Q1	3,0	
Q2	2,0	
Q3	4,0	
Q4	1,0	

Fundação CECIERJ – Vice Presidência de Educação Superior à Distância
Curso de Tecnologia em Sistemas de Computação
Disciplina: Programação de Aplicações Web
Professores: Diego Passos e Uéverton dos Santos Souza
Gabarito da AP3 – 2º Semestre de 2018
Nome:_____

Questão 1: Considere a seguinte função em PHP:

```
function parse($string) {
  $estado = 0;
  for ($i = 0; $i < strlen($string); $i++) {
    switch($estado) {
      case 0:
      if ($string[$i] >= "0" && $string[$i] <= "9") {
        $estado = 1;
        break ;
      }
      else return(false);
      case 1:
      if ($string[$i] >= "0" && $string[$i] <= "9") {
        $estado = 1;
        break ;
      }
      else if ($string[$i]=="x" || $string[$i]=="+" || $string[$i]=="-" || $string[$i]=="/")
{
        $estado = 2;
        break ;
      }
      else return(false);
      case 2:
      if ($string[$i] >= "0" && $string[$i] <= "9") {
        $estado = 3;
        break ;
```

```
      }
      else return(false);
      case 3:
      if ($string[$i] >= "0" && $string[$i] <= "9") {
        $estado = 3;
        break ;
      }
      else if ($string[$i]=="x" || $string[$i]=="+" || $string[$i]=="-" || $string[$i]=="/")
{
        $estado = 2;
        break ;
      }
      else return(false);
    }
  }
  if ($estado == 3) return(true);
  else return(false);
}
```

Essa função recebe como argumento uma *string* e verifica se ela está de acordo com um determinado formato. Se sim, a função retorna `true`. Caso contrário, retorna `false`. Sobre essa função, pede-se:

a) **(0,5 pontos)** Qual é o valor retornado pela função quando o argumento passado é a string "3x4+11" (**sem as aspas**)?

Resposta: executando o código linha a linha, conclui-se que a repetição termina com a variável `$estado` com valor 3, fazendo com que a condição do último `if` seja verdadeira. Assim, a função retorna `true`.

b) **(0,5 pontos)** Qual é o valor retornado pela função quando o argumento passado é a string "x7+1" (**sem as aspas**)?

Resposta: logo na primeira iteração da repetição principal da função, esta retorna o valor `false`.

c) **(0,5 pontos)** Qual é o valor retornado pela função quando o argumento passado é a string "44" (**sem as aspas**)?

Resposta: nesse caso, a execução da repetição termina com a variável `$estado` contendo o valor 1, o que torna a condição testada no último `if` do código falsa, fazendo com que a função retorne `false`.

d) **(0,5 pontos)** Qual é o valor retornado pela função quando o argumento passado é a string "37/" (**sem as aspas**)?

Resposta: nesse caso, a execução da repetição termina com a variável `$estado` contendo o valor 2, o que torna a condição testada no último `if` do código falsa, fazendo com que a função retorne `false`.

e) **(1,0 ponto)** Escreva uma expressão regular em PHP que seja equivalente a essa função. Em outras palavras, sua expressão regular deverá casar **com todas** as *strings* para as quais a função retorna `true` e **apenas essas**.

Resposta: a função basicamente reconhece expressões numéricas envolvendo números decimais inteiros e as quatro operações aritméticas básicas. Nenhum outro caractere é aceito. A expressão deve conter ao menos uma operação e deve ser iniciada e terminada por números. Uma possível solução, portanto, é a expressão regular:

```
^[0-9]+([+x/-][0-9]+)+$
```

A sequência `^[0-9]+` determina que a string deve começar por um número com um ou mais dígitos. Já a sequência `[+x/-][0-9]+` determina que deve haver exatamente um dos quatro operadores aritméticos seguido de um novo número com um ou mais dígitos. Essa sequência é colocada entre parênteses, denotando um grupo, enquanto o último caractere + permite que a sequência seja repetida uma ou mais vezes. Finalmente, o caractere $ finaliza a string.

Questão 2: Escreva uma função em PHP que receba como parâmetro uma matriz em que cada posição armazena um caractere e retorne aquele caractere que se repete mais vezes.

Observações: note que a matriz pode não ser quadrada (isto é, ela pode possuir números diferentes de linhas e colunas). Além disso, considere que, em caso de empate, sua função pode retornar qualquer dos caracteres que mais se repetem.

Resposta: uma possível solução é dada pela seguinte função:

```
function elementoMaisComum($m) {

    // Inicializa uma variável para guardar
    // quantas vezes o elemento mais comum até
    // agora já ocorreu.
    $ocorrencias = 0;

    // Conta o número de linhas na matriz.
    $nLinhas =    count($m);

    // Itera por cada linha
    for ($i = 0; $i < $nLinhas; $i++) {

       // Verifica o número de colunas
       $nColunas = count($m[$i]);

       // Itera pelos elementos da linha
       for ($j = 0; $j < $nColunas; $j++) {
```

```
            // Utiliza um vetor associativo para
            // contar quantas vezes certo elemento
            // ocorreu. Para isso, primeiro testamos
            // se a posição relativa ao elemento já
            // existe no vetor contador.
            if (isset($contador[$m[$i][$j]])) {

                  // Sim, basta incrementar.
                  $contador[$m[$i][$j]]++;
            }
            else {

                  // Não, inicializamos a posição.
                  $contador[$m[$i][$j]] = 1;
            }

            // Verifica se esse elemento é agora
            // o mais comum.
            if ($contador[$m[$i][$j]] > $ocorrencias) {

                  // Sim, atualizar.
                  $ocorrencias = $contador[$m[$i][$j]];
                  $maisComum = $m[$i][$j];
            }
      }
   }

   return($maisComum);
}
```

Em particular, nessa solução usamos a funcionalidade de vetores associativos do PHP, que permite que um vetor seja indexado por um caractere, por exemplo. Isso facilita o processo de contagem dos elementos repetidos.

Questão 3: Uma companhia mantém uma lista de preços de produtos armazenada em uma tabela chamada PRODUTO dentro de um banco de dados chamado BANCO e que contém as seguintes colunas: CODIGO (um número inteiro), NOME (uma string) e PRECO (um número decimal). Escreva em PHP uma mini-aplicação consistindo de dois módulos:

1. (2 pontos) O programa consulta.php mostra um formulário com apenas um campo, correspondente ao código de um produto. Uma vez que o usuário preenche o formulário e o submete, o programa carrega o programa resposta.php.

Resposta:

A solução mínima para consulta.php:

```
<form action="resposta.php" method="POST">      <!-- ou GET -->
        <input name="codigo" />
        <input type="submit" />
</form>
```

2. (2 pontos) O programa resposta.php consulta o banco de dados pelo produto correspondente ao código e, caso o encontre, exibe seu nome e preço. Caso o produto não se encontre no banco, exibe a mensagem "não encontrado".
Resposta: Uma solução aproximada para resposta.php:

```
<?php

$conn = mysql_connect("localhost","root","");
mysql_select_db("banco", $conn);

$sql = 'SELECT * FROM PRODUTO WHERE CODIGO = '
.$_REQUEST['codigo'];

$result = mysql_query($sql);

if(mysql_num_rows($result) == 0) echo "não encontrado";

//Deverá fazer apenas uma iteração, mas não há problema
while($row = mysql_fetch_array($result)){
        echo "Nome do produto: " . $row[0] . "<br/>";
        echo "Preço do produto: " . $row[1] . "<br/>";
}
?>
```

Questão 4: Que diferenças existem entre envio de dados por GET e por POST? Descreva sobre quando é preferível usar um ou outro e como as informações são enviadas por um e por outro.
Resposta:

A maior diferença entre os métodos GET e POST é a visibilidade: uma requisição GET é enviada como string acrescentado seu conteúdo a URL, enquanto que para POST é encapsulada junto ao corpo da requisição HTTP e não pode ser vista.

Outra diferença é o tamanho permitido a mensagem enviada. Já que com GET a mensagem é enviada via URL, seu tamanho é limitado, enquanto que usando POST não há limitações de comprimento, já que a mesma é enviada no corpo da requisição HTTP.

Sobre a escolha entre os dois, é preferível usar GET quando a mensagem é pequena e pode ser revelada por ser relativamente mais rápida. Por outro lado, é preferível usar post quando queremos preservar oculto seu conteúdo, quando a mensagem contém outras informações que não textos (GET só aceita textos e limitados ao padrão ASCII), ou ainda quando for grande.