FICHE DE RÉVISION : CARACTÈRE & BOOLÉEN

Informatique — 2ème Année Secondaire (Types Scalaires Non Numériques)

Algorithmique **Langage Python** **Programme Officiel Tunisien**

1. Le Type Caractère (Char)

Le type **Caractère** représente un unique symbole placé entre guillemets ou apostrophes : une lettre (majuscule/minuscule), un chiffre, un symbole spécial (`+` , `@` , `#`) ou un espace.

En Algorithme

- Type : `Caractère`
- Syntaxe : `"A"`, `"z"`, `"7"`, `"@"`

En Python

- Type : `str` (de longueur 1)
- Syntaxe : `'A'` ou `"A"`

Le Code ASCII & L'Ordre des Caractères

Chaque caractère est représenté en mémoire par un numéro entier unique appelé son **Code ASCII** (allant de 0 à 255).

📌 **Codes ASCII Clés à connaître par cœur**

- Chiffres `"0"` à `"9"` : de **48** à **57** (ex: Code ASCII de `"0"` = 48)
- Lettres Majuscules `"A"` à `"Z"` : de **65** à **90** (ex: Code ASCII de `"A"` = 65)
- Lettres Minuscules `"a"` à `"z"` : de **97** à **122** (ex: Code ASCII de `"a"` = 97)
- Espace `" "` : Code ASCII = **32**

Fonctions Standards sur les Caractères

RÔLE	ALGORITHME	PYTHON	EXEMPLE & RÉSULTAT
Code ASCII d'un caractère	`ORD(c)`	`ord(c)`	`ord("A")` → `65`
Caractère d'un code ASCII	`CHR(code)`	`chr(code)`	`chr(97)` → `'a'`
Caractère Suivant	`CHR(ORD(c) + 1)`	`chr(ord(c) + 1)`	`chr(ord('A') + 1)` → `'B'`
Caractère Précédent	`CHR(ORD(c) - 1)`	`chr(ord(c) - 1)`	`chr(ord('b') - 1)` → `'a'`
Convertir en Majuscule	`MAJUS(c)`	`c.upper()`	`"a".upper()` → `'A'`
Convertir en Minuscule	`MINUS(c)`	`c.lower()`	`"B".lower()` → `'b'`

Méthodes de Test de Caractères en Python

MÉTHODE PYTHON	DESCRIPTION / TEST	EXEMPLE	RÉSULTAT
`c.isalpha()`	Vérifie si `c` est une lettre (alphabétique)	`'g'.isalpha()`	`True`
`c.isdigit()`	Vérifie si `c` est un chiffre decimal	`'9'.isdigit()`	`True`
`c.isupper()`	Vérifie si `c` est une majuscule	`'A'.isupper()`	`True`
`c.islower()`	Vérifie si `c` est une minuscule	`'k'.islower()`	`True`

2. Le Type Booléen (Boolean)

Le type **Booléen** est un type logique qui ne peut prendre que deux valeurs possibles : **VRAI** ou **FAUX**.

En Algorithme

• Type : `Booléen`
• Valeurs : `VRAI` ou `FAUX`

En Python

• Type : `bool`
• Valeurs : `True` ou `False` (Sensible à la casse !)

Opérateurs de Comparaison (Relationnels)

Ces opérateurs permettent de comparer deux valeurs et renvoient un résultat Booléen.

RELATION	ALGORITHME	PYTHON	EXEMPLE (PYTHON)	RÉSULTAT
Égalité	`=`	`==`	`5 == 5`	`True`
Différent de	`≠` ou `<>`	`!=`	`'a' != 'b'`	`True`
Strictement inférieur / supérieur	`<` / `>`	`<` / `>`	`10 < 3`	`False`
Inférieur ou égal / Supérieur ou égal	`≤` / `≥`	`<=` / `>=`	`'A' <= 'B'` (ASCII: 65 ≤ 66)	`True`

Opérateurs Logiques (Booleens) & Tables de Vérité

Les trois opérateurs logiques de base sont **NON (not)**, **ET (and)** et **OU (or)**.

A	NON A (NOT A)
VRAI / True	**FAUX / False**
FAUX / False	**VRAI / True**

A	B	A ET B (AND)	A OU B (OR)
F	F	**FAUX**	**FAUX**
F	V	**FAUX**	**VRAI**
V	F	**FAUX**	**VRAI**
V	V	**VRAI**	**VRAI**

💡 **Règle de mémoire pour ET et OU**

- **A ET B :** Est `VRAI` uniquement si **les deux conditions sont vraies en même temps**.
- **A OU B :** Est `VRAI` si **au moins une condition est vraie**.
- **Ordre de Priorité Logique :** `NON` (priorité maximale) > `ET` > `OU` (priorité minimale).

3. Pièges Classiques en Examen

⚠️ **Attention aux erreurs courantes**

1. **Confusion Caractère vs Entier :** `"5"` n'est PAS égal à `5` ! `"5"` a pour code ASCII 53.
   Exemple : `ord("5")` donne `53`, alors que l'entier `5` est une valeur numérique.
2. **Casse des mots clés en Python :** Écrire `true` ou `false` avec une minuscule produit une erreur de nom (`NameError`). Il faut écrire `True` et `False`.
3. **Confusion entre `=` et `==` :** En Python, `=` sert à l'affectation (assigner une valeur), alors que `==` sert à la comparaison d'égalité.

4. Exercices d'Application Express

Exercice 1 : Évaluation d'expressions sur les Caractères

Donner le résultat des instructions suivantes :

1. `R1 = ord("C") - ord("A")`
2. `R2 = chr(ord("a") + 3)`
3. `R3 = "4".isdigit() and "x".islower()`

Correction :

1. `R1 = 67 - 65 = 2` (Entier)
2. `R2 = chr(97 + 3) = chr(100) = 'd'` (Caractère)
3. `R3 = True and True = True` (Booléen)

Exercice 2 : Test de voyelle d'une lettre

Ecrire en algorithme la condition qui vérifie si un caractère `ch` (supposé en minuscule) est une voyelle ('a', 'e', 'i', 'o', 'u', 'y').

Solution :

En Algorithme :

```
est_voyelle ← (ch = "a") OU (ch = "e") OU (ch = "i") OU (ch = "o") OU (ch = "u") OU (ch = "y")
```

En Python :

```
est_voyelle = ch in ['a', 'e', 'i', 'o', 'u', 'y']
```