DS610 – Big Data Analytics

Metin Senturk

Today we cover

- Basic Python Knowledge
- Concepts of Parallelism

Python –Functions /Classes

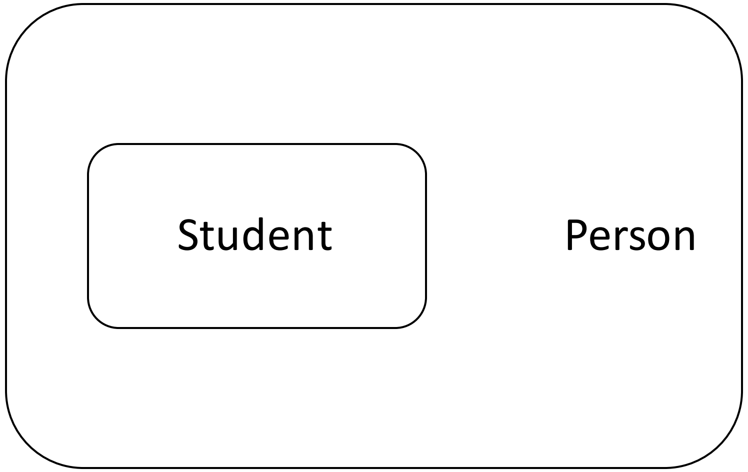
Class is the code template to create objects. **Function** is the block of code that runs when it is called.

```
# function
def get_weather_info():
pass

def kill_process(pid):
pass

def update_model(model, *args, **kwargs):
pass

lambda x, y, z: round((x / y) ** z, 2)
```

```
# class
class NLPException(Exception):
pass

class Processing(object):
def clean_stop_words(self):
pass

def clean_rare_words(self, rare_words_list):
pass
```

Python -Inheritance

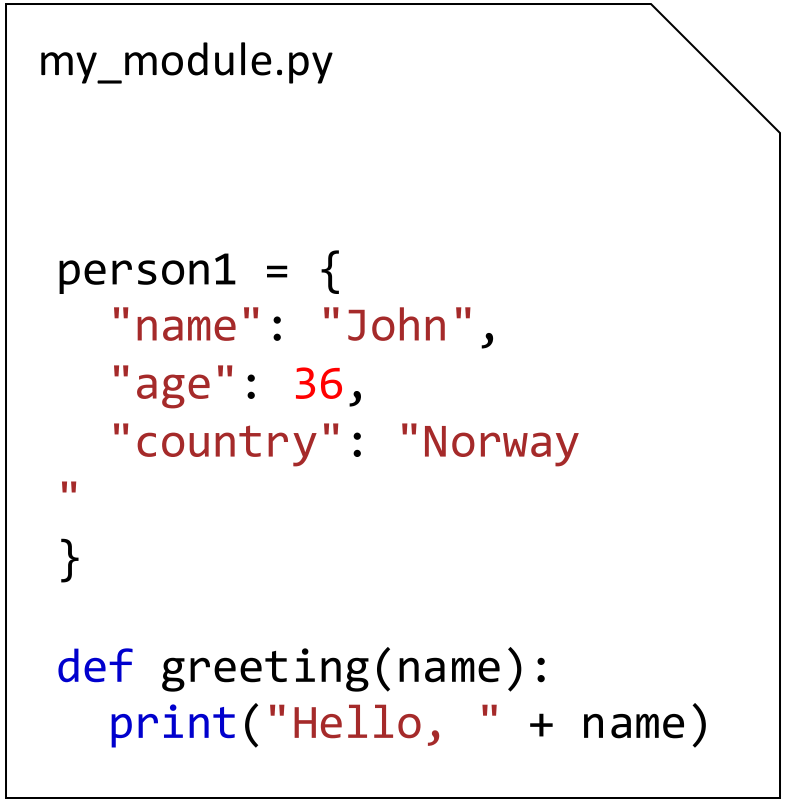
- Inheritance is the ability for a class to inherit another class's all of the **methods** and **properties**.



```python
class Person:
  def __init__(self, fname, lname):
    self.firstname = fname
    self.lastname = lname

  def printname(self):
    print(self.firstname, self.lastname)

class Student(Person):
  pass

x = Student("Mike", "Olsen")
x.printname()
```

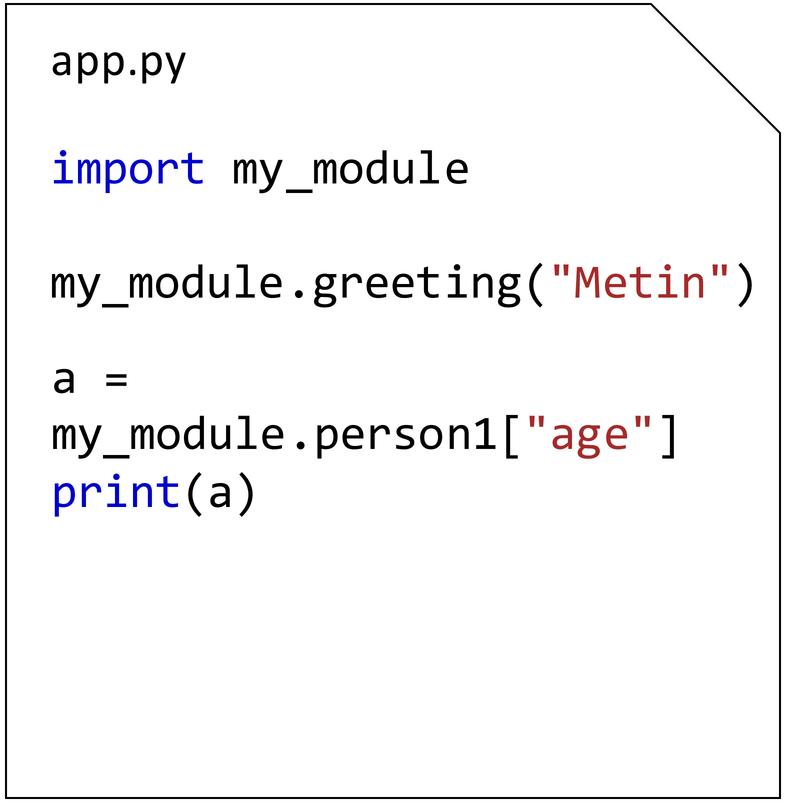
Python -Modules

- Modules is a code library in a file. A file that contains your variable, functions, classes, etc.



my_module.py

```
person1 = {
  "name": "John",
  "age": 36,
  "country": "Norway
"
}

def greeting(name):
  print("Hello, " + name)
```

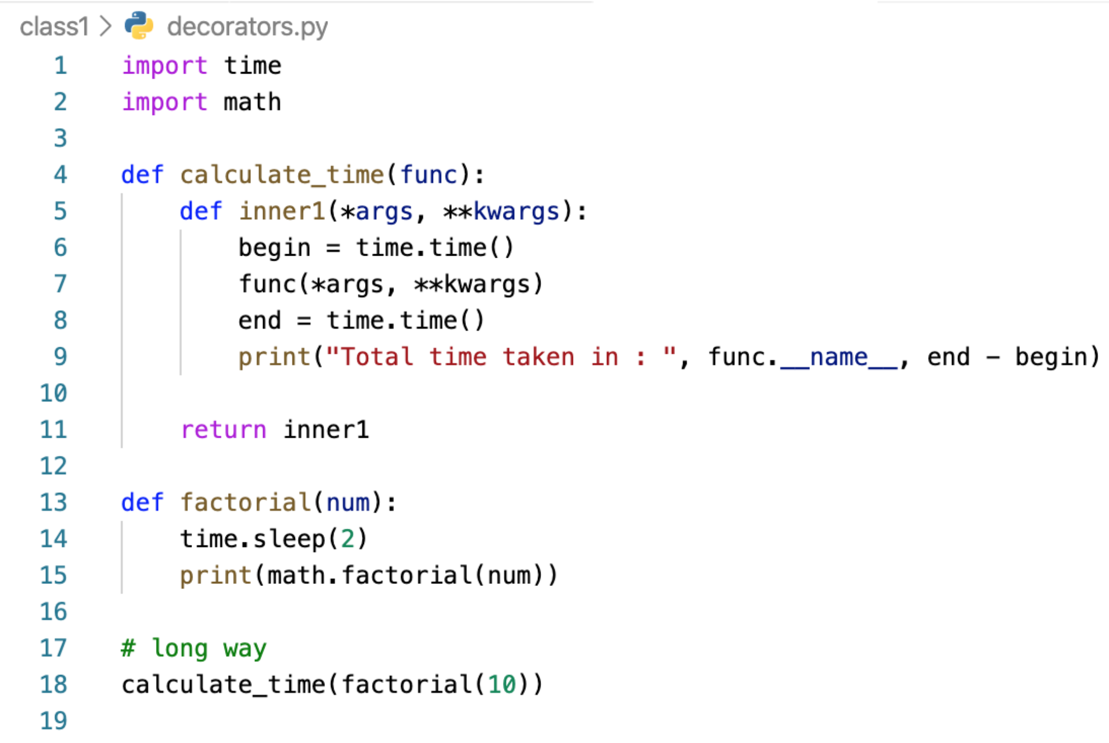


app.py

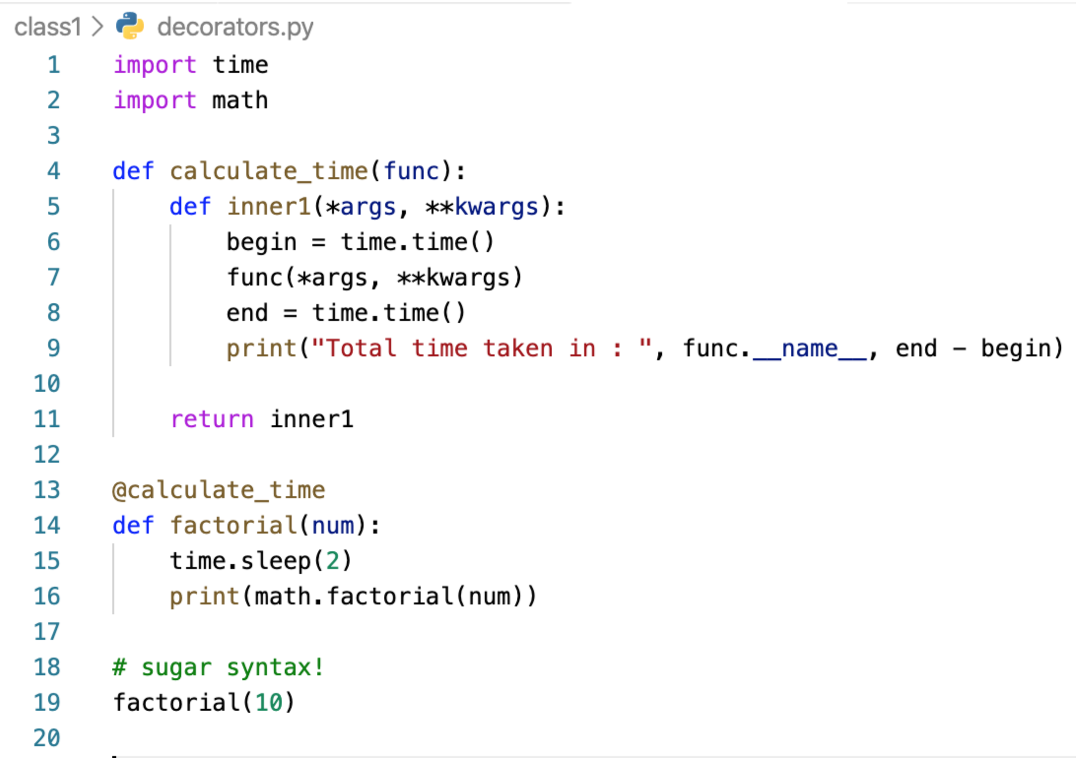
```
import my_module

my_module.greeting("Metin")

a =
my_module.person1["age"]
print(a)
```

Python - Decorators

- Decorators wraps a function, and changes it's behavior. Simply, they are functions that wraps functions!



```python
import time
import math

def calculate_time(func):
    def inner1(*args, **kwargs):
        begin = time.time()
        func(*args, **kwargs)
        end = time.time()
        print("Total time taken in : ", func.__name__, end - begin)

    return inner1

def factorial(num):
    time.sleep(2)
    print(math.factorial(num))

# long way
calculate_time(factorial(10))
```



```python
import time
import math

def calculate_time(func):
    def inner1(*args, **kwargs):
        begin = time.time()
        func(*args, **kwargs)
        end = time.time()
        print("Total time taken in : ", func.__name__, end - begin)

    return inner1

@calculate_time
def factorial(num):
    time.sleep(2)
    print(math.factorial(num))

# sugar syntax!
factorial(10)
```

Python – Virtual Environments

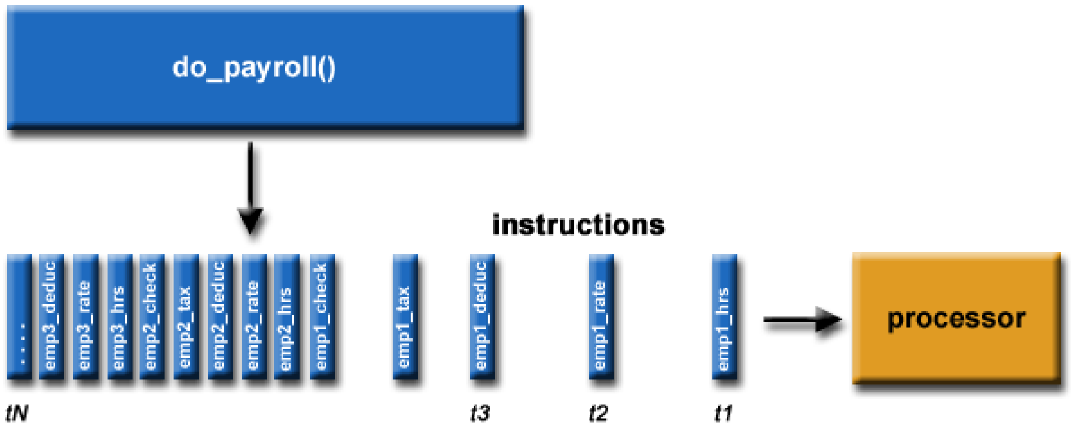
- Python virtual environments is to create an isolated environment for Python projects.
  - Comes default with python3
  - If not there, do "pip install virtualenv"
- Usage (linux):
  - Create env: "python –m venv my_new_environment"
  - Activate env: "source my_new_environment/bin/activate"
  - Deactivate env: "deactivate"
- Example, install bcrypt package in your environment
  - pip -q install bcrypt
  - python -c "import bcrypt; print(bcrypt.hashpw('password'.encode('utf-8'), bcrypt.gensalt()))"

Python –Where to go?

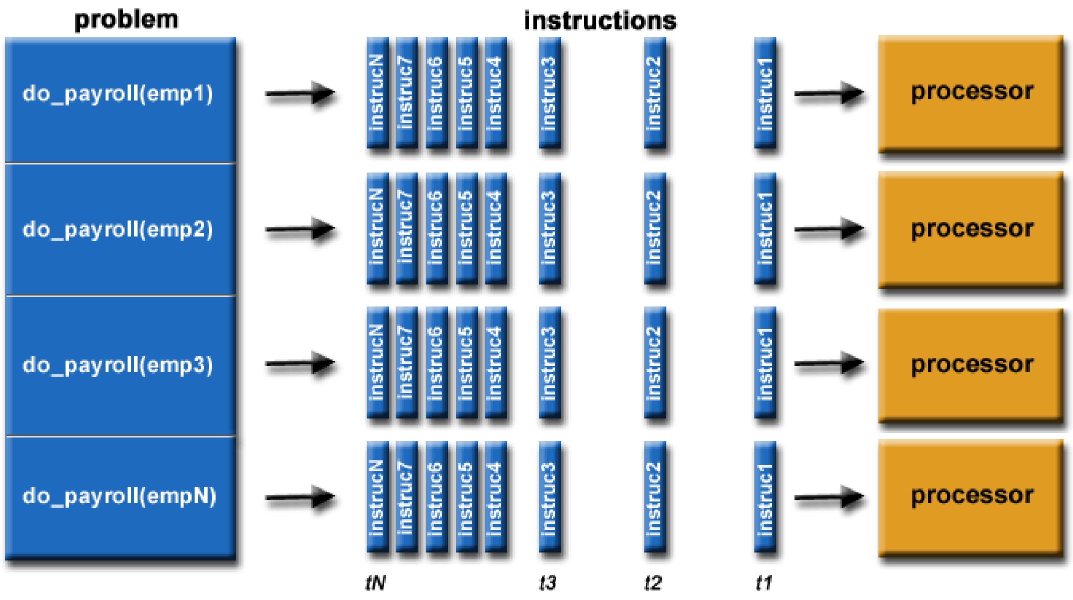
- W3School , https://www.w3schools.com/python/default.asp
- More on decorators, https://realpython.com/primer-on-python-decorators/
- Virtual env vs VirtualEnvWrapper, https://realpython.com/python-virtual-environments-a-primer/

Parallelism in Computer Science

Serial Computing



Parallel Computing



Parallel Computing Resources – We are lucky!

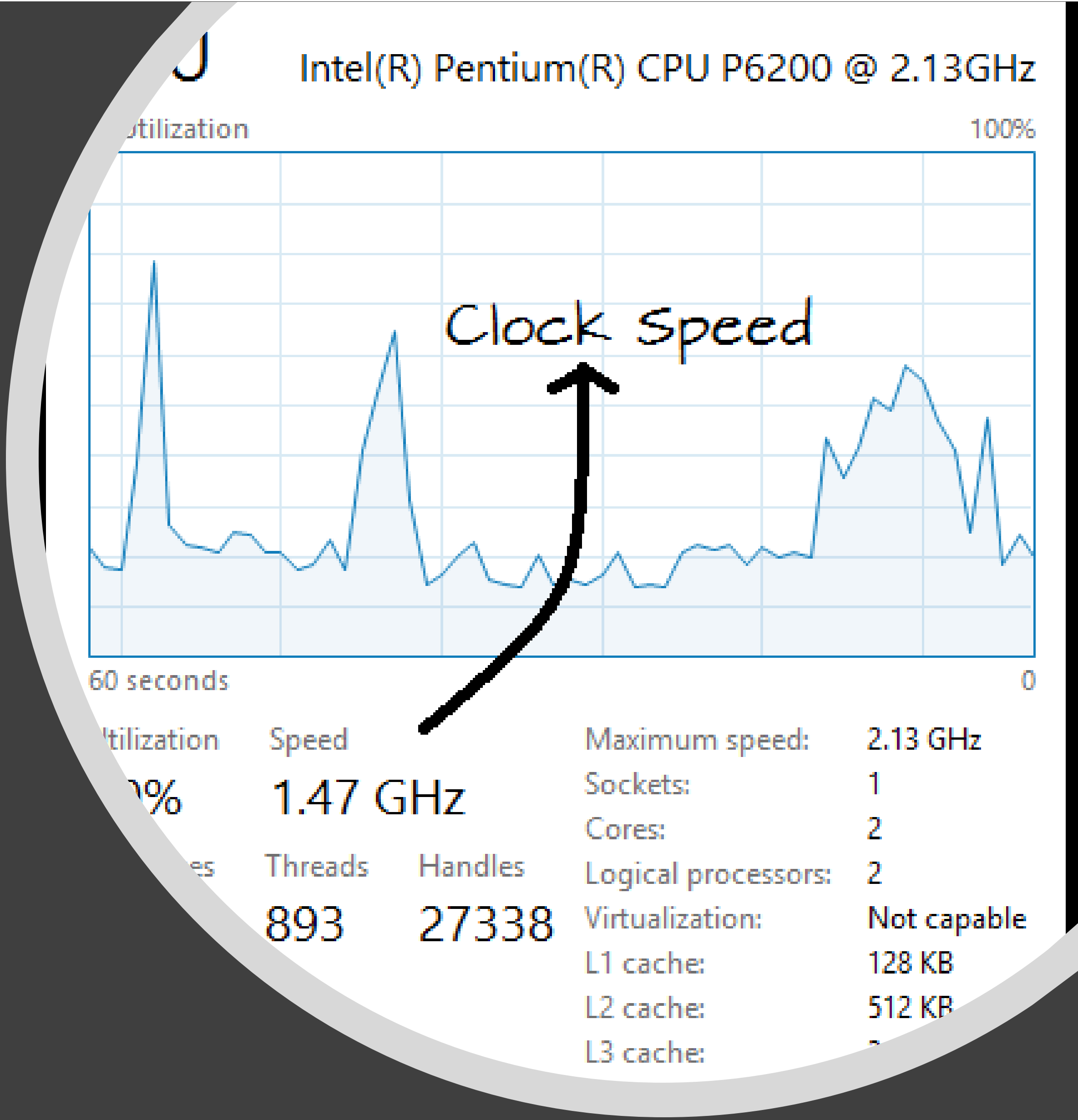
- Almost everything has more than one core, some GPUs has more than 1000 specialized cores, etc.
    - CPU frequencies are no longer increasing because of heat limitations
- Where to use these power? Simulation, large data analysis, deep learning, machine learning.

Parallel Computing

- Computer software are generally written for serial computing. Means, an algorithms divides a problem into smaller instructions, then CPU executes does tasks one after another.
- Three types of parallelism:
    - Data parallelism – A set of tasks operate on independently on disjoint partitions
    - Bit-level – increasing processors (CPU) size
    - Instruction-level – grouping instructions together to run concurrently
    - Task – dividing a problem/task into subtasks and run concurrently

CPU

- Clock Speed? (**clock rate** and **processor speed)**
  - The **CPU speed** determines how many calculations it can perform in one second of time.
  - The CPU requires a fixed number of clock ticks, or cycles, to execution or instruction.
  - Measured in MHz, 1 MHz representing 1 million cycles per second, or in GHz, 1 GHz representing 1 thousand million cycles per second.



tilization
0%

Speed
1.47 GHz

es

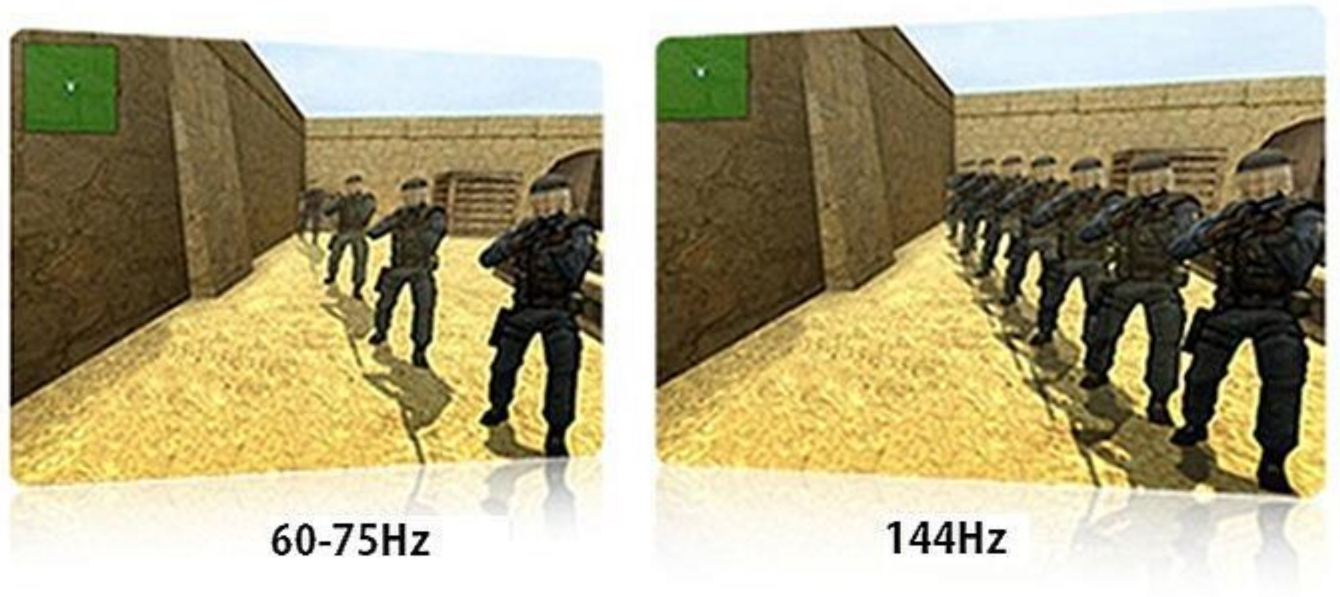
Threads
893

Handles
27338

Maximum speed:	2.13 GHz
Sockets:	1
Cores:	2
Logical processors:	2
Virtualization:	Not capable
L1 cache:	128 KB
L2 cache:	512 KB
L3 cache:	

The Importance of Hertz

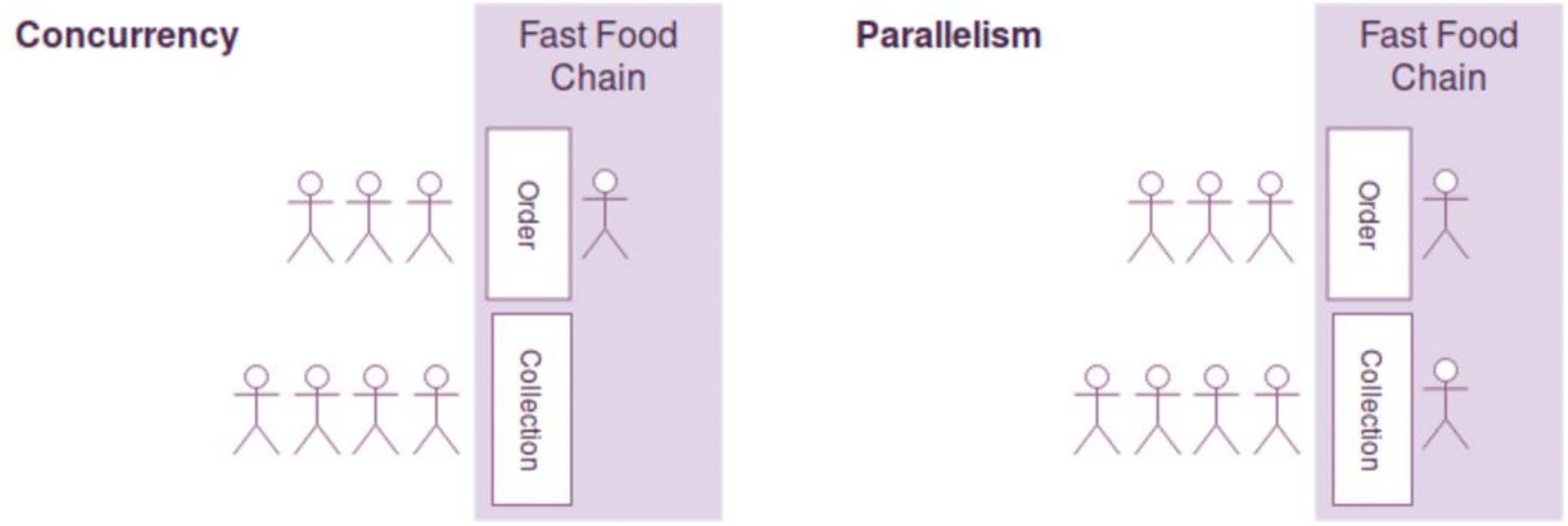
The first processor, the Intel 4004 operated at 740 kHz. Later processors operated in MHz, for example, the Intel Pentium processor was available in speeds of 60 MHz to 300 MHz Today's processors operate in the GHz range.



Parallelism Concepts

- What is a thread?
  - A Thread, or thread of execution, is a software term for the basic ordered sequence of instructions that can be passed through or processed by a single CPU core.
- What is a process?
  - An instance of execution of your file is called a process. Imagine a calc.py, when runs, it turns into a process.
- Multithreading
  - Ability of a CPU running threads of execution concurrently, supported by the OS. Multiple threads share the **same** memory space.
- Multiprocessing
  - The use of two or more CPU in a single computer system. Multiple processes have **different** memory space.
- Concurrent Computing
  - Concurrent computing is a form of computing in which several computations are executed during overlapping time periods—concurrently—instead of sequentially (one completing before the next starts).

Concurreny vs Parallelism

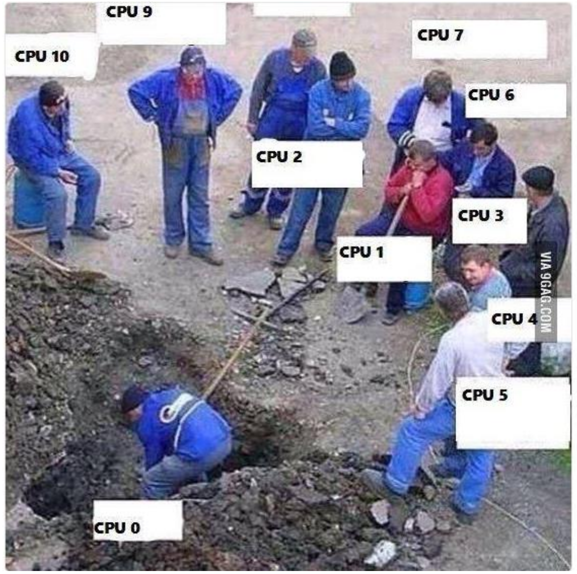


How to do parallel programming?

- Functional parallelism:
  - each process performs a different "function" or executes different code sections that are independent.
  - 2 brothers do yard work (1 edges & 1 mows)
  - 8 farmers build a barn
- Data parallelism:
  - each process does the same work on unique and independent pieces of data
  - 2 brothers mow the lawn
  - 8 farmers paint a barn
- Task parallelism:
  - each process performs the same functions but do not communicate with each other, only with a “Master” Process. These are often called “Embarrassingly Parallel” codes.

How to do parallel programming?

- Programming level
  - multiprocessing library in python
  - "ray", "numba" packages
- Computing level
  - Hypervisor systems
    - Single computer with multiple virtual computers
  - Clustered systems (Network connected single machines)
    - Apache Hadoop
    - Apache Spark



CPU usage with Python

Parallelism, Where to go?

- About parallel computer design and history, https://computing.llnl.gov/tutorials/parallel_comp/
- Threading vs Parallelism, http://www.danielmoth.com/Blog/threadingconcurrency-vs-parallelism.aspx
- Good article on parallelism with Python, https://datascienceplus.com/how-to-achieve-parallel-processing-in-python-programming/
- Another article about multiprocessing with Python, https://towardsdatascience.com/parallelising-your-python-code-85b59c558f97
- Lambda functions and Multiprocessing, https://medium.com/swlh/lambda-functions-and-parallel-programming-d83db203c483