**LABORATORUL 2**

1. Se citește un șir de caractere și un număr natural k. Să se șteargă din s caracterul de pe poziția k (pozițiile numerotate de la 0) și să se afișeze șirul nou obținut.

s=input()  
k=int(input())  
sir\_nou=s[:k]+s[k+1:] *#punem intai prima bucatica de sir(k este exclusiv), iar apoi a doua bucatica*print(sir\_nou)

sau:

s=input()  
k=int(input())  
sir\_nou=**" "** *#sirul vid***for** i **in** range(len(s)):  
 **if** i!=k:  
 sir\_nou=sir\_nou+s[i]  
print(sir\_nou)

sau:

s=input()  
k=int(input())  
sir\_nou=[] *#sirul vid***for** i **in** range(len(s)):  
 **if** i!=k:  
 sir\_nou.append(s[i])  
sir\_nou=**""**.join(sir\_nou)  
print(sir\_nou)

2. Se citește un cuvânt. Să se șteargă din cuvânt toate aparițiile primei litere. Se va afișa un mesaj de forma: După ștergerea literei 'X' șirul obținut este "S" de lungime L folosind diferite tipuri de formatare (cu parametri poziționali și f-stringuri).

cuv=input()  
litera=cuv[0]  
nou=cuv.replace(litera, **""**)  
print(**"Dupa stergerea literei "**, litera, **"sirul obtinut este "**, nou, **"de lungime "**, len(nou), sep=**" "**)

sau:

print(**"Dupa stergerea literei {} sirul obtinut este \"{}\" de lungime {}"**.format(litera, nou, len(nou)))

sau:

print(**f"Dupa stergerea literei {**litera**} sirul obtinut este \"{**nou**}\" de lungime {**len(nou)**}"**)

3. Se citește un cuvânt s de cel mult 10 de caractere. Sa se afișeze (folosind s[i:j]) pe câte o linie cuvintele obținute succesiv din s tăind prima și ultima literă (**afișate centrat pe 10 de caractere)**:

algoritm afara

lgorit far

gori a

or

s=input()  
print(s.center(10))  
**for** i **in** range (1, (len(s)+1)//2):  
 print(s[i:len(s)-i].center(10))

4.**Sir periodic**. Se citește un șir de caractere 𝑠. Să se verifice dacă există un șir t, diferit de 𝑠, astfel încât 𝑠 să se poată obține prin concatenarea de un număr arbitrar de ori (𝑘>1) a șirului t (adică să se verifice dacă șirul s este periodic). Dacă există mai multe astfel de șiruri t se va determina cel mai lung. Exemplu: s = "abbaabbaabbaabba" => t= abbaabba

s=input()  
n=len(s)  
**for** i **in** range(n//2, 1, -1): *#lungimea perioadei* **if** n%i==0:  
 t=s[:i]  
 **if** t\*(n//i)==s:  
 print(t)  
 **break**

5. **Rime**. Se citește un cuvânt w, un număr natural nenul 𝑝 și un sir de cuvinte separate prin spațiu sau virgulă. Să se afișeze toate cuvintele care sunt 𝑝-rime cu 𝑤, adică ultimele 𝑝 caractere coincid.

w=input()  
p=int(input())  
s=input()  
s=s.replace(**","**, **" "**) *#inlocuim virgulele cu spatii, pentru a putea face split doar dupa spatii*s=s.split(**" "**)  
**for** cuvant **in** s:  
 **if** cuvant[-p:]==w[-p:]: *#punem minus in fata la p pentru a ajunge si la ultimul cuvant care e p-rima cu w* print(cuvant)

6. Într-o propoziție a fost efectuată, posibil de mai multe ori, aceeași greșeală de ortografie.

a) Scrieți un program care citește propoziția, șirul greșit și șirul corect, după care afișează propoziția corectă. De exemplu, în propoziția "Problemele cu șiruri de caracteger nu sunt ggerle!"greșeală constă în faptul că în loc de șirul “re” a fost scris șirul “ger”.

sir=input()  
sir\_corect=input()  
sir\_gresit=input()  
**sir=sir.replace(sir\_gresit, sir\_corect)**print(sir)

b) Modificați programul astfel încât să corecteze maxim 10 astfel de greșeli, iar dacă sunt mai multe să afișeze mesajul: “textul contine prea multe greseli, doar 10 au fost corectate”

sir\_initial=input()  
sir\_corect=input()  
sir\_gresit=input()  
sir\_initial=sir\_initial.replace(sir\_gresit, sir\_corect, 10)  
sir\_nou=sir\_initial.replace(sir\_gresit, sir\_corect)  
**if** sir\_nou==sir\_initial:  
 print(sir\_initial)  
**else**:  
 print(**"textul contine prea multe greseli, doar 10 au fost corectate"**)

7. Scrieți un program care citește de la tastatură un șir de caractere format din mai multe cuvinte, separate prin unul sau mai multe spații și îl modifică astfel încât fiecare cuvânt să înceapă cu literă mare.

s=input()  
aux=s.split(**" "**) *#spargem sirul in cuvinte*s=aux[0][0].upper()+aux[0][1:] *#transformam primul cuvant din isr***for** i **in** range (1, len(aux)):  
 **if** aux[i]!=**" "**: *#verificam sa fim la inceput de cuvant* s=s+**" "**+aux[i][0].upper()+aux[i][1:]  
 **else**:  
 s=s+**" "**print (s)

8. Se citește un text codificat după regula: În fata fiecărui caracter este scris un număr de cel mult 2 cifre care reprezintă numărul de apariții consecutive ale acestuia. Scrieți un program care decodifica textul: 1G10o4l=>Goooooooooollll

s=input()  
nr=0  
**for** i **in** s:  
 **if** i.isdigit():  
 nr=nr\*10+int(i)  
 **else**:  
 print(i\*nr, end=**''**)  
 nr=0

9. Scrieți un program care să înlocuiască într-o propoziție toate aparițiile unui cuvânt 𝑠 cu un cuvânt 𝑡 (cuvintele sunt separate în propoziție printr-un spațiu).

prop=input()  
s=input()  
t=input()  
prop=prop.replace(s,t)  
print(prop)

10. Jurnalul electronic al Anei conține, în fiecare zi, câte o frază cu informații despre cheltuielile pe care ea le-a efectuat în ziua respectivă. Scrieți un program care să citească o frază de acest tip din jurnalul Anei și apoi să afișeze suma totală cheltuită de ea în ziua respectivă. De exemplu, pentru fraza “Astăzi am cumpărat pâine de 5 RON, pe lapte am dat 10 RON, iar de 15 RON am cumpărat niște cașcaval. De asemenea, mi-am cumpărat și niște papuci cu 50 RON!”, programul trebuie să afișeze suma totală de 80 RON. Fraza se consideră corectă, adică toate numerele care apar în ea sunt numere naturale reprezentând sume cheltuite de Ana în ziua respectivă.

fraza=input()  
suma=0  
aux=fraza.split(**" "**)  
**for** i **in** aux:  
 **if** i.isdigit():  
 suma=suma+int(i)  
print(suma)

11. Scrieți un program care citește un șir de caractere și decide dacă acesta este un nume corect al unei persoane. Se consideră că un nume este corect dacă respectă următoarele proprietăți:

-Orice nume sau prenume conține doar litere și cel mult o cratimă.

-Orice nume sau prenume este format din cel puțin 3 litere.

-Orice nume sau prenume începe cu literă mare.

-Persoana poate avea cel mult doua prenume, iar dacă sunt două atunci sunt despărțite printr-o cratimă (‘-’). La fel și în cazul numelui.

full\_name = input()  
names = full\_name.split(**" "**)  
nr=len(names)  
ok = **True  
for** name **in** names:  
 **if** len(name.split(**"-"**)) > 2:  
 ok = **False  
 break** single\_names = name.split(**"-"**)  
 **for** single\_name **in** single\_names:  
 *# check if name contains only letters and '-'* **for** letter **in** single\_name:  
 **if** (**not** letter.isalpha() **and** letter != **'-'**) :  
 ok = **False  
 break  
 if** len(single\_name) < 3:  
 ok = **False  
 break  
 if not** single\_name[0].isupper():  
 ok = **False  
 break  
if** ok **and** nr<=4:  
 print(**"corect"**)  
**else**:  
 print(**"gresit"**)