Creați o aplicație C++ cu interfața grafica utilizator pentru gestiunea de produse (id - int, nume - string, tip - string, preț - double). Produsele sunt salvate in fișier (se încarcă la pornirea aplicației, creați un fișier cu minim 10 produse). Cerințe:

1) Aplicația afișează produsele din fișier într-un tabel (QTableView) care are coloanele: id, nume, tip, preț si număr vocale (numărul de vocale din numele produsului). Produsele se prezintă sortat după preț pe tot parcursul rulării aplicației. (**1.5p**)
2) Adăugare de produse noi in fișier, id, nume, tip si preț se colectează din textfield -uri. Numele produsului nu poate fi vid, preț trebuie sa fie intre 1.0 si 100.0; nu putem avea doua produse cu același id. Aplicația validează datele si afișează (toate) problemele folosind QMessageBox (**1.5p**)
3) Folosind un slider (QSlider) pentru preț, utilizatorul poate filtra produsele. Daca valoare curenta a sliderului este 50 se colorează cu roșu toate produsele cu preț <=50. Filtrarea se realizează la fiecare modificare pe slider. (**1.5p**)
4) La pornire, aplicația deschide cate o fereastra pentru fiecare tip de produs existent. Fiecare fereastra are ca titlu tipul si conține un QLabel cu numărul de produse de acest tip existente in aplicație. La adăugarea de noi produse se actualizează numărul afișat pe fereastra corespunzătoare tipului produsului.(**2p**)

1p Of **1.5p** Teste si specificații **1p** Arhitectură

Observații:

Nu se accepta aplicații cu interfață tip consolă.
Daca datele nu sunt citite din fișier se scade 0.5 puncte la fiecare funcționalitate.
Daca nu folosiți QTableView se pierde 1p de la funcționalitatea 1.
Nu se pot folosi proiecte existente (trebuie pornit de la 0), se poate folosi QAssistant.
Pentru datele din fișiere puteți folosi orice format doriți (linie cu linie, csv, etc).

Punctajele de la Teste, specificații si Arhitectură se dau proporțional cu punctajul obținut la funcționalități.

Punctaj funcționalități	2	3	4	5	6	6.5
Punctaj maxim stil, specificații	0.75	1.25	1.75	2	2.25	2.5
Nota finală maximă	3.75	5.25	6.75	8	9.25	10

Creați o aplicație C++ cu interfață grafică utilizator pentru gestiunea de melodii. O melodie are: id:int (unic); titlu:string, artist: string, gen (unu dintre: “pop”,”rock”, “folk”, “disco”)
Melodiile sunt salvate in fișier si sunt încărcate la pornirea aplicației. Creați un fișier text care conține cel puțin 10 melodii. Fereastra principală:

1) Prezinta intr-un tabel de melodii sortate după autor. Coloane: id, titlu, artist, gen **(1p)**
2) Folosiți QTableView si un model custom, tabelul sa prezinte fiecare atribut al melodiei si două coloane adiționale care afișează numărul de melodii care au același autor si numărul de melodii care au același gen cu melodia curentă. Tabelul se actualizează la orice modificare a listei de melodii **(1.5p)**
3) Interfața permite adăugarea de melodii. Datele se colectează din textfielduri, la apăsarea unui buton se salvează o melodie noua. La salvare se generează automat un id unic pentru melodie. Adăugarea melodiei se reflecta atât in fișier cat si in tabel **(1.5p)**
4) Melodiile pot fi șterse selectând o linie in tabel si apăsând butonul delete. Melodia se șterge si din fișier, se actualizează tabelul **(1p)**
5) In cele patru colturi ale ferestrei se vor desena cercuri concentrice corespunzătoare celor 4 genuri posibile de melodii. Desenați un cerc mic in fiecare colt, apoi un număr de cercuri cu același centru dar cu raza din ce in ce mai mare, corespunzător numărului de melodii din fiecare gen in parte. Desenul se actualizează la orice modificare a listei de melodii. **(1.5p)**

1p Of **1.5p** Teste si specificații **1p** Arhitectură

Observații:

Nu se accepta aplicații cu interfață tip consolă.
Daca datele nu sunt citite din fișier se scade 0.5 puncte la fiecare funcționalitate.
Daca nu folosiți QTableView si model custom nu luați puncte la funcționalitatea 2.
Nu se pot folosi proiecte existente (trebuie pornit de la 0), se poate folosi QAssistant.
Pentru datele din fișiere puteți folosi orice format doriți (linie cu linie, csv, etc).
Punctajele de la Teste, specificații si Arhitectură se dau proporțional cu punctajul obținut la funcționalități.

Punctaj funcționalități	2	3	4	5	6	6.5
Punctaj maxim stil, specificații	0.75	1.25	1.75	2	2.25	2.5
Nota finală maximă	3.75	5.25	6.75	8	9.25	10

Creați o aplicație C++ cu interfața grafica utilizator pentru gestiunea melodiilor. Melodia are: id:int (unic); titlu: string, artist: string, rank (int intre 0 si 10). Melodiile sunt salvate in fișier si sunt încărcate la pornirea aplicației. Creați un fișier text care conține cel puțin 10 melodii.
Fereastra principala:

1) Prezinta intr-un tabel lista de melodii sortate după rank. Coloane: id, titlu, artist, rank Melodiile rămân sortate tot timpul.**(1p)**
2) Folosiți QTableView si un model custom, tabelul sa prezinta fiecare atribut al melodiei plus o coloana adiționala care afișează numărul de melodii care au același rank cu melodia curenta. Tabelul se actualizează la orice modificare a listei de melodii **(1.5p)**
3) Interfața permite modificarea rank-ului. Daca se selectează o linie in tabel se încarcă intr-un textfield titlu melodiei si pe un slider se setează rank-ul lui. Utilizatorul poate modifica atât titlul cat si rank-ul si daca apasă pe butonul update melodia se actualizează (actualizarea se reflecta si in fișier) **(1.5p)**
4) Melodiile pot fi șterse selectând o melodie in tabel si apăsând butonul delete (melodia se șterge si din fișier). Daca melodia selectata este ultima melodie a artistului atunci se afișează un mesaj si melodia nu se șterge.**(1p)**
5) Fereastra in partea de jos are desenate 10 bare, înălțimea barelor este proporționala cu numărul de melodii de anumit rank (prima bara corespunde la melodii cu rank 0, a doua la melodii cu rank 1, etc.) Desenul se actualizează la orice modificare a listei de melodii. **(1.5p)**

1p Of **1.5p** Teste si specificații **1p** Arhitectura

Observații:

Nu se accepta aplicații cu interfață tip consolă.
Daca datele nu sunt citite din fișier se scade 0.5 puncte la fiecare funcționalitate.
Daca nu folosiți QTableView si model custom nu luați puncte la funcționalitatea 2.
Nu se pot folosi proiecte existente (trebuie pornit de la 0), se poate folosi QAssistant.
Pentru datele din fișiere puteți folosi orice format doriți (linie cu linie, csv, etc).
Punctajele de la Teste, specificații si Arhitectură se dau proporțional cu punctajul obținut la funcționalități.

Punctaj funcționalități	2	3	4	5	6	6.5
Punctaj maxim stil, specificații	0.75	1.25	1.75	2	2.25	2.5
Nota finală maximă	3.75	5.25	6.75	8	9.25	10

Creați o aplicație C++ cu interfața grafica utilizator pentru gestiunea task-urilor de la o firma de soft (taskul are id – int, descriere – string, programatori – lista de string cu numele programatorilor care au lucrat la task, stare – string, poate fi: open, inprogress, closed). Taskurile sunt salvate într-un fișier text (se încarcă la pornirea aplicației, creați un fișier cu minim 10 taskuri). Funcționalități:

1) La pornire pe fereastra principala, se afișează un tabel (id, descriere, stare, numărul de programatori asignați) cu toate taskurile, sortat după stare. Tabelul rămâne sortat tot timpul, chiar daca se fac acțiuni (adăugare, căutare, etc) pe fereastra. (**1.5p**)
2) Adăugare de taskuri noi în fișier: datele se colectează din textfield -uri si se validează (id unic, descriere ne vid, stare una din cele 3, minim 1 - maxim 4 nume de programatori) (**1p**)
3) Funcționalitate de search după programator. Se introduce un string, la schimbarea textului se filtrează tabelul si se afișează doar acele task-uri care la numele programatorilor conțin stringul dat (**1.5p**)
4) La pornire, aplicația deschide 3 ferestre adiționale corespunzătoare stărilor: open, inprogress, closed. Ferestrele conțin lista taskurilor cu starea corespunzătoare si 3 butoane “Open”, “Inprogress” si “Close” prin care taskul selectat trece in starea dorita. Modificarea de stare se reflecta atât in fișier cat si in toate ferestrele din aplicație.(fereastra principala si cele 3 ferestre adiționale) (**2.5p**)

1p Of **1.5p** Teste si specificații **1p** Arhitectură

Observații:

Nu se accepta aplicații cu interfață tip consolă.
Daca datele nu sunt citite din fișier se scade 0.5 puncte la fiecare funcționalitate.
Daca la funcționalitatea 3 trebuie sa apăs un buton pentru a declanșa căutarea punctajul maxim la aceasta funcționalitate este 0.75p
Nu se pot folosi proiecte existente (trebuie pornit de la 0), se poate folosi QAssistant.
Pentru datele din fișiere puteți folosi orice format doriți (linie cu linie, csv, etc).
Punctajele de la Teste, specificații si Arhitectură se dau proporțional cu punctajul obținut la funcționalități.

Punctaj funcționalități	2	3	4	5	6	6.5
Punctaj maxim stil, specificații	0.75	1.25	1.75	2	2.25	2.5
Nota finală maximă	3.75	5.25	6.75	8	9.25	10

Creați o aplicație C++ care implementează jocul TicTacToe (X0). Aplicația gestionează mai multe jocuri, un joc are un **id**, **dim** - dimensiune tabla, o **tabla** de joc (Pentru dim=3 un string de 9 caractere de ‚X',O','-'- cate pătrate am pe tabla 3x3), **jucătorul** care urmează sa mute (X sau O) si **starea** jocului: „Neînceput, „In derulare", „Terminat". Tablele de joc sunt salvate in fișier (Exemplu de linie in fișier: 1 3 X-OXO-XOO X Terminat). Funcționalități:

1) La pornire pe fereastra principală, se afișează un tabel (QTableView) cu toate jocurile salvate, sortat după stare. Tabelul rămâne sortat tot timpul. (**1.5p**)

2) Creare joc: Se introduc (in textfield-uri) dimensiunea, dim x dim caractere (tabla de joc) si X sau O). Jocul se adaugă in fișier cu starea „Neinceput" (**1p**)

3) Utilizatorul poate modifica un joc existent. Datele trebuie validate: dim poate fi 3,4,5; tabla un string cu exact dim x dim caractere, doar caractere ‚X',O' sau '-'; jucătorul care urmează poate fi,X' sau ‚O' starea poate fi „Neinceput", „In derulare" „Terminat". Modificările se reflecta in fișier (**1.5p**)

4) Creează o tabla de joc (dim x dim butoane, dim linii dim coloane). La selectarea unui joc existent se modifica textul de pe butoane pentru a reflecta starea jocului (‚X',O',' ') (**1p**)

5) Aplicația permite desfășurarea unui joc. La apăsarea unui buton (doar daca reprezintă o căsuța liberă) se pune semnul jucătorului curent si se schimba jucătorul curent (din X in O respectiv din O in X). După fiecare mutare starea jocului este actualizat in fișier (**1.5p**)

1p Of **1.5p** Teste si specificații **1p** Arhitectură

Observații:

Nu se accepta aplicații cu interfață tip consolă.
Daca datele nu sunt citite din fișier se scade 0.5 puncte la fiecare funcționalitate.
Daca nu folosiți QTableView se pierde 1p de la funcționalitatea 1.
Nu se pot folosi proiecte existente (trebuie pornit de la 0), se poate folosi QAssistant.
Pentru datele din fișiere puteți folosi orice format doriți (linie cu linie, csv, etc).
Punctajele de la Teste, specificații si Arhitectură se dau proporțional cu punctajul obținut la funcționalități.

Punctaj funcționalități	2	3	4	5	6	6.5
Punctaj maxim stil, specificații	0.75	1.25	1.75	2	2.25	2.5
Nota finală maximă	3.75	5.25	6.75	8	9.25	10