Qa		**1,0**
Qb		**2,0**
Qc		**3,0**
Qd		**4,0**

Fundação CECIERJ – Vice Presidência de Educação Superior à Distância
Curso de Tecnologia em Sistemas de Computação
Disciplina: Programação de Aplicações Web
Professores: Cristina Nader Vasconcelos e Daniel de Oliveira
Gabarito da AD1 – 2º Semestre de 2017

Estão sendo avaliados o uso das funções date e time do PHP, a elaboração de funções e o uso de vetores.

Suponha que você foi contratado para escrever funções em PHP para dar suporte a um site de cálculo de aposentadoria.

a) **(1.0 pto)** Escreva uma função chamada CALCULO_DE_TEMPO_CONTINUO que recebe o dia, mês e ano de início de trabalho na forma de três parâmetros de entrada e calcula e **retorna quantos anos completos** o trabalhador tem de serviço. Conside que ele esteve empregado sem interrupções desde essa data e que na data atual continua neste trabalho. Você pode usar a função date do PHP para descobrir a data atual.

```
function CALCULO_DE_TEMPO_CONTINUO($dia, $mes, $ano) {

    return (date('Y') - $ano) -1*(date('m') < $mes) -1*(date('m') == $mes && date('d') < $dia);

}
```

```
echo CALCULO_DE_TEMPO_CONTINUO(31,8, 2016);

echo CALCULO_DE_TEMPO_CONTINUO(1,12, 2016);

echo CALCULO_DE_TEMPO_CONTINUO(31,1, 2016);

echo CALCULO_DE_TEMPO_CONTINUO(1,1, 2016);
```

b) **(2.0 pto)** Escreva uma função chamada REGRA_DE_APOSENTADORIA que além dos três parâmetros da data de contratação recebe também o dia, mês e ano de nascimento do trabalhador. Sua função calcula o tempo restante para aposentadoria usando duas regras:
   - supondo serem necessários 45 anos de contribuição para aposentadoria regular;
   - supondo idade mínima de 65 anos e que o trabalhador tenha realizado pelo menos 180 meses de contribuição na data correspondente para a aposentadoria por idade mínima;

Sua função deve escrever na tela a regra que será primeiro alcançada pelo trabalhador.

```
# retorna verdadeiro se data  A < B
function DATA_ANTERIOR($diaA, $mesA, $anoA, $diaB, $mesB, $anoB) {

 if ($anoA < $anoB)
    return true;
 if (($anoA == $anoB) and (($mesA < $mesB) or ($mesA == $mesB and $diaA<
$diaB)))
    return true;
 return false;

}

echo  DATA_ANTERIOR(1, 1,1, 1, 1, 2000) ;
echo "<br>";
echo  DATA_ANTERIOR(1, 1,2011, 1, 1, 2000);
echo "<br>";
```

```
function REGRA_DE_APOSENTADORIA($dia, $mes, $ano, $diaNasc, $mesNasc,
$anoNasc) {

  $diaAposIdade = $diaNasc;
  $mesAposIdade = $mesNasc;
  $anoAposIdade = $anoNasc+65;

  #verifica se tera 180 meses de trabalho
  if(DATA_ANTERIOR($diaAposIdade, $mesAposIdade, $anoAposIdade, $dia, $mes,
$ano+15) ){
   $diaAposIdade = $dia;
   $mesAposIdade = $mes;
   $anoAposIdade = $ano+15;
      echo "ao completar 65 anos, nao cumpriu 180 meses de trabalho";
      echo "<br>";
  }

  if (DATA_ANTERIOR($diaAposIdade, $mesAposIdade, $anoAposIdade, $dia,
$mes, $ano+45)  )
     echo "aposentadoria por idade" ;
  else
      echo "aposentadoria regular";
}

#testes:
echo REGRA_DE_APOSENTADORIA(1,1,2017,1,1,2017) ;
echo "<br>";
echo REGRA_DE_APOSENTADORIA(1,1,2017,1,1,2000) ;
echo "<br>";
echo REGRA_DE_APOSENTADORIA(1,1,2021,1,1,2000) ;
echo "<br>";
echo REGRA_DE_APOSENTADORIA(1,1,2060,1,1,2000) ;
echo "<br>";
```

c) (**3.0 ptos**) Escreva uma função chamada CALCULO_COM_INTERRUPCOES que recebe um vetor no qual cada elemento representa uma data na forma de segundos passados desde 1º de Janeiro de 1970 (12:00 h).
O vetor recebido como entrada é organizado de maneira que a primeira data equivale ao dia de início do primeiro emprego do trabalhador. Caso houver uma

segunda data, representa o dia de saída do primeiro emprego. Cada duas novas datas existentes no vetor representam a entrada em um novo emprego, seguida da data de saída do cargo correspondente. Caso o trabalhador esteja atualmente empregado, a última data registrada no vetor representará apenas a entrada no seu atual emprego.

Sua função deve calcular a soma dos dias de trabalho, e escrever na tela a quantidade total de dias, meses e anos completos de serviço. Você pode assumir um ano como contendo 365 dias. Ao analisar as datas a hora registrada pode ser ignorada, supondo que o trabalhador inicia tem seu dia de serviço contado como completo tanto no dia de início quanto ao final do contrato.

R: Se utilizarmos a estrutura DateTime ou as funções de date_create no PHP 5.3 ou superior, o parâmetro padrão esperado já é no formato timestamp unix. A resposta a seguir usa uma forma compatível com versões anteriores a 5.3 do PHP. Vamos trabalhar timestamps como inteiros, e converter apenas quando desejamos o valor formatado ao final, sabendo que a data se inicia em 01/01/1970 às 0:00:

```
function CALCULO_COM_INTERRUPCOES($datas) {
 if(count($datas) % 2 != 0)
$datas []= (new DateTime())->getTimestamp();
$dias = 0;
 for($i = 0, $j = 1; $i < count($datas); $i+=2, $j=$i+1) {
    $dias += $datas[$j]-$datas[$i];
  }
  $diasTotais = DateTime::createFromFormat('U', $dias);
  printf('%d ano(s), %d mes(es), %d dia(s)',
        $diasTotais->format('Y') - 1970,
        $diasTotais->format('m') - 1,
        $diasTotais->format('d') - 1);
}
```

d) **(4.0 ptos)** Escreva uma função chamada VALOR_DO_SALARIO_MEDIO que deve calcular e retornar o salário **mensal** médio do trabalhador, assumindo um mês como 30 dias corridos. Sua função recebe dois vetores, o primeiro contendo uma sequência de datas de ajuste salarial, na forma de segundos passados

desde 1º de Janeiro de 1970 (12:00 h). A primeira data é equivalente a entrada no emprego, e cada nova data registra o início de uma alteração salarial. Caso o valor armazenado de uma data seja negativo, representa a data de saída do emprego. O sinal então deve ser ignorado na interpretação desta data. A data seguinte a tal posição no vetor, se houver, passa a representar a entrada em um novo trabalho. Caso a última data contida no vetor seja positiva, considere que o trabalhador está empregado. A função recebe também um segundo vetor, contendo valores em reais descrevendo a sequência de salários e reajustes do trabalhador na ordem estabelecida pelas datas (positivas) do primeiro vetor.

Sua função deve escrever mensagens de erro na tela caso os vetores tenham tamanhos inconsistentes, ou seja, no caso de mais ou de menos valores de ajustes salariais do que intervalos de datas de dias de trabalho.

R: Esta função talvez seja a que permita um maior número de interpretações. O enunciado deixa em aberto o que seriam valores inconsistentes de tamanhos para os vetores de entrada, podendo o aluno considerar que para as demissões o salário é reajustado para zero ou mesmo que este seja omitido no segundo vetor, resultando em vetores de tamanhos distintos mas consistentes. Ambas estão corretas mas adotaremos a primeira interpretação aqui. Além disso, há várias opções de manipulação das datas, sendo talvez um caminho mais fácil considerar o salário como pago mês a mês, para um mês de 30 dias. Optaremos aqui por um caminho mais difícil, visando reaproveitar parte da lógica do código da resposta anterior. O salário aqui é considerado durante o período (em timestamp) em que o funcionário esteve empregado. Esta aproximação consegue distinguir períodos parciais de emprego (8 meses e 14 dias), mas não realiza os arrendondamentos costumeiros para fechar o dia de trabalho ou o mês de 30 dias, o que resulta em valores muito próximos mas não exatamente iguais aos valores arredondados sugeridos na questão.

```
 function VALOR_DO_SALARIO_MEDIO($datas, $salarios) {
   if(count($datas) == 0 || count($datas) != count($salarios))
        die('Vetores inconsistentes');
   $hoje = (new DateTime())->getTimestamp();
   $numerador = 0;
   $denominador = $hoje - (DateTime::createFromFormat('U',
$datas[0]))->getTimestamp();
   $datas []= $hoje;
   for($i = 0; $i < count($datas) - 1; $i++) {
        $periodo = abs($datas[$i+1]) - abs($datas[$i]);
        $numerador += $periodo * $salarios[$i]; if($salarios[$i] == 0) $denominador -=
$periodo;
```

```
    }
                    return $numerador/$denominador;
}
```