Gabarito da AD1 de Programação II

1º Semestre de 2009

1. (2 pontos) Considere o seguinte algoritmo

```
entrada N
enquanto N > 1
   imprime N
   se N é impar
      N = 3*N+1
   senao
      N = N/2
   fim se
fim enquanto
```

Pede-se codificar este algoritmo em PHP sob a forma de uma função cujo argumento de entrada é **N.** Por exemplo, se **N**=22, a função deve imprimir a seguinte lista de números: 22, 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2

Resp.

```
<?php

function geranumeros($n){
    while ($n>1){
        echo $n.",";
        if ($n%2 != 0)
            $n = 3*$n + 1;
        else
            $n = $n/2;

    }
}

geranumeros(22);

?>
```

2. (4 pontos) Dado um *array* bidimensional (matriz) *M* de números inteiros, escreva uma função que imprima a sub-matriz cuja soma de seus elementos é a maior entre todas as sub-matrizes de M. Note que, se *M* tem dimensões $n \times m$, pode-se construir sub-matrizes

de tamanhos $n \times (m-1)$, $n \times (m-2)$, ... , $n \times 1$, $(n-1) \times m$, $(n-1) \times (m-1)$, $(n-1) \times (m-2)$, ... , $(n-1) \times 1$, ... , 1×1. Por exemplo, dada a matriz

0	-2	-7	0
9	2	-6	2
-4	1	-4	1
-1	8	0	-2

A sub-matriz cuja soma de seus elementos é a maior entre todas as sub-matrizes é

9	2
-4	1
-1	8

Resp.

```
<?php

function submatriz($M, $ini_i, $ini_j, $fin_i, $fin_j, &$sum){
    $sub_M = array();
    for($j = $ini_j; $j<$fin_j; $j++){
        $row = array();
        for($i = $ini_i; $i<$fin_i; $i++){
            $sum += $M[$j][$i];
            $row[] = $M[$j][$i];
        }
        $sub_M[] = $row;
    }

    return $sub_M;
}

function encontramaiormatriz($M){
    $m = count($M);
    $n = count($M[0]);

    $max = 0;
    $result = array();
```

```
        for($j = 0; $j<$m; $j++){
            for($i = 0; $i<$n; $i++){
                for($J = $j+1; $J<=$m; $J++){
                    for($I = $i+1; $I<=$n; $I++){
                        $curr = 0;
                        $sub_M = submatriz($M, $i, $j, $I, $J, $curr);
                        if ($curr>$max){
                            $max = $curr;
                            $result = $sub_M;
                        }
                    }
                }
            }
        }

        return $result;

    }

    $M = array( array(0, -2, -7, 0),
                array(9, 2, -6, 2),
                array(-4, 1, -4, 1),
                array(-1, 8, 0, -2));

    $sub_M = encontramaiormatriz($M);

    echo "<pre>";
    print_r($sub_M);
    echo "</pre>";

    ?>
```

3. Considere as seguintes tabelas, **estoque, movimentacao e itens** respectivamente. Elas são o banco de dados de um sistema de estoque que permite operações de compra e venda de artigos.
   Note que o relacionamento entre as tabelas é dado por `id_estoque` e `id_mov`.

estoque

id_estoque	produto	valor_unit	quantidade
1	Camisa Polo X	10.00	12
2	Calça M	35.90	5
4	Casaco P	59.50	9

movimentacao

id_mov	data	tipo
4	10-10-2008	compra
5	12-11-2008	venda
6	01-02-2009	compra
7	02-02-2009	venda

itens

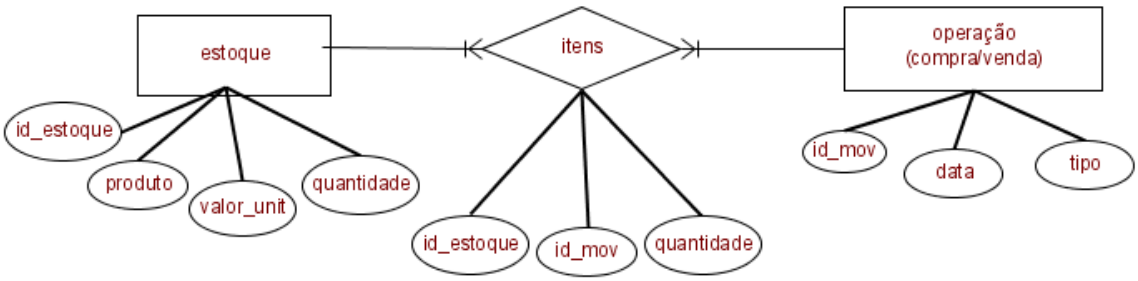
id_estoque	id_mov	quantidade
1	4	15
1	5	3
4	6	10
2	6	5
4	7	1

Pede-se:

1. (1 ponto) Criar um modelo de Entidade Relacionamento apropriado.
2. (1 ponto) Escrever o modelo físico correspondente.
3. (2 pontos) Escrever as consultas que você usaria para registrar as duas compras.

Resp:

1.



2.

```
CREATE TABLE `estoque` (
`id_estoque` int(10) unsigned NOT NULL AUTO_INCREMENT,
`produto` varchar(200) NOT NULL,
`valor_unit` decimal(10,2) unsigned NOT NULL DEFAULT '0.00',
`quantidade` tinyint(3) unsigned DEFAULT NULL,
PRIMARY KEY (`id_estoque`)
);
```

```
CREATE TABLE `movimentacao` (
`id_mov` int(10) unsigned NOT NULL AUTO_INCREMENT,
`data` date NOT NULL,
`tipo` varchar(50) NOT NULL,
PRIMARY KEY (`id_mov`)
);


CREATE TABLE `itens` (
`id_estoque` int(10) unsigned NOT NULL,
`id_mov` int(10) unsigned NOT NULL,
`quantidade` int NOT NULL,
PRIMARY KEY (`id_estoque`, `id_mov`)
);
```

3.

Compra 1

```
insert into movimentacao values (4, '2008-10-10', 'compra');
insert into itens values (1, 4, 15);
```

Se o produto não esta cadastrado use insert

```
insert into estoque (produto, valor_unit, quantidade) values
('Camisa Polo X', 10.00, 15);
```

Caso o produto já esteja cadastrado use update

```
update estoque set estoque.quantidade = estoque.quantidade + 15 where
id_estoque = 1;
```

Compra 2

```
insert into movimentacao values (6, '2009-01-02', 'compra');
insert into itens values (4, 6, 10);
```

Se o produto não esta cadastrado use insert

```
insert into estoque (produto, valor_unit, quantidade) values
('Casaco P', 59.50, 10);
```

Caso o produto já esteja cadastrado use update

```
update estoque set estoque.quantidade = estoque.quantidade + 10 where
id_estoque = 4;
```