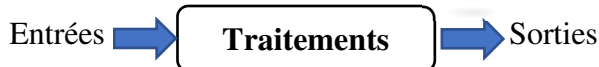


Résolution d'un problème :



ALGORITHME Nom
DEBUT

Les entrées
Traitements
Les sorties

FIN

Déclaration des objets

Objet	Type/nature

1. Les opérations simples

a. L'opération d'entrée :	
Afficher("Donner x") , Lire (x)	
x=input('donner x')	
b. L'opération de sortie	
Afficher ("Message", x, x+10)	
print('Message', x, x+10)	
c. L'opération d'affectation	
x ← 15	x=15
x ← x+1	x=x+1
z ← "Ok"	z='Ok'

2. Les types de données

Entier	int	15
Réel	float	12.3
Booléen	bool	True/False
Caractère	----	"A" / 'A'
Chaîne	str	"Ok" / 'Ok'
Tableau (array) d'entier, réel, booléen		
>>>from numpy import *		
>>> a=array([4,3,6])		
Liste (list): b=[5,'bonjour',True]		
>>>b[0];b[2]		

3. Les structures de contrôle conditionnelles

• La structure de contrôle conditionnelle simple

Initialisation
Si Condition **Alors** Traitement1
Sinon Traitement2
FinSi

```
if condition :
    Traitement1
else :
    Traitement2
```

• La structure de contrôle conditionnelle généralisée

Initialisation
Si Condition 1 **Alors** Traitement 1
Sinon Si Condition 2 **Alors** Traitement 2
Sinon
Sinon Si condition N-1 **Alors** Traitement N-1
Sinon Traitement N
FinSi

Div	//
Mod	%
x ^y	x**y
x	Abs(x)
Xor	^

opérations de base
Opérateurs: +, -, *, /, %, //
Opérateurs puissance, valeur absolue, ...: **, pow, abs, ...
Opérateurs de comparaisons :
=, !=, >, >=, <, <=, is, is not
Opérateurs bitwise :&, ^, |, <<, >>
Opérateurs logiques : or, and, not

4. Les structures de contrôle itératives

a. La structure de contrôle itérative complète :

Initialisation
Pour compteur **de** Début à Fin (pas) **Faire** Traitement
Fin Pour
En Python :

Algo	Python
x ∈ [0..5]	x in range(6)
x ∈ [2..8]	x in range(2,9)

```
for compteur in range(BI,BS,pas):
    Traitement
```

b. La structure de contrôle itérative à condition d'arrêt

• Première formulation
Initialisation
Répéter Traitement
Jusqu'à condition(s)
En Python :
while not(condition):
while True:
 Traitement
 if condition: break

• Deuxième formulation
Initialisation
Tant que Condition(s) **Faire** Traitement
Fin Tantque
En Python :
while condition:
 Traitement

Mini projet 4 :

Utiliser PyScripter pour écrire le programme1 Python suivant :

```
1 from random import *
2 x=randint(1,100)
3 while True:
4     y=input('Donner un entier (entre 1-100) = ')
5     if y==x :
6         print('Bravo')
7     elif y>x:
8         print('plus petit !')
9     else:
10        print('Plus grand :')
```



Algorithme	Python
Aléa(vi,vf)	randint(vi,vf)
Convch(10)	Str(10) → '10'
Valeur("4")	int('4') ou float('4')

- 1) Corriger le programme, puis en déduire son rôle.
- 2) Utiliser print(type(y)) pour voir le type de y.
- 3) Ajouter des commentaires (avec #) pour expliquer les lignes (2,3,4).
- 4) Ajouter l'instruction **break** permettant d'arrêter la boucle dans le cas où le nombre est trouvé.

Soit le programme2 suivant :

```
1 nb=0
2 while nb<5 :
3     nb+=1
4     print (nb)
```

Algorithme	Python
Pour c de 0 à long(ch)-1 (pas=1)faire	for c in ch :
Afficher(ch[c])	print(c)
isalnum(), isdigit(), isspace(), isupper(), islower()	
Pos(ch1,ch2)→ ch2.find(ch1) ou ch2.rfind(ch1)	

- 5) Que permet de faire le programme ci-dessus
- 6) Modifier le programme1 pour donner 10 essais seulement , et afficher le message Perdu lorsque(nbre essai=10 et x≠y)
- 7) Donner l'algorithme final du jeu proposé

Mini projet 5 :

Soit l'algorithme ci-dessous permettant de calculer le nombre de voyelles.

Algorithme Lettres

Début

Afficher("donner une chaine ch=")

Lire(ch)

Nb←0

Pour i de 0 à long(ch)-1 (pas=1) faire

Si ch[i] ∈ ["a","e","i","o","y","u"]

alors nb←nb+1 finsi

FinPour

Afficher("Le nombre de voyelles est ",

nb)

Fin

Algorithme	Python
x ∈ [1,5,6]	x in [1,5,6]
x ∈ ["a","e","i","o","y","u"]	x in 'aeioyu'



1) dresser le tableau de déclaration des objets :

Objet	Type/Nature

2)Traduire le programme en un script Python

3)Prévoir le cas des lettres minuscules et majuscules dans le test.

4) Ajouter les instructions nécessaires pour calculer le nombre des consonnes. 5)Afficher le nbre de caractères distincts(consonnes et voyelles)

Algorithme	Python
Majus(ch)	ch.upper()
Long(ch)	len(ch)
'A'+B' donne 'AB'	
3*'A' donne 'AAA'	