Ministère d'éducation tunisienne

LYCEE SECONDAIRE 7 AVRIL 1943 – MENZEL TEMIME

NOM & PRENOM & CLASSE

.....
.....
.....

Algorithme et programmation PYTHON

2022-2023

Éditer par : Mr. Ben Mansour Hassen

- *Bref du cours*
- *Activités du cours*
- *Exercices corrigés*
- *2ème Sciences*
  - *2ème Technologies Informatiques*
    - *3ème Sciences Informatiques*
      - *Bac Sciences Informatiques*
- *3ème Sciences :*
  - *Mathématiques*
  - *Expérimentales*
  - *Techniques*
    - *Bac Sciences Mathématiques*
    - *Bac Sciences Expérimentales*
    - *Bac Sciences Techniques*

Série rappel 1

Exercice n°1

Soit l'algorithme suivant :

```
Algorithme ex01
Début
        Nbélève←19
        ……
        len←nb max≥nb
        ……
        p% ← nb div nb max *100
        ……
        classe ←"Bac  Math  "+"2022-2023"
        ……
        écrire("La taille de ",CLASSE, "est  valide ? : ," len," et le pourcentage
        des élèves est :",p%)
        ……
Fin
```

Questions :

1- Corriger l'algorithme précédent par ce qui convient
2- Sachant qu'on a une constante nbmax =20, donner, en justifiant le choix des natures et des types, le TDO associé à cet algorithme.
3- Dans cet algorithme, est-ce qu'on peut utiliser lire avec l'objet len, si non justifier ta réponse

Exercice n°2

Compléter le tableau suivant :

Le nom d'une variable peut être comme suit	Algorithme			Python 3		
	Juste	Fausse	Justification (si fausse)	Juste	Fausse	Justification (si fausse)
For						
numéro						
num eleve						
2ds						
Ds2						
Num_tel						
from						
Remplaçant						

Exercice n°3 :

Soit la séquence d'affectations suivantes :

T[0] ← 2
T[1] ← 10
T[2] ← T[0] * T[1] DIV 6
T[3] ← 3* T[2] - T[1]
T[4] ← Pos("Bienvenu","venu") + 4
T[5] ← Long("Devoir")
T[6]← Valeur(Sous_Chaine("tu dois répondre correctement",8,9))
T[7] ← T[6] - T[2] MOD 30
T[9] ← T[4] -1

- Déclarer le tableau T puis donner le contenue final de chacune de ses cases

Objet	Type/Nature

- Déclarer le tableau T en Python 3

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

Exercice n°4

- Soit x une constante = “9”, b une constante =”h” et m une constante = “H”

Ordonner le caractère “m” avec x, b et m.

.....

- Dans un contexte algorithmique, comparer les chaines suivantes :

Ch1="Family".....ch2="Famille"
Ch1="famille".....ch2="Family"
Ch1="2".....ch2="199999"

- Évaluer les expressions écrites en python 3 suivantes :

Situation	Expression	Résultat	Type du résultat
ch1 = 'Bon' et ch2 = 'jour'	ch = ch1 + ch2		
ch = 'Code Python'	L1 = len(ch)		
	L2 = len(‘ch’)		
ch1 = 'Bonjour' ch2,ch3='jour', 'bon'	p1 = ch1.find(ch2)		
	p2 = ch1.find(ch3)		
	p3 = ch1.find(‘o’)		
n, ch = 2020,"Année "	msg = ch + n		
ch1= "2020"	int (ch1)		
	float (ch1)		
ch2 = "14.25 "	int (ch2)		
	float (CH2)		
ch = 'Python'	ch*3 donne		
	ch.isupper ()		
	ch. upper ()		

Exercice n°5

•En utilisant une structure conditionnelle simple (réduite ou complète), donner une séquence d’instructions équivalente à chacune des séquences suivantes :

Algorithme sequence_1 Début x← a > b Fin	....
Algorithme sequence_2 Début Si (x>0) alors Si (y>0) alors Si (z>0) alors Res← 1 Sinon Res ← 0 Fin si Fin	....

• En utilisant une structure simple, donner une séquence d'instructions équivalente à chacune des séquences suivantes :

Algorithme sequence_1 Début Si a>b alors INC←a-b Sinon INC←b-a Finsi Fin INCONNUE	 …… …… …… …… ……
Algorithme sequence_2 Début Si (x>y) alors Res ←x Sinon Res←y Fin si Fin	…… …… …… …… …… …… …… ……

Exercice n°6

On considère les séquences A, B, C, D, E et F suivantes :

Séquence A :

```
for i in range(1,3):
    print(i)
```

Séquence B :

```
for i in range(1,3):
    print(i)
print(i)
```

Séquence C :

```
for i in range(1,3):
    print(i)
    print(i)
```

Séquence D :

```
for i in range(3,1):
    print(i)
```

Séquence E:

```
for i in range(3,1,-1):
    print(i)
```

Séquence F:

```
for i in range(1,3):
    print(i,end=',')
```

Déterminer manuellement les affichages générés par chacune des séquences A, B, C, D, E et F.

Séquence A :	***Séquence B :***	***Séquence C :***	***Séquence D :***	***Séquence E;***
……	……	……	……	……
……	……	……	……	……
……	……	……	……	……
……	……	……	……	***Séquence F:***
……	……	……	……	……
……	……	……	……	……
……	……	……	……	……

Exercice n°7

- Donner la valeur de i donnée par les deux algorithmes suivants :

Algo alog_repeter debut i←0 répéter i←i+1 Jusqu'à i=5 fin	Algo algo_tant_que Debut i←0 Tant que (i=5) faire i←i+1 Fin tant que fin	
i=	i=	***Algorithme modifié***

- Si les deux résultats (i) ne sont pas identiques, modifier un de ces algorithmes pour avoir le même résultat (on veut un résultat fini).

Exercice n°8

En utilise une approche modulaire, on veut écrire un algorithme qui permet de saisir deux chaines de caractères alphanumériques non vides ch1 et ch2 puis les permuter et les afficher avec conversion de ch1 en majuscule et ch2 en miniscules.

Annexe

1.1 Les fonctions applicables sur les types numériques

SYNTAXE		RÔLE	EXEMPLE
Algorithme	**Python**		
Ent (x)	int (x)	Tronquer la partie décimale d'un réel x	Ent (3.14) =3 Ent (-3.14) = -3
Arrondi(x)	round(x)	Retourne l'entier le plus proche de la valeur de x	Arrondi (9.5) = 10 Arrondi (9.4) = 9 Arrondi(-9.5)= -10
Racine_carré(x)	sqrt (x)	Retourne la racine carrée d'un nombre x positif	Racine_carré (4)=2.0
Sin (x)	sin(x)	Donne le Sinus de x (x en radian)	Sin (π /2) =1
Cos(x)	cos(x)	Donne le cosinus de x (x en radian)	Cos(π/2)= 0
Abs(x)	fabs(x) abs(x)	Retourne la valeur absolue de x (tjs réel) Retourne la valeur absolue de x	Abs(1)=1 Abs (-3)= 3
Aléa	random()	Retourne un réel aléatoire de l'intervalle [0..1[	0.5869
Aléa(vi,vf)	randint(vi,vf)	Retourne un entier aléatoire de l'intervalle [vi, vf].	Aléa(100,200)= 180

Remarque : en Python,

- Pour que les fonctions **random** et **randint** nous donne des résultats il faut importer le module **random** (from random import *)
- Pour que les fonctions **sin**,**cos**,**sqrt** et **fabs** nous donne des résultats il faut importer le module **math** (from math import *)
- Arrondi permet de……

Exemple : Arrondi(12.5)=12

1.2 Les fonctions applicables sur le type caractère

Nom en Algorithme	Code en Python	Rôle	Exemples
ORD(c)	ORD(c)	Retourne le code ASCII du caractère **c**.	ORD ("B") vaut 66
CHR(n)	CHR(n)	Retourne le caractère dont le code ASCII est **n**.	CHR(97) vaut "a"

1.3 Les fonctions applicables sur les chaine de caractères

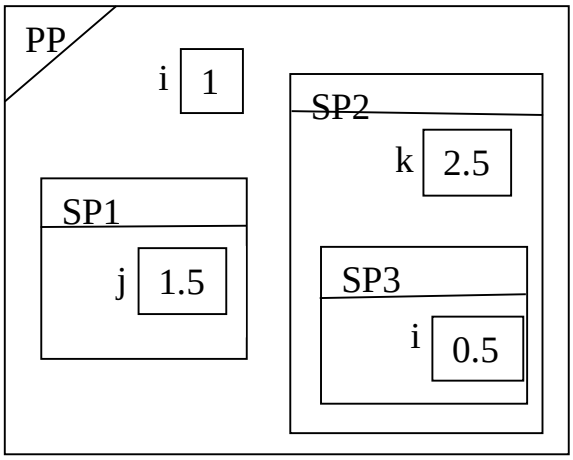
Algorithmique	Python	Rôle
+	+	Permet la concaténation d'un ensemble de chaînes de caractères.
Long(ch)	len(ch)	Retourne le nombre de caractères de la chaîne ch.
Pos (ch1, ch2)	Ch2.find(ch1)	Retourne la première position de la chaîne **ch1** dans la chaîne **ch2**.
Convch (x)	str(x)	Retourne la conversion d'un nombre x en une chaîne de caractères.
Estnum (ch)	ch.isnumeric () ch.isdigit()	Retourne **Vrai** si la chaîne ch est convertible en une valeur numérique, elle retourne **Faux** sinon.
Valeur (ch)	int(ch)	Retourne la conversion d'une chaine ch en un entier, sinon elle provoque une erreur.
	float(ch)	Retourne la conversion d'une chaine ch en un réel, sinon elle provoque une erreur.
Sous_chaine (ch, d, f)	ch [d:f]	Retourne une partie de la chaîne ch à partir de la position d jusqu'à la position f (f exclue).
Effacer (ch, d, f)	ch=ch [:d]+ch [f:]	Efface des caractères de la chaîne ch à partir de la position d jusqu'à la position f (f exclue).
Majus (ch)	ch.upper()	Convertit la chaîne ch en majuscules.

Remarque : En python, Les chaînes de caractères sont immuables (on ne peut pas changer leurs valeurs). Exemple l'instruction ch[0]= "a" est ***interdite***

Exercice n°1

Tracer le TDOG et les TDOL associés aux modules utilisés dans le schéma ci-contre

Visibilité de (+type)	i	j	k
PP			
SP1			
SP2			
SP3			



Exercice n°2

On veut calculer le C^n_p où n et p deux entiers données telles que $1 \leq p \leq n \leq 6$.

- Analyser le problème principal
- Élaborer les algorithmes du programme principal ainsi que les modules envisagés.
- implémenter le programme en Python.

Exercice n°3

Écrire l'algorithme et l'implémentation en Python d'un programme permettant de saisir un entier n (tel que $5 \leq N \leq 150$) puis de calculer la somme $S= 1- 1/2^2 + 1/3^2 - 1/4^2 +\ldots..\pm 1/N^2$;

Exercice n°4

Deux entiers naturels strictement positifs m et n sont dits **amis** si et seulement si :

- La somme des diviseurs de **m** sauf lui-même est égale à **n**
- La somme des diviseurs de **n** sauf lui-même est égale à **m**

Ex : 220 et 284 sont deux nombres amis, en effet :

D284 = {1, 2, 4, 71, 142, 284} ensembles des diviseurs de 284

D220 = {1, 2, 4, 5, 10, 11, 20, 22, 44, 55, 110, 220} ensembles des diviseurs de 220

284 = 1 + 2 + 4 + 5 + 10 + 11 + 20 + 22 + 44 + 55 + 110

220 = 1 + 2 + 4 + 71 + 142

Écrire l'algorithme du programme nommé **AMIS** qui permet de déterminer puis afficher si deux entiers naturels donnés m et n sont amis ou non.

Exercice n°5

Écrire un programme qui permet d'éliminer toutes les espaces supplémentaires d'une chaine CH données puis compter ses mots.

Ex : si CH=″ Bac scientifique 2019 ″ →le programme affichera ″Bac scientifique 2019″ et CH contient 3 mots.

Exercice n°6

Écrire un programme qui permet d'afficher à l'envers une chaine CH donnée mot par mot.

Ex : si CH=″Bac scientifique 2019″ →le programme affichera ″2019 scientifique Bac″

Exercice n°7

Écrire un programme qui permet de saisir une phrase ph de telle façon qu'elle ne contient pas des mots en double. Puis afficher le plus long mot de ph ainsi que sa longueur.

Ex : Si PH=″Bac scientifique 2019″ →le plus long mot est ″scientifique″ et sa longueur = 12

Exercice n°8

Tout nombre positif de deux chiffres ab, tel que a ≠ b, possède une liste appelée ″ **liste vers 9** ″.

Le principe est le suivant :

- on calcule la différence entre ab et son symétrique ba;
- Le résultat trouvé subit le même traitement;
- On répète ce processus jusqu'à obtenir une différence =9.

L'ensemble constitué par le nombre initial et les résultats des différences est appelé ″ liste vers 9 ″.

Ex :

Soit x =18 ; 81-18 = 63 ; 63-36=27 ; 72-27 = 45 ; 54-45 = 9 → Fin du traitement La liste vers 9 est la suivante : 18 63 27 45 9

Écrire un programme qui permet d'introduire un nombre positif composé de deux chiffres obligatoirement différents, de générer et afficher sa ″ liste vers 9 ″. La solution doit êtres modulaires

Exercice n°9

Sachant que :

Pour x très $\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots + \frac{(-1)^i * x^{2i+1}}{(2i+1)!}$ proche de zéro.

Écrire un programme Python intitulé SIN qui permet de calculer sin(x) en utilisant la formule ci-dessus. Le calcul s'arrête lorsque la différence entre deux termes consécutifs devient inférieure ou égale à 10^{-4}. La dernière somme calculée est une valeur approchée de sin(x).

Exercice n°10

On veut écrire un programme Python qui permet de déterminer et d'afficher toutes les représentations sous forme de sommes d'entiers consécutifs d'un entier n donné (où n est un entier positif de deux chiffres).

Ex : Si n=45 alors le programme affichera :

45 = 1+2+3+4+5+6+7+8+9

45= 5+6+7+8+9+10

45= 7+8+9+10+11

45= 14+15+16

45= 22+23

Exercice n°11

Écrire un programme en Python qui calcule et affiche les N premiers termes de la suite de **Fibonacci**, définie par la relation suivante :

$$U_{n+2} = U_{n+1} + U_n \text{ ; } U_0=1 \text{ et } U_1=2$$

Exercice n°12

On veut écrire un programme qui permet de saisir la date de naissance d'une personne **DATE** sous la forme suivante "jj/mm/aaaa", puis, additionner les chiffres correspondants de **DATE**.

Au nombre obtenu, on refait le même procédé jusqu'à ce qu'on obtienne un nombre composé d'un seul chiffre. Ce nombre est appelé nombre de chance.

Ex :

Si la date est 10/02/1998

On additionne 1+0+0+2+1+9+9+2 on obtient 24 qui est un nombre de deux chiffres

On additionne maintenant 2+4 on obtient 6 c'est un nombre d'un seul chiffre

D'où 6 est le nombre de chance de la personne née en 10/02/1998

Exercice n°13

Écrire un programme Python qui permet de saisir un entier n positif de 2 chiffres puis d'afficher s'il est premier ou non.

Modifier le programme a fin qu'il permet d'afficher (max 5 par ligne) tous les entiers premiers de 3 chiffres.

N.B. On rappelle qu'un entier >1 est dit premier s'il est divisible que sur 1 et lui-même.

Exercice n°14

Écrire un programme Python qui permet d'afficher (maximum 8 couples par ligne) tous les couples premiers cousins (a , b) ≤ 1000. Un couple (a , b) est dit premier cousin si et seulement si a et b sont premiers et a-b =4.

Ex : (7 , 11) est un couple premier cousin.

Exercice n°15

Un nombre est dit super premier s'il est un nombre premier de 5 chiffres et si en supprimant des chiffres à partir de sa droite, le nombre restant est aussi premier.

Ex : 59399 est **super premier**.

Écrire un programme qui permet d'afficher tous les entiers super premiers.

Annexe

Les modules

a) *Les Fonctions*

- **La déclaration**

Fonction Nom_fonction (pf_1: $type_1$, pf_2: $type_2$, … , pf_n : $type_n$) **: Type_résultat**
DEBUT
Traitement
Retourner Résultat
FIN

- **L'appel**

Objet ← Nom_fonction (pe_1, pe_2, …, pe_n)

b) *Les Procédures*

- **La déclaration**

Procédure Nom_procédure (pf_1: $type_1$, pf_2: $type_2$, … , pf_n: $type_n$)
DEBUT
Traitement
FIN

- **L'appel**

Nom_procédure (pe_1, pe_2, … , pe_n)

- **Le mode de passage**

Si le mode de passage est par référence (par adresse), on ajoutera le symbole @ avant le nom du paramètre.

N.B. : Une fonction retourne un seul résultat de type simple (entier, réel, booléen, caractère, chaîne).

Exercice n°1

Écrire une analyse qui permet calculer la somme des éléments d'un tableau de taille N (supposons que $5 \leq N \leq 25$)

Ex :

Si N=6 et T= | *15* | *2* | *13* | *10* | *18* | *1* | *Alors s=59*

Exercice n°2

Écrire un algorithme qui permet calculer la somme des éléments paire d'un tableau de taille N (supposons que $5 \leq N \leq 25$)

Ex :

Si N=6 et T= | *15* | *2* | *13* | *10* | *18* | *1* | *Alors s=30*

Exercice n°3

Écrire un programme Python qui permet calculer la somme des éléments d'indice impaire d'un tableau de taille N (supposons que $5 \leq N \leq 25$)

Ex :

Si N=6 et T= | *15* | *2* | *13* | *10* | *18* | *1* | *Alors s=46*

Exercice n°4

Écrire un programme qui permet de former un tableau T à partir des mots d'une CH données de telle façon chaque case de T prend l'entier formé par les chiffres contenus de chaque mot dans ch.si un mot ne contient pas un chiffre l'entier est 0.

Ex :

Si CH="Bac scientifique 2019" →T= | 0 | 0 | 2019 |

Exercice n°5

Écrire un programme qui permet de saisir les moyennes des n élèves ($5 \leq n \leq 15$) puis afficher la moyenne du premier de la classe, du dernier et la moyenne générale de la classe.

Exercice n°6

Écrire un programme qui permet de saisir n entiers positifs dans un tableau T puis d'éclater son contenue dans deux autres tableaux A et B de telle façon A prend les entiers pairs de T et B prend ceux qui sont impaires.

Ex : si T= | 25 | 32 | 14 | 0 | 18 | 9 | 11 | 3 | 5 | →le programme donne

A= | 32 | 14 | 0 | 18 |

ET

B= | 25 | 9 | 11 | 3 | 5 |

Exercice n°7

Écrire un programme qui permet de saisir deux tableaux A et B de tailles respectives n et m par des entiers positifs de deux chiffres puis fusionner A et B dans un autre tableau T de telle façon qu'il soit trié par ordre croissant. (Tableau trié= tableau ordonné)

Ex : si → A= | 32 | 14 | 10 | 18 |

ET

B= | 25 | 19 | 11 | 13 | 15 |

le programme donne

T= | 10 | 11 | 13 | 14 | 15 | 18 | 19 | 25 | 32 |

Exercice n°8

Un nombre est dit K-pp ″presque Premiers″, s'il s'écrit sous la forme d'un produit de k nombres premiers non nécessairement distincts.

Ex :

243 = 3*73 est un nombre 2-pp car il est le produit de deux nombres premiers.

32 = 2*2*2*2*2 est un nombre 5-pp car il est le produit de cinq nombres premiers.

17 = 17 est un nombre 1-pp car il est premier.

Écrire un programme qui permet de remplir un tableau T par N($5 \leq N \leq 50$) entiers positifs de 3 chiffres, de chercher et d'afficher les k-pp nombres du tableau T. sachant que K est un entier choisit aléatoirement de l'intervalle [2..5].

Ex :

Pour N = 5, k = 3 et T = | 231 | 846 | 187 | 722 | 490 |

Les nombres 231 et 722 sont dits 3-pp et seront affichés puisque :

231 = 3*11*7

722 = 2*19*19

Exercice n°9

Écrire un programme Python qui permet de décomposer un entier N donné (supposé $2 \leq N \leq 100$) en produit de **facteurs premiers** et d'afficher N et le produit de ses facteurs trouvés.

Ex : si n= 60 alors on affiche 60 = 2*2*3*5.

Exercice n°10

La ″**multiplication Russe**″ est une méthode particulière permettant la multiplication de deux entiers M et N en utilisant seulement la multiplication par 2, la division par 2 et l'addition.

Ex :

Pour M =19 et N = 17, le produit de M par N se fait comme suit :

M N

19 17

Le premier nombre est multiplié par 2 et le deuxième est divisé par 2 (division entière ou euclidienne): on aura :

38 8

Le processus se répète jusqu'à avoir dans la deuxième colonne **1** :

19	17
38	8
76	4
152	2
304	1

Le résultat est la somme des nombres de la première colonne qui sont en face des nombres impairs de la deuxième colonne (donc les nombres de la première colonne qui sont en face des nombres pairs de la deuxième colonne seront ignorés).

Multiplication par 2

19	17	
~~38~~	~~8~~	ignoré
~~76~~	~~4~~	ignoré
~~152~~	~~2~~	ignoré
304	1	

Division entière par 2

19*17= 19+304=323

On veut écrire un programme qui lit deux entiers M et N tels que 0 < N < M < 100, calcule et affiche le produit de deux entiers par la méthode Russe.

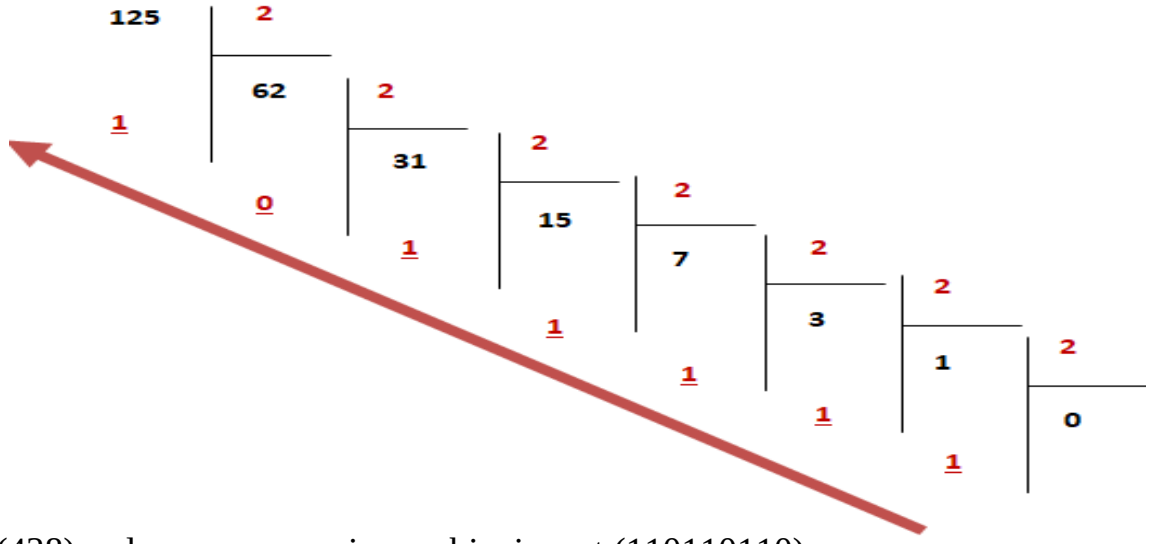
Exercice n°11

1- Écrire un programme qui permet de saisir un entier positif de 3 chiffres puis calculer et afficher sa conversion en binaire.

Ex :

Si n= $(125)_{10}$ alors sa conversion en binaire est $(1111101)_2$

NB



Si n= $(438)_{10}$ alors sa conversion en binaire est $(110110110)_2$

2- Écrire un programme qui permet de saisir un entier en binaire puis afficher sa conversion en décimale

Ex :

Si n= $(101)_2$ alors sa conversion en décimale est $(5)_{10}$

Si n= $(111000010)_2$ alors sa conversion en décimale est $(450)_{10}$

Les Tableaux (on se limite aux tableaux à une dimension)

- Ils sont appelés aussi types composés, personnalisés ou type utilisateurs
- Les tableaux (utiliser pour contenir un nombre fini et fixe d'éléments de même type)
- Les indices peuvent être de type scalaire (entier, caractère, booléen)

1.4 Déclaration en algorithme

<table>
<tr><td>Pour déclarer une variable de type tableau</td><td>En algorithme</td></tr>
<tr><td>Version 1
En le déclarant comme les autres variables d'autres types</td><td>
<table>
<tr><th>Objet</th><th>Type/Nature</th></tr>
<tr><td>Nom_tableau</td><td>Tableau de nmax de type_el</td></tr>
</table>
Nmax=nombre maximale de case
Type_elt=type des éléments du tableau
</td></tr>
<tr><td>Version 2
En utilisant nouveau type</td><td>TDNT
<table>
<tr><th>TYPE</th></tr>
<tr><td>Nom_tableau=tableau de nmaxde type_elt</td></tr>
</table>
TDO
<table>
<tr><th>Objet</th><th>Type/Nature</th></tr>
<tr><td>Nom_tableau</td><td>Tableau de nmax type_él</td></tr>
</table>
</td></tr>
</table>

1.5 Déclaration en Python

On va utiliser le type array de la bibliothèque numpy pour assurer (au contraire des listes) que les éléments du tableau soient tous de même type

Pour cela

1- On doit faire l'importation de la bibliothèqye numpy

```
from numpy import *

Ou

import numpy as np
```

2- Déclaration du tableau

a- 1ère méthode : en déclarant le tableau par les valeurs initiaux de ses éléments.

nom_tab=array([valeur_initiale]*taille) ou nom_tab = np.array([valeur_init]*taille)

nom_tab=array([val_init]*taille,"type") ou nom_tab=np.array([val_init]*taille,"type")

où Type= (int, i, float, f, d, bool, str, U, Ux)

Exemples de déclaration

Déclaration	Explication
T=array([0]*4)	Déclarer un tableau de 4 entiers et initialiser ses éléments par 0
T=array([0]*4,dtype="i")	
T=array([0]*4,dtype="int")	
T=array([0.0]*5)	Déclarer un tableau t de 5 réels et initialiser se éléments par 0.0
T=array([0.]*5,dtype="f")	
T=array([0.]*5,dtype="float")	
T=array([0.]*5,dtype="d")	
T=array([False]*5)	Déclarer un tableau de 5 booléens et initialiser ses éléments par False
T=array([False]*5,dtype="bool")	
T=array([""]*5)	Déclarer un tableau de 5 chaines vides
T=array([""]*5,dtype= "str")	
T=array([""]*5,dtype= "U")	
T=array([""]*5,dtype= "U1")	Déclarer un tableau de5 caractères vides
T=array([""]*5,dtype= "U10")	Déclarer un tableau de 5chaines, de taille max =10, et qui sont initialement vides

b- 2ème méthode nom_tableau=array([type_el]*taille)

Déclaration	Explication
T=array([int()]*4)	Déclarer un tableau de 4 entiers et initialiser ses éléments par 0
T=array([float()]*5)	Déclarer un tableau t de 5 réels et initialiser se éléments par 0.0
T=array([bool()]*5)	Déclarer un tableau de 5 booléens et initialiser ses éléments par False
T=array([str]*5)	Déclarer un tableau de 5 chaines vides
T=array([str()]*5)	Déclarer un tableau de 5 caractères vides

Attention

Pour les tableaux, on n'utilise que la déclaration avec la bibliothèque numpy
Toute utilisation d'une autre fonction de numpy n'est pas autorisée.
En python 3 si dans la définition d'un module, on utilise un paramètre de type tableau ce paramètre est passe automatiquement par @

Les Méthodes de Tri

1 Tri d'un tableau:

1.1 introduction:

Le tri consiste à mettre dans un ordre (croissant ou décroissant) un ensemble d'éléments, généralement, un tableau.

1.2 Énoncé pour tous le chapitre :

On se propose de remplir un tableau T de n éléments (1<n<20) par des entiers, le trier par ordre croissant et l'afficher.

Exemple n= 4

T	15	0	10	50	……..	

T	0	10	15	50	……..	

1.3 Algorithme du programme principal

Algorithme Tri_Tableau
Début
SAISIE(N)
REMPLIR(T ,N)
TRIER(T,N)
AFFICHER(T ,N)
FIN

On va se concentrer sur le module **TRIER** en utilisant les 3 méthodes de tri suivantes : le tri par sélection, tri à bulle et le tri par insertion

2 Tri par sélection

2.1 Principe :

1) On cherche le **pp,** l'emplacement de l'élément le plus petit du tableau en comparant son contenu avec t[1], s'ils sont différents , on les permute. donc t [1] contient la plus petite valeur.

2) Le sous tableau de t allant de 2 à n est non trié, on répète les étapes 1) et 2) et ainsi de suite jusqu'à l'avant dernier élément (n-1).

2.2 Exemple :

Exécuter le principe du tri par sélection sur le tableau suivant :

N=6 et T=

15	0	10	50	30	25

- ……
  ……

  T=

- ……
  ……

  T=

- ……
  ……

  T=

- ……
  ……

  T=

- ……
  ……

  T=

2.3 Algorithme trier et ses modules

……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……

Objet	Type/Nature

Objet	Type/Nature

3 Tri à bulle

3.1 Principe :

- N joue un double rôle (c'est la taille du tableau et elle est aussi le nombre des éléments non triés)
- On va utiliser une variable booléenne Echange (initialement echange←faux) qui repère les éventuels échanges effectués dans un parcours

Le tri à bulle

1) Consiste à parcourir le tableau en comparant les éléments adjacents deux à deux. Si deux éléments adjacents ne sont pas dans l'ordre (t[i]>t[i+1] dans le cas d'une tri croissante) on les permute. A la fin du parcours le plus grand élément doit être à la dernière case (t [n] et donc le nombre des éléments non triés dans le tableau se réduit par 1 (n←n-1)

2) On répète le parcours en prendre en considération avec n=n-1 et echange←faux jusqu'au nombre des éléments non triés égale à 1 (n=1) ou dans un parcours on n'a pas effectué aucun échange (echange= faux)

3.2 Exemple :

Exécuter le principe du tri à bulle sur le tableau suivant :

N=6 et T=

15	0	10	50	30	25

Parcours n°……

- ……
  ……

  T=

- ……
  ……

  T=

- ……
  ……

  T=

- ……
  ……

  T=

- ……
  ……

  T=

Parcours n°.....

- .....
  .....

  T= | | | | | | |

- .....
  .....

  T= | | | | | | |

- .....
  .....

  T= | | | | | | |

- .....
  .....

  T= | | | | | | |

Parcours n°.....

- .....
  .....

  T= | | | | | | |

- .....
  .....

  T= | | | | | | |

- .....
  .....

3.3 Algorithme du module tri_bulle

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

Objet	Type/Nature

Objet	Type/Nature

4 Tri par insertion

4.1 Principe :

Il consiste à :

1. Chercher la position de l'i^{ème} élément dans la partie qui le précède (de 1 a i-1) tout en considérant que cette partie est triée et cherchant à la garder triée. Si je dois changer l'emplacement de cet i^{ème} élément a un emplacement j, alors décaler à droite tous les éléments de j a i-1.

2. Insérer ensuite l'ième élément dans la case j.

Remarque :

- Cette méthode de tri nécessite l'utilisation d'une variable intermédiaire pour conserver la valeur à insérer.
- Le premier élément (t[1]) est considéré trié.

4.2 Exemple :

Exécuter le principe du tri par insertion sur le tableau suivant :

N=6 et T=

15	0	10	50	30	25

- ……
  ……

  T=

- ……
  ……

  T=

- ……
  ……

  T=

- ……
  ……

  T=

- ……
  ……

  T=

4.3 Analyse du module tri_insertion

.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....
.....

Objet	Type/Nature

Exercice n°1

Écrire un programme qui permet de saisir un numéro de carte d'identité CIN d'une personne puis afficher le plus grand nombre formé par ses chiffres.

NB. Un CIN est un nombre composé de 8 chiffres commençant soit par 0soit par 1

Ex :

Si CIN = 16382515 alors le programme affichera le plus grand nombre formé par les chiffres de ce cin est 86553211

Exercice n°2

On se propose de ranger dans un tableau V les numéros des cartes d'identité nationale des N élèves d'un lycée. ($8 \leq N \leq 25$)

Deux élèves ne peuvent pas avoir un même numéro de carte d'identité nationale. Un numéro de carte d'identité est composé obligatoirement de huit chiffres.

Écrire un programme Python qui permet de saisir les numéros de cartes d'identité de N élèves du lycée, de les afficher puis de calculer le plus grand numéro de CIN résultant de la concaténation des plus grands chiffres des CINs utilisés dans V.

Ex : si N = 9 et V= {09586725, 15263800,15263412, 51723070, 45211300, 75933012, 11022301, 12233423, 01223210}

Alors le programme affichera 99876543

Exercice n°3

Écrire un programme Python qui permet de :

- Saisir n entiers de 6 chiffres dans un tableau t avec $5 \leq n \leq 25$
- Saisir aléatoirement un entier k tel que $1 \leq k \leq 6$
- Trier le tableau T par ordre décroissant selon le $k^{ème}$ chiffre de T[i] puis par ordre décroissant selon la valeur de T[i]
- Afficher T tel que 3 éléments par ligne.

Ex : Si n= 10 et k=2 et

T= {253462, 100000, 685923, 452136, 968543, 326758, 142536, 785643, 956832, 102534}

Alors le programme doit trier T par ordre décroissant selon les $2^{èmes}$ chiffres de T[i]

Donc T devient {785643, 685923, 968543, 956832, 452136, 253462, 142536, 326758, 102534, 100000}

Puis afficher :

785643	685923	968543
956832	452136	253462
142536	326758	102534
100000		

Exercice n°4

Écrire un programme qui permet de saisie une phrase Ph décrivant les achats d'un élève puis la trier (deux mots sont séparés par un espace, entre les achats il existe une virgule, avant le dernier on trouve ″et″)

Ex : si PH = ″Sami a acheté 10 cahier, 4 livres, 12 stylos et 100 papiers dactylos.″

Devient ″Sami a acheté 4 livres, 10 cahiers, 12 stylos et 100 papiers dactylos.″

Exercice n°5

Écrire un programme qui permet de :

- Remplir un tableau T par N entiers strictement supérieurs à 1 ($10 \leq N \leq 45$).
- Trier dans l'ordre croissant les éléments premiers sûrs du tableau T suivis du reste des éléments sans tri.
- Afficher le tableau T résultant.

Ex : pour N= 10 et T :

5	25	59	23	13	47	31	100	7	107

Le programme affichera

5	7	23	47	59	107	25	13	31	100

1- Analyser le problème.
2- Établir les algorithmes des modules envisagés.

Exercice n°6

Écrire un programme Python qui permet :

- De remplir un tableau T par n entiers saisis dans un ordre croissant ($4 \leq n \leq 10$)
- De saisir un entier E et de l'insérer dans le tableau T à la bonne place de sorte que T reste Trié.
- Afficher le tableau T résultant.

Ex : pour n =7 et T=

6	8	12	14	28	37	43	

Et E=21 alors T devient

6	8	12	14	21	28	37	43

Exercice n°7

Écrire un programme qui permet de saisir dans un tableau T1 N ($6 \leq N \leq 20$), Prénoms puis remplir aléatoirement dans un tableau T2 par leurs couleurs {B pour les blancs, N pour les Noirs} puis les trier de telle façon que les noirs avant les blancs et les prénoms soient trier par ordres décroissant :

Ex : pour n =6 et

T1=

"ALI"	"SALAH"	"HICHEM"	"DONIA"	"AZIZ"	"RANIA"

T2=

B	N	N	N	N	B

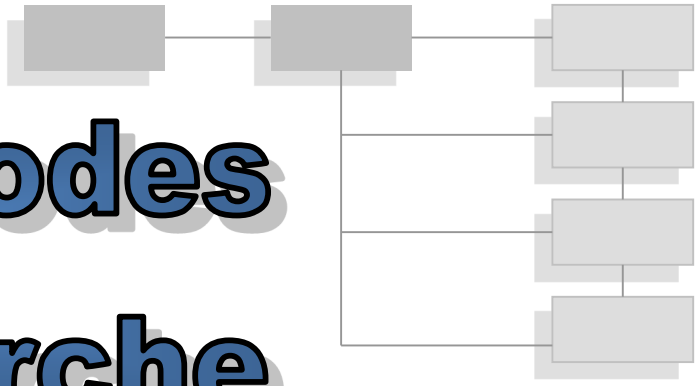
Le programme affichera

T1=

"SALAH"	"HICHEM"	"DONIA"	"AZIZ"	"RANIA"	"ALI"

T2=

N	N	N	N	B	B



Les Méthodes de Recherche

1 Méthode de recherche séquentielle :

1.1 Principe

La recherche séquentielle est un algorithme qui permet de vérifier l'existence d'un élément dans une série d'éléments. Cette méthode consiste à examiner les éléments de la liste un par un jusqu'à trouver la valeur recherchée (trouve= vrai) ou atteindre la fin du tableau (i = n)

1.2 Algorithme du module recherche_sequentielle

……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……

Objet	Type/Nature

2 Méthode de recherche Dichotomique :

2.1 Principe

La recherche dichotomique est un algorithme itératif où l'espace de recherche est limité à l'une de deux parties du tableau.

Il faut noter que cette méthode n'est appliquée que sur un tableau trié.

……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……
……

Objet	Type/Nature

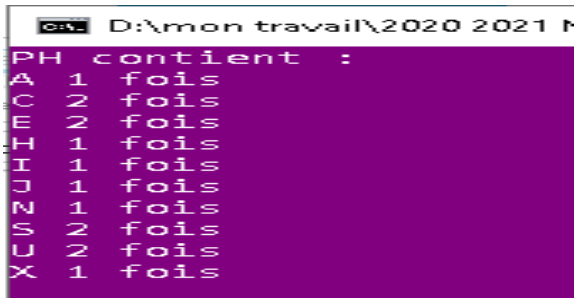
Exercice n°1

Écrire un algorithme puis son implémentation en python 3 qui permet de saisir une phrase **PH**, non vide, composée seulement par des caractères alphabétiques et des espaces (la saisie est valide si tous les mots sont séparés par une seule espace et la phrase ne doit ni commencée ni terminée par une espace). Puis afficher l'occurrence de tous les caractères alphabétiques existants dans **PH** (on ne prend pas en considération les majuscules ou les minuscules)

Exemple :

Si **PH** = "JE suis Chanceux".

Le programme affichera :



Exercice n°2

Écrire un algorithme puis son implémentation en python 3 qui permet de saisir une phrase **PH,** non vide**,** composée seulement par des caractères alphabétiques et des espaces (la saisie est valide si tous les mots sont séparés par une seule espace et la phrase ne doit ni commencée ni terminée par une espace). Puis afficher l'occurrence de chaque mot dans **PH** (on ne prend pas en considération les majuscules ou les minuscules)

Exemple :

Si **PH** = "JE suis heureux car je suis chanceux"

Le programme affichera :

```
JE se trouve 2 fois || SUIS se trouve 2 fois || HEUREUX se trouve 1 fois
|| CAR se trouve 1 fois ||CHANCEUX se trouve 1 fois
```

Exercice n°3

Écrire un programme intitulé **Cocogramme,** qui pour une chaine donnée ch de taille N avec $25 \leq N \leq 100$ (la chaine ch ne doit contenir que des caractères alphabétiques en majuscule ou des espaces et se termine par point) puis afficher si ch est **Cocogramme** ou non.

Une chaine est dite **Cocogramme** si et seulement si chaque mot de cette chaine contient le premier caractère de cette chaine.

Ex : si Ch= "**S**ami réu**S**sit **S**es exame**S**. " ch est cocogramme.

Ch = "Suède e**S**t un pay**S** **S**candinave. " n'est pas cocogramme

Exercice n°4

Un ″tautogramme″ est une chaîne dont chacun de ses mots commence par la même lettre (sans distinction entre majuscule et minuscule).

Ex : la chaîne ″Le lion lape le lait lentement″ est un ″tautogramme″.

Écrire un programme, permettant de saisir une chaîne de caractères composée uniquement de lettres et d'espaces (on suppose que deux mots consécutifs sont séparés par un seul espace), puis d'afficher un message indiquant si cette chaîne est ″tautogramme″ ou non.

Exercice n°5

Un ″totalogramme″ est une chaine dont chacun de ses mots commence et se termine par la même lettre.

Ex : la chaine ″ALLALA EMPRUNTE TEMPORAIREMENT A DAOUD SES SOULIERS″

Écrire un programme Python qui permet de saisir une chaine de caractères composée uniquement de lettres majuscules et d'espaces (on suppose que deux mots consécutifs sont séparés par un seul espace), puis d'afficher si cette chaine est totalogramme ou non.

Exercice n°6

Écrire l'algorithme d'un programme qui permet de saisir un entier **n** vérifiant la condition suivante $5 \leq n \leq 50$, remplir un tableau **T** par n chiffres puis déterminer et afficher **PLC** = la longueur de la plus longue séquence croissante d'éléments du tableau T.

Ex :

- Pour n= 13 et T=

2	4	3	1	5	7	2	8	4	8	9	0	1
1	2	3	4	5	6	7	8	9	10	11	12	13

Alors le programme affichera la plus longue croissante PLC =3

- Pour n= 12 et T=

2	4	6	7	4	5	7	8	9	0	1	0
1	2	3	4	5	6	7	8	9	10	11	12

Alors le programme affichera la plus longue croissante PLC =5

Exercice n°7

On dispose de deux tableaux T1 et T2 contenant respectivement n et m entiers positifs non nuls, avec $5 \leq n \leq 25$ et $5 \leq m \leq 25$.

On désire chercher si les éléments de T1 sont tous dans T2 ou inversement. On dira que ″T1 est inclus dans T2″ (Tous ses éléments est dans T2) ou ″ T2 est inclus dans T1 ″ (Tous ses éléments dans T1) ou ″ T1 et T2 sont non comparables ″.

Écrire l'algorithme d'un programme qui saisit m et n et remplir les deux tableaux T1 et T2 respectivement par m et n réels. Ensuite étudie la relation entre T1 et T2 comme indiqué plus haut et affiche le résultat.

Ex :- Si m=5 et n= 6 et

T1=

3	2	4	2	8	2

T2=

3	2	4	3	3

Alors le programme affichera T2 inclut dans T1

- Si m=5 et n= 6 , T1=

3	2	4	2	8	2

et T2=

1	1	1	1	1

Alors le programme affichera T1 et T2 ne sont pas comparables.

Exercice n°8

Soit un tableau T contenant N entiers naturels ***distincts***, non nuls et non triés ($2 < N < 50$). La taille du tableau doit être un multiple de 3. On veut écrire un programme qui permet de :

- Lire la taille N du tableau,
- Charger le tableau T par N entiers,
- Calculer les sommes de chaque triplet de valeurs contenues dans T et ranger le résultat dans ce même tableau, sans utiliser de tableau intermédiaire (voir schéma),
- Compléter le reste du tableau par des zéros,
- Afficher le tableau résultat.

Exemple : N=9

Données : T

1	6	9	2	7	10	4	8	3

Résultat : T

16	19	15	0	0	0	0	0	0

Exercice n°10

On se propose d'écrire un programme permettant de remplir un tableau **T** par **n** entiers strictement positifs ayant un premier chiffre impair (avec $5 \leq \mathbf{n} \leq 40$), puis on désire :

- Ranger dans un tableau **V** les éléments de T composés uniquement par des chiffres impairs
- Ranger dans un tableau **U** les autres

Enfin, le programme affichera les éléments des deux tableaux V et U.

Exemple : pour n=7 et le tableau T suivant :

⇨ T

532	1593	186	7684	9	146	59

Les tableaux V et U seront :

V

1593	9	59
1	2	3

U

532	186	7684	146
1	2	3	4

Exercice n°11

On cherche à réaliser un programme qui permet de demander à l'utilisateur d'introduire une valeur n (2<=n<=50), correspondant à la taille d'un tableau T. Puis, il permet de :

- ✓ Remplir le tableau VA par des mots composés au maximum de 10 caractères.
- ✓ Remplir un deuxième tableau V par des entiers correspondant aux nombres de caractères non alphabétiques dans chaque mot.
- ✓ Déterminer et afficher le nombre total des caractères non alphabétiques.
- ✓ Afficher le mot contenant le nombre maximal des caractères non alphabétiques.

Exemple :

Si la taille du tableau = 5

- **VA** va contenir par exemple :

A2+/ez	4tech1	bac	1234	a/+3:f

- **V** va contenir :

3	2	0	4	4

- Nombre total de caractères non alphabétiques est : 13
- Les mots contenant le nombre maximal des caractères non alphabétiques sont :

  "1234" et "a/+3:f"

Exercice n°12

On appelle « WORD CHAIN » une succession de chaines de caractères de même longueur ou chaque chaine diffère de la précédente par un seul caractère.

Soient les chaines suivantes: HEAL DEAD DEER HEAD BEER DEED.

Ainsi si on organise les chaines proposées ci-dessus dans l'ordre suivant on obtient « WORD CHAIN » :

HEAL – HEAD – DEAD – DEED – DEER – BEER.

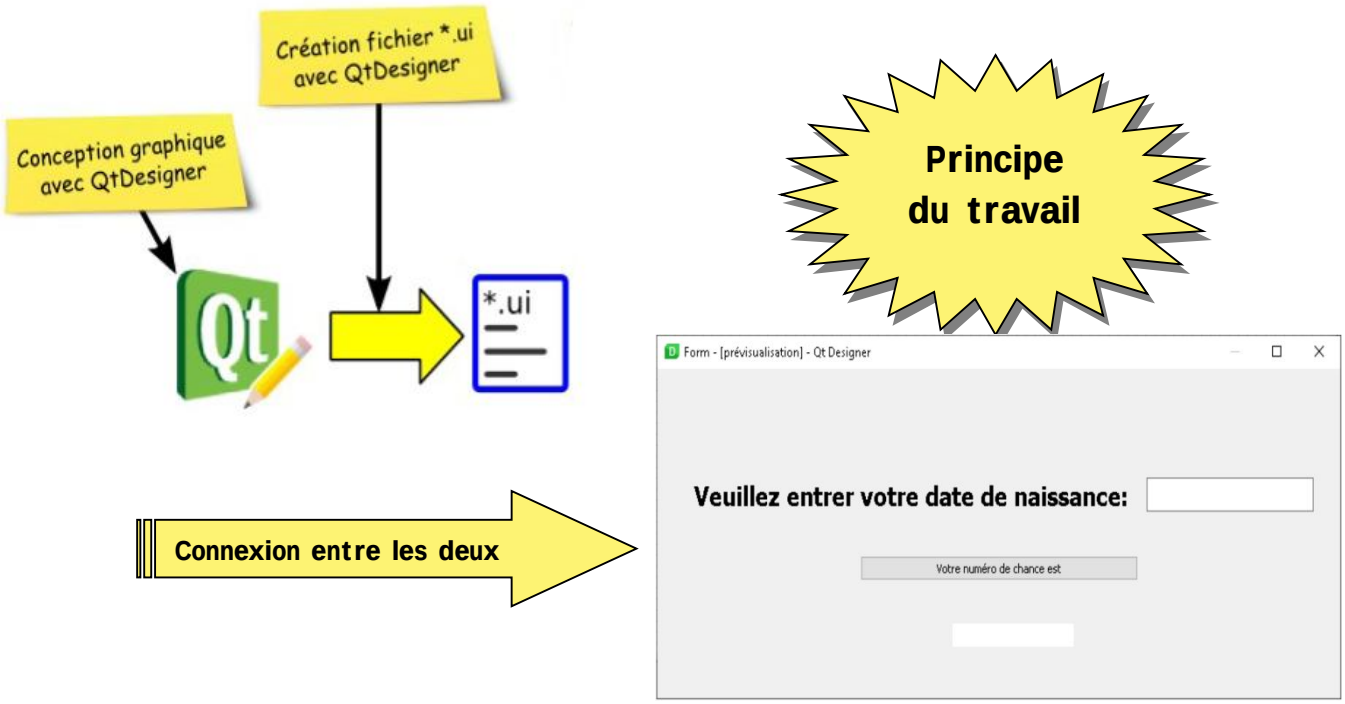
Soit T, un tableau de N chaines à saisir ($4 \leq N \leq 10$). Le tableau T doit contenir des chaines d'une même longueur M donnée ($3 \leq M \leq 6$). Les chaines de T doivent êtres distinctes et chaque chaine à saisir doit différer d'une chaine précédemment saisie (non obligatoirement la juste précédente) par un seul caractère.

On demande de vérifier si les éléments de T (comme on les a saisies) forment un « WORD CHAIN » ou non.

Interface Graphique

1 Présentation

1.1 Introduction



- Nous allons étudier la programmation des interfaces graphiques (en anglais, on parle de *GUI : Graphical User Interface*) grâce à PyQt5 qui permet d'utiliser la bibliothèque Qt version 5 avec Python.
- PyQt est la concaténation de deux mots :
  - **Python** (le langage de programmation qu'on est en train d'utiliser
  - **Qt**, initialement écrit en C++, qui est une bibliothèque, multiplateforme, de classes offrant entre autres des composants d'interface graphique appelés **widgets**.
- PyQt sert de couche de liaison entre ces deux mondes et apporte **Qt** à l'environnement **Python**.

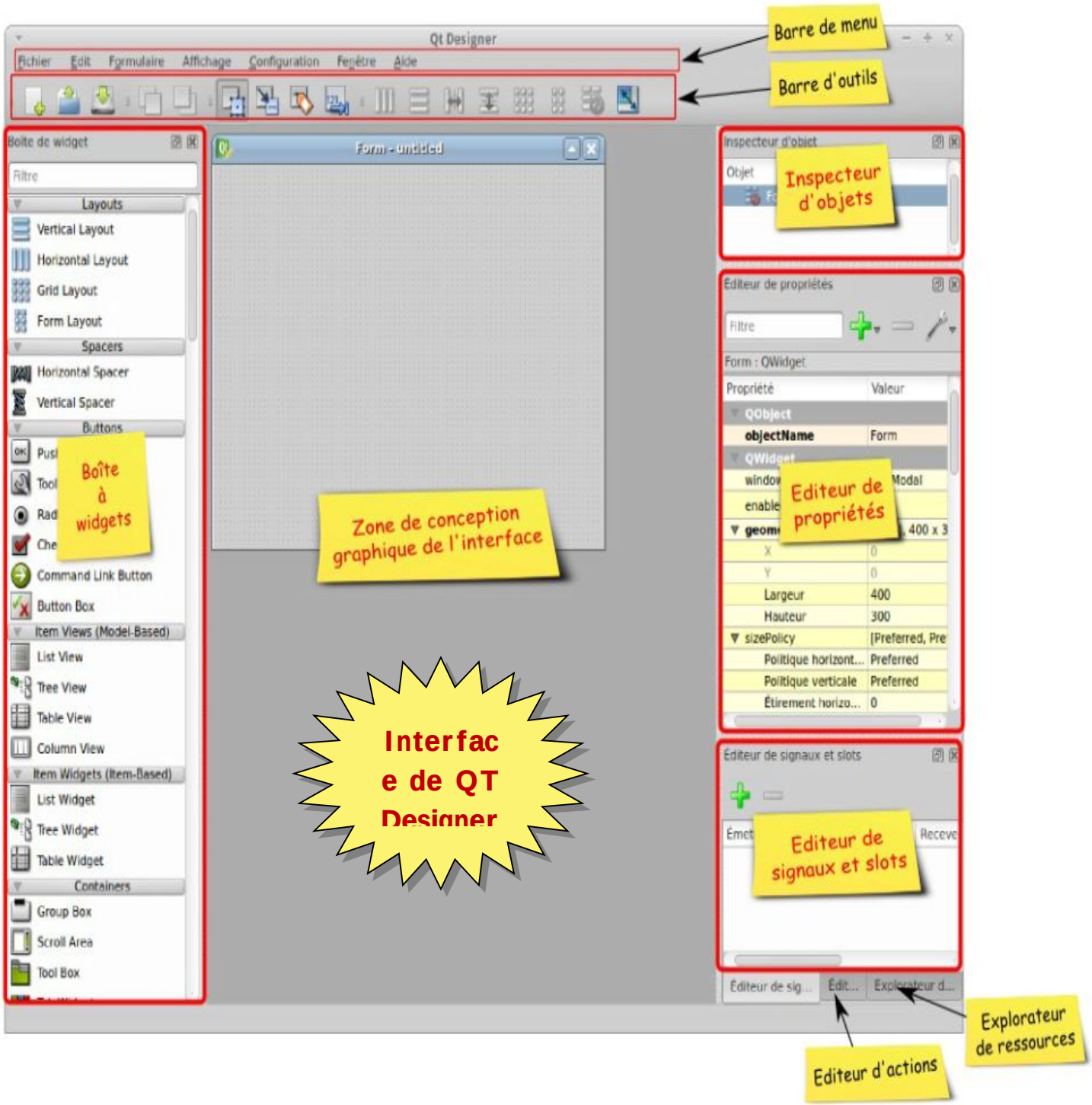
Remarques

•Alternatives à Qt :Python : Tk (intégré au langage), wxWidgets (wxPython)

• En langage Qt, un widget est un objet graphique. « Tout est widget » pourrait-on dire. Le widget correspondant à une fenêtre est le QWidget.

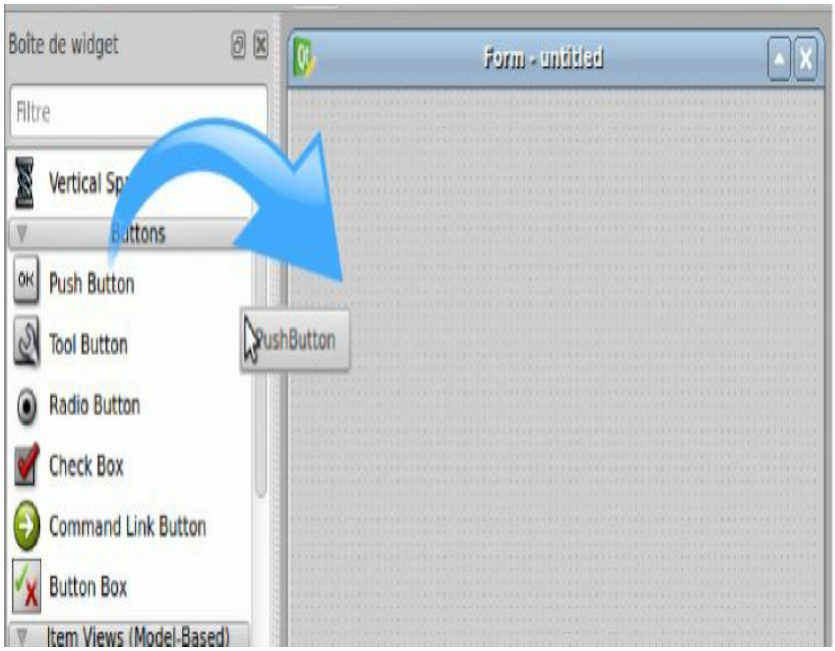
1.2 Interface de QT Désigner :

1.2.1 Fenêtre principale

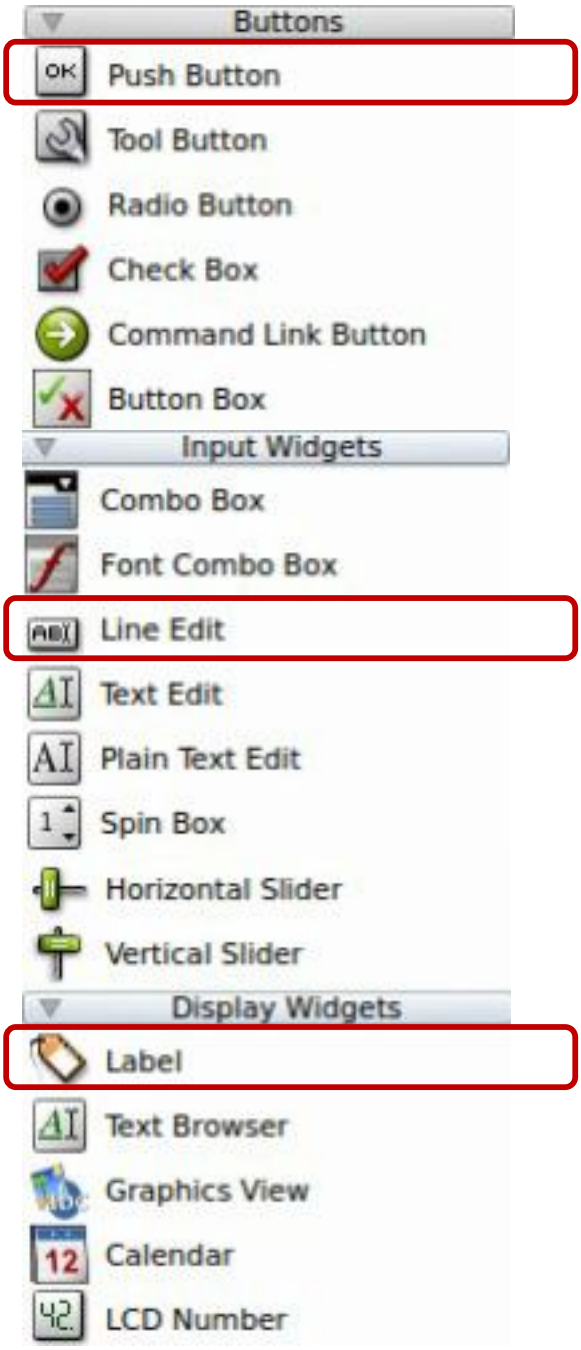


1.2.2 Boîte de widgets

Cette fenêtre contient tous les éléments graphiques disponibles pour l'édition. Le choix est important...



Les widgets sont classés par catégories. Seuls quelques widgets de base sont vraiment utiles au départ. Repérer notamment le Label (étiquette texte simple), le LineEdit (champ texte), le pushButton, (bouton)



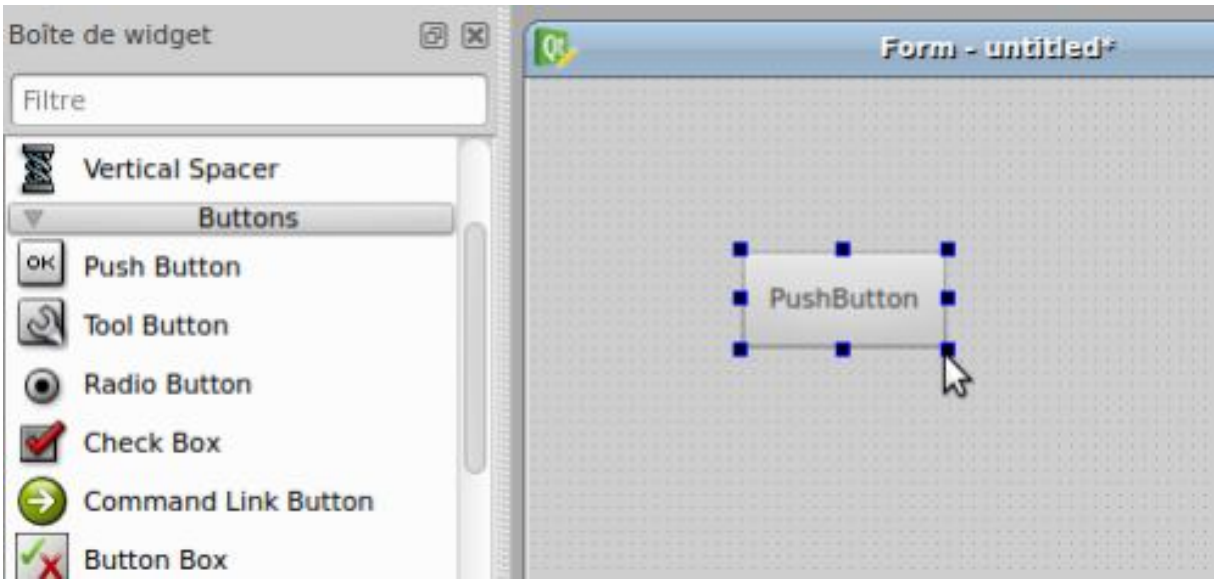
Push Button : widget qui permet d'introduire une information, généralement en utilisant le souris souvent un clic ou bien doubles clics

Line Edit : widget peut être utilisé en mode lecture et écriture mais on se limite pour l'utiliser pour afficher les résultats c'est équivalent à print

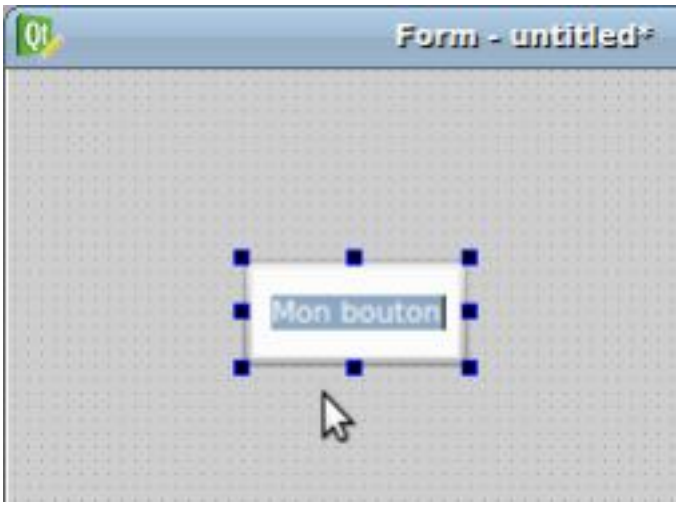
Label : widget utilisé seulement pour introduire du texte équivalent d'input

1.2.3 *Méthode de travail*

- Cliquer sur le widget voulu et garder le bouton gauche de souris enfoncé
- Tout en maintenant le bouton gauche de souris enfoncé, déplacer le curseur vers le widget « Form » (fenêtre de base) présent dans la zone d'édition
- Puis relâcher le bouton de souris : le widget a été ajouté à l'interface. Il ne reste plus qu'à effectuer les réglages de taille, à définir les propriétés.



- Ensuite, pour modifier le texte du widget, double cliquer dessus, ce qui le rend éditable. Modifier et valider :



Exercices :

En utilisant l'annexe suivant :

```
from PyQt5.uic import loadUi
from PyQt5.QtWidgets import QApplication
……
……
app = QApplication([])
windows = loadUi ("Nom_Interface.ui")
windows.show()
windows.Nom_Bouton.clicked.connect (Nom_Module)
app.exec_()
```

Où Nom_interface.ui : est l'interface créée par QT Designer

Concevoir une interface graphique permettant de :

1- Tester si un entier est pair ou non
2- Calculer le nombre de chance d'une date donné
3- Calculer le PGCD et le PPCM de deux entiers A et B
4- Retourner les code morse d'une chaine donnée
5- Tester si une chaine donnée est palindrome ou non
6- Tester si une chaine donnée est tautogramme ou non
7- Calculer le C_n^P où n et p sont deux entiers donnés.

…