1. Portmoneu cu bancnote.

Descriere: Să reprezentăm bancnotele care sunt într-un portmoneu. Presupunem că există bancnote de valoarea 1, 5, 10, 50 și 100.

Funcționalități:

- Să se adauge o bancnotă în portmoneu. Valoarea bancnotei trebuie să fie una validă (1, 5, 10, 50 sau 100).
- Să se afișeze suma totală din portmoneu.
- Să se cheltuie o sumă și să primească rest de la suma respectivă dacă este necesar. Suma cheltuită este multiplu de 1. Dacă suma cheltuită este mai mare decât suma totală, se va afișa un mesaj.

Exemple:

- Dacă portmoneul este inițial vid și adăugăm pe rând bancnotele 5, 50, 1, 1, 5, 10, 1, 100, 1, 5, 10 suma totală va fi 189.
- Dacă avem de cheltuit suma de 25, o variantă este să plătim cu bancnota de 50 și să primim rest 25 (evident, în bancnote valide). O variantă pentru conținutul portmoneului după plată și primirea restului este [5, 1, 1, 5, 10, 1, 100, 1, 5, 10, 10, 10, 5].
- Pentru ultima funcționalitate orice variantă de a alege bancnota (bancnotele) cu care să plătim și orice variantă de a primi rest (inclusiv și varianta de a primi rest numai în bancnote de 1) este acceptată, dacă conținutul portmoneului este corect (conține doar bancnote valide, și suma bancnotelor este corectă).

2. Competiția Eurovision

Descriere: Competiția Eurovision este o competiție muzicală, în care participă artiști din țări diferite. În ziua competiției artiștii prezintă piesele lor unul după altul în cadrul unui concert mare, care este difuzat în mai multe țări din Europa. După prezentarea pieselor, telespectatorii din toate țările pot vota piesa favorită, printr-un apel telefonic, dar nu pot vota pe artistul din țara lor. Când timpul alocat votului se termină, voturile sunt agregate, și transformate în puncte. Fiecare țară dă puncte pentru alte 5 țări: 10, 8, 6, 4 și 2 puncte în funcție de numărul apelurilor telefonice. Vrem să implementăm un sistem care gestionează punctele primite de toți artiști care participă.

Funcționalități:

- Adăugați punctele primite de la o țară (nu contează care țară). Vor fi introduse 5 țări, prima țară primește 10 puncte, a 2-a țară primește 8 puncte, a 3-a țară 6 puncte, a 4-a țară 4 puncte și ultima țară 2 puncte.
- Calculați punctajul primit de o țară (dacă au votat deja mai multe țări, o țară poate să aibă mai multe punctaje).
- Afișați care țară a câștigat competiția Eurovision (evident, țara cu numărul maxim de puncte câștigă).

Exemple:

- Dacă prima țară care votează dă puncte pentru țările: Suedia, România, Danemarca, Franța, Anglia, după vot avem punctele în modul următor: Suedia-10, România-8, Danemarca-6, Franța-4, Anglia-2.
- Dacă votează încă o țară: Danemarca, Islanda, Germania, Austria, România, după vot avem punctele în modul următor: Suedia-10, România-8, Danemarca-6, Franța-4, Anglia-2, Danemarca-10, Islanda-8, Germania-6, Austria-4, România-2.
- Dacă votează încă o țară: Germania, România, Slovenia, Suedia, Anglia, după vot avem punctele în modul următor: Suedia-10, România-8, Danemarca-6, Franța-4, Anglia-2, Danemarca-10, Islanda-8, Germania-6, Austria-4, România-2, Germania-10, România-8, Slovenia-6, Suedia-4, Anglia-2.
- În acest moment România are 18 puncte (8+2+8), Germania are 16 puncte (10 + 6) și Suedia are 14 puncte (10+4).
- Competiția este câștigată de România (18 puncte). Restul punctațelor – nu trebuie afișat tot clasamentul, doar țara câștigătoare - Germania(16), Danemarca (16), Suedia (14), Islanda (8), Slovenia (6), Franța (4), Anglia (4), Austria (4).

3. Meci de baschet

Descriere: Vrem să reținem evenimentele întâmplate pe parcursul unui meci de baschet. Evenimentele importante fiind aruncările la coș, evident. Fiecare aruncare este caracterizată de 3 informații: numele jucătorului care a aruncat la coș, de câte puncte a fost coșul (1, 2 sau 3) și dacă a fost coș sau ratare (o valoarea booleană, adevărat dacă a fost coș, fals dacă a fost ratare). Numele jucătorilor din prima echipă începe cu litera A urmat de numărul jucătorului (de ex. A23, sau A2). Numele jucătorilor din a 2-a echipă începe cu litera B urmat de numărul jucătorului (de ex. B23 sau B1). Nu sunt 2 jucători cu același nume (poate exista același număr în ambele echipe, de ex. A11 și B11, dar nu sunt 2 jucători cu același număr într-o echipă).

Funcționalități:

- Adăugați un eveniment, specificând numele jucătorului, punctajul și dacă a fost ratare sau nu.
- Aflați cei mai bun marcatori ai celor 2 echipe (jucătorii cu numărul maxim de puncte).
- Afișați scorul final al meciului și cât putea să fie scorul dacă toate coșurile intrau.

Exemple:

- Dacă la un moment dat adăugăm următoarele evenimente (A11, 3, fals), (A9, 1, true), (B11, 2, true), (A4, 2, true), (A4, 3, fals), (B8, 1, true), (B8, 1, true), (A5, 3, true), (B3, 2, true), (A11, 3, true), (B3, 2, true), (B3, 1, true), (A5, 2, true), (B11, 3, true), (A9, 2, true), (B3, 2, true), (A9, 2, fals).
- Cei mai buni marcatori sunt A5 (5 puncte) și B11 (7 puncte).
- Scorul final este: 13 – 14 și putea să fie 21-14 dacă toate coșurile intrau.

4. Anagrame

Descriere: Având 2 stringuri de lungime egală vrem să calculăm câți pași sunt necesari pentru a transforma primul string într-o anagramă a celui de al doilea string (2 stringuri sunt anagrame dacă sunt alcătuite din aceleași litere, de ex. listen și silent). Transformarea se face în pași, și la fiecare pas putem transforma o literă din primul string într-o altă literă (stringul 2 nu se modifică deloc). Pentru a număra câți pași sunt necesari să transformăm primul string într-o anagramă a celui de-al doilea string putem să adăugăm caracterele din fiecare cuvânt în câte un container ordonat, și putem să parcurgem containerele în paralel, și de fiecare dată când literele nu se potrivesc putem considera că este necesar un pas de transformare (aveți un exemplu mai jos). Vom numi acest algoritm *varianta1*. Algoritmul *varianta1* nu ne dă tot timpul numărul minim de pași. Implementați un algoritm care găsește numărul minim de pași (folosind tot 2 containere) (algoritmul *varianta2)*.

Funcționalități:

- Implementați algoritmul *varianta1* (aveți niște exemple mai jos).
- Implementați algoritmul *varianta2* (algoritm care gaseste numărul minim de transformări necesare).

Exemple:

- Dacă primul cuvânt este *baschet* iar cuvântul 2 este *handbal* (cele 2 cuvinte au tot timpul aceeași lungime). Vom construi câte un container ordonat din fiecare: [a, b, c, e, h, s, t] și [a, a, b, d, h, l ,n]. *Varianta1* parcurge elementele containerelor în paralel și numără câte un pas de transformare când avem elemente diferite:
  - a – a – e ok
  - b – a – pas de transformare
  - c – b – pas de transformare
  - e – d – pas de transformare
  - h – h – e ok
  - s – l – pas de transformare
  - t – n – pas de transformare
- Deci algoritmul *varianta1* va returna că este nevoie de 5 pași de transformare.
- Există o variantă mai optimă de transformare, care necesită doar 4 pași de transformare. De exemplu c în a, e în d, s în l și t în n. Algoritmul *varianta2* trebuie să găsească aceast numări de pași.
- Pentru cuvintele *postuniversitar* și *structuridedate*:
  - *Varianta1* ar trebui să returneze 8 pași
  - *Varianta2* ar trebui să returneze 6 pași

5. Fursecuri dulci

Descriere: Oana iubește fursecurile și are o cutie de *n* fursecuri. Fiecare fursec are o valoare asociată, care arată cât de dulce e fursecul (câte grame de zahăr conține). Oanei îi plac doar fursecurile care au mai mult de *K* grame de zahăr. Pentru a avea doar fursecuri cu minim *K* grame de zahăr, ea a inventat o operație specială prin care ia fursecul cu număr minim de grame de zahăr (*f1*) și fursecul cu a 2-a cea mai mică valoare de grame de zahăr (*f2*) și le combină într-un fursec care are *f1 + 2* f2* grame de zahăr. (*f1* și *f2* pot avea aceeași valoare).

Funcționalități:

- Câți pași sunt necesari pentru a avea doar fursecuri în care cantitatea de zahăr este mai mare sau egală cu K? (dacă nu se poate ajunge la o astfel de situație, afișați -1).
- Câte fursecuri va avea Oana când toate au minim K grame de zahăr?
- Afișați câte grame de zahăr sunt în fiecare fursec la finalul transformărilor.

Exemple:

- Dacă inițial sunt 8 fursecuri cu următoarele cantități de zahăr: [8, 2, 10, 3, 5, 11, 9, 8] și K este 10
    - Inițial combinăm fursecul cu 2 și 3 într-unul cu 2+2*3 = 8 grame de zahăr, și vom avea [8, 8, 10, 5, 11, 9, 8]
    - Combinăm 5 cu 8 și avem fursec cu 21 de grame de zahăr, [21, 8, 10, 11, 9 ,8]
    - Combinăm 8 cu 8 și avem fursec cu 24 de grame de zahăr, [21, 24, 10, 11, 9]
    - Combinăm 9 cu 10 și avem fursec cu 29 de grame de zahăr, [21, 24, 29, 11]
- Toate fursecurile au minim 10 grame de zahăr. Au fost necesare 4 transformări. Avem 4 fursecuri la final cu valori [21, 24, 29, 11].
- Dacă inițial sunt 5 fursecuri cu următoarele cantități de zahăr: [1,2,3,4,5] și K este 20
    - Combinăm 1 cu 2 pentru un fursec cu 5, [5,3,4,5]
    - Combinăm 3 cu 4 pentru un fursec cu 11, [5, 11, 5]
    - Combinăm 5 cu 5 pentru un fursec cu 15 [15, 11]
    - Combinăm 11 cu 15 pentru un fursec cu 41, [41]
- Toate fursecurile au minim 20 de grame de zahăr. Au fost necesare 4 transformări. Avem 1 fursec la final cu valoarea 41.

6. ***Scrisoare Anonimă***

Descriere: Vrem să trimitem o scrisoare anonimă cuiva. Ca să fie anonimă, nu vrem să o scriem de mână (calculator și imprimantă nu avem) ci vrem să decupăm literele dintr-o revistă și să le lipim una după alta ca să formeze mesajul care vrem să fie transmis. Pentru a trimite această scrisoare de o săptămână tot adunăm propoziții din reviste. Când găsim mai multe exemplare dintr-o revistă, putem adăuga aceeași propoziție de mai multe ori.

Funcționalități:

- Adăugați încă o propoziție (sau doar niște cuvinte) la propozițiile adunate (ne interesează doar literele – fără semne de punctuație - și presupunem că toate literele sunt mici).
- Verificați dacă mesajul pe care vrem să-l trimitem poate fi format din literele adunate până acum.
- Să scriem mesajul (adică să eliminăm acele litere care sunt necesare pentru scrierea mesajului). După scrierea mesajului ar trebui să avem posibilitatea să verificăm dacă un alt mesaj mai poate fi format (prin funcționalitatea nr. 2).

Exemple:

- Inițial pornim fără orice cuvânt. Dacă adăugăm propoziția ”verde pentru fotbal pe noul stadion” vom avea 2 litere de *a*, 4 litere de *e*, 2 litere de *r*, etc. Dacă mai adăugăm propoziția ”cele mai urate telefoane construite vreodata”, vom avea 7 litere de *a*, 12 litere de *e*, 5 litere de *r*, 2 litere *f*, etc.
- Având cele 2 propoziții adăugate, mesajul ”a murit dupa manevre de resuscitare gresite” nu poate fi formulat (lipsește un *m*, un *s* și un *g*), dar mesajul ”trecerea la ora de iarna” poate fi format.
- Dacă scriem mesajul ”trecerea la ora de iarna”, vom mai avea 6 litere de *t*, 1 literă de *r*, 8 litere de *e,* etc.

7. Mărgele pentru rochie

Descriere: Tania se duce la un croitor ca să comande o rochie nouă. Pentru a avea o rochie specială, ea dorește ca pe rochie să fie puse grupuri de mărgele. Fiecare grup este alcătuit din mărgele de aceeași culoare dar fiecare grup este alcătuit dintr-un număr diferit de mărgele (nu pot exista 2 grupuri cu același număr de mărgele). Croitoreasa are un număr infinit de mărgele (de toate culorile) și fiecare bucată costă suma *s*. Tania știe câte grupuri de mărgele vrea pe rochie, și știe pentru fiecare grupă numărul minim de mărgele care trebuie să fie în grupă.

Funcționalități:

- Afișați numărul minim de mărgele care vor fi pe rochia Taniei în total.
- Afișați suma totală care va fi plătită de Tania.
- Afișați câte mărgele vor fi în fiecare grupă.

Exemple:

- Dacă Tania vrea 4 grupe de mărgele cu minim 1, 3, 3 și 6 mărgele, ea va avea o grupa cu 6 mărgele, una cu 3, una cu 4 (nu poate avea 3 din nou și trebuie să fie minim 3) și una cu 1, deci în total 14 mărgele.
- Presupunând că fiecare mărgele costă 2 dolari, Tania va plăti 28 de dolari în total.
- Va avea grupe cu 1, 3, 4 și 6 mărgele pe rochie.

8. Jaful de bancă

Descriere: O bancă a fost jefuită și poliția vrea să vadă cine e suspect de jaf, folosind date GPS de la telefoanele mobile care au fost în apropierea băncii. Poliția primește de la compania de telefonie pentru fiecare persoană 4 informații: codul IMEI al telefonului (unic pentru un telefon), timestamp (ora când s-a făcut înregistrarea, un număr întreg între 0 și 1440 – minutul din cadrul zilei) și coordonatele GPS: coordonata X și coordonata Y (ambele valori de tip float/double).

Funcționalități:

- Adăugați datele pentru încă o persoană.
- Citind coordonatele GPS pentru banca jefuită și timestampul cu momentul când banca a fost jefuită, afișați lista suspecților. Suspect este o persoană care în momentul jafului era în bancă. Datele GPS nu sunt 100% precise, de aceea vom considera ca fiind în bancă orice persoană care avea coordonatele GPS la o distanță Euclidiană de maximum 1.1 de la poziția băncii. Distanța Euclidiană dintre coordonatele GPS (x,y) și (xm, ym) se calculează cu formula $\sqrt{(x - xm)^2 + (y - ym)^2}$.
- Folosind camere video, poliția și-a dat seama de direcția în care s-a dus hoțul, și a cerut de la compania de telefonie și datele pentru o altă locație de pe acest traseu. Având mulțimea informațiilor primite de la compania de telefonie și știind că hoțul avea nevoie de 10-15 minute să ajungă la locul respectiv, afișați lista suspecților (persoanele care au fost în bancă în timpul jafului – calculat la punctul anterior - și care peste 10-15 minute sunt la o distanță maximă de 1.1 de la a 2-a locație).

Exemple:

- Să presupunem că avem următoarele date pentru persoane de la compania de telefonie. Fiecare elemente e sub forma IMEI, Timestamp și coordonate GPS.
  - (123, 751, 12.3, 15.2)
  - (143, 750, 15.5, 12)
  - (521, 760, 16, 11)
  - (481, 750, 20, 10)
  - (811, 750, 14, 12.4)
  - (731, 739, 12.3, 15.2)
  - (355, 750, 15.7, 11.9)
- Presupunând că jaful s-a petrecul la timpul 750 și banca e la coordonatele 15, 12:
  - (123, 751, 12.3, 15.2) – nu e ok timpul
  - (143, 750, 15.5, 12) – timpul e ok, distanța e 0.5 –posibil suspect
  - (521, 760, 16, 11) – nu e ok timpul
  - (481, 750, 20, 10) – timpul e ok, distanța e 5.38 – nu e suspect posibil
  - (811, 750, 14, 12.4) – timpul e ok, distanța e 1.07 – posibil suspect
  - (731, 739, 12.3, 15.2) – nu e ok timpul
  - (355, 750, 15.7, 11.9) – timpul e ok, distanța e 0.7 – posibil suspect
- Lista primită de la a 2-a locație, aflată la coordonatele 21, 11.
  - (123, 763, 10, 14) – nu era suspect la bancă
  - (433, 766, 21.1, 10.9) – nu era la bancă în timpul jafului
  - (355, 762, 20,11.3) – era suspect la bancă și e suspect în continuare
  - (481, 764, 20.2, 11.1) – nu era suspect la bancă

- (811, 762, 19.8, 11.8) – era suspect la bancă dar e prea departe de locul 2.
- (143, 765, 20.2, 11.6) – era suspect la bancă și e suspect în continuare

9. Numere de înmatriculare

Descriere: Poliția vrea să rețină toate numerele de înmatriculare care sunt libere pentru un județ. Numerele de înmatriculare sunt alcătuite din 2 litere, care reprezintă județul, 2 cifre și 3 litere.

Funcționalități:

- Adăugați un număr de înmatriculare (când cineva renunță la mașină, numărul devine liber, poate fi dat la o altă mașină).
- Când un client nou vine pentru un număr de înmatriculare, prima dată i se oferă un număr la întâmplare, dacă îi convine va primi numărul respectiv (care este eliminat din sistemul de numere libere). Dacă nu îi convine, poate să încerce următoarea opțiune/funcționalitate.
- Utilizatorul cere un anumit număr de înmatriculare. El poate să dea un număr complet (de exemplu: CJ-12-ABC) caz în care, dacă numărul este liber îl va primi, sau poate să dea părți din număr (doar numărul, sau doar literele) caz în care aplicația va afișa toate numerele de înmatriculare libere care conțin partea specificată și utilizatorul poate să aleagă numărul pe care îl vrea. În orice caz, odată ce a ales un număr, numărul respectiv este eliminat din sistemul de numere libere.

Exemple:

- Presupunem că am adăugat în mulțimea numerelor de înmatriculare următoarele numere: CJ-32-AXD, CJ-44-IFT, CJ-21-FJU, CJ-91-RQW, CJ-71-ETA, CJ-08-AAA, CJ-32-NTT, CJ-06-ETA
- Dacă clientul vreau doar un număr, poate să primească oricare dintre numere
- Dacă clientul vrea numărul CJ-11-AAA, el nu îl poate primi, pentru că nu este liber. Dacă cineva vrea numărul CJ-21-FJU, el va primi numărul. Dacă cineva vrea număr de înmatriculare cu textul ETA, I se va afișa CJ-71-ETA și CJ-06-ETA și poate să aleagă din aceste 2 numere.

10. Înmulțire

Descriere: Având un tablou (vector) de numere naturale, pentru fiecare poziție *p* din vector să se afișeze produsul celor mai mari (sau celor mai mici) K elemente din tablou până la poziția *p*.

Funcționalități:

- Să citiți elementele tabloului
- Să citiți numărul K și pentru fiecare poziție *p* afișați produsul celor mai mari K elemente din tablou. Dacă până la o poziție *p* nu sunt K elemente, afisați -1.
- Să citiți numărul K și pentru fiecare poziție *p* afișați produsul celor mai mici K elemente din tablou. Dacă până la o poziție *p* nu sunt K elemente, afisați -1.

Exemple:

- Presupunem că tabloul conține elementele [5, 1, 8, 4, 6, 3, 10, 9].
- Dacă vrem produsul celor mai mari K = 3 elemente vom avea:
    - -1 (nu sunt 3 elemente)
    - -1 (nu sunt 3 elemente)
    - 40 (5 * 1 * 8)
    - 160 (5 *8 * 4)
    - 240 (5 * 8 * 6)
    - 240 (5 * 8 * 6)
    - 480 (8 * 6 * 10)
    - 720 (8 * 9 * 10)
- Dacă vrem produsul celor mai mici K = 4 elemente, vom avea:
    - -1 (nu sunt 4 elemente)
    - -1 (nu sunt 4 elemente)
    - -1 (nu sunt 4 elemente)
    - 160 (5 * 1 * 8 * 4)
    - 120 (5 * 1 * 4 * 6)
    - 60 (5 * 1 * 3 * 4)
    - 60 (5 * 1 * 3 * 4)
    - 60 (5 * 1 * 3 * 4)

11. Atracții turistice de vizitat

- **Descriere:** Emilia vrea să viziteze cât mai multe atracții turistice din jurul ei. Dar din păcate ea nu are foarte mult timp liber, deci decide să vizitele doar cele mai apropiate K atracții. Fiecare atracție turistică este reprezentată prin coordonatele ei GPS, x și y. Emilia locuiește la coodonatele GPS 0,0. Pentru a măsura distanța față de atracțiile turistice, folosim distanța Euclidiană. Distanța Euclidiană dintre coordonatele GPS (x,y) și (xm, ym) se calculează cu formula $\sqrt{(x - xm)^2 + (y - ym)^2}$.

Funcționalități:

- Adăugați o nouă atracție turistică.
- Afișați cele mai apropiate K atracții turisitice care să fie vizitate de Emilia (K se citește de la tastatură). După afișare aceste atracții nu dispar din lista Emiliei.
- Știind că Emilia poate să parcurgă distanța maximă M, afișați maximum câte atracții poate să viziteze (după fiecare atracție vizitată ea se întoarce acasă, și de acolo pornește în vizita următoare).

Exemple:

- Dacă inițial avem atracții la coordonatele:
  - (10, 13) – distanță 16.40
  - (16, 20) – distanță 25.61
  - (7, 40) – distanță 40.6
  - (100, 32) – distanță 104.99
  - (90, 50) – distanță 102.95
  - (7, 15) – distanță 16.55
  - (45, 67) – distanță 80.70
- Dacă Emilia are timp să viziteze cele mai apropiate K=4 atracții turisitce ea se va duce la coordonatele (10, 13), (7, 15), (16, 20), (7, 40).
- Dacă distanța maximă M este 120, Emilia se va duce la:
  - (10,13) – 32.8 dus-întors
  - (7, 15) – 33.1 dus-întors (în total 65.9)
  - (16, 20) – 51.22 dus-întors (în total 117.12)

12. Comoară la capătul traseului

Descriere: Manasa a ieșit la plimbare cu prietenii. Ea găsește un traseu, alcătuit din pietre, și pe fiecare piatră este un număr. Manasa observă că diferența dintre numerele de pe 2 pietre consecutive este valoarea *a*, *b*, sau *c*. Legenda spune că la capătul traseului este o comoară și dacă Manasa poate să ghicească ce număr este pe ultima piatră comoara poate fi a ei. Știm că pe prima piatră este valoarea 0, cunoaștem lungimea traseului și valorile *a, b*, și *c.*

Funcționalități:

- Afișați câte numere diferite pot exista pe ultima piatră.
- Verificați dacă un număr dat poate fi pe careva dintre pietrele de pe traseu.
- Afișați numerele posibile de pe pietre pentru fiecare pas.

Exemple:

- Dacă traseul are lungime 4 și *a*= 3, *b = 4* și *c =* 2:
    - Pe prima piatră este valoarea 0
    - Pe piatra a 2-a putem avea valorile 3, 4, 2
    - Pe piatra a 3-a putem avea valorile:
        - 6 (3+3 sau 4+2 sau 4+2)
        - 7 (3+4 sau 4+3)
        - 5 (3+2 su 2+3)
        - 8 (4+4)
        - 4 (2+2)
    - Pe piatra a 4-a putem avea valorile:
        - 9 (6+3 sau 5+4 sau 7+2)
        - 10 (7+3 sau 6+4 sau 8+2)
        - 8 (5+3 sau 4+4 sau 6+2)
        - 11 (8+3 sau 7+4)
        - 7 (4+3 sau 5+2)
        - 12 (8+4)
        - 6 (4+2)
- Pe ultima piatră pot exista 7 valori diferite (9, 10, 8, 11, 7, 12, 6)
- Numerele 1 și 13 (de exemplu) nu există pe nicio piatră dar 8 si 12 (de exemplu) da.
- Numerele de pe pietre pentru fiecare pas sunt cele calculate mai sus.
- Dacă traseul are lungime 3 și *a*=3, *b = 7* și c= 13
    - Pe prima piatră avem valoarea 0
    - Pe piatra a 2-a putem avea valorile 3, 7, 13
    - Pe piatra a 3-a putem avea:
        - 6 (3+3)
        - 10 (3 + 7 sau 7 +3)
        - 16 (3 + 13 sau 13 + 3)
        - 14 (7+7)
        - 20 (7 + 13 sau 13+7)
        - 26 (13+13)
    - Pe ultima piatră pot exista 6 valori diferite
    - Valoarea 11, 12, 8 (sunt doar exemple) nu există pe nicio piatră, dar 14 și 16 există
    - Numerele de pe pietre fentru fiecare pas sunt cele calculate mai sus.

13. Teme la cursul de programare

Descriere: Într-un curs intensiv de programare studenții trebuie să trimită prin mail temele. Fiecare temă este identificată de numărul de înmatriculare pentru student, numărul temei (L1, L2, L3, etc.) și de data la care a fost trimisă tema. Profesorul reține temele primite pe email pe baza datei la care a fost trimisă tema.

Funcționalități:

- Adăugați încă o temă.
- Afișați studenții care au terminat cursul. Pentru a termina cursul, un student trebuie să trimită toate cele 7 teme primite într-o perioadă de maximum 21 de zile. Afișarea trebuie făcută în ordinea crescătoare a numărului matricol.
- Dintre studenții care nu au terminat încă cursul, afișați-i pe cei care mai au șansa să termine cursul. Pot termina cursul doar studenții care, chiar dacă nu au trimis toate temele încă, dar începând din data când au trimis prima temă, au trimis câte o temă cu maxim 5 zile după trimiterea temei anterioare.

Exemple:

- Să presupunem că următoarele teme au fost trimise până acum (fiecare temă conține nr matricol, cod și data):
  - (1523, L1, 1.11.2017)
  - (1723, L1, 3.11.2017)
  - (1523, L3, 3.11.2017)
  - (1965, L2, 3.11.2017)
  - (1472, L1, 4.11.2017)
  - (1723, L2, 5.11.2017)
  - (1723, L3, 6.11.2017)
  - (1963, L5, 6.11.2017)
  - (1523, L2, 7.11.2017)
  - (1723, L4, 10.11.2017)
  - (1965, L1, 11.11.2017)
  - (1723, L5, 12.11.2017)
  - (1523, L4, 12.11.2017)
  - (1523,L5, 13.11.2017)
  - (1723, L6, 15.11.2017)
  - (1723, L7, 20.11.2017)
  - (1523, L6, 21.11.2017)
  - (1523, L7, 25.11.2017)
- Singurul student care a terminat cursul e cel cu nr matricol 1723 (si 1523 a trimis toate temele dar în 24 de zile)
- Studenții care mai au șansa să termine:
  - 1472 (a trimis o singură temă)
  - 1963 (a trimis o singură temă)

14. Statistici meci de fotbal de sală

Descriere: Într-o echipă de fotbal de sală sunt 5 jucători. Putem reține toți jucătorii pentru un meci într-un container cu 10 elemente, primele 5 elemente reprezentând jucătorii din prima echipă și ultimele 5 elemente reprezintând jucătorii din a 2-a echipă. Fiecare jucător este reținut prin coordonatele lui pe teren (x,y), numere reale, unde x este între 0 și 90, și y este între 0 și 70. Pentru a reține mai bine evenimentele din timpul meciului, reținem coodronatele jucătorilor după fiecare interval de 10 secunde într-un container numită *poziții* (avem un container în care fiecare element este un container cu coordonate).

Funcționalități:

- Adăugați coordonatele pentru toți jucători în containerul *poziții* (se citesc cele 10 coordonate, se pun într-un container și aceast container se adaugă la finalul containerului de *poziții).*
- Citind coordonatele pentru minge (sub forma x, y), afișați care jucător (echipa 1 sau 2 și numărul jucătorului) este în posesia mingii pe baza ultimelor poziții. Un jucător este în posesia mingii dacă distanța Euclidiană față de minge este mai mică ca 10. Distanța Euclidiană între jucătorul cu coordonate (x,y) și mingea cu coordonate (xm, ym) este $\sqrt{(x-xm)^2+\ (y-ym)^2}$. Dacă sunt mai mulți jucători care sunt la o distanță mai mică de 10 de la minge, atunci nimeni nu este în posesia mingii.
- Afișați distanța alergată de fiecare jucător și distanța totală pentru cele 2 echipe.

Exemple:

- Să presupunem că la începutul înregistrării pozițiilor jucătorii sunt la pozițiile următoare: [(50, 2), (20, 20), (32, 60), (65, 21), (50, 50), (50, 69), (43, 31), (23, 11), (66, 21), (42, 59)] – primele 5 perechi sunt coordinate pentru prima echipă, ultimele 5 sunt coordonate pentru jucători din a 2-a echipă.
- Dacă mingea este la coordonata (19, 35) distanțele jucătorilor fată de minge sunt: [45.27, 15.03, 28.17, 48.08, 34.43, 46.01, 24.33, 24.33, 49.04, 33.24]. Nimeni nu este în posesia mingii.
- Să presupunem că peste 10 secunde pozițiile sunt: [(52, 2),(25, 17), (30, 65), (61, 30), (40, 47), (52, 65), (40, 33), (20, 15), (62, 19), (40, 57)].
- Dacă mingea este la coordonata (25, 24) distanțele jucătorilor față de minge sunt: [34.82, 7, 41.30, 36.49, 27.45, 49.09, 17.49, 10.29, 37.33, 36.24]. Mingea e în posesia jucătorului 2 din prima echipă.
- Să presupunem că peste 10 secunde pozițiile sunt: [(40, 5), (29, 20), (23, 59), (54, 41), (49, 43), (58, 61), (37, 31), (22, 29), (67, 20), (43, 54)].
- Dacă mingea e la coordonata (26, 31), distanțtele jucătorilor față de minge sunt: [29.52, 11.40, 28.16, 29.73, 25.94, 43.86, 11, 4.47, 42.44, 28.60]. Mingea e în posesia jucătorului 3 din a 2-a echipă.
- Distanțele parcurse de jucători:
    - (50,2) – (52,2) – (40,5) = 2 + 12.36 = 14.36
    - (20,20) – (25, 17) – (29,20) = 5.83 + 5 = 10.83
    - (32, 60) – (30, 65) - (23, 59) = 5.38 + 9.21 = 14.59
    - (65,21) – (61, 30) – (54, 41) = 9.84 + 13.03 = 22.87
    - (50,50) – (40, 47) - (49, 43) = 10.44 + 9.84 = 20.28
    - (50,69) – (52, 65) - (58, 61) = 4.47 + 7.21 = 11.68
    - (43,31) - (40, 33) - (37, 31) = 3.60 + 3.60 = 7.20
    - (23, 11) – (20, 15) - (22, 29) = 5 + 14.14 = 19.14

- (66, 21) – (62, 19) – (67, 20) = 4.47 + 5.09 = 9.56
- (42, 59) - (40, 57) - (43, 54) = 2.82 + 4.24 = 7.06

- Prima echipă a alergat 82.93 metrii iar a 2-a 54.64.

15. Scorul de la Bowling

Descriere: Un joc de bowling este alcătuit din *X* ture, și în fiecare tură un jucător poate să arunce de 2 ori cu bila. Scorul jucătorului pentru fiecare tură este numărul popicelor dărâmate plus un eventual bonus. Bonusul poate fi primit în 2 situații, pentru *strike* și pentru *spare*. *Spare* avem când jucătorul dărâmă toate cele 10 popice cu cele 2 aruncări din tura respectivă. În acest caz bonusul este numărul de popice dărâmate la aruncarea următoare. *Strike* avem când juctorul dărâmă toate cele 10 popice dintr-o singură aruncare (în acest caz aruncarea a 2-a nu mai există). Bonusul pentru un strike este numărul popicelor dărâmate în tura următoare (deci la 2 aruncări, dacă există). În caz că avem mai multe *strike*-uri sau *spare*-uri consecutive, bonusul pentru fiecare caz se ia doar pentru tura următoare.

Funcționalități:

- Citiți și memorați numărul de popice dărâmate de un jucător pentru X ture (se citește X-ul și după aceea numărul de popice pentru cele X ture). Dacă în ultima tură jucătorul are *spare* sau *strike*, jocul se termină, nu mai are voie la aruncări extra.
- Afișați câte *spare*-uri și câte *strike*-uri a avut jucătorul (calculați valorile doar dacă se alege această funcționalitate, nu le calculați și rețineți în timpul citirii).
- Afișați scorul după fiecare tură (calculați valorile doar dacă se alege această funcționalitate, nu le calculați și rețineți în timpul citirii).

Exemple:

- Dacă au fost în total 6 ture: cu următoarele numere de popice dărâmate: 1, 4, 4, 5, 6, 4, 5, 5, 10, 0, 1 scorul pe ture, este:
  - Tura 1: 1 + 4 = 5
  - Tura a 2-a: 4 + 5 = 9
  - Tura a 3-a: 6 + 4 = 10 + 5 = 15 – avem *spare* (10 popice darâmate din 2 aruncări, bonusul este numărul de popice dărâmat la aruncarea următoare)
  - Tura a 4-a: 5 + 5 = 10 + 10 = 20 – avem *spare* din nou
  - Tura a 5-a: 10 + 1 = 11 – avem *strike* (toate popicele dărâmate dintr-o singură lovitură), bonusul este numărul popicelor dărâmate în tura următoare (în cele 2 aruncări)
  - Tura a 6-a: 0 +1 = 1
- Scor total: 5+9+15+20+11+1 = 61, au fost 2 spare-uri și 1 strike

16. Magazinul cu pantofi

Descriere: Într-un magazin care vinde pantofi există un singur raft pe care sunt puse toate cutiile de pantofi, una după alta, în ordine crescătoare a numărului de pe pantofi (de ex. 36, 37, 42, etc.). Fiecare pantof are o mărime, o culoare și un preț.

Funcționalități:

- Adăugați pantofi noi la raft (când vânzătorul aduce pantofi noi)
- Verificați dacă oferta de pantofi este echilibrată (oferta de pantofi se consideră echilibrată dacă diferența maximă dintre stocurile pentru fiecare mărime existentă este 3.
- O clientă mai *specială,* Marisol, are o metodă *interesantă* de a cumpăra pantofi. Alege în mod aleator 2 poziții de pe raft, și cere să-i fie aduse toate perechile de pantofi dintre cele 2 poziții (aceste elemente sunt șterse). Clienta parcurge șirul de pantofi și dintre acele perechi care corespund cu mărimea ei, alege fiecare a 2-a pereche de pantofi. Restul pantofilor sunt puși înapoi pe raft. Afișați lista pantofilor cumpărați de clienta, lista pantofilor rămași pe raft după plecarea ei și prețul plătit de ea.

Exemple:

- Să presupunem că la un moment dat avem următoarele pantofi:
  - (34, negru, 25)
  - (34, verde, 50)
  - (34, gri, 32)
  - (35, negru, 35)
  - (35, negru, 10)
  - (35, rosu, 71)
  - (36, gri, 13)
  - (36, verde, 43)
  - (36, negru, 39)
  - (36, negru, 32)
  - (37, negru, 15)
  - (37, verde, 43)
  - (38, negru, 33)
  - (38, negru, 27)
- Oferta de pantofi este echilibrată: avem 3 pantofi 34, 3 pantofi 35, 4 pantofi 36, 2 pantofi 37, 2 pantofi 38. Diferența maximă 4-2 = 2.
- Dacă Marisol poartă 36 și alege pozițiile 6, 12, ea va primi pantofii:
  - (36, gri, 13)
  - (36, verde, 43)
  - (36, negru, 39)
  - (36, negru, 32)
  - (37, negru, 15)
  - (37, verde, 43)
  - (38, negru, 33)
- Din care va alege (36, gri, 13) și (36, negru, 39) și va plăti 52 sau va alege (36, verde, 43) și (36, negru, 32) și va plăti 75 (oricare variantă e corectă).

17. Notificări pentru activitate frauduloasă

Descriere: O bancă are o regulă simplă de a notifica utilizatorii despre posibile activități frauduloase în contul lor. Ei trimit o notificare dacă găsesc o tranzacție care este mai mare decât de 2 ori cheltuiala mediană pentru ultimele *d* zile. Pentru a afla mediana unei secvențe de valori este necesară ordonarea crescătoare a secvenței și găsirea elementului de pe poziția de mijloc (de exemplu la 5 elemente se ia elementul de pe locul 3). Dacă numărul elementelor este par, se ia media celor 2 numere de la mijloc (de exemplu la 6 elemente se ia media numerelor de pe pozițiile 3 și 4). Dacă înaintea unei tranzacții sunt mai puțin de *d* tranzacții, nu se trimite nici-o notificare.

Funcționalități:

- Adăugați o tranzacție (tranzacția se reține prin data tranzacției și suma). Nu este garantat că utilizatorul introduce tranzacțiile în ordine cronologică, dar ar trebui reținute în acest fel.
- Verificați dacă o tranzacție este frauduloasă (se iau cele *d* tranzacții anterioare – pe baza datei tranzacției verificate și se verifică condiția din descriere).
- Verificați câte tranzacții frauduloase sunt în toată istoria utilizatorului.

Exemple:

- Să presupunem că utilizatorul a introduc următoarele tranzacții:
  - (42, 1.11.2017)
  - (64, 1.11.2017)
  - (66, 2.11.2017)
  - (121, 3.11.2017)
  - (70, 4.11.2017)
  - (32, 5.11.2017)
  - (101, 7.11.2017)
  - (42, 8.11.2017)
  - (79, 9.11.2017)
  - (99, 10.11.2017)
  - (132, 11.11.2017)
  - (551, 13.11.2017)
- Dacă d = 8 (adică ne uităm la cele 8 tranzacții anterioare:
  - Primele 8 tranzacții nu sunt frauduloase (nu avem suficiente date pentru verificare)
  - Pentru tranzactia (79, 9.11.2017) sumele pentru cele 8 tranzacții anterioare sunt:
    - (32, 42, 42, 64, 66, 70, 101, 121)
    - Medianul este (64+66)/2 = 65
    - 79 e mai mic decât 2*65 – e în regulă
  - Pentru tranzacția (99, 10.11.2017) sumele pentru cele 8 tranzacții anterioare sunt:
    - (32, 42, 64, 66, 70, 79, 101, 121)
    - Medianul este (66+70)/2 = 68
    - 99 e mai mic decât 2* 68 – e în regulă
  - Pentru tranzacția (132, 11.11.2017)
    - (32, 42, 66, 70, 79, 99, 101, 121)
    - Medianul este (70+79)/2 = 74.5
    - 132 e mai mic decât 2*74.5 – e în regulă
  - Pentru tranzacția (551, 13.11.2017)
    - (32, 42, 70, 79, 99, 101, 121, 132)
    - Medianul este (79+99)/2 = 89
    - 551 e mai mare decât 2*89 – activitate frauduloasă

18. Gâsca Roșie

Descriere: Jocul Gâsca Roșie este un joc de cărți, jucat de 2 persoane cu un set de cărți de 2 culori: negru și roșu (valoarea cărților nu contează, doar culoarea). La începutul jocului setul de cărți se împarte în mod aleator la cei 2 jucători, fiecare primește același număr de cărți. Pe parcursul jocului, cei 2 jucători pun câte o carte din partea de sus a propriului teanc pe masă. Dacă un jucător pune o carte de culoare roșie pe masă, celălalt jucător ia toate cărțile de pe masă și le pune la capătul de jos al teancului propriu. Câștigătorul este jucătorul care rămâne fără cărți.

Funcționalități:

- Simulați jocul Gâsca Roșie pentru n cârți (n e par, în teanc vor fi n/2 cărți roșii și n/2 cărți negri).

Exemple:

- Dacă inițial avem n = 10 cărți. Împărțim aleator cărțile la jucători și vom avea:
  - (partea de jos) R N N R R (partea de sus)
  - (partea de jos) N R N N R (partea de sus)
- Jocul (verde indică situația când un jucător a plasat o carte pe masă, roșu indică situația când un jucător a ridicat cărțile de pe masă):

Jucator 1	Masa	Jucator 2	Explicație
R N N R R		N R N N R	Începem jocul
R N N R	R	N R N N R	Jucător 1 pune carte roșie
R N N R		R N R N N R	Jucător 2 ridică cărțile de pe masă
R N N R	R	R N R N N	Jucător 2 pune carte roșie
R R N N R		R N R N N	Jucător 1 ridică cărțile de pe masă
R R N N	R	R N R N N	Jucător 1 pune carte roșie
R R N N		R R N R N N	Jucător 2 ridică cărțile de pe masă
R R N N	N	R R N R N	Jucător 2 pune carte neagră
R R N	N N	R R N R N	Jucător 1 pune carte neagră
R R N	N N N	R R N R	Jucător 2 pune carte neagră
R R	N N N N	R R N R	Jucător 1 pune carte neagră
R R	N N N N R	R R N	Jucător 2 pune carte roșie
N N N N R R R		R R N	Jucător 1 ridică cărțile de pe masă
N N N N R R	R	R R N	Jucător 1 pune carte roșie
N N N N R R		R R R N	Jucător 2 ridică cărțile de pe masă
N N N N R R	N	R R R	Jucător 2 pune carte neagră
N N N N R	N R	R R R	Jucător 1 pune carte roșie
N N N N R		N R R R R	Jucător 2 ridică cărțile de pe masă
Etc...			

19. Ieșire din labirint cu un robot

Descriere: Un robot se alfă într-un labirint (reprezentat ca o matrice, în care valori de 1 reprezintă perete, iar valori de 0 reprezintă culoar unde robotul poate să treacă). Robotul se poate deplasa doar în zonele libere (culoare) în cele 4 direcții: Nord, Sud, Est și Vest.

Funcționalități:

- Să determinați dacă robotul poate ieși din labirint. Se consideră că robotul a ieșit din labirint dacă ajunge pe o poziție pe marginea labirintului (prima linie, ultima linie, prima coloană, ultima coloană).
- Să determinați dacă robotul poate ajunge la o anumită poziție în labirint.
- Să determinați numărul minim de pași în care robotul poate ieși din labirint.

Exemple:

- Dacă avem labirintul următor (7 x 7):
  - 0 0 0 1 0 1 1
  - 1 0 1 1 0 1 1
  - 1 0 0 0 0 0 0
  - 0 1 1 R 1 0 0
  - 0 0 0 1 0 0 1
  - 0 1 1 1 0 0 1
  - 0 0 0 1 0 1 0
- Robotul este pe poziția 3, 3 (primul rând e rândul 0), acolo unde e litera R.
- Robotul poate ieși din labirint.
- La poziția 5, 0 (de exemplu) robotul nu poate ajunge, la poziția 5,4 poate.
- Numărul minim de pași pentru a ieși din labirint este: 4 (3,3) – (2,3) – (2, 4) – (2,5) – (2,6)

20. Joc de cărți

Descriere: Vrem să implementăm un joc de cărți care poate fi jucat de 2, 3 sau 4 jucători, și este jucat folosind un set de 24 de cărți. Cărțile sunt de 4 culori (R, T, S și M), și din fiecare culoare sunt următoarele valori: 11, 10, 4, 3, 2, 9 (valorile sunt date în ordine descrescătoare, cartea cu numărul 9 valorează 0 puncte). Se împart cărțile în mod aleator jucătorilor (dacă sunt 2 jucători, vor primi câte 12 cărți, dacă sunt 3 jucători câte 8, iar la 4 jucători câte 6). Jucătorii vor ordona cărțile după importanța lor. Cărțile sunt ordonate după câte puncte valorează. De exemplu, dacă un jucător are cărțile (T 3) (R 2) (M 10) (R 11) (M 3) (S 9) ordinea lor va fi (R 11) (M 10) (T 3) (M 3) (R 2) (S 9) – (T 3) și (M 3) pot veni în ordine inversă. Se alege aleator un jucător care să înceapă.
Pe parcursul jocului fiecare jucător pune jos cartea mai valoroasă și jucătorul care a pus cartea de valoare maximă ia toate cărțile de pe masă. Dacă mai mulți jucători pun cărți cu aceeași valoare (de exemplu avem (T 10) (S 10) (M 2) (S 4) la un joc cu 4 jucători) jucătorul care a pus prima dată cartea maximă ia cărțile. Jucătorul care a luat cărțile va pune prima dată carte în tura următoare. La finalul jocului pentru fiecare jucător se adună punctele de pe cărți luate, și câștigătorul este cel care a luat cele mai multe puncte (cărțile cu 9 valorează 0 puncte și aici).

Funcționalități:

- Simulați jocul alegând la început numărul de jucători.

Exemple:

- Să simulăm un joc cu 4 jucători. Cărțile primite inițial și ordonate după valoare
  - J1: (11, T) (11, R) (4, T) (3, T) (9, R) (9, M)
  - J2: (10, S) (10, R) (3, R) (3, M) (2, T) (9, S)
  - J3: (11, S) (4, R) (4, S) (4, M) (2, R) (2, S)
  - J4: (11, M) (10, T) (10, M) (3, S) (2, M) (9, T)
- Presupunem că jucătorul 3 începe:
  - Tura 1: (11, S) (11, M) (11, T) (10, S) – jucătorul 3 ia cărțile.
  - Tura 2: (4, R) (10, T) (11, R) (10, R) – jucătorul 1 ia cărțile
  - Tura 3: (4, T) (3, R) (4, S) (10, M) – jucătorul 4 ia cărțile
  - Tura 4: (3, S) (3, T) (3, M) (4, M) – jucătorul 3 ia cărțile
  - Tura 5: (2, R) (2, M) (9, R) (2, T) – jucătorul 3 ia cărțile
  - Tura 6: (2, S), (9, T), (9, M), (9, S) – jucătorul 3 ia cărțile
- Punctaje:
  - Jucătorul 1: 4+10+11+10 = 35
  - Jucătorul 2: 0
  - Jucătorul 3: 11+11+11+10+3+3+3+4+2+2+2+2 = 64
  - Jucătorul 4: 4+3+4+10 = 21
- Jucătorul 3 a câștigat

21. Ieșire din labirint cu robot stricat

Descriere: Un robot se alfă într-un labirint (reprezentat ca o matrice, în care valori de 1 reprezintă perete, iar valori de 0 reprezintă culoar unde robotul poate să treacă). Robotul se poate deplasa doar în zonele libere (culoar) în cele 4 direcții: Nord, Sud, Est și Vest. Din păcate anumite rotițe s-au stricat în robot și nu poate să meargă spre Vest. Dacă vrea să se deplaseze spre Vest, trebuie să efectueze 3 întoarceri spre dreapta după care face pasul și mai face o întoarcere spre dreapta ca să privească spre Nord din nou.

Funcționalități:

- Verificați dacă robotul poate ieși din labirint fără să meargă spre Vest. Afișați numărul de pași necesari.
- Calculați timpul minim necesar pentru robot să iasă din labirint. Deplasarea spre Nord și Sud iau câte 1 secundă, mersul spre Est ia 2 secunde (pasul efectiv, dar merge mai încet lateral), mersul spre Vest ia 5 secunde (pasul efectiv și cele 4 întoarceri).

Exemple:

- Dacă avem labirintul următor (7 x 7):
    - 0 0 0 1 1 0 1
    - 1 0 1 1 0 0 1
    - 0 0 0 0 0 1 1
    - 0 1 1 0 1 0 0
    - 0 R 0 0 0 0 1
    - 0 1 1 1 0 0 1
    - 0 0 0 1 0 1 0
- Robotul este pe poziția 4, 1 (primul rând e rândul 0), acolo unde e litera R.
- Robotul poate ieși din labirint fără a merge spre Vest (4,1)-(4,2)-(4,3)-(3,3)-(2,3)-(2,4)-(1,4)-(1,5)-(0,5)
- Robotul poate ieși din labirint în minim 5 secunde (un pas spre vest). Drumul fără mers spre vest necesită 8 secunde.

22. Codare bazată pe frecvență

Descriere: Pentru a coda un mesaj, putem folosi algoritmul următor: înlocuim cu numărul 1 litera care apare de cele mai multe ori în mesaj (are frecvență maximă), înlocuim cu numărul 2 litera care e a 2-a cea mai mare frecventă și așa mai departe. Dacă 2 litere au frecvență egală, le considerăm în ordine alfabetică.

Funcționalități:

- Citiți un mesaj și construiți tabelul cu frecvențe folosind mesajul. Construiți și tabelul de codare (containerul în care pentru fiecare literă aveți numărul cu care va fi codat) și de decodare (containerul în care pentru fiecare număr aveți litera care este codată cu el). Pentru a construi containerul de codare, literele trebuie sortate pe baza frecvențelor. Puteți folosi orice algoritm de sortare, dar trebuie să implementați algoritmul.
- Codați un mesaj.
- Decodați un mesaj.

Exemple:

- Pentru mesajul: ”*pentru mai multe informatii intreaba angajatii din restaurante*” avem următoarele frecvențe: {a:9, b:1, c: 0, d:1, e:5, f:1, g:1, h: 0, i:8, j:1, k:0, l:1, m:3, n:6, o:1, p:1, r:5, s:1, t:7, u:3, v:0, z:0}.
- Codarea literelor:
  - a - 1
  - b - 9
  - c - 18
  - d - 10
  - e - 5
  - f - 11
  - g - 12
  - h - 19
  - i - 2
  - j - 13
  - k - 20
  - l - 14
  - m- 7
  - n - 4
  - o - 15
  - p - 16
  - r - 6
  - s - 17
  - t - 3
  - u - 8
  - v - 21
  - z – 22
- Codarea mesajului ”Drifturi in curtea scolii” va fi: 10 6 2 11 3 8 6 2 2 4 18 8 6 3 5 1 17 18 15 14 2 2
- Decodarea nesajului ”17 18 1 4 10 1 14 14 1 18 1 17 1 6 5 12 1 14 1”va fi: scandallacasaregala

23. Joc cu populația țărilor

Descriere: Pentru a facilita învățarea geografiei vrem să implementăm un joc prin care elevii pot reține mai ușor populația țărilor. Având o *bază de date* în care avem țări și populația lor, la începutul jocului vor fi alese *n* țări la întâmplare și vor fi afișate utilizatorului (într-o ordina aleatoare). Utilizatorul trebuie să le aranjeze într-o ordine crescătoare a populației. Utilizatorul introduce ordinea propusă, iar aplicația îi afișează numărul de perechi de țări consecutive care sunt în ordine bună – doar numărul de perechi, nu și perechile efective. Utilizatorul are maximum 10 încercări, scorul lui este 10 minus numărul de încercări greșite.

Funcționalități:

- Implementați jocul cu 3 niveluri de dificultate: începător (5 țări), avansați (10 țări) și experți (15 țări). La începutul jocului utilizatorul alege dificultatea, iar la final aplicația îi afișează scorul. Nu folosiți operație de sortare deloc.

Exemple:

- Presupunând că avem următoarele țări:
  - România – 20.14 (milioane)
  - Portugalia – 10.45
  - Lituania – 2.87
  - Suedia – 9.90
  - Franta – 66.9
  - Germania – 82.67
  - Ungaria – 9.81
  - Spania – 47.12
- Jucătorul alege varianta începătoare și primește 5 țări, de ex. Franta, Portugalia, Romania, Lituania, Spania
- Jucătorul introduce ordinea: Lituania, Romania, Portugalia, Franta, Spania
- Programul afișează că 2 perechi sunt bune.
- Jucătorul introduce ordinea: Lituania, Romania, Portugalia, Spania, Franta
- Programul afișează că 3 perechi sunt bune.
- Jucătorul introduce ordinea: Lituania, Portugalia, Romania, Spania, Franta
- Programul afișează că ordinea e bună, scorul jucătorului e 8.

24. Portmoneu cu valută

Descriere: Să reprezentăm conținutul unui portmoneu în care avem sume (formate din bancnote) din mai multe valute (de ex. RON, EUR, USD). Presupunem că sumele sunt valori întregi.

Funcționalități:

- Să se adauge o sumă (dintr-o valută existentă deja sau dintr-o valută nouă).
- Să se afișeze suma totală din portmoneu schimbată într-o valută. Pentru fiecare valută existentă se citește cursul de schimb în valuta respectivă și se calculează cât ar fi suma dacă ar fi schimbată în valuta respectivă. Se va afișa suma totală dar conținutul portmoneului nu este modificat.
- Schimbarea unei sume dintr-o valută în alta. Se citește cursul de schimb, valoarea schimbată și valuta în care schimbăm. Valoarea schimbată este ștearsă din suma de la valuta respectivă și se adaugă valoarea schimbată. Dacă valoarea schimbată nu este număr întreg, vom adăuga doar partea întreagă. De exemplu, dacă vrem să schimbăm o bancnotă de 10 EUR în RON la cursul 4.67, vom avea 46 de RON (și vom pierde .70 RON).

Exemple:

- Dacă la un moment dat avem următoarele sume și valute:
  - RON – 142
  - SEK – 1150
  - EUR – 45
  - USD - 255
  - HUF – 15000
- Suma totală din portmoneu în USD este:
  - RON – USD (3.98) – 35 USD
  - SEK – USD (8,36) – 137 USD
  - EUR – USD (0.86) – 52 USD
  - USD – 255
  - HUF - USD (267) – 56 USD
  - Total: 535 USD
- Dacă schimbăm 550 SEK în RON vom avea
  - 600 SEK
  - 261 + 142 = 403 RON

25. Meci de handbal

Descriere: Vrem să reținem golurile date de jucători la un meci de handbal. Numele fiecărui jucător din prima echipa începe cu litera A urmat de numărul jucătorului (de ex. A13), iar numele fiecărui jucător din echipa a 2-a începe cu litera B urmat de numărul jucătorului (de ex. B8). Nu există 2 jucători cu același nume. Într-o echipă pot fi maximum 16 jucători (din care maxim 7 pe teren dar acest lucru nu contează acum). Vrem să reținem pentru fiecare jucător numărul de goluri marcate.

Funcționalități:

- Adăugați la un jucător un număr de goluri. Dacă jucătorul are deja goluri marcate, golurile se adună, altfel se adaugă jucătorul nou.
- Aflați golgheterii pentru cele 2 echipe (jucătorii care au marcat cele mai multe goluri)
- Afișați rezultatul meciului.

Exemple:

- Presupunând că am adăugat următoarele jucători și goluri:
  - A4 – 3
  - A6 – 2
  - A10 – 1
  - A3 – 6
  - B1 – 1
  - B2 – 3
  - B6 – 6
  - B8 – 2
- Golgheterul echipei 1 e jucătorul A3 iar golgheterul echipei 2 e jucătorul B6.
- Rezultatul meciului e: 12 - 12

26. Joc cu complexități:

Descriere: Vrem să implementăm un joc prin care să învățăm mai bine complexitatea pentru niște algoritmi. Pentru început vrem să construim o *bază de date* în care să avem numele unor algoritmi și complexitatea lor (de ex. LSI_adaugareInceput și Teta(1), sau bubbleSort și O(n^2)). Pe parcursul jocului aplicația selectează aleator *N* algoritmi și afișează numele lor. Aplicația afișează și cele *N* complexități pentru acești algoritmi, dar într-o ordine aleatoare. Jucătorul trebuie să dea ca răspuns perechile corecte. Scorul jucătorului este numărul de perechi corecte.

Funcționalități:

- Implementați jocul cu 3 niveluri de dificultate: începător (5 algoritmi), avansați (10 algoritmi) și experți (15 algoritmi). La începutul jocului utilizatorul alege dificultatea, iar la final aplicația îi afișează scorul.

Exemple:

- Dacă în aplicație avem reținut:
  - Interclasare Teta(n+m)
  - SelectionSort Teta(n*n)
  - LSI_adaugaSfarsit Teta(n)
  - LSI_adaugaPozitie O(n)
  - LDI_adaugaSfarsit O(1)
  - Stiva_sterge O(1)
  - VD_adaugaInceput Teta(n)
  - Bubblesort O(n*n)
  - MergeSort Teta(nlog_2n)
- Dacă aplicația afișează 5 algoritmi aleator cu complexități:
  - Stiva_sterge BubbleSort Interclasare LSI_adaugaSfarsit VD_adaugaInceput
  - Teta(n) Teta(n) Teta(n+m) Teta(1) O(n*n)
- Dacă utilizatorul răspunde cu
  - 1 – 4
  - 2 – 1
  - 3 – 3
  - 4 – 2
  - 5 – 5
- El va avea 2 puncte (primul si al 3-lea sunt ok)

27. Numere Romane Speciale

Descriere: Numere romane sunt de fapt niște litere, unde fiecare literă reprezintă o valoare (de exemplu I reprezintă 1, X reprezintă 10, C reprezintă 100, etc.) Vrem să construim propriul nostru sistem de cifre romane în care noi decidem ce litere sunt folosite și ce valori reprezintă fiecare.

Funcționalități:

- Adăugați o pereche literă – valoare în sistemul de cifre romane.
- Afișați toate literele și valorile asociate.
- Calculați ce număr reprezintă un string alcătuit din litere din sistemul nostru. Dacă stringul conține litere care nu au numere asociate nu se poate calcula numărul și un mesaj va fi afișat. Atenție, în sistemul de numere romane, literele sunt plasate în ordine descrescătoare a valorii (CXII este 100 si 10 si 2, adică 112). Dacă avem o valoarea mai mică în fața unei valori mai mari, valoarea mică se scade din cea mare (XC, care e 10, 100 înseamnă 90). Însă nu putem avea 2 valori mai mici în fața unei valori mai mari (XXC nu există). Dacă stringul nu e corect un mesaj va fi afișat.

Exemple:

- Dacă la un moment dat avem următoarele litere și cifre asociate:
  - T – 5
  - H – 8
  - C – 32
  - Q – 40
  - G – 97
  - A – 9
- Stringul GGQ reprezintă numărul 97+97+40 = 234
- Stringul AGCQHHT reprezintă numărul 88 (97-9) + 8 (40-32) + 8 + 8 + 5 = 117

28. Recomandare de filme

Descriere: Utilizatorii pot asocia fiecărui film vizualizat anumite cuvinte-cheie, care reprezintă filmul respectiv. Un film poate avea mai multe cuvinte-cheie asociate, pentru că nu toată lumea folosește aceleași cuvinte. De exemplu, filmul Titanic, poate fi descris prin cuvinte cheie: dragoste, ocean, vapor, iceberg, etc.

Funcționalități:

- Să se adauge un cuvânt cheie pentru un film (filmul poate să existe deja, sau poate să nu existe)
- Să se afișeze toate cuvintele cheie asociate unui film.
- Să se recomande un film unui utilizator. Utilizatorul introduce titlul unui film la care s-a uitat deja și de care i-a plăcut, și aplicația caută filmul care are cele mai multe cuvinte cheie comune cu filmul introdus. Dacă nu există cuvinte cheie comune cu niciun film, sau filmul introdus nu are cuvinte cheie asociate, un mesaj va fi afișat.

Exemple:

- Dacă la un moment dat avea următoarele filme și cuvinte asociate:
  - Titanic – dragoste, ocean, vapor
  - ManInBlack – extraterestru, sci-fi, negru
  - ShawshankRedemption – puscarie, evadare, speranță, crimă
  - TheMatrix – sci-fi, negru, luptă, verde
  - Shrek – verde, animatie, capcaun, magarus
- Dacă utilizatorul introduce filmul TheMatrix, aplicația îi sugerează filmul ManInBlack (2 cuvinte cheie comune, cu Shrek este unul singur)

29. Loteria bonurilor

Descriere: Vrem să implementăm un sistem care ne ajută să ținem evidența bonurilor fiscale adunate pe parcursul unei perioade de timp ca să verificăm mai repede dacă am câștigat la loteria bonurilor. Vrem să reținem pentru fiecare dată calendaristică în care am fost la cumpărături valoarea bonurilor. Valoarea bonurilor se reține ca o valoarea float/double (de ex. 19.21 RON).

Funcționalități:

- Adăugați un bon în sistem (se dă data cumpărăturii și valoarea bonului).
- După extragerea unui bon câștigător verificați dacă ați câștigat. La verificare se ia în considerare doar partea întreagă a valorii (de ex. suma 19.21 se consideră ca și cum ar fi 19).
- Calculați în care zi s-a cheltuit cea mai mare sumă.

Exemple:

- Dacă la un moment dat am adunat următoarele bonuri:
  - 1.11.2017 – 103
  - 5.11.2017 – 4.5
  - 7.11.2017 – 198
  - 2.11.2017 – 52
  - 3.11.2017 – 4.5
  - 3.11.2017 – 26.54
  - 6.11.2017 – 43.8
  - 1.11.2017 – 91.2
  - 2.11.2017 – 3.9
- Dacă se extrage data de 2.11.2017 și suma 3 am câștigat (ultimul bon), dar dacă suma e 4, nu am câștigat.
- Am cheltuit:
  - 1.11.2017 – 194.2
  - 5.11.2017 – 4.5
  - 7.11.2017 - 198
  - 2.11.2017 – 55.9
  - 3.11.2017 – 31.04
  - 6.11.2017 – 43.8
- Cea mai mare sumă a fost cheltuită în 7.11.2017

30. Music Quiz

Descriere: Vrem să construim o *bază de date* cu artiști și melodiile lor și să folosim aceste date pentru un joc Music Quiz. Jocul este compus din 10 ture, în fiecare tură utilizatorul primește o întrebare de forma: *Care melodie a fost cântată de X?* Unde X reprezintă numele unui artist ales aleator din baza de date. Utilizatorul primește 4 variante de răspuns, dintre care una reprezintă titlul unei melodii a artistului respectiv, iar celelalte 3 melodii au fost alese în mod aleator de la alți artiști din baza de date. Utilizatorul introduce A, B, C sau D în funcție de răspunsul ales. La finalul jocului utilizatorul poate să vadă câte răspunsuri corecte a dat.

Funcționalități:

- Adăugați un artist și titlul unei melodii.
- Simulați jocul Music Quiz.

Exemple:

- Dacă avem următoarea bază de date:
  - Pink
    - So What?
    - Stupid Girls
    - Try
    - Just Give Me a Reason
  - Alternosfera
    - Ploile nu vin
    - Epizodia
  - MIKA
    - Grace Kelly
    - Lollipop
    - Big Girls
  - Katy Perry
    - Firework
    - Hot n Cold
    - Dark Horse
  - Ada Milea
    - Femur
    - Grasa
- O intrebare poate fi de exemplu:
  - Cine cântă melodia Big Girls?
    - A. Pink
    - B. Mika
    - C. Katy Perry
    - D. Ada Milea
- Dacă utilizatorul răspunde B primește punct, altfel nu.

31. Morse Code

Descriere: Codul Morse asociază fiecărui literă din alfabet o secvență de maxim 4 caractere, fiecare punct sau liniuță. Din moment ce secvențele asocate sunt de lungimi diferite, o anumită secvență de puncte și linuțe poate fi interpretată în moduri diferite în funcție de unde considerăm că se termină o literă (mai multe detalii la exemplu). Figura de mai jos conține codul Morse pentru fiecare literă (puteți ignora cifrele).

A ·−	J ·−−−	S ···	1 ·−−−−
B −···	K −·−	T −	2 ··−−−
C −·−·	L ·−··	U ··−	3 ···−−
D −··	M −−	V ···−	4 ····−
E ·	N −·	W ·−−	5 ·····
F ··−·	O −−−	X −··−	6 −····
G −−·	P ·−−·	Y −·−−	7 −−···
H ····	Q −−·−	Z −−··	8 −−−··
I ··	R ·−·	0 −−−−−	9 −−−−·

Funcționalități:

- Afișați Codarea Morse pentru un cuvânt dat.
- Citind o serie de cuvinte, afișați cea mai mare submulțime dintre ele care au aceeași reprezentare în cod Morse.

Exemple:

- Dacă citim următoarele 5 cuvinte: ca, nna, abc, nnet, cat reprezentarea lor în codul morse este:
  - ca -.-..-
  - nna -.-..-
  - abc .--...-.-.
  - nnet -.-..-
  - cat -.-..-
- ca, nna și nnet au aceeași reprezentare în cod morse.

32. Harry Potter și Piatra Filozifală

Descriere: Harry Potter vrea să ia piatra filizofală, dar ea este păzită de un monstru lacom care are o punga de bani în care poate să adauge bani sau poate să scoată ultimul ban introdus. Monstrul adoarme și Harry poate să ia piatra dacă în punga de bani are măcar suma X. Pentru a-l ajuta pe Harry, Dumbledore îi dă și lui Harry un sac similar cu sacul monstrului, și o secvență de instrucțiuni. Fiecare instrucțiune e ori ”Harry” (când Harry ia prima monedă din punga și o aruncă spre monstru care îl ia și îl pune în punga lui), sau ”Sterge” (când monstrul scoate ultimul ban pus în sacoșă).

Funcționalități:

- Având valoarea X necesară monstrului, monedele din punga lui Harry, și secvența de instrucțiuni de la Dumbledore, returnați după a câta instrucțiune adoarme monstrul. Dacă niciodată nu adoarme, returnați -1.
- Dacă monstrul își schimbă punga într-una din care poate să ia primul ban introdus (și nu ultimul) după câte instrucțiuni va adormi?

Exemple:

- Dacă Harry are inițial 6 monede: 3, 1, 1, 4, 1, 3, suma necesară monstrului este 7 iar instrucțiunile sunt: Harry, Harry, Harry, Sterge, Sterge, Harry, Harry, Sterge vom avea:
  - Harry aruncă banul 3 spre monstru, care va avea suma 3.
  - Harry aruncă banul 1 spre monstru, care va avea suma 4.
  - Harry aruncă banul 1 spre monstru care va avea suma 5.
  - Monstrul aruncă ultimul ban, are suma 4.
  - Monstrul aruncă ultimul ban, are suma 3.
  - Harry aruncă banul 4 spre monstru, care va avea suma 7 și adoarme.
- Dacă monstrul schimbă sacoșa (dar restul valorilor din exemplul anterior rămân nemodificate)
  - Harry aruncă banul 3 spre monstru, care va avea suma 3.
  - Harry aruncă banul 1 spre monstru, care va avea suma 4.
  - Harry aruncă banul 1 spre monstru care va avea suma 5.
  - Monstrul aruncă primul ban, are suma 2.
  - Monstrul aruncă primul ban, are suma 1.
  - Harry aruncă banul 4 spre monstru, care va avea suma 5.
  - Harry aruncă banul 1 spre monstru, care va avea suma 6.
  - Harry aruncă banul 3 spre monstru, care va avea suma 9 și adoarme.

33. Democrația în pericol

Descriere: Pe o insulă ascunsă, locuitorii aveau un sistem simplu de vot: oricând trebuia luată o decizie toți locuitorii se adunau și votau. Un grup de persoane care vrea să preia puterea a convins locuitorii să modifice regula votului, pe motiv că sunt mult prea mulți locuitori ca să se adune toată lumea într-un singur loc. Astfel ei au propus împărțirea populației în K grupe (grupele pot avea număr diferit de persoane) și se votează separat în ficare grupă. Dacă peste jumătate din membrii grupei votează ”DA”, toată grupa e considerată că a votat ”DA” (și la fel cu voturi NU). După voturi în grupe se numără câte grupe au votat ”DA” și câte grupe au votat ”NU”. Dacă peste jumătate dintre grupe au votat ”DA”, atunci votul final va fi ”DA”.

Funcționalități:

- Având numărul de persoane din fiecare grupă, afișați cât este numărul minim de persoane (și care sunt acele persoane) care trebuie să voteze ”DA”, ca să fie ”DA” răspunsul final.

Exemple:

- Dacă sunt 3 grupe cu 5, 7, 5 persoane, e suficient ca 6 persoane să voteze ”DA” (3 din grupa cu 5 și 3 din grupa cu 5).
- Dacă sunt 8 grupe cu 6, 20, 12, 5, 10, 7, 9, 11 persoane, e suficient ca 22 de persoane să voteze ”DA”