Minesweeper

**Lucrare pentru atestarea competentelor profesionale**

**realizata de Mercas Stefan-Alexandru,**

**elev al Liceului Teoretic de Informatica “Grigore Moisil”,**

**sub indrumarea Prof. Lacramioara Tufescu**

Cuprins:

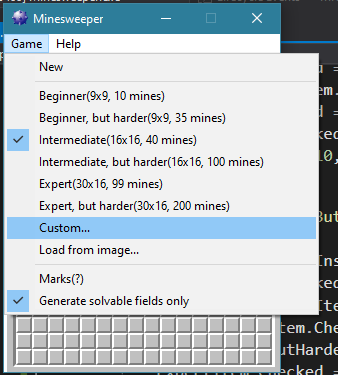
1. **Motivul alegerii temei**
2. **Structura aplicatiei**
3. **Detalii tehnice si Limbajul utilizat**
4. **Evenimente Importante**
5. **Algoritmica in spatele aplicatiei**
6. **O alta posibila modalitate de rezolvare a unui camp**
7. **Motivul alegerii temei**

In primii ani in care am avut propriul meu PC nu detineam fondurile necesare pentru a cumpara jocuri si, daca intentionam sa imi utilizez computerul in scopuri recreative, aveam doar cateva optiuni, printre care se aflau si jocurile incluse in Windows 7. Intr-o zi, in timp ce incercam mai multe din acele aplicatii, mi-am descoperit prima pasiune, Minesweeper. Astfel, pentru urmatorii cativa ani, mi-am ocupat timpul liber cu acest joc. Cand eram in fata monitorului, ma jucam Minesweeper, iar cand nu eram, ma gandeam la Minesweeper, insa, intr-o zi, din motive necunoscute, m-am oprit.

Sarind peste cativa ani, am redescoperit jocui in 2019 si am inceput sa ma gandesc la logica din spatele acestuia. Cand am aflat despre algoritmul lui Lee in clasa a 10-a, m-am gandit la modul in care ar putea sa fie implementat in descoperirea spatiilor libere dintr-un joc de Minesweeper, dupa ce am inteles recursivitatea, m-am gandit la metode de eliminare a probabilitatii din acesta, iar, la inceputul clasei a 12-a, mi-am propus sa-mi pun in aplicare ideile prin aceasta lucrare.

Am ales aceasta tema din atat de multe motive. Vreau sa imi demonstrez ca pot sa pun in aplicare teoria invatata in ultimii 4 ani. Vreau sa imi probez capabilitatile, cautand o implementare pentru ideile pe care nu am avut initiativa sa le dezvolt pana acum. Vreau sa compun un proiect in care pot sa-mi distribui atentia exclusiv asupra detaliilor teoretice. De asemenea, vreau sa explorez domeniul informaticii si sa aflu ce vreau sa fac in viitor.

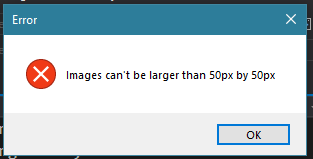
1. **Structura aplicatiei**

Graphical user interface, application

Description automatically generatedAplicatia contine o fereastra principala, numita MainWindow. Aceasta contine formele secundare MinefieldBackDrop si Minefield, iar fereastra CreateField poate fi accesata prin optiunea “Custom...” din meniul Game.

**CreateField**

Meniul Game ofera modalitati de a incepe un joc nou, alaturi de 8 optiuni exclusive pentru schimbarea campului de mine:

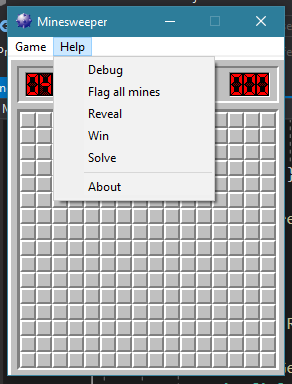
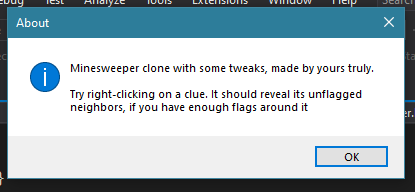
* “Beginner”, una dintre dificultatile valabile jocului original, genereala un camp de 9 x 9 cu 10 mine.
* Beginner, but harder” genereala un camp de 9 x 9 cu 35 mine.
* “Intermediate”, dificultatea de inceput a aplicatiei, genereaza un camp de 16 x 16 cu 40 de mine.
* “Intermediate, but harder” genereaza un camp de 16 x 16 cu 100 de mine
* “Expert”, reprezentant ultima dificultate prezenta in jocul original, genereaza un camp de 30 x 16 cu 99 de mine.
* “Expert, but harder” genereaza un camp de 30 x 16 cu 200 de mine.
* “Load from image...” genereaza un camp dupa imaginea selectata, cat timp are dimensiuni sub 50x50 si plaseaza mine pentru pixelii cu o culoare diferita de RGB:255, 255, 255. Daca imaginea e prea mare, se afiseaza:
* Graphical user interface, application

  Description automatically generated“Custom...” deschide fereastra CreateField, prin care utilizatorul poate sa furnizeze dimensiunile unui camp si numarul de mine, urmand sa se genereze un camp cu valorile respective, cat timp minele nu reprezinta mai mult de 75% din camp. In caz contrar, se afiseaza mesajul:

Tot in meniul Game sunt prezente doua optiuni:

* “Marks(?)” – Activarea acestei optiuni permite folosirea semnelor de intrebare. Pentru a marca un spatiu, utilizatorul trebuie sa apese un steag cu butonul drept al mouse-ului. Spre deosebire de steaguri, semnele de intrebare sunt o schimbare pur estetica, iar un spatiu marcat cu semn de intrebare se comporta ca un spatiu normal.
* “Generate solvable fields only” – Aceasta optiune limiteaza campul prezentat jucatorului, daca nu a fost generat dupa o imagine, spre a avea cel putin o solutie garantata. Voi dezvolta asupra acestui proces mai tarziu.

Al doilea meniu, numit Help, contine mai multe optiuni pentru a ajuta utilizatorul, alaturi de sectiunea “About”. Aceste optiuni sunt, in aceasta ordine:

* “Debug” – Afiseaza, daca spatiul nu este descoperit, un patrat de 4 x 4 in coltul din stanga-sus al spatiilor cu mine sau numarul de mine din jur pentru indicii sau spatii libere. Aceste informatii sunt afisate doar pentru spatii acoperite fara steaguri sau semne de intrebare. Optiunea poate fi activata sau dezactivata.
* “Flag all mines” – Plaseaza steaguri peste spatiile cu mine
* “Reveal” – Descopera toate spatiile libere sau cu indicii
* “Win” – Sfarseste jocul in conditii favorabile utilizatorului
* “Solve” – Invoca metoda RunVisualSolver pentru a rezolva cat mai mult dintr-un camp
* “About” – Afiseaza date despre aplicatie

1. **Detalii Tehnice si Limbajul Utilizat**

Initial intentionam sa fac acest proiect in CodeBlocks folosind C++ dar, pentru a crea o aplicatie, trebuia sa imi instalez wxWidgets. Probabil nu este atat de greu sa instalezi aceasta extensie, dar pentru mine a fost. Dupa 8 ore de incercari esuate, in prima zi de lucru, am renuntat la CodeBlocks si, in ziua urmatoare, am inceput sa caut platforme alternative. Eventual, am incercat sa folosesc Unity, impreuna cu Visual Studio pentru scripturi. Dupa ce am creat o parte din joc, am renuntat la Unity si am continuat proiectul exclusiv in Visual Studio. Unity este o platforma ideala pentru crearea jocurilor, insa ma simteam pierdut cautand unelte in cadrul aplicatiei. Prin tranzitia de la Unity Editor la Windows Forms App, am putut sa-mi implementez singur metodele, scapand de confuzia cauzata de optiunile interminabile prezentate de Unity. Visual Studio contine, de asemenea, unelte pentru proiectare, insa, spre deosebire de Unity Editor, acestea sunt mult mai usor de inteles si de aplicat.

Am reusit sa ma familiarizez rapid cu C#, desi difera de C++. Cele doua limbaje de programare sunt destul de similare, mai ales prin structura codului. C#, insa, compileaza codul in CLR(common language runtime), aloca memorie dinamic si, in opinia mea, este mult mai usor de utilizat.

In acest proiect, pe langa algoritmi, am utilizat evenimente, spre a inregistra interactiunile utilizatorului cu tabla de joc, pentru crearea si actualizarea constanta a spatiilor din camp. Voi dezvolta aceste procese, daca va fi nevoie, mai tarziu. De asemenea, am folosit task-uri pentru a genera in acelasi timp mai multe solutii pentru camp in acelasi timp, desi, in urma dezvoltarii metodei de rezolvare, nu avem nevoie de threading pentru a genera o solutie garantata.

1. **Evenimente importante**

* OnMouseDown override

OnMouseDown este apelat la inceputul unui click si isi executa instructiunile doar daca poti sa interactionezi cu aplicatia si daca jocul nu s-a sfarsit.

In cazul pozitiv, salveaza pozitia cursorului in x si y. Daca x si y se incadreaza in dimensiunile matricei se verifica daca utilizatorul a apasat butonul stang sau drept al mouse-ului.

In primul caz aplica un efect special, eliminand temporar orice textura in spatiul apasat, daca este acoperit, sau in spatiile acoperite vecine, daca nu este acoperit.

In al doilea caz, daca spatiul apasat este acoperit, verifica daca este marcat cu un steag, cu semn de intrebare sau daca nu este marcat cu nimic. In primul caz, elimina steagul si, daca are optiunea cu semnele de intrebare activata, marcheaza spatiul cu un semn de intrebare. In al doilea caz, elimina semnul de intrebare iar, in al treilea caz, marcheaza spatiul cu un steag.

* OnMouseUp override

OnMouseUp este apelat la sfarsitul unui click si aplica functia DiscoverTiles pentru parametrii spatiului apasat, daca este acoperit sau, daca nu este acoperit si are mai multe steaguri in jur decat mine, aplica aceeasi functie pentru spatiile vecine fara steaguri. Efectul mentionat anterior este eliminat pentru spatiile descoperite cu ajutorul functiei.

* OnMouseMove override

OnMouseMove actualizeaza acel efect special sau il elimina complet din spatiile marginale daca x si y nu se incadreaza in camp.

* OnMouseLeave override

OnMouseLeave elimina, intr-o maniera similara, efectul special din spatiile marginale.

* OnPaint override

OnPaint actualizeaza reprezentarea campului in cadrul aplicatiei. In primul rand, sterge reprezentarea anterioara si creeaza un camp de spatii descoperite si libere, sau fundalul campului. In continuare, aplica o textura fiecarui spatiu din camp prin functia DrawTile, alaturi de o evidentiere rosie(50, 255, 0, 0) pentru indicii invecinate cu prea multe steaguri si/sau o evidentiere roz(150, 255, 0, 255) pentru spatiul evidentiat de RunVisualSolver.

* DrawTile

Daca spatiul are acel efect special mentionat anterior aplicat, functia DrawTile ii aplica textureBlank, o textura vida. Ar putea foarte bine sa nu ii aplice nimic si am obtine acelasi rezultat.

In caz contrar, daca spatiul este acoperit, functia ii aplica textureField si, daca spatiul este marcat cu un steag sau semn de intrebare, ii atribuie textureFlag, respectiv textureQuestionmark.

Daca spatiul nu este acoperit, insa, sunt verificate 4 cazuri:

* Daca spatiul are o mina:
  + si acea mina este mina descoperita de utilizator, functia atribuie spatiului textura unei mine normale(textureMine) cu un fundal rosu.
  + si are un steag, functia aplica spatiului textureFlaggedRight
  + normala, DrawTile atribuie spatiului textureMine
* Daca spatiul nu are o mina, dar are un steag, functia insuseste spatiului textureFlaggedWrong
* Daca spatiul nu are nicio mina in vecinatatea sa, este aplicat permanent efectul special, desi, asemanator lui, aplicarea unei texturi nu este necesara.
* Daca spatiul are cel putin o mina in jur, DrawTile ii aplica textura corespunzatoare numarului de mine vecine.

La final, daca optiunea Debug este activa si spatiul este acoperit, DrawTile aplica o reprezentare a spatiului.

1. **Algoritmica**

* DiscoverTiles

DiscoverTiles este folosit in minefield ca metoda generala pentru descoperirea spatiilor. Aceasta metoda genereaza un camp nou, descopera spatii si/sau muta o mina, depinzand de caz. DiscoverTiles nu genereaza un camp nou si nu schimba pozitia minelor pentru campuri generate dupa imagini, pentru a pastra imaginea intacta.

* GenerateField

GenerateField cauta de 10 ori in acelasi timp un camp cu o solutie garantata, timp de 20 de secunde. Daca dureaza mai mult de 20 de secunde sa se genereze un camp valid, algoritmul atribuie campului principal ultimul camp generat. GenerateField este folosit, de asemenea, pentru a genera un camp simplu, in cazul in care punctul de origine solverOrigin nu are nicio valoare. Aceasta cale este folosita pentru a genera un camp la inceput, inainte de primul click, prmanent in cazul in care nu trebuie generat un camp cu solutie garantata.

* FindValidField plaseaza mine aleatoriu si calculeaza indiciile pentru restul spatiilor cat timp:
  + trebuie generat un camp cu o solutie garantata

si

* + nu a fost incalcata limita de 20 de secunde

si

* + nu a fost gasita un camp cu o solutie garantata
* TryToSolve

TryToSolve este folosit in cautarea unui camp valid, aceasta incearca sa rezolve cat mai mult din campul primit si returneaza true daca functia a descoperit toate spatiile libere. Daca instanta minefield nu are nicio valoare, campul trebuie rezolvat de la origine. In caz contrar, insa, campul a fost partial rezolvat de utilizator, deci doar trebuie adaugate indiciile descoperite in lista. Ca masura de siguranta, cand apelam RunVisualSolver, TryToSolve elimina toate steagurile, pentru a elimina posibilitatea unei rezolvari gresite.

In continuare, TryToSolve apeleaza functiile SolveViaRevealedNumbers, pentru rezolvarea propriu-zisa si CheckWin, pentru a verifica daca SolveViaRevealedNumbers a rezolvat tot campul.

* SolveViaRevealedNumbers

In cadrul functiei SolveViaRevealedNumbers, pot fi rezolvate 3 tipuri de indicii

* + Indiciile care au destule steaguri in spatiile vecine (numarul indiciului este egal cu numarul de steaguri vecine)
  + Indiciile care pot fi imediat completate cu steaguri (numarul de spatii vecine acoperite fara steaguri este egal cu numarul de mine fara steag)
  + Indiciile care prezinta mai multe posibilitati de completare cu steaguri, iar toate posibilitatile au ceva in comun(completeaza alte indicii apropiate si/sau atribuie un steag unor spatii acoperite vecine)

Daca toate indiciile incluse in tipurile de ai sus au fost rezolvate, atunci scimbam o mina din vecinatatea unui indiciu cu un spatiu fara mina acoperit.

Din cele 3 cazuri mentionate mai sus, primele doua sunt relativ usor de rezolvat. Al treilea, insa, necesita mai mult efort.

Mai intai sunt calculate toate combinarile de n luate cate k, unde n este numarul de spatii vecine acoperite fara steaguri, iar k este numarul de mine fara steag.

In acest caz, indiciul de pe linia 3, coloana 6 are 4 mine vecine, 2 mine marcate cu steaguri, 2 mine fara steag si 4 spatii vecine acoperite fara steaguri.

 Exista modalitati de a plasa restul steagurilor. Acestea sunt:

A picture containing calendar

Description automatically generated

In primul caz, indiciile de 3 de pe liniile 3, respectiv 4 si coloana 4 au prea multe steaguri vecine, iar in ultimul caz, indiciul de 2 de pe linia 3 si coloana 7 are, de asemenea, prea multe steaguri in vecinatatea sa. Astfel, aceste cazuri pot fi eliminate. Ramanem cu 4 cazuri.

 In continuare, putem observa faptul ca indiciile din pozitiile L3C4, L4C4 si L3C7 sunt rezolvate in toate cele 4 cazuri posibile. Astfel, putem sa descoperim spatiul din L5C5 si L4C8. Vom aplica aceste schimbari campului.

In continuare, observam ca indiciul de 2 din L3C8 Poate fi completat doar in doua moduri:



 Observam ca in amandoua cazurile, indiciul de 2 din L4C8 este rezolvat, astfel, descoperim spatiile din L5C7, L5C8 si L5C9.

 De asemenea, indiciul din L3C7 ar rezolva indiciul din L5C7, dar nu voi continua, fiindca am demonstrat deja ce voiam sa demonstrez.

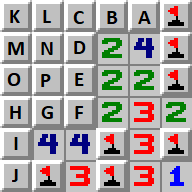
In acest caz, indiciul de 2 din L3C3 poate fi completat in 3 modalitati. Ultima atribuie indiciului de 1 prea multe steaguri, deci doar primele doua modalitati sunt valide.



Toate cazurile posibile atribuie spatiului din L2C4 un steag, deci putem sa notam acel steag ca fiind permanent. In mod normal, ar trebui sa vorbesc si despre indiciul din L4C3, dar consider ca nu este necesar. Voi spune doar ca spatiile L5C2, L5C3 si L5C4 ar trebui sa fie descoperite.

 In cazul in care algoritmul nu mai poate sa rezolve niciun indiciu, acesta muta o mina din vecinatatea unui indiciu descoperit in primul spatiu acoperit fara mine. Pentru a pastra corectitudinea indiciilor, trebuie actualizate indiciile din jurul spatiilor afectate. In plus, exista un caz particular in care mina si spatiul in care ar trebui sa fie transferata sunt vecine. In acel caz, una din actualizarile mentionate trebuie sa tine cont de cealalta. Algoritmul acesta, de exemplu, nu actualizeaza indiciul vecin spatiului minei mutate, daca pozitiile acestuia coincid cu fostele pozitii ale minei.

1. **O alta posibila modalitate de rezolvare a unui camp**

 In timpul proiectarii acestui proiect, m-am gandit la mai multe metode de a rezolva anumite probleme. Astfel, in ultimele zile, am realizat ca un camp poate fi rezolvat utilizand matrice.

In acest exemplu, avem spatiile A, B, C, D, E, F, G, H, I, J cu macar un vecin descoperit. Considerand ca, daca un spatiu este egal cu 1, acesta este mina si, daca este egal cu 0, acesta este ori un indiciu, ori un spatiu liber, avem urmatoarele ecuatii:

A + B = 4 – 3 = 1;

A + B + C + D + E = 2;

D + E + F = 2;

E + F = 2 – 1 = 1;

F + G = 4 – 3 = 1;

F + G + H + I + J = 4 – 1 = 3;

Daca adaugam fiecarei ecuatii restul termenilor inmultiti cu zero obtinem:

1A + 1B + 0C + 0D + 0E + 0F + 0G + 0H + 0I + 0J = 1;

1A + 1B + 1C + 1D + 1E + 0F + 0G + 0H + 0I + 0J = 2;

0A + 0B + 0C + 1D + 1E + 1F + 0G + 0H + 0I + 0J = 2;

0A + 0B + 0C + 0D + 1E + 1F + 0G + 0H + 0I + 0J = 1;

0A + 0B + 0C + 0D + 0E + 1F + 1G + 0H + 0I + 0J = 1;

0A + 0B + 0C + 0D + 0E + 1F + 1G + 1H + 1I + 1J = 3;

Acest sistem de ecuatii poate fi scris ca o matrice in felul urmator:

1 1 0 0 0 0 0 0 0 0 | 1

1 1 1 1 1 0 0 0 0 0 | 2

0 0 0 1 1 1 0 0 0 0 | 2

0 0 0 0 1 1 0 0 0 0 | 1

0 0 0 0 0 1 1 0 0 0 | 1

0 0 0 0 0 1 1 1 1 1 | 3

Scazand linia 1 din linia 2 si linia 4 din linia 3 obtinem:

1 1 0 0 0 0 0 0 0 0 | 1

0 0 1 1 1 0 0 0 0 0 | 1

0 0 0 1 0 0 0 0 0 0 | 1

0 0 0 0 1 1 0 0 0 0 | 1

0 0 0 0 0 1 1 0 0 0 | 1

0 0 0 0 0 1 1 1 1 1 | 3

Daca scadem linia 3 din linia 2 si linia 5 din linia 6 ajungem la:

1 1 0 0 0 0 0 0 0 0 | 1

0 0 1 0 1 0 0 0 0 0 | 0

0 0 0 1 0 0 0 0 0 0 | 1

0 0 0 0 1 1 0 0 0 0 | 1

0 0 0 0 0 1 1 0 0 0 | 1

0 0 0 0 0 0 0 1 1 1 | 2

Din aceasta matrice pot fi extrase urmatoarele ecuatii:

C + E = 0; => C si E nu sunt mine

D = 1; => D este mina

E + F = 1; din moment ce E nu este mina, F este mina

F + G = 1; din moment ce F este mina, G nu este mina

Astfel, desi nu am selectat cel mai bun exemplu, am demonstrat o cale prin care as fi putut sa introduc calcului matricelor in algoritmul de rezolvare. Nu am facut asta, insa, fiindca la inceput intentionam sa folosesc un algoritm recursiv, chiar daca, in forma sa finala, acesta contine doar o singura functie recursiva, numita BT.