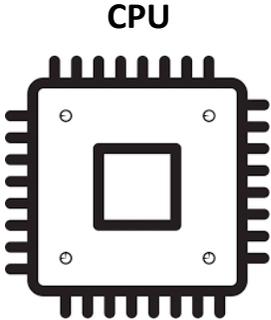


# Activités en première

P. Mariano



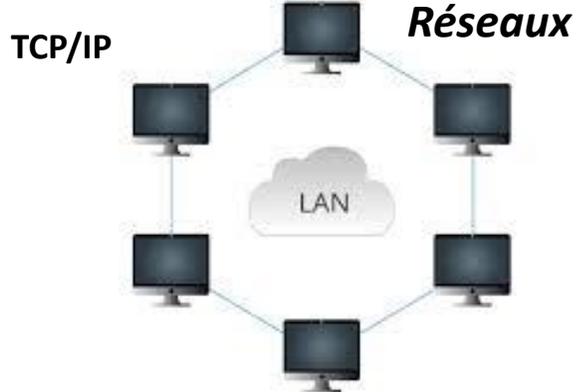
CPU

Modèle de von Neumann

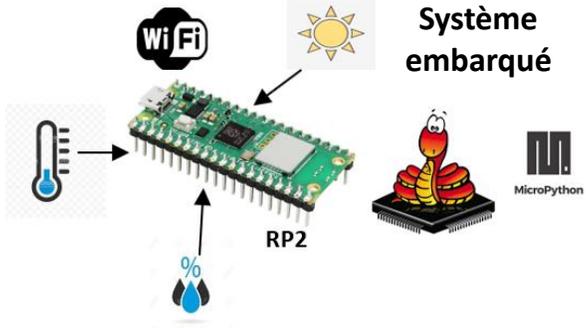


Programmation  
Assembleur

Architectures ...



Architectures ...



Architectures ...

La base 10	15	14	13
La base 2	1111	1110	1101
La base 16	F		

	E	D	F
	Decimal	Hex	Char
	64	40	Ⓞ
	65	41	A

Représentation des données : types et valeurs de base

IHM

**Mesures physiques - Domotique**

Les mesures de température, d'humidité et de pression sont réalisées par un capteur BME280 (Sparkfun).

- Température : de -40°C à 85°C, +/-1°C
- Humidité : de 0 à 100% HR, +/-3%
- Pression : 30kPa à 110 kPa, +/-1hPa

La mesure de luminosité est réalisée par un capteur BH1750 (DFRobot).

- Plage de mesure : 0 à 65536lux
- Précision : 1,2lux

Le template est affichable à partir de ce lien

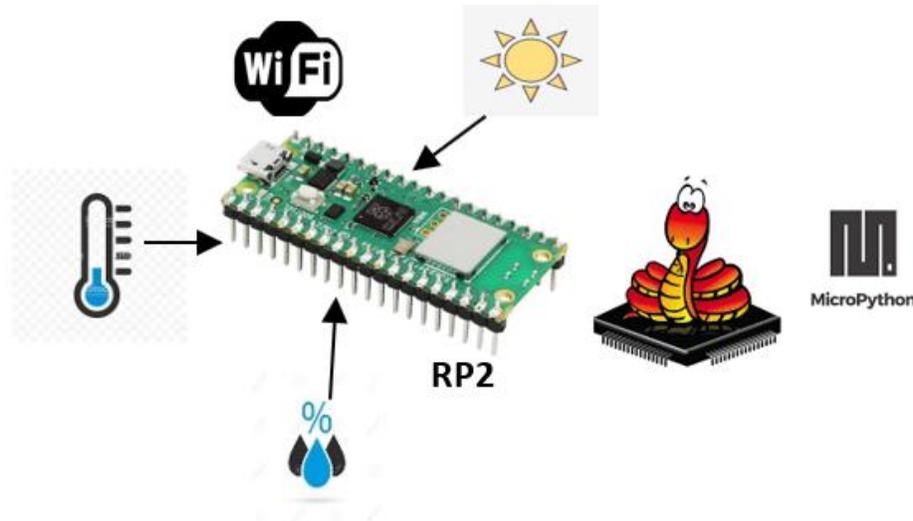
Mesures	Valeur	Unité
Température	20	°C
Humidité	61	%
Pression	980	hpa
Luminosité	250	lux

Nom - Prénom - Classe - Spécialité - Adresse du lycée

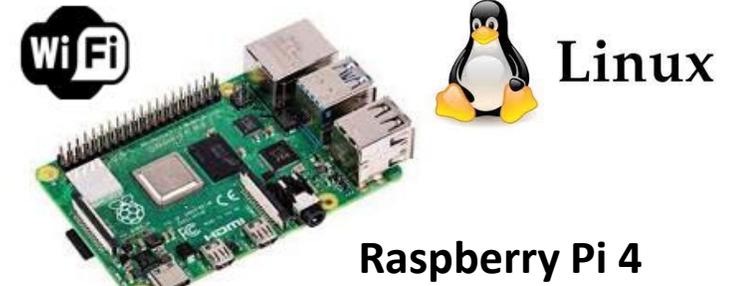
**Linux**

Systèmes d'exploitation

## Exemple de mini projet (guidé)



## Domotique



Raspberry Pi 4

 Mesures physiques - Domotique

Les mesures de température, d'humidité et de pression sont réalisées par un capteur BME280 (SparkFun).

- Température : de -40°C à 85°C, +/- 1°C
- Humidité : de 0 à 100% HR, +/- 3%
- Pression : 30kPa à 110 kPa, +/- 1hPa

La mesure de luminosité est réalisée par un capteur BH1750 (DFRobot).

- Plage de mesure : 0 à 65536lux
- Précision : 1,2lux

Le template est affichable à partir de ce lien

Mesures	Valeur	Unité
Température	20	°C
Humidité	61	%
Pression	980	hpa
Luminosité	250	lux

Nom - Prénom - Classe - Spécialité - Adresse du lycée



```
import csv
with open('scientifiques.csv') as fichier:
    scientifiques = list(csv.DictReader(fichier,
    delimiter=";"))
```

Code partiel

	A	B	C	D	E	F
1	nom	prenom	jour	mois	annee	projet
2	Kernighan	Brian	1	1	1942	programmer avec style
3	Hopper	Grace	9	12	1906	code machine
4	Torvald	Linus	28	12	1969	systeme d'exploitation
5	Knuth	Donald	10	1	1938	tout sur les algorithmes

### Traitement de données en tables

### Représentation des données : types construits

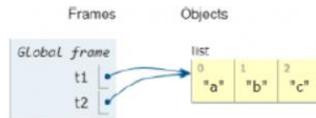


#### Dictionnaire

```
dict = {} # dictionnaire vide
dict = {'nom': 'Martin', 'prenom': 'Pierre-Emile'}
```

#### Tableau

```
t1 = ['a', 'b', 'c']
t2 = t1
```



# Activités en première

Y. Leclerc



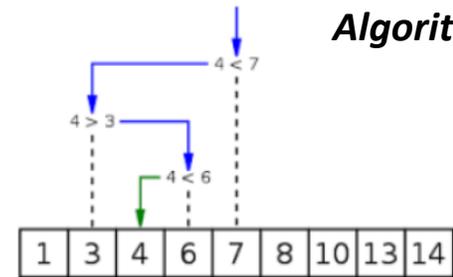
Tri par sélection

```
def TriSelection(T,n):
    for i in range(n-1):
        Posmin=i
        for j in range(i+1,n):
            if T[j]<T[Posmin]:
                Posmin=j
        T[i],T[Posmin]=T[Posmin],T[i]
```

3	7	2	6	5	1	4
1	7	2	6	5	3	4
1	2	7	6	5	3	4
1	2	3	6	5	7	4
1	2	3	4	5	7	6
1	2	3	4	5	7	6
1	2	3	4	5	6	7

### Recherche dichotomique

### Algorithmique



```
//Boucle de recherche partielle
Tant que trouvé != vrai et début <= fin:
    mil ← partie_entière((début + fin)/2)
    si t[mil] == val:
        trouvé ← vrai
    sinon:
        si val > t[mil]:
            début ← mil+1
        sinon:
            fin ← mil-1
```

## Exemples de mini projet (guidé)

```
A vous de jouer!  
 0 1 2 3 4 5 6
```

```
-----  
5 |.|.|.|.|.|.|.|  
4 |.|.|.|.|.|.|.|  
3 |.|.|.|x|.|.|.|  
2 |.|.|0|x|0|.|.|  
1 |.|.|x|x|x|.|.|  
0 |0|0|0|x|x|.|0|  
-----
```

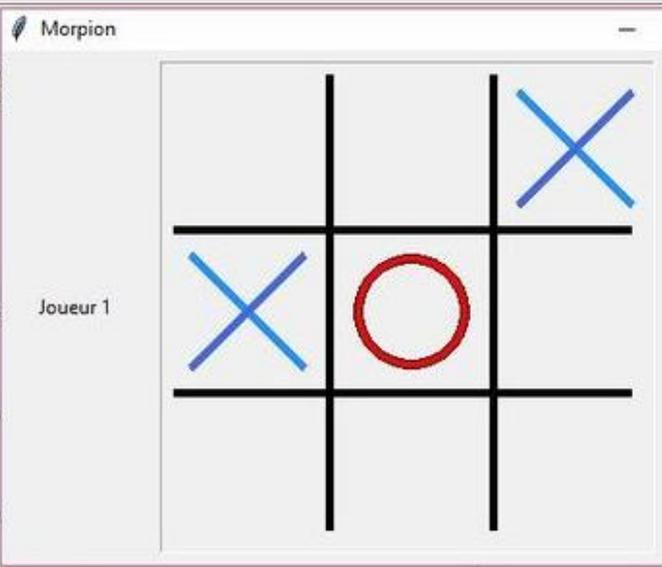
```
#####  
Bravo ! Humain (1-x) Aléa vous avez gagné !  
La partie est terminée !
```

```
#####
```

**Puissance 4**

## Tic-Tac-Toe

```
self.joueur = 1  
  
def analyser_position_clic(self, event):  
    #Detection de la position de la case  
    # 0 | 1 | 2  
    # - + - + -  
    # 3 | 4 | 5  
    # - + - + -  
    # 6 | 7 | 8  
  
    if self.grille[0] == 0:  
        if event.x > 0 and event.x < 100:  
            if event.y > 0 and event.y < 100:  
                i = 1  
                self.x = 55  
                self.y = 55  
            if event.y > 100 and event.y < 200:  
                i = 4  
                self.x = 55  
                self.y = 155  
            if event.y > 200 and event.y < 300:  
                i = 7  
                self.x = 55  
                self.y = 255  
        elif event.x > 100 and event.x < 200:  
            if event.y > 0 and event.y < 100:  
                i = 2  
                self.x = 155  
                self.y = 55
```



Morpion

Joueur 1

Morpion