RPG

Universidade Federal Fluminense - PURO
Monitor: Victor Santos
Professores: Alessandro Copetti e Patrick Moratori
Programação de computadores II

O que é?

- Role Playing Game (Jogo de Interpretação de papéis).
- Jogadores são personagens da história.
- O personagem se desenvolve no decorrer da história (Ex: entra na faculdade, sobe de nível, ganha reputação).

O que sempre tem?

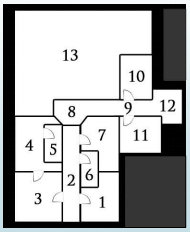
- Um enredo!
- Presença de um narrador ou mestre. Neste caso, será o programa.
- O personagem deve ter poder de decisão a cada "etapa".
- Fator sorte presente (Ex: batalhas, sorteio de perguntas, sorteio de estória).

Cuidados:

- O enredo pode evitar muitos problemas na hora de programar!
- Pensem no RPG a ser desenvolvido mais parecido com um livro-jogo do que com um RPG online famoso. Motivo: o mapa e as interações!
- Em um RPG online, o usuário interage com outros usuários. No trabalho, não!
- Em um RPG online, o usuário tem um mapa muito maior. Aqui, o ideal é que o mapa seja bem "fechado".

Exemplo de mapa

O usuário deve poder escolher o caminho que ele quer seguir.



Várias salas, cada sala deve ter sua interação com o usuário prevista!

Requisitos: Usar a grande maioria dos conteúdos de Prog II !!

1/2

1. Incluir um componente aleatório
   Por exemplo, a aleatoriedade pode ser o texto/resposta que aparece para o jogador ou decidir sobre o número de personagens que o personagem principal encontrará em determinado momento
2. Usar um vetor de estruturas
   Por exemplo, um vetor contendo informações de diversos personagens
3. Criar funções parametrizadas com poucas linhas de código e com tarefas específicas

Requisitos: Usar a grande maioria dos conteúdos de Prog II !!

2/2

4. Usar no mínimo 5 funções das bibliotecas do C ainda não utilizadas nas listas de exercícios
   - Por exemplo, sprintf() e outras de strings
5. Fazer uso intensivo de arquivos
   - Por exemplo, quando mostrar os diálogos com o usuário ou gravar etapas ou resultados alcançados por personagens

Critérios de avaliação do programa de RPG

1. criatividade da estória e aderência aos princípios do RPG
2. Implementação dos requisitos e a complexidade do programa

Opcional:

- Interface gráfica. (Allegro ou conio)
- Lista encadeada.
- Busca e ordenação.
- Matrizes.

Estrutura básica

Função Main()

- Função main() pequena se comparada ao código todo.
- Uso extensivo de funções.
- Na main, uma estrutura de repetição costuma conter todas as chamadas de outras funções.
- Outra estratégia é chamar uma função na main() e nunca mais voltar nela.
- Essa outra função puxa outras funções e por ai vai até o final do programa.

Exemplo com laço de repetição:

```
//Declaração das variáveis
char *nome;
int classe;
   ...
//Chamada de funções de inicialização
fflush(stdin);
gets(nome);
classe = escolher_classe();//Pede ao usuario para escolher uma classe e retorna um número id.
inicializa_personagem(nome,classe);
   ...
//Chamada da função que inicia a história
hist_introdução();
   ...
//Laço de repetição que segura a história
while(jogo_acabou() == 0)   //Jogo acabou retorna falso
{
   ...
   switch(escolhas_do_jogador)
   {
   case 1:
      ... break;
   case 2:
      ... break;
   case 3:
      ... break;
      ...
   default:
   }
```

Função main() não tão pequena assim!

Porém, mais legível

Exemplo sem laço:

```
int main(void){
  char nome[51];
  setlocale(LC_ALL,"Portuguese");

  printf("Digite o nome do seu personagem\n\n");
  scanf(" %50[^\n]",nome);
  printf("\n\n%s é um homem que passa por dificuldades na vida, em contrapartida após inúmera
  printf(" e fracassos sucessivos ficou a mercê de escolhas que foram aparecendo à medida que
  printf(" de garantir um futuros de sua família e não vivenciar as dificuldades enfrentadas,
  printf(" na qual irá mudar seu destino.\n\n");
  tela();
  printf("\n%s é policial desmerecido e que busca chegar ao alto escalão do comando de operaç
  printf("Depois de todas essas tentativas frustradas, %s vê a oportunidade certa, mesmo não
  printf("Os principais suspeitos são seus rivais políticos, sabendo disso %s sabe que se des
  printf("\n\nMuito bem. Sua jornada começa agora mesmo. Boa sorte.\n\n");
  tela();
  Personagem *p = criarPersonagem(nome);
  casa(p);
  return 0;
}
```

Função que conterá as interações

Mais compacta.

Menos legível.

Estruturas de armazenamentos

- Registros (structs).
- Geralmente mais de um.
- Vocês verão mais detalhes sobre structs no decorrer do curso.

Exemplo:

```
typedef struct local{      /// Estrutura do Local

    int id_sala; // indetificador da sala
    int id_per; // identificador do personagem

} Local;

typedef struct personagem { /*Estrutura do personagem geral*/

  char nome[81];
  char frase[201];
  int life;
  int forca;

  } Personagem;
```

Números randômicos

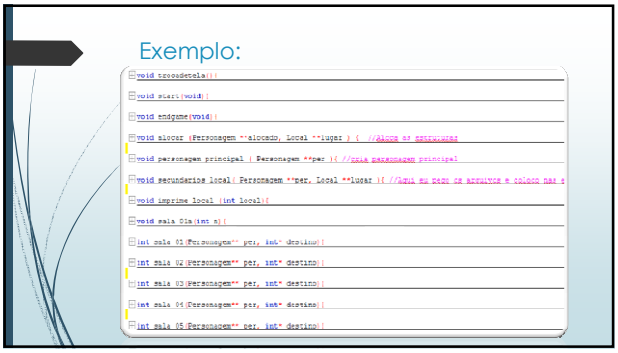
- <stdlib.h> e <time.h>
- srand() e time()
- rand()

Exemplo:

```
#include <stdio.h>
#include <stdlib.h>     // Para funções rand() e srand()
#include <time.h>       // Para função time()

   ...

   int i;
   float f;

   srand(time(NULL));

   i = rand();             //valor de i: [0, RAND_MAX]
   i = rand() % 6;         //valor de i: [0,5]
   i = (rand() % 6) + 1;   //valor de i: [1,6]
   i = (rand() % 10) * 10  //valor de i: [0,90]
   f = (rand() % 10) * 1.1 //valor de i: [0,9.9]

   ...
```

Funções extras

- A maioria das ações será através de funções:
- Ex: Combate, criação do personagem, etc...
- Muitas delas serão criadas por vocês.
- Uso extensivo, com nomes intuitivos, ajuda na leitura do código.


Exemplo:
void trocadetela();
void start(void);
void endgame(void);
void alocar (Personagem **alocado, Local **lugar) {
void personagem principal (Personagem **per){
void secundarios_local(Personagem **per, Local **lugar){
void imprime_local (int local){
void sala_01a(int s){
int sala_01(Personagem** per, int* destino);
int sala_02(Personagem** per, int* destino);
int sala_03(Personagem** per, int* destino);
int sala_04(Personagem** per, int* destino);
int sala_05(Personagem** per, int* destino);