
Proiect realizat de Fariseu Adriana-Teodora

Grupa 30235

An 2021-2022

# 1. Introducere/Poveste

Cu toții am fost aici la un momentdat: Un student care trăiește pe pilot automat, inconștient de ceea ce se întâmplă în jurul nostru. Într-o zi, totuși, suntem treziți de o voce interioară care ne aduce aminte de cât de frumoasă ar putea să fie viața pe care o ratăm. Afecțiune, inteligență, creativitate, prietenie ... atât de multe experiențe pe care le ratăm fiind sclavii unei instituții.

Pornind aventura noastră însoțiți de vocea interioară misterioasă, suntem întâmpinați de o persoană care merge la aceeași facultate cu noi care ne trezește ceva inexplicabil în interior. Oare ar putea fi destinul? Oare în sfârșit ne-am găsit partenerul? Vocea din capul nostru pare să ne spună asta. Există un singur mod de a afla: ghidați de sentimente, încercăm să interacționăm cu colegul nostru, mergând în diferite locuri și făcând diferite activități. La fiecare pas suntem încurajați de vocea din interiorul nostru care. Dar oare... vocea aceasta este de încredere?

# 2. Descriere proiect

Autolove urmărește să fie un joc de genul Dating Simulator (prescurtat dating sim): un gen de joc care simulează situații din viața reală, mai exact în circumstanțe romantice. Personajul principal (jucătorul) este un student plictisit într-o vacanță care când iese afară la plimbare se întâlnește cu un coleg de facultate. Pentru diverse motive necunoscute, acel coleg trezește ceva în el, ceva care vrea să îl facă să îl cunoască mai bine. Pentru a realiza acest lucru, protagonistului îi sunt oferite mai multe locuri în care el poate să meargă la întâlniri cu colegul, aflând încet-încet mai multe lucruri despre el și cucerindu-l prin conversații și poate chiar cadouri. Mergând în diferite locuri, verificând obiecte din împrejurimea ta, vorbind despre diferite subiecte și realizând diferite acțiuni, creează-ți amintiri minunate cu colegul tău sau folosește vacanța ca să te descoperi pe tine însuți. Doar ai grijă la timpul și energia ta!

## Flow

### a. Început

Atunci când se va deschide executabilul cu jocul, utilizatorului i se va da opțiunea de a începe un joc nou sau de a relua un joc existent dintr-un save file (din maxim 4 save file-uri). Dacă se va începe un nou joc, vei fi introdus în povestea jocului și vei primi tutoriale utile pe parcurs. Dacă reîncepi de la un anumit punct, vei reîncepe de la ultimul dialog citit (Notă: Opțiunile de save și load vor fi separate de comenzile din lex și yacc. Ele vor fi realizate dintr-un program wrapper de Python care va chema executabilul de a.out)

### b. Gameplay

Ce ai la dispoziție în aventura ta?

Energie și timp: Timpul este cea mai prețioasă valută. Cheltuiește-l cu atenție pe activități și conversații cu colegul tău pentru a-ți crește stats-urile. În funcție de stats-uri- poate o sa ai ocazia să deblochezi un ending diferit ;)

La fel ca în viața reală, timpul nu se poate câștiga, însă energia se poate reface

Inventory/bag: în care poți păstra până la 6 obiecte. Aceste obiecte pot fi oferite colegului tău pentru a-i crește afecțiunea sau a afla mai multe despre ele.

3 zile rămase din vacanță în care poți să te relaxezi

Dating sim-urile tradiționale funcționează ca un fel de visual novel: Personajele dialoghează până când la un momentdat ți se va oferi ocazia să alegi o opțiune care poate duce povestea într-o anumită direcție. Aici însă, spre deosebire de dating sim-urile tradiționale, opțiunile nu se limitează doar la multiple choices. Ți se va da un input în care poți scrie mai multe acțiuni. Aceste acțiuni pot depinde de circumstanțele în care te afli. No worries, dacă nu știi ce activități ai putea să faci, poți întotdeauna să interoghezi folosind cuvântul cheie ”what”:

what options? – Uneori nu poți realiza orice acțiune. Acest lucru îți va spune ce acțiuni poți realiza la inputul respectiv.

what places? - Îți va da o listă cu locurile în care te poți duce, alături de timpul și energia care îți ia să mergi până la ele

what activities? Returnează o listă cu activitățile pe care le poți face într-un anumit loc

what topics? Oferă subiecte de discuții pe care le poți avea cu colegul tău. Diferite subiecte pot să scoată la iveală diferite informații despre el sau chiar despre tine

Poți să faci orice activitate posibilă din opțiunile oferite la un momentdat (detalii asupra acțiunilor vor fi explicate într-un capitol ulterior). Singura ta limită este timpul și energia. Atunci când nu mai ai timp sau energie, vei fi trimis acasă să te odihnești. De asemenea, încearcă să ai grijă și la energia partenerului tău. Nu ai vrea să-l epuizezi înainte de finalul jocului!

### c. Ending-uri

Vacanța, la fel ca restul lucrurilor bune din viață, trebuie din păcate să aibă un final. După 3 zile, în funcție de ce informații ai deblocat și de ce stats-uri ai avansat, vei avea parte de o surpriză diferită.

## Implementarea flow-ului

Deși la prima vedere trecerea printr-un dialog pare ușoară, trebuie să existe și un mod în care se citesc comenzi. Astfel, avem starea de PLAYING (în care se citește pur și simplu dialog, eventual cu unele lucruri în plus) și INPUT, în care se așteaptă o comandă. Astfel, pe lângă liniile de text se mai pot afla și alte lucruri precum variabile sau keywords.

Dialogurile de la scenariile posibile vor fi păstrate în fișiere (schimbate în funcție de schimbarea locului sau a firului narativ/zilei) Pattern-ul lor general este:

```
^Car_[0-9]+_(%drawSprite:sprite_name%)*< [A-Z|a-z]+>
```

Prima cifră reprezintă indexul personajului din vectorul de personaje posibile. Aceste cifre vor fi acoperite în partea despre variabile.

%drawSprite reprezintă emoția sau sprite-ul(numele fișierului în care se află desenul ASCII) pe care o are personajul când spune linia de dialog

Textul de după cele 2 puncte este dialogul propriu-zis și poate la rândul său să aibă diferite variabile text sau intregi strecurate, marcate cu %getVar_index_% sau %getVari_index_% (pentru ca C nu are reflection :( ), ale căror valoare va fi afișată în acel loc

Pe lângă linii de dialog, se mai pot afla diferiți marcatori care pot realiza lucruri despre metadate. Marcatorii cu ~~ schimbă starea jocului din PLAYING în INPUT și invers

- `~~input~~` așteaptă comenziu cu acțiuni
- `~%setVar_index_ %~` Schimbă variabila text respectivă cu valoarea introdusă de la tastatură. Dacă se pun și : și se scrie o valoare între index și %, se va considera valoarea aia

De asemenea, mai avem și maxStat, care schimbă fișierul de yyin cu dialoguri branched în funcție de care stat al protagonisului este maxim(afecțiune, agilitate sau inteligență). În cazul în care vă întrebați dacă yyin se poate schimba la runtime din yacc, răspunsul este da, atâta timp cât schimbarea se realiează în main

```
togglePrint: MAX_STAT{
    branch=1;
    sprintf(branchFile,"./Dialogues/day_%d_%d.txt",metaInfo-
>globalsi[3],maxStat());
    };
```

```
    yyin=startDay();
    while(yyin!=NULL){
        yyparse();
        if(branch==0)
            yyin=startDay();
        else
            {
                yyin=fopen(branchFile,"r");
                branch=0;
            }

    }
```

## Tipuri de date

*ENUM-uri:*

- `static enum options{ OPTIONS, PLACES, ACTIVITIES, TOPICS} option;`

*Structuri:*

- Cost: Cu câmp de time(în minute) și energy, ambele întregi
- Stats: Statistici despre personaj ca valori întregi: afecțiune, energie, intelect, agility
- Personaj: Cu câmp de nume și stats
- Acțiune: Cu nume, cost, stats(+ sau – număr cu care se modifică stat-ul respectiv)
- Place: Cu nume și o listă de acțiuni ce pot fi realizate când ești acolo
- Global_variables: Care conține toate câmpurile cu variabilele globale pentru a nu scrie multe declarații cu extern în fișierul .y. Primele 3 variabile sunt array-ul cu numele token-ilor hardcodați pentru a le găsi eventual mai ușor o valoare numerică. Restul vor fi explicați ulterior

```
typedef struct _shared{
    const char* placesNames[MAX_PLACES];
    const char* optionsNames[MAX_OPTIONS];
    const char* topicsNames[MAX_PLACES];
    cost movingCost[MAX_PLACES][MAX_PLACES];
    char globals[MAX_GLOBALS][255];
    int globalsi[MAX_GLOBALS_INT];
    place places[MAX_PLACES];
    character characters[MAX_CHARACTERS];

}global_variables;
```

## Variabile

1. Matrici de costuri de deplasare între locurile posibile
2. Variabile globale care pot fi
   a. Variabile text

      0: Numele jucătorului curent

   b. Variabile întregi

      0:Energia disponibilă

      1: Timpul disponibil

      2: ID-ul locului curent

      3: Ziua curentă

3. Locurile posibile în care poți să mergi
4. Vector de personaje posibile. Acest vector reține personajele participante la acțiune pentru a le putea lua mai ușor numele și stat-uri

      0: rezervat pentru ”???”, atunci când nu se știe cine vorbește... sau pur și simplu când se dorește să se creeze suspans ;)

      1: protagonistul

      2: Colegul

      3: Naratorul: Tutorialul/vocea noastră interioară care ne ghidează spre deciziile corecte în viață

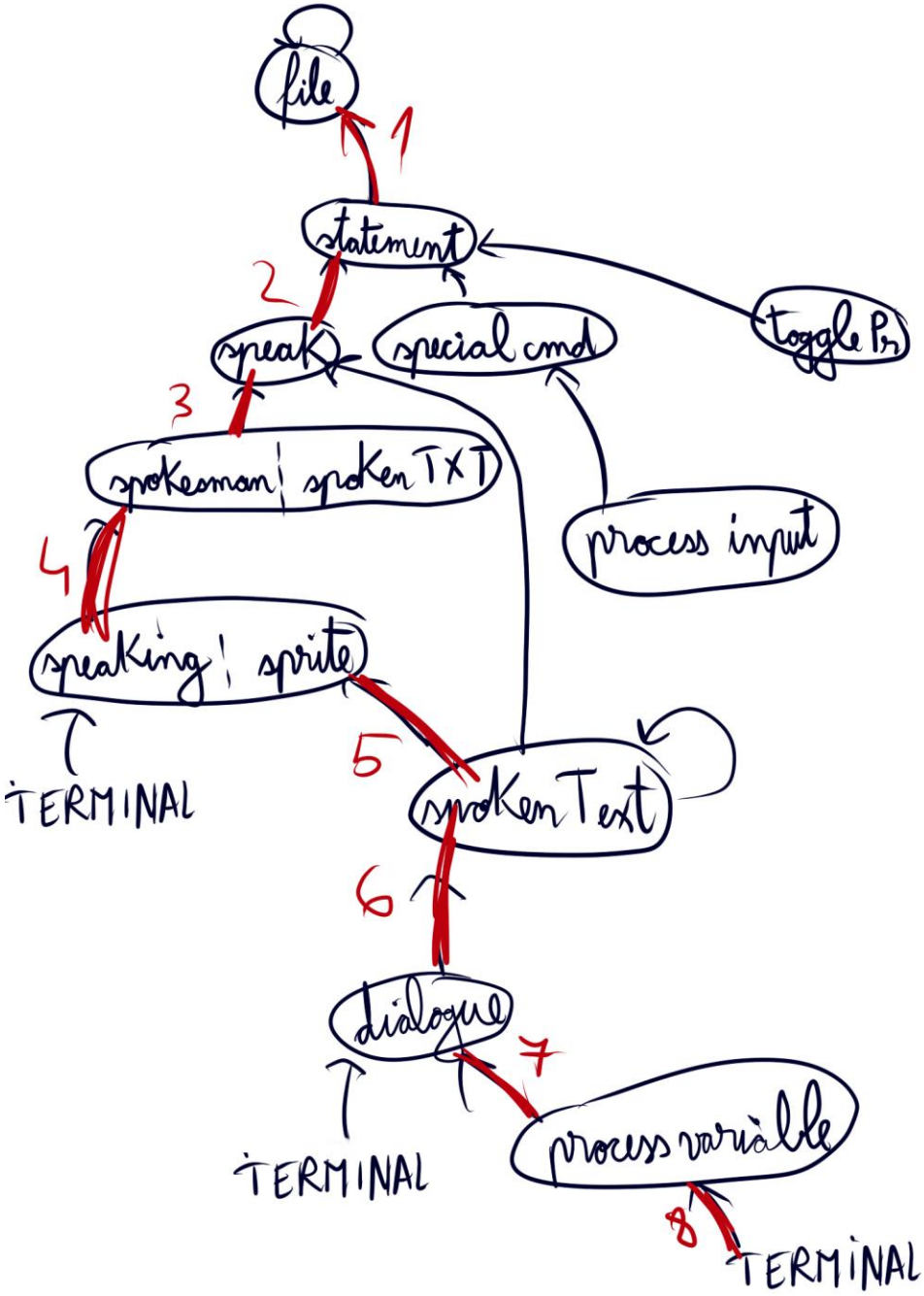
## Compilatoare

Deoarece avem două aspecte: citirea din fișier a dialogului și comenzile din gameplay-ul propriu-zis, putem să separăm aceste lucruri în două compilatoare de yacc separate. Atunci când vom avea nevoie de comenzi, vom apela compilatorul de comenzi cu system(./a.out) din compilatorul de dialog (singura problemă e că vor fi două procese separate și compilatorul de comenzi nu ar avea acces la valorile inițializate deja în cel de dialog și mmap am înțeles că nu merge bine pe WSL, așa că am folosit fișiere ca să le trimit între procese)

## Schiță arbori de parsare

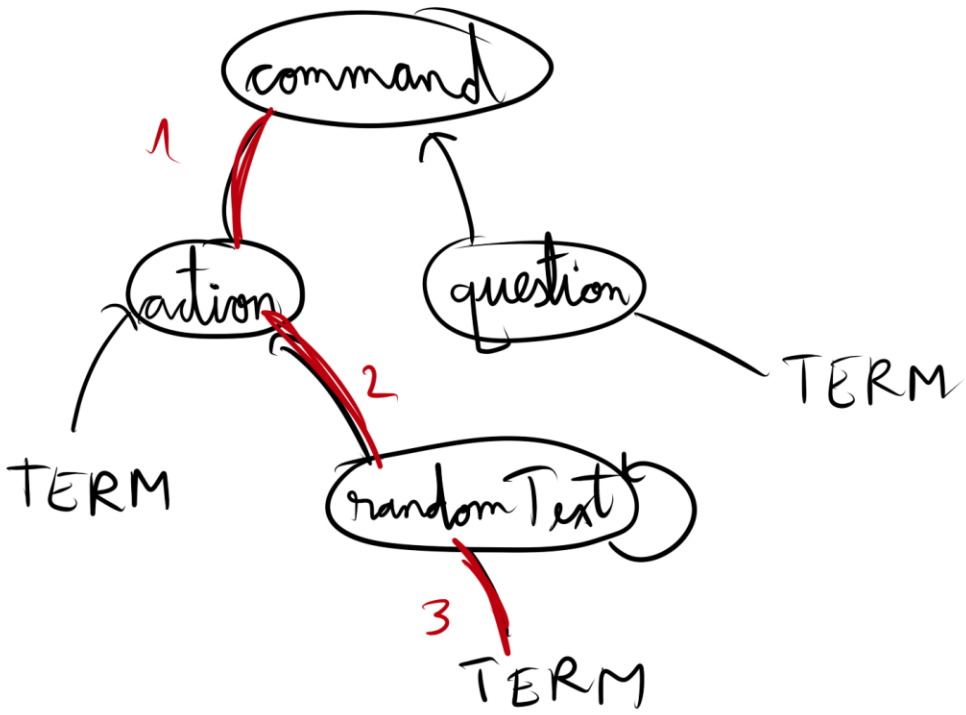
1. Compilator dialog. Dacă e să numărăm arcele din graf, acesta ar avea înălțimea 8 (9 dacă vrem nodurile)

```
file: statement
    | file statement
    ;
statement: speak | specialCommand |togglePrint
speak: spokesman '<' spokenText '>' { ···
};
specialCommand: MARK_INPUT processInput MARK_INPUT;
processInput: VAR_SET INDEX
{···
spokesman: speaking
        | speaking sprite;
sprite: '%' EMOTION ':' spokenText '%'
{···
}
togglePrint: MAX_STAT{···
speaking: SPEAKER INDEX{ strcpy(currDialogue.speaker,metaInfo->characters[$2].name);}
spokenText: dialogueText { strcpy(currDialogue.text,$1); }
          |  spokenText dialogueText{ strcat($1,$2);};
dialogueText: TEXT
          | processVariable;
processVariable:'%' VAR_VALUE INDEX '%' { strcpy($$,metaInfo->globals[$3]); };
```




2. Compilator comenzi compilatorul de comenzi este mult mai simplu deoarece nu are atât de multe comenzi diferite

```
command:
        | action
        | question {wasQuestion=0;};
action: RELOCATE PLACE
    {…
    | DISCUSS TOPIC
    {…
    | DISCUSS GAME
    {…
    | DO randomText
    {…
    };
question: ASK OPTION { int u=identifyWord($2,metaInfo->optionsNames,MAX_OPTIONS); printOptions(u); }…
randomText: TEXT{ strcpy($$,$1); }…
```




## Exemplu de flow

1. Utilizatorul deschide executabilul cu jocul

```
LFT/proiect final$ ./a.out
```

2. Utilizatorul este întâmpinat de un mesaj care marchează începutul jocului, după care va fi primit de un tutorial care îl va familiariza cu comenzile

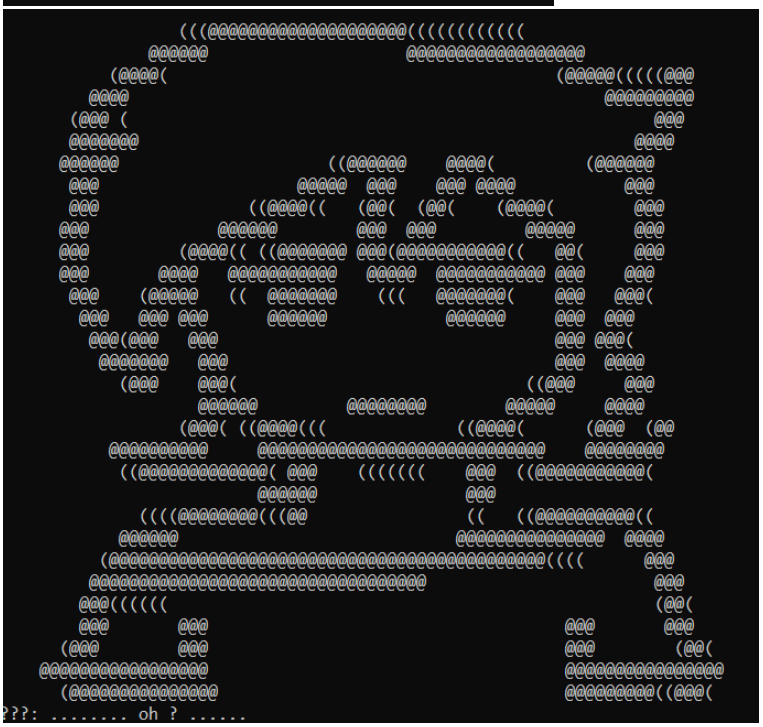
```
You wake up, ready for a new day!
================ Day 0 ================
```

```
Narrator: Here, try telling me your name (between 3 and 254 characters), then press enter to confirm your choice

Write your input:
```

```
Narrator: Now try it for yourself. Tell me what you'd like to do or ask me anything

Tip: if you forgot what you can do, type 'what options' to get a list of all actions
tutorial , day 0. energy:10, time:30 minutes
Write your input:
```

3. Jucătorul completează tutorialul și începe o prima zi din joc

```
Me: Well, that sure was a weird dream
```



```
???: ........ oh ? ......
```

4. Jucătorul parcurge dialogul până când ajunge la partea de introdus comenzi. Comenzile pot fi introduse până când energia  sau timpul protagonistului ajung la 0

```
Narrator: How shall you entertain yourselves?

Tip: if you forgot what you can do, type 'what options' to get a list of all actions
home , day 1. energy:100, time:480 minutes
Write your input:
```

5. Jucătorul este confuz și scrie ”What options”, primind la schimb o listă de acțiuni posibile

```
Narrator: How shall you entertain yourselves?

Tip: if you forgot what you can do, type 'what options' to get a list of all actions
home , day 1. energy:100, time:480 minutes
Write your input: what options

Here's everything you can do:
          a) Actions you can choose:
          1: go (to the )place_name - consume a certain amount of energy and time to go to the place you write. You also gain 5 agility per place
          2: let's activity - consume a certain amount of energy and time to do the specific activity available at the current place
          3: discuss topic_name: starts a conversation on the topic specified (always costs 30 minutes). Sometimes, discussing the right thing at the right place might unlock special things
          b) Questions that you can ask (always free):
          1: what places - shows the places where you can go and the costs to get there
          2: what activities - shows what you can do
          3: what topics - shows a list of things you can discuss

home , day 1. energy:100, time:480 minutes
Write your input:
```

6. Jucătorul vrea să meargă undeva, dar nu știe unde, așa că alege să scrie What places, care îi va afișa locurile posibile, alături de costul lor

```
home , day 1. energy:100, time:480 minutes
Write your input: what places

Here's the places you can visit, along with their costs:
park energy:20 time:15
cinema energy:10 time:5
library energy:15 time:15
gym energy:5 time:10
shop energy:5 time:5

home , day 1. energy:100, time:480 minutes
Write your input:
```

7. Utilizatorul alege să meargă la cinema, acțiune care îi va scădea timpul și energia, alături de locația curentă și fișierul de input din care se citește. Alternativ, dacă nu mai are energie pentru vreo acțiune, va fi dus automat acasă și ziua se va termina. În particular, unele zile au ending-uri diferite în funcție de stat-ul maxim de la acel moment

```
home , day 1. energy:30, time:400 minutes
Write your input: let's cook

You cook

You can't do anything anymore except for pressing a key to continue
```

8. Utilizatorul continuă jocul până când îl închide sau ajunge la un ending

## 3. Comenzi

Când ești într-o stare de input, există mai multe comenzi pe care le poți juca. Ele sunt procesate de un alt compilator de yacc separat, care va trimite înapoi la program informații necesare în funcție de output-ul comenzii

### a. Acțiuni de interogare care nu costă timp sau energie

What

options: ce poți face la momentul respectiv (dacă poți să mergi undeva, dacă poți să discuți etc.)

places : locurile unde te poți duce

activities: ce poți face în locul în care te afli

### b. Acțiuni care costă energie (sau care o pot chiar și reumple)

Go (to the) %place% : Schimbă locul în care te afli

Let's %activity% : Realizează acțiunea respectivă

Discuss %topic%: Discută cu NPC-ul despre un anumit subiect. Acest lucru poate să îți crească afecțiunea dacă alegi subiectul potrvit la locul potrivit

https://github.com/FTeodora/Autolove_Yacc_Game Pentru eventual povestea actualizată. Codul și logica din spatele parser-ului sunt gata, dar dialogul nu e prea grozav din punct de vedere artistic