[Note R++]

[17.01.2023]

Îndrumător:

dr. ing. Daniel Morariu

Student:

Bolca Florin Rares

(224/2)

Istoric Versiuni

Data	Versiune	Descriere	Autor
20.12.2023	1.0	Prima versiune a proiectului	Bolca Florin Rares
09.01.2024	1.1	Modificări de clase și metode	Bolca Florin Rares
16.01.2024	1.2	Implementare clasa de căutare	Bolca Florin Rares

Cuprins

1 Specificarea cerințelor software

1.1 Introducere

Proiectul Note R++ este un simplu editor vizual de text. Note R++ a fost creat pentru a crea, edita, salva fișiere de tipul formatului ".txt". La prima vedere aplicația are o interfață grafică asemănătoare cu Microsoft Notepad de la care m-am inspirat cel mai mult :).

1.1.1 Obiective

Aplicația Note R++ va trebuii sa indeplinească urmatoarele cerinte:

- Convertirea textului din controlul CutieText în baza 16 (Hexa), respectic in baza 8 (Octal)
- Convertirea textului din controlul CutieText în baza 8 (Octal)
- Colorarea instrucțiunilor a limbajelor de programare

Pentru ca aplicația să fie considerată completă cerințele enumerate mai sus trebuie să fie îndeplinite. În prezent în versiunea 1.2 cele două obiective legate de convertirea în alte baze au fost îndeplinite Ultima cerință funcționează aproape perfect în proiect.
Aplicația NoteR++ trebuie să coloreze cuvintele de programare la fel cum o fac aplicațiile Visual Studio Community și Notepad++ ca exemplu.

1.1.2 Definiții, Acronime și Abrevieri

La conceperea proiectului am respectat conventia Pascal Case în mare parte la numirea variabilelor și a metodelor.
Dicționar:

1. CutieText - controlul RichTextBox din aplicație
2. BaraButoane - controlul menuStrip care contine toate butoanele ex: Fisier, Editare
3. ForEach - bucla de iterație folosită în C# pentru a repeta o secventa de instructiuni
4. int RichTextBox.Find(string str) - Metoda din C# care returneaza indexul primului caracter gasit in RichTextBox.
5. TabelaDeCautare - clasa folosita pentru a cauta si indexa cuvinte date ca pa dintr-un string dat ca parametru
6. KeywordComune - clasa din proiect care conține lista de culori și un dicționar de cuvinte
7. C++ - clasa din proiect care la momentul actual adăuga cuvinte în dicționarul din KeywordComune
8. Java - clasa din proiect care la momentul actual adăuga cuvinte în dicționarul din KeywordComune

1.1.3 Tehnologiile utilizate

Aplicatia a fost dezvoltata cu ajutorul mediului de dezvoltare integrat Visual Studio Community, limbajul de programare utilizat pentru dezvoltare este C#. Proiectul este dezvoltat exclusiv cu ajutorul pachetului standard C# din Visual Studio Community.

1.2 Cerințe specifice

Următoarele obiective au fost completate fi funcționează în aplicație:

- Convertire în Hexa
- Convertire în Octal
- Colorarea cuvintelor de programare

2 Funcționalitate

2.1 Descriere

Functionalitatea care scoate proiectul în evidenta este colorarea instrucțiunilor de limbaj de programare, funcția la momentul actual merge aproape perfect, are câteva “bug-uri” pe care nu le-am reușit să rezolv

2.2 Fluxul de evenimente

2.2.1 Fluxul de bază

Pentru a acționa funcționalitatea utilizatorul va trebui sa introducă cuvinte din limbaje de programare spre exemplu C++.

După poza data mai sus se pot vedea bug-urile la care m-am referit. Backbone-ul care realizează funcționalitatea din imagine sunt clasele TabelaDeCautare si KeywordComune. Clasa cea mai relevantă este TabelaDeCautare care poate fi utilizata și în alte proiecte care au nevoie de o indexare precisă a cuvintelor plus și alte caracteristici ale cuvantului ex lungimea, poziția în control.
Am creat clasa TabelaDeCautare deoarece pachetul standard C# nu mi-a permis găsesc poziția exactă în controlul CutieText, de exemplu metoda **int RichTextBox.Find(string str)** îmi returna doar poziția de start a primului cuvânt găsit, în teorie aveam nevoie de ceva care să țină minte fiecare cuvânt relevant și indexul lui din control un fel de Find modificat.

//constructor din clasa TabelaDeCautare

```
 public TabelaDeCautare(string Text, string[] CuvinteCheie)
 {
    string cuvant = String.Empty; //cuvânt dat ca parametru în metoda Add
    bool Prima = true; //bool folosit cu scopul de a seta poziția primului caracter din string-ul dat
    int PrimulCaracter = 0; //index pentru primul caracter
    for (int c = 0; Text.Length != c; c++)  //pana cand ajung la finalul textului executa
    {
       if (Text[c] != ' ')            //orice caracter diferit de spațiu formează un cuvant
       {
          cuvant += Text[c];     //formeaza cuvantul
          if (Prima) { PrimulCaracter = c; Prima = false; }//setam indexul primului      caracter
       }
       else { cuvant = String.Empty; Prima = true; }//daca caracterul este spațiu restam cuvantul format
       {
          if (Contains(cuvant, CuvinteCheie))//condiția verifica daca cuvantul se potrivește cu cuvintele
//dat de programator
          {
             //dacă da adăugăm cuvantul şi caracteristicile acestuia în tablouri
             Add(cuvant, cuvant.Length, PrimulCaracter, elemente_totale);
             cuvant = String.Empty;  //după adaugare resetam cuvantul
             Prima = true;
          }
       }
    }
 }
```

constructor din clasa TabelaDeCautare

Algoritmul de mai sus preia fiecare cuvant setat de programator şi pune caracteristicile acestuia în tablou de string, tablou de int, tablou de int.(Linia Add(cuvant,cuvant….))

<Flux alternativ>

```
//foreach (string Cuvant in CutieText.Text.Split(' '))
//{
//    //Daca cuvantul din text se potriveste in lista de cuvinte
//    //atunci coloram cuvantul
//    if (C.VerificaLista(Cuvant))
//    {
//        Color CuloareCuvant = C.CuloarePotrivita(Cuvant);
//       int IndexSelectie = CutieText.Find(Cuvant);
//        int Length = Cuvant.Length;
//        CutieText.SelectionColor = CuloareCuvant;
//        CutieText.SelectionStart = CutieText.TextLength;
//    }
//    CutieText.SelectionColor = Color.Black;
//    CutieText.SelectionLength = 0;
//}
```

Bucata de sus realizează colorarea cuvintelor în controlul CutieText, implementarea prezentată mai sus nu funcționa bine din cauza metodei int RichTextBox.Find(string str).

2.2.2 Pre-condiții

Pentru obținerea cat mai buna a functionalitatii utilizatorul va trebui sa lase spațiu între instrucțiuni.
Cat pentru celelalte funcționalități din aplicație:
Executarea funcționalității ConvertToHexa sau ConvertToOctal va rescrie orice fișier text “Hexa.txt” respectiv “Octal.txt”, utilizatorul este rugat să-și salveze convertirea antecedenta dacă aceasta există.
Pentru obținerea funcționării optime a conversiei în hexazecimal și octal utilizatorul va trebui să introducă în controlul CutieText un cuvânt sau mai multe pentru a obține rezultate cât mai bune.
Atenționez, în caz că nu se introduce nimic în controlul CutieText fișierul “Hexa.txt” va fi creat și va fi gol, același lucru și pentru funcționalitatea ConvertToOctal.

Pentru colorarea cuvintelor limbajelor de programare am avut mai multe modalități de implementare:

1.Prima fiind o bucla ForEach care preia fiecare cuvânt din text, stochează indexul și lungimea și coloreaza instant cuvantul.Implementarea ei este încă prezentă în versiunea 1.2 dar neutilizată.

2.A doua modalitate ar fi sa folosesc functionalitatea Regex din System.Regular.Expressions, modalitate de implementare pe care nu am inteles-o concret ca să o pot pune în practica, dar principiul de colorare fiind același cu prima implementare.

3.A treia ar fi să implementez o clasa care se ocupă cu indexarea cuvintelor.(cea utilizată în program)

2.2.3 Post-condiții

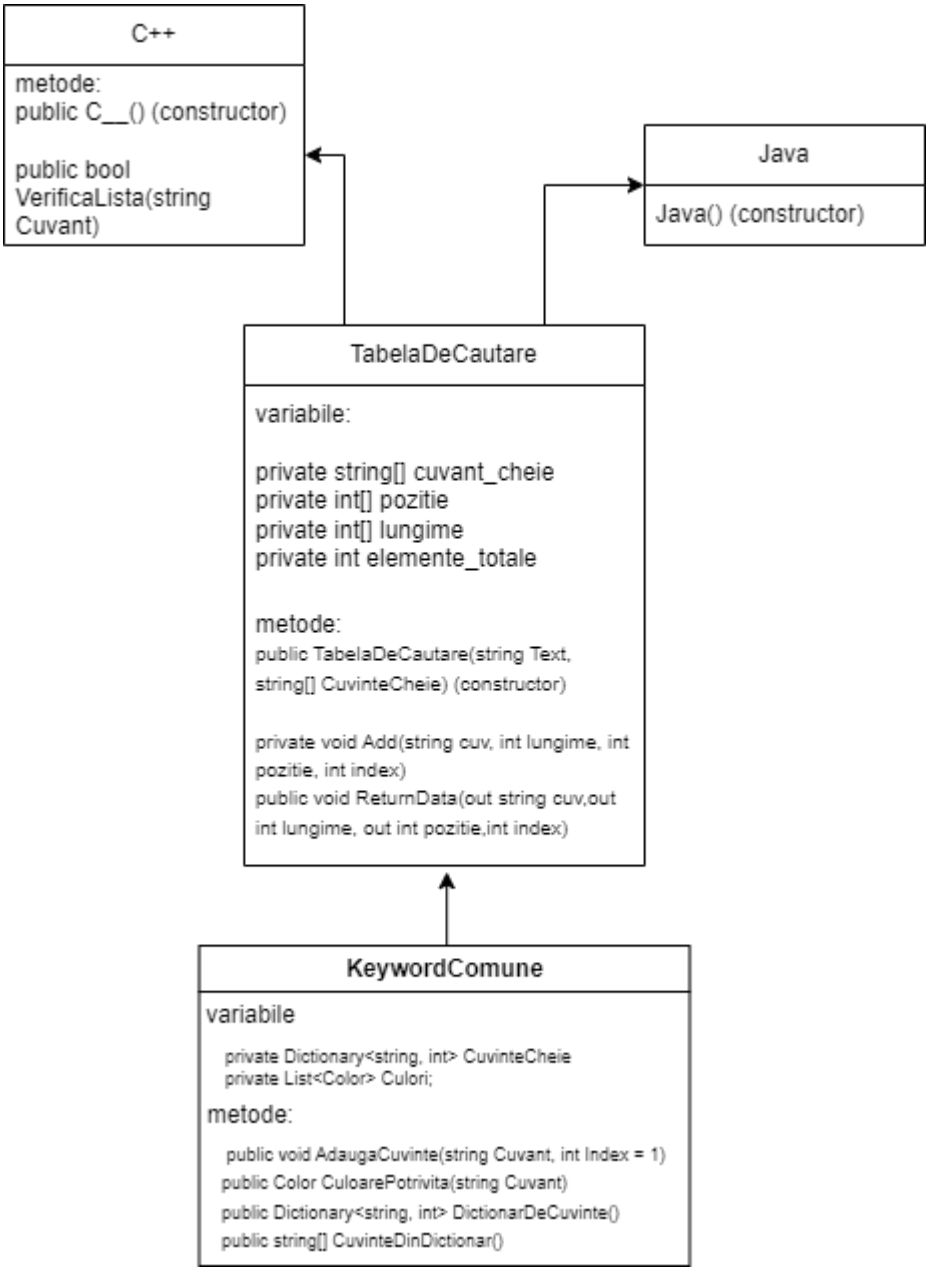
Utilizatorul ar trebui sa vadă în aplicație cum se colorează cuvintele instant în controlul CutieText.

2.2.4 Pentru Convertiri

Dacă funcționalitatea s-a executat corect utilizatorul ar trebui sa obțină un fișier cu numele “Hexa.txt” în locul în care acesta a decis să-l salveze.
Același caz va fi obținut și pentru funcționalitatea ConvertToOctal, doar că numele fișierului va fi “Octal.txt”.

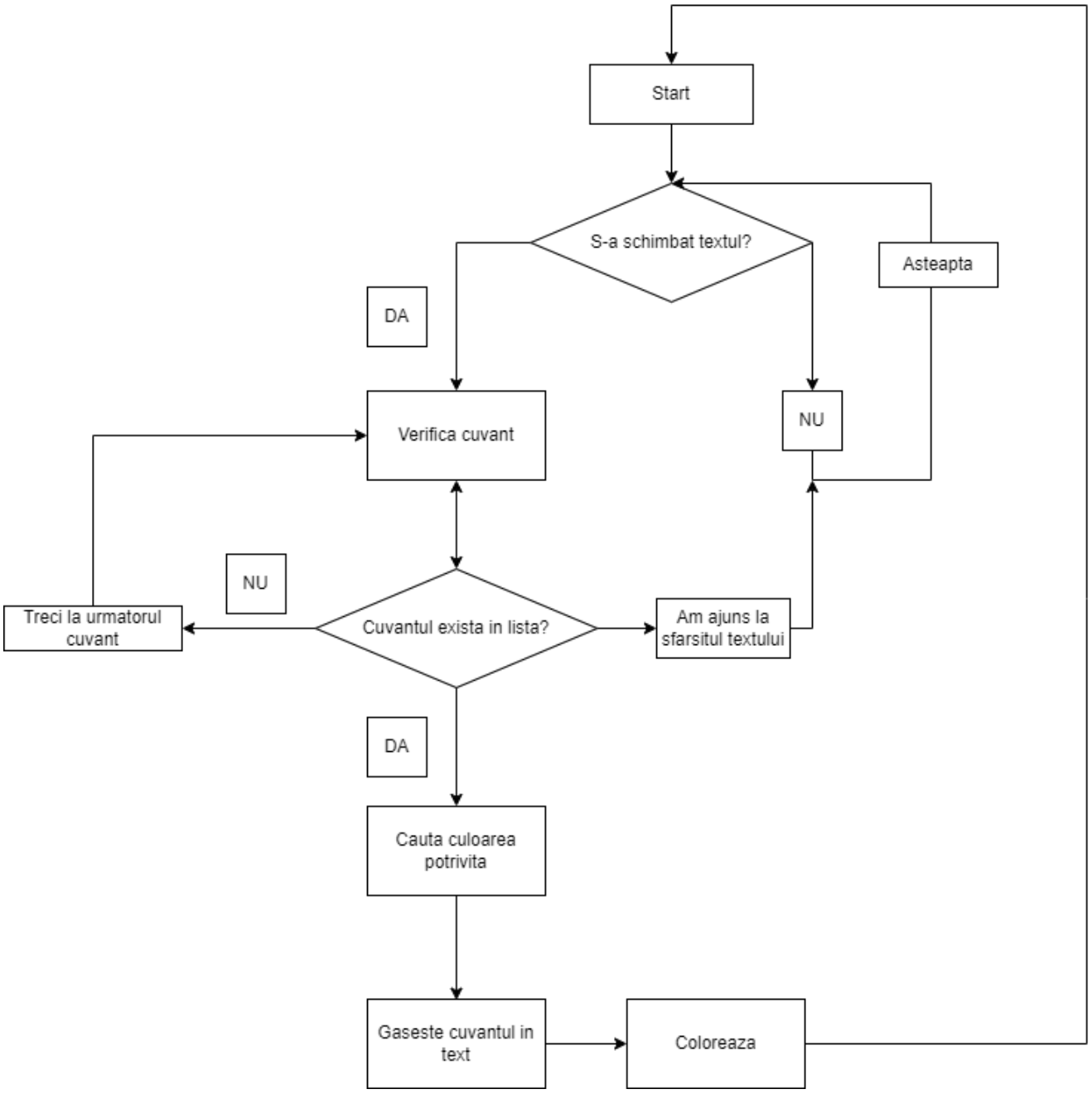
3 Implementare

3.1 Diagrama de clase



3.2 Descriere detaliată

Algoritmul propus de mine pentru colorarea cuvintelor a limbajelor de programare



4 Bibliografie

https://www.youtube.com/watch?v=KRelbsE7VTI&t=896s
https://learn.microsoft.com/en-us/dotnet/api/system.collections.generic.dictionary-2?view=net-8.0
https://www.c-sharpcorner.com/article/syntax-highlighting-in-rich-textbox-control-part-1/
https://app.diagrams.net (pentru desene)