Universidade Federal da Fronteira Sul
Curso de Ciência da Computação
Organização de Computadores
Professor Luciano L. Caimi

Trabalho de Programação Assembly - RISC-V

Componentes 1 ou 2 alunos

Datas
30/08/2021: Entrega 1 - código fonte da função `insere_embarcacoes`
20/09/2021: Entrega 2 - implementação completa (código fonte) + relatório
01/10/2021: Data limite para a apresentação ao professor

Local de entrega
Somente via Github
Manter projeto privado e convidar o usuário lcaimi para o projeto

Entrega

a) Relatório:
Cada grupo deve produzir um relatório com quatro seções

1. Problema: esta seção deve descrever o problema apresentado.
2. Solução: deve descrever como foi solucionado o problema apontando a lógica do programa em detalhes, as funções utilizadas, os protótipos das funções e o algoritmo das funções utilizadas. Esta seção pode ter o suporte de diversas figuras (devidamente identificadas e referenciadas no texto) incluindo um fluxograma da solução que facilite a compreensão tanto do algoritmo como do código.
3. Conclusões: o grupo deve discutir as dificuldades encontradas no desenvolvimento do programa e o que aprendeu com o trabalho.
4. Programa: deve ser anexada a listagem do programa fonte.

b) Arquivo Fonte:
Deve incluir comentários das instruções ou blocos de código, cabeçalho em cada função descrevendo o que a mesma faz e quais são os parâmetros de entrada e de saída; um cabeçalho no início que identifica o nome e matrícula dos componentes do grupo.

Avaliação

Os trabalhos serão avaliados através de diversos fatores, tais como, atendimento aos requisitos do trabalho, organização do código, documentação do código, utilização da linguagem Assembly, nível algorítmico da solução, explicações e esclarecimentos durante a apresentação.

A avaliação do trabalho levará em consideração os seguintes aspectos:

- codificação e funcionalidades implementadas (60%): modularidade do código, uso de chamadas de função; lógica de implementação; documentação do código; menus; atendimento aos requisitos; nível algorítmico da solução.

- relatório: (15%): atendimento do formato solicitado; capacidade de expressão; clareza da escrita; utilização da língua portuguesa quanto à concordância, pontuação, etc.
- apresentação do trabalho (25% - nota individual): clareza da explicação; demonstração de entendimento do código fonte

Descrição do Problema

O trabalho a ser implementado utilizando o conjunto de instruções do processador RISC-V RV32IM é o controle do jogo de batalha naval em uma matriz 10 x 10. Batalha naval é um jogo para dois jogadores cujo objetivo é afundar os navios (de diferentes tamanhos) do adversário, os quais são dispostos em uma matriz. Cada linha e coluna da matriz é identificada por um número entre 0 e 9.

Na versão a ser implementada teremos apenas um jogador que fará os disparos contra os navios do inimigo, que será a própria “máquina” ou “sistema”. O programa a ser desenvolvido deverá fazer o controle do jogo e a interface com o usuário.

A cada jogada, uma coordenada (linha e coluna) é fornecida para que o programa verifique e diga se acertou algo. O jogo/rodada termina quando o jogador afunda todas as embarcações presentes na matriz.

Os navios do inimigo estão colocados em uma string chamada “navios” presente na área de dados (.data), que deve ser lida pela função `insere_embarcacoes` no início do jogo. A string navios possui o seguinte padrão. Na primeira linha é informado o número de navios inseridos. Cada uma das linhas seguintes possui um navio. As linhas que especificam navios possuem 4 valores, separados por um espaço, sendo: o primeiro valor é a disposição do navio sendo, 0 para navio na horizontal e 1 para navio na vertical; o segundo valor é o comprimento do navio; o terceiro valor é a linha inicial do navio e; o quarto valor é a coluna inicial do navio. Observe o Exemplo:

```
3
1 5 1 1
0 5 2 2
0 1 6 4
```

O posicionamento de navios mostrado acima resulta no seguinte posicionamento:

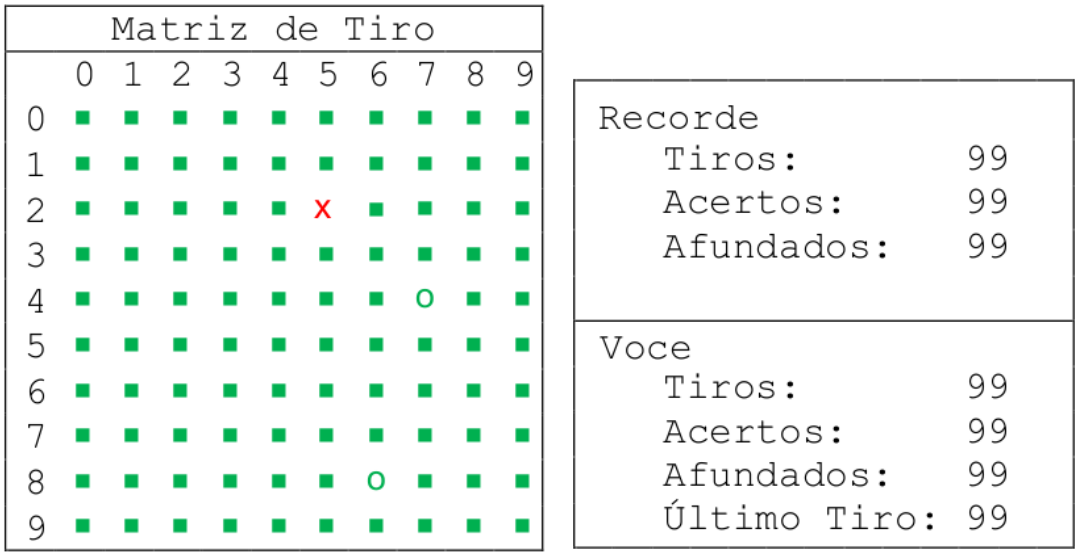
```
   Matriz de Navios
   0 1 2 3 4 5 6 7 8 9
0
1    A
2    A B B B B B
3    A
4    A
5    A
6          C
7
8
9
```

A função insere_embarcacoes deve verificar a validade do posicionamento dos navios, gerando uma mensagem de erro para as seguintes situações:

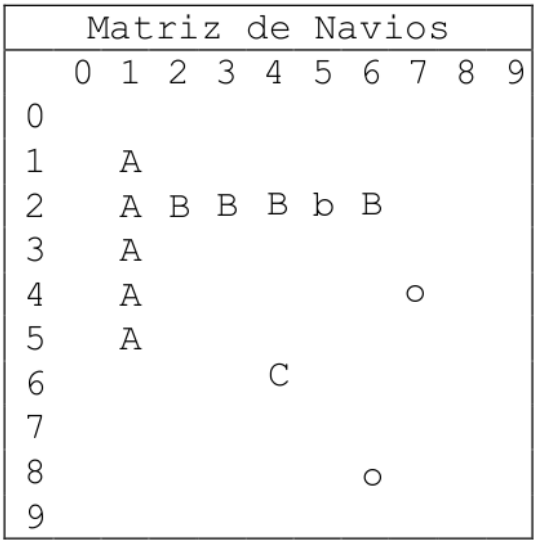
a) A posição do navio é inválida. Exemplo: 0 3 11, 7
b) O navio extrapola as dimensões da matriz. Exemplo: 0 4 9 2
c) Ocorre sobreposição nos navios. Exemplo 0 4 2 2 e 1 3 0 3

Durante a interação do jogo (a cada jogada) deve ser possível: reiniciar o jogo; mostrar o estado atual da matriz de navios; fazer uma nova jogada

A interface de jogo deve apresentar três grupos de informações: 1) a situação atual da matriz de jogo; 2) a quantidade de tiros já disparados, a quantidade de tiros que acertaram o alvo, a quantidade de barcos já afundados e a posição do último tiro disparado e; 3: os recordes do jogo que consiste no menor número de tiros para afundar todas as embarcações de algum jogo anterior. Veja um exemplo:



Internamente a implementação do jogo pode conter duas matrizes, uma matriz de tiro a ser apresentada para o usuário com a posição dos disparos certos e errados e uma matriz de navios que armazena a posição dos barcos e os tiros certos e errados realizados. Observe o exemplo:



Onde “b” representa o tiro no alvo, “o” representa tiro errado, “A”, “B” e “C” representam navios com posição não alvejada.

Funcionalidade opcional: usuário selecionar diferentes strings “navios” (navios1, navios2, navios3) antes do início de cada jogo/rodada.