

PROIECT ÎN CADRUL DISCIPLINEI POO -Spânzurătoarea-

Autor:

Student Nistor Elena-Simona

Grupa

3122B

CUPRINS

CUPRINS	2
TEMA ȘI MOTIVAȚIA ALEGERII :	3
DESCRIEREA PROBLEMEI:	3
DESCHIDEREA PROIECTULUI ÎN CONSOLA LINUX.....	4
ELEMENTE SPECIFICE POO:.....	8
Cazuri excepționale , prezentarea opțiunii 2, 3 și 4.....	13

TEMA ȘI MOTIVAȚIA ALEGERII :

Tema aleasă pentru proiectul din cadrul disciplinei POO este jocul Spânzurătoarea. Acesta presupune o interfață destul de simplă, cu o grafică la fel de simplă, ce include o mini hartă cu cele 6 nivele ale jocului , după care apariția spânzurătorii după începerea nivelului, și continuarea spânzurătorii la alegerea unei litere greșite.

Am ales proiectarea acestui joc deoarece este un joc popular și destul de cunoscut, ceea ce va face ca acest proiect să fie relevant și accesibil pentru un public larg. De asemenea un alt motiv îl reprezintă dezvoltarea abilităților cognitive, adică acest joc implică gândirea critică , analiza și sintetizarea informațiilor. Un alt motiv îl reprezintă implementarea jocului, care are o interfață simplă și intuitivă, ceea ce îl face ușor de utilizat și de înțeles pentru orice utilizator de orice vârstă.

Prin urmare, acest proiect poate aduce o valoare reală și practică pentru toți utilizatorii care doresc să se distindă prin a juca un joculeț de dificultate medie și ușor de utilizat.

DESCRIEREA PROBLEMEI:

Algoritmul pentru jocul Spânzurătoarea poate fi divizat în două părți principale: inițializarea jocului și ciclul de joc. Iată o posibilă descriere a algoritmului:

1. Inițializarea jocului:

- Alege un cuvânt aleatoriu dintr-o listă prestabilită.
- Afișează utilizatorului numărul de litere din cuvânt sub forma de spații goale(underscore).
- Inițializează numărul de încercări rămase la o valoare prestabilită (de exemplu, 6).
- Inițializează lista cu litere introduse de utilizator la o listă goală.
- Odată ce o literă a fost aleasă , nu va mai putea fi introdusă din nou până la nivelul următor .

2. Ciclul de joc:

- Așteaptă până când utilizatorul introduce o literă.
- Verifică dacă litera introdusă se află în cuvântul ales.

Dacă litera introdusă este în cuvânt:

- Actualizează spațiile goale corespunzător cu litera introdusă.
- Verifică dacă utilizatorul a ghicit tot cuvântul. Dacă da, afișează un mesaj de felicitare și încheie jocul.

Dacă litera introdusă nu este în cuvânt:

- Scade numărul de încercări rămase.
- Afișează o parte a imaginii cu spânzurătoarea corespunzătoare numărului de încercări rămase.
- Verifică dacă utilizatorul a epuizat toate încercările. Dacă da, afișează cuvântul și un mesaj de înfrângere și încheie jocul.
- Aduă litera introdusă la lista cu litere introduse de utilizator.
- Afișează starea jocului utilizatorului (spațiile goale, literele introduse, imaginea cu ghilotina).
- Repetă ciclul de joc până când utilizatorul ghicește cuvântul sau epuizează toate încercările.

DESCHIDEREA PROIECTULUI ÎN CONSOLA LINUX

De precizat este faptul că am folosit virtual box pentru a putea lucra în Linux.

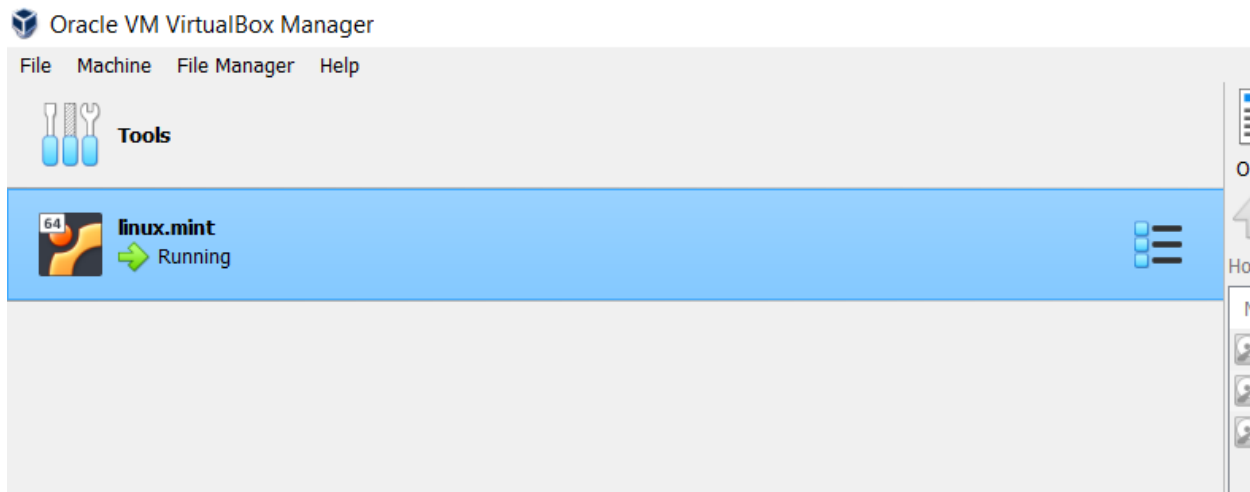


Figura 1. Virtual box și versiunea de linux instalată

Pentru a putea deschide proiectul în consola Linux se vor respecta pașii:

1. Click dreapta pe ecran

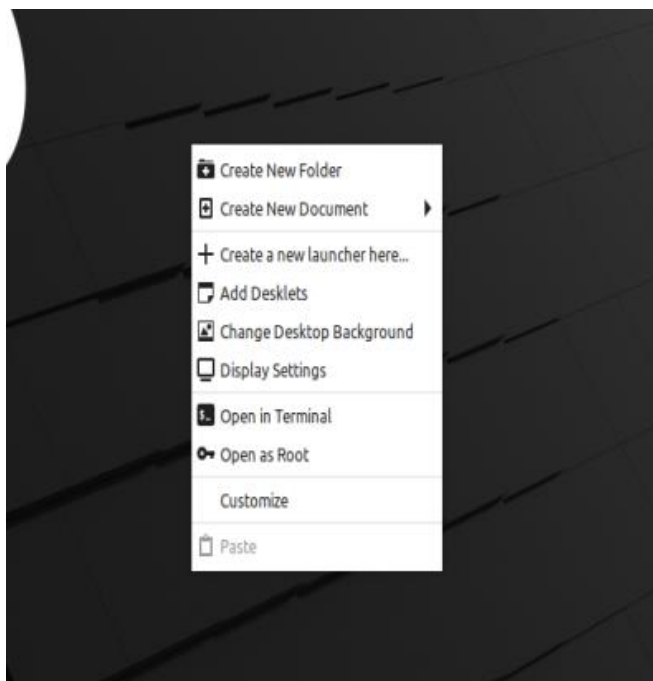


Figura 2. Pasul 1 pentru a deschide aplicația în consola Linux

2. Se alege opțiunea “Open in Terminal”:

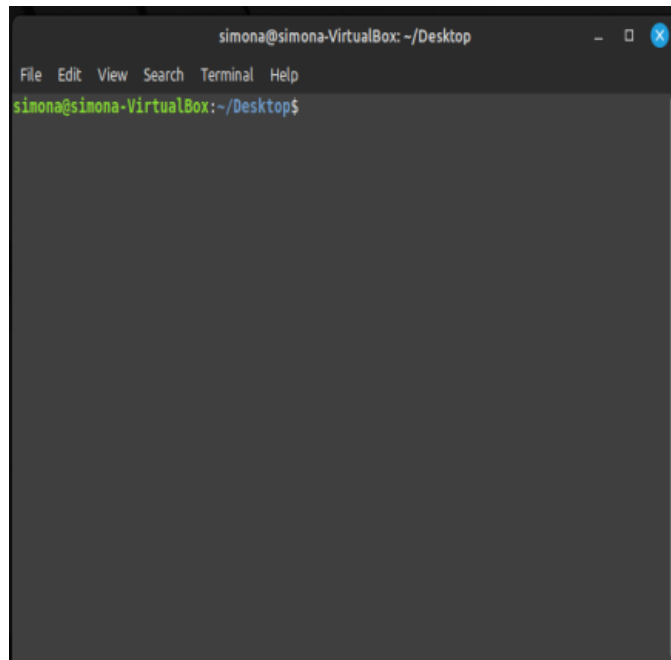


Figura 3. Deschiderea terminalului în linux

3. Se va deschide consola și vom scrie următoarele două linii în consolă:
- a) ls(cu care verificăm dacă avem folderul cu proiectul curent)

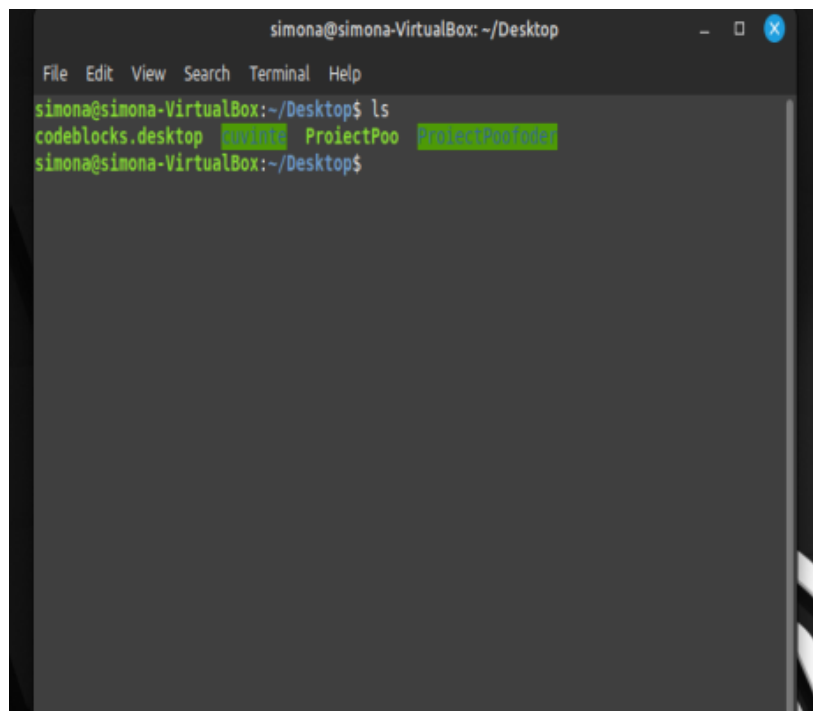


Figura 4. Prima comandă pentru a deschide proiectul.

Se observă după scrierea comenzii ls apare și folderul cu proiectul , deci îl putem deschide.

b) ./și denumirea folderului cu proiectul , în cazul acesta “./ProiectPoo” și apăsarea tastei ENTER

```
simona@simona-VirtualBox:~/Desktop$ ls
codeblocks.desktop  cuvinte  ProiectPoo  ProiectPooFolder
simona@simona-VirtualBox:~/Desktop$ ./ProiectPoo
```

Figura 5. A doua comandă pentru a deschide proiectul

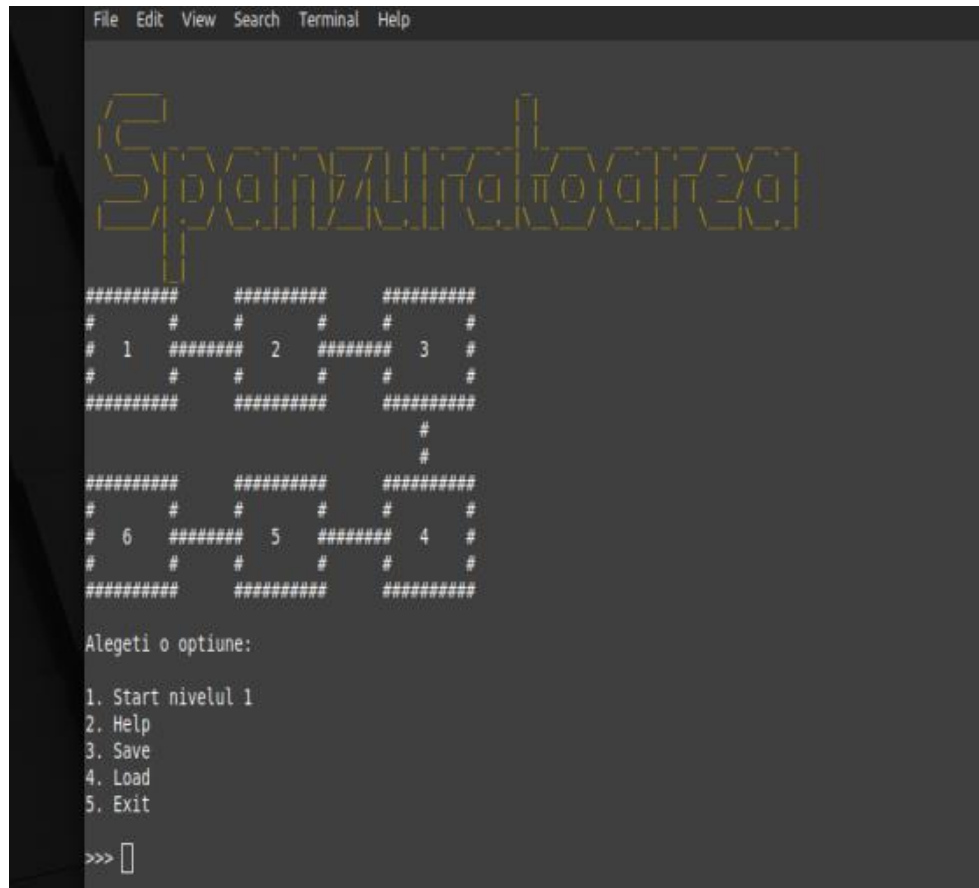


Figura 6. După apăsarea tastei ENTER

ELEMENTE SPECIFICE POO:

Pentru a crea jocul Spânzurătoarea în C++, vom folosi POO (Programare Orientată pe Obiecte) pentru a structura și organiza codul.

De menționat este faptul că acest proiect a fost creat și va fi rulat din sistemul de operare linux, mai exact din linia de comandă linux.

Meniul jocului va fi destul de intuitiv conform figurii 7.

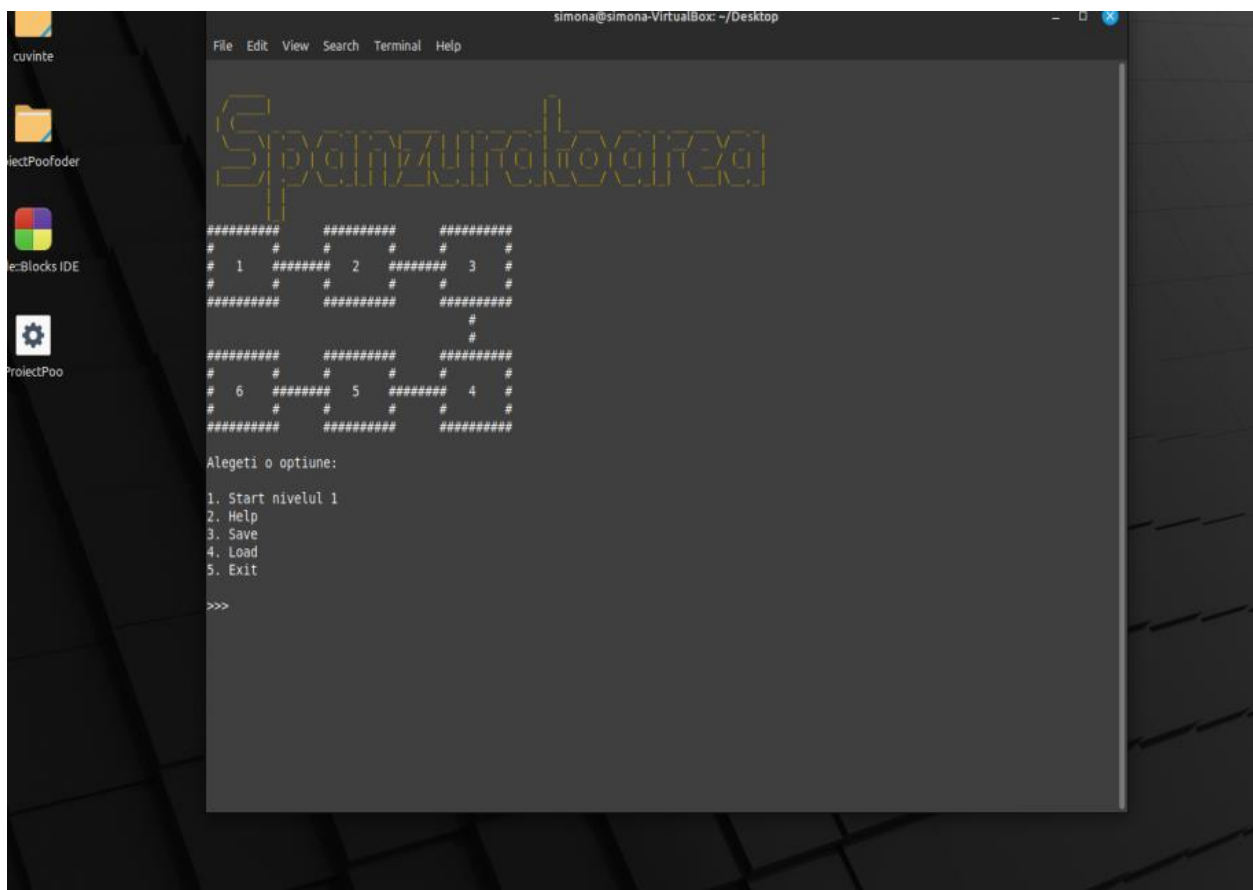


Figura 7. Meniul jocului

În continuare, vom descrie detaliat fiecare element al programului OOP:

CLASE:

- a) Clasa "JocHandler": Această clasă va fi folosită pentru menținerea meniului , jocului , setărilor și statusul programului.



```
main.cpp | include/MeniuNivele.h | src/MeniuNivele.cpp | src/Nivel.cpp | include/JocHandler.h | src/JocHandler.cpp | include/N
1  #ifndef JOCHANDLER_H
2  #define JOCHANDLER_H
3
4  #include "Nivel.h"
5  #include "MeniuNivele.h"
6
7  /// Clasa pentru mentinerea meniului, jocului, setarilor, statusului nivelului, etc
8  class JocHandler
9  {
10     MeniuNivele meniu; // Meniul
11     Nivel nivelHandler; // Clasa care ruleaza logica jocului
12
13     // Status joc
14     int nivelCurent = 1; // La intrarea in program, nivelul de start este 1
15
16     public:
17         JocHandler();
18         virtual ~JocHandler();
19
20         int Start(); // Punctul de intrare in program, returneaza int, 0 = good, other = erori
21         int PlayCurrentLevel(); // Logica de jucat nivelul
22         int SaveStateToFile(); // Saveaza starea jocului in fisier
23         int LoadStateFromFile(); // Incarca jocul din nivel
24     };
25
26 #endif // JOCHANDLER_H
27
```

Figura 8 . Header-ul pentru Clasa JocHandler

Aici avem 4 metode folosite după cum urmează

- Start () = punctul de intrare în program, returnează int, 0=ok, altceva=eroarea
- PlayCurrentLevel() = Logica de jucat nivelul
- SaveStateToFile() = Saveaza starea jocului in fisier
- LoadStateFromFile() = încarcă jocul de la nivelul pe care l-am salvat anterior(adică dacă am rămas la nivelul 4 și am salvat, chiar dacă închid programul, la redeschiderea consolei , voi alege din meniu această opțiune și se va redeschide jocul de la nivelul 4).

- b) Clasa "MeniuNivele": Această clasă va va afisa harta cu nivele și va citi opțiunea din meniu a utilizatorului.

```

1  #ifndef MENIUNIVELE_H
2  #define MENIUNIVELE_H
3
4  //clasa va afisa harta cu nivele si va citi optiunea din mneiu a utilizatorului
5  class MeniuNivele
6  {
7
8
9      public:
10         MeniuNivele();
11         virtual ~MeniuNivele();
12         void AfisareMeniu(int nivelCurent);
13         int CitireOptiuneTastatura();
14         void ClearScreen();
15
16 };
17
18 #endif // MENIUNIVELE_H
19

```

Figura 9. Header-ul pentru Clasa MeniuNivele

Aici avem 3 metode după cum urmează

- AfisareMeniu(int nivelCurent)= Funcția pentru afișarea meniului principal, cu nivelele
- CitireOptiuneTastatura() = Citim opțiunea din meniu a utilizatorului
- ClearScreen() = Ruleaza comanda de clear a terminalului

c) Clasa "Nivel": Această clasă va afișa spânzurătoarea cuvântul și decriptia acestuia și va citii literele introduse de utilizator.

```

1  #ifndef NIVEL_H
2  #define NIVEL_H
3  #include <string>
4  #include <vector>
5  #include <algorithm>
6
7  using namespace std;
8
9  //aceasta clasa va afisa spanzuratoarea cuvantul si decriptia acestuia si va citii literele introduse de utilizator
10 class Nivel
11 {
12     private:
13         vector<char> litereAleseCurrent;
14         string cuvantCurent;
15         string descriptieCuvantCurent;
16
17         int numarNivel;
18         int numarLitereAleseGresit;
19         int litereGhicie[100]; // un vector cu numere, 0 = litera la index nu e ghicita
20
21     public:
22         Nivel();
23         virtual ~Nivel();
24         void AfisareSpanzuratoare(); // Afiseaza spanzuratoarea
25         void AfisareCuvant(); // Afiseaza cuvantul si descriptia acestuia
26         void AfiseazInfoNivel(); // Afiseaza informatii despre nivelul curent
27
28         void AlegereCuvantRandom(int nrNivel); // Alege un cuvant la intamplare din fiserele cu cuvinte
29         bool IsWordCompleted(); // Returneaza true daca utilizatorul a ghicit tot cuvantul corect
30         bool PlayerLost(); // Decide daca jucatorul a pierdu nivelul
31         void ResetNrGreseli(); // Reseteaza numarul de litere alese gresit
32         void SetNrNivel(int); // Seteaza numarul nivelului
33         void HandleInput(); // Citeste o litera de la tastatura, si verifica daca este in cuvant
34         void HandleWinning(); // Face ce se intampla cand castigi un nivel
35         void HandleLosing(); // Face ce se intampla cand pierzi un nivel
36     };
37
38 #endif // NIVEL_H
39

```

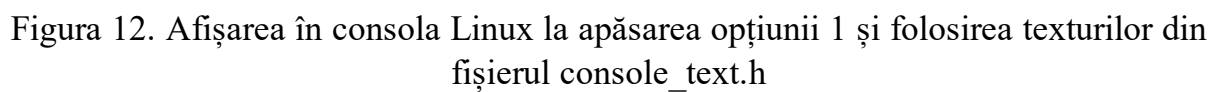
Figura 10. Header-ul pentru Clasa Nivel

Aici avem 11 metode după cum urmează

- AfisareSpanzuratoare() = afișează spânzuratoarea
- AfisareCuvant() = afișează cuvântul și descriția acestuia
- AfiseazInfoNivel() = afișează informații despre nivelul curent
- AlegereCuvantRandom(int nrNivel) = alege un cuvânt la întâmplare din fișierele cu cuvinte
- IsWordCompleted() = returnează true dacă utilizatorul a ghicit tot cuvântul corect
- PlayerLost() = decide dacă jucătorul a pierdut nivelul
- ResetNrGreseli() = resetează numărul de litere alese greșit
- SetNrNivel(int) = setează numărul nivelului
- HandleInput() = citește o literă de la tastatură, și verifică dacă este în cuvânt
- HandleWinning() = face ceea ce se întâmplă când câștigi un nivel (trece la următorul nivel)

- De asemenea avem un fișier text predestinat definirii emoji-urilor apărute în meniu. Acest fișier se numește `console_text.h`.

Figura 11. Fișierul pentru console_text.h.



În ceea ce privește fișierele în proiect , mai sunt prezente alte 6 fișiere folosite pentru listele de cuvinte de la fiecare nivel de unde se va alege un cuvânt random alături de descripția sa. De exemplu fișierul de cuvinte pentru nivelul 1 care conține numai cuvinte cu 3 litere prezentat în figura 13.

```
AIA pronume demonstrativ folosit pentru a indica un obiect sau o fiinta apropiata de vorbitor
AZI referitor la ziua curenta
BOB mica bila rotunda sau sferica
CUI instrument folosit pentru a fixa doua obiecte prin impingerea unei parti a acestuia in mate
DAR cuvant de legatura folosit pentru a indica o opozitie sau o limitare
FIE cuvant folosit pentru a introduce o alegere
GOL fara continut sau obiecte
HAI incurajare adresata cuiva pentru a se alatura
IAR cuvant de legatura folosit pentru a exprima o relatie repetata sau continuata
JOC activitate desfasurata in scopul distractiei sau al competitiei
LAC masa de apa mai mica decat un lac
MAI cuvant de grad care indica o crestere sau o modificare
NOU referitor la ceva recent sau neincercat inainte
PAS miscare facuta cu un picior in timpul mersului
RAI loc imaginar de fericire si binecuvantare
SEC unitate de timp egala cu o saizeci si a patra parte dintr-un minut
UNU primul numar natural
VAR persoana apropiata sau prietena
ZID structura de piatra sau caramida care separa doua spatii
BAI loc unde se fac bai sau se inota
CUM in ce fel sau mod
DAT referitor la un moment sau o perioada de timp trecuta.
```

Figura 13. Fișierul de cuvinte pentru nivelul 1

Cazuri excepționale , prezentarea opțiunii 2, 3 și 4

În cazul în care utilizatorul va pierde, în consolă se va afișa:

În cazul în care se va alege opțiunea 2 sau Help, se va afișa o descriție nu foarte lungă, dar care să ajute utilizatorul în folosirea programului creat :

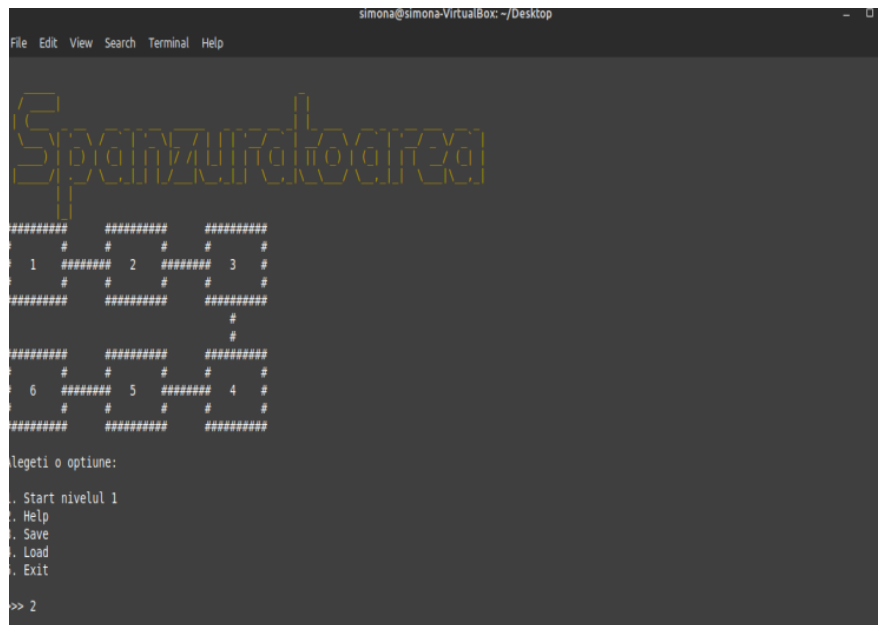


Figura 16. Alegerea opțiuni 2



Figura 12. Afișarea mesajului pentru opțiunea 2

În cazul în care se va alege opțiunea 3 , se va salva într-un fișier nivelul current al jocului

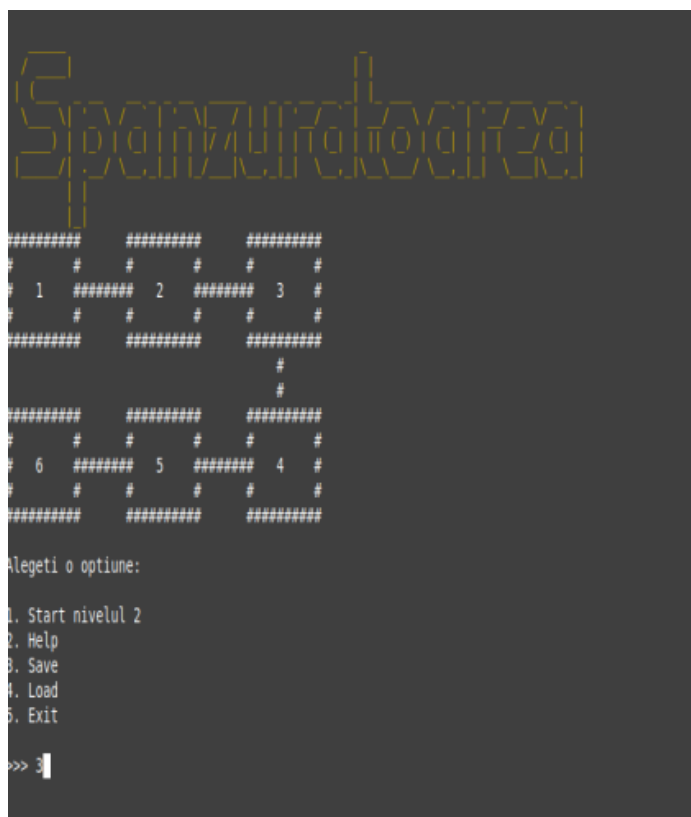
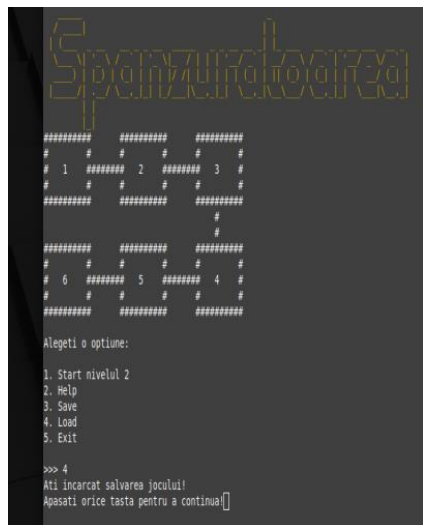
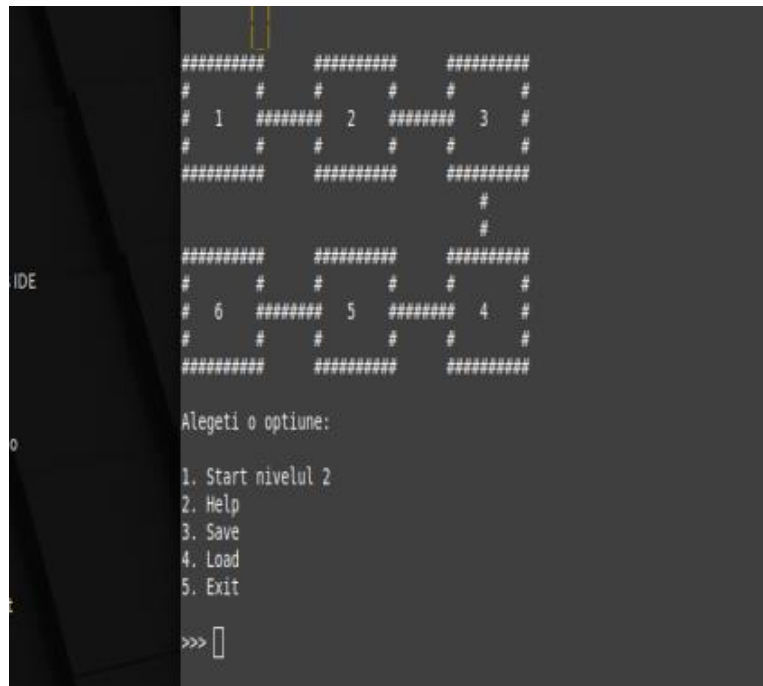


Figura 13. Rezultatul opțiunii 3

În cazul în care se va alege opțiunea 4, chiar dacă vom închide consola se va relua jocul de la nivelul salvat în fișier



a)



```
#####
# 1 ##### 2 ##### 3 #
# # # # #
#####
#####
#####
#
#
#####
# 6 ##### 5 ##### 4 #
# # # # #
#####
Alegeti o optiune:
1. Start nivelul 2
2. Help
3. Save
4. Load
5. Exit
>>> 
```

b)

Figura 14. a), b) Rezultatul opțiunii 4

Atașez link-ul către github : <https://github.com/NistorElena-Simona/poo/tree/main>