CÂMPUS CHARQUEADAS

RAFAEL TEIXEIRA DE ABREU
SAMUEL TOILLIER DE MORAES

BREAKOUT:

RELATÓRIO FINAL

Charqueadas

Maio de 2021

1- Funcionamento das nossas funções:

void movimentoJose:

É a função que define o nosso movimento contínuo. Na nossa lógica, definimos 4 tipos de movimentos, conforme imagem abaixo:

 move19	 move73
 move91	 move37

Onde:

move19: Adiciona uma posição da bola em X e subtrai uma em Y dentro dos limites do mapa até que haja a colisão com um caractere ASCII 178 ou o limite do mapa. Caso a bola colida com a parede direita, muda para a move37. Caso a bola colida com a parede superior, muda para a move73. Caso a bolinha colida com um caractere ASCII 178, apaga o caractere (substituindo por um espaço) e muda para move73.

move73: Adiciona uma posição da bola em X e Y dentro dos limites do mapa até que haja a colisão com um caractere ASCII 178 ou o limite do mapa. Caso a bola colida com a parede direita, muda para a move91. Caso a bola colida com a paleta, muda para a move19. Caso a bolinha colida com um caractere ASCII 178, apaga o caractere (substituindo por um espaço) e muda para move19.

move91: Subtrai uma posição da bola em X e adiciona em Y dentro dos limites do mapa até que haja a colisão com um caractere ASCII 178 ou o limite do mapa. Caso a bola colida com a parede esquerda, muda para a move73. Caso a bola colida com a paleta, muda para a move37. Caso a bolinha colida com um caractere ASCII 178, apaga o caractere (substituindo por um espaço) e muda para move37.

move37: Subtrai uma posição da bola em X e Y dentro dos limites do mapa até que haja a colisão com um caractere ASCII 178 ou o limite do mapa. Caso a bola colida com a parede esquerda, muda para a move19. Caso a bola colida com a parede superior, muda

para a move91. Caso a bolinha colida com um caractere ASCII 178, apaga o caractere (substituindo por um espaço) e muda para move91.

Caso a bolinha chegue na linha 22 (BOLA_POS_Y == 22), é descontada uma vida e o jogo recomeça com o mapa igual a vida anterior, até que atinja o limite de 5 vidas.

Além disso, para que a bolinha acesse todas as posições com blocos, implementamos ao código “uma escorregada”, onde pulamos uma posição no eixo X.

void placar:

Imprime pontuação atual na posição X = 11 e Y = 24. Imprime a quantidade de vidas atual na posição X = 26 e Y = 24. Imprime nível atual na posição X = 41 e Y = 24.

void zeroMaquina:

Ao jogador perder uma vida, imprime a paleta na posição inicial e a bolinha na posição inicial, mantendo o estado do mapa e tamanho da paleta da vida anterior até que as vidas sejam zeradas.

void binomo:

A partir da função srand e rand, é gerado um número aleatório de 0 a 10. Este número é divido para que seja descoberto se é par ou ímpar. Com isso, fazemos com que o lado para qual a bolinha vai no começo do jogo é sempre aleatório.

void reiniciaMapa:

Quando todos os blocos são eliminados, limpamos a tela (limparTela), recarregamos o cenário (carregarCenario) e escrevemos o cenário novamente (escreverCenario), decrementando em 1 o comprimento da paleta e incrementando o nível em 1, deixando a bolinha e as paletas na posição inicial novamente.

2- Funcionamento geral do código (int main):

É executada srand para que o número aleatório seja diferente a cada execução do código, garantindo a aleatoriedade. É impresso o menu inicial do jogo e é escaneado o valor digitado pelo usuário para selecionar o modo jogar ou o automático.

Caso seja digitado 1, o jogador escolhe uma velocidade de 0 a 50 para jogar, utilizando as teclas esquerda e direita para controlar a paleta.

Caso seja digitado 2, o jogador tem medinho de jogar e prefere assistir, então ele escolhe a velocidade de jogo e assiste a condição autonomia trabalhar (o jogo funciona automaticamente).

Caso seja digitado um valor diferente de 1 ou 2, é impresso uma mensagem de “valor inválido” e é solicitado uma entrada novamente.