**Documentatie proiect PMP**

**-**An 3, sem I-

Robert-Gabriel Teodorescu

Grupa 30238

1. **Prezentarea generala a proiectului**

Pentru elaborarea proiectului am ales ideea dezvoltarii unui joc in care utilizatorul trebuie sa controleze un personaj astfel incat sa evite obstacole ce vin pe traseu. Controlul personajului se face printr-un buton si consta in posibilitatea de a-l face sa sara cu scopul evitarii atingerii frontale cu un obstacol. Traseul este format din obstacole si spatii. Un obstacol a fost definit ca un caracter ce are forma unui patrat “umplut”, pentru a putea fi reperat usor pe ecran. Obstacolele si spatiile sunt generate succesiv, in blocuri.

Jocul este format din trei nivele, fiecare cu un nivel de dificultate mai mare decat precedentul. Nivelul de dificultate este proportional cu viteza de joc a fiecarui nivel. Nivelele 2 si 3 sunt mai intai blocate, intrucat necesita un anumit scor obtinut de utilizator pentru a putea fi deblocate. Fiecare nivel dispune si de notiunea de cel mai bun scor, care se actualizeaza la fiecare depasire a celui mai bun scor de catre utilizator.

Totodata, progresul realizat de utilizator in urma unei runde de joc este salvat intr-un fisier text aflat in memoria calculatorului la care este conectata placa. Astfel, la urmatoarea rulare a codului jocului pe placa, utilizatorul poate relua jocul din punctul in care l-a lasat. Acest proces are loc printr-o comunicare seriala cu un program Java, ce este capabil sa citeasca informatiile din acel fisier si sa le trimita programului Arduino care va procesa informatiile primite, si sa primeasca informatii din partea programului Arduino pentru a le procesa si sa le suprascrie in fisierul text. Fisierul contine, pentru fiecare nivel, cel mai bun scor obitnut si starea de blocat/deblocat. Daca este deblocat, se va scrie “true”, iar daca este blocat, se va scrie “false”. Fiecare din aceste informatii sunt despartite prin cate un spatiu. Exmeplu:

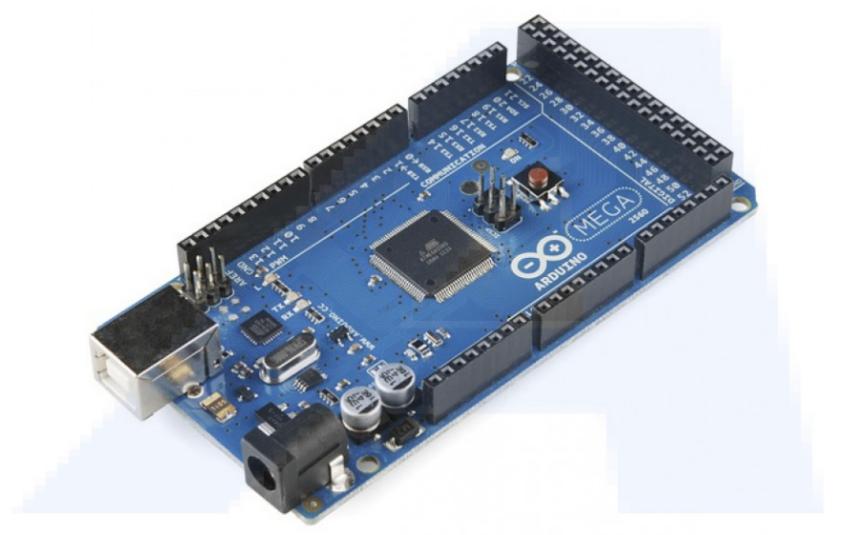
*60 true 30 true 0 false*

De precizat este faptul ca jocul poate fi rulat si fara aceasta comunicare seriala, doar ca la fiecare rulare a jocului pe placa informatiile despre nivele vor fi initializate cu niste valori default.

1. **Cerinte hardware**

Pentru elaborarea proiectului am avut nevoie de urmatoarele componente:

- placa Arduino Mega 2560



- shield LCD 16x2



- placa 4 butoane



- 4 fire tata-mama



1. **Cerinte software**

Pentru rularea proiectului sunt necesare urmatoarele componente:

- Arduino IDE : <https://www.arduino.cc/en/software>

-JDK1.8: <https://www.oracle.com/java/technologies/javase/javase-jdk8-downloads.html>

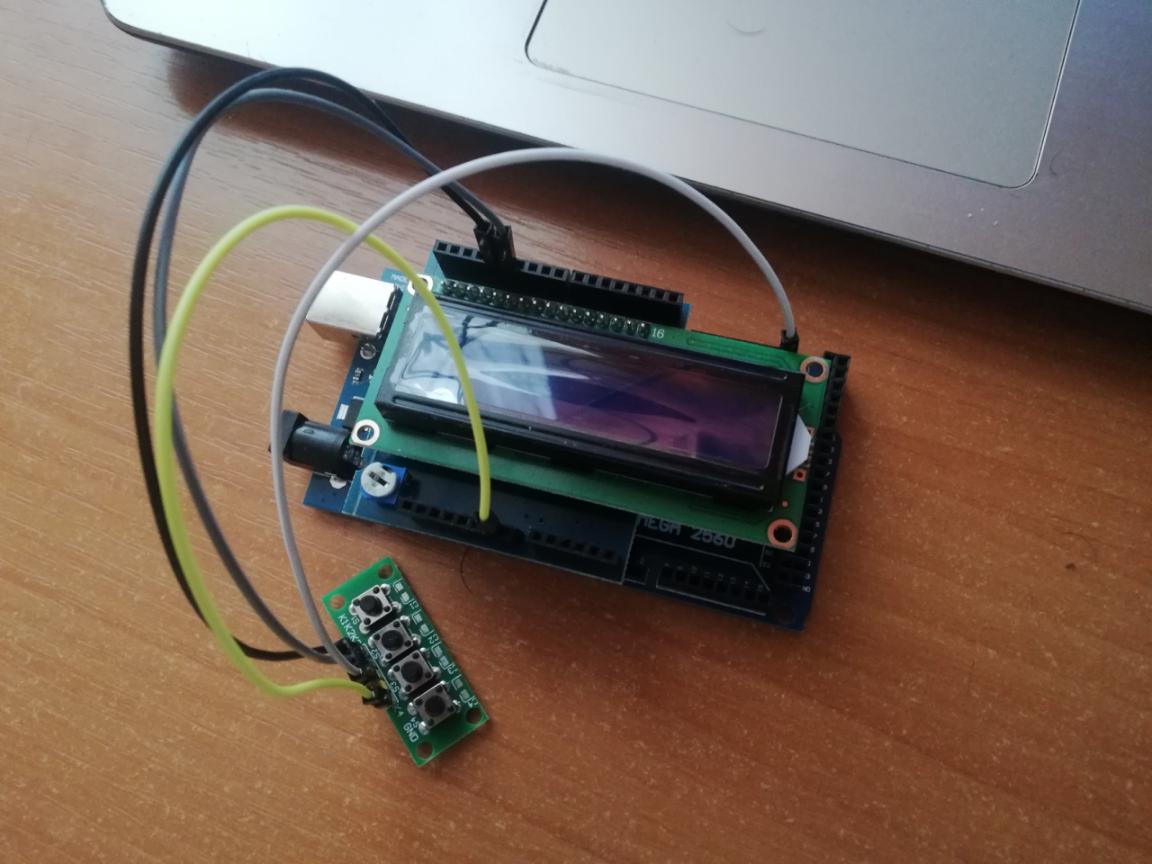
- un editor text pentru scrierea programului Java, sau o optiune mai fiabila este instalarea unui IDE, cum ar fi IntelliJ:

<https://www.jetbrains.com/idea/download/>

- libraria JSerialComm, necesara pentru realizarea comunicarii seriale:

<https://fazecast.github.io/jSerialComm/>

1. **Montaj**

****

**-**butonul K1 conectat la pinul 13

-butonul K2 conectat la pinul 12

-butonul K3 conectat la pinul 21

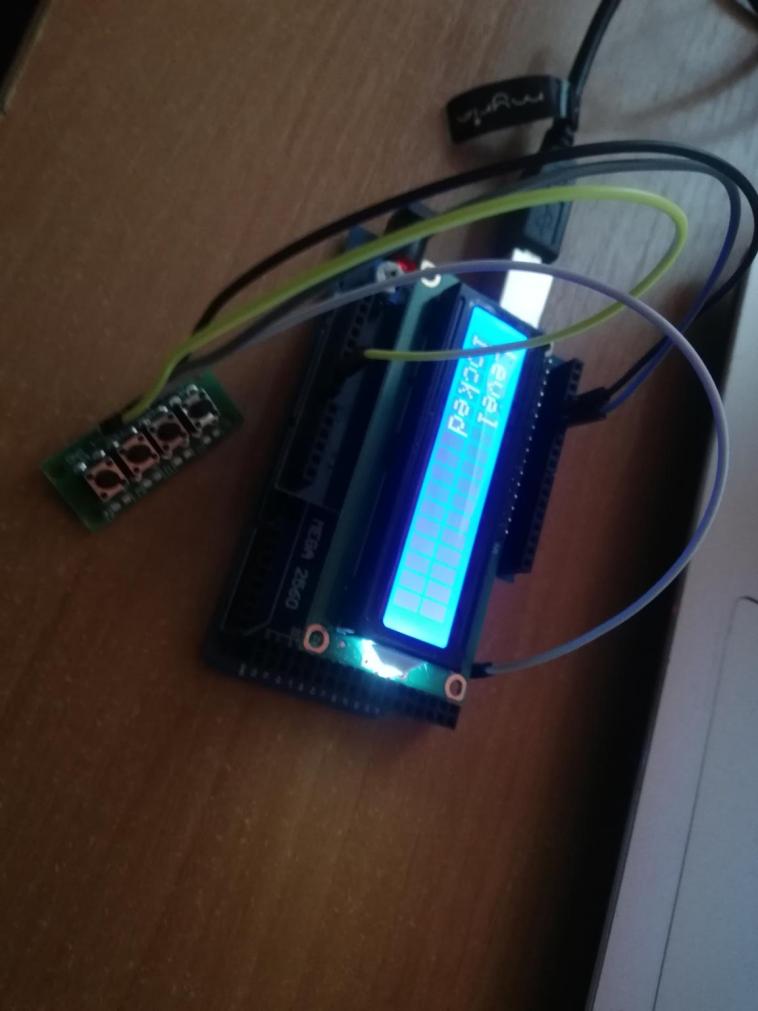
-pinul GND conectat la pinul GND din pinii POWER

1. **Interfetele jocului**

**-** meniul de start: in acest meniu, utilizatorul poate alege nivelul pe care doreste sa-l joace, dupa care trebuie sa apese pe butonul K2 pentru a incepe nivelul; daca nivelul este deblocat, atunci va aparea interfata jocului, iar daca este blocat, va aparea un mesaj care sa confirme ca nivelul este blocat; utilizatorul poate alege nivelul in urma apasarii butonului K1; la fiecare apasare a butonului, se deplaseaza underscore-ul catre urmatorul nivel



- mesaj care confirma ca nivelul este blocat, dupa care se revine la meniul de start



- interfata jocului



- mesaj care sa confirme ca utilizatorul a pierdut



- afisarea scorului recent obtinut si a celui mai bun scor pentru acel nivel; in urma apasarii butonului K2, se revine la meniul de start



1. **Explicatii cod**

-liniile 6-14:

Se defineste structura pentru nivele. Fiecare nivel va avea un cel mai bun scor obtinut, un sir care arata daca este blocat sau nu, un scor necesar pentru a putea fi deblocat, obtinut la nivelul anterior, o constanta folosita la generarea blocurilor de caractere de pe traseu si o viteza de rulare jocului, ce se transpune in viteza de deplasare a caracterelor de pe traseu.

-liniile 16-58:

Se definesc caracterele folosite pentru realizarea animatiei. Pentru personaj au fost definite 2 caractere ce seamana cu un om. Aceste 2 caractere vor fi afisare alternativ pentru a crea impresia ca personajul alearga pe traseu. Pentru obstacol a fost definit un dreptunghi plin, iar pentru spatiile din traseu a fost definit un caracter gol.

-liniile 60-81:

Sunt definite variabilele folosite in cod. Rolul este explicat in comentariul atasat fiecarei variabile.

-liniile 84-152:

Este definita functia ce afiseaza meniul de start. Este afisat mesajul “Select a level” urmat de cifrele 1, 2, 3, reprezentative pentru fiecare nivel, In variabila pickLevel este retinuta valoarea nivelului curent selectat. Aceasta se incrementeaza la fiecare apasare a butonului K1. Dupa ce utilizatorul apasa butonul K2, se verifica daca nivelul selectat este deblocat. Daca este deblocat, se seteaza pe true variabilele de control ce vor permite aplicatiei sa treaca la interfata de joc. Este creat si un debounce pentru butonul K1, realizat cu ajutorul a 2 variabile in care se vor retine valoarea curenta si valoarea recenta a butonului.

-liniile 155-180

Este definita functia ce actualizeaza informatiile despre nivele in urma ultimei runde de joc. Daca scorul obtinut este mai mare decat cel mai bun scor obtinut la acel nivel pana in momentul respectiv, atunci se actualizeaza cel mai bun scor obtinut. Se verifica daca scorul obtinut este necesar pentru deblocarea urmatorului nivel. In sirul info se introduc datele actualizate ale nivelelor si va fi trimis pe cale seriala catre programul Java.

-liniile 183-217

Este definita functia ce va afisa cele 2 mesaje dupa pierderea jocului. Prima data va fi afisat timp de 2 secunde mesajul care confirma ca jocul a fost pierdut, dupa care este setata variabila showScore pe true pentru ca, la urmatoare iteratie a functiei, programul sa treaca la ramura in care se afiseaza scorul obitnut si cel mai bun scor obtinut pentru acel nivel. Acest mesaj va ramane pe ecran cat timp nu este apasat butonul K2. Dupa apasarea butonului se va seta pe true variabila de control care va determina programul sa afiseze meniul de start.

-liniile 220-243

Este definita functia ce va citi informatia venita pe calea seriala de la programul Java. Cat timp sunt caractere pe calea seriala, salvam caracterul citit si il concatem la un sir. Intrucat informatiile din sir sunt despartite prin spatiu, pentru a putea salva informatiile fiecarui nivel trebuie facuta o separare a sirului. Aceasta separare se poate realiza prin functia strtok. Astfel, intr-un while, se desparte sirul actual prin spatiu pentru a ajunge la informatia dorita. Informatia se salveaza in vectorul de structuri definit pentru nivele.

-liniile 246-259

Este definita functia de initializare a jocului. Se umple vectorul definit pentru traseu cu codificarea definita pentru un spatiu. Se initializeaza variabilele de control folosite ulterior apelarii acestei functii in loop().

-liniile 262-270

Este definita functia care realizeaza deplasarea caracterelor de pe traseu cu o pozitie spre stanga, urmand ca pe ultima pozitie din dreapta sa intre codificarea urmatorului caracter de pe traseu.

-liniiile 273-292

Este definita functia care afiseaza personajul pe ecran. Daca variabila jumping este false inseamna ca personajul nu se afla in aer. Astfel, acesta va fi afisat pe randul de jos al ecranului, In schimb, daca variabila jumping este true, atunci personajul se afla in aer. Astfel, acesta va fi desenat pe primul rand al ecranului. Totodata, daca se afla in aer, va fi afisat si primul caracter al traseului, intrucat personajul poate ramane pe un obstacol daca a apucat sa sara peste el.

-liniile 295-308

Este definita functia care verifica daca personajul se izbeste de un obstacol. Daca personajul se afla in aer si sub el se afla un spatiu, atunci va cobori, prin setarea variabilei jumping pe false. Daca personajul nu se afla in aer si pe prima pozitie din stanga a cranului ajunge un obstacol, atunci jocul este pierdut si se seteaza variabila endGame pe true. Aceasta variabila va face posibila trecerea la afisarea mesajelor de final.

-liniile 312-373

Este definita functia care afiseaza intreaga animatie. Este afisat personajul, dupa care este afisat traseul. Scorul curent este afisat pe primul rand al ecranului, in partea dreapta. Se apeleaza funtia care verifica daca exista coliziune. In restul functiei se realizeaza generarea traseului. Am decis ca obstacolele si spatiile de pe traseu sa fie generate in blocuri pentru a facilita saritura personajului, controlata de utilizator. Daca aceste caractere nu ar fi generate in blocuri, ar pute aparea situatia in care langa un obstacol sa fie un spatiu, dupa care inca un obstacol. La viteze mari ale jocului, utilizatorului i-ar fi imposibil sa evite al doilea obstacol dupa ce l-a evitat pe primul. Numarul de caractere dintr-un bloc este dat de functia random si valoarea constanta pentru generarea bolcurilor definita pentru fiecare nivel. Pentru a evita situatia in care am putea avea un bloc lung de acelasi tip de caractere, s-a realizat o restrictie prin aplicarea a doua countere, definite pentru fiecare tip de caracter de pe traseu. Astfel, daca a ajuns la 0 counterul definit pentru un anumit caracter, programul se asigura ca urmatorul bloc generat va avea caractere de celalalt tip. La fiecare iteratie a functiei se incrementeaza scorul obitnut si contorul folosit pentru alternarea celor 2 caractere definite pentru personaj.

-liniile 379-398

Este definita functia ce initializeaza informatiile despre nivele. Aceasta functie face posibila rularea jocului si fara o comunicare seriala.

-liniile 400-414

Este definita functia de setup. Se deschide calea seriala. Sunt create caracterele definite pentru joc si salvate in memoria shield-ului. Sunt setati pinii pentru butoane, iar pentru butonul K3 se creeaza o intrerupere, la care se ataseaza functia jump. In aceasta functie este setata variabila jumping pe true, realizandu-se program, in acest mod, sarirea personajului. Este apelata functia de initializare a nivelelor.

-liniile 417-448

Este definita functie de loop. Cu ajutorul variabilelor de control se apeleaza functiie definite pentru fiecare etapa a aplicatiei. Astfel, daca variabila playing este setata pe false, inseamna ca utilizatorul se afla in meniul de start, apelandu-se functia corespunzatoare. Totodata, daca utilizatorul se afla in aceasta etapa si pe calea seriala se afla date de citit, atunci se apeleaza functia care sa proceseze acele date. Dupa realizarea citirii, se seteaza variabila de control pentru citire pe false, astfel incat sa nu se produca citirea de mai multe ori. Este suficienta doar o citire a datelor din fisier pentru ca acele informatii sunt stocate intr-un vector de structuri ce va fi folosit pe toata durata rularii programului. Astfel, o recitire a informatiilor din fisier ar fi redundanta. Daca utilizatorul este in etapa de joc, intai se apeleaza functia de initializarea a traseului si a celorlalte variabile de control. Dupa aceasta, se trece in etapa efectiva de joc. Se verifica starea variabilei endGame. Daca este setata pe false, atunci utilizatorul inca joaca si se apeleaza functia care afiseaza animatia jocului. Daca este setata pe true, inseamna ca jocul a fost pierdut si se va trece in urmatoarea etapa a aplicatiei, in care se actualizeaza informatiile despre nivele, dupa care se trece la mesajele de final afisate prin functia endPage().

1. **Bibliografie**

<https://www.arduino.cc/reference/en/language/variables/data-types/stringobject/>

<https://www.arduino.cc/en/Reference/LiquidCrystalCreateChar>

<https://arduino.stackexchange.com/questions/16750/reading-arduino-serial-stream-in-java-using-jserialcomm>

http://gmac.2600tech.com/category/jserialcomm-arduino-java-serial/