COMISSÃO PERMANENTE DE CONCURSOS

Universidade Estadual da Paraíba

PROCESSO SELETIVO

CARGO:

DESENVOLVEDOR (ANALISTA DE SISTEMAS)

EXAME GRAFOTÉCNICO:

(Transcreva a frase abaixo no local indicado na sua Folha de Respostas)

"Duvide do que vem fácil e não desista do que é difícil."

INSTRUÇÕES:

01 Verifique se este caderno de provas contém **10 (dez)** questões de múltipla escolha. Observe se há falhas ou imperfeições gráficas que causem dúvidas. Caso existam, comunique imediatamente ao Fiscal de Sala.

02 Esta Prova Teórica tem duração de **2 (duas)** horas. Não é permitida a saída do candidato antes de esgotado o tempo mínimo de **1 (uma)** hora.

03 Verifique se os dados existentes na Folha de Respostas conferem com os dados do Cartão de Inscrição e da etiqueta afixada na sua carteira.

04 É vetado, durante a prova, o intercâmbio ou empréstimo de material de qualquer natureza entre os candidatos, bem como o uso de celulares, calculadoras e/ou qualquer outro tipo de equipamento eletrônico.

05 A fraude, ou tentativa, a indisciplina e o desrespeito às autoridades encarregadas dos trabalhos são faltas que eliminam o candidato.

06 **Assine**, ao sair da sala, a **Lista de Presença** e entregue o seu Caderno de Prova e a Folha de Respostas, devidamente assinados, ao Fiscal de Sala.

PROVA OBJETIVA

01ª QUESTÃO
Sobre *Java*, considere as seguintes afirmações.

I- *JVM* (*Java Virtual Machine*) é uma especificação de uma máquina de computação abstrata que provê o ambiente de execução no qual o *bytecode Java* é executado. Estão disponíveis para diversas plataformas de hardware e software.
II- *JRE* (*Java Runtime Environment*) é uma implementação da *JVM*.
III- *JDK* (*Java Development Kit*) é um conjunto ferramentas de desenvolvimento de software em *Java* que inclui a *JRE*.

Está CORRETO o que se afirma em

a) II.
b) I e III.
c) II e III.
d) I e II.
e) I, II e III.

02ª QUESTÃO
É um padrão de projeto usado para criar um objeto sem expor a lógica de sua criação ao cliente, buscando o uso de interfaces comuns.

a) *Singleton*
b) *Factory*
c) *Iterator*
d) *Memento*
e) *Adapter*

03ª QUESTÃO
Considere a definição da seguinte classe Pessoa:

```
public class Pessoa {

  private Long id;
  private String nome;

  public boolean equals(Object obj) {
    Pessoa p = (Pessoa) obj;
    return this.id == p.getId() && this.nome.equals(p.getNome());
  }

  public int hashCode() {
    return this.nome.charAt(0);
  }

  /*Para economizar espaço, considere a existência de métodos get e set*/
}
```

A execução do código abaixo produz a seguinte resposta:

```
// No contexto de um método main

Collection<Pessoa> pessoas = new HashSet<Pessoa>();

Pessoa p1 = new Pessoa(1, "Maria");
Pessoa p2 = new Pessoa(2, "Maria");

pessoas.add(p1);
pessoas.add(p2);

for (Pessoa p : pessoas) {
  System.out.println("ID: " + p.getId() + " Nome: " + p.getNome());
}
```

a) null
b) ID: 1 Nome: Maria
c) ID: 2 Nome: Maria
d) ID: 2 Nome: Maria
ID: 1 Nome: Maria
e) [ ]

04ª QUESTÃO
Considere as duas funções Javascript definidas a seguir:

```
function funcao1()
{
 return {
    objeto: "olá"
 };
}

function funcao2()
{
 return
 {
    objeto: "olá"
 };
}
```

Agora considere as seguintes chamadas a essas funções:

```
console.log("funcao1 retorna:");
console.log(funcao1());
console.log("funcao2 retorna:");
console.log(funcao2());
```

Assinale a alternativa que representa a saída do console para as chamadas acima:

a) funcao1 retorna:
funcao2 retorna:

b) funcao1 retorna:
Object {objeto: "olá"}
funcao2 retorna:
Object {objeto: "olá"}

c) funcao1 retorna:
undefined
funcao2 retorna:
undefined

d) funcao1 retorna:
undefined
funcao2 retorna:
Object {objeto: "olá"}

e) funcao1 retorna:
Object {objeto: "olá"}
funcao2 retorna:
undefined

05ª QUESTÃO
Considere as duas funções *Javascript* definidas a seguir:

```
console.log(0.2 + 0.3);
console.log(0.2 + 0.3 == 0.5);
```

A saída do código abaixo sempre será

a) 0.5 e false.
b) 1.0 e true.
c) 0.5 e true.
d) undefined.
e) 1 e false.

06ª QUESTÃO
Hibernate contempla anotações *JPA* e possui outras anotações no pacote *org.hibernate.annotations*. NÃO é uma anotação *JPA* e nem *Hibernate*:

a) *javax.persistence.DataSet*
b) *javax.persistence.Entity*
c) *javax.persistence.Table*
d) *javax.persistence.Access*
e) *javax.persistence.Id*

07ª QUESTÃO
É uma classe *Java* registrada no *JSF* que gerencia a interação entre a interface do usuário e a lógica de negócio:

a) *ManagedBean*
b) *ManagedProperty*
c) *UIComponent*
d) *ActionEvent*
e) *ViewScoped*

08ª QUESTÃO
Sobre as características não funcionais do Spring Boot, julgue os itens a seguir:

I- A funcionalidade *spring-boot-starter-actuator* serve para funcionalidades avançadas tais como monitoramento e rastreamento para aplicações em configuração fora da caixa.
II- As funcionalidades *spring-boot-starter-undertow*, *spring-boot-starter-jetty*, *spring-boot-starter-tomcat* servem para escolher sua opção específica de *Embedded Servlet Container*.
III- A funcionalidade *spring-boot-starter-logging* serve para *logging* usando o *Logback*.

Está CORRETO o que se afirma em

a) II e III apenas.
b) I, II e III.
c) III apenas.
d) I e III apenas.
e) I apenas.

09ª QUESTÃO
Considere os itens a seguir:

I- *Hibernate* é uma especificação/interface.
II- *JPA* é uma das implementações do *Hibernate*.
III- Quando usamos *JPA*, utilizamos anotações e interfaces de *javax.persistence.package*, sem importar pacotes do *Hibernate*.

Está CORRETO o que se afirma em

a) II.
b) I.
c) III.
d) I, II e III.
e) II e III.

10ª QUESTÃO
De acordo com o ciclo de vida de um componente no *React.js*, assinale a alternativa em que todos os métodos são considerados como *Updating*:

a) getState(), forceUpdate() e render()
b) constructor() e componentWillMount()
c) componentDidMount() e componentWillUnmount()
d) componentWillReceiveProps(), static getDerivedStateFromProps() e render()
e) componentDidCatch(), constructor() e render()

Universidade Estadual da Paraíba

COMISSÃO PERMANENTE DE CONCURSOS

ATENÇÃO:

1. O Candidato poderá destacar esta folha de gabarito, após esgotado o tempo mínimo de 2 (duas) horas, e levá-lo para sua conferência.
2. O Gabarito Provisório será publicado no site da CPCON (www.cpcon.uepb.edu.br).
3. Qualquer dúvida, chame o fiscal.

GABARITO:

01	02	03	04	05	06	07	08	09	10