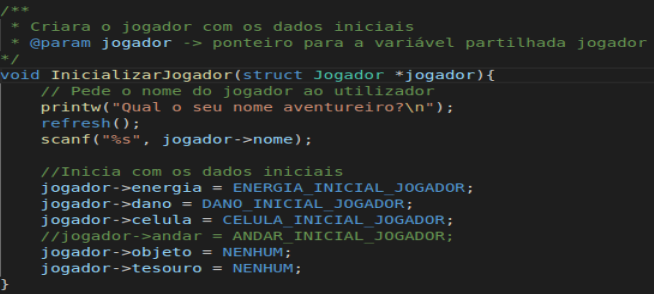
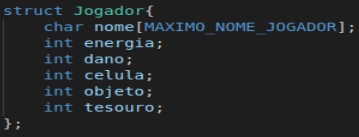
Criação do elementos do jogo

Criação de um processo para realização de um menu onde verifica se o utilizador é um super user, se ele quer carregar um jogo, ou se ele quer começar do 0..



1. Criação do jogador

Criamos a estrutura do jogador onde tem os atributos do jogador.

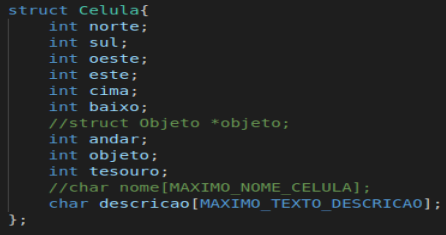


Depois vamos à função onde perguntamos ao utilizador qual será o nome do jogador.

E damos os valores iniciais pré definidos.

1. Criação do local de aventura

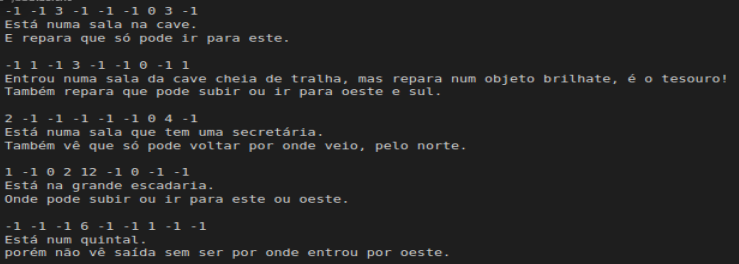
Criamos a estrutura do jogador onde tem os atributos de cada Célula do mapa



De seguida, iremos inicializar o mapa, para tal, iremos ao ficheiro do mapa e fazemos o tratamento dos dados parao vetor com estruturas do tipo células., onde guarda as diversas salas do mapa

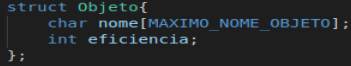


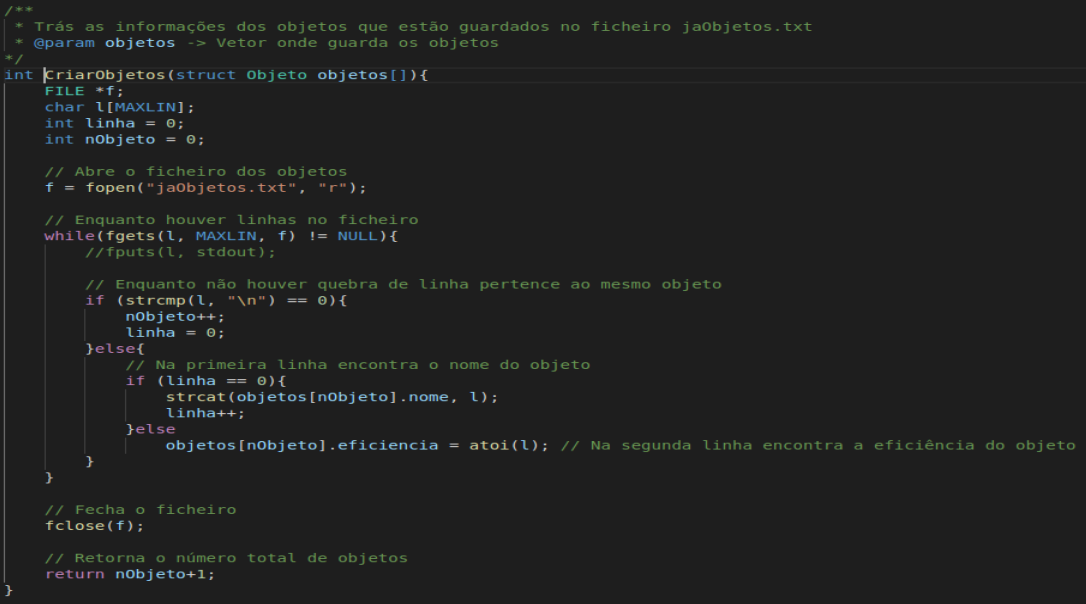
Onde ele irá buscar os dados da sala ao seguinte ficheiro



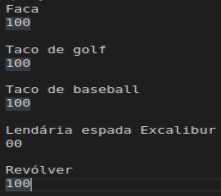
1. Criação de objetos

Mesma merda que o mapa só que objetos do mapa



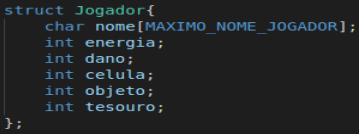


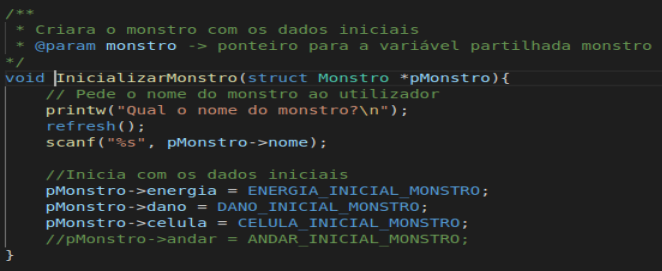
Onde irá buscar os dados ao seguinte ficheiro



1. Criação do monstro

Igual ao jogador só que em monstro xD





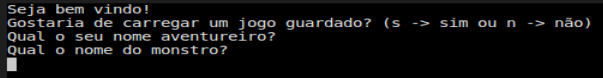
**Resultado**

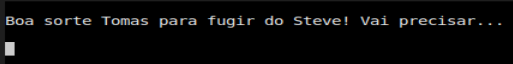
Aqui mostramos como é que a criação se aplica no projeto.

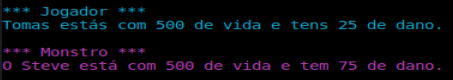
Primeiro pergunta ao jogador se quer carregar um jogo já criado, depois pergunta o nome do jogador, e por fim o nome do monstro.

É limpo o conteúdo da criação do ecrã e é mostrado uma mensagem de boas vindas.

Por fim é limpa a mensagem de boas vindas do ecrâ e é mostrado os dados do monstro e do jogador.

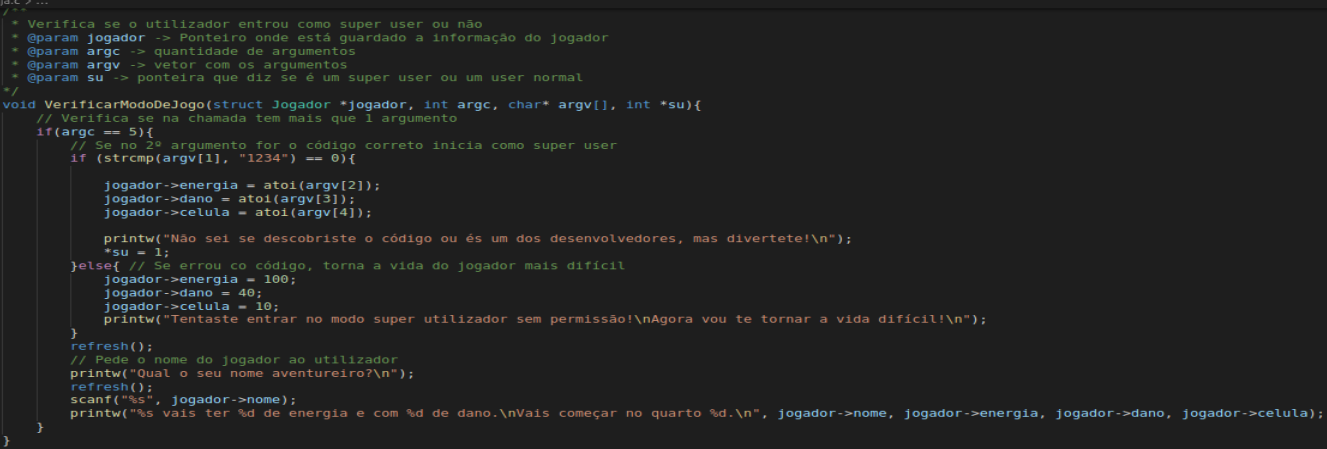






Modo Super User

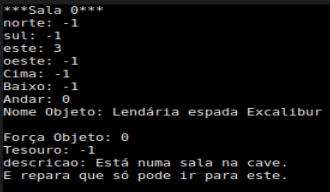
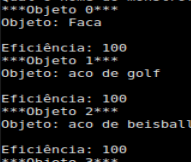
Aqui é criado um modo de super usrer que se os dados da chamada do programa foram introduzidos corretamente, o utilizador decide a vida, o dano, e em que sala é que irá começar. Mas se ele introduzir o código errado o jogo vai se tornar muito mais difícil, como uma forma de castigar por tentar trapacear.



Que atravéss do seguinte comando é acionado o modo, sendo que os valores podem mudar.



No caso se for super user é apresentado as informações sobre o mapa e os objetos.



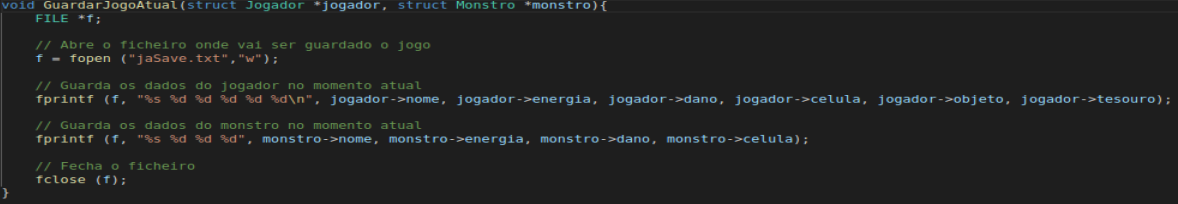
Além de que em todas as jogadas é apresentada a informação sobre onde o monstro está.



Guardar Situação do jogo

Aqui a qualquer momento do jogo o jogador pode guardar a situação do jogo. Além de que ao iniciar o jogo temos a opção se queremos carregar um jogo que foi salvo anteriormente.

Para guardar o jogo, guardamos as informações do jogador, se em que sala é que ele se enonctra e do monstro no momento em que o jogo é guardado.



O jogo é guardado da seguinte forma

O jogador é guardado, o nome, a energia, o dano, a sala, o objeto e o tesouro.

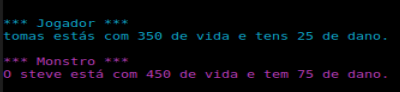
O monstro é guardado, o nome, a energia, o dano e a sala.



Sendo que inicialmente é perguntado ao jogador se quer carregar um jogo já guardado.



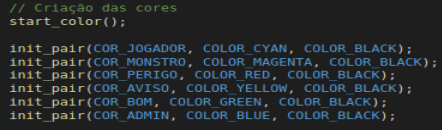
Aqui demonstramos o carregamento do jogo



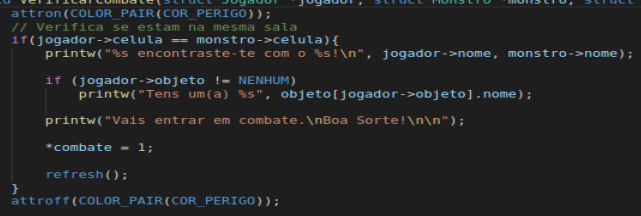
Biblioteca ncurses

Como já foi mostrado, usamos a biblioteca ncurses para conseguimos melhorar a interface do jogo, mudando as cores do texto e limpando o ecrã de forma a termos menos informações que já não são necessárias a aparecerem no ecrã, através do comando clear.

Aqui temos a criação das diferentes cores que depois são atribuídas às diferentes mensagens.

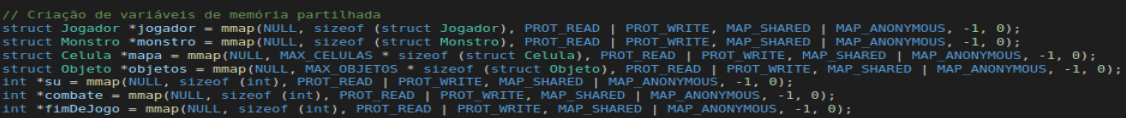


Que atravès do attronon e do attronoff conseguimos selecionar qual a cor que queremos atribuir a cada texto.

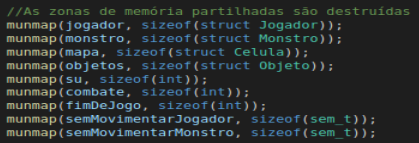


Criação de Processos

Para conseguirmos criar processos que possam aceder à mesma variável necessitamos de criar uma variável de memória partilhada. Assim sendo foi necessário atribuir as seguintes variáveis esse fator.

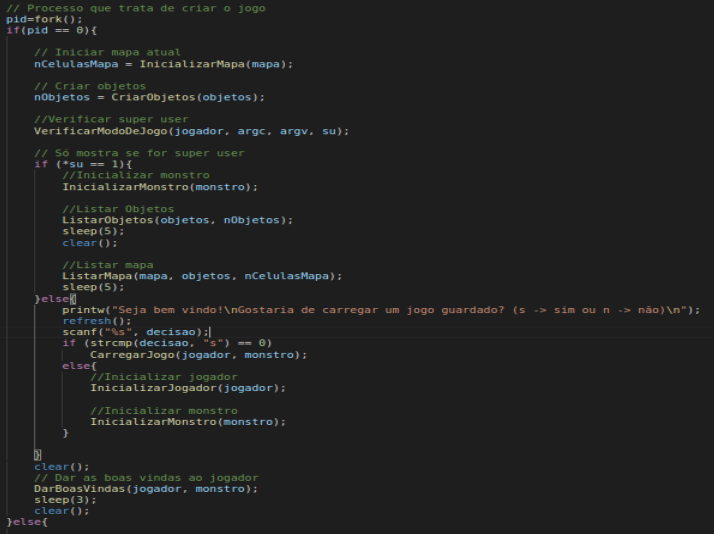


Onde no final necessitam de ser destruídas



1. Criação do jogo

Primeiramente criamos um processo que trata da criação do jogo, ou seja inicialização do jogador, do monstro, do mapa e dos objetos

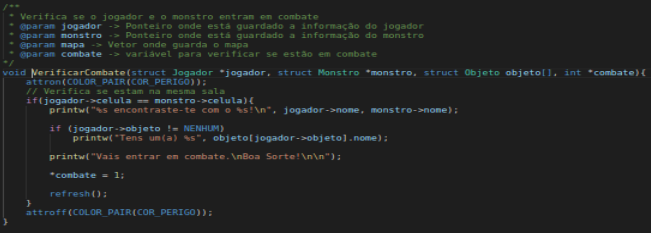


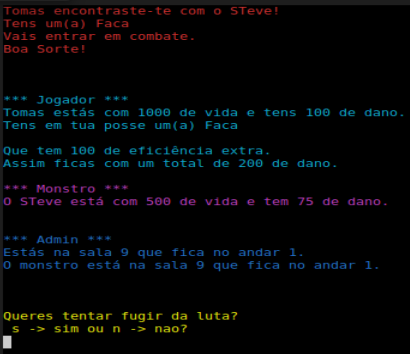
1. Processo principal

Este processo trata da criação dos processos de movimentação do monstro do jogador, e de combate, além de estar sempre a verificar se o jogador e o monstro se encontram na mesma sala para batalharem, além de verificar se chegou a um dos possíveis finais.

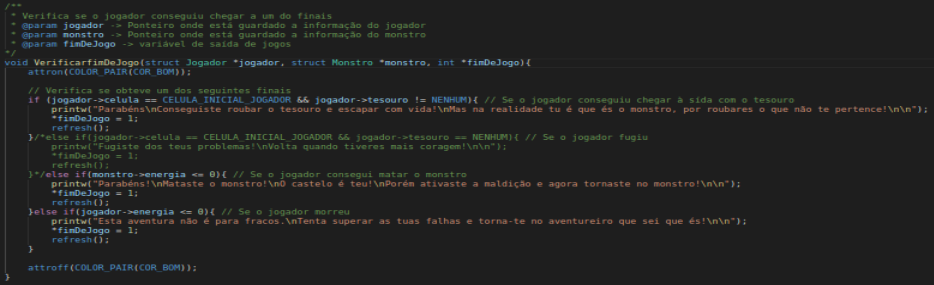


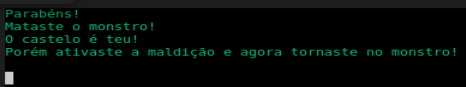
Para verificar se o monstro e o jogador entraram em combate é feita uma verificação para ver se eles se encontram na mesma sala.





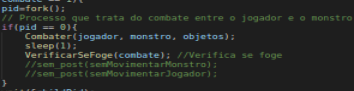
Também tem que verificar se o jogador obteve algum dos seguintes finais



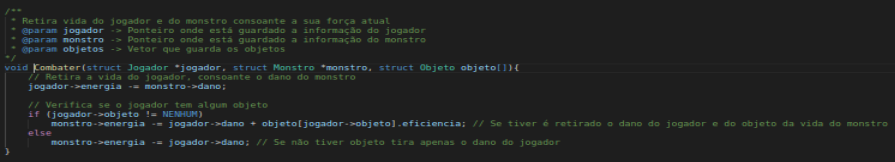


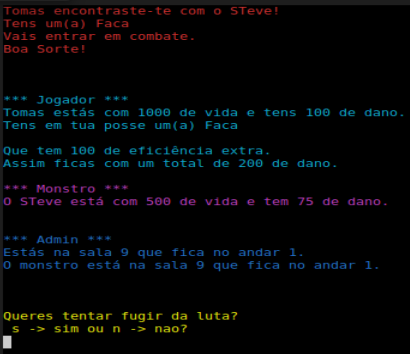
1. Processo de luta

Este processo trata de tirar a vida do jogador e do monstro, além de ver se o utilizador conseguirá escapar da luta.

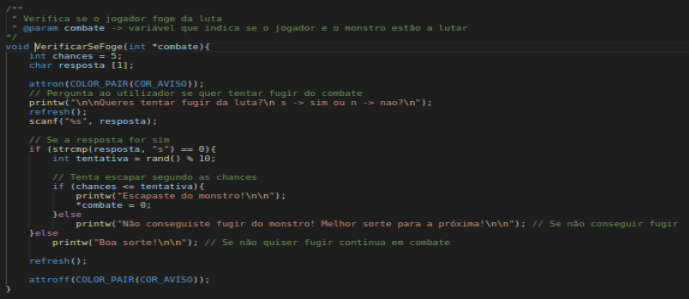


Se estes entrarem em combate é retirada a vida a cada um deles consoante o dano que possuem, e no caso do monstro, se o jogador tiver em sua posse um objeto, é retirado esse dano extra ao monstro.



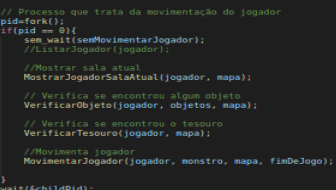


Para verificar se foge, é perguntado ao jogador se este quer tentar fugir do monstro, se a resposta for sim este terá 50% de chances de fugir.

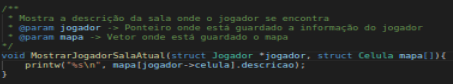


1. Movimentação do jogador

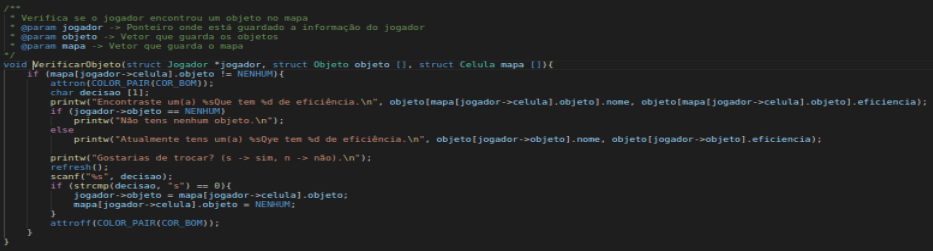
Aqui é tratada a movimentação do jogador, apresentando a descrição de onde este se encontra, além de verificar se este encontrou algum objeto na sala, ou o tesouro.

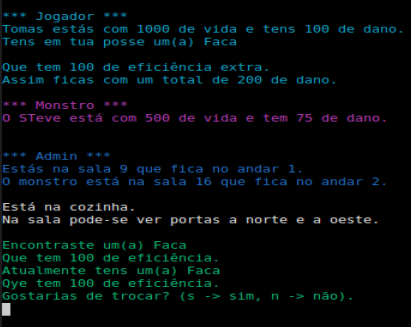


Para mostrar a descrição da sala apenas é apresentado em texto a descrição

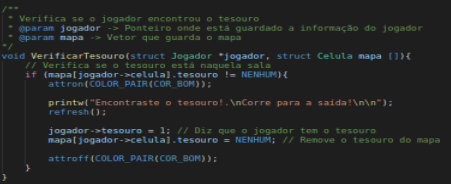


Ao verificar se existe algum objeto na sala, se é encontrado, mostra ao jogador o objeto e a suma eficiência que o jogador possui no momento e pergunta se quer trocar pelo novo objeto mostrando as suas características.



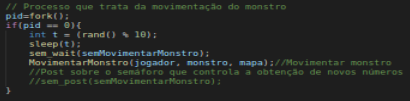


Para o tesouro é apenas apresentada uma mensagem a dizer que encontrou o tesouro.



1. Processo movimentação monstro

O seguinte processo movimenta o monstro ao fim de um tempo aleatório.



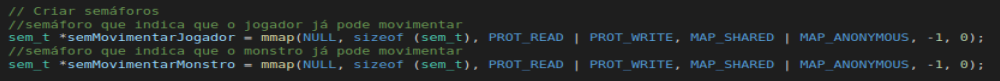
Para o movimentar, criamos um algoritmo que verifica se o monstro está em um andar diferente do jogador, se sim, este tenta ir diretamente para as escadas, para ir para o mesmo andar que o jogador. Se estiverem no mesmo andar o monstro irá se movimentar aleatoriamente pelo andar.



Sincronização de processos

Através de semáforos é possível fazermos a sincronização de processos. No nosso caso foi necessário criar sincronização em relação a quando o jogador e o monstro estão a batalhar.

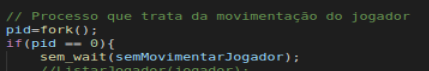
Para tal criamos 2 semáforos, um para a movimentação do monstro e outro para a movimentação do jogador.



No final é necessária a eliminação dos semáforos criados.



Para a utilização destes semáforos nos processos da movimentação do monstro e do jogador, informamos que só quando os semáforos estão a 1 é que o processo pode ser executado.



Demonstração do jogo

Agora iremos demonstrar uma demo do jogo.

Iremos iniciar como super user



