WENDELL LUIS NERIS - 2311100035
BRUNO VENDRUSCOLO - 2221100004

USO DA MEMÓRIA:

.data:

SEEDS_INIT: variável tipo .word com valor 4, que é o valor de sementes que é setado em cada cavidade.

poco_p1: variável tipo .word com valor 24, que é o deslocamento do poço do jogador 1 em relação ao endereço base do tabuleiro.

poco_p2: variável tipo .word com valor 52, que é o deslocamento do poço do jogador 1 em relação ao endereço base do tabuleiro.

tamanho_tabuleiro: variável tipo .word com valor 56, que é o tamanho total do tabuleiro em bytes(14 cavidades * 4 bytes).

player_atual: variável do tipo .word com valor 1, indica qual é o jogador que está jogando 1 (para player 1) ou 2 (para player 2).

vitorias_p1 e vitorias_p2 : variáveis do tipo .word com valor 0, que é um contador de vitórias para cada player.

{**msg_bem_vindo, …, linha}:** variável do tipo .asciz, são todas strings com mensagens de texto para interação e impressão que aparecem no console.

.align:

.aling 2: para alinhar com o número de bytes de uma word.

tabuleiro: é um rótulo com uma alocação de .space 56 bytes, que contém 14 words de 4 bytes cada.

USO DOS REGISTRADORES:

s0: Ponteiro que armazena a base do tabuleiro, no código ele foi usado para calcular o endereço de qualquer cavidade.

s1: Armazenar o valor do jogador que está jogando(1 ou 2 para player 1 ou 2).

s2: Armazenar o deslocamento de 24 bytes do poço do jogador 1.

s3: Armazenar o deslocamento de 52 bytes do poço do jogador 2.

s4 - s7: Somar as sementes com os endereços dos poços.

a0 - a7: Passar argumentos, chamadas de sistema e retorno de funções(a0).

a0: Também usa os resultados retornados(1 ou 0) para jogar novamente ou acabar o jogo caso for 1 e para trocar de turno caso for 0.

ra: Armazenar o endereço de onde a instrução foi chamada para futuro retorno.

sp: Manipular as pilhas e armazenar o ra nas funções que chamavam outras funções.

t0: Usado como contador nos loops, índice das cavidades na função “distribui_sementes“ e para armazenar o valor lido na entrada do usuário.

t1: Usado como limite dos loops para os lados dos players(6 ou 13) e para armazenar a quantidade de sementes na mão na hora de distribuir as sementes.

t2: Calcular os offsets e acessar as cavidades do tabuleiro.

t3: Armazenar o valor da cavidade escolhida pelo usuário durante o jogo e sua posição na memória.

t4: Armazenar valores lidos da memória, por exemplo o número de sementes em uma cavidade.

t5: Usado para verificar limites de jogada, e calcular os índices opostos das cavidades.

t6: Usados como constantes, foi bastante usado com valor 4 para calcular offsets.

FUNÇÕES IMPLEMENTADAS:

main: A função main inicializa os registradores globais s0, s2 e s3, com os ponteiros e deslocamentos de memória, imprime a mensagem de bem vindo, chama a função que inicializa o tabuleiro e então entra na main_loop.

main_loop: É o loop que roda o jogo, chama mostrar_tabuleiro, verifica_fim e processa_jogada, essas funções são chamadas em sequência até acabar o código.

inicializa_tabuleiro: preenche todas as posições do tabuleiro com o SEEDS_INIT, menos os poços dos jogadores, que são preenchidos com 0, e permite definir o player_atual, setado inicialmente como 1.

mostrar_tabuleiro: imprime o estado atual do tabuleiro durante o jogo, o tabuleiro na parte de cima imprime as cavidades do jogador 2 em ordem decrescente, do índice 12 ao 7, e as cavidades do jogador 1 de ordem crescente, 0 ao 5, e também os poços do jogador 1 e 2, índices 6 e 13 respectivamente.

game_over: Função chamada quando o jogo termina, imprime o tabuleiro no estado final, e compara a quantidade de sementes nos poços para informar quem venceu ou se empatou. Também atualiza o placar com “vitorias_p1 e vitorias_p2”, após isso pergunta se o jogador quer jogar novamente (1 para sim, 0 para exit).

processa_jogada: A partir do player atual, solicita a cavidade que ele quer jogar e faz as validações, se está no intervalo permitido e se tem sementes na cavidade. Caso sim, chama a função “distribui_sementes” e com base no valor retornado em a0, decide se troca de turno ou não.

distribui_sementes: Faz a distrubuição das sementes uma a uma, antes esvazia a cavidade escolhida, caso chegar no poço adversário, pula o poço seguindo com a distribuição. Verifica se a última semente caiu no poço do próprio jogador, caso sim, joga de novo pois retorna 1 em a0. Também verifica se a última semente caiu em uma cavidade vazia do próprio lado, para caso houver sementes na cavidade oposta, fazer a captura das sementes através da label “captura” e armazená-las no próprio poço.

troca_turno e distribui_denovo: São responsáveis por retornar o valor de a0 para a distribui_sementes, para ajustar de quem é a próxima jogada.

verifica_fim_jogo: Verifica se existe algum lado do tabuleiro vazio, somando as sementes dos dois lados, caso algum lado tenha soma = 0, vai para a função “captura_final” e retorna 1 para acabar o jogo, caso contrário retorna.

captura_final: Move as sementes que sobraram do lado que não está vazio para o poço que corresponde a esse lado, e também zera todas as cavidades através das labels “limpa_p1” e “limpa_p2” restando apenas os poços com os valores.

FLUXOGRAMA:

1. **Início:** O programa inicia no rótulo “main” inicializando os registradores globais “s0”(tabuleiro), “s2”(poço do player 1l) e “s3”(poço do player 2).
    a. Imprime a mensagem de bem vindo.

    b. Chama “inicializar_tabuleiro” que preenche os tabuleiros com sementes e os poços com 0.

2. **Main_loop:** O código entra no “main_loop”, chama “mostrar_tabuleiro” que imprime o estado atual do tabuleiro no console.
    a. Chama a função “verifica_fim_jogo”, que decide se o jogo terminou(a0 é igual a 1?).
        a. **Sim:** Pula para “game_over”
        b. **Não:** Continua.

3. Chama “processa_jogada”.
    a. Dentro de “processa_jogada” pede a jogada desejada, valida e chama a função “distribui_sementes”.

    b. Dentro de “distribui_sementes”: distribui as sementes nas cavidades, verifica captura e verifica se o jogador obteve um turno extra.

    c. Dentro de “processa_jogada” atualiza o player atual se não houver turno extra.

    d. Retorna ao inicio de “main_loop”.

4. **Fim de jogo:** Entra no rótulo “game_over”
    a. Chama “mostrar_tabuleiro”, que após a “captura_final” apresenta o tabuleiro final no console.

b. Carrega os valores dos poços, poço p1 em s4 e poço p2 em s5.

c. Compara os valores:
    i. s4 > s5: pula para vitoria_p1
    ii. s5 > s4: pula para vitoria_p2
    iii. s4 == s5: imprime mensagem de empate e pula para "fim_contagem".

d. Rótulo vitoria_p1:
    i. Imprime mensagem de vitória do player 1
    ii. Incrementa 1 em "vitorias_p1"

e. Rótulo vitoria_p2:
    i. Imprime mensagem de vitória do player 2
    ii. Incrementa 1 em "vitorias_p2"

f. Rótulo fim_contagem:
    i. Imprime o placar atual dos players com suas vitórias.
    ii. Imprime a mensagem de jogar novamente(1 para sim 0 para sair).
    iii. Lê a entrada do usuário
g. Decide se vai jogar novamente(se a entrada == 1)
    i. Sim: Pula para "novo_jogo"
        1. Chama "inicializar_tabuleiro".
        2. Pula de volta para "main_loop".
    ii. Não:
        1. Chama ecall com 10 no a7 para encerrar o programa.

h. O programa se encerra.