INF1 : Algorithmique et Programmation

Cours 4 :
Types et Chaines de caractères

Domitile Lourdeaux

Université de technologie de Compiègne

Printemps 2024

Définition et généralités

Définition

Un **Type** définit la nature des valeurs que peut prendre une donnée ou une variable, ainsi que les opérateurs qui peuvent lui être appliqués

Dans de nombreux langages :

- Les variables doivent être « déclarées » et associées à un type
- Par exemple en langage C, une variable x de type entier doit être déclarée de la façon suivante :
  - `int x;`

En Python :

- Il n'y a pas de déclarations
- Le type d'une variable est inféré à partir de sa valeur
- La fonction type permet de connaître le type d'une donnée ou d'une variable

Exemples

Fonction type

```
>>> message = 'bonjour'
>>> type(message)
<class 'str'>
>>> n = 43
>>> type(n)
<class 'int'>
```

Remarque

On peut changer le type d'une variable

```
>>> n = 3.5
>>> type(n)
<class 'float'>
```

Principaux types

Entier

- int (pour integer)

Réel

- float (pour virgule flotante)

Complexe

- complex

Booléen

- bool (pour boolean)

Chaines de caractères

- str (pour string)

Caractère

- str (en python, un caractère est une chaine à un caractère)

Remarque

Nous verrons d'autres types dans la suite du cours (e.g. list)

Type entier

Type int (integer)

- Python peut traiter des entiers de taille illimitée (sauf par la taille de la mémoire)
- Dans d'autres langages, les entiers sont représentés sur 4 ou 8 octets (Plus grand entier : $2^{31} - 1$ ou $2^{63} - 1$)
- Mais, en python, le temps de traitement des opérations augmente pour de très grands nombres

Type réel

Type float (virgule « flottante »)

- Un nombre est de type float s'il contient un point ou une puissance de 10
  - Exemples : 3.14159, -273.15 , $1.83e + 17 = 1.83.10^{17}$
- La taille maximale dépend de la machine sur laquelle le programme est en cours d'exécution
  - Exemples :
    - $max = 1.7976931348623157e + 308$
    - $min = 2.2250738585072014e - 308$

Type caractère : Diversité

Diversité des caractères

- Lettres majuscules
- Lettres minuscules
- Caractères de ponctuation
- Espace
- Chiffres
- Caractères non imprimables

Problème des caractères nationaux

- Caractères accentués

Type caractère : ASCII

Table ASCII (7 bits : 0-127)

Ctrl	Dec	Hex	Char	Code	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
^@	0	00		NUL	32	20	sp	64	40	@	96	60	`
^A	1	01	☺	SOH	33	21	!	65	41	A	97	61	a
^B	2	02	☻	STX	34	22	"	66	42	B	98	62	b
^C	3	03	♥	ETX	35	23	#	67	43	C	99	63	c
^D	4	04	♦	EOT	36	24	$	68	44	D	100	64	d
^E	5	05	♣	ENQ	37	25	%	69	45	E	101	65	e
^F	6	06	♠	ACK	38	26	&	70	46	F	102	66	f
^G	7	07	•	BEL	39	27	'	71	47	G	103	67	g
^H	8	08	◘	BS	40	28	(	72	48	H	104	68	h
^I	9	09	○	HT	41	29	)	73	49	I	105	69	i
^J	10	0A	◙	LF	42	2A	*	74	4A	J	106	6A	j
^K	11	0B	♂	VT	43	2B	+	75	4B	K	107	6B	k
^L	12	0C	♀	FF	44	2C	,	76	4C	L	108	6C	l
^M	13	0D	♪	CR	45	2D	–	77	4D	M	109	6D	m
^N	14	0E	♫	SO	46	2E	.	78	4E	N	110	6E	n
^O	15	0F	☼	SI	47	2F	/	79	4F	O	111	6F	o
^P	16	10	►	SLE	48	30	0	80	50	P	112	70	p
^Q	17	11	◄	CS1	49	31	1	81	51	Q	113	71	q
^R	18	12	↕	DC2	50	32	2	82	52	R	114	72	r
^S	19	13	‼	DC3	51	33	3	83	53	S	115	73	s
^T	20	14	¶	DC4	52	34	4	84	54	T	116	74	t
^U	21	15	§	NAK	53	35	5	85	55	U	117	75	u
^V	22	16	▬	SYN	54	36	6	86	56	V	118	76	v
^W	23	17	↨	ETB	55	37	7	87	57	W	119	77	w
^X	24	18	↑	CAN	56	38	8	88	58	X	120	78	x
^Y	25	19	↓	EM	57	39	9	89	59	Y	121	79	y
^Z	26	1A	→	SUB	58	3A	:	90	5A	Z	122	7A	z
^[	27	1B	←	ESC	59	3B	;	91	5B	[	123	7B	{
^\	28	1C	∟	FS	60	3C	<	92	5C	\	124	7C	|
^]	29	1D	↔	GS	61	3D	=	93	5D	]	125	7D	}
^^	30	1E	▲	RS	62	3E	>	94	5E	^	126	7E	~
^_	31	1F	▼	US	63	3F	?	95	5F	_	127	7F	⌂

ASCII étendu (8 bits)

Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char	Dec	Hex	Char
128	80	Ç	160	A0	á	192	C0	└	224	E0	α
129	81	ü	161	A1	í	193	C1	┴	225	E1	ß
130	82	é	162	A2	ó	194	C2	┬	226	E2	Γ
131	83	â	163	A3	ú	195	C3	├	227	E3	π
132	84	ä	164	A4	ñ	196	C4	─	228	E4	Σ
133	85	à	165	A5	Ñ	197	C5	┼	229	E5	σ
134	86	å	166	A6	ª	198	C6	╞	230	E6	µ
135	87	ç	167	A7	º	199	C7	╟	231	E7	τ
136	88	ê	168	A8	¿	200	C8	╚	232	E8	Φ
137	89	ë	169	A9	⌐	201	C9	╔	233	E9	Θ
138	8A	è	170	AA	¬	202	CA	╩	234	EA	Ω
139	8B	ï	171	AB	½	203	CB	╦	235	EB	δ
140	8C	î	172	AC	¼	204	CC	╠	236	EC	∞
141	8D	ì	173	AD	¡	205	CD	═	237	ED	φ
142	8E	Ä	174	AE	«	206	CE	╬	238	EE	ε
143	8F	Å	175	AF	»	207	CF	╧	239	EF	∩
144	90	É	176	B0	░	208	D0	╨	240	F0	≡
145	91	æ	177	B1	▒	209	D1	╤	241	F1	±
146	92	Æ	178	B2	▓	210	D2	╥	242	F2	≥
147	93	ô	179	B3	│	211	D3	╙	243	F3	≤
148	94	ö	180	B4	┤	212	D4	╘	244	F4	⌠
149	95	ò	181	B5	╡	213	D5	╒	245	F5	⌡
150	96	û	182	B6	╢	214	D6	╓	246	F6	÷
151	97	ù	183	B7	╖	215	D7	╫	247	F7	≈
152	98	ÿ	184	B8	╕	216	D8	╪	248	F8	°
153	99	Ö	185	B9	╣	217	D9	┘	249	F9	∙
154	9A	Ü	186	BA	║	218	DA	┌	250	FA	·
155	9B	¢	187	BB	╗	219	DB	█	251	FB	√
156	9C	£	188	BC	╝	220	DC	▄	252	FC	ⁿ
157	9D	¥	189	BD	╜	221	DD	▌	253	FD	²
158	9E	₧	190	BE	╛	222	DE	▐	254	FE	■
159	9F	ƒ	191	BF	┐	223	DF	▀	255	FF	

Type caractère : Conversion entre caractère et code

Fonction ord

- ord(caractère) renvoie un entier correspondant au code ASCII (et plus généralement UTF-8)
  - Exemples :
    - ord('A') vaut 65
    - ord('a') vaut 97
    - ord ('€') vaut 8364

Fonction chr

- chr (entier) renvoie le caractère correspondant au code UTF-8 de l'entier
  - Exemples :
    - chr(65) vaut 'A'
    - chr(8364) vaut '€'

Type caractère : Conversion Minuscule - Majuscule

Soit c une variable de type string, si sa valeur correspond à une lettre minuscule, comment la convertir en majuscule ?

```
c = input("saisir un caractere : ")
if (c >= 'a') and (c <= 'z'):
        c = chr(ord(c) - ord('a') + ord('A'))
print(c)

if c in range (ord('a'), ord('z')+1):
        c = chr(ord(c) - ord('a') + ord('A'))
print(c)
```

Remarque

En Python, la « méthode » upper() de la classe string permet la conversion :

- c = c.upper()

Type Chaine de caractères : Définition

Définition

- Une chaîne de caractères est une suite de caractères regroupés dans une même variable
- En python, le type est str (pour string)

Remarque

- Il s'agit d'un type dit « composite »
- On parle aussi de « séquence »
- Nous verrons plus tard qu'il existe d'autres types de séquences

Type Chaine de caractères : formatage (1)

Deux possibilités pour définir une chaine de caractères :

- Soit des simples quotes : 'Une chaîne de caractères'
- Soit des doubles quotes : "Une chaîne de caractères"
- Mais pas les deux !

Si la chaîne contient une apostrophe, on utilise les doubles quotes :

- "L'étudiant s'attend à être reçu"

Si la chaîne contient des guillemets, on utilise les simples quotes :

- 'Il dit : "Je pense être reçu" '

On peut aussi banaliser un caractère à l'aide d'un anti-slash

- "L'étudiant dit \"Je m'attends à être reçu \""

Chaine vide : '' ou ""

Type Chaine de caractères : formatage (1)

Ajout d'un saut à la ligne

- print("Anna \nBob \nToto")

Caractères spéciaux

```
>>> print("I \u2764 Toto")
I ❤ Toto
>>> print("I \u2764 : \n\tAnna \n\tBob \n\tToto ")
I ❤ :
    Anna
    Bob
    Toto
```

Type Chaine de caractères : Accès à un caractère

On peut accéder à chaque caractère au moyen d'un index (commençant à 0)

- Exemple
  - s = 'INF1' # s[0] vaut I, s[1] vaut N

Attention

Les chaines sont « immuables »

On ne peut donc pas modifier un caractère directement :

- Exemple
  - s[3]= '2' provoquerait une erreur
    TypeError : 'str' object does not support item assignment

Type Chaine de caractères : Extraction d'une sous-chaîne (slicing)

		chaine ='Université'
• chaine[n :m]	# extrait la sous-chaine de chaine de l'index n à l'index m (m non compris)	*chaine*[3 : 7] 'vers'
• chaine[:m]	# extrait la sous-chaine de chaine de l'index 0 à l'index m (m non compris)	*chaine*[: 7] 'Univers'
• chaine[n :]	# extrait la sous-chaine de chaine de l'index n à la fin de chaine	*chaine*[6 :] 'sité'
• chaine[n :-m]	# extrait la sous-chaine de chaine de l'index n à la fin de chaine -m	*chaine*[2 : −2] 'iversi'

Type Chaine de caractères : Longueur d'une chaine

Fonction len(...) :

- Revnoie un entier représentant la longueure de la chaîne

Exemples

```
len('INF1')
>>> 4

s = 'INF1'
len(s)
>>> 4
```

Type Chaine de caractères : Comparaison de chaines

Opérateurs

- $==, >, >=, <, <=, !=$

Ordre

- Lexicographique

Exemples

$s1 =$"Dupont"

$s2 =$"Dupond"

$s3 =$"Du pont"

$s4 =$"Du Pont"

```
>>> s2 < s1
True
>>> s3 < s1
True
>>> s4 < s3
True
>>> s4 < s3 < s2 < s4
True
```

Type Chaine de caractères : Concaténation

Définition

La concaténation est une opération qui permet d'accoler 2 ou plusieurs chaines de caractères

Syntaxe Python

- $s = s + s2 + s3...$

Exemples

$s1 =$"Du"

$s2 =$"pont"

```
>>> s = s1 + s2
>>> s
"Dupont"

>>> s = s1 + " " + s2
>>> s
"Du pont"
```

Type Chaine de caractères : Appartnance ou inclusion

- On peut tester l'appartenance d'un caractère ou l'inclusion d'une sous-chaîne dans une chaîne à l'aide de in

Exemples

```
>>> s ="Université"
>>> "v" in s
True

>>> "u" in s
False

>>> "Univers" in s
True
```

Type Chaine de caractères : Quelques méthodes de la classe string

En Python tout est « objet »

- Un objet regroupe des propriétés et des méthodes

Une chaine de caractère est donc un objet

- Quelques de méthodes de la classe string
  - upper(), lower(), replace(old, new), split(), etc.

```
>>> print("bonjour").upper()
"BONJOUR"

>>> "toto".replace("o","i")
"titi"

"Mon UV preferee est INF1".split()
["Mon", "UV", "preferee", "est", "INF1"]
```

Exemple 1

- Ecrire un programme permettant de remplacer les espaces par un '+' dans une chaine

```
chaine = "Mon UV preferee est INF1"
chaine_plus = ""
for c in chaine :
    if c == " " :
        chaine_plus = chaine_plus + "+"
    else :
        chaine_plus = chaine_plus + c

print(f"Chaine modifiee : {chaine_plus}")
```

"Chaine modifiee : Mon+UV+preferee+est+INF1"

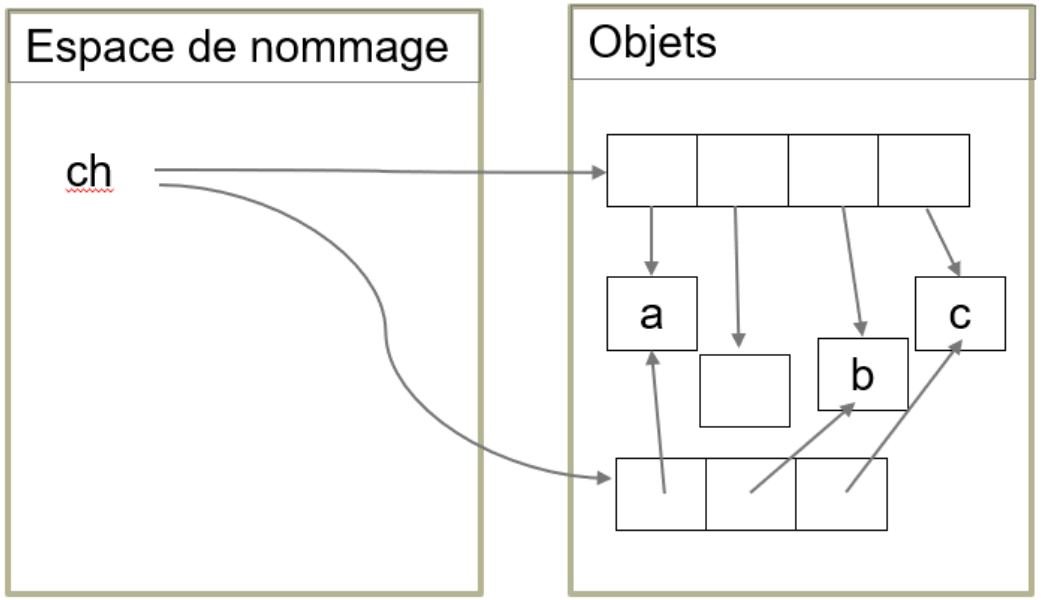
Exemple 2

- Ecrire un programme qui enlève les espaces d'une chaine de caractères saisie par l'utilisateur

```
chaine = input("Entrer une chaine de caracteres : ")
chaine_ssespace1 = ""
for c in chaine :
    if c != " " :
        chaine_ssespace1 += c
print(f"Chaine modifiee : {chaine_ssespace1}")
# Autre version
chaine_ssespace2 = ""
for i in range(len(chaine)) :
    if chaine[i] != " " :
        chaine_ssespace2 += chaine[i]
print(f"Chaine modifiee : {chaine_ssespace2}")
```

En mémoire

```
>>> ch ="a bc"
>>> ch = ch[: 1] + ch[2 : 4]
```



Remarque

ch référence un nouvel objet constitué à partir de "tranches" de la chaine initiale

Exemple 4 (1)

Inverser une chaine de caractères

- En parcourant la chaine à l'envers
- En parcourant la chaine à l'endroit
- En utilisant le slicing

Exemple 4 (2)

Inverser une chaine de caractères

- En parcourant la chaine à l'envers

```
# Inversion d'une chaine de caractere
# Methode 1 : partir de la fin
s = "Bonjour"
s_inv = ""
for i in range(len(s) - 1, -1, -1):
    s_inv = s_inv + s[i]
print(s_inv)
```

Exemple 4 (3)

Inverser une chaine de caractères

- En parcourant la chaine à l'endroit

```
# Inversion d'une chaine de caractere
# Methode 2 : partir du debut
s = "Bonjour"
s_inv = ""
for c in s :
    s_inv = c + s_inv
    print(s_inv)
```

Exemple 4 (4)

Inverser une chaine de caractères avec le slicing

Exemple 4 (5)

Inverser une chaine de caractères avec le slicing

```
# Method 3: slicing
s = "Bonjour"
s_inv = s[3:0:-1]
print('1 : ',s_inv)
s_inv = s[7:0:-1]
print('2 : ',s_inv)
s_inv = s[len(s):0:-1]
print('3 : ',s_inv)
s_inv = s[-1:0:-1] # -1 de la fin
print('4 : ',s_inv)
s_inv = s[-1:-1:-1]
print('5 : ',s_inv)
s_inv = s[::-1]
print('6 : ',s_inv)
```

```
Version 3----
1 :  jno
2 :  ruojno
3 :  ruojno
4 :  ruojno
5 :
6 :  ruojnoB
```

Traitement des erreurs : Problème

Certaines erreurs peuvent se produire à l'exécution

```
>>> n = int(input('Saisir un entier : '))
Saisir un entier : >? 4.7
Traceback (most recent call last):
  File "<input>", line 1, in <module>
ValueError: invalid literal for int() with base 10: '4.7'
```

Comment éviter cela ?

Traitement des erreurs : Récupération des erreurs (1)

- On peut anticiper certaines erreurs. On parle alors de « gestion des exceptions »
- En python, on dispose des instructions try ... except ... else

```
# Affichage de l'erreur
try :
    n = int(input("Saisir un entier : "))
except :
    print("Attention, erreur de saisie")

try :
    n = float(input("Saisir un float : "))
except :
    print("Attention, erreur de saisie")
# si on saisie 3 transforme en 3.0
```

Traitement des erreurs : Récupération des erreurs (2)

```
# Pour repeter la saisie
b = False
while not b :
    try :
        n = int(input("Saisir un entier : "))
    except :
        print("Attention, erreur de saisie")
    else :
        b = True
```

Questions...