Segundo EP
Máquina Virtual - Entrega dia 23/11

4 de novembro de 2009

Neste exercício programa você vai implementar um processador virtual simples, capaz de executar programas em "linguagem de máquina".

Um programa deste processador consiste em uma sequência de números (*bytes*) e uma coleção de 256 registradores capazes de armazenar um valor *double* cada um.

Além disso, o processador possui dois registradores especiais: o ponteiro de instruções (**PI**) e um armazenador de estado (**E**), discutidos abaixo.

Os registradores são representados simplesmente por um vetor de *doubles*:

```
double []reg = new double[256];
```

e são referenciados por seu índice no vetor, veja os exemplos na seção 2.

De modo similar, as instruções ficam em um outro vetor de *bytes*:

```
byte []prog = new byte[MAXMEM];
```

`MAXMEM` é uma constante com o tamanho máximo do programa, use pelo menos 1024.

O registrador **PI** indica a próxima instrução a ser executada. Cada instrução corresponde a um código de operação (somar, ler, imprimir, desvio, etc) e pode conter alguns operandos.

A entrada é a lista das instruções, terminadas pelo código especial 1000, seguida por uma segunda lista, de pares de valores:

1. Um inteiro indicando o índice da variável
2. Um *double* indicando o valor correspondente no início do programa

A segunda lista termina com o valor -1.

1 Instruções

Os códigos de instruções estão listados a seguir, separados por classes.

Aritméticas As instruções aritméticas começam em 10 e possuem 3 argumentos: os índices dos dois operandos e onde será guardado o resultado.

Estes são os códigos:

10 soma
11 subtração
12 multiplicação
13 divisão

Por exemplo, a sequência:

12 5 1 42

significa que o conteúdo do registrador 5 será multiplicado pelo conteúdo do registrador 1 e o resultado será armazenado no registrador 42. Em **Java** ficaria assim:

```
reg[42] = reg[5] * reg[1];
```

Fácil, não?

Lógicos As operações lógicas (comparação, negação, etc, seguem uma forma similar, mas o resultado é armazenado no resgistrador especial **E**. O valor 0 (zero) corresponde a falso, qualquer outro significa verdadeiro, como na linguagem **C**.

Os códigos começam em 20:

20 maior
21 menor
22 maior ou igual
23 menor ou igual
24 igual
25 diferente
26 negação (apenas um operando)
27 compara com 0 (apenas um operando)

Movimentação Começam em 30 e possuem os seguintes significados:

30 *orig dest*	copia o valor da *orig* para *dest*
31 r_1 r_2	troca o valor de r_1 com o de r_2
32 *reg*	copia o valor de **E** em *reg*
33 *reg*	copia o valor de *reg* em **E**

Desvios Os desvios são especiais, eles alteram especificamente o valor do registrador **PI**, somando o argumento. São usados para *if* e *laços*.

Começam em 40:

40 *delta*	soma *delta* ao valor de **PI** (`PI += prog[PI];`)
41 *delta*	soma *delta* ao valor de **PI**, se $\mathbf{E} \neq 0$
42 *delta*	soma *delta* ao valor de **PI**, se $\mathbf{E} = 0$

Miscelânea Em 50 começam algumas instruções especiais:

50 *dest*	imprime na saída o conteúdo da variável em *dest*
51 *dest*	lê um valor[1] da entrada e coloca na variável *dest*
52 r	incrementa a variável em r
53 r	decrementa a variável em r
-1	término do programa

Matemáticos A partir da posição 100 ficam as funções matemáticas:

100 *orig dest*	*dest* recebe a raiz quadrada de *orig*
101 *orig dest*	*dest* recebe seno de *orig*
102	idem, cosseno
103	tangente
etc	

2 Exemplo

Um exemplo de entrada. Siga os códigos e tente entender.

2.1 Fibonacci

Lê um número n e imprime os n primeiros elementos da série de Fibonacci, a partir do terceiro. Os dois primeiros estão inicialmente colocados nas variáveis 1 e 2.

```
51 0 27 0 42 18 10 2 3 1 30 2 3 30 1 2 50 1 53 0 40 -18 -1 1000
1 1
2 1
-1
```

Explicando melhor:

0	51 0	lê n em $reg[0]$
2	27 0	$n = 0$?
4	42 18	se for, vai para 22
6	10 2 3 1	$reg[1] = reg[2] + reg[3]$
10	30 2 3	$reg[3] = reg[2]$
13	30 1 2	$reg[2] = reg[1]$
16	50 1	imprime $reg[1]$
18	53 0	$reg[0] - -$
20	40 -18	vai para 2
22	-1	fim do programa

3 Bônus especial

Ponto adicional para quem fizer a impressão do programa em código de máquina formatada, com uma instrução por linha e trocando o código da instrução por um nome adequado. Por exemplo, o programa acima ficaria:

```
 0: Leia      0
 2: Zero?     0
 4: Sim->    22
 6: Soma      2   3 ->  1
10: Copia     2   3
13: Copia     1   2
16: Imprime   1
18: Dec       0
20: Desvia    2
22: Fim
```

3.1 Sobre a avaliação:

- É sua responsabilidade manter o código do seu EP em sigilo, ou seja, apenas você e seu par devem ter acesso ao código.

- No caso de exercícios feitos em dupla, a mesma nota da correção será atribuída aos dois alunos do grupo;
- Não serão toleradas cópias! Exercícios copiados (com ou sem eventuais disfarces) receberão nota ZERO (inclusive o original);
- Exercícios atrasados não serão aceitos;
- Exercícios com erros de sintaxe (ou seja, erros de compilação) receberão nota ZERO;
- É muito importante que seu programa tenha comentários e esteja bem indentado, ou seja, digitado de maneira a ressaltar a estrutura de subordinação dos comandos do programa (conforme visto em aula). A qualidade do seu trabalho sob esse ponto de vista influenciará sua nota!
- As informações impressas pelo seu programa na tela devem aparecer da forma mais clara possível. Este aspecto também será levado em consideração no cálculo da sua nota;
- Uma regra básica é a seguinte: do ponto de vista do monitor responsável pela correção dos trabalhos, quanto mais convenientemente apresentado estiver o seu programa, melhor será a disposição dele para dar-lhe uma nota generosa.

importante: Para EPs feitos individualmente o bônus pode valer nota a mais. No caso de duplas, o bônus é obrigatório.

3.2 Sobre a entrega:

- O prazo de entrega é o dia 23/11/2009;
- Deverão ser entregues um arquivo por problema dentro de um arquivo zip nomeado EP1-<número USP>. No caso de EPs feitos por programação em pares, basta entregar uma versão, incluindo ambos números USP no nome do arquivo (EP1-<número USP 1>e<número USP 2>).
- No início do arquivo, acrescente um cabeçalho bem informativo, como o seguinte:

```
/*****************************************************************/
/**   MAC 115 - Introdução à Computação                         **/
/**   IF-USP - Segundo Semestre de 2009                         **/
/**   <turma> - <nome do professor>                             **/
/**                                                             **/
/**   Segundo  Exercício-Programa  --  Problema                 **/
/**   Arquivo:                                                  **/
/**                                                             **/
/**   <nome do(a) aluno(a)>                <número USP>         **/
/**   <nome do(a) aluno(a)>  (se houver)   <número USP>         **/
/**                                                             **/
/**   <data de entrega>                                         **/
/*****************************************************************/
```

Não é obrigatório que o cabeçalho seja idêntico a esse, apenas que contenha pelo menos as mesmas informações.

- Para a entrega, utilize o Paca. Você pode entregar várias versões de um mesmo EP até o prazo, mas somente a última será armazenada pelo sistema.
- Guarde uma cópia do seu EP pelo menos até o fim do semestre!