CONCURSO PÚBLICO
EDITAL 01/UFSC-UFFS/2009

CAMPO DE CONHECIMENTO: **ALGORITMOS E PROGRAMAÇÃO**

Atenção: NÃO ABRA este caderno antes do início da prova.
Tempo total para resolução desta prova: 4 (quatro) horas.

I N S T R U Ç Õ E S

Confira, no cartão-resposta, seu nome, seu número de inscrição e o campo de conhecimento para o qual se inscreveu. Transcreva seu nome e seu número de inscrição nos campos abaixo. Transcreva também os números correspondentes ao local, setor, grupo e ordem. Assine no local indicado.

Verifique no caderno de prova se faltam folhas, se a sequência de questões está correta e se há imperfeições gráficas que possam causar dúvidas. Comunique imediatamente ao fiscal qualquer irregularidade.

Para cada uma das **20 (vinte)** questões objetivas são apresentadas **5 (cinco)** alternativas (de **"A"** a **"E"**), das quais apenas **1 (uma)** é **correta**.

A interpretação das questões é parte integrante da prova, não sendo permitidas perguntas aos fiscais. Utilize os espaços e/ou páginas em branco para rascunho. **Não destaque folhas do caderno de prova**.

Examine o cartão-resposta e veja se há marcações indevidas no campo destinado às suas respostas. Se houver, reclame imediatamente.

Transcreva com **caneta esferográfica fabricada em material transparente, de tinta preta (preferencialmente)** ou **azul**, as respostas das questões objetivas para o cartão-resposta, que será o único documento válido para efeito de correção.

Em nenhuma hipótese haverá substituição do cartão-resposta ou da folha oficial da questão discursiva por erro de preenchimento ou qualquer dano causado pelo candidato.

Questões objetivas em branco, que contenham mais de uma resposta, emendas ou rasuras, não serão consideradas.

A resposta da questão discursiva deverá ser transcrita, **com caneta esferográfica de tinta preta** ou **azul** e dentro do tempo de duração da prova, para a folha oficial de resposta, a qual não deverá ser assinada nem identificada pelo(a) candidato(a).

Não será permitida, durante a realização da prova, a comunicação entre candidatos, o porte nem a utilização de aparelhos celulares ou similares, de calculadoras ou similares, de relógios, de livros de anotações, de impressos nem de qualquer outro material de consulta, sendo eliminado do concurso o(a) candidato(a) que descumprir esta determinação.

Ao terminar, entregue ao fiscal o caderno de prova, o cartão-resposta e a folha de resposta da questão discursiva. Você só poderá se retirar definitivamente do grupo de realização da prova a partir das **16h30min** (horário oficial de Brasília).

Para conferir suas respostas com o gabarito oficial, anote-as no quadro constante da última folha, o qual poderá ser destacado e levado com você.

INSCRIÇÃO	NOME DO(A) CANDIDATO(A)

ASSINATURA DO(A) CANDIDATO(A)	LOCAL / SETOR / GRUPO / ORDEM

QUESTÃO DISCURSIVA

A Lei n° 12.029, de 15 de setembro de 2009, que cria a Universidade Federal da Fronteira Sul, estabelece em seu Artigo 2° que "A UFFS terá por objetivo ministrar ensino superior, desenvolver pesquisa nas diversas áreas do conhecimento e promover a extensão universitária, caracterizando sua inserção regional mediante atuação *multicampi*, abrangendo, predominantemente, o norte do Rio Grande do Sul, com *campi* nos Municípios de Cerro Largo e Erechim, o oeste de Santa Catarina, com *campus* no Município de Chapecó, e o sudoeste do Paraná e seu entorno, com *campi* nos Municípios de Laranjeiras do Sul e Realeza".

Considerando que a Universidade Federal da Fronteira Sul (UFFS) é uma instituição universitária comprometida com a promoção da extensão e com o desenvolvimento da região na qual se insere, e considerando que uma das atribuições do professor da UFFS será desenvolver atividades de extensão, apresente e discuta, em um texto de até **30 (trinta) linhas**, uma proposta de extensão que possa produzir impacto positivo junto à sociedade e que, na sua maneira de ver, deva ser considerada prioritária pela UFFS. Explicite as razões que sustentam esta priorização, tendo em vista as contribuições que a sua área do conhecimento poderá trazer à região.

FOLHA DE RASCUNHO – QUESTÃO DISCURSIVA

ESTE RASCUNHO **NÃO** SERÁ CORRIGIDO!

	TÍTULO
01	
02	
03	
04	
05	
06	
07	
08	
09	
10	
11	
12	
13	
14	
15	
16	
17	
18	
19	
20	
21	
22	
23	
24	
25	
26	
27	
28	
29	
30	

TRANSCREVA A **QUESTÃO DISCURSIVA** DESTE RASCUNHO PARA A **FOLHA OFICIAL DA QUESTÃO DISCURSIVA**.

PROVA OBJETIVA

01) Dentre as características desejáveis de um programa de computador estão: Integridade, Clareza, Simplicidade, Eficiência, Modularidade e Generalidade. Assinale a alternativa que apresente a definição **CORRETA** de **Eficiência** de um programa.

A( ) Refere-se à precisão das informações geradas pelo programa.

B( ) Refere-se à solução de problemas de maneira simples com um código bem estruturado.

C( ) Refere-se à efetiva divisão do programa em módulos menores, bem identificados e com funções específicas.

D( ) Refere-se à velocidade de execução com a melhor utilização dos recursos de memória.

E( ) Refere-se à característica de um programa em permitir a reutilização de seus componentes em outros projetos.

02) Com relação aos métodos (técnicas) de desenvolvimento de algoritmos, considere as seguintes afirmativas:

I. *Branch-and-bound* e *Backtracking* são versões melhoradas do processo de Busca Exaustiva.

II. Divisão e Conquista é um método recursivo enquanto Programação Dinâmica e Método Guloso são métodos interativos.

III. o grau de aplicabilidade dos métodos de desenvolvimento de algoritmos é inversamente proporcional ao grau de especificidade desses métodos.

IV. os métodos Guloso e de Programação Dinâmica são tipicamente empregados na formulação de algoritmos que envolvem questões de otimização.

Assinale a alternativa **CORRETA**.

A( ) Somente as afirmativas I, II e III estão corretas.

B( ) Somente as afirmativas II e III estão corretas.

C( ) Somente as afirmativas II, III e IV estão corretas.

D( ) Somente as afirmativas I e IV estão corretas.

E( ) Todas as afirmativas estão corretas.

03) Assuma que uma lista encadeada é construída de elementos com dois campos: *valor* e *proximo*, sendo que o campo *proximo* de cada elemento é usado para apontar para o próximo elemento na lista. Se elementos da lista são descritos em C pela estrutura abaixo, à esquerda,

```
typedef struct elemento *ElemPtr;
    struct elemento {
        int valor;
        ElemPtr proximo;
    };
```

```
void insira_p_depois_de_q (ElemPtr p, ElemPtr q) {
        ...
}
```

qual das seguintes opções é uma implementação **CORRETA** em C da operação "insira *p* depois de *q*", apresentada acima, à direita, onde *q* aponta para um elemento da lista e *p* para um elemento a ser inserido?

A( ) p->prox = q->prox; q->prox = p;

B( ) q->prox = p->prox; p->prox = q;

C( ) q->prox = p->prox; p->prox = q->prox;

D( ) p->prox = q; q->prox = p->prox;

E( ) q->prox = p; p->prox = q->prox;

04) Com relação a estruturas de dados, identifique se são **verdadeiras (V)** ou **falsas (F)** as afirmativas a seguir.

( ) Estruturas do tipo "árvore" correspondem a estruturas do tipo "grafo acíclico dirigido".

( ) Tipos abstratos de dados como listas, pilhas e filas podem ser implementadas sobre uma mesma estrutura de dados do tipo "grafo", diferindo apenas nas operações de inserção e remoção de seus elementos.

( ) No pior caso, uma árvore não balanceada exige a metade das pesquisas necessárias a uma lista encadeada simples para a localização de um elemento.

( ) Tabela de espalhamento (ou tabela *hash*) é uma estrutura de dados que associa chaves de pesquisa a valores, sendo que, a partir de uma chave simples e de uma função de espalhamento, é possível fazer uma busca rápida e obter o valor desejado.

Assinale a alternativa que apresenta a sequência **CORRETA**, de cima para baixo.

A( ) F – F – F – V

B( ) V – V – F – V

C( ) V – V – V – F

D( ) V – F – V – V

E( ) F – V – F – F

05) Assinale a alternativa que apresenta **CORRETAMENTE** a árvore binária que dá origem aos percursos descritos abaixo:

Pré-ordem: J L G N D M E A F B I H C
In-ordem: N G L M E D J A B I F H C

A( ) N(G, L(M(E), D(J(A))), B(I(F, H(C))))
B( ) J(L(G(N), D(M(E))), A(F(B(I), H(C))))
C( ) J(L(G(N(D), M(E, A(F))), B(I)), H(C))
D( ) J(L(G(N(D), M(E)), A (F(B), I(H(C)))))
E( ) N(G(L(M), E(D)), J(A(B(I), F(H(C)))))

06) Sobre o assunto **algoritmos de ordenação**, assinale a alternativa **CORRETA**.

A( ) Os algoritmos *Quicksort* e *Bubblesort* utilizam o método Divisão e Conquista.
B( ) A ideia básica do algoritmo de ordenação *Bubblesort* é percorrer o conjunto de dados e, a cada passagem, sempre trocar cada elemento com o seu sucessor.
C( ) No algoritmo *Quicksort* o pivô pode ser um elemento arbitrário escolhido no conjunto de dados a serem ordenados.
D( ) No método de ordenação *Mergesort* a cada iteração pelo menos um elemento é posto em sua posição final e não será mais manipulado nas iterações seguintes.
E( ) O algoritmo de ordenação baseado em passar sempre o elemento mais apto (maior ou menor, dependendo da ordenação escolhida) do conjunto para a primeira posição, depois o segundo para a segunda posição, e assim sucessivamente, é denominado *Insertionsort*.

07) Sobre a estrutura de registros e as formas para acesso ao registro, considere as seguintes afirmativas:

I. para poder realizar a leitura aleatória de qualquer registro num arquivo, é necessário que os arquivos possuam registros de tamanho fixo.
II. o uso de índices associados a registros é útil quando fazemos a leitura sequencial de todos os registros de um arquivo.
III. delimitadores de final de registro são úteis quando os arquivos são compostos por registros de tamanho fixo e conhecido.
IV. uma chave primária é utilizada para identificar unicamente um registro.
V. nos arquivos indexados podem existir tantos índices quantas forem as chaves de acesso aos registros.

Assinale a alternativa **CORRETA**.

A( ) Somente as afirmativas I e V estão corretas.
B( ) Somente as afirmativas IV e V estão corretas.
C( ) Somente as afirmativas III e IV estão corretas.
D( ) Somente as afirmativas I, III e V estão corretas.
E( ) Somente as afirmativas II, IV e V estão corretas.

08) Um programador necessita fazer um programa de consulta a dados armazenados na memória de um computador. Considere que os dados sejam registros armazenados de forma totalmente desordenada na memória principal. Considere ainda que o sistema de consultas utilizará uma quantidade de buscas muito grande e que o tempo de resposta do programa é um fator crítico.

A alternativa **CORRETA** que leva a um programa mais eficiente é:

A( ) manter os registros desordenados e realizar uma busca binária a cada pesquisa.
B( ) manter os registros desordenados e realizar uma busca sequencial a cada pesquisa.
C( ) ordenar os registros antes de iniciar a fase de consultas e realizar uma busca sequencial a cada pesquisa.
D( ) ordenar os registros antes de iniciar a fase de consultas e realizar uma busca binária a cada pesquisa.
E( ) efetuar uma ordenação temporária dos registros a cada pesquisa e realizar uma busca binária.

09) Considere que o método dado parcialmente a seguir, escrito em Java, deve implementar um algoritmo de busca binária. O *array* **a** e a chave de busca **v** são os argumentos. O *array* **a** está ordenado em ordem não decrescente. O método retorna o índice do elemento, se ele for encontrado, e -1 caso contrário.

```
int buscabinaria (int [ ] a, int v) {
        int x, e, d;
        e = 0;
        d = a.length-1;
        while ( ..... ) {
                x = (e+d)/2;
                if (v<a[x])
                        d = x-1;
                else if (v>a[x])
                        e = x+1;
                else
                        return x;
        }
        return -1;
}
```

Assinale a alternativa que apresente a condição que, se inserida na linha pontilhada, torne esta implementação **CORRETA**.

A( ) e < d
B( ) e > d
C( ) e >= d
D( ) e != d
E( ) e <= d

10) Um exemplo típico do uso de **recursividade** é a solução do problema conhecido como Torres de Hanói. O problema consiste em mover ***n*** discos empilhados (os menores sobre os maiores), de uma haste de origem (A), para uma haste de destino (C), na mesma ordem, respeitando as seguintes regras: apenas um disco pode ser movido por vez, não colocar um disco maior sobre um menor e poder usar uma haste auxiliar (B). Uma implementação em Java para este problema é dada parcialmente a seguir.

```
public class Hanoi {
        public static void main (String [ ] args) {
                moverTorre (3, 'A', 'C', 'B');
        }
        public static void moverDisco (char inicio, char fim) {
                System.out.println (inicio+"\t"+ fim);
        }
        public static void moverTorre (int n, char inicio, char fim, char aux) {
                if (n == 1) {
                        moverDisco (inicio, fim);
                }
                else {
                        moverTorre ( ..... , ..... , ..... , ..... );        // comando 1
                        moverDisco (inicio, fim);
                        moverTorre ( ..... , ..... , ..... , ..... );        // comando 2
                }
        }
}
```

Assinale a alternativa que apresenta uma forma **CORRETA** de completar os comandos 1 e 2 do código anterior, resultando em uma solução válida para o problema das Torres de Hanói.

A( ) *moverTorre (**n-1, inicio, aux, fim**)* *// comando 1*
*moverTorre (**n-1, aux, fim, inicio**)* *// comando 2*

B( ) *moverTorre (**n-1, fim, aux, inicio**)* *// comando 1*
*moverTorre (**n-1, fim, aux, inicio**)* *// comando 2*

C( ) *moverTorre (**n-1, aux, inicio, fim**)* *// comando 1*
*moverTorre (**n-1, inicio, aux, fim**)* *// comando 2*

D( ) *moverTorre (**n, fim, aux, inicio**)* *// comando 1*
*moverTorre (**n, aux, inicio, fim**)* *// comando 2*

E( ) *moverTorre (**n, inicio, aux, fim**)* *// comando 1*
*moverTorre (**n, fim, aux, inicio**)* *// comando 2*

11) Com relação à complexidade de algoritmos, identifique se são **verdadeiras (V)** ou **falsas (F)** as afirmativas:

( ) dentre as complexidades de melhor caso, de caso médio e de pior caso, a mais importante, na prática, é a complexidade de melhor caso.
( ) problemas computacionais tratáveis e intratáveis possuem, respectivamente, algoritmos com complexidade de pior caso Polinomial e Exponencial.
( ) os algoritmos *Quicksort* e *Mergesort* possuem a mesma complexidade de pior caso.
( ) um algoritmo pertence à classe *P* se, para quaisquer entradas de tamanho *n*, sua complexidade no pior caso é $O(n^k)$ para alguma constante *k*.
( ) um algoritmo cuja complexidade de pior caso é igual ao limite assintótico inferior do problema em questão é um algoritmo ótimo.
( ) algoritmos para adicionar e multiplicar matrizes, que possuam complexidade de pior caso $O(n^2)$ e $O(n^3)$ respectivamente, são algoritmos ótimos.

Assinale a alternativa que apresenta a sequência **CORRETA**, de cima para baixo.

A( ) V – F – V – F – V – F
B( ) F – V – F – V – F – V
C( ) V – V – F – F – V – F
D( ) F – F – V – V – F – V
E( ) F – V – F – V – V – F

12) Com relação aos paradigmas de programação, considere as seguintes afirmativas:

I. Os paradigmas de programação Lógica e Funcional são considerados paradigmas declarativos, enquanto os paradigmas de programação Estruturada e Orientada a Objetos são paradigmas imperativos.
II. Os paradigmas de programação Funcional e Lógica baseiam-se, respectivamente, em *Funções Recursivas* e *Lambda Calculus*.
III. Concorrência e Distribuição são recursos suportados exclusivamente por Linguagens Imperativas.
IV. A maior eficiência das linguagens imperativas com relação às declarativas está diretamente relacionada à arquitetura Von Neumann prevalecente na quase totalidade dos computadores digitais existentes.
V. Em linguagens Funcionais puras a repetição só pode ser efetuada por recursão, pois elas não suportam variáveis, nem atribuição e nem iteração.

Assinale a alternativa **CORRETA**.

A( ) Somente a afirmativa II, III e V são corretas.
B( ) Somente as afirmativas II, III e IV são corretas.
C( ) Somente as afirmativas I, IV e V são corretas.
D( ) Somente as afirmativas I, II e V são corretas.
E( ) Todas as afirmativas são corretas.

13) Aplicações para web com alto grau de interatividade com o usuário, em que parte do processamento é feito na máquina cliente, são chamadas de aplicações WEB 2.0.

Assinale a alternativa que apresenta **CORRETAMENTE** a especificação que define um modelo de processamento baseado em eventos a ser executado na máquina cliente.

A( ) DOM Level 1.

B( ) DOM Level 2.

C( ) CSS 1.

D( ) CSS 2.

E( ) JSON.

14) Preencha as lacunas com o padrão de projeto correspondente.

( ) Define a dependência um-para-muitos entre objetos para que, quando um objeto muda de estado, todos os seus dependentes sejam avisados e atualizados automaticamente.

( ) Anexa responsabilidades adicionais a um objeto dinamicamente. Fornece uma alternativa flexível de subclasse para estender a funcionalidade.

( ) Define uma interface para criar um objeto, mas permite às classes decidir qual classe instanciar.

( ) Garante que uma classe tenha apenas uma instância e fornece um ponto global de acesso a ela.

( ) Encapsula uma solicitação como um objeto, permitindo parametrizar outros objetos com diferentes solicitações, enfileirar ou registrar solicitações e implementar recursos de cancelamento de operações.

Assinale a alternativa que apresenta a sequência **CORRETA**, de cima para baixo.

A( ) *Adapter – Template – Singleton – Factory – Command*.

B( ) *Iterator – Decorator – Facade – Proxy – Observer*.

C( ) *Observer – Proxy – Template – Singleton – Decorator*.

D( ) *Observer – Decorator – Factory – Singleton – Command*.

E( ) *Adapter – Strategy – Template – Factory – Observer*.

15) Com relação a **componentes de *software***, considere as seguintes afirmativas:

I. componentes compõem o conjunto de ferramentas de *software* utilizadas para gerar uma aplicação completa em formato binário, o que inclui principalmente os componentes de compilação, ligação e carga.
II. os desenvolvedores de diferentes aplicações precisam conhecer detalhes de implementação dos componentes que usarão em suas aplicações para verificar se eles podem ser reusáveis.
III. componentes podem ser definidos como uma unidade independente com possibilidade de reutilização, podendo ser conectados a outros componentes para formar uma aplicação completa.
IV. componentes são análogos às funções-membro (métodos) de objetos no desenvolvimento orientado a objeto.

Assinale a alternativa **CORRETA**.

A( ) Somente a afirmativa I está correta.
B( ) Somente a afirmativa IV está correta.
C( ) Somente a afirmativa III está correta.
D( ) Somente as afirmativas I e II estão corretas.
E( ) Somente as afirmativas III e IV estão corretas.

16) Sobre conceitos relacionados ao paradigma de orientação a objetos, considere as afirmativas a seguir:

I. Encapsulamento é a capacidade de uma operação atuar de modos diversos em classes diferentes.
II. Polimorfismo é o compartilhamento de atributos e métodos entre classes com base em um relacionamento hierárquico.
III. Herança consiste no processo de ocultação dos detalhes internos de implementação de um objeto.
IV. Sobreposição é a redefinição das funções de um método herdado. Os métodos apresentam assinaturas iguais.

A partir da análise das afirmativas anteriores, é **CORRETO** afirmar que:

A( ) somente a afirmativa IV está correta.
B( ) somente as afirmativas III e IV estão corretas.
C( ) somente as afirmativas I e IV estão corretas.
D( ) somente as afirmativas II e III estão corretas.
E( ) todas as afirmativas estão incorretas.

17) Em programação orientada a objetos os atributos e métodos de uma classe podem ser privados, protegidos ou públicos. Considere que alguns atributos e métodos de uma dada classe devam ser acessados somente por instâncias desta mesma classe e de suas subclasses.
Assinale a alternativa que indica **CORRETAMENTE** o modificador de acesso mais adequado para estes atributos e métodos.

A( ) Todos privados.
B( ) Todos públicos.
C( ) Todos protegidos.
D( ) Privados ou protegidos.
E( ) Privados, protegidos ou públicos.

18) Considere as seguintes afirmativas sobre classes *container*:

I. É qualquer tipo de classe cujas instâncias são coleções de outros objetos.
II. Usualmente implementam alguma estrutura de dados tal como lista, pilha ou fila.
III. Uma restrição ao uso de *containers* é que os objetos componentes não podem ser incluídos ou removidos dinamicamente.
IV. Um exemplo de *container* para Web é o Tomcat.

A partir da análise das afirmativas anteriores, é **CORRETO** afirmar que:

A( ) somente as afirmativas I e III estão corretas.
B( ) somente as afirmativas I e IV estão corretas.
C( ) somente as afirmativas I, II e IV estão corretas.
D( ) somente as afirmativas II e III estão corretas.
E( ) todas as afirmativas estão corretas.

19) Considerando as características da linguagem de programação Java, analise as seguintes afirmativas:

I. diz-se que o nome de um método foi sobrecarregado (*overloaded*) se dois métodos têm o mesmo nome, mas assinaturas diferentes.
II. dois métodos de classes diferentes sobrepõem-se se têm a mesma assinatura e necessariamente o mesmo tipo de resultado.
III. o construtor da superclasse pode ser invocado utilizando-se o *super( )* na primeira linha do construtor da subclasse.
IV. o uso do *this* serve para resolver ambiguidade, para passar o objeto atual como parâmetro e também para invocar um construtor da mesma classe.
V. se uma variável ***x*** é redefinida numa subclasse, essa declaração oculta (*hides*) a variável definida na superclasse.

A partir da análise das afirmativas anteriores, é **CORRETO** afirmar que:

A( ) somente as afirmativas II, III e IV estão corretas.
B( ) somente as afirmativas I, II, III e V estão corretas.
C( ) somente as afirmativas I, II e IV estão corretas.
D( ) somente as afirmativas I, III e V estão corretas.
E( ) todas as afirmativas estão corretas.

20) Considere o programa a seguir, escrito em Java:

```
class Box {
        private int i;
        public Box (int i) {
                set (i);
        }
        public void set (int i) {
                this.i = i;
        }
        public int get ( ) {
                return i;
        }
}
class UsaBox {
        public static void main (String [ ] args) {
                Box i = new Box (5);
                Box j = i;
                j.set (10);
                System.out.println ("i = "+i.get( )+"   j = "+j.get( ));
        }
}
```

Assinale a alternativa que apresenta **CORRETAMENTE** o resultado que será impresso.

A( ) i = 5 j = 10
B( ) i = 10 j = 10
C( ) i = 10 j = 5
D( ) i = 5 j = 5
E( ) Nenhuma das alternativas anteriores apresenta o resultado que será impresso.

✂----------

GRADE DE RESPOSTAS (Somente esta parte poderá ser destacada)

QUESTÕES	01	02	03	04	05	06	07	08	09	10	11	12	13	14	15	16	17	18	19	20
RESPOSTAS																				