

Proiect OOP



Simulare și administrare service pentru electrocasnice

I. Descriere

FixItNow este un service autorizat pentru repararea electrocasnicelor. În cadrul sau, angajați specializați (tehnicienii) se ocupă de repararea aparatelor, recepționează preiau și înregistrează cererile de reparatii din partea clienților, iar supervisorul se ocupa cu raportările.

Se dorește dezvoltarea unei aplicații orientate pe obiecte în C++ care să :

- administreze angajații, electrocasnicele și cererile de reparație
- simuleze procesului de alocare a cererilor și realizarea reparațiilor în timp real
- să genereze o serie de raportări

Echipa service-ului include trei tipuri de angajați. Un angajat are ID unic – atribuit la angajare și nemodificabil, nume și prenume – de minim 3 și maxim 30 caractere fiecare, CNP valid – conform specificațiilor oficiale*, data angajării, oraș de domiciliu.

*Validare CNP [https://ro.wikipedia.org/wiki/Cod_numeric_personal_\(Rom%C3%A2nia\)](https://ro.wikipedia.org/wiki/Cod_numeric_personal_(Rom%C3%A2nia))

Pe lângă acestea: recepționează are o listă cu ID-urile cererilor de reparații înregistrate de el în sistem, tehnicianul are asociată o listă de tipuri și mărci de electrocasnice pe care le poate repara (e.g. poate repara Televizoare LG și Frigidere Samsung), supervisorul nu are attribute suplimentare.

Salariul fiecărui angajat se calculează pornind de la un salariu de baza (4000 RON) la care se adaugă bonus de fidelitate (+5% din salariul de bază la fiecare 3 ani lucrați la service). Tehnicienii primesc bonus 2% din valoare reparațiilor efectuate. Prima de transport de 400 RON se acordă tuturor angajaților cu domiciliul în afara orașului București. Supervisorul are spor de conducere de 20% din salariul de bază.

Există posibilitatea ca, de-a lungul timpului să se facă angajări. O persoana poate să fie angajată dacă la data angajării are cel puțin 16 ani. Trebuie să existe posibilitatea modificării numelui unei persoane (e.g. dacă s-a căsătorit). Există posibilitatea ca un angajat să vrea să își dea demisia, deci va trebui să dispară din evidența service-ului. Oricând este solicitat, trebuie să fie posibilă afișarea datelor unui angajat și a salariului acestuia în luna curentă.

Service-ul repară mai multe tipuri de **electrocasnice**: frigidere, TV-uri, mașini de spălat. Pentru fiecare electrocasnic se cunosc mărcile (e.g. LG, Samsung, Arctic, etc) și modelele care pot fi reparate în service (e.g. frigider Samsung – modele reparabile: Frost200 , CoolMax). Despre fiecare

electrocasnic care poate să fie reparat se mai știe, deci, tipul, marca și modelul precum și anul de fabricație, prețul de catalog. Considerați că un anumit model are asociat doar un an de fabricație. Se pot adăuga / șterge oricând mărci și modele care pot să fie reparate.

În plus fiecare tip de electrocasnice are propriile specificități: frigiderul are sau nu congelator, TV-ul ar dimensiunea diagonalei reținută fie în cm fie în inci, despre mașina de spălat se știe capacitatea acesteia (în kg).

O cerere de reparație conține:

- ID unic
- model și marcă aparat de reparat și toate celelalte detalii complete despre electrocasnic, dacă acesta poate să fie reparat
- timestamp depunere cerere (data, ora complete)
- nivel de complexitate problemă de rezolvat (1–5, afectează durata reparației; 0 dacă nu poate fi reparat)
- durată estimată de reparație în unități de timp (calculată pe baza complexității și vechimii în ani în funcție de anul de fabricație după formula: $\text{durata} = \text{vechime} * \text{complexitate}$).
- preț reparație (estimat ca: preț catalog electrocasnic * durată estimată reparație).

Nu exista 2 cereri cu același timestamp.

Reguli legate de cererile de reparație:

Recepționarul înregistrează cererea și o adaugă în lista sa.

Dacă cererea e validă (tipul și marca de electrocasnic apare în lista service-ului) se repartizează automat unui tehnician după criteriile:

1. tehnicianul trebuie să fie specializat pe acel tip și marcă
2. are mai puțin de 3 cereri active
3. de-a lungul timpului, tehnicienii trebuie să aibă relativ aceeași încărcare (din punct de vedere a duratei de muncă depuse).

Dacă nu există tehnicieni eligibili, cererea se plasează în așteptare.

Cererile se asignează în ordinea timestampului.

Simularea asignării de cereri și simularea reparațiilor se face în timp: la fiecare tic (unitate de timp aleasă de voi – ideal secunde), aplicația:

- procesează cererile active (reduce durată rămasă),
- finalizează cererile pentru care s-a terminat reparația,
- încearcă să aloce cereri valide din lista de așteptare sau care acum au fost citite.

Trebuie afișate mesaje de status la fiecare tic care sa spună dacă o cerere e asignată sau finalizată, respectiv, ce tehnician lucrează la ce reparație. Aici este dat un exemplu orientativ:

[Timp 1] Tehnician 1 primește cererea cu id 1

Tehnician 2 primește cererea cu id 2

..

[Timp 12] Tehnician 11 procesează cererea cu id 43 (rămân 13 unități de timp)

Tehnician 12 procesează cererea cu id 44 (rămân 1 unități de timp)

Cereri în așteptare: 45, 46, 47.

[Timp 13] Tehnician 11 procesează cererea cu id 43 (rămân 12 unități de timp)

Tehnician 12 finalizează cererea 44

Tehnician 12 primește cererea 45

Cereri în așteptare: 46, 47.

II. Funcționalități minimale

Service-ul trebuie să poată să permită gestiunea angajaților, electrocasnicelor pe care le poate repara și a cererilor de reparații. Dacă service-ul nu are: minim 3 tehnicieni, un recepționer și un supervisor, nu poate funcționa și se afișează un mesaj în acest sens.

1. Gestiune angajați

- adăugare, modificare, ștergere angajat
- afișare date și calcul salariu curent angajat (căutat după CNP)
- afișare listă angajați

2. Gestiune electrocasnice

- adăugare/ștergere modele/mărci care pot să fie reparate și datele tehnice aferente
- afișarea tuturor aparatelor reparate cu toate detaliile specifice
- afișarea tuturor aparatelor (model și marcă) care au apărut în cereri și nu au putut să fie reparate – dacă același tip de aparat apare de mai multe ori se afișează numărul de apariții. Datele afișate sunt sortate descrescător în funcție de numărul de apariții.

3. Procesare cereri

- simulare primire cereri (citire din fișiere) și stocare cereri invalide (pentru electrocasnice care nu pot fi reparate de service)

- alocarea automată a cererilor valide
- simularea reparațiilor - în timp real, cu afișarea cererilor asignate, în curs, finalizate, în așteptare.

4. Raportări

- Top 3 angajați cu cel mai mare salariu în luna curentă – ordonați în funcție de nume și prenume.
- Toate datele tehnicianul cu cea mai de durată reparație.
- Cererile încă în așteptare grupate pe tipuri de electrocasnice, mărci și modele (sortate alfabetic).

Pentru fiecare cerință se generează un raport în format .csv.

5. Cerințe privind Testarea

Pentru validarea corectă a funcționării aplicației, trebuie să includeți în proiect fișiere de test, organizate într-un folder dedicat (tests/), care să verifice situații reale de utilizare, precum și cazuri de eroare. Fiecare test va fi construit sub formă de fișier .csv/.txt (sau alt format structurat), și va conține date pentru: angajați valizi (ex. recepționeri, tehnicieni, supervizori cu CNP valid etc), angajați invalizi (ex. CNP invalid, etc), cererile valide de reparație (ex. modele cunoscute), cererile incorecte (ex. model nerecunoscut, etc), scenarii diferite care să testeze simularea asignării de cereri și realizării de reparații și care să permită verificarea tuturor cazurilor limită și a încărcării relativ egale a tehnicienilor.

Conținutul și rolul tuturor fișierelor va fi explicat într-un fișier README.md pus în acest folder.

Fiecare linie invalidă trebuie identificată și raportată cu un mesaj clar distinct, de exemplu: Eroare la citire: Angajat invalid pe linia 3, cauza: CNP invalid (12345)

Programul nu trebuie să se încheie la întâlnirea unei erori, ci să continue procesarea liniilor valide, să raporteze toate problemele detectate și să realizeze cerințele dacă e posibil.

Testele vor fi folosite atât pentru evaluarea automată, cât și pentru demonstrarea funcționalităților în prezentarea finală.

6. Documentație tehnică

- maxim 4 pagini

- explică succint arhitectura aplicației - clasele folosite și relațiile dintre ele (ideal sub forma unei diagrame), modul de funcționare al aplicației, elemente tehnice noi/interesante de care v-ați lovit, respectiv, descrierea testelor făcute.

7. Alte cerințe

Pentru a realiza partea de management a service-ului trebuie să folosiți un meniu din care utilizatorul alege ce acțiune dorește să facă: gestiune angajați, gestiune electrocasnice, procesare cereri, raportare, cu submeniurile aferente. Dați mesaje clare când o operație nu poate să fie realizată și precizați care e cauza problemei.

Codul trebuie organizat în fișiere .h/.cpp pentru fiecare clasă.

Trebuie să folosiți biblioteci existente pentru dată/oră (cel puțin ca să generați timestampuri). Puteți considera că o unitate de timp pentru durata de reparație = o secundă.

Folosiți clasa string în loc de char* sau char[] acolo unde se poate/are sens.

Trebuie să folosiți structuri de date (STL) adecvate situației (vector, listă, coadă, etc când este cazul) – vezi C11 și elemente de Modern C++ (inițializare uniformă, lambda, smart pointers etc - vezi 12).

Respectați toate principiile OOP, SOLID învățate.

Nerespectarea acestor cerințe conduce la depunțare.

III. Punctare

- Gestiune angajați cu meniu și teste – 25 p
- Gestiune electrocasnice care pot fi reparate de service cu meniu și teste – 25 p
- Gestiune cereri cu teste – 10 p
- Simulare asignare cereri, dinamică reparații, cereri în așteptare cu teste – 30 p
- Raportare – 10 p (se obține punctajul doar dacă simularea pentru procesarea de cereri funcționează).

- Documentație tehnică – 10 p

_____110p

- *Design patterns minim: Factory si Singelton – 10p bonus*

_____120p

Punctajul se obține în urma prezentării aplicației și a răspunsurilor la întrebările pe care vi le voi pune. Prezentarea o să fie de aprox. 7 min/pers așa că organizați-vă cât mai eficient datele pentru teste.

Punctajul integral se obține DOAR DACĂ:

- toate cazurile limită sunt testate și aceasta se poate observa în prezentarea pe care o faceți
- sunt folosite structuri de date (STL) adecvate fiecărei situații

- maniera de organizare și scriere a codului este una ordonată și lizibilă, fiecare clasa e implementată în fișiere distincte
- sunt utilizate cât mai multe elemente de Modern C++ (acolo unde are sens).
- se respecta toate principiile OOP si SOLID.

IV. Deadline și detalii legate de încărcarea și prezentarea proiectului

Deadline hard: duminica 11.01.2026 23:59

Bonus: +20% din punctajul obținut, dacă proiectul final e încărcat înainte de 01.01.2026.

Se încarcă pe Moodle doar fișierele .h și .cpp, folderul cu date de test și un fișier (Popescu_Ion_321_a.pdf) pentru documentație, care trebuie să fie cât mai concisă. Pentru simplitate faceți o arhiva Popescu_Ion_321_a.7z . Orice proiect care nu respectă cerințele de încărcare nu va fi evaluat.

În săptămânile 13 si 14, fiecare student, la semigrupa de care aparține, va prezenta proiectul (ideal de pe laptopul personal) – 7 min/prezentare.

Proiectele o sa fie testate pentru similitudini folosind Moss. Temele cu similitudini o să fie punctate cu **-10p** din nota finală (atât tema „sursă” cât și tema „destinație”).