Disciplina: Computação I
Professora Ministrante: Doutora Valéria Menezes Bastos

Grupo:
Guilherme Cappelli Bouzon de Amorim Cruz - DRE: 121170269
Júlio Ricardo Burlamaqui dos Santos - DRE: 121125214
Pedro Mateus Melo Ormesino Lins - DRE: 121159394

Jogo da Memória - “Mnemônico”

Universidade Federal
do Rio de Janeiro

Introdução

O grupo teve como tarefa criar um código em linguagem de programação C que simule um jogo de tabuleiro, especificamente o Jogo da Memória, estimulando assim a nossa dinâmica em equipe, através de reuniões virtuais entre os integrantes e atribuições de responsabilidades para com o código, promovendo nossas habilidades interpessoais.

O jogo não foi feito para ser fácil. Iniciando com 5 vidas, o jogador tem 5 chances para concluir o jogo, ou seja, acertar todos os 4 pares de cartas, e ao cometer erros, através da história desenvolvida para o jogo, perde vidas (1 vida por erro). Com figuras que facilitam o entendimento por parte do jogador de quais cartas ele está selecionando e com menu intuitivo, que permita o jogador escolher entre ler as regras, ler a história e jogar, tivemos a ideia de oferecer uma história de fundo para o jogo, acrescentando temática nas cartas e garantindo a imersão do jogador.

O jogo possui uma particularidade quanto à portabilidade no Sistema Operacional Windows, visto que o terminal do Windows não utiliza da codificação em UTF-8, sendo assim, os elementos estéticos podem não seguir o mesmo padrão apresentado em uma distribuição Linux, porém o funcionamento do código e do jogo em si permanece o mesmo.

Descrição

Desenvolvemos uma história para o jogo da memória, que consiste num pretexto para o jogo acontecer. Como criamos uma história, optamos pela liberdade criativa de nomear o jogo como Mnemônico, pois esta palavra significa “algo referente à memória”. Nela, um antediluviano chamado Éon Baxter sequestra os animais protetores da floresta para roubar-lhes de seus poderes. O nome do antagonista é uma alusão aos éons geológicos, como referência ao seu nascimento antigo, ainda, é menção ao herói bíblico Noé, que assim como Éon, angariou pares de bichos, porém, Éon o fez por motivos nefastos, e portanto, é o oposto de Noé, conferindo seu nome ao contrário. O jogador deve então assumir o papel do protagonista sem nome que irá libertá-los achando seus similares. Tivemos o cuidado de não definir o gênero deste protagonista para todos poderem se sentir envolvidos na história como se de fato fossem a personagem. O jogo também apresenta um desenvolvimento da história conforme as ações do jogador por meio de comentários dos personagens. “Mnemônico” tem dois encerramentos de acordo com o desempenho do protagonista, um final bom e outro ruim.

Regras

Quatro espécies de animais são escolhidas e ocultadas dentre oito cartas, estando cada espécie presente em exatamente duas cartas. Estas cartas são então apresentadas ao jogador viradas para baixo de forma que o mesmo não identifique onde cada animal se encontra. O jogador deve, então, selecionar um número de 1-8 que corresponda à carta que deseja tornar, revelando assim o animal contido nela por meio de um desenho aliado à legenda. Em seguida, deve escolher outro número de 1-8, sendo este número diferente do

primeiro, para revelar o segundo animal. Ao escolher a segunda carta, o jogador não pode apertar nada até que o jogo peça para fazê-lo. O objetivo é achar os pares de animais, isto é, os animais de mesma espécie. Caso os animais sejam da mesma espécie, estes animais são libertos e as cartas não poderão ser escolhidas, o jogador deve repetir o processo. Caso os animais sejam de espécies diferentes, eles retornam às suas respectivas cartas e uma vida é deduzida do montante do jogador, isto posto, poderá escolher mais duas cartas. O jogador tem um total de cinco vidas, dessa forma, podendo perder até quatro vidas sem o jogo encerrar. Ao perder mais uma vez, o jogo acaba e o jogador é derrotado. A vitória surge quando todas as espécies forem correspondidas aos seus respectivos pares, ou seja, sejam escolhidas simultaneamente em uma rodada apenas.

Principais Ferramentas Usadas

Abaixo listamos todas as ferramentas da linguagem C utilizadas no código do jogo:

- Estruturas condicionais: if, else, switch, case;
- Laços de repetição: while, for, break, continue;
- Estruturas manipulação I/O: printf, scanf, getchar, fgetc;
- Apontadores;
- Arrays estáticos;
- Macros;
- Protótipos de funções;
- Manipulação de arquivos: fopen, fclose;
- Manipulação de pseudo aleatórios: srand, time, rand;
- Temporizador: sleep;
- Saída do programa: exit;
- Função de localização: setlocale.

Bibliotecas Usadas

No código do jogo, as seguintes bibliotecas foram implementadas:

stdio.h - Standard Imput/Output, ou seja, biblioteca padrão de entrada e saída. Possui funções de exibição para a saída padrão e leitura da entrada padrão. Importa funções como printf e scanf.

stdlib.h - Standard Library, ou seja, biblioteca padrão. Possui funções de alocação de memória, geração de sequências e controle de processos. Importa funções como system e rand.

time.h - Como o nome sugere, trabalha com datas e horários. Possui macros referentes ao tempo que interagem com a biblioteca padrão. Com ela, importamos o macro NULL para coletar o tempo instantâneo a fim de gerar um número pseudo-aleatório.

locale.h - Como o nome sugere, auxilia problemas de localização, importando caracteres específicos que não fazem parte dos caracteres padrões da linguagem C. Ela se relaciona com a stdio, pois manipula as saídas. Importamos a função setlocale para usarmos acentuação e caracteres especiais da língua portuguesa.

windows.h - Como o nome sugere, contém declarações para funções da API do sistema operacional Windows. Ela é importada caso o sistema operacional que o programa esteja rodando seja Windows.

unistd.h - Análoga à windows.h, mas é importada caso o sistema operacional que o programa esteja rodando seja uma distribuição Linux.

Funções Usadas

Abaixo estão todas as funções utilizadas no trabalho:

- int main();
- void cabecalho();
- void limpa_tela();
- void regras();
- void historia_do_jogo();
- void ordena_cartas(char cartas[]);
- void banco_de_dados(char mensagens_acertou[], char mensagens_errou[], char animais[]);
- void imprime_animal(char animais[], char animal);
- void imprime_mensagem(char mensagens_acertou[], char mensagens_errou[], int mensagem);
- int tela_vitoria();
- int tela_derrota();
- int escolha();

A seguir explicaremos os objetivos de cada função listada acima, juntamente com o código referente à algumas delas.

Main

A main chama a função banco_de_dados, inicia um while “infinito” que limpa a tela e imprime o cabecalho com o menu com as opções de ler as regras ou história e encerra quando o jogador escolher um dos dois.

```
int main() {
    setlocale(LC_ALL, "");   //Para importar caracteres do UTF-8, como
acentuações.
   int erro = 0, selecao;

    banco_de_dados(mensagens_acertou, mensagens_errou, animais); //Gera
todos os bancos de dados.
```

```
   while(1) {  //While infinito!

       limpa_tela();
       cabecalho();

       if (erro == 1) {
           printf("[ERRO] Digite uma opção válida!\n");
           erro = 0;
       }

               printf("\t\t\t\t\t            Bem  vindo(a)  ao  Jogo  da
Memória!\n\n\n");
           printf("\n\t\t\t\t\t\t\t\t\t\t   Pressione  Ctrl+C  para  sair  do
jogo\n");

       printf("Pressione (1) para ler as regras\n"
              "Pressione (2) para prosseguir para a História\n");

       if (scanf("%d", &selecao) == 0) {
           while(getchar() != '\n');
           erro = 1;
           continue;
       }

       switch (selecao){
           case 1:
               regras();
               break;
           case 2:
               historia_do_jogo();
               break;
           default:
               erro = 1;
               break;
       }
   }
   return 0;
}
```

Cabeçalho
Função que imprime o título do jogo.

Limpa Tela
Dependendo do sistema operacional do jogador, a função executa um comando específico para limpar a tela do terminal.

```
void limpa_tela() {
   //Função que limpa a tela
   #if defined _WIN32
       system("cls");
   #endif
   #if defined linux
       system("clear");
   #endif
}
```

Regras
Função que imprime as regras do jogo e pergunta ao jogador se deseja prosseguir para a história ou execução do jogo.

História do Jogo
Função que imprime a história do jogo e pergunta ao jogador se deseja prosseguir com o jogo.

Ordena Cartas
Função usada para “aleatorizar” o conteúdo do vetor char casas[] que contém a numeração dos animais em seu respectivo arquivo de texto. Após isso, são escolhidas três primeiras posições deste vetor que são alocadas ao vetor char cartas[], duplicando-as no preenchimento. Após ter todas posições do vetor preenchidas, é feito novamente a “aleatorização” do conteúdo dentro do vetor, sendo esses valores os correspondentes aos animais que aparecerão na carta selecionada.

```
void ordena_cartas(char cartas[]) {
   //Função faz o shuffle de um array
   int i, j, temp;
   char casas[8] = {'1', '2', '3', '4', '5', '6', '7', '8'};

   srand(time(NULL));
    for (i=0; i < 8; i++){   // Inverte posicoes umas com as outras do
vetor casas[8]
       j = (rand() % 7) + 1;
       temp = casas[i];
       casas[i] = casas[j];
       casas[j] = temp;
   }
```

```
    for(i = 0; i < 4; i++){   // Seleciona os 4 primeiros indices a
partir do zero desse vetor casas[8]
       cartas[i] = casas[i];  // A partir desses 4 ele duplica essas 4
casas para obtermos 8 cartas, com 4 pares de animais
       cartas[i+4] = casas[i];
   }

    for (i=0; i < 8; i++){   // Agora, como já temos o baralho com 8
cartas e 4 animais
        j = (rand() % 7) + 1;  // Podemos embaralhar esse deque novo,
para o jogador advinhar as posiçoes dos animais
       temp = cartas[i];
       cartas[i] = cartas[j];
       cartas[j] = temp;
   }
}
```

Banco de Dados

Essa função realiza o armazenamento das figuras e mensagens contidas nos blocos de texto nos vetores char mensagens_acertou[], char mensagens_errou[] e char animais[], através de um apontador FILE e iterando sobre cada caractere dos arquivos .txt (um por vez).

```
void banco_de_dados(char mensagens_acertou[], char mensagens_errou[],
char animais[]) {
    //Função que importa as mensagens que serão apresentadas que estão
armazenadas em arquivos .txt externos
   int i;
      char nomearquivo_1[30] = "animais.txt", nomearquivo_2[30] =
"mensagens_acertou.txt", nomearquivo_3[30] = "mensagens_errou.txt", c;
   FILE *ponteiro;

    ponteiro = fopen(nomearquivo_1, "r"); //Abre para leitura o arquivo
"animais.txt"
   if (ponteiro == NULL){ //Caso não consiga abrir o arquivo
       printf("Erro ao tentar abrir o arquivo de animais!\n\n");
       exit(1);
   }
   c = fgetc(ponteiro);
    for (i = 0; c != EOF; i++) { //Laço for até o fim da leitura do
arquivo
```

```
            animais[i] = c; //Cada posição do animais[] vai receber
caractere do arquivo
        c = fgetc(ponteiro); //Pega o próximo caractere
    }
    fclose(ponteiro); //Para ler o próximo arquivo

     // A partir daqui aplica-se o mesmo raciocínio do anterior para
diferentes arquivos
    ponteiro = fopen(nomearquivo_2, "r");
    if (ponteiro == NULL){
                printf("Erro ao tentar abrir o arquivo de mensagens
acertou!\n\n");
        exit(1);
    }
    c = fgetc(ponteiro);
    for (i = 0; c != EOF; i++) {
        mensagens_acertou[i] = c;
        c = fgetc(ponteiro);
    }
    fclose(ponteiro);

    ponteiro = fopen(nomearquivo_3, "r");
    if (ponteiro == NULL){
                printf("Erro ao tentar abrir o arquivo de mensagens
errou!\n\n");
        exit(1);
    }
    c = fgetc(ponteiro);
    for (i = 0; c != EOF; i++) {
        mensagens_errou[i] = c;
        c = fgetc(ponteiro);
    }
    fclose(ponteiro);
}
```

Imprime Animal

A partir da consulta do vetor char animais[], imprime as figuras dos animais, de acordo com a carta que o jogador selecionou.

```
void imprime_animal(char animais[], char animal) {
    //Função que imprime os animais
    int i = 0;
    for (i = 0; animais[i] != animal; i++);
```

```
    i++;
    for (; animais[i] != '#'; i++)
        printf("%c", animais[i]);
     /*Para separar as mensagens, colocamos um caractere que atua de
maneira similar ao \0, no
      qual mandamos o compilador ler até chegar na cerquilha. Essa ideia
se repetirá no programa*/
}
```

Imprime Mensagem

Essa função imprime mensagens aleatórias pelos vetores: char mensagens_errou e char mensagens_acertou[]. Caso o jogador tenha acertado o par de cartas, a função escolhe uma das mensagens no vetor mensagens_acertou[] "aleatoriamente", e, dependendo da mensagem, informa que o vilão Éon Baxter ou um animal indefeso que a disse. Se o jogador errou, ela seleciona "aleatoriamente" uma das mensagens do vetor mensagens_errou[] e imprime a fala do vilão.

```
void imprime_mensagem(char mensagens_acertou[], char mensagens_errou[],
int mensagem) {
     //Função que imprime mensagens de maneira aleatória conforme a
necessidade do programa
    int i = 0, j = 0;
     char frases[10] = {'0', '1', '2', '3', '4', '5', '6', '7', '8',
'9'};

    srand(time(NULL));
    j = (rand() % 10);  //Geração de números aleatórios de 0 a 9

    if (mensagem == 0) {  //Mensagens quando o jogador erra
        printf("Éon Baxter: ");
        for (i = 0; mensagens_errou[i] != frases[j]; i++);
        i++;
        for (; mensagens_errou[i] != '#'; i++) //Mesma ideia do \0
            printf("%c", mensagens_errou[i]);

    } else {  //Mensagens de acerto!
          if(j < 3)   //Mensagens com índice de 0 a 2 são ditas pelos
animais resgatados
            printf("Animal Indefeso: ");
        else
             printf("Éon Baxter: ");  //Mensagens com índice de 3 a 9 são
ditas pelo vilão
```

```
        for (i = 0; mensagens_acertou[i] != frases[j]; i++);
        i++;
        for (; mensagens_acertou[i] != '#'; i++) //Mesma ideia do \0
            printf("%c", mensagens_acertou[i]);
    }
}
```

Tela Vitória

Essa função é chamada quando o jogador acerta todos os quatro pares de cartas, limpando a tela e imprimindo na tela a mensagem de “Você Venceu!”, depois de 2 segundos limpa a tela novamente, imprime o final da história e pergunta ao jogador se deseja jogar novamente ou não, pressionando (1) ou (2), respectivamente.

Tela Derrota

Essa função é chamada quando o jogador perde todas as suas vidas, chegando ao valor 0. A função limpa a tela e imprime a mensagem de “Você Morreu”, depois de 2 segundos limpa a tela novamente, imprime o final da história e pergunta ao jogador se deseja jogar novamente ou não, pressionando (1) ou (2), respectivamente.

Escolha

Pode ser considerada a principal função do jogo, onde são realizadas as checagens de tentativas errôneas do jogador, escolhas das cartas, contabilização da vida e dos acertos. Durante as escolhas das cartas, o jogador tem as opções de 1-8 para selecionar a primeira e logo em seguida a segunda carta. Caso haja a combinação dos animais correspondentes, suas respectivas casas no vetor layout[] são substituídas por um caractere de espaço, sendo a condição necessária para a vitória no jogo a substituição de todas as casas do vetor. Quanto aos erros do jogador, dependendo de qual for o erro, uma notificação personalizada é exibida na tela informando uma frase específica e sua penalidade, os tipos de erros estão detalhados no código. De forma a perder o jogo, o jogador precisa cometer 5 erros, perdendo assim consequentemente 5 vidas.

```
int escolha() {
    // Função principal que rege o funcionamento do jogo em si, ela
própria é composta de várias outras funções
    char cartas[8], layout[8] = {'1', '2', '3', '4', '5', '6', '7',
'8'};
   char prim, seg;
   int pos1, pos2, mensagem, acertos, vidas = 5, erro = 0, i;

   ordena_cartas(cartas);

   while (1){
       acertos = 0, mensagem = 0;
```

```
       limpa_tela();
       cabecalho();

         //Switch que monitora os erros. Significado dos erros na linha
482
       switch(erro) {
           case 1:
                printf("[ERRO] Você não pode escolher a mesma carta, ser
ludibrioso!\n"
                                    "Perderá 1 vida pela tentativa de me
enganar!\n\n");
               vidas--;  //Reduz uma vida
               erro = 0;  //Retorna a variável ao estado inicial
               break;
           case 2:
                 printf("[ERRO] Você não pode escolher uma carta que já
libertou.\n"
                             "Perderá 1 vida pela possível tentativa de me
enganar...\n\n");
               vidas--;
               erro = 0;
               break;
           case 3:
                 printf("[ERRO] Você não pode escolher uma carta que não
está na mesa, membro pernicioso!\n"
                           "Perderá 1 vida pelo insulto.\n\n");
               vidas--;
               erro = 0;
               break;
           default:
               break;
       }

       if (vidas < 1) tela_derrota(); //Verificação se o jogador morreu

         printf("\t\t\t\t    .------.          .------.          .------.
.------.\n"
                "\t\t\t\t    |%c.--. |         |%c.--. |         |%c.--. |
|%c.--. |\n"
                "\t\t\t\t    | :/\\: |         | (\\/) |         | :(): |
| :/\\: |\n"
                 "\t\t\t\t    | (__) |         | :\\/: |         | ()() |
| :\\/: |\n"
```

```
                "\t\t\t\t\t   | '--'%c|        | '--'%c|        | '--'%c|
| '--'%c|\n"
                 "\t\t\t\t\t   .------.        .------.        .------.
.------.\n\n"
                 "\t\t\t\t\t   .------.        .------.        .------.
.------.\n"
                "\t\t\t\t\t   |%c.--. |        |%c.--. |        |%c.--. |
|%c.--. |\n"
                "\t\t\t\t\t   | :/\\: |        | (\\/) |        | :(): |
| :/\\: |\n"
                 "\t\t\t\t\t   | (__) |        | :\\/: |        | ()() |
| :\\/: |\n"
                "\t\t\t\t\t   | '--'%c|        | '--'%c|        | '--'%c|
| '--'%c|\n"
                 "\t\t\t\t\t   .------.        .------.        .------.
.------.\n\n\n", layout[0], layout[1], layout[2], layout[3], layout[0],
layout[1],  layout[2],  layout[3],  layout[4],  layout[5],  layout[6],
layout[7], layout[4], layout[5], layout[6], layout[7]);
      printf("\t\t\t\t\t\t\t\t\t\t\t   Vidas restantes: %d\n\n"
                  "\t\t\t\t\t\t\t\t\t\t\t   Pressione Ctrl+C para sair do
jogo\n", vidas);

      printf("Escolha a primeira carta: ");

      /*Significado dos erros
        Erro 1: Valor designado para carta não pertecem ao intervalo
1-8. Vida é deduzida
        Erro 2: A carta selecionada já foi liberta, você não pode a
escolher novamente. Não deduz vida!
       Erro 3: Ambas primeira e segunda cartas são a mesma carta. Deduz
ponto!*/

      //Entrada da primeira carta e blindagem da entrada
      if (scanf("%d", &pos1) == 0) {
          while(getchar() != '\n');
          erro = 3;
          continue;
      }
      if (!(pos1 <= 8 && pos1 > 0)) {
          erro = 3;
          continue;
      }
      while(getchar() != '\n');
```

```
        pos1--;
        prim = cartas[pos1];
          imprime_animal(animais, prim); //Impressão do animal de acordo
com a entrada

        if (layout[pos1] == ' ') { //Verificação de carta já acertada
            erro = 2;
            continue;
        }

        //Entrada da segunda carta e blindagem da entrada
        printf("Escolha a segunda carta: ");
        if (scanf("%d", &pos2) == 0) {
            while(getchar() != '\n');
            erro = 3;
            continue;
        }
        if (!(pos2 <= 8 && pos2 > 0)) {
            erro = 3;
            continue;
        }
        while(getchar() != '\n');
        pos2--;
        seg = cartas[pos2];
          imprime_animal(animais, seg); //Impressão do animal de acordo
com a entrada

        if (pos1 == pos2) { //Verificação da escolha de mesma carta
            erro = 1;
            continue;
        }

        if (layout[pos2] == ' ') { //Verificação de carta já acertada
            erro = 2;
            continue;
        }

        if (prim == seg) {  //Caso as cartas formem um par:
                layout[pos1] = ' ';   //Tira-se a numeração das cartas,
indicando que ela já foram liberadas
            layout[pos2] = ' ';
            mensagem = 1;  //Imprime mensagem = 1: de acerto!
```

```
                    imprime_mensagem(mensagens_acertou,  mensagens_errou,
mensagem);
        } else {  //Caso as cartas não formem um par:
                    imprime_mensagem(mensagens_acertou,  mensagens_errou,
mensagem);
            vidas--;  //Reduz uma vida
        }
        sleep(2);

         for (i = 0; i < 8; i++){  //A cada carta removida, adicione uma
pontuação em acertos
            if (layout[i] != ' ') continue;
            else acertos++;
        }

          if (acertos == 8) tela_vitoria();   //Chegado em oito acertos,
declara-se vitória
    }
}
```

Saídas do Jogo

Abaixo seguem prints do programa em diferentes situações, juntamente com suas descrições.

Menu principal do jogo.

Tela de regras do jogo.

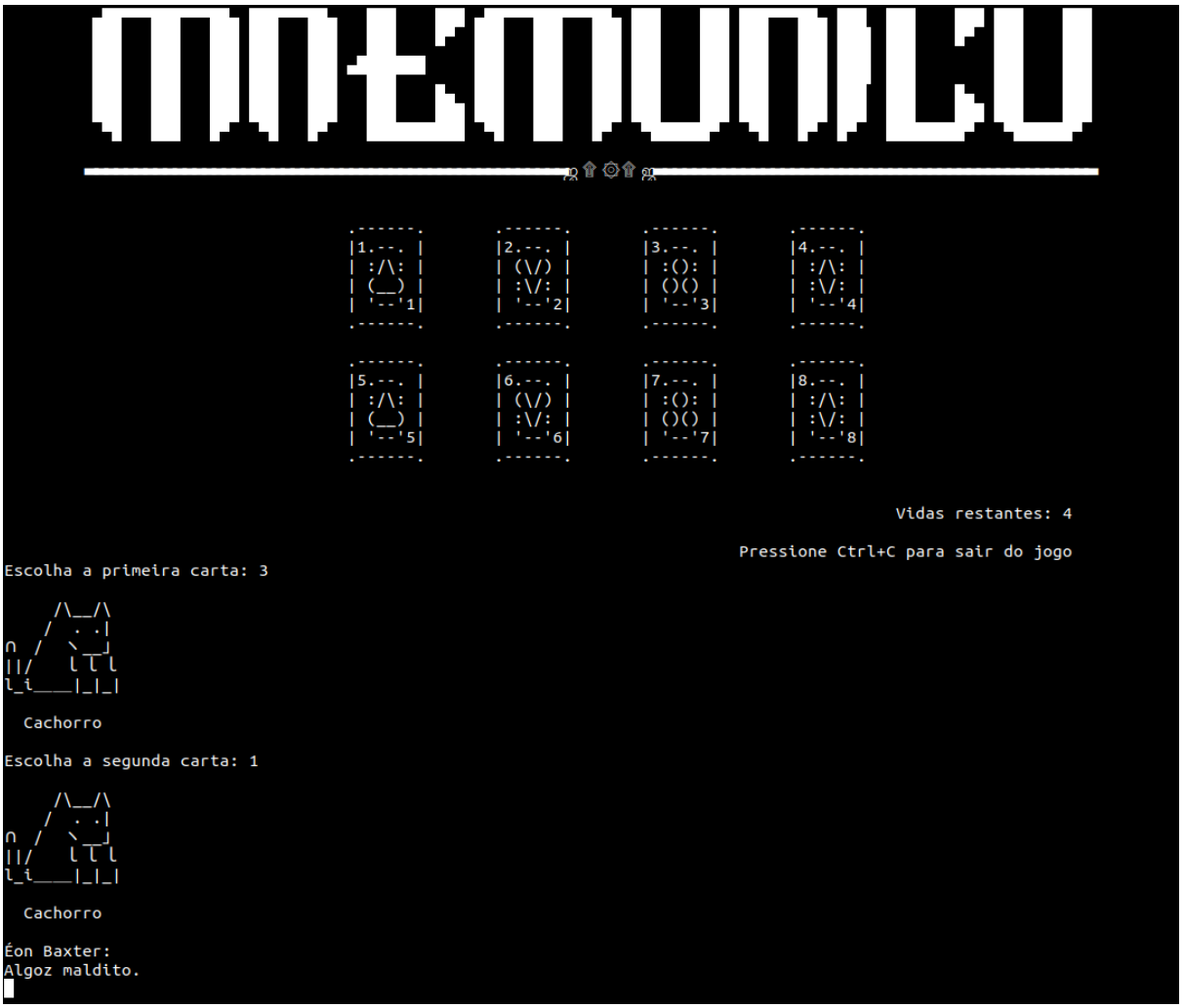
Tela de história do jogo.

Tela de seleção de cartas.

Escolha da primeira carta e impressão do animal nela contido.

Escolha da segunda carta. Nesse caso o jogador errou pois os animais não combinaram.

Tela seguinte ao erro com a vida reduzida.



Acerto na escolha das cartas por parte do jogador, pois os animais combinaram.

Devido ao acerto do jogador, a numeração das cartas liberadas some.

Tela seguinte à morte do jogador.

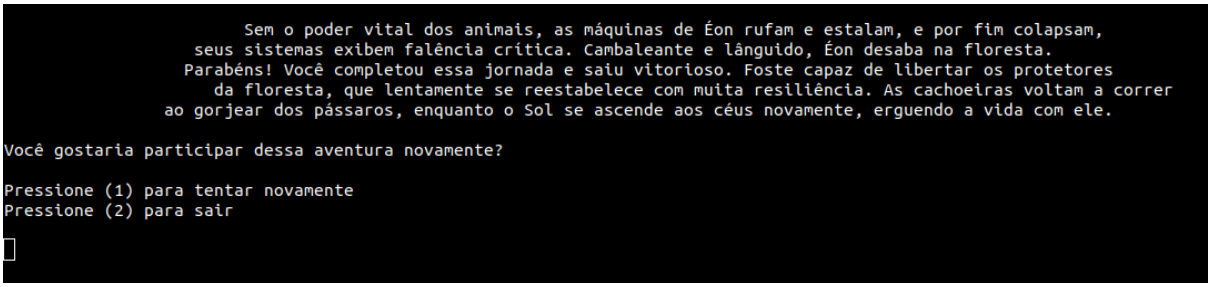
```
                    Sentindo cheiro de sangue e estranha sequidão na garganta, você contempla
                 os céus pela última vez enquanto mira o Sol, apesar de sentir-se com tanto frio.
              Tu jazes na floresta, e ela é, pois, teu receptáculo. Acostuma-te à lama que te espera.
                 Não fostes capaz de libertar os protetores da floresta, que rui a passos largos.
            Poderoso, Éon Baxter lança sua investida contra o restante de natureza no mundo, submetendo-a
             à obscuridade. A tecnocracia rege sobre o globo.

Você gostaria de tentar mais uma vez e mudar o destino do mundo?

Pressione (1) para tentar novamente
Pressione (2) para desistir
```

Tela descrevendo a morte do jogador de acordo com a história e pergunta de continuação ao jogador.

Tela seguinte à vitória do jogador.



Tela descrevendo a vitória do jogador de acordo com a história e pergunta de continuação ao jogador.

Conclusão

No decorrer do projeto, colocamos em prática alguns dos conhecimentos adquiridos na disciplina, além de conteúdos obtidos via vídeos no YouTube e a documentação da linguagem C (https://www.tutorialspoint.com/c_standard_library/float_h.htm). Percebemos que, por mais que o jogo fosse relativamente simples de fazer, gostamos do desafio proposto e nos sentimos entusiasmados em relação à implementação de novas funcionalidades sempre que possível, como por exemplo criação de uma história própria para o jogo, com elementos extras como falas e figuras ASCII. Prezamos bastante ao decorrer da criação do jogo, com seu elemento estético, sendo assim optamos por desenvolver o jogo na codificação UTF-8, que nos possibilitou usos de caracteres especiais que garantem a imersão do jogador.

Apesar da entrega do trabalho, desejamos continuar com esse projeto posteriormente, implementando algumas funcionalidades novas que não tivemos tempo para estudar sobre e adicionar, são elas: música do jogo e de efeitos sonoros, jogabilidade via setas do teclado, distinção de cor em certos elementos, adição de cronômetro para escolha das cartas, ranking com as maiores pontuações e diferentes níveis de dificuldade.