UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL
INSTITUTO DE INFORMÁTICA - DEPARTAMENTO DE INFORMÁTICA APLICADA

INF01108 - Arquitetura e Organização de Computadores I – semestre 2010/1 - Turmas A e B

Trabalho Prático 2 - Simulador CESAR

Escrever para o Cesar um programa que prepare o "cenário" para o jogo do campo minado e mostre o resultado no visor.

Definições

O programa deverá utilizar, para simular o campo minado, uma matriz de 10 linhas e 10 colunas, que inicialmente deverá conter 0 em todos os seus 100 elementos. Cada elemento da matriz representa uma posição do campo e será identificado por um par de coordenadas, que corresponderão ao número da linha e da coluna em que está cada elemento. As linhas e colunas serão numeradas de 1 a 10, conforme mostrado no desenho a seguir, e não poderão ser colocadas minas na área marcada em cinza (linhas e colunas com números 1 e 10). No desenho abaixo já estão posicionadas 15 minas a título de exemplo, mas a posição delas é definida pelo usuário a cada ciclo (ver especificação).

	1	2	3	4	5	6	7	8	9	10
1	0	0	0	0	0	0	0	0	0	0
2	0	0	-1	0	0	0	0	0	-1	0
3	0	0	0	-1	0	0	0	-1	0	0
4	0	0	0	0	-1	0	-1	0	0	0
5	0	0	0	0	0	-1	0	0	0	0
6	0	0	0	0	0	0	0	0	0	0
7	0	-1	-1	-1	0	0	0	0	0	0
8	0	-1	0	-1	0	0	0	0	0	0
9	0	-1	-1	-1	0	0	0	0	0	0
10	0	0	0	0	0	0	0	0	0	0

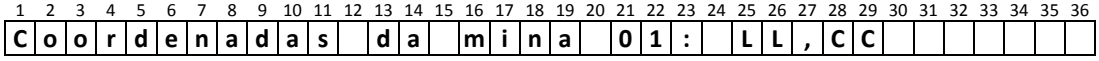
Após a inicialização (lembrar que todas as variáveis usadas no programa devem ser inicializadas quando começar a execução, pois serão feitos testes sucessivos sem recarregar o programa na memória do simulador), o programa deverá executar as seguintes tarefas, na sequência indicada:

- ler do teclado as coordenadas (números da linha e da coluna) das posições onde devem ser colocadas as 15 minas que serão usadas no jogo e colocar o valor numérico "-1" nestas posições
- calcular a quantidade de minas em volta de casa posição do campo minado, preenchendo cada posição com a quantidade correspondente (note que neste trabalho é possível que as minas sejam colocadas nas linhas e colunas 2 e 9; isto vai exigir que a contagem das minas seja feita de uma maneira específica, para evitar dificuldades com o tratamento das bordas do campo)
- exibir no visor o conteúdo final do campo de batalha, linha por linha

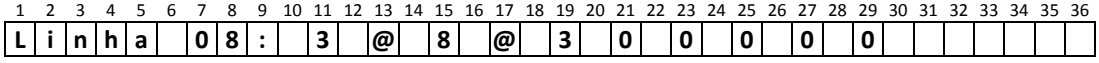
A implementação

Escrever um programa para o CESAR que inicialize o campo minado e exiba o resultado no visor. A inicialização do campo minado inclui:

1. Iniciar o jogo preenchendo o campo com 0 em todas as posições e inicializando todas as variáveis que necessitam de um valor inicial determinado e são alteradas durante a execução.
2. Para cada uma das 15 minas que o jogo vai usar, pedir ao usuário a posição onde deve ser colocada a mina, a qual deverá ser especificada pelo número da linha e da coluna onde fica esta posição. Para cada mina, ao pedir as coordenadas identificar o número da mina, conforme exemplo abaixo (no desenho, os números 1 a 36 acima da linha de LEDs indicam os LEDs do visor). O exemplo é meramente ilustrativo, não sendo obrigatório o uso desta abordagem (ver especificação na próxima página).



3. Verificar se a posição informada é válida. Para isto, não aceitar valores de coordenadas que estejam fora do intervalo de 1 a 10 ou que não sejam numéricos. O programa deve permitir a correção do(s) valor(es) digitado(s) antes de ler o(s) mesmo(s), usando a tecla “backspace”. Se a posição não for válida, avisar o jogador e voltar a pedir as coordenadas daquela mina.
4. Se ela for válida, verificar se é possível colocar uma mina nesta posição. Se a posição já estiver ocupada por uma mina, ou se ficar nas linhas 1 ou 10, ou nas colunas 1 ou 10, informar ao usuário qual o problema e voltar a pedir as coordenadas daquela mina.
5. Se a posição for válida e puder conter uma mina, colocar o valor “-1” nela.
6. Repetir as etapas 2 a 5 para as 15 minas possíveis.
7. Fazer a contagem do número de minas em torno de cada posição do campo, colocando em cada posição a quantidade de minas que estão em torno da mesma. Ao fazer isto, lembrar de que somente as bordas externas do campo não podem conter minas.
8. Depois de terminada a contagem, exibir no visor o conteúdo de cada uma das 10 linhas da matriz (1 linha de cada vez), esperando que o usuário pressione a tecla “Enter” antes de exibir a linha seguinte. Note que cada posição do campo deve ser representada apenas por 1 caractere, para que todas as 10 posições de cada linha caibam no visor de forma legível. Portanto, a representação de uma mina deve ser feita apenas com um caractere, de preferência não numérico. Um exemplo de exibição da linha utilizando o caractere “@” para representar minas seria (use como referência a linha 8 da figura da página anterior):



9. Depois de exibidas as 10 linhas, perguntar se o usuário quer preencher outro campo. Se ele responder que sim, voltar à etapa 1. Se responder que não, terminar o programa com uma mensagem adequada.

Para simplificar o tratamento das bordas durante a etapa 7 (contagem), o computador deve varrer o campo procurando por posições que contenham minas. Quando encontrar uma mina, somar 1 em todas as posições em volta daquela onde está a mina que não contenham outras minas (no máximo 8 posições). O trabalho deve ser desenvolvido usando o montador Daedalus.
Os trabalhos serão corrigidos manualmente. Devem ser observadas rigorosamente as seguintes especificações:

- O código do programa deve iniciar no endereço 0 da memória.
- A primeira instrução executável deve estar no endereço 0.
- O programa será executado em ciclos (ou rodadas).
- Ao iniciar cada ciclo, o programa deverá zerar o campo, inicializar variáveis e recomeçar a pedir coordenadas das 15 minas, a partir da primeira mina.
- Não há garantia que os dados digitados para as coordenadas (coluna e linha) estão corretos, o programa deve verificar a consistência desses dados.

Especificação

O programa deve iniciar identificando o autor, listando no visor o seu nome e o seu número de identificação UFRGS. Caso sejam utilizadas várias linhas, deve-se aguardar que seja digitada a tecla *Enter* (código ASCII 13) após cada linha, antes de passar para a próxima.

Após esta identificação, escrever nas posições mais à esquerda do visor uma frase indicando o número da mina e pedindo a linha, terminando por "> " ou "**:** ", e aguardar a digitação do número da linha (1 a 10), permitindo correção do valor digitado. Depois, escrever uma frase pedindo a coluna, também terminando por "> " ou "**:** ", e aguardar a digitação do número da coluna (1 a 10), permitindo correção do valor digitado. O formato mostrado no exemplo da etapa 2, no qual as duas coordenadas são pedidas simultaneamente, é mais difícil de implementar, mas também poderá ser usado alternativamente. ;-)

Cada caractere digitado deve ser ecoado no visor. Além disso, deve ser apresentado (e controlado) um cursor, representado pelo caractere *Undescore* ("_", código ASCII 95). Também deve ser permitido o uso da tecla *Backspace* (código ASCII 8) para corrigir erros de digitação (cuidado com o tratamento do *Backspace* **antes** do primeiro caractere ser digitado!). Na leitura das coordenadas, o programa deve considerar que os dados terminam se for digitada a tecla *Enter* ou for atingido o final do visor.

Quando terminar a digitação das coordenadas de uma posição, o programa deve determinar se a posição é válida (linha de 1 e 10 e coluna de 1 e 10) e se está desocupada (sem mina). Ocorrendo erro, deve ser escrita uma mensagem para o jogador informando qual o erro e o programa deve solicitar novamente as coordenadas para a mesma mina.

Se coluna e linha forem válidas, o programa preenche aquela posição do campo com o valor -1, indicando que ali está uma das 15 minas.

Depois de posicionadas as 15 minas, o programa deve fazer a contagem delas e logo após iniciar a exibição da situação final do campo, linha por linha, conforme exemplo mostrado na etapa 8 da página anterior. O ciclo termina após a exibição da décima linha do campo.

Terminado um ciclo, o programa deve perguntar se o jogador deseja preencher outro campo, com uma frase do tipo "Preencher outro campo?". Se for digitado "s" ou "S", o programa deve iniciar um novo ciclo. Caso contrário, o programa deve terminar (instrução Halt).

Não esqueça que os caracteres digitados pelo usuário são recebidos em ASCII e podem ser enviados para o visor neste formato. No entanto, para endereçar as posições do campo minado é necessário que as coordenadas da coluna e da linha sejam convertidas para valores numéricos.

O trabalho deverá ser feito de forma individual e entregue via Moodle, contendo um arquivo fonte comentado (.ced ou .txt) e um executável (.mem). Para nomear os arquivos, utilize a inicial de seu primeiro nome, seguida de seu número de cartão de identificação da UFRGS. Por exemplo, o aluno **Luis Inácio** deverá nomear seus arquivos **Lnnnnnn**.ced e **Lnnnnnn**.mem, onde **nnnnnn** é o número de seu cartão. Os arquivos do trabalho deverão ser compactados em um arquivo **Lnnnnnn**.rar (ou **Lnnnnnn**.zip) e entregues através do Moodle.

Os testes serão realizados com a opção "Atualizar Registradores" desligada (ver Menu "Executar" no simulador). Leve isto em conta caso o seu programa utilize rotinas que dependam de tempo (como um cursor piscante, por exemplo).

Dia e hora limites para entrega: 09/05/2010, 23h59, via Moodle