*Colegiul Naţional:”Ion C. Brătianu”*

*Clasa a XII-a C*

**Proiect pentru atestatul profesional la informatică**

**Cu și despre cărți**

Elev: Pană Alexandra-Ioana

Profesor: Benghe Adrian

Cuprins

1) Concepte web

* HTML
* CSS
* JavaScript

2) Explicarea programului

3) Bibliografie

Concepte web

Internetul este o rețea globală de calculatoare interconectate, care permite comunicarea între milioane de utilizatori din întreaga lume. Rețeaua Internet a fost creată inițial ca o uriașă bază de date pentru a fi utilizată în scop științific și educațional. În acest scop, rolul predominant al rețelei Internet era cel de documentare și de comunicare. Ulterior, datorită creșterii numărului de calculatoare conectate la Internet, a crescut și numărul de site-uri ale diferitelor organizații care oferă conținut informațional și implicit numărul de vizitatori ai acestor site-uri, posibili consumatori de produse și servicii.

**HTML**

Definiția HTML este Hypertext Markup Language:

* HyperText este metoda prin care vă deplasați pe Internet - prin click pe texte speciale, numite hyperlink-uri, care fac legătura către alte pagini web. Această metodă conține în denumire textul hyper pentru a evidenția faptul că deplasarea pe Internet este neliniară, utilizatorii putând naviga oriunde pe Internet prin intermediul acestor hyperlink-uri.
* Markup face referire la marcarea textului dintr-un site într-un anumit mod (text boldat sau cursiv), prin intermediul tag-urilor HTML.

Cum funcționează?

HTML constă într-o serie de coduri scurte (numite tag-uri), tastate într-un fișier text de către autorul site-ului. Textul este apoi salvat ca un fișier HTML și vizualizat prin intermediul unui browser web, precum Internet Explorer, Mozila Firefox sau Google Chrome.

Acest browser citește fișierul și traduce textul într-o formă vizibilă, afișând pagina așa cum autorul și-ar fi dorit.

**CSS**

Limbajul HTML este folosit pentru a oferi o structură conținutului unei pagini web. Însă, el nu este de ajuns pentru a face un site atractiv și inovativ. În acest scop a fost creat limbajul CSS (Cascade Style Sheets), ce permite cosmetizarea unui site, dându-i exact forma dorită, indiferent de complexitatea design-ului grafic.

O foaie de stil sau un CSS este un set de reguli și caracteristici care, aplicate pe o pagină web, pot modifica aspectul său. În acest fel, putem cumva standardiza și automatiza munca necesară proiectării site-ului. Putem defini un stil pentru titluri și altul pentru text astfel încât să nu trebuiască să modificăm de fiecare dată textul șu titlurile pentru a avea aspectul dorit.

Liniile de cod CSS pot fi introduse în interiorul limbajului HTML ca atribute ale tag-urilor HTML, proces care se numește „inline style”. Însă acest lucru nu este recomandat deoarece ar cauza o încărcare mai înceată a site-ului. Ca urmare, este recomandat ca liniile de cod CSS să fie stocate într-un fișier cu formatul „.css” pentru o încărcare mai rapidă și o editare mai ușoară.

Beneficiile utilizării CSS sunt:

* ușurința în mentenanța și actualizarea site-ului;
* încărcarea mai rapidă a site-ului;
* oferirea unor opțiuni suplimentare de formatare;
* dimensiunea redusă a fișierelor site-ului;
* control bun al design-ului site-ului.

Un stil CSS este format din trei părți: selector, proprietate și valoarea acesteia.

Un selector reprezintă „obiectul” pe care dorim să îl modificăm cu ajutorul stilului. Selectorii pot fi un tag în sine precum „p” (care reprezintă paragrafele) sau „h1” (care reprezintă titlurile).

Selectorii pot fi și denumiri de clase. Acestea se folosesc atunci când dorim să modificăm mai multe elemente ce au aceeași clasă și nu toată categoria (precum selectorul „p”, care înseamnă toate paragrafele din site). Selectorul de tip clasă arată astfel: „.numeleClasei”. Selectorul poate fi și un id unic, prin care modificăm un singur element. Aceasta arată astfel: „#numeleIDului”.

Sintaxa unui stil este următoarea: **selector{proprietate:valoare;}**

**JavaScript**

JavaScript (JS) este un limbaj de programare obiect-orientat bazat pe conceptul prototipurilor. Este folosit mai ales pentru introducerea unor funcționalități în paginile web, codul JavaScript din aceste pagini fiind rulat de către browser. A fost dezvoltat inițial de către Brendan Eich de la Netscape Communications Corporations sub numele Mocha, apoi LiveScript, și denumit în final JavaScript.

În ciuda numelui și a unor similarități în sintaxă, între JavaScript și limbajul Java nu există nicio legătură. Ca și [Java](https://ro.wikipedia.org/wiki/Java), JavaScript are o sintaxă apropiată de cea a [limbajului C](https://ro.wikipedia.org/wiki/Limbajul_de_programare_C), dar are mai multe în comun cu [limbajul Self](https://ro.wikipedia.org/w/index.php?title=Limbajul_Self&action=edit&redlink=1) decât cu [Java](https://ro.wikipedia.org/wiki/Limbajul_Java).

Până la începutul lui 2005, ultima versiune existentă a fost JavaScript 1.5, care corespunde cu Ediția a 3-a a ECMA-262, [ECMAScript](https://ro.wikipedia.org/wiki/ECMAScript), cu alte cuvinte, o ediție standardizată de JavaScript. Versiunile de [Mozilla](https://ro.wikipedia.org/wiki/Mozilla) începând cu 1.8 Beta 1 au avut suport pentru [E4X](https://ro.wikipedia.org/w/index.php?title=E4X&action=edit&redlink=1), care este o extensie a limbajului care are de a face cu [XML](https://ro.wikipedia.org/wiki/XML), definit în standardul ECMA-357. Versiunea curentă de Mozilla, 1.8.1 (pe care sunt construite [Firefox](https://ro.wikipedia.org/wiki/Firefox) și [Thunderbird](https://ro.wikipedia.org/wiki/Thunderbird) versiunile 2.0) suportă JavaScript versiunea 1.7.

Cea mai des întâlnită utilizare a JavaScript este în scriptarea [paginilor web](https://ro.wikipedia.org/wiki/Pagin%C4%83_web). Programatorii web pot îngloba în paginile HTML script-uri pentru diverse activități cum ar fi verificarea datelor introduse de utilizatori sau crearea de meniuri și alte efecte animate.

Browserele rețin în memorie o reprezentare a unei pagini web sub forma unui [arbore](https://ro.wikipedia.org/wiki/Arbore) de [obiecte](https://ro.wikipedia.org/wiki/Obiect_(informatic%C4%83)) și pun la dispoziție aceste obiecte script-urilor JavaScript, care le pot citi și manipula. Arborele de obiecte poartă numele de [Document Object Model](https://ro.wikipedia.org/wiki/Document_Object_Model) sau [DOM](https://ro.wikipedia.org/wiki/DOM). Există un standard [W3C](https://ro.wikipedia.org/wiki/W3C) pentru DOM-ul pe care trebuie să îl pună la dispoziție un browser, ceea ce oferă premiza scrierii de script-uri [portabile](https://ro.wikipedia.org/w/index.php?title=Portabilitate&action=edit&redlink=1), care să funcționeze pe toate browserele. În practică, însă, standardul W3C pentru DOM este incomplet implementat. Deși tendința browserelor este de a se alinia standardului W3C, unele din acestea încă prezintă incompatibilități majore, cum este cazul [Internet Explorer](https://ro.wikipedia.org/wiki/Internet_Explorer).

O tehnică de construire a paginilor web tot mai întâlnită în ultimul timp este [AJAX](https://ro.wikipedia.org/wiki/AJAX), abreviere de la „Asynchronous JavaScript and XML”. Această tehnică constă în executarea de cereri [HTTP](https://ro.wikipedia.org/wiki/HTTP) în fundal, fără a reîncărca toată pagina web, și actualizarea numai anumitor porțiuni ale paginii prin manipularea DOM-ului paginii. Tehnica AJAX permite construirea unor interfețe web cu timp de răspuns mic, întrucât operația (costisitoare ca timp) de încărcare a unei pagini HTML complete este în mare parte eliminată. JavaScript este cu adevărat un limbaj dinamic. Rar este necesar să utilizați operatorul de evaluare pentru lucruri de genul acesta, pentru că, practic, puteți scrie orice doriți, dacă sintaxa este corectă și dacă ceea ce ați scris nu există, veți primi o eroare de execuție.

Internetul și HTML-ul există de foarte mult timp, iar limbajul în care se scriu paginile Web a fost standardizat de către un grup de aproximativ 300 de organizații membre din întreaga lume. Acest grup se numește Word Wide Web Consortiun sau consorțiul W3C. W3C a fost creat în octombrie 1994 pentru a „duce World Wide Web la întregul său potențial de dezvoltare a unor protocoale comune care să promoveze evoluția și asigurarea interoperabilității sale.” W3C este un consorțiu internațional, format din membri permanenți și parteneri, care are ca scop dezvoltarea de standarde pentru World Wide Web (HTML, XHTML, XML etc.) astfel încât Internetul să evolueze într-o singură direcție.

Recomandările consorțiului W3C au ca scop următoarele:

* îmbunătățirea interoperabilității site-urilor web în cadrul browser-elor web – permite vizualizarea paginilor web cu ajutorul diverselor dispozitive și browsere web, indiferent de tipul sau versiunea acestora.
* asigurarea unei accesibilități sporite – este esențial ca Internetul sp fie accesibil considerat un drept uman esențial și ca urmare toate persoanele (chiar dacă au anumite dizabilități sau sunt în vârstă sau se află în zone rurale sau țări în curs de dezvoltare) trebuie să aibă acces la aceste informații. Astfel, programatorii web și organizațiile care doresc să faciliteze accesul acestor categorii de persoane la informațiile prezentate în cadrul site-ului web.
* asigurarea unor declarații consecvente – documentele HTML trebuie să aibă o structură standard, astfel încât paginile web să aibă un aspect consecvent. Aceste declarații indică browser-ului versiunea HTML în care este scris documentul, setul de elemente și atribute care pot fi folosite, precum și diverse reguli privind structura documentului.

**Programarea orientată pe obiecte** (POO, uneori și Programarea orientată obiect, uneori denumită ca și în limba engleză, Object Oriented Programming (OOP)) este o [paradigmă de programare](https://ro.wikipedia.org/wiki/Paradigm%C4%83_de_programare), axată pe ideea încapsulării, adică grupării datelor și codului care operează asupra lor, într-o singură structură. Un alt concept important asociat programării orientate obiect este polimorfismul, care permite abstractizări ce permit o descriere conceptuală mai simplă a soluției.

Programarea orientată pe obiect (Programare Orientată Obiectual) este unul din cei mai importanți pași făcuți în evoluția [limbajelor de programare](https://ro.wikipedia.org/wiki/Limbaj_de_programare) spre o mai puternică abstractizare în implementarea programelor. Ea a apărut din necesitatea exprimării problemei într-un mod mai simplu, pentru a putea fi înțeleasă de cât mai mulți programatori. Astfel unitățile care alcătuiesc un program se apropie mai mult de modul nostru de a gândi decât modul de lucru al calculatorului. Până la apariția programării orientate pe obiect, programele erau implementate în [limbaje de programare](https://ro.wikipedia.org/wiki/Limbaj_de_programare) procedurale (C, Pascal) sau în limbaje care nici măcar nu ofereau o modalitate de grupare a instrucțiunilor în unități logice (funcții, proceduri) cum este cazul limbajului de asamblare (asembler). Altfel spus o problemă preluată din natură trebuia fragmentată în repetate rânduri astfel încât să se identifice elementele distincte, implementabile într-un limbaj de programare. O mare problemă a [programării procedurale](https://ro.wikipedia.org/wiki/Programare) era separarea datelor de unitățile care prelucrau datele (subrutinele), ceea ce făcea foarte dificilă extinderea și întreținerea unui program. Astfel s-a pus problema ca aceste două [entități](https://ro.wikipedia.org/wiki/Entitate) (date și subrutine) să fie grupate într-un anumit mod, astfel încât subrutinele să "știe" în permanență ce date prelucrează și, mai mult decât atât, ele să formeze un modul, adică o unitate care separă implementarea de interfață, ceea ce implică posibilitatea refolosirii codului. A apărut astfel conceptul de [clasă](https://ro.wikipedia.org/wiki/Clas%C4%83). [Clasa](https://ro.wikipedia.org/wiki/Clasa) realizează, în speță, ceea ce am văzut mai înainte: grupează datele și unitățile de prelucrare a acestora într-un modul, unindu-le astfel într-o entitate mult mai naturală. Deși tehnica se numește "Programare Orientată Obiectual", conceptul de bază al ei este [Clasa](https://ro.wikipedia.org/wiki/Clasa). Clasa, pe lângă faptul că abstractizează foarte mult analiza/sinteza problemei, are proprietatea de generalitate, ea desemnând o mulțime de obiecte care împart o serie de proprietăți.

Explicarea programului

HTML:

<!DOCTYPE html>

<html lang="en">

<head>

  <meta charset="UTF-8">

  <meta name="viewport" content="width=device-width, initial-scale=1.0">

  <meta http-equiv="X-UA-Compatible" content="ie=edge">

  <link href="style.css" rel="stylesheet"><!--am importat documentul css-->

  <script src="javascript.js" async></script><!--am importat documentul JavaScript-->

  <title>Cu si despre carti</title><!--titlul propriu-zis al paginii web-->

</head>

<body>

  <h1 class="page-title">Cu si despre carti</h1><!--titlul paginii web-->

  <div class="overlay-text visible">

    Click ca sa incepi

  </div>

  <div id="game-over-text" class="overlay-text">

    GAME OVER

    <span class="overlay-text-small">Incearca din nou</span>

  </div>

  <div id="victory-text" class="overlay-text">

    VICTORIE

    <span class="overlay-text-small">Incerca din nou</span>

  </div>

  <div class="game-container">

    <div class="game-info-container">

      <div class="game-info">

        Timp <span id="time-remaining">100</span>

      </div>

      <div class="game-info">

        Miscari <span id="flips">0</span>

      </div>

    </div>

    <!--fiecare div cu clasa card contine atributele unei cărți: partea din spate, cea din față și detaliile

celor două părți-->

    <div class="card">

      <div class="card-back card-face">

       <img class="sweets" src="Imagini/bomboana.png">

      </div>

      <div class="card-front card-face">

        <img class="card-value" src="Imagini/bufnita.png">

      </div>

    </div>

    <div class="card">

      <div class="card-back card-face">

        <img class="sweets" src="Imagini/bomboana.png">

      </div>

      <div class="card-front card-face">

        <img class="card-value" src="Imagini/bufnita.png">

      </div>

    </div>

    <div class="card">

      <div class="card-back card-face">

        <img class="sweets" src="Imagini/bomboana.png">

      </div>

      <div class="card-front card-face">

        <img class="card-value" src="Imagini/avion.png">

      </div>

    </div>

    <div class="card">

      <div class="card-back card-face">

        <img class="sweets" src="Imagini/bomboana.png">

      </div>

      <div class="card-front card-face">

        <img class="card-value" src="Imagini/avion.png">

      </div>

    </div>

    <div class="card">

      <div class="card-back card-face">

        <img class="sweets" src="Imagini/bomboana.png">

      </div>

      <div class="card-front card-face">

        <img class="card-value" src="Imagini/cheie.png">

      </div>

    </div>

      <div class="card">

        <div class="card-back card-face">

          <img class="sweets" src="Imagini/bomboana.png">

        </div>

        <div class="card-front card-face">

          <img class="card-value" src="Imagini/cheie.png">

        </div>

      </div>

    <div class="card">

      <div class="card-back card-face">

        <img class="sweets" src="Imagini/bomboana.png">

      </div>

      <div class="card-front card-face">

        <img class="card-value" src="Imagini/veverita.png">

      </div>

    </div>

    <div class="card">

      <div class="card-back card-face">

        <img class="sweets" src="Imagini/bomboana.png">

      </div>

      <div class="card-front card-face">

        <img class="card-value" src="Imagini/veverita.png">

      </div>

    </div>

    <div class="card">

      <div class="card-back card-face">

        <img class="sweets" src="Imagini/bomboana.png">

      </div>

      <div class="card-front card-face">

        <img class="card-value" src="Imagini/pisica.png">

      </div>

    </div>

    <div class="card">

      <div class="card-back card-face">

        <img class="sweets" src="Imagini/bomboana.png">

      </div>

      <div class="card-front card-face">

        <img class="card-value" src="Imagini/pisica.png">

      </div>

    </div>

    <div class="card">

      <div class="card-back card-face">

        <img class="sweets" src="Imagini/bomboana.png">

      </div>

      <div class="card-front card-face">

        <img class="card-value" src="Imagini/inima.png">

      </div>

    </div>

    <div class="card">

      <div class="card-back card-face">

        <img class="sweets" src="Imagini/bomboana.png">

      </div>

      <div class="card-front card-face">

        <img class="card-value" src="Imagini/inima.png">

      </div>

    </div>

    <div class="card">

      <div class="card-back card-face">

        <img class="sweets" src="Imagini/bomboana.png">

      </div>

      <div class="card-front card-face">

        <img class="card-value" src="Imagini/scorpion.png">

      </div>

    </div>

    <div class="card">

      <div class="card-back card-face">

        <img class="sweets" src="Imagini/bomboana.png">

      </div>

      <div class="card-front card-face">

        <img class="card-value" src="Imagini/scorpion.png">

      </div>

    </div>

    <div class="card">

      <div class="card-back card-face">

        <img class="sweets" src="Imagini/bomboana.png">

      </div>

      <div class="card-front card-face">

        <img class="card-value" src="Imagini/cal.png">

      </div>

    </div>

    <div class="card">

      <div class="card-back card-face">

        <img class="sweets" src="Imagini/bomboana.png">

      </div>

      <div class="card-front card-face">

        <img class="card-value" src="Imagini/cal.png">

      </div>

    </div>

  </div>

<p id="rights">© Pana Alexandra</p>

</body>

</html>

CSS:

@font-face {

    font-family: "SuperFun";

    src: url("Fonturi/FBSuperFun.woff") format("woff"),

         url("Fonturi/FBSuperFun.woff2") format("woff2");

}

@font-face {

    font-family: "SM";

    src: url("Fonturi/Sunday Morning.woff") format("woff"),

         url("Fonturi/Sunday Morning.woff2") format("woff2");

}

/\*am importat stilurile de scris pentru titluri, timp și mișcări\*/

\* {

  box-sizing: border-box;

}

html {

  min-height: 100vh;/\*fundalul colorat în gradient mai jos va fi văzut în întregime pe ecran\*/

  font-family: SuperFun;

}

body {

  margin: 0;/\*fără margini\*/

  background: radial-gradient(rgb(163, 45, 81),rgb(204, 111, 139));

    /\*fundalul este colorat în gradient de la prima culoare la a doua\*/

}

.page-title {

  color:beige;

  font-family: SM, serif;/\*primul font este unul din cele importate mai sus, iar al doilea este pus în caz că primul nu se încarcă\*/

  font-weight: normal;

  text-align: center;

  font-size: 6em;/\*mărimea titlului\*/

}

.game-info-container {

  /\*aliniază conținutul astfel încât primul rând să conțină parametrii jocului\*/

  grid-column: 1 / -1;

  display: flex;

  justify-content: space-between;

}

.game-info {

  color: #FFBB89;

  font-size: 4em;

}

.game-container {

  margin: 50px auto;

  display: grid;/\*se aseaza cartile in coloane\*/

  grid-template-columns: repeat(4, auto);/\*in fiecare coloana sunt cate 4 carti\*/

  grid-gap: 10px;/\*spatiul dintre cartile de joc\*/

  justify-content: center;

  perspective: 500px;

}

.card {

  position: relative;

  height: 175px;

  width: 125px;

}

.card-face {

  position: absolute;

  display: flex;

  justify-content: center;/\*alinierea conținutului în centrul cărții de joc\*/

  align-items: center;

  width: 100%;

  height: 100%;

  border-radius: 12px;

  border-width: 1px;

  border-style: solid;

  overflow: hidden;/\*tot ce iese în exteriorul cărții de joc nu se vede\*/

  transition: transform 500ms ease-in-out;/\*tranziția pe care o face bomboana de pe cartea de joc\*/

  backface-visibility: hidden;

}

.card.visible .card-back {

  transform: rotateY(-180deg); /\*rotește cartea de joc\*/

}

.card.visible .card-front {

  transform: rotateY(0);

}

.card.matched .card-front .card-value {

  animation: dance 1s linear infinite 500ms;/\*animație când este găsită o pereche de cărți\*/

}

.card-back {

  background-color:lightskyblue;

  border-color: #FF6D00;

  transform: rotateY(0);

}

.sweets {

  align-self: flex-center;

  transition: transform 100ms ease-in-out;

  transform: translateY(-10px);

}

.card-back:hover .sweets {

  transform: translateY(0);

}

.card-value {

  position: relative;

  transition: transform 100ms ease-in-out;

  transform: scale(.9);

  width:100px;

}

.card-front:hover .card-value {

  transform: scale(1);

}

.card-front {

  background-color: #FFBB89;

  border-color: #333;

  transform: rotateY(180deg);

}

.overlay-text {

  top: 0;

  left: 0;

  right: 0;

  bottom: 0;

  z-index: 100;

  display: none;/\*textul va apărea doar la începutul/sfârșitul jocului\*/

  position: fixed;

  justify-content: center;

  align-items: center;

  flex-direction: column;

  color:beige;

  font-family: SM, serif;

  transition: background-color 500ms, font-size 500ms;

}

.overlay-text-small {

  font-size: .3em;

}

.overlay-text.visible {

  display: flex;

  animation: overlay-grow 500ms forwards;

}

@keyframes dance {

  /\*modul în care se mișcă desenele din interiorul cărților de joc în cazul în care se găsește o pereche\*/

  0%, 100% {

    transform: rotate(0)

  }

  25% {

    transform: rotate(-30deg)

  }

  75% {

    transform: rotate(30deg)

  }

}

@keyframes overlay-grow /\*animația care introduce textul de început/încheiere al jocului\*/

{

  from {

    background-color: rgba(0, 0, 0, 0);

    font-size: 0;

  }

  to {

    background-color: rgba(0, 0, 0, .8);

    font-size: 10em;

  }

}

@media (max-width: 600px)

/\*încadrează jocul în pagina web pentru a putea fi vizionat și pe ecrane mai mici(ex.:cele ale telefoanelor)\*/

{

  .game-container {

    grid-template-columns: repeat(2, auto)

  }

  .game-info-container {

    flex-direction: column;

    align-items: center;

  }

}

#rights{

  font-size:2em;

  text-align:center;

}

JavaScript:

class MixOrMatch {

    constructor(totalTime, cards) //se creează un nou obiect

    {

        this.cardsArray = cards;

        this.totalTime = totalTime;

        this.timeRemaining = totalTime;

        this.timer = document.getElementById('time-remaining')

        this.ticker = document.getElementById('flips');

    }

    startGame() {//de fiecare dată când se începe un joc se resetează parametrii

        this.totalClicks = 0;

        this.timeRemaining = this.totalTime;

        this.cardToCheck = null;//cartea de joc căreia i se caută perechea

        this.matchedCards = [];

        this.busy = true;

        setTimeout(() => {

            this.shuffleCards(this.cardsArray);

            this.countdown = this.startCountdown();

            this.busy = false;

        }, 500)

        this.hideCards();

        this.timer.innerText = this.timeRemaining;

        this.ticker.innerText = this.totalClicks;

    }

    startCountdown() {

        return setInterval(() => {//se începe numărătoare inversă a celor 100 de secunde

            this.timeRemaining--;

            this.timer.innerText = this.timeRemaining;

            if(this.timeRemaining === 0)//când se termină timpul jocul este pierdut

                this.gameOver();

        }, 1000);

    }

    gameOver() {//Această funcție afișează animația de Game over.

        clearInterval(this.countdown);

        document.getElementById('game-over-text').classList.add('visible');

        this.hideCards();//după trecerea timpului se întorc cărțile la loc

    }

    victory() {//Această funcție afișează animația asociată victoriei.

        clearInterval(this.countdown);

        document.getElementById('victory-text').classList.add('visible');

        this.hideCards();//după victorie se întorc cărțile din nou

    }

    hideCards() {//În cazul în care 2 cărți de joc nu sunt identice se apelează această funcție.

        this.cardsArray.forEach(card => {

            card.classList.remove('visible');

            card.classList.remove('matched');

        });

    }

    flipCard(card) {

        if(this.canFlipCard(card)) {

            this.totalClicks++;

            this.ticker.innerText = this.totalClicks;//se actualizează numărul de mișcări

            card.classList.add('visible');//se întoarce cartea

            if(this.cardToCheck) {//se verifică dacă sunt 2 cărți sau e doar una

                this.checkForCardMatch(card);

            } else {

                this.cardToCheck = card;

            }

        }

    }

    checkForCardMatch(card) {//Această funcție verifică dacă cele 2 cărți de joc sunt identice.

        if(this.getCardType(card) === this.getCardType(this.cardToCheck))

            this.cardMatch(card, this.cardToCheck);

        else

            this.cardMismatch(card, this.cardToCheck);

        this.cardToCheck = null;

    }

    cardMatch(card1, card2) {

        this.matchedCards.push(card1);

        this.matchedCards.push(card2);

        card1.classList.add('matched');//se execută animația din CSS când se găsesc 2 cărți

asemenea

        card2.classList.add('matched');

        if(this.matchedCards.length === this.cardsArray.length)

        //când s-au găsit toate perechile de cărți se execută funcția ”victory”

            this.victory();

    }

    cardMismatch(card1, card2) {//cazul în care cărțile de joc selectate sunt diferite

        this.busy = true;

        setTimeout(() => {

            card1.classList.remove('visible');

            card2.classList.remove('visible');

            this.busy = false;

        }, 1000);//se acordă timp jucătorului pentru a reține desenele de pe cărți

    }

    shuffleCards(cardsArray) { // Algoritmul Fisher-Yates Shuffle (amestecă cele 16 cărți de joc)

        for (let i = cardsArray.length - 1; i > 0; i--) {

            let randIndex = Math.floor(Math.random() \* (i + 1));

            cardsArray[randIndex].style.order = i;

            cardsArray[i].style.order = randIndex;

        }

    }

    getCardType(card) {//tipul cărții de joc(imaginea de pe interior)

        return card.getElementsByClassName('card-value')[0].src;

    }

    canFlipCard(card) {//se verifică dacă se poate întoarce cartea

        return !this.busy && !this.matchedCards.includes(card) && card !== this.cardToCheck;

        /\*sunt trei cazuri în care nu se poate da click:

        1) se execută o animație

        2) cărțile selectate au deja perechea găsită

        3) nu se poate întoarce cartea căreia i se caută perechea

        \*/

    }

}

if (document.readyState == 'loading')

//JavaScript se va încărca după ce se încarcă HTML și CSS în pagina web

{

    document.addEventListener('DOMContentLoaded', ready);

} else {

    ready();

}

function ready() {

    let overlays = Array.from(document.getElementsByClassName('overlay-text'));

    let cards = Array.from(document.getElementsByClassName('card'));

    let game = new MixOrMatch(100, cards);

    overlays.forEach(overlay => {//se trece de la ecranul de pornire și se începe jocul

        overlay.addEventListener('click', () => {

            overlay.classList.remove('visible');

            game.startGame();

        });

    });

    cards.forEach(card => {

        card.addEventListener('click', () => {

            game.flipCard(card);//se întoarce cartea

        });

    });

}

Bibliografie

1. Manea George Cristian, *ECDL Web Editing –KompoZer 0.8–,* Editura ECDL ROMANIA, București, 2018

2. <https://ro.wikipedia.org/wiki/JavaScript>

Pentru fonturi:

3. <https://www.dafont.com/sunday-morning-5.font>

4. <https://www.dafont.com/fbsuperfun.font>

5. <https://www.dafont.com/sketch-3d.font>

6.<https://www.imgonline.com.ua/eng/crop-photo-into-various-shapes.php>