Curs 3 - (str vs list)

1. Clasa String (str)

- Obiectele de tip string se declară folosind operatorul “” (ghilimele) sau ‘’ (apostrof). Practic nu exista nicio diferență între cele două moduri de declarare. Exemplul 3.1.1
- Funcția ***len(s)*** returnează lungimea stringului ***s***. Exemplul 3.1.2
- **Observație: Nu există tipul char**. Caracterele sunt pur și simplu stringuri de lungime 1.
- Valorile de tip string sunt **IMMUTABLE.** Altfel spus, ca o variabilă string să își schimbe valoarea, se crează un nou obiect (cu un nou ID) iar variabila va referenția acel nou obiect creat. Exemplul 3.1.3
- Stringurile sunt **ITERABILE** - pot fi parcurse element cu element. În acest caz, fiecare element este un caracter (string de lungime 1). Exemplul 3.1.4
- Operatorul **[ ]** poate fi folosit pentru a **accesa un anumit caracter** sau o anumită felie (slice) din string. Asemenea vectorilor, **indexarea se face de la 0**. Spre deosebire de vectori, operatorul **[ ] nu poate fi folosit în atribuiri.** Exemplul 3.1.5
- Verificarea dacă două șiruri sunt egale ca valoare (case sensitive) se poate face cu operatorul **==**. Exemplul 3.1.6

2. Metode (funcții) ale clasei str

- **Observație** - aceste funcții întorc un nou string pe baza obiectului de tip string apelant, **NU MODIFICĂ** valoarea obiectului apelant.

Nume metodă	**Tip valoare intoarsa de către metoda**	**Explicații**
upper()	str	Întoarce un string pe baza celui apelant care are toate caracterele majuscule
lower()	str	Întoarce un string pe baza celui apelant care are toate caracterele litere mici
title()	str	Întoarce un string pe baza celui apelant în care prima literă a fiecărui cuvânt este majusculă. Exemplul 3.2.1
isupper(), islower(), istitle()	bool	Întoarce True sau False în funcție dacă stringul apelant este format doar din majuscule, doar din litere mici, sau este în format “title”
isnumeric()	bool	Întoarce True sau False în funcție dacă string-ul apelant este alcătuit doar din cifre. Exemplul 3.2.2
startswith(pref)	bool	Întoarce True sau False în funcție dacă stringul *pref*, primit ca parametru, este sau nu prefix pentru string-ul apelant. Exemplul 3.2.3

endswith(suf)	bool	Întoarce True sau False în funcție dacă stringul *suf*, primit ca parametru, este sau nu sufix pentru string-ul apelant.
split(sep=None, n=-1)	list	”Taie” stringul apelant în funcție de stringul primit ca separator. Al doilea parametru reprezinta numarul maxim de taieturi (se numără de la stanga la dreapta). Stringurile rezultate sunt retinute în lista întoarsă de funcție. Exemplul 3.2.4
find(st,[s,[t]])	int	Întoarce indicele primei apariții stringului st în stringul apelant. Dacă st nu apare, va întoarce -1. Opțional se mai pot adăuga parametrii întregi s, t care reprezintă intervalul indicilor pe care se face căutarea.
replace(old,new,[maxreplace])	str	Întoarce un string obținut prin înlocuirea tuturor aparițiilor lui *old* cu valoarea lui *new* în stringul apelant. Opțional se poate introduce al 3-lea parametru: numărul maxim de substituții permise. Valoarea implicită pentru al 3-lea parametru este -1.

O listă exhaustivă a tuturor metodelor membre ale clasei str o găsiți aici.
Exercițiu: Se citesc mai multe cuvinte separate prin spațiu. Să se afișeze suma acelor cuvinte care pot fi interpretate ca numere întregi. Cuvintele care nu pot fi convertite la numere întregi vor fi ignorate. Exemplu: pentru intrarea "23 15 14a -7 -10b 9 -1 b -", suma ar fi 23+15+(-7)+9+(-1)=**39**.
Rezolvare

3. Clasa list

- În Python listele au proprietăți asemănătoare cu listele dar și vectorii din C++
- Folosind **[ ]** putem accesa un element de pe o anumită poziție a listei. Putem atribui valori unui element aflat la un anumit index (spre deosebire de ce am văzut la clasa str). Indexarea se face de la 0.
- O listă este declarată folosind **[ ]**. Lungimea (numărul de elemente) listei poate fi aflat folosind funcția **len().** Exemplul 3.3.1
- Listele sunt **ITERABILE**. Pot fi parcurse cu ajutorul unei instrucțiuni de tip *foreach*. Exemplul 3.3.2
- Listele sunt **MUTABLE.** Altfel spus, o listă, **când își schimbă valoarea** (se adaugă, se șterg, sau se modifică elemente ale listei), nu îți schimbă id-ul, **NU SE CREAZĂ UN OBIECT NOU.** Consecințe: Exemplul 3.3.3
- Putem folosi operatorul **[ ]** pentru a accesa doar o “felie” din listă, cuprinsă între 2 indici. Exemplul 3.3.4
- Putem înlocui o felie din listă cu o altă listă. Exemplul 3.3.5

4. Metode

//To Be Continued in *Cursul 4*