**Proiect la PCLP II**

**Profesor Coordonator :**

Giorgian Neculoiu

Gabriela Olteanu

**Studenti :**

Besciu Cristian-Darius

Lite Karina Stefania

Cuprins

1.Introducere.............................................................................. 3 1.1Motivarea alegerii temei ………………………………………………. 3

1.2 Despre Minesweeper ………………………………………………....... 3

2.Concepte și tehnologii folosite în elaborarea proiectului …...5

2.1 Visual Studio Code …………………………………………………… 5

2.2 C++ ……………………………………………………………………6

3.Scenariu de functionare ……………………………………..7

3.1 Scurta prezentare ……………………………………………………….7

3.2 Prezentare detaliata …………………………………………………….8

4.Concluzie …………………………………………………..15

5.Bibliografie …………………………………………………16

1. Introducere

1.1 Motivarea alegerii temei

Am decis să cream o versiune a jocului Minesweeper deoarece acesta excelează la un mix de logică, strategii și programare. Este un joc clasic fiind cunoscut de majoritatea oamenilor ce utilizează un calculator și deși pare simplu la prima vedere, necesită o gândire analitică și o bună înțelegere a regulilor pentru a fi eficientîn rezolvare. De asemenea, este un joc care te duce cu gândul înapoi deoareceacesta a fost pus pe Windows foarte mulți ani la rând și pentru mulți a însemnat primul joc la care au avut acces. Această latură nostalgică a contribuit considerabil la decizia de a folosi acest joc ca temă.

1.2 Despre Minesweeper



Minesweeper este un joc video clasic care a cunoscut o popularitate uriașă odată cu includerea sa în sistemele de operare Microsoft Windows, începând cu versiunea Windows 3.1 din anul 1992. Deși mulți îl asociază exclusiv cu Windows, conceptul jocului este mult mai vechi, rădăcinile sale întinzându-se până în anii ’60 și ’70, când primele variante de jocuri de tip "mine" au fost dezvoltate pe mainframe-uri și în limbaje simple, precum BASIC.

Forma modernă a Minesweeper-ului a început să prindă contur în anii ’80, odată cu apariția unor variante rudimentare pe diferite sisteme de operare timpurii, precum Unix sau DOS. Totuși, versiunea creată de Curt Johnson și Robert Donner pentru Microsoft a fost cea care a consacrat jocul, datorită distribuirii gratuite împreună cu sistemul de operare Windows. Scopul său inițial, dincolo de divertisment, era și unul educativ: familiarizarea utilizatorilor cu utilizarea mouse-ului, în special a butonului din dreapta.

Succesul jocului a fost imediat, devenind o prezență constantă pe calculatoarele personale din întreaga lume. Interfața simplă, dar provocarea logică ridicată, l-au transformat într-un mod rapid și accesibil de a petrece timpul sau de a testa gândirea strategică. De-a lungul anilor, Minesweeper a fost reimaginat în diverse versiuni și stiluri grafice, atât oficiale cât și dezvoltate de comunitatea open-source, menținându-și farmecul în ciuda evoluției tehnologice.

Astăzi, Minesweeper este considerat un simbol al erei timpurii a calculatoarelor personale. Este adesea menționat cu nostalgie de cei care l-au descoperit în copilărie sau în primii ani ai familiarizării cu Windows. Deși aparent simplu, el rămâne un exemplu de design de joc eficient și atemporal, fiind recreat frecvent ca exercițiu de programare sau ca omagiu adus epocii de aur a software-ului de birou.

1. Concepte și tehnologii folosite în elaborarea proiectului
   1. Visual Studio Code

A logo with blue letters

AI-generated content may be incorrect.

Visual Studio Code (prescurtat VS Code) este un editor de cod sursă dezvoltat de Microsoft, lansat oficial în anul 2015. Este gratuit, open-source și cross-platform, ceea ce înseamnă că poate fi utilizat pe Windows, macOS și Linux. VS Code s-a impus rapid ca unul dintre cele mai populare editoare de cod din lume datorită simplității sale, performanței excelente și ecosistemului extins de extensii.

Editorul oferă suport încorporat pentru un număr mare de limbaje de programare, cum ar fi JavaScript, Python, C++, Java, HTML/CSS și altele. În plus, prin extensii din marketplace, utilizatorii pot adăuga suport pentru tehnologii și limbaje suplimentare, pot integra debuggere, terminale, controlul versiunilor (Git) sau chiar medii de dezvoltare virtuale.

Visual Studio Code este folosit frecvent în mediul universitar, mai ales în cadrul facultăților cu profil informatic. Studenții îl utilizează pentru scrierea și testarea codului în diverse limbaje de programare, fiind preferat pentru interfața intuitivă, instalarea rapidă și compatibilitatea cu platforme educaționale. De asemenea, multe cursuri și laboratoare recomandă sau solicită folosirea acestui editor, datorită integrării sale eficiente cu GitHub, terminalul integrat și suportul pentru debugging, facilitând astfel procesul de învățare și dezvoltare.

2.2 C++



C++ este un limbaj de programare de nivel înalt, dezvoltat inițial de Bjarne Stroustrup la începutul anilor 1980, ca o extensie a limbajului C. A fost creat cu scopul de a oferi programatorilor un limbaj care să combine eficiența și controlul specific C-ului cu facilități moderne de programare orientată pe obiecte. De-a lungul timpului, C++ a evoluat semnificativ, ajungând să fie un limbaj puternic, flexibil și utilizat într-o gamă foarte largă de domenii, de la sisteme de operare și software embedded până la jocuri video, aplicații financiare și simulări științifice.

C++ oferă suport atât pentru programarea procedurală, cât și pentru cea orientată pe obiect, fiind astfel potrivit pentru proiecte de orice dimensiune. Limbajul permite un control detaliat asupra resurselor sistemului (memorie, procesor), ceea ce îl face foarte eficient din punct de vedere al performanței, dar presupune și o atenție sporită din partea programatorului. Standardele moderne ale limbajului, precum C++11, C++14, C++17 și C++20, au adus numeroase îmbunătățiri, inclusiv în ceea ce privește expresivitatea și siguranța codului.

În mediul universitar, C++ este adesea unul dintre primele limbaje predate în cadrul programelor de informatică sau inginerie. Este considerat ideal pentru învățarea conceptelor fundamentale ale programării, precum structuri de control, tipuri de date, pointeri, clase și moștenire. De asemenea, oferă studenților o înțelegere profundă asupra modului în care funcționează computerele la un nivel mai apropiat de hardware, în comparație cu alte limbaje de nivel mai înalt. Prin proiecte, laboratoare și examene practice, studenții deprind atât noțiuni teoretice, cât și abilități concrete de programare în C++.

3. Scenariu de functionare

3.1 Scurtă prezentare

Acest program reprezintă o implementare simplă, dar funcțională, a jocului Minesweeper (în română, „Săpătorul de mine”), dezvoltată în limbajul C++.

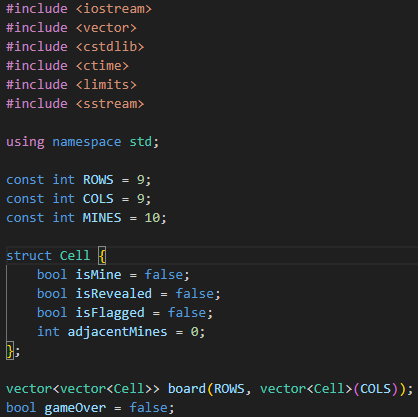
Jocul se desfășoară într-o interfață de tip consolă și folosește o tablă de joc fixă, de dimensiuni 9x9, pe care sunt plasate aleatoriu 10 mine. Scopul jucătorului este să descopere toate celulele care nu conțin mine, evitând în același timp să declanșeze vreuna dintre acestea.

La fiecare pas, utilizatorul poate introduce o comandă pentru a dezvălui o celulă sau pentru a marca o celulă suspectă cu un steag, semnalizând că presupune că acolo se află o mină. După prima mutare (care este întotdeauna sigură), minele sunt generate pe tablă, iar fiecărei celule i se calculează numărul de mine adiacente. Dacă o celulă fără mine și fără vecini cu mine este dezvăluită, se declanșează o revelație recursivă, care extinde zona sigură. Jocul se termină fie atunci când jucătorul dezvăluie o mină (caz în care pierde), fie când toate celulele sigure sunt descoperite (caz în care câștigă). La finalul fiecărei runde, utilizatorul are opțiunea de a începe o nouă partidă.

Programul se folosește de concepte fundamentale din programare, precum structuri, vectori bidimensionali, recursivitate, validare a inputului și generare aleatorie.

3.2 Prezentare detaliată

1. Inițializarea jocului și definirea structurii datelor



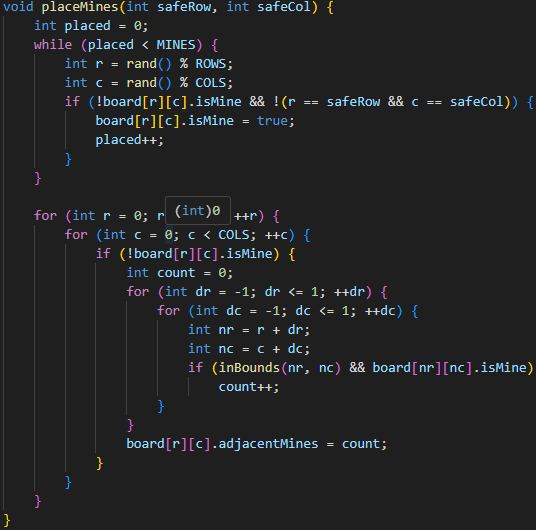
Programul începe prin includerea bibliotecilor necesare pentru funcționalități precum input/output, lucrul cu vectori, generare aleatorie și procesarea comenzilor din consolă. Se folosesc trei constante **(ROWS, COLS, MINES)** pentru a seta dimensiunea tablei (9x9) și numărul de mine (10). Structura Cell definește starea fiecărei celule de pe tablă, incluzând informații despre prezența unei mine, dacă a fost dezvăluită sau marcată, și câte mine are în vecinătate. Tabla de joc este reprezentată de un vector bidimensional de celule, inițializat la începutul jocului. De asemenea, variabila globală gameOver este utilizată pentru a controla starea generală a jocului – dacă s-a pierdut sau nu.

2. Verificarea limitelor tablei



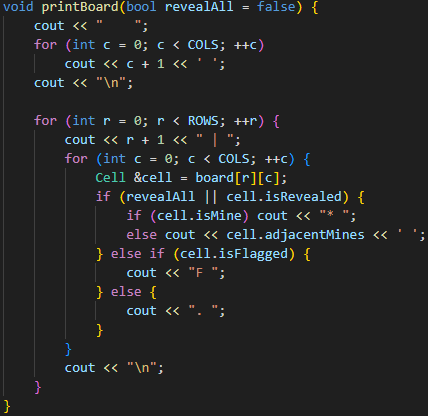
Funcția **inBounds** are rolul de a verifica dacă o anumită poziție (linie r, coloană c) se află în interiorul limitelor tablei de joc. Returnează true doar dacă atât linia, cât și coloana sunt în intervalul valid (0 până la 8, corespunzător dimensiunii 9x9). Această verificare este esențială pentru a preveni accesul în afara vectorului bidimensional, evitând astfel erori de execuție sau comportamente neprevăzute în timpul parcurgerii celulelor vecine.

3. Plasarea minelor și calculul numerelor adiacente



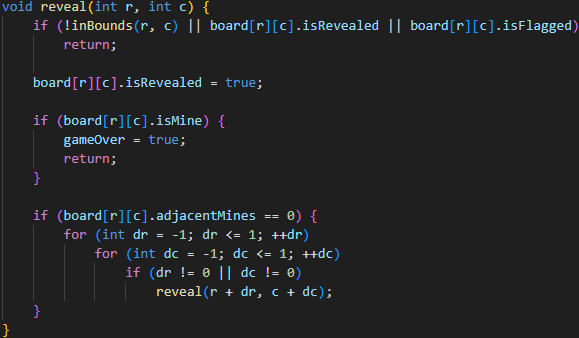
Funcția **placeMines** are rolul de a plasa aleatoriu cele 10 mine pe tablă, asigurând că prima celulă selectată de jucător (safeRow, safeCol) nu va conține niciodată o mină. Se generează poziții aleatorii până când toate minele sunt plasate, evitând dublarea plasării în aceeași celulă. După ce toate minele au fost amplasate, funcția parcurge întreaga tablă și calculează pentru fiecare celulă (care nu conține mină) numărul de mine adiacente. Acest număr este stocat în atributul adjacentMines al celulei și va fi afișat jucătorului pentru a-l ajuta să deducă unde ar putea fi ascunse celelalte mine.

4. Afișarea grafică a tablei de joc în consolă



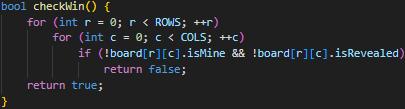
Funcția **printBoard** afișează în consolă starea actuală a tablei de joc într-un format lizibil. Coloanele și liniile sunt numerotate pentru ușurința identificării celulelor. Pentru fiecare celulă, se afișează simbolul potrivit în funcție de starea ei: dacă **revealAll** este activat sau celula este dezvăluită, se afișează un asterisc (**„\***”) dacă este mină, sau numărul de mine adiacente dacă este o celulă sigură; dacă celula este marcată cu steag (isFlagged), se afișează „F”, iar în rest se afișează un punct („**.”)** care indică o celulă ascunsă. Această funcție este folosită pentru a oferi feedback vizual jucătorului pe parcursul jocului și la final .

5. Dezvăluirea celulelor și propagarea în vecinătate



Funcția **reveal** dezvăluie o celulă selectată de jucător la poziția indicată (r, c), cu verificări pentru limitele tablei și starea celulei. Dacă celula este deja dezvăluită sau marcată cu steag, funcția nu face nimic. Dacă celula conține o mină, variabila globală **gameOver** este setată pe true, semnalând sfârșitul jocului. În cazul în care celula nu are mine în vecinătate (adjacentMines egal cu 0), funcția declanșează o revelație recursivă pentru toate celulele adiacente, extinzând automat zona dezvăluită, facilitând astfel explorarea rapidă a spațiilor sigure.

6. Verificarea condiției de câștig



Funcția **checkWin** verifică dacă jucătorul a câștigat jocul prin parcurgerea întregii table și confirmarea că toate celulele care nu conțin mine au fost dezvăluite. Dacă există măcar o celulă sigură încă ascunsă, funcția returnează false. Dacă toate celulele sigure sunt dezvăluite, aceasta returnează true, semnalând astfel victoria jucătorului.

7. Controlul principal al jocului și interacțiunea cu utilizatorul

Funcția main coordonează întreaga logică a jocului Minesweeper.

Structura sa principală este un **loop do-while**, care permite jucătorului să joace de câte ori dorește, la finalul fiecărei partide fiind întrebat dacă dorește să continue.

A screen shot of a computer program

AI-generated content may be incorrect.

A screen shot of a computer program

AI-generated content may be incorrect.A screen shot of a computer program

AI-generated content may be incorrect.

La începutul execuției, se apelează **srand(time(0))** pentru a inițializa generatorul de numere aleatoare, esențial pentru plasarea aleatorie a minelor pe tablă. Apoi, în interiorul buclei do, se reinitializează tabla de joc (variabila globală board) cu o grilă nouă de celule (Cell), toate în stare neutră, și se setează **gameOver** pe false.

O variabilă **firstMove** este marcată cu true pentru a garanta că mina nu este plasată pe prima celulă selectată de jucător — astfel se evită pierderea jocului din prima mutare, o funcționalitate comună în variantele moderne de Minesweeper.

**Ciclu principal al jocului**

Cât timp jocul nu este terminat (!gameOver), funcția:

1. **Afișează tabla** curentă prin printBoard(), arătând celulele revelate, cele marcate cu steag și pe cele ascunse.
2. Solicită o **comandă de la utilizator**, care poate fi de forma:
   * r <linie><coloana> – pentru a dezvălui o celulă.
   * f <linie><coloana> – pentru a pune sau elimina un steag pe o celulă.
3. Comanda este citită ca un șir și analizată folosind un stringstream pentru a extrage tipul comenzii și coordonatele.
4. Se verifică dacă poziția este validă (inBounds), iar dacă nu, este afișat un mesaj de eroare.
5. În funcție de comanda dată:
   * Dacă utilizatorul alege f, iar celula nu a fost deja revelată, se inversează starea isFlagged.
   * Dacă utilizatorul alege r (reveal) și este prima mutare, se apelează placeMines() pentru a plasa minele aleatoriu, cu excluderea celulei alese.
   * Apoi, se apelează funcția reveal() care:
     + Dezvăluie celula aleasă.
     + În cazul în care conține o mină, jocul se termină (gameOver = true).
     + Dacă are 0 mine adiacente, recursiv dezvăluie celulele vecine.
6. După fiecare mutare de tip reveal, se verifică dacă toate celulele non-mină au fost descoperite, caz în care jucătorul câștigă **(checkWin()).**

**Sfârșitul jocului**

Dacă jucătorul pierde (apasă pe o mină) sau câștigă (toate celulele sigure sunt revelate), se afișează tabla completă cu toate minele și se oferă un mesaj corespunzător. Ulterior, jucătorul este întrebat dacă dorește să mai joace o rundă.

Dacă răspunsul este y sau Y, jocul reîncepe cu o tablă nouă. În caz contrar, aplicația se închide cu return 0.

4. Concluzie

Proiectul dezvoltat ( implementarea jocului *Minesweeper* în limbajul C++ ) a reprezentat un exercițiu complet de aplicare a conceptelor fundamentale de programare in limbajul C ++. Scopul principal a fost recrearea funcționalității esențiale a jocului clasic într-un mod intuitiv, stabil și ușor de utilizat din consolă, asigurând o experiență de joc predictibilă și corectă, respectând totodată principiile de proiectare logică și organizarea clară pe module a codului.

Pe parcursul dezvoltării, au fost utilizate structuri de date eficiente (precum vectori bidimensionali pentru reprezentarea tablei de joc) și concepte fundamentale precum recursivitatea (în procesul de dezvăluire automată a celulelor fără mine adiacente), validarea inputului, generarea aleatorie de date (plasarea minelor) și controlul ciclurilor de joc. Toate acestea au fost gestionate într-un mod clar și coerent, asigurând o experiență de joc predictibilă și corectă.

Funcționalitățile cheie ale aplicației includ:

* Plasarea sigură și aleatorie a minelor, cu evitarea primei poziții selectate;
* Afișarea în timp real a tablei de joc, cu marcarea celulelor revelate, a celor marcate cu steag și a celor ascunse;
* Posibilitatea de a marca sau demarca o celulă suspectată ca mină;
* Detecția automată a finalului jocului, fie prin pierdere (revelarea unei mine), fie prin câștig (dezvăluirea tuturor celulelor sigure);
* Posibilitatea de reluare a jocului în mod repetat, după dorința utilizatorului.

Din punct de vedere educațional, realizarea jocului ne - a oferit o platformă excelentă pentru consolidarea cunoștințelor legate de programare C++, precum și o oportunitate de a înțelege mai bine modul în care algoritmii și structurile de date pot fi aplicate în contexte reale și interactive. În plus, am pus accent pe claritatea codului si organizarea logică a funcțiilor.

În concluzie, această aplicație demonstrează nu doar capacitatea de a reproduce un joc clasic într-un mediu de programare de bază, ci și abilitatea de a crea o interfață funcțională și accesibilă într-un mediu non-grafic. Jocul Minesweeper implementat poate fi extins cu ușurință în viitor – prin adăugarea unei interfețe grafice, a unor niveluri de dificultate variabile sau a unui sistem de scor – constituind astfel o bază solidă pentru proiecte viitoare de complexitate mai mare.

5. Bibliografie

<https://en.cppreference.com/w/>

<https://cplusplus.com/doc/tutorial/>

<https://cplusplus.com/reference/vector/vector/>

<https://www.geeksforgeeks.org/introduction-to-recursion-2/>

<https://cplusplus.com/reference/cstdlib/rand/>