Programarea Algoritmilor
– LABORATOR NR. 2 –

Șiruri de caractere

1. Într-o propoziție/frază a fost efectuată, posibil de mai multe ori, aceeași greșeală de ortografie. Scrieți un program care citește propoziția, șirul greșit și șirul corect, după care afișează propoziția corectă. De exemplu, în propoziția "Problemele cu șiruri de caractedf nu sunt gdfle!" greșeală constă în faptul că în loc de șirul "re" a fost scris șirul "df".

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

3. Scrieți un program care să înlocuiască într-o propoziție toate aparițiile unui cuvânt s cu un cuvânt t. Atenție, NU se poate utiliza metoda *replace*! De ce?

4. Scrieți un program care să se verifice dacă două șiruri de caractere sunt anagrame sau nu. Două șiruri sunt anagrame dacă unul se poate obține din celălalt printr-o permutare a caracterelor sale. De exemplu, șirurile *emerit* și *treime* sunt anagrame, dar șirurile *emerit* și *treimi* nu sunt! **Indicație de rezolvare (fără structuri de date auxiliare și fără sortare):** Se caută, pe rând, fiecare caracter din primul șir în cel de-al doilea. În cazul în care caracterul nu este găsit înseamnă că șirurile nu sunt anagrame, altfel se șterge caracterul din cel de-al doilea șir și se trece la următorul caracter din primul șir. Atenție, folosind această metodă, cel de-al doilea șir va fi modificat!

5. O metodă simplă (dar nesigură!!!) de criptare a unui text o reprezintă *cifrul lui Cezar*, prin care fiecare literă dintr-un text dat este înlocuită cu litera aflată peste k poziții la dreapta în alfabet în mod circular. Valoarea k reprezintă *cheia secretă comună* pe care trebuie să o cunoască atât expeditorul, cât și destinatarul mesajului criptat. Decriptarea unui text constă în înlocuirea fiecărei litere din textul criptat cu litera aflată peste k poziții la stânga în alfabet în mod circular. Scrieți un program care să realizeze criptarea sau decriptarea unui text folosind cifrul lui Cezar. **Indicație de rezolvare:** se va utiliza formula $e_k(x) = (x + k) \bmod 26$ pentru criptarea unui caracter x folosind cheia secretă k, respectiv formula $d_k(x) = (x - k) \bmod 26$ pentru decriptare. De asemenea, se vor utiliza funcțiile *ord* și *chr* pentru manipularea caracterelor.

6. 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!

7. Știind că 1 ianuarie 1702 a picat într-o zi de duminică, să se citească de la tastatură o dată mai recentă, și să se spună în ce zi a săptămânii cade aceasta.
Puteți să faceți 2 cazuri - în care inputul este dat de forma "1 10 2019", sau "1 octombrie 2019". Folosiți funcția *range()* pentru a itera printre ani, respectiv instrucțiuni if/elif/else pentru a trata cazurile de ani bisecți. Puteți folosi un dicționar sau o listă pe post de switch/case ca să aflați în ce zi a săptămânii pică data respectivă.

Formula pentru an bisect (leap year) este:
if (*year* is not divisible by 4) **then** (it is a common year)
else if (*year* is not divisible by 100) **then** (it is a leap year)
else if (*year* is not divisible by 400) **then** (it is a common year)
else (it is a leap year)

8. "Traduceri"

a) Se citește de la tastatură un text. Se cere să se "traducă" în limba păsărească textul dat astfel: după fiecare vocală se adaugă litera p și încă o dată acea vocală (după a, e, i, o, u se adaugă respectiv pa, pe, pi, po, pu). **Exemplu:** "Ana are mere." devine "Apanapa aparepe meperepe."
Fiind dat un astfel de text în limba păsărească, se poate obține textul original? Dacă da, faceți asta.

b) Se citește de la tastatură un text în care cuvintele sunt despărțite în silabe cu ajutorul cratimelor. Se cere să se "traducă" textul dat în limba păsărească astfel: după fiecare silabă se adaugă litera p și se repetă ultima literă din acea silabă. Afișați traducerea și cu cratime, dar și fără.
Exemplu: "A-na a-re mul-te me-re ro-sii si de-li-cioa-se." devine
"Apa-napa apa-repe mulpl-tepe mepe-repe ropo-siipi sipi depe-lipi-cioapa-sepe." și
"Apanapa aparepe mulpltepe meperepe roposiipi sipi depelipicioapasepe."
Fiind dat un astfel de text în limba păsărească (cel care conține și cratime), se poate obține textul original? Dacă da, faceți asta.

9. Negociere

Se cere prelucrarea unui discuții dintre persoana A, care vinde un obiect, și persoana B, care oferă bani pentru el. O astfel de discuție se poate desfășura în modul următor:

- "Eu am de gând să vând vaza aceasta pentru $5. Ce plăcut, chiar mi-ar plăcea să o achiziționez, doar că am numai $3 la mine. Este suficient? Nu, insist să obțin 5$ pe ea. Bine, atunci voi scoate niște bani și-ți aduc cei $5."
- "Salut, am văzut în acel anunț că vindeți o mașină second-hand. M-ar interesa s-o achiziționez pentru suma de $2700. Vă amintesc că suma din anunț este de $3000, sir. Desigur. Și totuși, n-am putea ajunge la mijloc? $2850? $2850 de dolari, spuneți? Mi se pare corect, s-a făcut."

După cum se poate observa din exemple, regulile sunt acestea:

- se știe că fiecare persoană face câte o ofertă, pe rând, iar suma se apropie spre o valoare aflată între primele două oferte.
- Considerăm că persoana A este cea care oferă, cum este și logic :), prețul mai mare dintre primele două oferte.
- Când ultimele două oferte sunt egale, știm că cele două persoane au ajuns la un acord comun (*).

Cerințe:

a) Extrageți din text primele două valori. (hint: orice sumă este un număr după semnul $ în șir)
b) Decideți dacă cele două persoane "s-au înțeles" :). (vezi (*))

10. Numele Pre-Nume

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.
- Prenumele sunt cel mult două, iar dacă sunt două atunci sunt despărțite printr-o cratimă ('-').

11. Propoziții

Se dă un text (puteți de exemplu să luați text dintr-un paragraf de pe https://en.wikipedia.org/wiki/Python_(programming_language)).

1. Folosiți funcția *split* pentru a despărți textul în propoziții (care se termină prin '.').
2. Încercați și o altă abordare: în loc de a despărți numai prin '.', care poate fi prezent și după prescurtări și in punctele de suspensie (…), puteți să verificați ca fiecare propoziție să înceapă cu literă mare. Astfel, regula va fi că o propoziție începe cu literă mare și (cu excepția primeia) este precedată de un '.'.

12. Operații pe stringuri:

Vom considera următoarea operație <op>:

- dacă ultima literă a primului string este diferită de prima a celui de-al doilea, atunci:
  - <op> este echivalentă cu concatenarea de string-uri: "ad" <op> "a" = "ad" + "a" = "ada"
- dacă ultima literă a primului string este aceeași cu prima literă a celui de-al doilea string, atunci:
  - "de" <op> "eea" = "d" <op> "a" = "d" + "a" = "da" (toate literele 'din mijloc' comune dispar, și se continuă cu <op> aplicat resturilor din fiecare șir.
- Încă un exemplu:
  - "absf" <op> "ffsc" = "abs" <op> "sc" = "ab" <op> "c" = "ab" + "c" = "abc"

Cerință:

Pentru un șir format numai din litere citit de la tastatură decideți din care dintre operațiile sir_1 <op> sir_1_oglindit sau sir_1_oglindit<op> sir_1 se obține un șir mai lung. Afișați șirurile obținute la fiecare pas după aplicarea operației <op>.