Atestat la informatică - PacMan

realizat de Alexandrescu Andra

C.N.Gh.Șincai

clasa 12E

anul școlar 2022-2023

Am ales să realizez proiectul de atestat la informatică intitulat PacMan, reprezentând o replică a jocului cunoscut la scară largă, proiectat la începuturile dezvoltării domeniului programării și lansat pentru prima oară în 1980, deoarece consider că prezintă algoritmi (game mechanics) fundamentali în înțelegerea jocurilor video. PacMan este ”an action maze chase video game”, în care personajul PacMan parcurge un labirint, având ca obiectiv adunarea punctelor, figurată prin acțiunea de a mânca ”pellets”, fiind urmărită evitarea celor patru fantome, ce prezintă comportamente diferite și necesită o anumită abordare din partea jucătorilor pentru câștigarea nivelurilor jocului pe măsură ce dificultatea crește.

O replică a jocului Pacman implică înțelegerea noțiunilor de bază ale conceperii de jocuri video, deoarece este un joc clasic ce necesită un anumit nivel de cunoștințe în domeniul designului de jocuri, cum ar fi: concepte de gameplay - cum funcționează jocul și care este scopul îndeplinirii task-urilor, ce obiective trebuie atinse, ce obstacole trebuie depășite; concepte de grafică și animație - cum se creează personajele, cum se realizează animațiile pentru mișcarea personajelor și alte efecte vizuale.

Limbajul în care l-am programat este C++ și am apelat la o librărie externă pentru aplicații grafice numită Raylib, deoarece are o funcționalitate crescută, cu o sintaxă intuitivă, cuprinzând funcții utile pentru crearea jocurilor video, inclusiv grafică 2D și 3D, sunet, fizică și input. Pentru scrierea codului am utilizat drept software Visual Studio, considerat atât un compilator, cât și o interfață de dezvoltare a aplicațiilor.

Astfel, o replică a jocului Pacman poate fi un proiect reprezentativ pentru începătorii în domeniul dezvoltării de jocuri video, deoarece le oferă ocazia de a exersa și de a îmbunătăți abilitățile necesare în acest domeniu. Menționez că PacMan este adresat jucătorilor de orice vârstă, fiind un joc clasic cunoscut în întreaga lume.

**Despre istoria jocului**

PacMan este un joc video clasic creat de designerul de jocuri japonez Toru Iwatani și a devenit rapid un fenomen cultural, iar caracterul PacMan este încă unul dintre cele mai recunoscute personaje de jocuri video din toate timpurile. Succesul său poate fi atribuit mecanicii sale simple, dar captivante, graficii colorate și fermecătoare și atractivității sale universale pentru jucătorii de toate vârstele și genurile.

La începutul anilor 1980, jocurile video erau asociate cu spațiile dominate de bărbați atât în Japonia, cât și în SUA. Este considerat că arcade-urile (arcade games) au apărut după ce jocurile video deveniseră deja un succes, iar prin intermediul fenomenului de popularizare, acestea au început să apară în baruri, săli de bowling și cinematografe. Spre deosebire de jocurile arcade apreciate de public, care, prin natura lor violentă, ofereau o experiență senzorială copleșitoare, definită de lumină scăzută, zgomote puternice, PacMan a reprezentat o noutate în domeniul celor pasionați de jocuri video. Astfel, Toru Itawani consideră conceperea unui joc adecvat copiilor, femeilor și familiilor.

Inspirația programatorului japonez în crearea lui PacMan provine dintr-o idee referitoare la asemănarea unei plăcinte cu o gură deschisă, înfometată, asociată unei creaturi rotunde. Forma i-a amintit de o versiune rotunjită a caracterului japonez pentru "gură", numit kuchi. După ce a decis asupra unui design de caracter, Iwatani a derivat numele din "paku paku", o onomatopee japoneză pentru a mânca - sunetul care a inspirat semnătura jocului, pe măsură ce PacMan înghite puncte și fructe. De asemenea, PacMan a fost unul dintre primele jocuri care a prezentat scene și o poveste, adăugând experienței sale captivante. Deși povestea jocului este simplă, a fost interpretată în multe moduri diferite de-a lungul anilor, fiind de majoritatea văzut ca o metaforă a luptei dintre bine și rău, PacMan reprezentând forțele binelui și fantomele reprezentând forțele răului.

În Japonia, jocul a debutat ca Puck-Man, dar a fost ajustat ulterior pentru publicul american. Este impresionantă popularitatea de durată a jocului și a personajului, care continuă să captiveze publicul la mai bine de 40 de ani de la lansarea inițială.

**Despre principiile utilizate în joc**

Jocul este presupune controlarea personajului PacMan, care navighează printr-un labirint în timp ce mănâncă puncte și evită fantomele. Fantomele, numite Blinky, Pinky, Inky și Clyde, încearcă să-l prindă pe PacMan, iar dacă o fac, jucătorul pierde o viață din cele patru, de fiecare dată. Jocul include, de asemenea, granule de putere care pot fi consumate de PacMan pentru a schimba comportamentul fantomelor, ele devenind vulnerabile la atingerea lor de către Pac-Man, fiind astfel inversate rolurile de chasing și running. Jocul prezintă algoritmi deosebiți referitor la modul de comportare a fantomelor, non-playable characters ce funcționează pe baza logicii de căutare a căii utilizate de joc (path finding).

Fiecare rundă începe cu fantomele ieșind din centrul labirintului, începând urmărirea lui PacMan. De fiecare dată când PacMan este capturat de o fantomă, ele se întorc pe poziția inițială și un nou PacMan este plasat la pornire înainte ca jocul să continue. Când labirintul este curățat de toate punctele, harta este resetată și începe o nouă rundă. Pe măsură ce PacMan progresează prin niveluri, labirinturile devin mai complexe, iar fantomele devin mai dificil de evitat. Jocul include și bonusuri de fructe care apar în diferite puncte de-a lungul jocului, care pot fi colectate pentru puncte suplimentare.

Fantomele au trei moduri de comportament diferite în timpul jocului: urmărire, împrăștiere și înspăimântare:

CHASE — Obiectivul unei fantome în modul urmărire este să găsească și să-l captureze pe PacMan în labirint. Fiecare fantomă prezintă un comportament unic atunci când îl urmărește pe PacMan, oferindu-le personalități diferite: Blinky (roșu) este foarte agresiv și greu de evitat odată ce ajunge în spatele tău, Pinky (roz) tinde să ajungă în fața ta și să te întrerupă, Inky (albastru deschis) este cel mai puțin previzibil dintre toate, iar Clyde (portocaliu) pare să rămână departe de target.

SCATTER—În modul scatter, fantomele renunță la urmărire pentru câteva secunde și se îndreaptă spre colțurile de acasă. Destul de curând, ele vor reveni în modul urmărire.

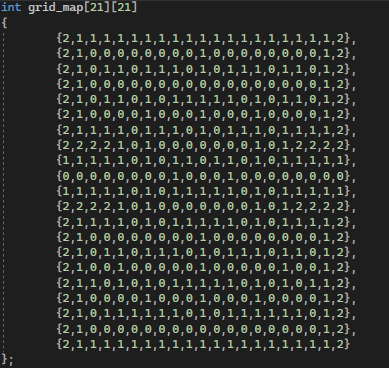
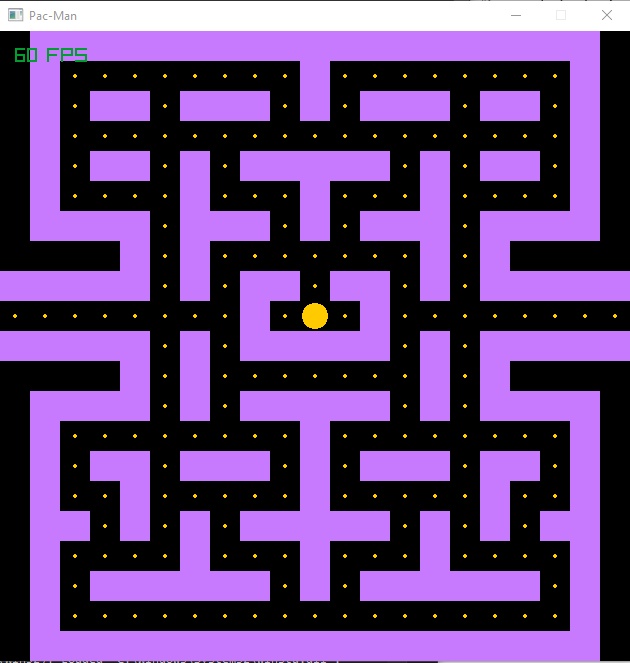
FRIGHTENED — Fantomele intră în modul înspăimântat ori de câte ori PacMan mănâncă unul dintre cele patru energizante situate în colțurile îndepărtate ale labirintului. În primele niveluri, fantomele vor deveni albastru închis (însemnând că sunt vulnerabile) și vor rătăci fără scop prin labirint timp de câteva secunde. Ele vor clipi câteva momente înainte de a reveni la modul lor anterior de comportament.

În toate cele trei moduri de comportament, fantomelor le este interzis să-și inverseze direcția de deplasare. De aceea, ele pot alege doar între a-și continua cursul curent sau a se întoarce într-o parte sau alta la următoarea intersecție. Astfel, odată ce o fantomă alege direcția în care să meargă la o intersecție labirint, nu are altă opțiune decât să continue înainte pe acea cale până când va ajunge la următoarea intersecție.

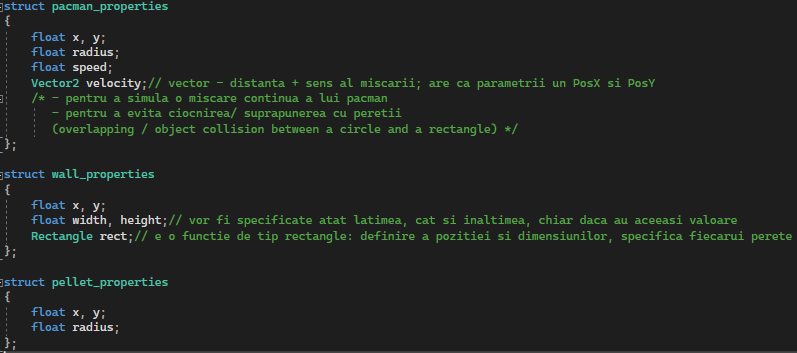
PacMan nu trebuie să aștepte până când ajunge la mijlocul unei viraj pentru a schimba direcția, așa cum o fac fantomele, el poate începe să rotească câțiva pixeli înainte de a ajunge la centrul unei viraj și după mai mulți pixeli după ce a trecut, practic în orice moment al jocului. Virajele luate cu unul sau mai mulți pixeli înainte de a ajunge în centru sunt ”pre-turns”; virajele luate după unul sau mai mulți pixeli sunt ”post-turns”. Această tehnică este cunoscută sub numele de ”cornering” și este una dintre primele abilități pe care un nou jucător Pac-Man ar trebui să le stăpânească. Pentru fiecare manevră reușită înainte de întoarcere, există o distanță mai mare între PacMan și celelalte fantome.

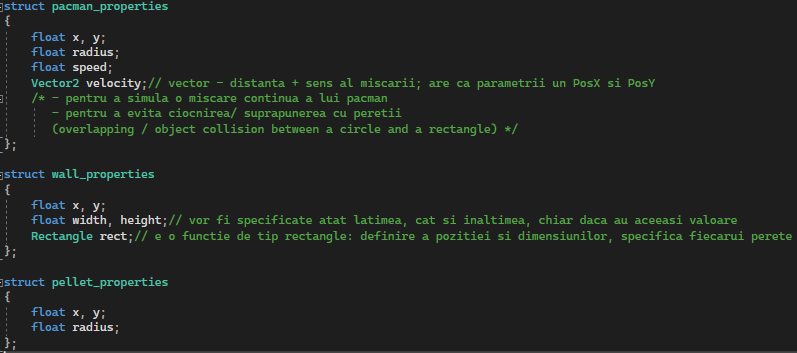
**Descrierea programului**

Pentru a implementa jocul, e necesară conceperea hărții sub forma unei matrice numite grid\_map[21][21], care să o replice pe cea a jocului original, fiind regăsită pe tot parcursul nivelurilor. Valorile matricei sunt de 0 – ce înseamnă poziționarea unui punct (pellet) pe care PacMan îl poate mânca, putând să parcurgă traseul format din ”walkable tiles”; 1 – ce reprezintă obstacolele definite prin pereți (walls), prin care PacMan nu poate pătrunde și 2 – ce definește marginile hărții în care e imposibilă deplasarea, dar care vizual oferă o perspectivă îmbunătățită jucătorului, deoarece singurele părți prin care marginea hărții nu e închisă, oferă posibilitatea de teleportare a lui PacMan.

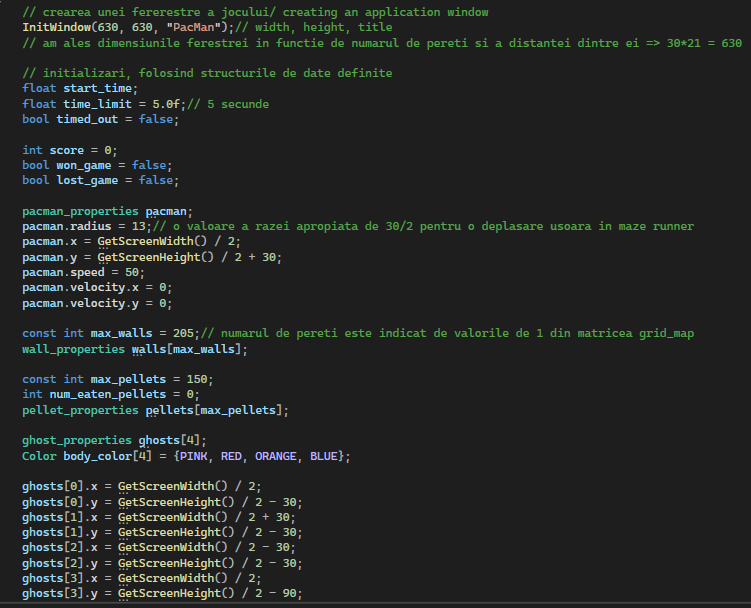
Deoarece este important a menține evidența poziției lui PacMan, dar și a punctelor și a pereților, în momentul detectării ciocnirilor (collisions), am creat structuri de date în care memorez coordonate (posX/ posY), raze (radius), viteze (speed) și velocity. Speed și velocity reprezintă ambele măsurători ale vitezei cu care un obiect se mișcă, dar au semnificații diferite, deoarece viteza este o cantitate scalară care se referă la rata cu care un obiect parcurge o distanță, în timp ce viteza deplasării este o cantitate vectorială care se referă la rata cu care un obiect își schimbă poziția. De aceea, velocity include nu numai viteza obiectului, ci și direcția în care se mișcă, fiind definită ca un vector.

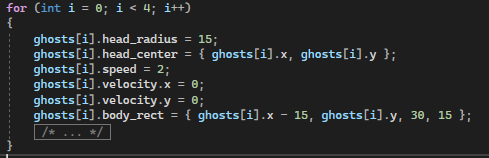




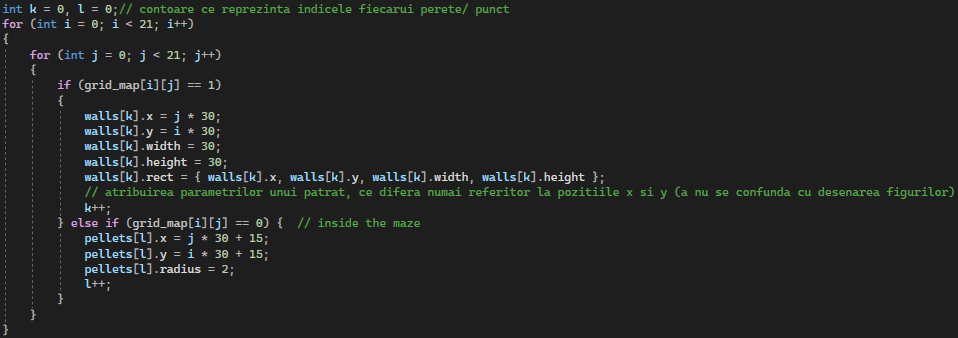
Inițializarea presupune crearea unei ferestre a jocului, pe care am ales-o de dimensiuni 630 (width și height), întrucât fiecare tile (atât pe care se deplasează PacMan, cât și un obstacol) are dimensiunea de 30X30, iar dimensiunea specificată în grid\_map este de 21X21 (30\*21=630). Raza cercului prin care e reprezentat PacMan este de 13, aproximativ jumătate din dimensiunea unei plăci/ tile. Pentru a determina numărul maxim de pereți si pellets, am scris in CodeBlocks un algoritm de baza prin care un contor număra valorile de 1, respectiv de 0 din matrice. Vor fi 4 fantome de culori diferite, dintre care 3 sunt poziționate înainte de intrarea în maze. Fiecare fantomă e alcătuită dintr-un dreptunghi și un cerc.

Referitor la progresul făcut, există o variabilă ce reține scorul curent și altele două de tip bool pentru a decide asupra statusului. La atingerea unui milestone (losing or winning), există câteva variabile pentru a separa afișarea mesajelor (în cazul acesta vreau să afișez ”Congratulations!” pentru 5 secunde, înaintea mesajului de ”Please exit the game”).

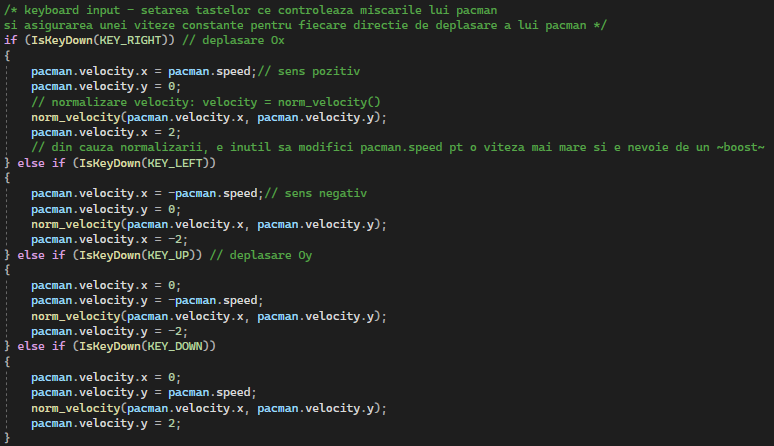


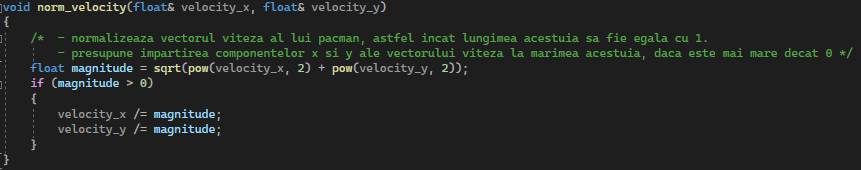


De asemenea, fiecărui wall și pellet le sunt atribuiți indici, căpătând fiecare câte o poziție pe axele Ox și Oy. Funcția Rect (Rectangle) este specifică librăriei raylib, definind atributele unui pătrat, care e necesar a fi identificat drept obstacol.

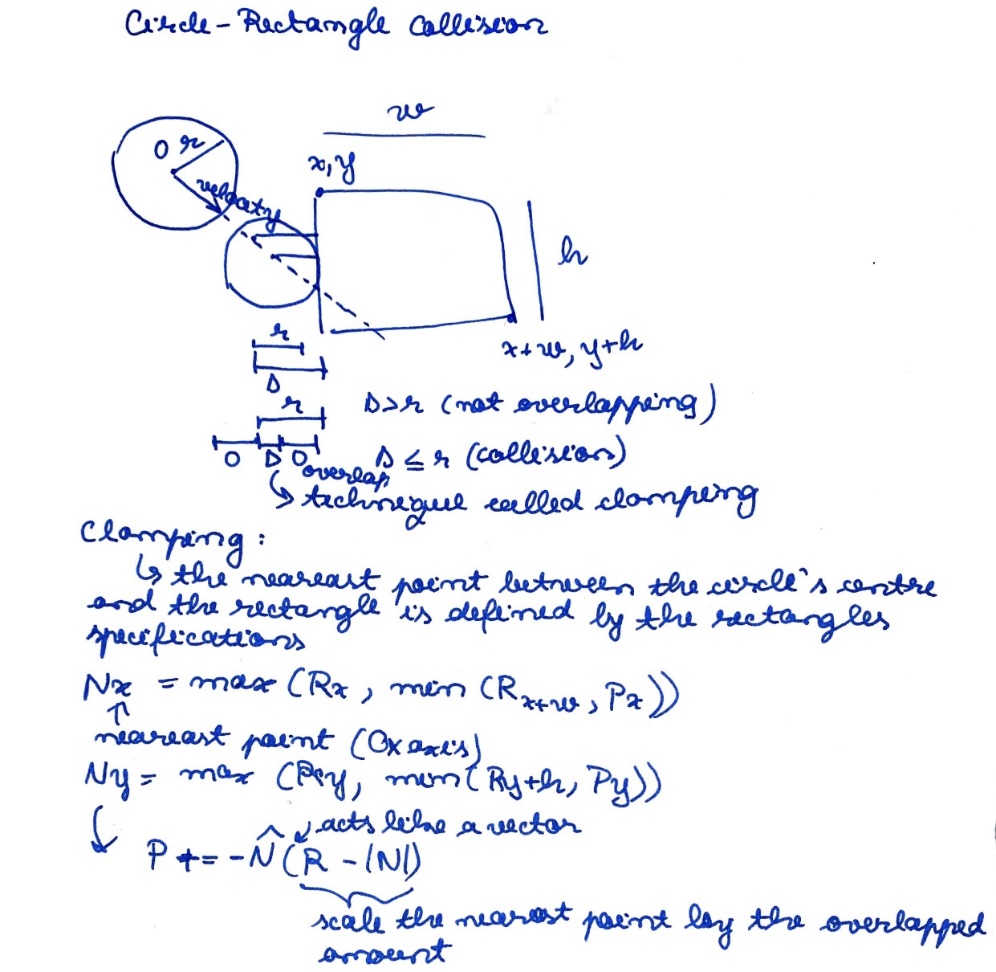


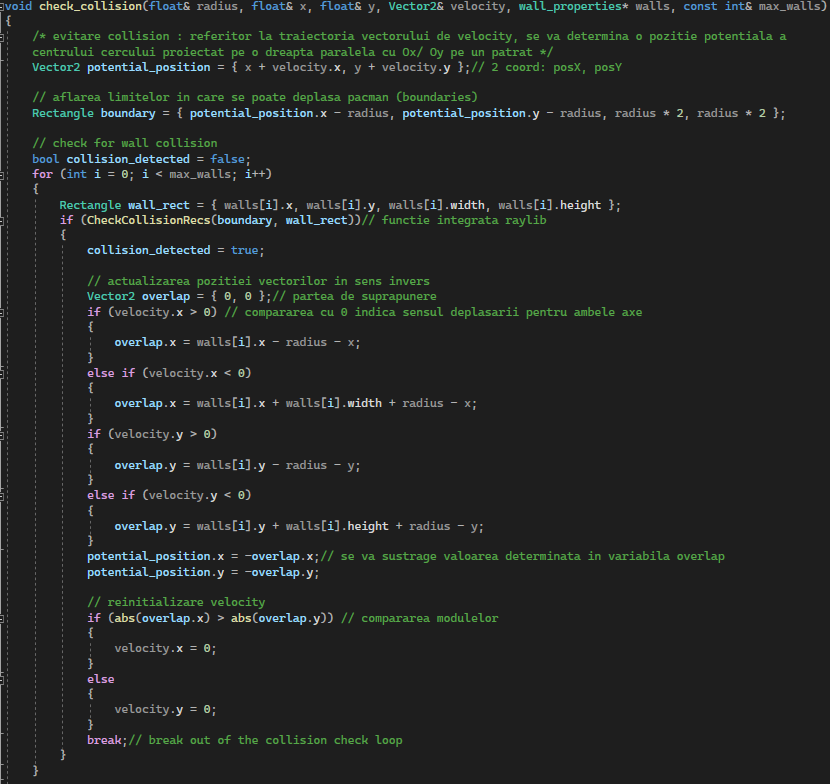
În game loop-ul acestui program, se vor defini tastele utilizate pentru mișcarea lui PacMan, însemnând modificarea velocității în sens pozitiv, respectiv negativ pe ambele axe, în mod corespunzător. În plus, pentru o mișcare constantă, neaccelerată, trebuie normalizată velocity, folosind împărțirea la mărimea vectorului de viteză, printr-o relație fundamentală matematică. Este de menționat că din cauza normalizării, indiferent de modificarea făcută asupra vitezei pacman.speed, nu va fi observată nicio modificare, motiv pentru care trebuie adăugați niște parametri.

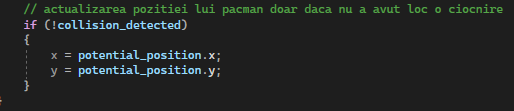




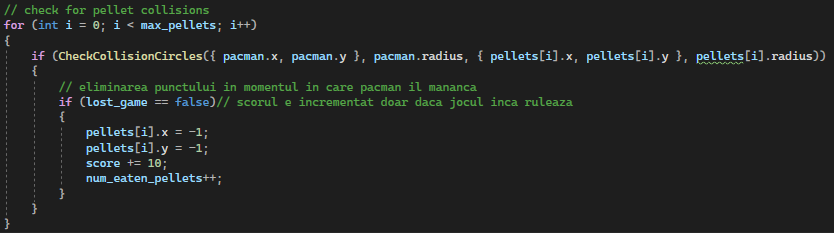
Coliziunea cu pereții este o parte esențială a jocului, deoarece determină mișcarea lui PacMan și a fantomelor în cadrul labirintului. Când PacMan se ciocnește de un perete, mișcarea lui este restricționată, în mod similar cu fantomele. Pentru a obține acest efect de coliziune a pereților, matricea bidimensională grid\_map[ ][ ] stochează locația pereților pe hartă. Fiecare element al matricei reprezintă un tile pe hartă, iar dacă el este un perete, valoarea elementului este setată la 1, iar când PacMan sau fantomele se mișcă, poziția lor este verificată față de matrice. Așadar, PacMan poate să se miște numai într-o zonă lipsită de obstacole, fiind nevoie de a verifica poziția potențială a jucătorului prin raportare la traiectoria vectorului velocity pe fiecare perete. Se numește poziție potențială deoarece acea poziție se verifică constant față de obstacolele din grid\_map[ ][ ]. Se va implementa și un boundary ”fictiv” al deplasării lui PacMan, definit drept Rectangle. În momentul în care o ciocnire e detectată, vectorul velocity va fi actualizat, în sens invers al deplasării curente și se modifică și poziția potențială a lui PacMan, prin scaderea valorii de overlap, care matematic reprezintă diferența dintre rază și distanța potențială.



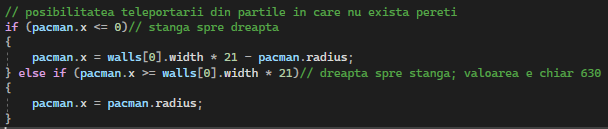




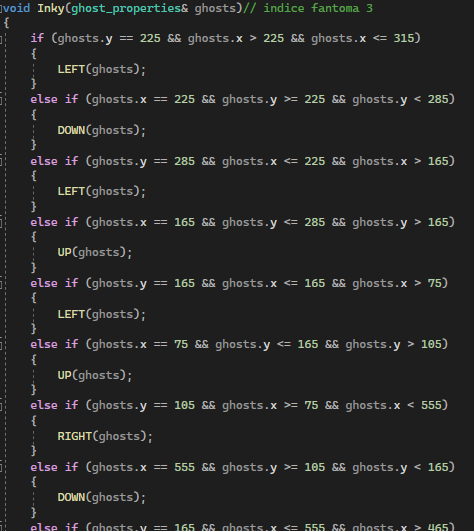
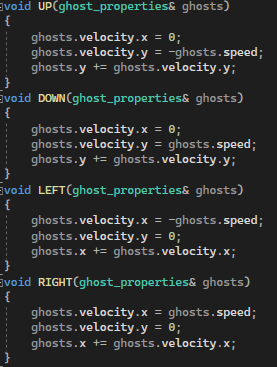
Există și un pellet collision check în momentul în care e necesar ca PacMan să obțină puncte, acestea urmând să dispară, simulând consumarea lor. Astfel, fiecare punct va fi eliminat prin poziționarea lui la coordonatele (-1, -1) din plan, ele nefiind vizibile pentru jucător. Se vor calcula și scorul, dar și numărul de eaten pellets.

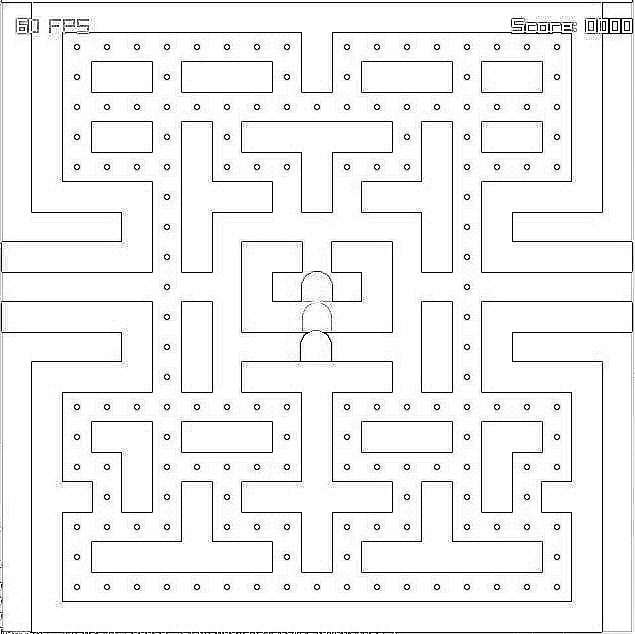
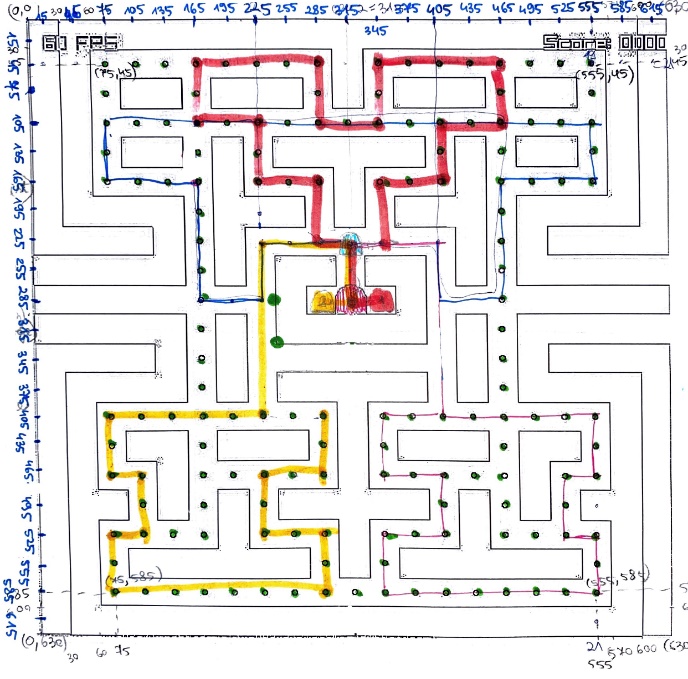


PacMan se poate deplasa și prin teleportare dintr-o margine a ferestrei, unde nu e obstacol, în cealaltă. În acest caz, trebuie modificată poziția lui PacMan, pentru a da impresia unei mișcări continue.

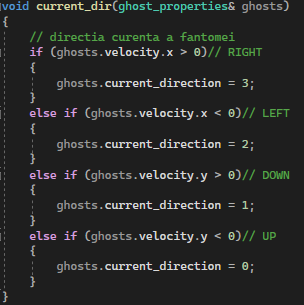
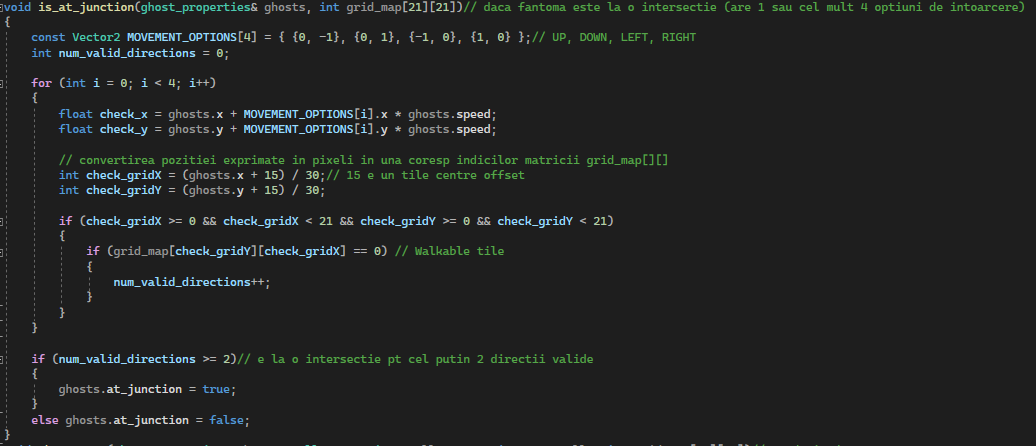


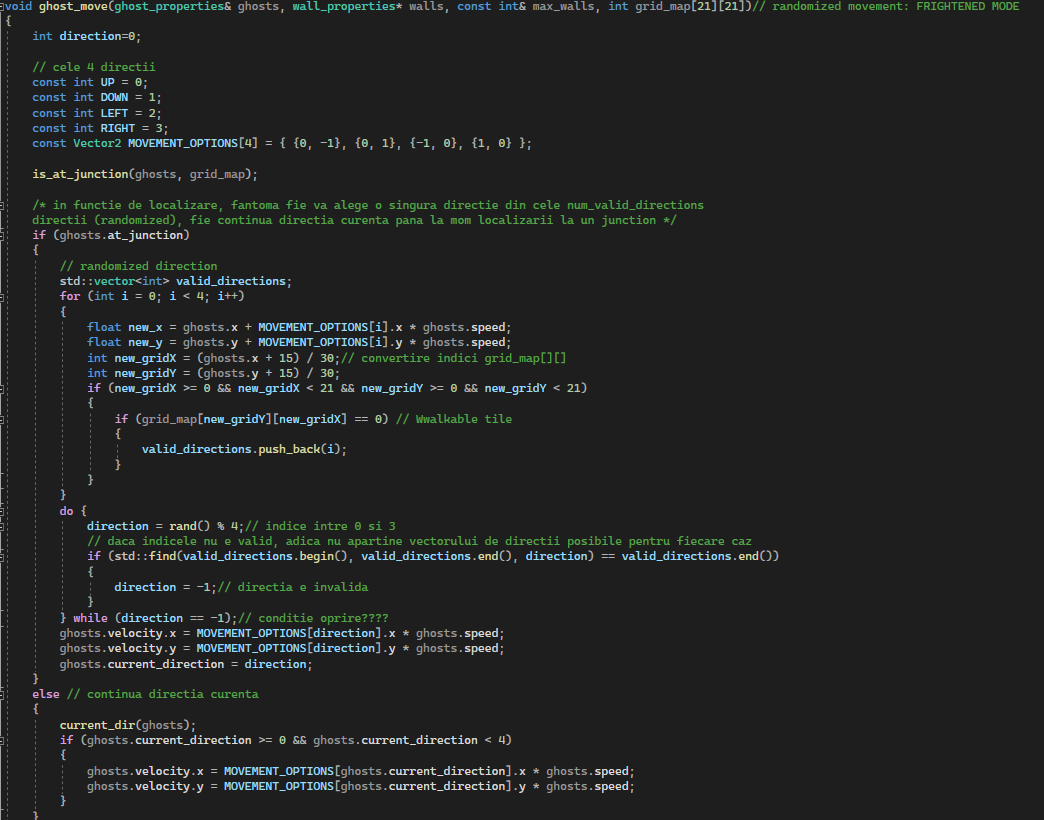
În modul de SCATTER, fantomele urmează un traseu predefinit, pe care îl repetă într-un loop. Am implementat acest behaviour prin stabilirea unor poziții, pe care acestea le urmăresc. Am ales să mă raportez la niște poziții exprimate în pixeli, în funcție de care modific sensul direcției vectorului de viteză pentru fiecare caz particular. În acest fel, nu e nevoie să verific ciocnirea cu pereții.

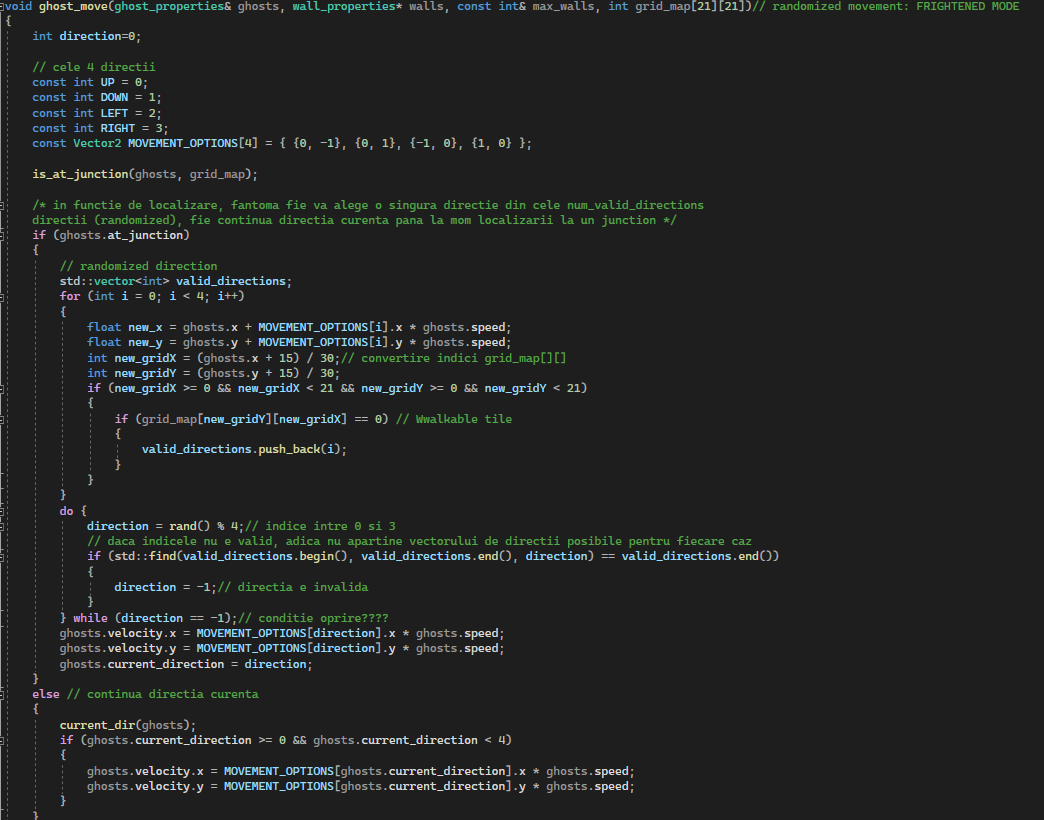


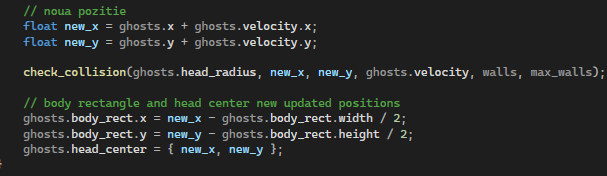


Un alt comportament al fantomelor descrie posibilitatea alegerii direcțiilor de deplasare în funcție de pozișionarea fiecăreia. Funcția current\_dir() are rolul de a actualiza direcția curentă a fantomului în funcție de vectorul de viteză. Funcția is\_at\_junction() verifică dacă fantoma se află într-o intersecție (junction) din maze, adică are una sau cel mult 4 opțiuni de întoarcere. Se verifică dacă există cel puțin două direcții valide în care fantoma poate să se miște la pasul curent. Verificarea se face pe baza hărții grid\_map[ ][ ] și a vitezei/ poziției curente ale fantomei. Funcția ghost\_move() gestionează mișcarea fantomei în mod aleatoriu (randomized): FRIGHTENED MODE. Se verifică dacă fantoma se află într-o intersecție sau dacă e obligată să urmeze direcția curentă. Dacă fantoma se află într-un junction, se alege o direcție aleatorie validă din cele disponibile și se setează viteza în consecință. La final, se actualizează poziția fantomei pe baza vitezei și se verifică coliziunile cu pereții prin apelul funcției check\_collision(). De asemenea, se actualizează poziția dreptunghiului corpului și centrului capului fantomei.

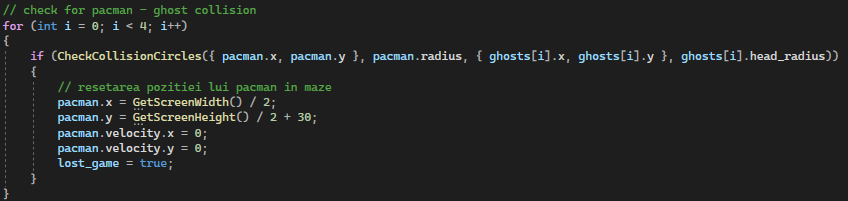
 



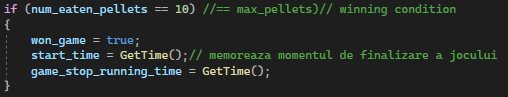




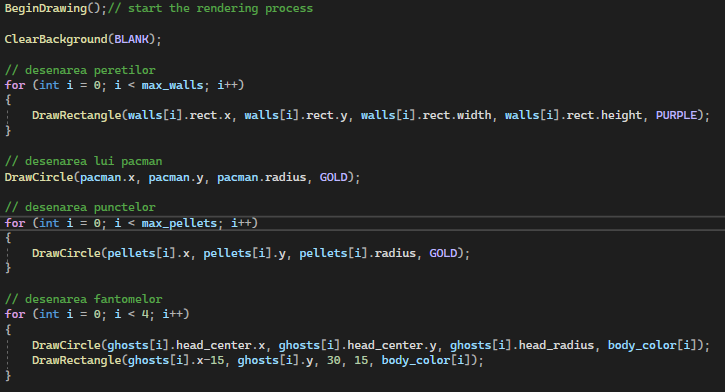
În momentul în care o fantomă îl atinge pe PacMan, înseamnă că jocul s-a sfârșit, iar poziția lui PacMan se resetează.



Jocul poate fi câștigat în momentul în care PacMan consumă toate punctele din maze, iar atunci va fi reținut timpul pentru afișarea mesajelor din final.

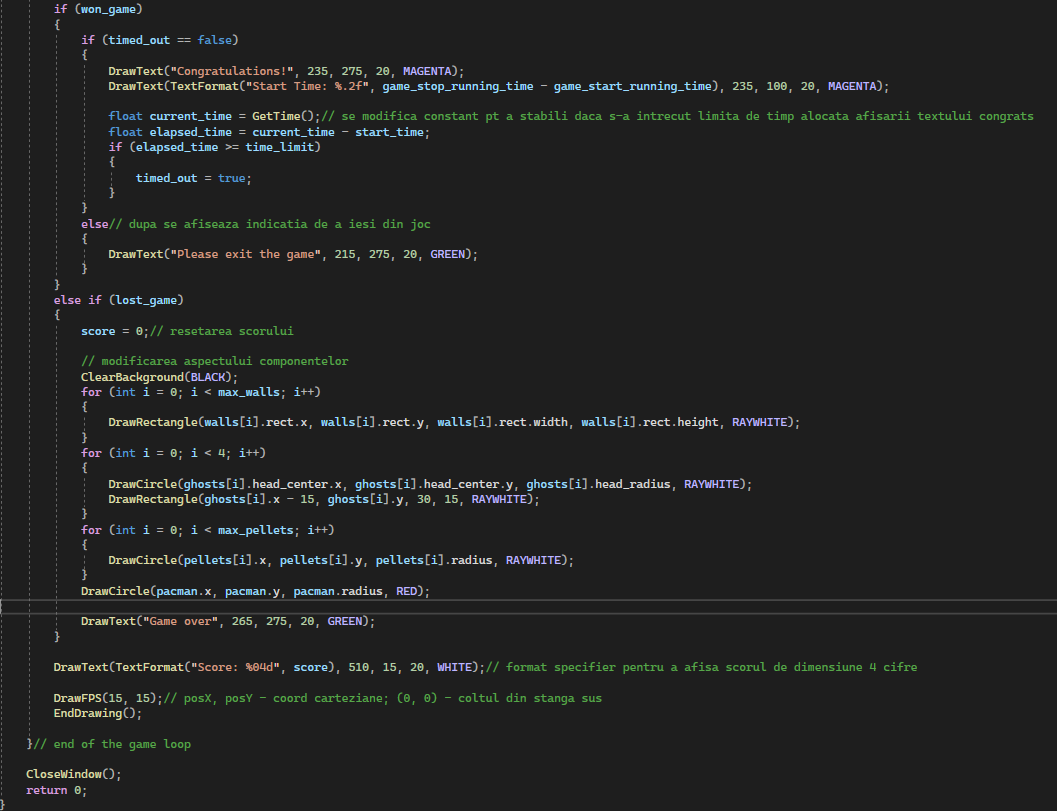


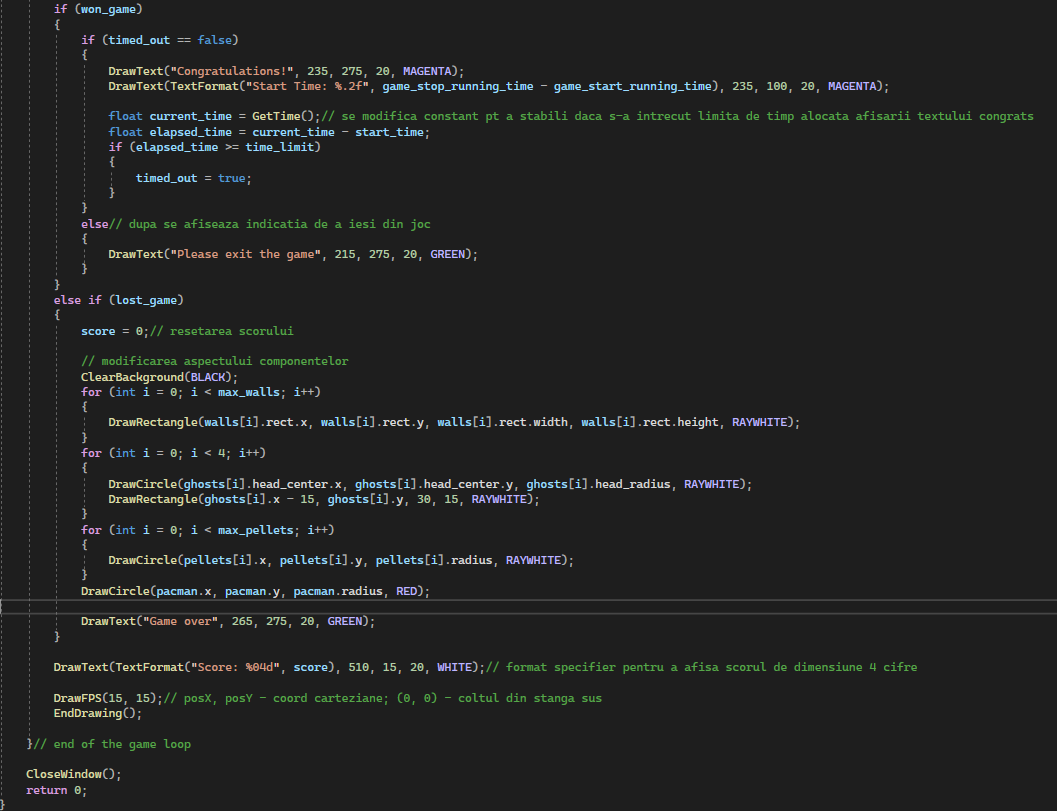
Procesul de rendering presupune desenarea formelor obiectelor, iar după această parte, se încheie ”the game loop”.



Dacă PacMan căștigă, vor fi afișate consecutiv, la un interval de 5 secunde două texte. În cazul în care pierde, se resetează scorul și se modifică aspectul fundalului, al pereților, al punctelor și al fantomelor pentru un efect dramatic. Pentru a vedea timpul total de la începutul până la finalizarea jocului, vor exista alte două variabile care folosesc funcția GetTime().







**Bibliografie**

<https://www.youtube.com/watch?v=D2a5fHX-Qrs>

<https://www.youtube.com/watch?v=GXlckaGr0Eo&t=3511s>

<https://www.youtube.com/watch?v=fdw-HGIMZFY&t=2290s>

<https://www.youtube.com/watch?v=vC0d1rDmPBs&t=165s>

<https://github.com/Kofybrek/Pacman>

<https://pacman.com/en/history/>

<https://pacman.holenet.info/>

<https://www.smithsonianmag.com/innovation/why-players-around-world-gobbled-up-pac-man-180974902/>

<https://www.raylib.com/cheatsheet/cheatsheet.html>

<https://www.raylib.com/examples.html>

<https://jeannoelseneque.medium.com/game-development-with-raylib-c-day-4-b3f2d9729836>

<https://github.com/Rabios/awesome-raylib#tutorials>

<https://monsterbraininc.com/2018/12/raylib-game-tutorial-space-invaders-using-vscode/>

<https://docs.racket-lang.org/raylib/Generated_Raylib_Bindings.html>

<https://ring-lang.github.io/doc1.12/ringraylib.html>

<https://cppsecrets.com/users/167511041011141171099811064103109971051084699111109/Raylib-Collision-Detection.php>