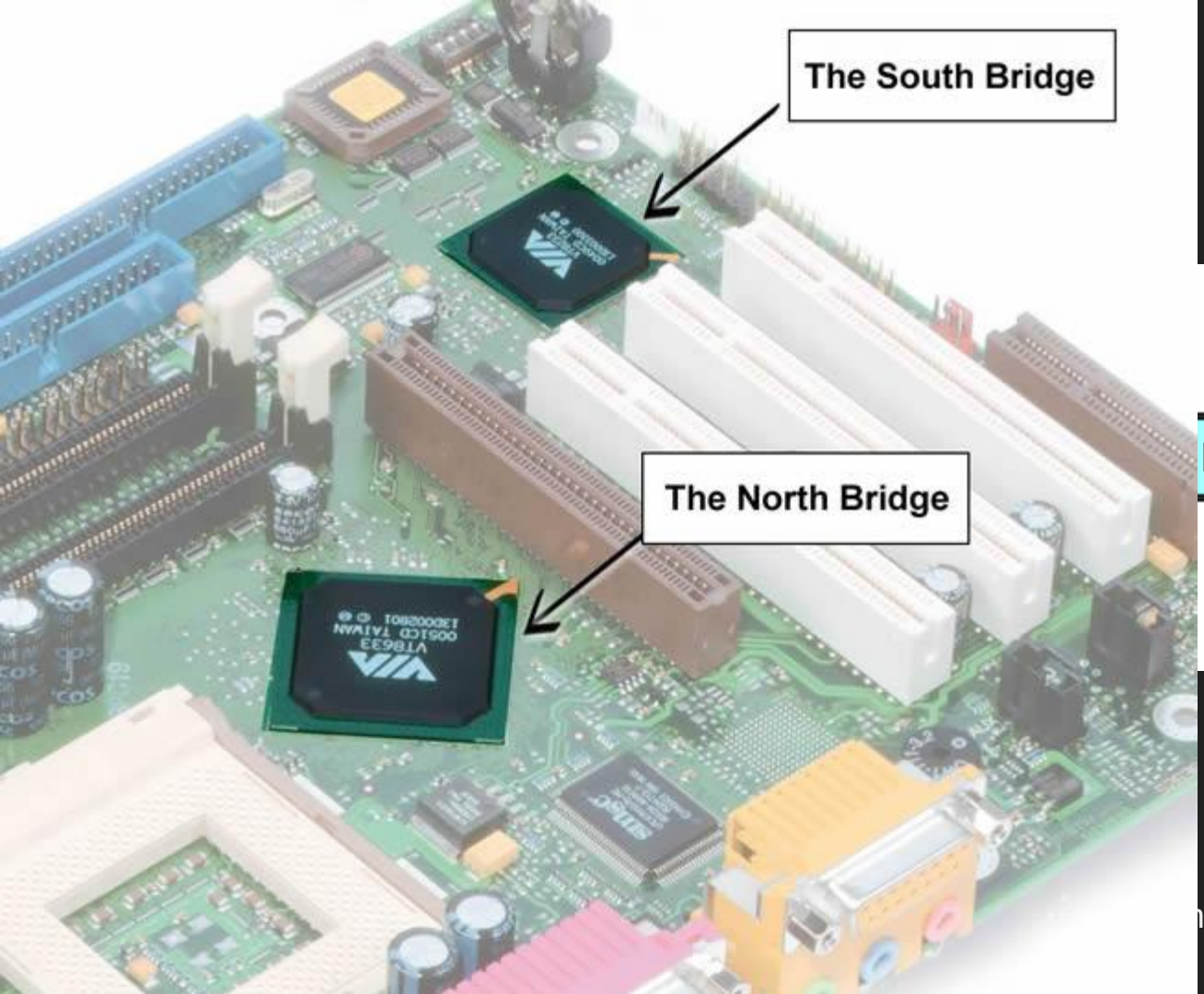


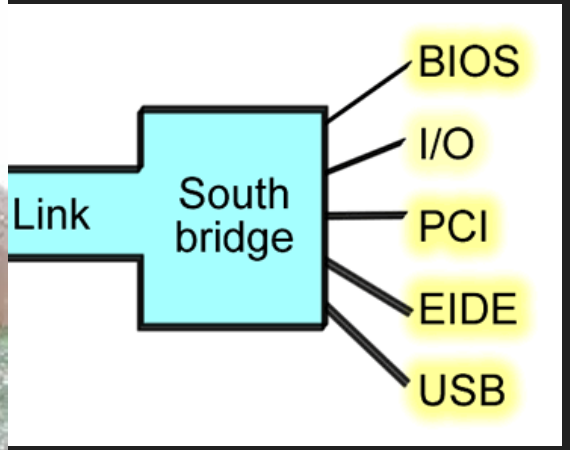
Chipset architecture until 2008 [1]

- Buses = data channels
- Some handle small transfers
  - E.g. keyboard (bytes per second)
- Some handle large transfers
  - E.g. RAM (GB per second)



[1] book.huihoo.com. (n.d.). PC Architecture. A book by Michael B. Karbo. [online] Available at: https://book.huihoo.com/pc-architecture/ [Accessed 8 Jun. 2022].

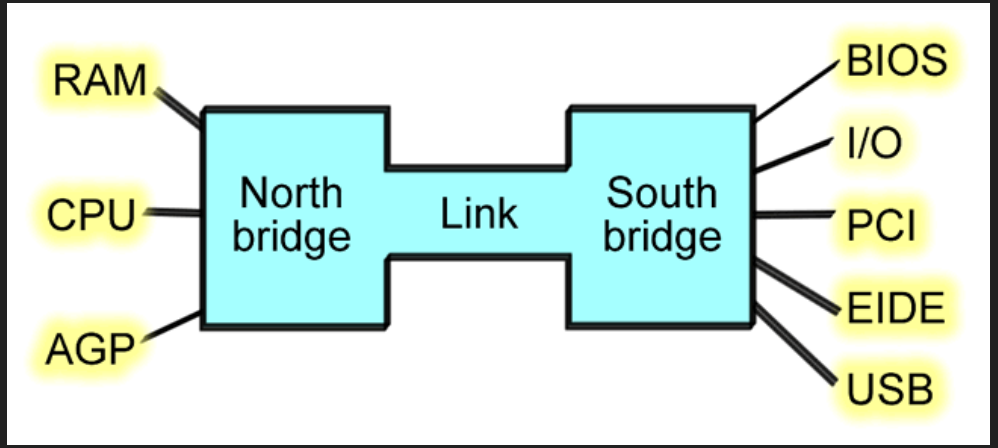




ne] Available at:

Bridge [3]

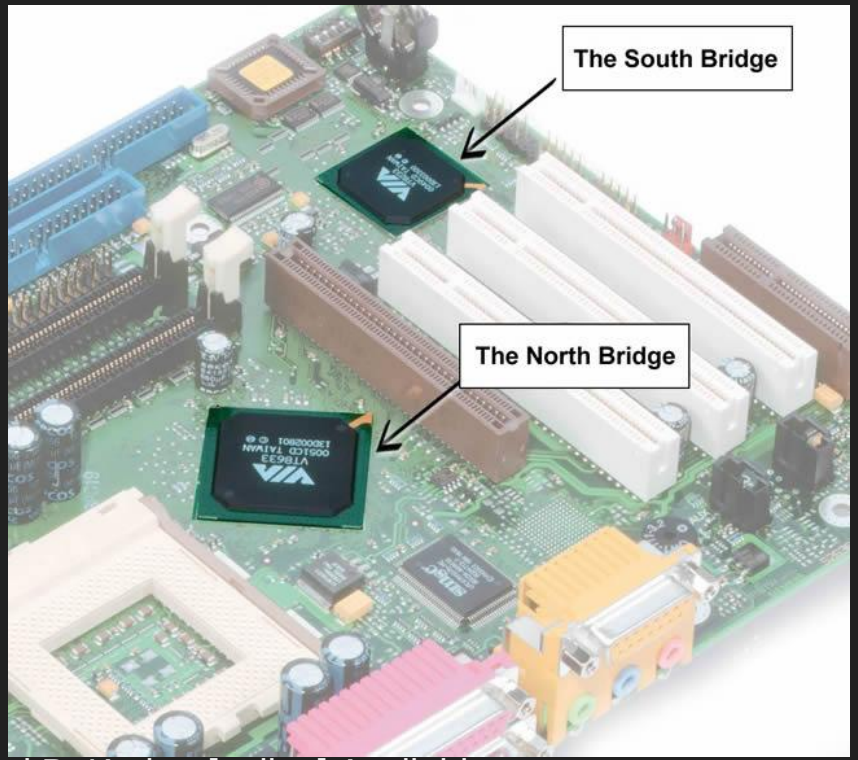
- Until 2008 the most widespread chipset architecture consists of two chips, usually called the north and south bridges.



[3]book.huihoo.com. (n.d.). PC Architecture. A book by Michael B. Karbo. [online] Available at: https://book.huihoo.com/pc-architecture/ [Accessed 8 Jun. 2022].

Bridge [4]

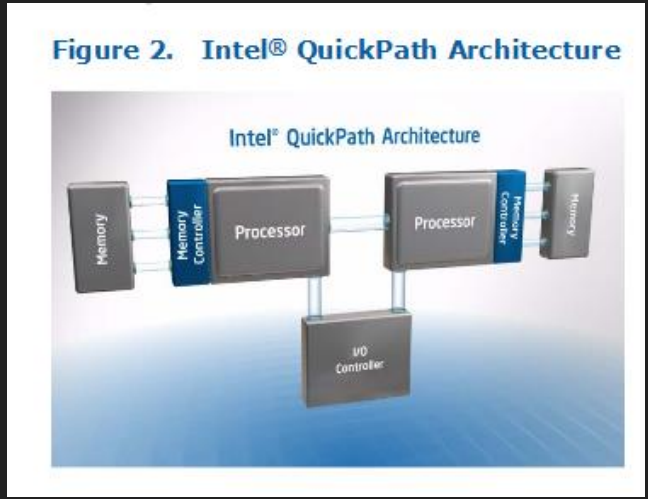
- This division applies to the most popular chipsets from VIA and Intel.
- The **north bridge** and **south bridge** are connected by a powerful bus, which sometimes is called a link channel



[4]book.huihoo.com. (n.d.). PC Architecture. A book by Michael B. Karbo. [online] Available at: https://book.huihoo.com/pc-architecture/ [Accessed 8 Jun. 2022].

Chipset architecture nowadays… [5]

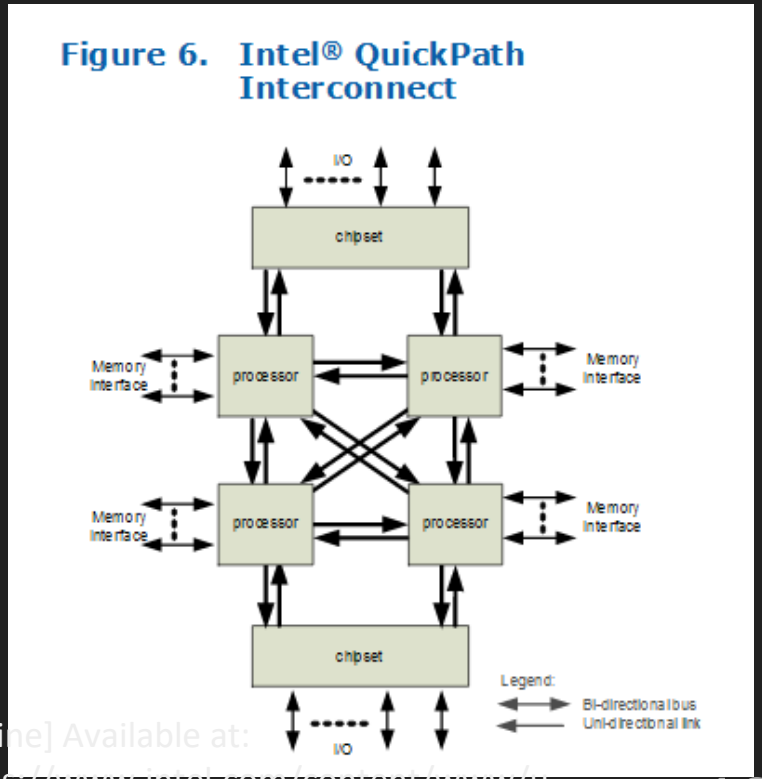
- More modern designs use point-to-point connections
- E.g.: AMD's HyperTransport, Intel's DMI 2.0 or QuickPath Interconnect (QPI).
- These implementations remove the traditional northbridge in favor of a direct link from the CPU to the Platform Controller Hub, southbridge or I/O controller.



[5] Intel. (n.d.). *Intel | Data Center Solutions, IoT, and PC Innovation*. [online] Available at: https://www.intel.com/content/www/us/en/homepage.html?ref=https://www.intel.com/content/www/us/en/quick-path-interconnect-introduction-paper.html [Accessed 8 Jun. 2017].

Chipset architecture nowadays… [6]

- memory controllers integrated into the microprocessors, which are connected together with a high-speed, point-to-point interconnect.
- **Result**: high bandwidth and low latency => faster performance



[6] Intel. (n.d.). *Intel | Data Center Solutions, IoT, and PC Innovation*. [online] Available at: https://www.intel.com/content/www/us/en/homepage.html?ref=https://www.intel.com/content/www/us/en/quick-path-interconnect-introduction-paper.html [Accessed 8 Jun. 2017].

How a CPU works?

- How a CPU works? [7]

[7] Futurology — An Optimistic Future. “How a CPU Works | the CPU Explained.” Www.youtube.com, 7 Dec. 2017, www.youtube.com/watch?v=XQq_1yaVDpM. Accessed 8 June 2022.

Part B: Python

Topics to cover

- Review Function and return values
- Strings
- For and While task more advanced with string

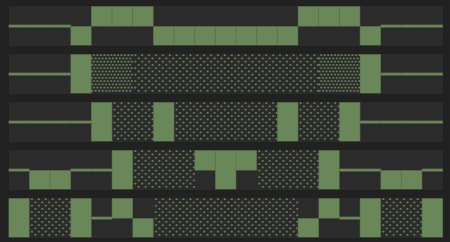
Functions and return values

So far most of the functions we wrote were void, they have no return values, in other words their return value is `None`

Functions and return values

```
def fancy_hello_world():

    fancy_string = """
```



```
Triple quotes allow to span string
across multiple lines
"""

    print(fancy_string)
```

How to make it return a value?

Functions and return values

So far most of the functions we wrote were void, they have no return values, in other words their return value is `None`

Calling the function generates a return value, which we usually assign to a variable or use as part of an expression.

```
def area(radius):

    a = math.pi * radius**2

    return a
```

```
def area(radius):
    return math.pi * radius**2
```

Functions and return values

Since these return statements are in an alternative conditional, only one runs.

```
def absolute_value(x):
    if x < 0:
        return -x
    else:
        return x
absolute_value(3)
```

As soon as a return statement runs, the function terminates without executing any subsequent statements.

Functions and return values

It is important to ensure that every possible path through the program hits a return statement.

```
def absolute_value(x):
    if x < 0:
        return -x
    elif x > 0:
        return x

absolute_value(0)
```

Functions and return values

The function on previous slide is incorrect because if x happens to be 0, neither condition is true, and the function ends without hitting a return statement. If the flow of execution gets to the end of a function, the return value is None , which is not the absolute value of 0.

```
>>> print(absolute_value(0))
```

```
None
```

Solution

```
def absolute_value(x):
    if x < 0:
        return -x
    elif x > 0:
        return x
    elif x == 0:
        return 0
```

Task 4

Write a def compare() function that takes two values, x and y , and returns 1 if x> y , 0 if x == y , and -1 if x < y .

Push code to your GitHub repository

Task 5

Implement the function def test_is_valid(test):

This function must return True if the argument test is an int (no floating point allowed here) and its value is between 1 and 3. To check if a variable is of int type, use the function isinstance().

Example: isinstance(my_variable, int). The former example tests if the variable my_variable is of type int.

Push code to your GitHub repository

String

Strings are not like integers, floats, and booleans.

A string is a sequence, which means it is an ordered collection of other values, i.e. sequence of characters

```
>>> fruit = 'banana'
>>> letter = fruit[1]
```

NB: The offset of the first letter is `zero`

String index

As an index you can use an expression that contains variables and operators:

```
>>> fruit = 'banana'
>>> i = 1
>>> fruit[i]
'a'
>>> fruit[i+1]
'n'
```

But the value of the index has to be an i**nteger.** Otherwise you get:

```
>>> letter = fruit[1.5]

TypeError: string indices must be integers
```

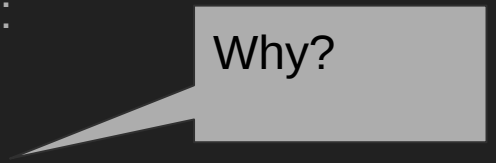
len

len is a built-in function that returns the number of characters in a string:

```
>>> fruit = 'banana'
>>> len(fruit)
6
```

To get the last letter of a string:

```
>>> length = len(fruit)
>>> last = fruit[length-1]
```



Traversal with a for loop

Traversal - pattern of processing when it often starts at the beginning of the string, selects each character in turn, does something to it, and continues until the end

```
index = 0

while index < len(fruit):

    letter = fruit[index]

    print(letter)

    index = index + 1
```

```
for letter in fruit:
    print(letter)
```

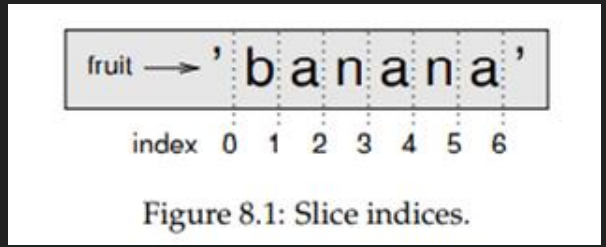


Figure 8.1: Slice indices.

String slices

A segment of a string is called a slice. Selecting a slice is similar to selecting a character:

```
>>> s = 'Monty Python'

>>> s[0:5]

'Monty'

>>> s[6:12]

'Python'
```

String slices

If you omit the first index (before the colon), the slice starts at the beginning of the string. If you omit the second index, the slice goes to the end of the string:

```
>>> fruit = 'banana'

>>> fruit[:3]

'ban'

>>> fruit[3:]

'ana’
```

Strings are immutable

```
>>> greeting = 'Hello, world!'

>>> greeting[0] = 'J'

TypeError: 'str' object does not support item assignment
```

You can’t change an existing string. The best you can do is create a new string that is a variation on the original

```
>>> greeting = 'Hello, world!'
>>> new_greeting = 'J' + greeting[1:]
>>> new_greeting
'Jello, world!
```

Searching

```
def find(word, letter):
    index = 0
    while index < len(word):
        if word[index] == letter:
            return index
        index = index + 1
    return -1

print(find('computer', 'p'))
```

Traversing a sequence and returning when we find what we are looking for is called a search.

Task 5

Modify find function from previous slide so that it has a third parameter, the index in word where it should start looking.

Push updates to Github repository

Looping and counting

```
word = 'banana'
count = 0
for letter in word:
    if letter == 'a':
        count = count + 1
print(count)
```

This pattern of computation called a counter. The variable count is initialized to 0 and then incremented each time an a is found. When the loop exits, count contains the result—the total number of a’s.

String methods

```
>>> word = 'banana'

>>> new_word = word.upper()

>>> new_word

'BANANA
```

dot notation specifies the name of the method, **upper**, and the name of the string to apply the method to, ***word***. The empty parentheses indicate that this method takes no arguments.

A method call is called an **invocation**

we are invoking upper on word.

The in operator

The word in is a boolean operator that takes two strings and returns True if the first appears as a substring in the second:

```
>>> 'a' in 'banana'

True

>>> 'seed' in 'banana'

False
```

Homework 0

The following program counts the number of times the letter a appears in a string:

```
word = 'banana'
count = 0
for letter in word:
    if letter == 'a':
        count = count + 1
    print(count)
```

Rewrite the function so that instead of traversing the string, it uses the three parameter version of find from slide 26 **to count the number of letter ‘a’ occurances.** Push code to your GitHub repository

Homework 1

Implement the function def is_the_same(message1, message2):

This function must return True if the arguments message1 and message2 have the same textual content, independently of the characters' case. The function must abort execution if any of the arguments are not of the type str.

Consider the tests below:

is_the_same("Flamengo", "fLaMeNgO") –> Expected result: True

is_the_same("flamengo", "FLAMENGO") –> Expected result: True

is_the_same("flamengo", "flamingo") –> Expected result: False

Push to https://classroom.github.com/a/U8MfQwDD

Homework 2

Implement the function below

```
def format_name(name, surname):
```

This function must return the name and surname formatted according to the following rule: Initial letter of the name + dot + space + surname + " (" + name + ")". Observe the example below and the expected result:

```
print(format_name("Wallace", "Corbo Ugulino"))
W. Corbo Ugulino (Wallace)
```

Before formatting the name, this function must validate if the arguments are valid. Name and Surname must be of the type str and must have len >=2. In case of invalid arguments, print the appropriate message (see below) and abort the execution:

inv_name ="Invalid argument. Name must be string (len >= 2)."

inv_surname ="Invalid argument. Surname must be string (len >= 2)."

Push to GitHub

Recommended reading

- Dale, Computer Science Illuminated, **Chapter 5, end of the book exercises**
- Downey, Think Python, Chapter 6, 8, end of the Chapter exercises

- Refer to motherboard details and study its components.

- https://github.community/t/the-difference-between-forking-and-cloning-a-repository/10189

- Learn How Chipset directs Traffic

- Learn How PCI Express breaks the Bus barrier

Solutions for seminar 5 homework with explanation

https://www.w3resource.com/python-exercises/python-conditional-exercise-4.php

References

1. Dale, N., & Lewis, J. (2020). Computer Science Illuminated (7th ed.). Jones & Bartlett Learning, Chapter 5
2. Downey, A. (2015). Think python. O'Reilly. Chapter 6, 8
3. Hamedani, M., 2020. Python for Beginners - Learn Python in 1 Hour. [online] Youtube.com. Available at: <https://www.youtube.com/watch?v=kqtD5dpn9C8> [Accessed 5 June 2022].
4. Fernandez, Alex. “The Difference between Forking and Cloning a Repository.” GitHub Support Community, 28 Nov. 2017, github.community/t/the-difference-between-forking-and-cloning-a-repository/10189. Accessed 8 June 2022.
5. book.huihoo.com. (n.d.). PC Architecture. A book by Michael B. Karbo. [online] Available at: https://book.huihoo.com/pc-architecture/ [Accessed 8 Jun. 2022].
6. Intel. (n.d.). Intel | Data Center Solutions, IoT, and PC Innovation. [online] Available at: https://www.intel.com/content/www/us/en/homepage.html?ref=https://www.intel.com/content/www/us/en/quick-path-interconnect-introduction-paper.html [Accessed 8 Jun. 2022].
7. Futurology — An Optimistic Future. “How a CPU Works | the CPU Explained.” Www.youtube.com, 7 Dec. 2017, www.youtube.com/watch?v=XQq_1yaVDpM. Accessed 8 June 2022.