**MINISTERUL EDUCATIEI NAȚIONALE**

**COLEGIUL NAȚIONAL “MIHAIL SADOVEANU” PAȘCANI**

**PROIECT PENTRU SUSȚINEREA EXAMENULUI DE ATESTARE A COMPETENȚELOR PROFESIONALE LA INFORMATICĂ**

**TITLU PROIECT:Spanzuratoarea**

**FILIERA: TEORETICA**

**PROFIL: REAL**

**SPECIALIZAREA: MATEMATICA-INFORMATICA, INTENSIV INFORMATICA**

**NUME CANDIDAT PROFESOR COORDONATOR**

**Bolota Constantin Cosmin Mitrica Eduard Gabriel**

**CLASA**

**XII-D**

**SESIUNEA: MAI 2020**

**Cuprins:**

1. **Tema proiectului** pag. 3

**2.Limbajul C#** pag. 9-19

**3.Structura aplicatiei** pag. 4-7

**4.Softul utilizat** pag. 19-28

**5.Resurse de hard si soft** pag. 29

**Tema proiectului**

Lucrarea de fata isi propun sa studieze functionalitatea, eficienta si utilitatea limbajului de programare C# prin compunerea unui cod si compilarea acestuia in practica.

Problema aleasa ca domeniu de studiu este jocul “*Spanzuratoarea*”, un joc emblematic pentru utilizatorii de calculator din toate grupele de varsta, iar grafica pentru ilustrarea “spanzuratorii” este realizata prin functia DrawBodyPart(). In continuare va vom prezenta caracteristicile si regulile jocului ilustrat:

**Informatii utile**:

* *Jocul “Spanzuratoarea” a fost adaptat pentru un singur utilizator al aplicatiei*
* *Se poate ghici atat o litera, cat si un grup de litere (sir de caractere)*

- Cuvântul ce trebuie ghicit este reprezentat de un şir de linii, fiecare linie reprezentând o literă a cuvântului.

- Dacă jucătorul care ghiceşte sugerează o literă ce se află în cuvânt, programul completează spatiile cu litera corespunzatoare în toate poziţiile unde aceasta apare.

- Dacă litera nu se află în cuvânt, programul desenează un element din diagrama „spânzurătoarea”.

- Limita de greseli este de: **9**

**Jocul se încheie** când:

* Jucătorul care ghiceşte, completează tot cuvântul, sau îl ghiceşte exact.
* Este completata diagrama.

**Limbajul C#**

Limbajul C# este unul dintre cele mai utilizate limbaje de programare multiparadigmă din lume, fiind clasat în acest moment pe locul 5 mondial, ca nivel de popularitate. Este un limbaj simplu, modern, cu o flexibilitate foarte mare în ceea ce privește dezvoltarea de aplicații și portabilitatea acestora (deși există unele voci care spun că portabilitatea este limitată, vom demonstra în continuare lipsa de veridicitate a acestei afirmații).

***1 – Date istorice***

Istoria acestui limbaj își are originile la începutul mileniului în care ne aflăm. În iulie 2000 a fost lansată prima distribuție a limbajului C#, deși zvonurile apariției unui limbaj puternic au apărut încă de prin anul precedent. Microsoft a fost corporația care a decis să intre mult mai în forță în lumea dezvoltatorilor, întrucât limbajul Basic își pierduse popularitatea ceea ce a dus la o ușoară scădere a nivelului economic al companiei.

Unul dintre principalele motive care au inițiat crearea acestui limbaj de programare a fost legat de faptul că în anul 1995, cei de la Sun Microsystems au creat limbajul Java, primul limbaj multiparadigmă cu o portabilitate foarte mare. A înregistrat un nivel de popularitate exponențial în primii ani de la lansare, și chiar în zilele noastre, este pe primul loc mondial în ceea ce privește popularitatea. Astfel, Microsoft a decis să creeze un limbaj care să ofere mult mai multe elemente dezvoltatorilor.

***2 – Caracteristicile limbajului***

Printre principalele calități ale limbajului, putem distinge:

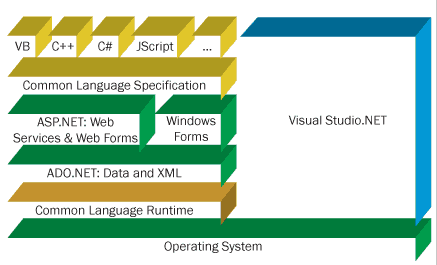
* modernitate, simplitate, utilitate generală, productivitate mare
* stabilitate în cadrul aplicațiilor complexe, durabilitate
* este un limbaj total orientat pe obiect (orice entitate din acest limbaj este de fapt un obiect)
* oferă suport complet pentru dezvoltarea de componente necesare în medii distribuite, deci este și un limbaj
* orientat către componente

Limbajul C# are aplicabilitate foarte mare în cadrul sistemelor embeded, care includ dispozitive precum mp3 playere, semafoare, automate (vending machines), controlere, telefoane mobile, ceasuri digitale, s.a.m.d.

Deci acum știți că atunci când sunteți opriți la semafor,  un algoritm transpus în C# coordonează controller-ul care dirijază fluxul de trafic.

***3 – Platforma .NET***

Portabilitatea limbajului se leagă în mod direct de platforma pe care rulează. Așa cum Java rulează în mediul său special, numit JVM (Java Virtual Machine), C# rulează în .NET Framework. Vom discuta acum despre această platformă, întrucât înțelegerea mecanismului care lucrează în spatele acesteia contribuie substanțial la înțelegerea funcționalității limbajului precum și la portabilitatea acestuia.



Să analizăm această diagramă (în partea stângă se află componentele platformei .NET, prezentate generic): pe primul nivel se află limbajele specifice .NET, printre care și C#. Al doilea nivel corespunde CLS-ului, adică un set de trăsături ale platformei, care asigură interoperabilitatea limbajelor .NET. Aceste trăsături pot se regăsesc și la nivel de aplicație, în sensul că developer-ul poate seta anumite feature-uri care să ajusteze execuția programului. Cu alte cuvinte, se poate controla modul de execuție al aplicației prin specificarea acestor elemente care indică particularitatea de execuție a aplicației pe care o creezi.

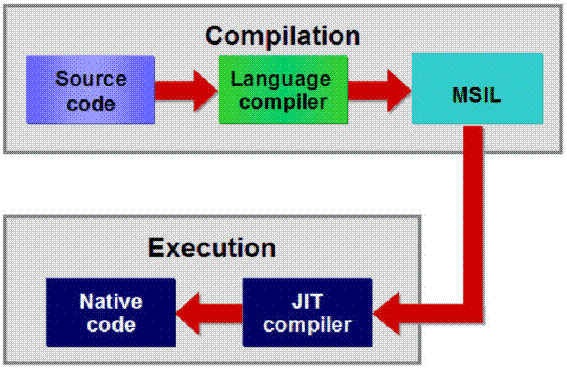
Dintre cele trei elemente de pe fundal verde, noi ne ocupăm doar de Windows Forms, adică de crearea de aplicații cu interfață grafică.

Common Language Runtime este probabil cea mai importantă componentă a platformei, responsabilă de executarea fiecărei aplicații. Practic CLR-ul este mediul de execuție al platformei. Acesta conține toate tool-urile care permit executarea raportată la sistemul de operare a aplicației dezvoltate, printre care: garbage collector-ul, adică mecanismul de gestionare automată a memoriei – spre deosebire de C sau C++, acolo unde programatorul trebuia să decidă în ce punct al aplicației să aloce o zonă de memorie, și în ce punct să elibereze acea zonă, C# are acest feature foarte important, care eliberează automat memoria ocupată de obiecte atunci când știe că acestea nu vor mai fi folosite în cadrul aplicației; mecanismul de tratare automată a excepțiilor, adică a erorilor runtime, cele care apar în timpul execuției programului. Desigur că programatorul trebuie să își creeze propriul sistem de tratare a erorilor de execuție, numai că există situații în care nu se acoperă toate cazurile în care execuția poate fi întreruptă, de aceea acest mecanism de tratare automată a excepțiilor face medierea între sistemul de operare și platforma .NET.

De asemenea o parte foarte importantă a platformei, alături de CLR, este și bibloteca de clase .NET sau *base class library,*pe scurt BCL. Practic, aici este tot codul creat de Microsoft pentru a construi limbajele .NET. Toate funcțiile, clasele, metodele, sunt aici în stare precompilată, ceea ce ușurează substanțial execuția programelor, timpul alocat pentru compilarea lor fiind practic inexistent.

***4 – Compilarea in .NET***

Să analizăm următoarea diagramă sugestivă:



Practic, codul scris în limbaj .NET este compilat cu ajutorul compilatorului special al limbajului. Acest compilator nu generează cod executabil, ci un cod specific platformei, denumit CIL, adică Common Intermediate Language. Pe diagramă scrie MSIL, de la Microsoft Intermediate Language. Acest cod specific platformei este de fapt nucleul portabilității limbajului. Oriunde este instalată platforma, ea poate rula cod în C#, întrucât acest CIL este direct dependent de platformă, și nu de sistemul de operare.

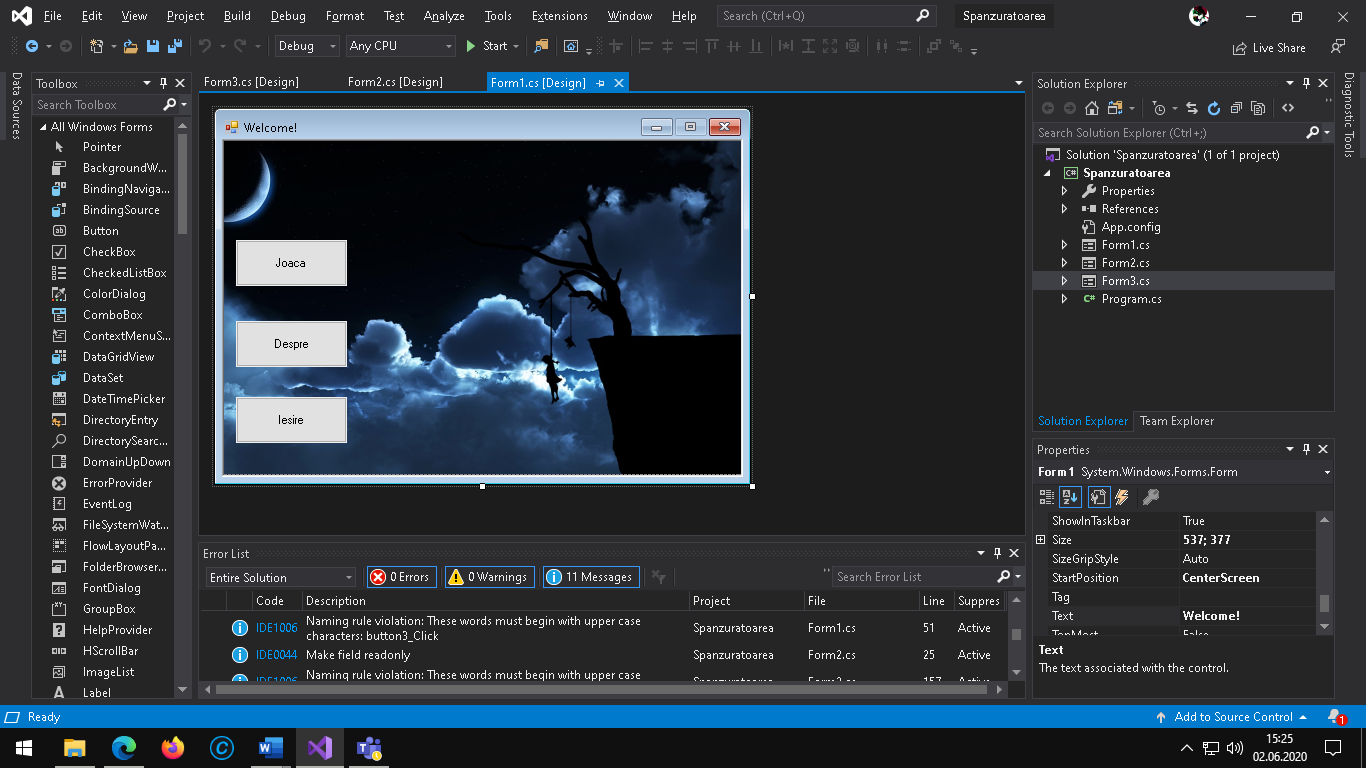
După compilare, intrăm în faza de execuție, deci CLR-ul intră în scenă. Codul intermediar, CIL-ul, intră într-o a doua compilare, dar de data aceasta compilatorul este unul special, denumit JIT (Just-In-Time), care generează codul în binar, accesibil mașinii de pe care se rulează.

Acestea sunt principalele elemente pe care orice programator în C# ar trebui să le cunoască, pentru a putea ajunge la un nivel rezonabil în cunoașterea limbajului. Evident, pentru cei pasionați, există sute de site-uri specializate pe acest limbaj, pagina de wikipedia, și multe alte surse, din care puteți învăța multe lucruri interesante despre elementele ce gravitează în jurul acestui limbaj.

**Structura aplicatiei**

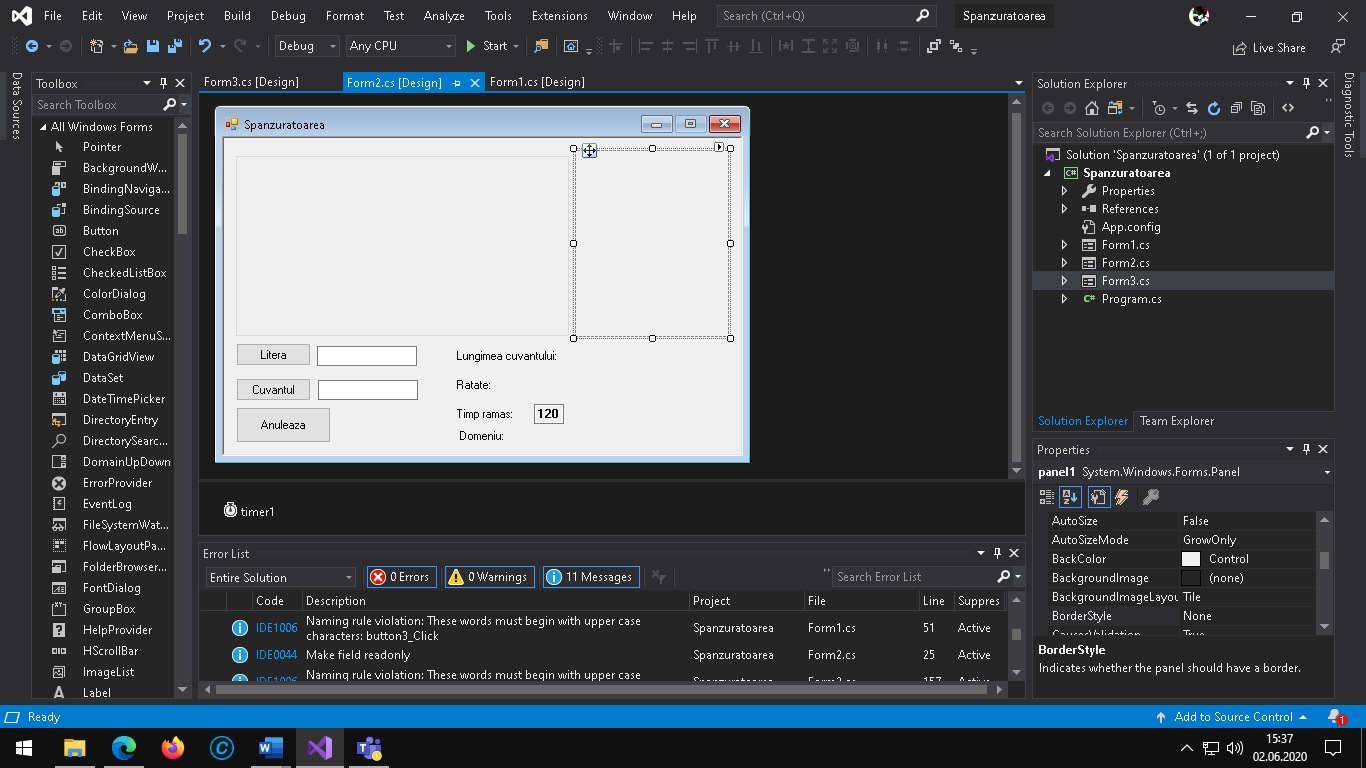
Aplicatia este alcătuită din 3 ferestre (Form-uri) astfel incat:

* Prima fereastra contine meniul principal alcatuit din butoanele: ”Joaca”, ”Despre” si „Iesire”.

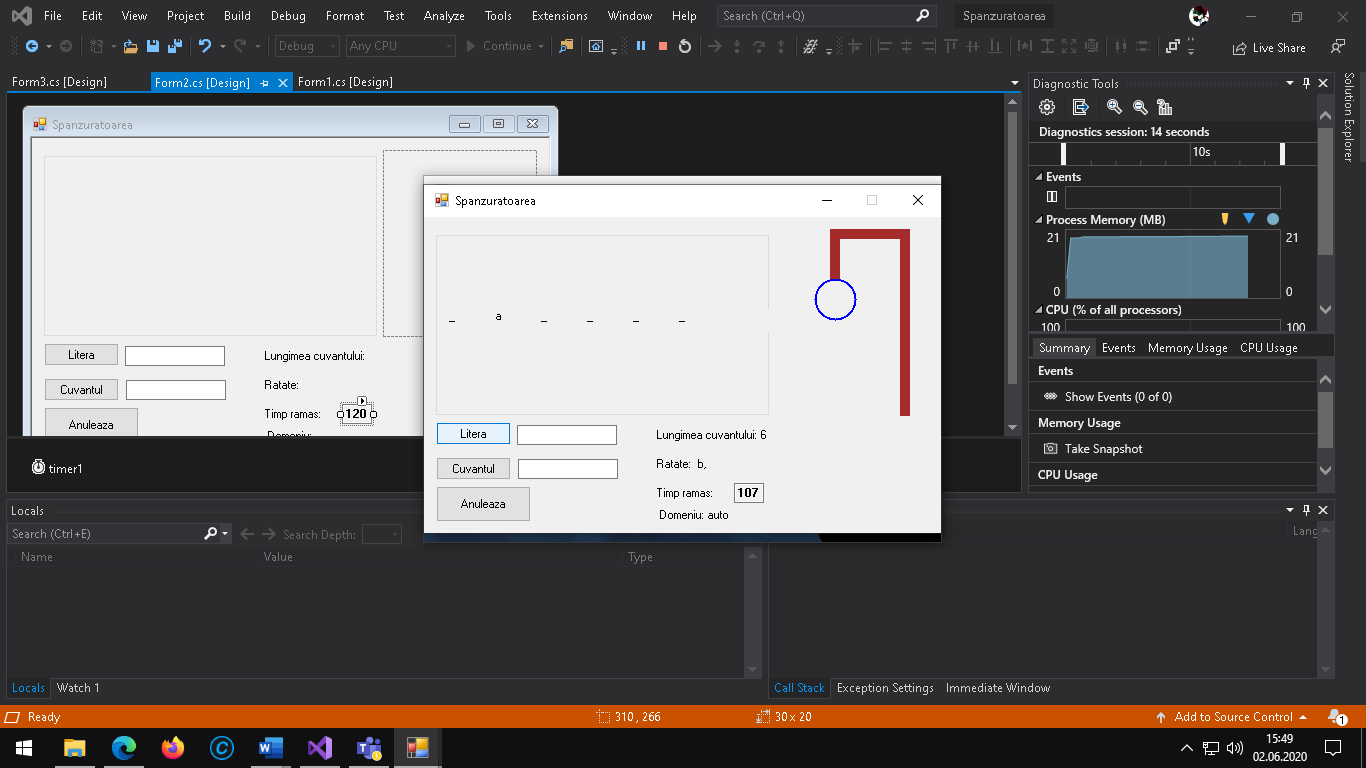


Butonul „Joaca” ne deschide a doua fereastra unde se desfasoara jocul, butonul „Iesire” paraseste jocul iar butonul „Despre” deschide a treia fereastra in care sunt descrise pe scurt regurile jocului.

* A doua fereastra contine butoanele: „Litera”, „Cuvant” si „Anuleaza” iar pe langa acestea mai contine si trei textbox-uri.



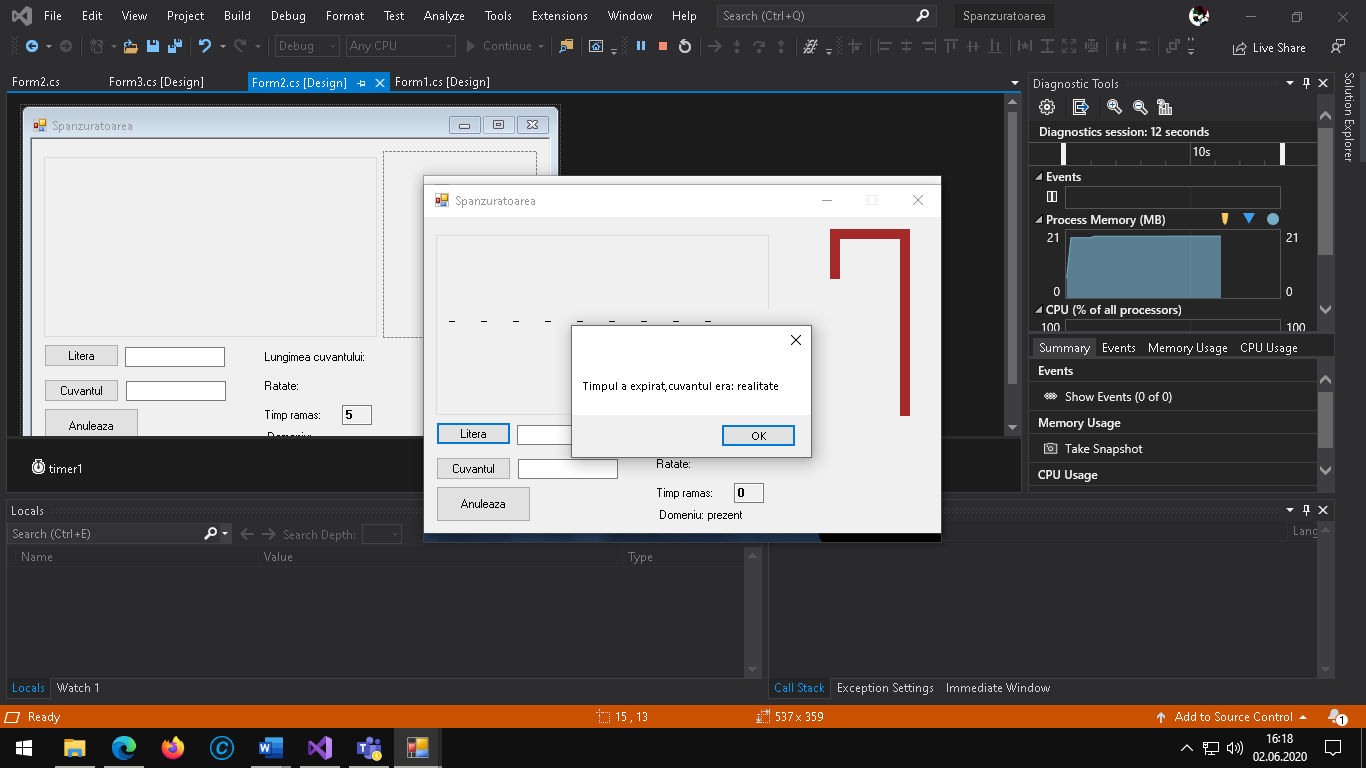
Butonul „Litera” verifica daca litera din primul textbox apare in text,astfel se va inlocui caracterul ”\_” cu litera curenta.



Al doilea buton „Cuvant” verifica daca cuvantul inserat in al doilea textbox este indentic fata de cuvantul cautat comparand cele doua cuvinte litera cu litera.

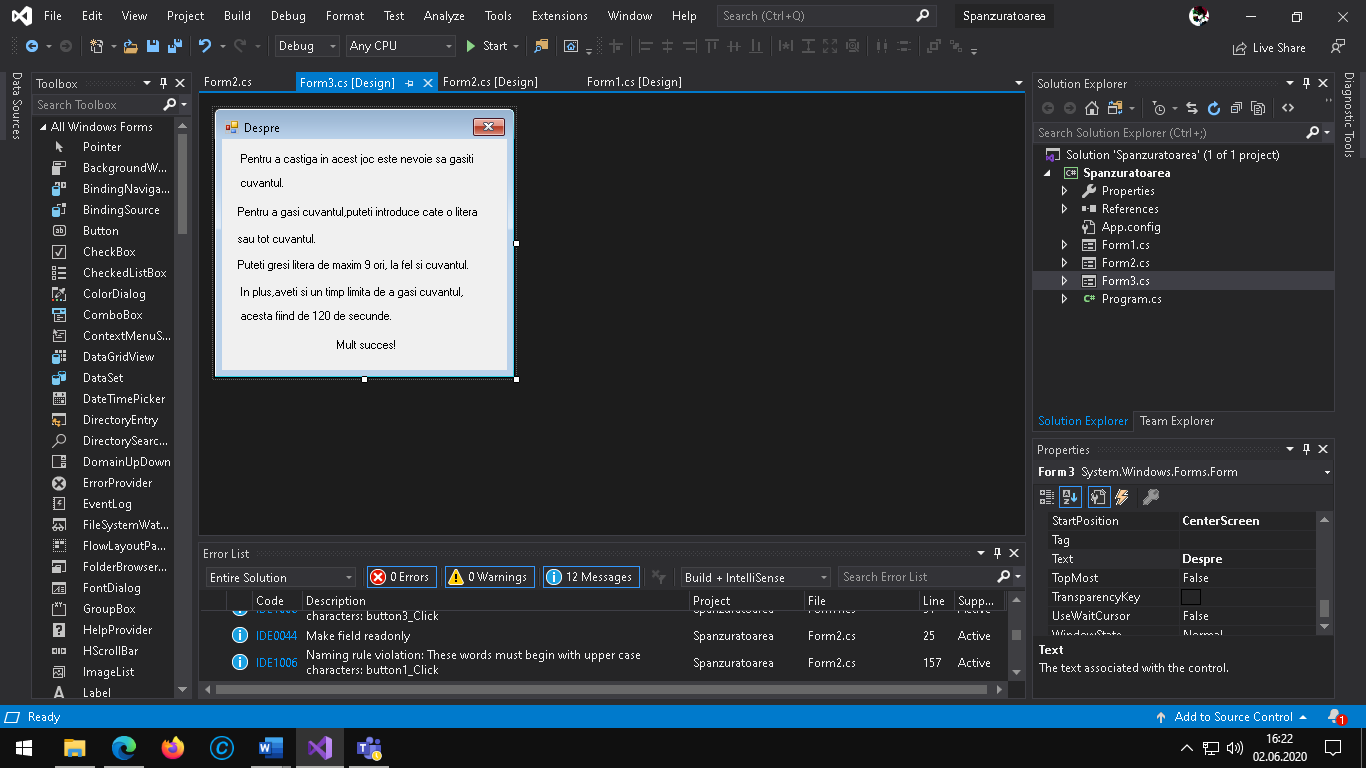
Al treilea buton paraseste fereastra curenta.

Pentru al treilea textbox este setat un timer de 120 de secunde,timpul maxim acordat jucatorului pentru a gasi cuvantul.La terminarea timpului va aparea un messagebox cu mesajul: „Timpul a expirat,cuvantul era: ’x’ urmat de cuvantul cautat.



Pe langa acestea, mai avem si patru label-uri in care primul contine lungimea cuvantului, al doilea contine literele ratate, al treilea este o extensie initiala pentru timerul din al treilea textbox iar al patrulea contine domeniul cuvantului.

* A treia fereastra contine o scurta descriere a jocului si a regulilor acestuia.



**Softul utilizat**

Microsoft Visual Studio este un mediu de dezvoltare integrat (IDE) de la Microsoft. Este utilizat pentru a dezvolta programe de calculator, precum și site-uri web, aplicații web, servicii web și aplicații mobile. Visual Studio folosește platforme de dezvoltare software Microsoft, cum ar fi Windows API, Windows Forms, Windows Presentation Foundation, Windows Store și Microsoft Silverlight. Poate produce atât codul nativ, cât și codul gestionat.

Visual Studio include un editor de cod care acceptă IntelliSense (componenta de completare a codului), precum și refactorizarea codului. Debuggerul integrat funcționează atât ca un depanator la nivel sursă, cât și ca un depanator la nivel de mașină. Alte instrumente încorporate includ un profilator de coduri, proiectant pentru construirea aplicațiilor GUI, designer web, designer de clase și proiectare de scheme de baze de date. Accepta plug-in-uri care îmbunătățesc funcționalitatea la aproape toate nivelurile - inclusiv adăugarea de suport pentru sistemele de control sursă (precum Subversion și Git) și adăugarea de noi seturi de instrumente precum editori și designeri vizuali pentru limbaje sau seturi de instrumente specifice domeniului pentru alte aspecte ale dezvoltării software ciclul de viață (cum ar fi clientul Azure DevOps: Team Explorer).

Visual Studio acceptă 36 de limbaje de programare diferite și permite editorului de coduri și depanatorului să sprijine (în grade diferite) aproape orice limbaj de programare, cu condiția să existe un serviciu specific limbajului. Limbile încorporate includ C, [8] C ++, C ++ / CLI, Visual Basic .NET, C #, F #, [9] JavaScript, TypeScript, XML, XSLT, HTML și CSS. Asistența pentru alte limbi, cum ar fi Python, [10] Ruby, Node.js și M, printre altele, este disponibilă prin pluginuri. Java (și J #) au fost acceptate în trecut.

Cea mai de bază ediție a Visual Studio, ediția comunitară, este disponibilă gratuit. Sloganul pentru ediția Visual Studio Community este „IDE gratuit, cu caracteristici complete pentru studenți, open-source și dezvoltatori individuali”.

Versiunea Visual Studio acceptată în prezent este 2019.

**Resurse de hard si soft**

**Minimul:**

Sistem de operare: Windows 7, 8 , 8.1, 10 x64

Procesor:Intel dual core 2.8 GHz

Ram: 4 GB

Disk:10 GB

**Recomandat:**

Sistem de operare: Windows 7, 8 , 8.1, 10 x64

Procesor:Intel quad core 2.6 GHz

Ram: 8 GB

Disk:30 GB