Echipa: Ștefănescu Anastasia– grupa 152, Luculescu Teodor – grupa 151, Ciobanu Dragoș – grupa 151, Dinu Aurelian – grupa 151

# **Descriere program Wordle**

**Conexiunea între două programe prin socket**

WinSock reprezintă API-ul celor de la Microsoft pentru a realiza conexiunea între două programe prin intermediul socketului în cadrul unui program C++.

În realizarea proiectului nostru am folosit API-ul WinSock2.

Pentru a realiza conexiunea, cele două programe au trebuit definite în două componente principale:

* Serverul – reprezentat de Jocul Wordle cu interfața grafică
* Clientul – reprezentat de Jucător

Conexiunea dintre cele două programe se face prin protocolul de Internet TCP, doar în cadru local.

Serverul:

* Inițializează API-ul (WSADATA wsData; WORD ver = MAKEWORD(2, 2);)
* Creează un socket (SOCKET listening = socket(AF\_INET, SOCK\_STREAM, 0);)
* Unește adresa ip cu portul dat (bind(listening, (sockaddr\*)&hint, sizeof(hint));)
* Îndreaptă socketul să asculte conexiuni noi (listen(listening, SOMAXCONN);)
* Așteaptă o conexiune nouă
  + Dacă conexiunea a fost realizată se creează un nou socket pentru trimiterea și primirea de date
  + socketul inițiat pentru ascultarea de conexiuni este închis
  + se intră în program
* La finalul progranului socketul de date și API-ul sunt închise. (closesocket(clientSocket); WSACleanup();)

Clientul:

* Definește adresa de ip locală 127.0.0.1 și portul 54000 la care serverul să asculte
* Inițializează API-ul (WSADATA wsData; WORD ver = MAKEWORD(2, 2);)
* Creează un socket (sock = socket(AF\_INET, SOCK\_STREAM, 0);)
* Se conectează la server (int connResult = connect(sock, (sockaddr\*)&hint, sizeof(hint));) și intră în program
* La finalul progranului socketul de date și API-ul sunt închise. (closesocket(sock); WSACleanup();)

Modul prin care comunică cele două programe este printr-un buffer de tip *char* în care se încarcă pe rând câte un cuvânt. Serverul așteaptă de la client cuvântul cu care începe jocul apoi serverul trimite către client un feedback printr-un cuvânt de aceeași lungime ca și cuvântul inițial trimis de forma „vvygy”, unde v reprezintă că litera de pe poziția respectivă a fost marcată cu verde de către joc, g reprezintă că litera de pe poziția respectivă a fost marcată cu galben, iar y reprezintă că litera de pe poziția respectivă a fost marcată cu gri. După fiecare primire și procesare al datelor, bufferul este golit prin comanda ZeroMemory(buf, 4096); . Acest proces continuă până când se ajunge la cuvântul corect, moment în care jocul nu mai acceptă input și trebuie închis manual de către utilizator.

**Interfaţa grafică (GUI) şi SFML:**

Pentru a realiza interfața grafică din proiectul Wordle, ne-am folosit de biblioteca Graphics din SFML (Simple and Fast Multimedia Library). Documentația acestuia se găsește aici: <https://www.sfml-dev.org/documentation/2.5.1/>.

**Structura jocului Wordle (serverul):**

1. Jocul este deshis într-o nouă fereastră de dimensiunea 1024x860 pixeli (sf::RenderWindow din SFML).
2. Se generează aleator un cuvânt din lista de cuvinte disponibilă (not. cuv\_rand).
3. Programul primește cuvântul de entropie maximă (not. cuv\_entr) de la client (jucătorul optim de wordle). În funcție de cuvântul ales avem două cazuri:
   * cuv\_rand = cuv\_entr, caz în care jocul rămâne deschis, dar nu mai comunică cu clientul, iar acesta poate fi doar închis.
   * cuv\_rand ≠ cuv\_entr, caz în care jocul trimite clientului un hash de forma “yygvy”, unde:
     + y – cuv\_rand[i] ≠ cuv\_entr[i]
     + g - cuv\_rand[i] = cuv\_entr[j], i ≠ j
     + v – cuv\_rand[i] = cuv\_entr[i]

și se reia pasul III).

**Structura Jucătorului (clientul):**

**Jucătorul**

Pentru a realiza programul-jucător am utilizat calculul entropiei:

E(I(x))=

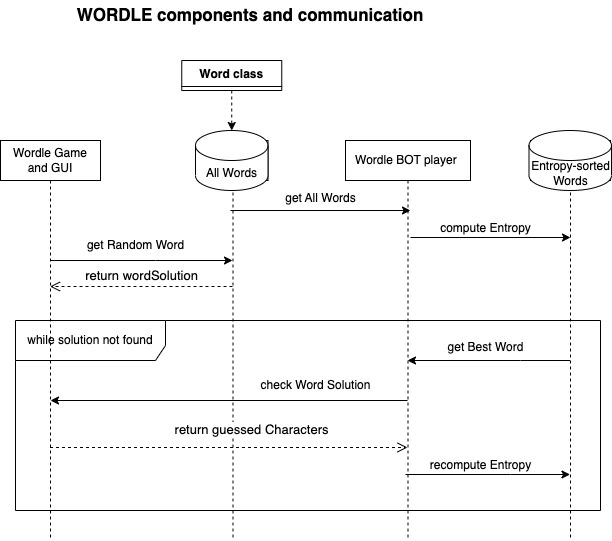
Pentru a utiliza această formulă am procedat astfel:

* Avem două liste de cuvinte: o listă cu toate cuvintele din baza de date (numită în continuare ”lista completă”) și o listă ce reprezintă o submulțime a primei liste și care se va actualiza pe măsură ce jucătorul face încercări (numită în continuare ”lista redusă”).
* Când un cuvânt este încercat (trimis către programul Wordle), se va primi înapoi un șir de 5 caractere din mulțimea {v, g, y} (v – verde, g – galben, y – gri) (de exemplu, ”vvgyv”). Numărul de astfel de șiruri, adică de răspunsuri posibile din partea jocului, este 35 (numărul de funcții f:{a, b, c, d, e} → {v, g, y}). Vom avea un tablou ”t” de dimensiune 35 = 243 ce va avea rol de vector de frecvențe. Pentru un cuvânt ales realizez o comparație cu toate cuvintele din lista redusă și incrementez tabloul pe poziția corespunzătoare rezultatului comparației. Obținem, astfel, pentru un cuvânt fixat, numărul de apariții ale fiecărui rezultat posibil (”vvvvv”, ”vvvvg”, ...). Pe baza acestui tablou se va calcula entropia corespunzătoare cuvântului ales E(I(x))=, unde n = 243, iar probabilitatea pi = , unde *l* este lungimea listei reduse de cuvinte (i. e. numărul de comparații realizate). Am obținut, astfel, entropia unui cuvânt din lista completă de cuvinte.

**Algoritmul**

Inițial cele două liste (lista completă și lista redusă) sunt egale.

1. Se calculează entropia fiecărui cuvânt (din lista completă).
2. Se alege cuvântul de entropie maximă și se trimite ca încercare către Wordle.
3. Pe baza răspunsului primit se actualizează lista redusă: se elimină acele cuvinte care nu mai pot fi cuvântul ascuns, care trebuie ghicit, adică cele care comparate cu ultimul cuvânt trimis către joc dau un rezultat diferit de acela primit de la Wordle.
4. Se repetă pașii 1., 2., 3. până când răspunsul primit de la Wordle este ”vvvvv” (adică am ghicit cuvântul corect).

**OOP:**

Pentru implementarea jocului am structurat codul in clase (folosind programarea orientata pe obiecte). Astfel informatia este structurata in jurul:

* Conceptului de cuvant - clasa Word – prin care sunt reprezentate sirurile de caractere de lungime 5 implicate in joc
* Conceptului de bază de date de cuvinte – clasa WordDatabase – prin care reținem informatiile legate de lista de cuvinte pe care o folosim pentru căutări

Numărul mediu de încercări: 4,14196

**Bibliografie:**

<https://bitbucket.org/sloankelly/youtube-source-repository/src/master/cpp/networking/?at=master>

<https://learn.microsoft.com/en-us/windows/win32/winsock/windows-sockets-start-page-2>

<https://www.nytimes.com/games/wordle/index.html>

<https://www.youtube.com/watch?v=C06eGdy7C6k&ab_channel=SurajSharma>

https://aditya-sengupta.github.io/coding/2022/01/13/wordle.html

https://towardsdatascience.com/information-theory-applied-to-wordle-b63b34a6538e

https://www.youtube.com/watch?v=v68zYyaEmEA

https://www.enjoyalgorithms.com/blog/introduction-to-oops-concepts-in-cpp

https://logicmojo.com/oops-concepts-in-cpp