“Nu te supăra, frate!” în C++ Builder 6

15.01.2020

Îndrumător:
dr. ing. Daniel Morariu

Student:
Giubega Larisa-Nicoleta
Grupa 222/2

Istoric versiuni

Data	Versiune	Descriere	Autor
26.11.2019	1.0	Crearea claselor Pion si Pozitii, unde clasa Pion moștenește clasa Pozitii.	Giubega Larisa-Nicoleta
02.12.2019	1.1	Rezolvarea problemei de corelație a fiecărei imagini de pion cu clasa Pion.	Giubega Larisa-Nicoleta
14.12.2019	1.2	Rezolvarea problemei de mutare a pionului în funcție de numărul generat aleator al zarului.	Giubega Larisa-Nicoleta
16.12.2019	1.3	Rezolvarea problemei de mutare a pionului după un tur complet al tablei în funcție de culoarea sa (pentru toți cei 16 pioni).	Giubega Larisa-Nicoleta
19.12.2019	1.4	Modificarea programului astfel încât să utilizeze integral manipularea imaginilor prin intermediul claselor.	Giubega Larisa-Nicoleta
20.12.2019	1.6	Mutarea completă a pionilor pe tablă (de la poziția din casă până la poziția din casa finală).	Giubega Larisa-Nicoleta
21.12.2019	1.7	Crearea clasei Joc care moștenește la rândul ei clasa Pion.	Giubega Larisa-Nicoleta
23.12.2019	1.8	Crearea unui client pentru jucătorul roșu și un server care să primească text culoarea, numarul și coordonatele pionului mutat.	Giubega Larisa-Nicoleta
25.12.2019	1.9	Crearea unui client galben și a unui server care schimbă informații cu clientul jucătorului roșu și mută pe tabla clientului galben în funcție de ce se mută pe tabla clientului roșu.	Giubega Larisa-Nicoleta
01.01.2020	2.0	Crearea unei rețele funcționale pentru 3 pioni.	Giubega Larisa-Nicoleta
03.01.2020	2.1	Crearea unei rețele funcționale pentru 4 pioni, adăugarea zarului la codul de pe server și manipularea acestuia pentru ceilalți jucători.	Giubega Larisa-Nicoleta

05.01.2020	2.2	Adăugarea pop-up-ului "Este rândul jucătorului ---" în funcție de jucătorul al carui rând este (din codul de pe server).	Giubega Larisa-Nicoleta
07.01.2020	2.3	Crearea unui nou cod pentru rețea și modificarea fiecărei aplicații.	Giubega Larisa-Nicoleta
08.01.2020	2.4 (finală)	Adăugarea facilității de a schimba jucătorul în cazul în care jucătorul curent dă 6 cu zarul și toți pionii săi sunt în casă.	Giubega Larisa-Nicoleta

Cuprins

1 Specificarea cerințelor software

1.1 Introducere

Proiectul propune abordarea algoritmică a jocului "Nu te supăra, frate!" pentru 4 jucători și implementarea sa în limbajul de programare C++.

1.1.1 Obiective

Principalele obiective sunt crearea unei interfețe interactive (tablă, pioni, zar) ușor de utilizat pentru publicul larg, implementarea jocului în cauză și crearea cu succes a unui server care să accepte 4 jucători diferiți, pentru fiecare funcționând independent aplicația. În realizarea completă a proiectului intervin crearea tablei pe care se poziționează pionii în funcție de culoare, datul cu zarul, mutarea pionului dorit la click, trimiterea pe server a unui cod care să ilustreze acest eveniment, în cazul mutării pionului unui jucător peste pionul altui jucător se va trimite pionul deja existent pe acea poziție în casa de care aparține, iar ciclul se va continua până ce toți pionii unui jucător sunt în casa finală.

1.1.2 Definiții, acronime și abrevieri

În crearea proiectului am folosit cuvinte-cheie ușor de înțeles. Un exemplu concludent este reprezentat de numele PionCX, unde C=culoarea (R=roșu, G=galben, A=albastru, V=verde), iar X=1,2,3 sau 4; de asemenea, numerotarea jucătorilor s-a facut de la 0 la 3, unde 0=jucătorul roșu, 1=jucătorul galben, 2=jucătorul albastru, 3=jucătorul verde (s-a folosit numerotarea în crearea serverului pentru facilitarea înțelegerii codului). Pionii s-au numerotat ca indecși în vectorul ce reține imaginea corespunzătoare în ordinea numerotării jucătorilor: primii 4 sunt ai jucătorului roșu, numerotați 0-3, următorii, ai jucătorului galben, numerotați 4-7, următorii, ai jucătorului albastru 8-11, iar cei din urmă ai jucătorului verde, numerotați 12-15.

Vom numi casă de bază locul din care pornește fiecare pion, iar casă finală locul unde trebuie să ajungă pionii unui jucător pentru a câștiga.

1.1.3 Tehnologiile utilizate

S-au utilizat Paint pentru crearea unei table ale cărei poziții consecutive să fie aparent echidistante pentru facilitarea mutărilor și pentru crearea pionilor pentru fiecare jucător, zaruri și diagrama de clase prezentată în cap. 3 și Microsoft Visio Drawing pentru realizarea schemei logice a jocului.

1.2 Cerințe specifice

Funcționalitățile aplicației sunt următoarele: afișarea pionilor pe tablă, verificarea pionului în casa de început în funcție de culoare, datul cu zarul și apariția valorii corespunzătoare pe ecran în dreptul pionului al cărui rând este, mutarea pionilor on-click în funcție de zar, verificarea și mutarea pionului în casa finală în funcție de numărul de valori rămase de pe zar, trimiterea unui cod conținând culoarea+numărul pionului, coordonatele la care acesta se află, numărul de ordin al jucătorului la rând, afișarea jucătorului al cărui rând este în cazul unei

încercări de a da cu zarul din partea jucătorului nepotrivit, decodificarea mesajului de pe server pentru mutarea pionului corespunzător.

2 Funcționalitatea MutaPion

2.1 Descriere

O funcționalitate vitală pentru aplicația "Nu te supăra, frate!" o constituie mutarea pionilor (prin intermediul metodei MutaPion(int, int, int) din clasa Pozitii), întrucât întregul cod depinde de aceasta.

2.2 Fluxul de evenimente

2.2.1 Fluxul de bază

Funcționalitatea de față folosește coordonate de pe tabla de joc calculate ulterior. În metodă se primesc ca parametri 3 numere intregi: dx, reprezentând abscisa de pe tabla (left-ul imaginii manipulare), dy, reprezentând ordonata de pe tablă (top-ul imaginii manipulate) și valoareZar ce reține valoarea zarului dată aleator. În cazul în care zarul este 0, dx al obiectului și left-ul imaginii devin parametrul dx, iar dy al obiectului și top-ul imaginii devin parametrul dy.

Consideram că pozițiile consecutive sunt echidistante (diferența pe orizontal este de 60, iar pe vertical de 54). Dacă zarul are valoare mai mare decât zero și tabla nu a fost încă parcursă (condiție verificată de către metoda TablaParcursa din aceeași clasă), se modifica left-ul și top-ul în funcție de valoarea pixelilor, calculată de asemenea. Valoarea zarului se scade și la sfârșit, când zarul ajunge la 0, conform celor spuse mai sus, se modifică și dx și dy al pionului mutat cu coordonatele corespunzătoare.

Pentru a folosi funcționalitatea de MutaPion trebuie dat click pe zar pentru a genera o valoare aleatoare și a se da click pe pionul dorit, iar utilizatorul ar trebui să obțină o nouă poziție a pionului, poziție corespunzătoare valorii zarului.

Metoda este apelată:

- în constructorul clasei Poziții (care mută pionii fiecărui jucător în casa de bază) cu parametri dx și dy și valoarea zarului 0 (pentru a deveni left-ul imaginii pionului dx, iar top-ul dy);
- la click pe pionul care se dorește a fi mutat (dacă este rândul jucătorului pentru care se face mutarea) cu parametri left-ul actual, top-ul actual și valoarea zarului;
- la mutarea celorlalți jucători pe tabla unui jucător în rețea, analog la on-click pe pion.

2.2.2 Pre-condiții

Funcționalitatea curentă se comportă corespunzător dacă au fost toate clientele pornite (ale tuturor jucătorilor) și serverul. Nu s-a realizat o implementare pentru afișarea conectării la rețea.

2.2.3 Post-condiții

Funcționalitatea curentă are rolul de a muta pionii jucătorului al cărui rând este, așadar, efectul său este de a-i muta în locul corect în funcție de poziția actuală și de zar.

3 Funcționalitatea Client

3.1 Descriere

Această funcționalitate este o piesă-cheie în crearea rețelei:.lucrează ca punte între jucători datorită faptului că lucrează direct cu serverul.

3.2 Fluxul de evenimente

3.2.1 Fluxul de bază

Funcționalitatea de față folosește un cod trimis de către server. Întrucât un jucător își poate muta doar pionii proprii, este necesară implementarea unui cod care să permită vizionarea mutărilor celorlalți pioni. Se manipulează direct codul (care poate fi reprezentat doar de o cifră în cazul schimbării rândului jucătorului – în acest caz se reține doar rândul jucătorului) prin convertirea sa la double, divizată în două numere de tip int, astfel: un aux1 care conține numărul, left-ul și top-ul pionului și un aux2 care conține valoarea zarului, rândul jucătorului și două cifre care sugerează dacă left-ul, respectiv top-ul imaginii este un număr mai mare decât 100 pentru a ști cum să le calculăm în aux1. Se extrag apoi în alte variabile informațiile de care avem nevoie, se mută pionul jucătorului corespunzător pe coordonatele trimise pe server și se afișează valoarea zarului pe care a primit-o.

3.2.2 Pre-condiții

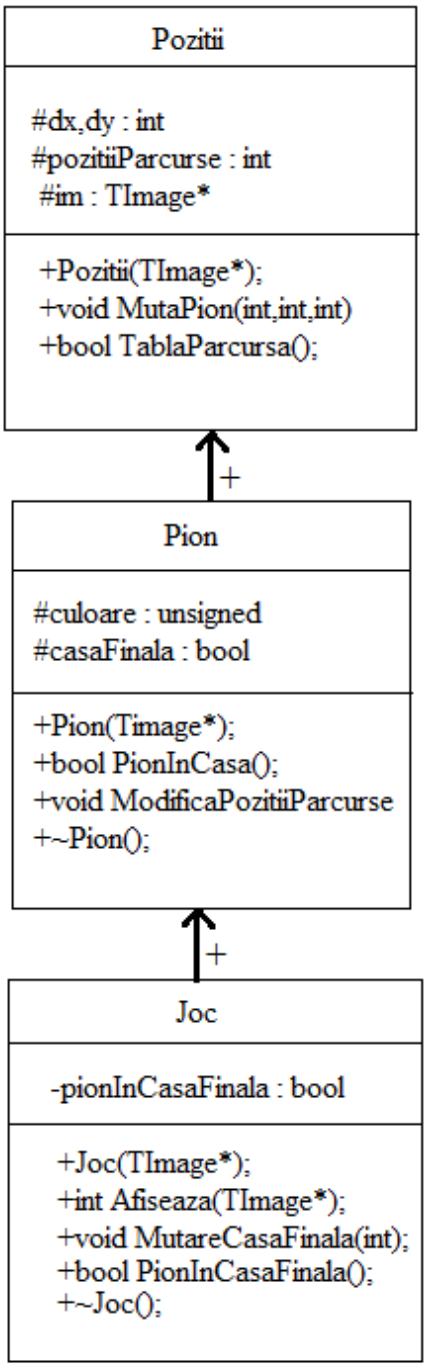
Clientul, respectiv mutările se comportă adecvat atât dacă au fost toate clientele pornite (ale tuturor jucătorilor) și serverul, cât și dacă se transmite corect codul (în funcție de factori externi dezvoltatorului sau utilizatorului, precum memoria). Nu s-a realizat o implementare pentru afișarea conectării la rețea.

3.2.3 Post-condiții

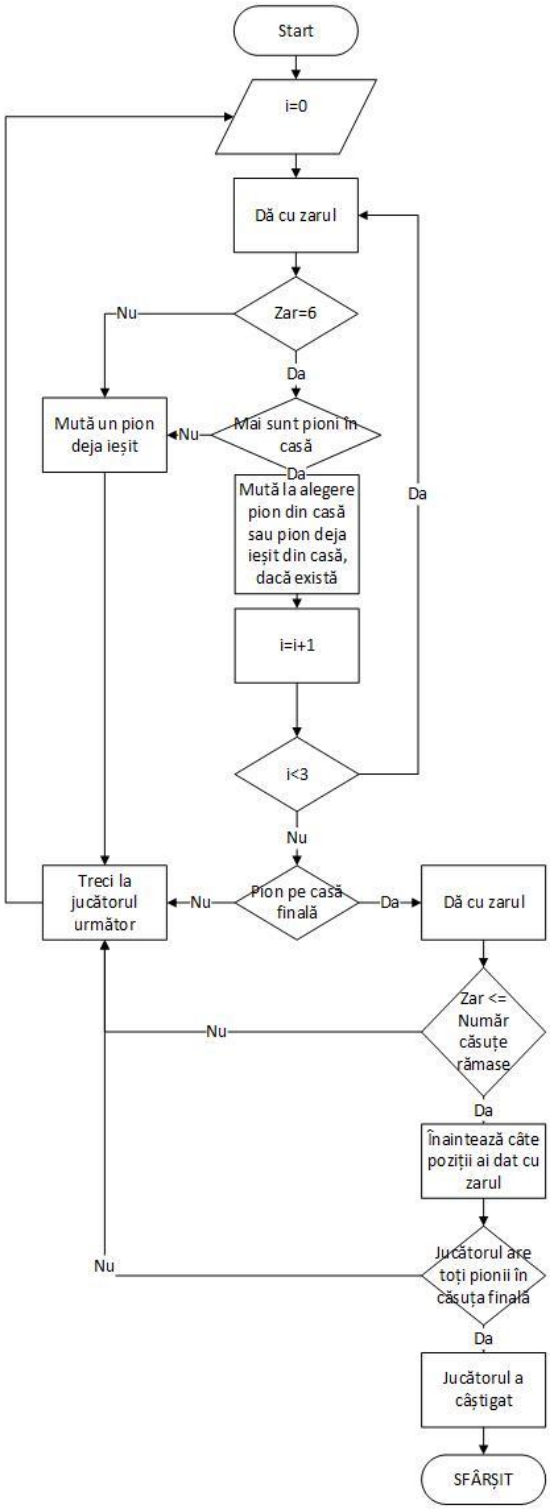
Această funcționalitate are rolul de a stabili rândul jucătorului (iar dacă este cazul, doar acest rol), valoarea zarului, numărul pionului mutat și coordonatele la care s-a mutat, efectul său fiind deci de a muta pionii celorlalți jucători pe tabla fiecaruia în funcție de ceea ce s-a transmis pe server.

4 Implementare

4.1 Diagrama de clase



4.2 Descriere detaliată



5 Bibliografie

http://www.functionx.com/cppbuilder/
https://www.visual-paradigm.com/guide/uml-unified-modeling-language/uml-classdiagram-tutorial/
Programare orientată pe obiect – Macarie Breazu

Menționez colegul Necșuleu Claudiu-Ionuț din grupa 222/2 care mi-a explicat cum se creează și cum funcționează corect rețeaua.