Programação I 2023/2024

Trabalho de grupo

Ouri

Descrição do problema

Pretende-se recriar um jogo chamado Ouri utilizando a linguagem C em programação. É necessário recriar o jogo seguindo todas as regras do jogo original e criar medidas de segurança de modo a minimizar o máximo de bugs possíveis. De modo a desenvolver este jogo, foram criadas quatro funções para além do main, a função tabuleiro, computador, jogadorA e jogadorB. Com a realização deste trabalho foram encontradas algumas dificuldades.

Dificuldades

- De modo a adicionarmos pedras às casas seguintes tivemos que criar um ciclo "for" onde adicionamos pedras às casas seguintes dependendo do valor da casa inicial escolhida pelo utilizador e de modo ao jogo decorrer ao sentido contrário ao relógio. A última casa seria o 11 e a primeira o 0 pelo que quando desse a volta completa teria que ir da posição 11 para a 0, neste caso tivemos dificuldade pois não só tínhamos que ir da última posição para a inicial, como tínhamos de adicionar as bolas que faltavam nas casas seguintes. De modo a resolver este problema uma das soluções foi quando chegasse a casa 11, o "i" do ciclo "for" voltava a ser 0 e eram subtraídas i das bolas totais da casa inicial.
- Tivemos dificuldade em perceber como funcionava a manipulação de ficheiros pelo que recorremos à ajuda da internet de forma a entender como funcionava.

Funções criadas

Função tabuleiro

Esta função tem como propósito imprimir na tela o tabuleiro do jogo ouri, esta recebe dois arrays como argumentos,”casas[]”e “depositos[]”, representando o estado atual do jogo.

´casas[]´: é um array que representa as casas do tabuleiro, onde guarda o número de pedras contidas em cada cada, tendo 12 casas no total.

´depositos[]´: é um array que representa os depósitos do jogo, onde guarda o número de pedras capturadas.

A função não possui um valor de retorno, sendo apenas responsável por exibir o estado atual do jogo, em que cada casa contém o seu número de pedras e os depósitos apresentam as pedras capturadas.

Função computador

Esta função tem como propósito simular a jogada do computador, fazendo escolhas aleatórias porém atendendo às regras específicas do jogo. Esta função recebe três argumentos, ´casas[]´,´depositos[]´ e soma.

´casas[]´: é um array que representa as casas do tabuleiro, onde contém o número de pedras contidas em cada cada, tendo 12 casas no total.

´depositos[]´: é um array que representa os depósitos do jogo, onde guarda o número de pedras capturadas.

´soma´: onde representa a soma das pedras existentes em todas as casas de um lado do tabuleiro.

Esta função dependendo do valor recebido em soma pode funcionar de duas formas, se soma for diferente de 0, ou seja o tabuleiro oposto ainda tem pedras, funciona da forma normal ou seja é escolhido um número aleatório de 1 a 6 de modo a escolher a casa, e apartir dai essa casa fica com 0 pedras e o número de pedras que tinha nessa casa

é adicionado às casas seguintes uma a uma. Caso o valor de soma for igual a 0, ou seja todas as casas do lado oposto do tabuleiro tem 0 pedras, é escolhido uma casa que contenha pedras suficientes de modo a colocar pedras no lado oposto do tabuleiro de modo a continuar o jogo. Caso isso não seja possível, o jogo acaba e ganha quem tem o valor mais elevado no array depósitos.

Função jogadorA

Esta função tem como propósito representar o jogador A, realizando as jogadas do mesmo dependendo da casa escolhida pelo jogador. Esta função recebe três argumentos, ´casas[]´,´depositos[]´ e soma.

´casas[]´: é um array que representa as casas do tabuleiro, onde contém o número de pedras contidas em cada cada, tendo 12 casas no total.

´depositos[]´: é um array que representa os depósitos do jogo, onde guarda o número de pedras capturadas.

´soma´: onde representa a soma das pedras existentes em todas as casas de um lado do tabuleiro.

Esta função dependendo do valor recebido em soma pode funcionar de duas formas, se soma for diferente de 0, ou seja o tabuleiro oposto ainda tem pedras, funciona da forma normal ou seja é escolhido um número pelo jogador A de 1 a 6 de modo a selecionar a casa pretendida, e apartir daí essa casa fica com 0 pedras e o número de pedras que tinha nessa casa é adicionado ás casas seguintes uma a uma. Caso o valor de soma for igual a 0, ou seja todas as casas do lado oposto do tabuleiro tem 0 pedras, é escolhido uma casa que contenha pedras suficientes de modo a colocar pedras no lado oposto do tabuleiro de modo a continuar o jogo caso isso não seja possível o jogo acaba e ganha quem tem o valor mais elevado no array depósitos.

Função jogadorB

Esta função tem como propósito representar o jogador B, realizando as jogadas do mesmo dependendo da casa escolhida pelo

jogador. Esta função recebe três argumentos, ´casas[]´,´depositos[]´ e soma.

´casas[]´: é um array que representa as casas do tabuleiro, onde contém o número de pedras contidas em cada cada, tendo 12 casas no total.

´depositos[]´: é um array que representa os depósitos do jogo, onde guarda o número de pedras capturadas.

´soma´: onde representa a soma das pedras existentes em todas as casas de um lado do tabuleiro.

Esta função dependendo do valor recebido em soma pode funcionar de duas formas, se soma for diferente de 0, ou seja o tabuleiro oposto ainda tem pedras, funciona da forma normal ou seja é escolhido um número pelo jogador B de 1 a 6 de modo a selecionar a casa pretendida, e a partir daí essa casa fica com 0 pedras e o número de pedras que tinha nessa casa é adicionado às casas seguintes uma a uma. Caso o valor de soma for igual a 0, ou seja todas as casas do lado oposto do tabuleiro tem 0 pedras, é escolhido uma casa que contenha pedras suficientes de modo a colocar pedras no lado oposto do tabuleiro de modo a continuar o jogo caso isso não seja possível o jogo acaba e ganha quem tem o valor mais elevado no array depósitos.

Função guardarficheiros

Esta função tem como propósito criar um ficheiro casa não exista ou apagar o anterior caso exista e guardar dentro dele o estado atual do jogo caso o utilizador deseje. Esta função recebe dois arrays como argumentos,”casas[]”e “depositos[]”, de modo a receber as informações para guardar no ficheiro de texto.

´casas[]´: é um array que representa as casas do tabuleiro, onde guarda o número de pedras contidas em cada cada, tendo 12 casas no total.

´depositos[]´: é um array que representa os depósitos do jogo, onde guarda o número de pedras capturadas.

A função não possui um valor de retorno, sendo apenas responsável por guardar os valores do estado atual do jogo, em que cada casa

contém o seu número de pedras e os depósitos contém o número de pedras capturadas.

Função guardarficheiros

Esta função tem como propósito criar um ficheiro casa não exista ou apagar o anterior caso exista e guardar dentro dele o estado atual do jogo caso o utilizador deseje. Esta função recebe dois arrays como argumentos,”casas[]”e “depositos[]”, de modo a receber as informações para guardar no ficheiro de texto.

´casas[]´: é um array que representa as casas do tabuleiro, onde guarda o número de pedras contidas em cada cada, tendo 12 casas no total.

´depositos[]´: é um array que representa os depósitos do jogo, onde guarda o número de pedras capturadas.

A função não possui um valor de retorno, sendo apenas responsável por guardar os valores do estado atual do jogo, em que cada casa contém o seu número de pedras e os depósitos contém o número de pedras pedras capturadas.

Função bolas12243648

Esta função tem como propósito assegurar uma das regras do jogo em que caso as bolas sejam superiores a 12, as pedras iram dar uma volta completa ao tabuleiro porém a casa inicial escolhida pelo utilizador mantém se com 0 pedras, esta função assegura que a casa inicial tenha 0 pedras e adiciona as pedras ás casas seguintes dependendo da quantidade de pedras que a casa inicial tinha, sendo as pedras adicionadas ás casas seguintes diferentes caso as pedras iniciais sejam 12, 24, 36 ou 48. Esta função recebe quatro argumentos senda elas: ”casas[]”, ”pedrasinicias”, ”ultimajogada e jogada”.

´casas[]´: é um array que representa as casas do tabuleiro, onde guarda o número de pedras contidas em cada cada, tendo 12 casas no total.

´pedrasinicias´: é um int que representa o número de pedras que a casa inicial escolhida pelo utilizador continha.

´ultimajogada´: é um int que representa a última casa em qual foi adicionada uma pedra.

´jogada´: é um int que representa a casa inicial selecionada pelo utilizador.

A função não possui um valor de retorno, sendo apenas responsável por assegurar uma das regras do jogo.

Função main

Esta função tem como propósito implementar o jogo em si, onde é possível escolher dois modos de jogo, jogador vs jogador ou jogador vs computador, a partir daí o mesmo é responsável para realizar o jogo, se a opção pretendida pelo jogador for 1, ou seja jogador vs jogador, são implementadas as funções jogadorA e jogadorB, caso a opção pretendida for 2, ou seja jogador vs computador, são implementadas as funções jogadorA e computador. Em ambas as opções é também implementado medidas de segurança de modo a assegurar as regras do jogo. No final desta função temos por fim a parte do código responsável para determinar o vencedor do jogo, onde é determinado pelo jogador que tenha o valor mais elevado dentro do array depositos.