Q1	**3,0**	
Q2	**2,0**	
Q3	**3,0**	
Q4	**2,0**	

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 – 1º Semestre de 2018

Questão 1: Considere a seguinte função em PHP:

```php
function teste($a) {
        $s = ereg("^[0-9]+$", $a);
        if ($s == FALSE) {
                echo "Valor '$a' eh invalido!";
                return ;
        }
        for ($i = strlen($a) - 1; $i >= 0; $i--) {
                $continua = 0;
                switch($a[$i]) {
                        case "0":
                                $a[$i] = "1";
                                break ;
                        case "1":
                                $a[$i] = "2";
                                break ;
                        case "2":
                                $a[$i] = "3";
                                break ;
                        case "3":
                                $a[$i] = "4";
                                break ;
                        case "4":
                                $a[$i] = "5";
                                break ;
                        case "5":
                                $a[$i] = "6";
                                break ;
                        case "6":
                                $a[$i] = "7";
                                break ;
                        case "7":
                                $a[$i] = "8";
                                break ;
                        case "8":
                                $a[$i] = "9";
                                break ;
                        case "9":
                                $a[$i] = "0";
                                $continua = 1;
                                break ;
                }
                if ($continua == 0) break ;
        }
        if ($continua == 1) return("1" . $a);
        else return($a);
}
```

Com base no código apresentado, pede-se:

a) **(0,5 pontos)** Determine a saída da função quando a entrada é a *string* “25999”.
b) **(1,0 ponto)** Descreva em uma frase o que o código faz (*i.e.*, qual é o seu objetivo).
c) **(0,5 pontos)** Explique qual é o propósito das 4 primeiras linhas de código da função (chamada à função `ereg()` e `if` subsequente).
d) **(1,0 ponto)** Reescreva a função de uma forma mais simples. Em particular, sua versão da função **não deve conter repetições**.

Respostas:

a) Realizando a execução do código linha a linha, conclui-se que a saída é a *string* “26000”.
b) O código recebe como entrada uma string representando um número inteiro em base 10 e calcula o valor do incremento desse número manipulando a *string*.
c) Esse trecho realiza uma validação do formato da entrada, garantindo que a *string* represente um valor inteiro decimal válido.
d) Uma possível solução é a conversão prévia do parâmetro de *string* para inteiro, o que tornaria o processo de incremento uma simples soma com 1. O seguinte trecho ilustra essa solução, utilizando a funcionalidade de PHP de automaticamente converter o tipo de uma variável com base na operação realizada (também é possível o uso de funções explícitas para essa conversão como a **intval**()):

```
function teste($a) {

        $a = $a + 1;
        return(strval($a));
}
```

Questão 2: Escreva uma função PHP que receba dois parâmetros: `vetorPalavras` e `letras`. O primeiro parâmetro, `vetorPalavras`, é um vetor contendo palavras escritas apenas com letras minúsculas. O segundo, *letras*, é uma *string* contendo, também, apenas letras minúsculas **não repetidas**. Sua função deverá retornar todas as palavras em `vetorPalavras` que são **anagramas** da *string* `letras`.

Observação: nesse contexto, considere um anagrama como sendo uma *string* formada da reorganização dos caracteres de outra *string*. Em outras palavras, o anagrama deverá conter **todos os caracteres da string original (e apenas esses)**

exatamente uma vez, mas em uma ordem distinta. Exemplo: “qwerty” é um anagrama de “rytwqe”.

Resposta: Uma possível solução é dada a seguir:

```
function anagramas($vetorPalavras, $letras) {

        // Remover palavras do vetor que nao sejam do
        // mesmo comprimento de $letras.
        $ultima = sizeof($vetorPalavras) - 1;
        for ($i = 0; $i <= $ultima; $i++) {

                if (strlen($vetorPalavras[$i]) != strlen($letras)) {

                        $temp = $vetorPalavras[$ultima];
                        $vetorPalavras[$ultima] = $vetorPalavras[$i];
                        $vetorPalavras[$i] = $temp;
                        $ultima--;
                        $i--;
                }
        }

        // Para cada letra em $letra, verificar se esta se
        // encontra em cada palavra em $vetorPalavras.
        for ($i = 0; $i < strlen($letras); $i++) {

                for ($j = 0; $j <= $ultima; $j++) {

                                if (strstr($vetorPalavras[$j], $letras[$i]) ==
FALSE){

                                  $temp = $vetorPalavras[$ultima];
                                  $vetorPalavras[$ultima] = $vetorPalavras[$j];
                                  $vetorPalavras[$j] = $temp;
                                  $ultima--;
                                  $j--;
                        }
                }
        }

        // Montar vetor a ser retornado.
        $i = 0;
```

```
        $saida = array();
        for ($j = 0; $j <= $ultima; $j++) {

                if ($vetorPalavras[$j] != $letras) {
                        $saida[$i] = $vetorPalavras[$j];
                        $i++;
                }
        }

        return($saida);
}
```

Questão 3: Suponha que você está implementando um sistema para controle de vendas num restaurante. O restaurante possui um número determinado de mesas, uma lista de pratos que pode servir, e uma lista de bebidas disponíveis. Cada bebida, assim como cada prato, tem seu nome e seu preço. Quando uma mesa paga a sua conta, o sistema registra que todos os pedidos da mesa foram pagos. Cada pedido é marcado com a data e hora em que foi feito. O objetivo do sistema é de cadastrar pedidos para poder consultar consumo de cada mesa e o faturamento do restaurante.

a) **(1,0 ponto)** Crie a modelagem física MySQL do banco de dados.

Resposta:

```
CREATE TABLE `mesa` (
`id_mesa` INT(10) NOT NULL AUTO_INCREMENT,
`descricao` VARCHAR(255) NULL DEFAULT '',
PRIMARY KEY (`id_mesa`)
)

CREATE TABLE `produto` (
`id_produto` TINYINT(10) NOT NULL AUTO_INCREMENT,
`nome` VARCHAR(255) NULL DEFAULT '',
`tipo` VARCHAR(50) NULL DEFAULT '',
`valor` DECIMAL(10,2) NULL DEFAULT NULL,
PRIMARY KEY (`id_produto`)
)

CREATE TABLE `pedidos` (
`id_mesa` INT(10) NULL DEFAULT NULL,
`id_produto` INT(10) NULL DEFAULT NULL,
`data` DATE NULL DEFAULT NULL,
`hora` TIME NULL DEFAULT NULL,
`estado` VARCHAR(10) NULL DEFAULT NULL
)
```

b) **(1,0 ponto)** Escreva uma consulta em MySQL para determinar o total não pago da mesa 4.

Resposta:

```
select sum(prod.valor) from mesa as m, pedidos as ped, produto as prod
where
ped.id_mesa = m.id_mesa and
ped.id_produto = prod.id_produto and
ped.estado = "nao pago"
```

c) **(1,0 ponto)** Escreva uma consulta em MySQL para saber o faturamento do dia 13/04/18.

Resposta:

```
select sum(prod.valor) from pedidos as ped, produto as prod
where
ped.id_produto = prod.id_produto and
ped.data = "2018-04-13" and
ped.estado = "pago"
```

Questão 4: Sobre o funcionamento de sessões: Por que quando colocamos os dados na sessão, eles não aparecem ao tentarmos utilizar outro navegador?

Resposta:

Para o PHP saber qual usuário e o dono de uma sessão, ele guarda algumas informações nos Cookies do navegador. Na verdade, a informação mais importante fica em um cookie chamado **PHPSESSID**, que guarda um código único de identificação da sessão daquele usuário que está acessando a aplicação PHP.