Introdução

Descrição do jogo

- R-Type é um jogo no estilo Shoot 'em up que não possui fim definido. Você controla a nave R-9C War-Head, e seu objetivo é destruir ou evitar quaisquer obstáculos em seu percurso, como asteroides ou outras naves inimigas.

- O jogador pode destruir naves ao atirar nelas, pressionando a barra de espaço. Há dois modos de tiro: padrão e carregado.

  - Tiro comum: Ao colidir com qualquer superfície, o projétil é destruido.
  - Tiro carregado: Atravessa qualquer superfície.

- O jogador não pode destruir asteroides.

- A cada nave destruída, é concedida uma determinada quantidade de pontos, que varia para cada tipo de inimigo. Ao final do jogo, caso sua pontuação supere o recorde, esta será armazenada em arquivo.

- Apesar de não ter fim, o jogo se torna mais difícil a cada 15s decorridos. Com isso, objetos passam a se mover mais rápido, e a taxa de geração de inimigos é aumentada.

- O jogo termina quando a nave é atingida três vezes por qualquer obstáculo. Quando isso acontece, o jogador deve optar por tentar novamente ou encerrar a sessão.

Controles

- Movimentação

  - W: Mover para cima
  - A: Mover para esquerda
  - S: Mover para baixo
  - D: Mover para direita

- Atirar

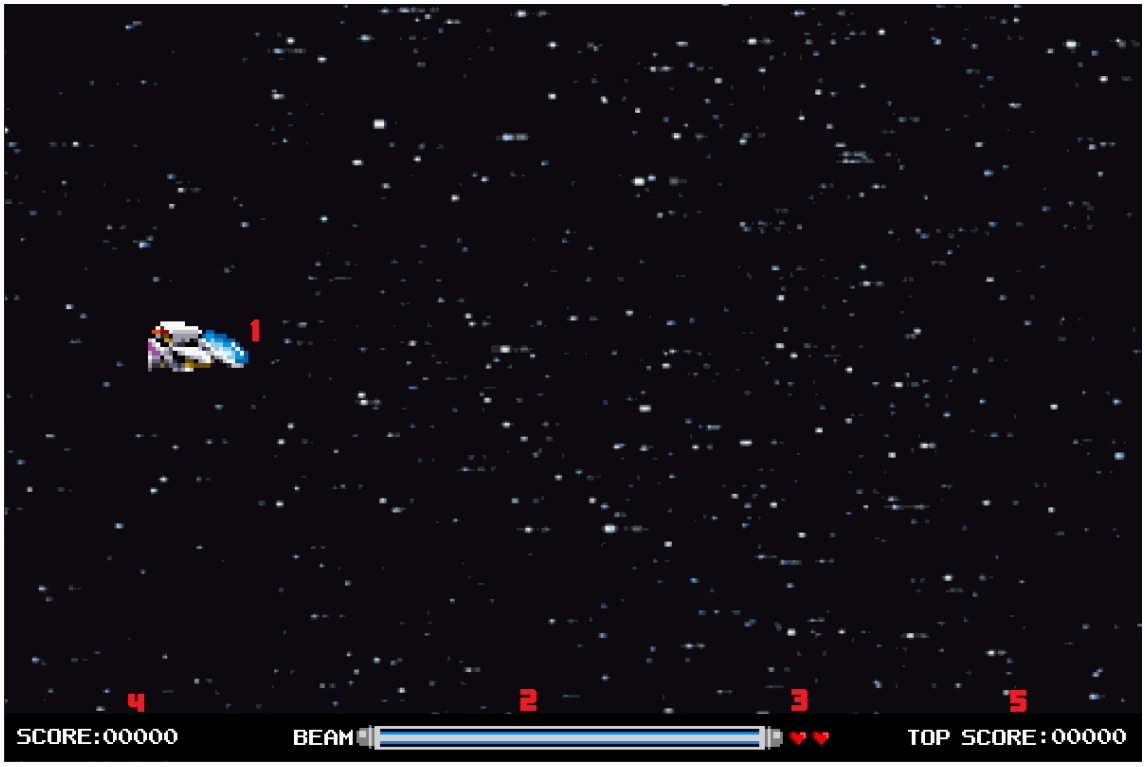
  - Espaço (segurar): Carregar tiro
  - Espaço (solar): Solar tiro

Cenário

- Tela de início

  - Animação de introdução ao jogo.
  - Para começar a jogar, pressione a tecla ENTER.

- Interface do jogo



1. Nave controlada pelo jogador (R-9C War-Head)
2. Barra para carregar tiro
3. Vidas extra restantes
4. Pontuação
5. Recorde

Recursos do jogo

Ambiente

- O ambiente do jogo é gerado através de especificações globais;

  - SCREEN_H: Altura da tela
  - SCREEN_W: Largura da tela
  - BOUNDS: Limitação de movimento nas bordas da tela
  - FPS: Frames por segundo

- O ambiente é construído através da chamada das variáveis globais em funções, que constroem a janela e os temporizadores, usados em diferentes situações

Nave

- A struct Nave possui variáveis referentes à posição, cor, movimentação e ivulnerabilidade do objeto nave.

- A hitbox da nave é desenhada na forma de um retângulo, de posição variável (inicialmente em x = 150, y = SCREEN_H/2) e dimensão fixa, especificada através de variáveis globais.

Controles de movimento

- O movimento da nave é realizado através da mudança de posição do objeto nave.

- Os comandos de teclado alteram a variável que indica o estado de movimentação. Esta, por sua vez, altera a posição da nave nos eixos X e Y para a direção desejada, em função da variável VELOCIDADE_NAVE.

- A atualização de posição ocorre repetidamente, até que a tecla de movimentação seja solta.

Tiro

- A struct Tiro possui variáveis referentes à posição, velocidade, cor e dimensão do projétil, tal como variáveis booleanas que auxiliam nas especificações de seu comportamento

- Todas as funções referentes ao tiro recebem um vetor de dimensão equivalente à variável global NUM_BALAS. Com isso, é possível inicializar e atualizar múltiplos projéteis independentes de forma simultânea.

- A hitbox do tiro é desenhada na forma de um quadrado, com posição de origem à direita da nave que, quando liberado, caminha para a direita, mantendo sua posição Y até alcançar o final da tela

- Ao pressionar a barra de espaço, o tiro é criado à direita da nave e permanece estático, acompanhando apenas a movimentação vertical do objeto. Neste momento, está em estado de carregamento.

- Ao soltar a barra de espaço, o projétil se desprende da nave, é ativado e caminha para a direita até se chocar com outro objeto ou chegar ao final da tela.

- Caso o tiro permaneça por tempo suficiente em estado de carregamento, este se comportará como carregado, e sua trajetória não será interrompida por colisões. Desta maneira, atravessa qualquer objeto em sua trajetória.

Bloco

- A struct Bloco possui variáveis referentes à posição, cor e dimensão do objeto bloco.

- A hitbox do bloco é desenhada por um retângulo, que é criado fora dos limites da tela, com posições X e Y geradas aleatoriamente.

- É um objeto indestrutível, que destroi qualquer objeto com que faz contato, com exceção do tiro carregado.

- Se move da direita para esquerda e, ao chegar ao final da tela, é recriado com suas posições originais.

Inimigos

- A struct Inimigo é a mais complexa do código, pois possui variáveis que se comportam de maneira distinta ao receberem determinados valores da variável quantidade, que indica qual inimigo será inicializado.

- As variáveis gerais dos inimigos são referentes à posição, velocidade, cor, dimensão e vida dos objetos.

- Assim como o tiro, todas as funções referentes ao inimigo recebem um vetor de dimensão equivalente à variável quantidade, que indica, simultaneamente, a quantidade de objetos criados por vez e como estes irão se comportar.

- A variável quantidade, chamada por todas as funções do inimigo, recebe um valor aleatório sempre que determinadas condições são atingidas. Desta maneira, a geração de diferentes inimigos é aleatória, e depende do valor gerado pela função aleatorizaInimigo().

- Gemeos (quantidade = 2)

  - São criados dois objetos, um acima do outro, cujas hitbox são desenhadas por um retângulo.

  - Compartilham a mesma dimensão e posição X a todo momento.

  - Ao chegarem na posição X de 4/5 da largura da tela, os gêmeos disparam dois projéteis cada, que se movem para a esquerda com velocidade superior à deles, em trajétoria diagonal.

  - O comportamento dos projéteis é semelhante aos dos tiros da nave, e possuem suas respectivas posições, dimensões, velocidades, cor e uma variável booleana para indicar se está ativo ou não.

  - Tanto os inimigos quanto seus respectivos tiros deixam de existir ao colidir com outro objeto ou alcançar os limites da tela.

- Cobra (quantidade = 6)

  - São criados seis objetos, um ao lado do outro, cujas hitbox são desenhadas por um quadrado.

- Caminham para a esquerda a todo momento, descrevendo um movimento senóide no eixo Y.

- Cada segmento da cobra é independente, ou seja, para destrui-la por completo é necessário 6 tiros ou 1 tiro carregado.

- Os segmentos deixam de existir ao colidir com outro objeto ou alcançar os limites de tela.

- A cada vez que o percurso de um inimigo é concluido, este é reinicializado com seus parâmetros de origem, respectivos ao novo valor para quantidade gerado pela função aleatorizaInimigo().

Colisões

- Com exceção do tiro carregado, dois objetos nunca se sobrepõem na tela do jogo. Quando se colidem com outro objeto, estes deixam de existir. Há dois tipos de colisão:

- Colisões da nave

  - Colisões são detectadas quando as posições da nave sobrepõem as de qualquer outro objeto na tela. Quando isso ocorre, a nave é danificada e o jogador perde uma vida.

  - Ao colidir, a nave entra em um curto estado de ivulnerabilidade para evitar a detecção de múltiplas colisões com um mesmo objeto.

  - A nave pode se colidir com inimigos, tiros do inimigo e blocos.

- Colisões de demais objetos

  - Demais colisões são detectadas entre tiros, inimigos, tiros do inimigo e blocos.

  - Com exceção do bloco e do tiro carregado, objetos deixam de existir ao colidirem com outro.

Vidas

- A cada instância de jogo, a nave inicia com duas vidas extras, ou seja, pode se colidir duas vezes sem ser destruida.

- Na terceira colisão, o jogo é encerrado.

Dificuldade

- A cada 15s decorridos em uma instância de jogo, esta se torna mais difícil. Com isso, o timer fundamental do jogo se torna mais rápido, e a velocidade de atualização dos objetos se torna, também, mais rápida.

- Quando isso ocorre, progressivamente é aumentada a velocidade de movimentação dos objetos, tal como a velocidade de geração destes.

Pontuação

- A cada inimigo destruído, são concedidos pontos, que são exibidos em tempo real na interface do jogo.

- Gêmeos concedem 150 pontos cada, e a cobra concede 50 pontos por segmento.

- A pontuação máxima obtida é armazenada em recorde, que também é exibida na interface. Durante uma instância de jogo, caso a pontuação atual supere à do recorde, esta se tornará o novo recorde, e será armazenada em arquivo ao final do jogo.

Sprites

- Sprites são ilustrações que acompanham as posições de todos os objetos do jogo. São desenhadas por cima da hitbox, que existe de forma invisível e tem a principal função de detectar colisões.

- Alguns sprites são estáticos, enquanto outros são animados.

- As animações foram realizadas de diferentes maneiras para atender aos resultados desejados. Alguns exemplos são

  - Nave: A animação do sprite da nave é chamada através dos controles de movimento. Dependendo da tecla pressionada, a nave é ilustrada andando para cima ou para baixo

  - Tela de início e background: Animação em loop, que intercala frames através de um switch em função de um timer. Estas são automáticas, ou seja, não dependem de interações do jogador para ocorrer.

    OBS: Cada animação automática possui seu timer específico, para controlar de maneira específica a velocidade da troca de frames.

- Animação da cobra: Animação em função do ΔY dos segmentos, em que os frames são alterados na medida em que o objeto está se movendo para cima ou para baixo, na finalidade de suavizar o movimento senoidal do inimigo.

Efeitos sonoros

- São categorizados em músicas de fundo e efeitos.

- Músicas são reproduções sonroas extensas, que tocam em loop. Para melhor qualidade, estas são carregadas no mixer padrão da biblioteca, e alternadas para diferentes momentos do jogo, como a tela de abertura, o início e a tela de Game Over.

- Efeitos são reproduções sonoras simples, desencadeadas por determinados eventos no jogo. Alguns exemplos são tiros, colisões com a nave ou pressionar enter para começar o jogo na tela de início.

Estrutura do código

Bibliotecas

- Bibliotecas Allegro
- Essenciais da linguagem C
- <time.h> para função rand();

Variáveis Globais

- Especificações da tela
- Especificação do FPS
- Especificações das dimensões e velocidade da nave
- Especificação das dimensões do bloco
- Número de balas para o tiro da nave

Funções

- Random: Aplicada em diversas situações para gerar um número aleatório

- Struct Nave: Especificações gerais da nave, que são inicializadas e modificadas nas seguintes funções:

  - InitNave: Atribui valor às variáveis da struct nave

  - DesenhaNave: Desenha a hitbox da nave (retângulo) nas coordenadas especificadas

  - AtualizaNave: Movimenta a nave em função das variáveis dir_x e dir_y (controladas pelo teclado)

  - ParaCima, ParaBaixo, ParaDireita, ParaEsquerda: Muda o valor das variáveis dir_x e dir_y ao pressionar as teclas WASD

    W: dir_y += -VELOCIDADE_NAVE
    S: dir_y += VELOCIDADE_NAVE
    A: dir_x += -VELOCIDADE_NAVE
    D: dir_x += VELOCIDADE_NAVE

- Struct Bloco: Especificações gerais do bloco, que são inicializados e modificados nas seguintes funções:

  - p