INSTITUTO NACIONAL DE PESQUISA ESPACIAL – INPE

PROVA OBJETIVA

TG19

DESENVOLVIMENTO DE SOFTWARE EMBARCADO

SUA PROVA

- Além deste caderno contendo **45 (quarenta e cinco)** questões objetivas, você receberá do fiscal de prova o cartão de respostas;
- As questões objetivas têm **5 (cinco)** opções de resposta (A, B, C, D e E) e somente uma delas está correta.

TEMPO

- Você dispõe de **4 (quatro) horas** para a realização da prova;
- **2 (duas) horas** após o início da prova, é possível retirar-se da sala, sem levar o caderno de questões;
- A partir dos **30 (trinta) minutos** anteriores ao término da prova é possível retirar-se da sala **levando o caderno de questões**.

NÃO SERÁ PERMITIDO

- Qualquer tipo de comunicação entre os candidatos durante a aplicação da prova;
- Anotar informações relativas às respostas em qualquer outro meio que não seja no caderno de questões e nas folhas de textos definitivos;
- Levantar da cadeira sem autorização do fiscal de sala;
- Usar o sanitário ao término da prova, após deixar a sala.

INFORMAÇÕES GERAIS

- Verifique se seu caderno de questões está completo, sem repetição de questões ou falhas e também confira seu cargo. Caso tenha recebido caderno de cargo **diferente** do impresso em seu cartão de respostas, o fiscal deve ser **obrigatoriamente** informado para o devido registro na ata da sala;
- Confira seus dados pessoais, especialmente nome, número de inscrição e documento de identidade e leia atentamente as instruções para preencher o cartão de respostas;
- Para o preenchimento do cartão de respostas, use somente caneta esferográfica, fabricada em material transparente, com tinta preta ou azul;
- Assine seu nome apenas no(s) espaço(s) reservado(s) no cartão de respostas;
- Reserve tempo suficiente para o preenchimento do seu cartão de respostas. O preenchimento é de sua responsabilidade e **não será permitida a troca do cartão de respostas em caso de erro cometido pelo candidato**;
- Para fins de avaliação, serão levadas em consideração apenas as marcações realizadas no cartão de respostas;
- A FGV coletará as impressões digitais dos candidatos na lista de presença;
- Os candidatos serão submetidos ao sistema de detecção de metais quando do ingresso e da saída de sanitários durante a realização das provas.

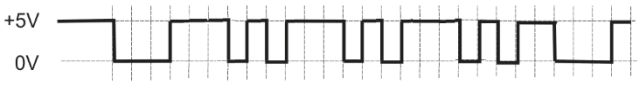
Boa Prova!

FGV CONHECIMENTO

CONHECIMENTOS ESPECÍFICOS

1

A figura a seguir mostra a forma de onda gerada por uma UART a ser posteriormente convertida em padrão RS-232, sem paridade e com palavras de 8 bits. Repare que o tracejado delimita o tamanho de um bit.



Assinale a opção que indica o primeiro *byte* gerado, em binário.

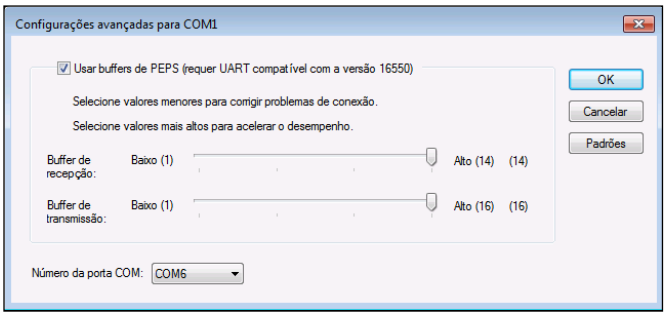
(A) 00111010.
(B) 00011101.
(C) 11101011.
(D) 01011100.
(E) 00101000.

2

A figura a seguir mostra propriedades de uma UART retirada do Painel de Controle de um sistema operacional de 32 bits.

Para transmitir alguns caracteres, usou-se pela primeira vez o seguinte código em linguagem C:

```
handle = CreateFile("COM6",...);
WriteFile(handle,"Transmissao de uma sequencia de bytes",
                  38,&ok,NULL).
```

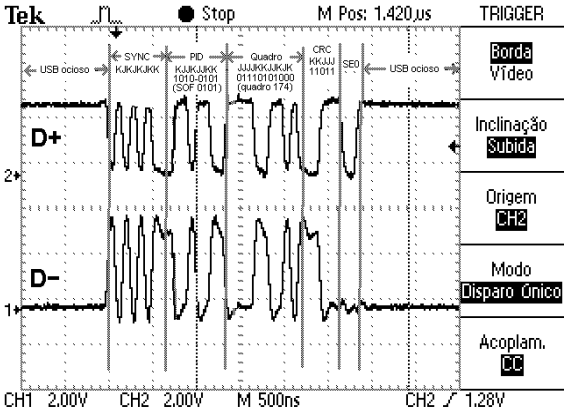


O último caractere transmitido com o código foi

(A) **'u'**
(B) **'o'**
(C) **'s'**
(D) **0** (caracter nulo)
(E) pula linha (LF)

3

A tela de osciloscópio a seguir foi capturada de uma comunicação USB 1.1 durante um token do tipo *"Start Of Frame"*.



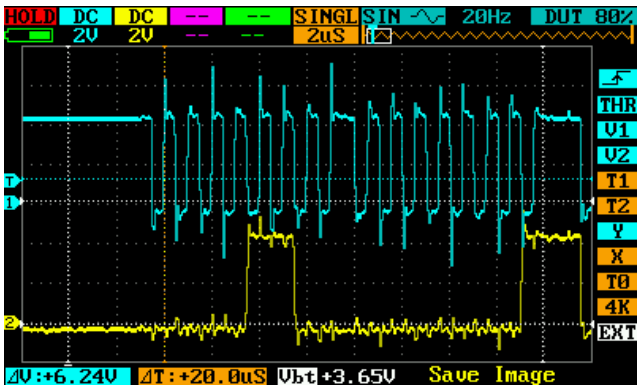
Fonte:"Microprocessadores x86: Arquitetura e Interfaceamento." Ed. Interciência.

Os campos destacados (SYNC, PID, Quadro, CRC e SE0) representam, respectivamente,

(A) sincronismo, identificador de pacote, número do quadro, checagem de paridade dos dados do grupo de pacotes e desconexão.
(B) sincronismo, pacote inicial, número do quadro, checagem de paridade dos dados do próprio pacote e desconexão.
(C) sincronismo, pacote inicial, número do quadro, checagem de paridade dos dados do próprio pacote e inversão do sentido de transmissão.
(D) sincronismo, pacote inicial, número do quadro, checagem de paridade dos dados do grupo de pacotes e inversão do sentido de transmissão.
(E) sincronismo, identificador de pacote, número do quadro, checagem de paridade dos dados do próprio pacote e desconexão.

4

A figura a seguir foi retirada de uma comunicação síncrona SPI usando o mostrador digital MAX 7219.



Fonte: Módulos e Sensores: Guia de Interface com o Arduino. Editora Interciência.

Sabe-se que a amostragem é no flanco de subida do relógio. Os dois bytes comunicados, em hexadecimal, são

(A) 06 e 03.
(B) 0C e 03.
(C) 07 e 03.
(D) 0C e 01.
(E) 06 e 01.

5

Com relação aos diversos modelos de ciclo de vida de um *software*, considere a seguinte situação hipotética:

Uma equipe inicia um projeto com uma abordagem linear e sequencial. No entanto, ao longo do projeto substitui esta abordagem por um modelo com ênfase em protótipos no início de cada estágio para validação de conceitos e na análise de riscos.

Assinale a opção que apresenta o modelo que esta equipe empregou inicialmente e o modelo pelo qual este foi substituído.

(A) cascata e *scrum*.
(B) desenvolvimento *lean* e espiral.
(C) cascata e espiral.
(D) espiral e RAD.
(E) cascata e incremental.

6

O *Scrum* é um *framework* de gerenciamento que as equipes usam para se auto-organizar e trabalhar em direção a um objetivo em comum.

Sobre o *Scrum*, analise as afirmativas a seguir.

I. Por ser uma metodologia ágil, um de seus pilares é a eliminação de reuniões.
II. O *backlog* do produto é uma lista dinâmica de funcionalidades organizada por prioridades.
III. O *Product Owner* planeja os recursos necessários para cada *Sprint*.

Está correto o que se afirma em

(A) I, apenas
(B) II, apenas.
(C) III, apenas.
(D) I e II, apenas.
(E) II e III, apenas.

7

No contexto de Projetos Orientados a Objetos, padrões de *design* são soluções generalizadas para problemas comuns de *design* de *software*.

Considere uma situação em que um desenvolvedor foi incumbido de elaborar um sistema de criação de documentos de diversos formatos, como Texto, Planilha e Apresentação, a serem definidos com base nos comandos do usuário.

Para lidar com esses requisitos, o padrão de *design* de *software* mais adequado seria o

(A) Singleton.
(B) Factory Method.
(C) Heritage.
(D) Builder.
(E) Strategy.

8

Um sistema distribuído tem como objetivo coletar e distribuir dados meteorológicos para um conjunto de usuários. Para tal, foram empregados pelo desenvolvedor dois padrões de *design*:

Padrão (1): Criação de uma classe com apenas uma instância, responsável por controlar o acesso a um determinado sensor operado por equipes técnicas remotas.

Padrão (2): Definição de um objeto principal, de forma que todos os usuários dependentes são notificados e atualizados em tempo real.

Assinale a opção que indica os padrões de *design* (1) e (2), respectivamente.

(A) Singleton e Heritage.
(B) Strategy e Composite.
(C) Strategy e Observer.
(D) Singleton e Observer.
(E) Builder e Factory.

9

Existem diversas abordagens para técnicas, modelos e processos de um ciclo de vida de desenvolvimento de *software*.

Relacione as afirmativas I, II, III e IV às denominações mais adequadas correspondentes a essas técnicas, modelos e processos.

I. Usa uma abordagem que capacita o desenvolvedor e o cliente a entender e reagir aos riscos em cada etapa evolutiva.
II. Requisitos dos usuários são priorizados e os requisitos de mais alta prioridade são incluídos nas iterações iniciais.
III. Apropriado quando os requisitos são bem compreendidos e as mudanças serão bastante limitadas durante o design.
IV. Usado quando o desenvolvedor não tem certeza da eficiência de um algoritmo, ou da forma da interação homem/máquina

(A) I=Espiral; II=Incremental; III=Cascata; IV=Prototipação
(B) I= Incremental; II= Espiral; III=Cascata; IV=Prototipação
(C) I= Incremental; II= Espiral; III= Prototipação; IV= Cascata
(D) I=Espiral; II=Incremental; III= Prototipação; IV= Cascata
(E) I=Espiral; II=Cascata; III= Incremental; IV=Prototipação

10

A integração de módulos e sistemas é uma etapa crucial no desenvolvimento de *software*, em que a abordagem de integração a ser tomada depende do cenário concreto.

Assinale a opção que apresenta o cenário para o qual a abordagem *Big Bang* é a mais adequada.

(A) Testar gradualmente os diferentes componentes do sistema.
(B) Resolver os problemas conhecidos de integração entre determinados módulos.
(C) Identificar os problemas de integração de maneira incremental.
(D) Realizar uma abordagem cuidadosa para garantir a integração correta entre os módulos.
(E) Integrar simultaneamente todos os módulos e testá-los em conjunto.

11

O desenvolvimento orientado a testes (TDD) engloba um conjunto de práticas que visam garantir a detecção de erros e a qualidade do código gerado.
No escopo do TDD, assinale a opção que apresenta a característica do teste unitário.
(A) Ser realizado desde o início do projeto.
(B) Ser desnecessário em pequenos projetos.
(C) Avaliar mudanças em um sistema como um todo.
(D) Ser substituível por testes de regressão.
(E) Verificar o funcionamento de módulos em conjunto.

12

Em um projeto de desenvolvimento de *software*, uma equipe está focada em diferentes aspectos do sistema.

Assinale a opção que indica o diagrama UML mais adequado para se entender como os objetos do sistema interagem entre si ao longo do tempo.

(A) Diagrama de Sequência.
(B) Diagrama de Classes.
(C) Diagrama de Objetos.
(D) Diagrama de Estado.
(E) Diagrama de Casos de Uso.

13

Um sistema gera um vetor fixo de dados para efetuar o controle de um processo usando o código em C abaixo, em um compilador que trabalha com inteiros de 32 bits:

```
int matriz[] = { 1, 2, 3, 4, 5, 6, 7, 8 };
int *v,final;
v = &matriz[1];
final = v[3];
```

Após a execução desse trecho de código, o valor de **final** é:

(A) 6
(B) 5
(C) 4
(D) 3
(E) 2

14

Softwares para sistemas embarcados podem ser desenvolvidos por meio de programação em linguagens *assembly*.

Com relação a linguagens *assembly*, analise as afirmativas a seguir.

I. São linguagens de programação de nível relativamente baixo, que utilizam mnemônicos equivalentes a representações de código binário (ou *opcode*) das instruções de uma determinada arquitetura de processador.
II. Em geral, possuem instruções de acesso direto à memória RAM, permitindo leitura e escrita de dados em posições determinadas por endereçamento direto.
III. São linguagens adequadas para se trabalhar com estruturas de dados complexas, tais como as classes, próprias de programação orientada a objetos.

Está correto o que se afirma em

(A) I, apenas.
(B) III, apenas.
(C) I e II, apenas.
(D) I e III, apenas.
(E) I, II e III.

15

Em um trecho de um programa em linguagem *assembly* para um microcontrolador 8051, foi encontrada a seguinte sequência de instruções:

```
MOV A, #049h
MOV B, #05h
MUL AB
RL A
RL A
```

Após o processamento dessas instruções, o valor armazenado no registrador acumulador é

(A) B4h.
(B) B5h.
(C) 16Fh.
(D) 1Bh.
(E) 5Bh.

16

Na programação de sub-rotinas em linguagem *assembly*, é comum o armazenamento temporário de dados e/ou de endereços, para que sejam restaurados logo após o retorno à função principal. As pilhas (*stacks*) são estruturas de dados do tipo LIFO (*Last In, First Out*) adequadas para armazenar dados e/ou endereços quando alguma sub-rotina é chamada. Os conjuntos de instruções das arquiteturas mais comumente utilizadas possuem instruções específicas de acesso à pilha.

Na arquitetura 8051, as instruções usadas para retirar dados da pilha e para armazenar dados na pilha, colocando-os em registradores de uso geral, são, respectivamente,

(A) RET e ACALL.
(B) RETI e LCALL.
(C) SWAP e SJMP.
(D) POP e PUSH.
(E) DEC SP e INC SP.

17

Com relação às linguagens de programação C/C++, analise as afirmativas a seguir.

I. Seja x uma variável do tipo inteiro. Na declaração abaixo, o ponteiro p é inicializado com o endereço de x. int *p = &x.
II. O comando *break* somente pode ser utilizado em conjunto com o comando *switch.*
III. O comando *return* encerra a execução de uma função.

Está correto o que se afirma em

(A) I, apenas.
(B) II, apenas.
(C) III, apenas.
(D) I e II, apenas.
(E) I e III, apenas.

18

Com relação à linguagem de programação C++ e o paradigma da orientação a objeto, analise as afirmativas a seguir.

I. Uma classe define o comportamento dos objetos que são instâncias da classe.
II. Em C++ é permitido criar classes derivadas, seguindo o conceito de herança de classes.
III. O polimorfismo permite que objetos de classes diferentes respondam de forma diferente à mesma função.

Está correto o que se afirma em

(A) I, apenas.
(B) II, apenas.
(C) I e II, apenas.
(D) II e III, apenas.
(E) I, II e III.

19

A técnica de alocação de memória utilizada por um sistema operacional está intimamente ligada à ocorrência de fragmentação dessa memória.

Com relação à gerência de memória em sistemas operacionais, assinale a afirmativa correta.

(A) A fragmentação interna ocorre entre blocos alocados de memória adjacentes.
(B) Na segmentação, os processos são armazenados em blocos de memória de tamanho fixo.
(C) A segmentação elimina a possibilidade de fragmentação interna.
(D) A paginação evita a fragmentação externa mais eficientemente do que a segmentação.
(E) A paginação aloca memória de acordo com a estrutura lógica dos programas.

20

A divisão dos processos em *threads* permite a execução de tarefas de maneira paralela ou concorrente.

As *threads,* em um sistema operacional Linux,

(A) podem ser terminadas seletivamente através do comando *kil*.
(B) são executadas de forma independente para minimizar conflitos.
(C) compartilham, em um mesmo processo, o mesmo espaço de endereçamento.
(D) apresentam um custo de processamento maior durante mudanças de contexto do que os processos.
(E) não são vantajosas em sistemas multiprocessadores, por causarem perda de desempenho.

21

Um conceito fundamental para o melhor aproveitamento de recursos por um sistema operacional é o de *multithreading*.

Para o emprego eficaz de *multithreading* em um sistema operacional, é fundamental a existência de mecanismos de sincronização eficientes.

Nesse contexto, analise as afirmativas a seguir.

I. Por definição, semáforos possuem um contador, cujos valores podem ser 0,1 ou 2.
II. *Mutexes* são projetados para garantir que apenas uma *thread* possa acessar um recurso compartilhado por vez.
III. *Mutexes* podem ser considerados uma generalização de semáforos, por conta da maior flexibilidade do contador de um *mutex*.

Está correto o que se afirma em

(A) I, apenas.
(B) II, apenas.
(C) III, apenas.
(D) II e III, apenas.
(E) I, II e III.

22

Os sistemas de arquivos proporcionam uma interface para armazenamento e recuperação de dados em um sistema operacional, cujas implementações podem diferir significativamente em estrutura e funcionalidades.

Em relação aos sistemas de arquivos dos sistemas operacionais, analise as afirmativas a seguir:

I. O sistema NTFS tem como um de seus componentes fundamentais o MFT (*Master File Table*), responsável por armazenar os metadados de todos os arquivos e diretórios presentes em um volume.
II. No sistema ext4, o sistema de arquivos é dividido em grupos de blocos (*Block Groups*), e cada um desses grupos possui seu próprio controle de metadados.
III. O registro de transações (*journaling*) tem como principal objetivo garantir a integridade dos dados e está presente no Linux desde o sistema ext2.

Está correto o que se afirma em

(A) I, apenas.
(B) III, apenas.
(C) I e II, apenas.
(D) II e III, apenas.
(E) I, II e III.

23

Com relação as situações em que a camada de abstração de *hardware* (*HAL*) de um sistema operacional é necessária, analise as afirmativas a seguir.

I. Quando um programador desenvolve um aplicativo sem ter que se preocupar com as características de *hardware* da máquina na qual ele vai ser executado.
II. Quando um aplicativo precisa ser altamente otimizado para um *hardware* específico, sem se preocupar com a portabilidade para outros dispositivos.
III. Quando é necessário acessar recursos de *hardware* específicos de um dispositivo, como sensores e periféricos, de maneira independente do *hardware* subjacente.

Está correto o que se afirma em

(A) I, apenas.
(B) III, apenas.
(C) I e II, apenas.
(D) I e III, apenas.
(E) I, II e III.

24

As variáveis são uma ferramenta essencial para a programação, as quais permitem armazenar dados definidos apenas na execução, executar e salvar o resultado de operações lógicas e aritméticas, entre outras possibilidades.

A respeito dos diferentes tipos de variáveis que podem ser usadas em um programa, é correto afirmar que

(A) *overflow* e *underflow* não podem ocorrer ao se realizarem operações aritméticas com variáveis do tipo real com representação em ponto-flutuante.
(B) os vetores possuem uma estrutura que permite armazenar uma quantidade pré-definida de variáveis de tipos distintos entre si.
(C) todos os caracteres representados por uma variável do tipo char utilizando codificação ASCII podem ser impressos na tela.
(D) o maior número que um inteiro sem sinal de 8 bits pode representar é 256.
(E) as matrizes são armazenadas de forma contígua na memória.

25

O correto entendimento sobre os conceitos relacionados a operadores é essencial para o desenvolvimento de programas. Considere as seguintes instruções de parte de um programa desenvolvido em linguagem de programação C:

```
int x, y;
x = 25;
++x;
y = x++;
y+=  x+13-11*2;
```

Os valores das variáveis x e y, após a execução dessas instruções acima, serão, respectivamente:

(A) 27 e 704
(B) 27 e 44
(C) 26 e 43
(D) 27 e 45
(E) 28 e 703

26

A Notação Polonesa Reversa (RPN, do inglês *Reverse Polish Notation*) foi desenvolvida como uma forma de escrever expressões lógicas e aritméticas sem usar parênteses. Essa notação ganhou popularidade ao ser implementada em calculadoras científicas, onde permite reduzir a quantidade de acionamento de teclas no cálculo de expressões.

Quando uma calculadora opera no modo RPN, os operandos são inseridos previamente em uma estrutura de dados e, ao utilizar-se um operador (soma, subtração, ...), a quantidade de operandos necessários são retirados da estrutura na ordem inversa da inserção e, após o cálculo da operação, o resultado é inserido na estrutura de dados. Assim, por exemplo, caso se deseje calcular a expressão A + (B – C)*D em uma calculadora operando no modo RPN, pode-se seguir o seguinte procedimento:

- Insere A
- Insere B
- Insere C
- Realiza a operação de subtração
- Insere D
- Realiza a operação de multiplicação
- Realiza a operação de soma

De acordo com a descrição acima, assinale a opção que indica a estrutura de dados que melhor caracteriza a utilizada pelo modo RPN para armazenar os operandos e resultados.

(A) Lista duplamente encadeada.
(B) Lista encadeada circular.
(C) Pilha.
(D) Fila.
(E) Árvore.

27

As estruturas de dados utilizadas em programação determinam como as informações serão armazenadas, organizadas e acessadas, sendo uma parte importante no projeto de *software*, com impacto no seu desempenho e eficiência.

Sobre estruturas de dados lineares, analise as afirmativas a seguir.

I. Para realizar uma busca por um elemento em uma lista simplesmente encadeada pode-se começar a busca pelo início ou fim da lista.
II. Listas duplamente encadeadas não permitem a exclusão de elementos que não sejam o último ou o primeiro elemento da lista.
III. Uma lista circular pode ser simplesmente encadeada ou duplamente encadeada.

Está correto o que se afirma em

(A) I, apenas.
(B) II, apenas.
(C) III, apenas.
(D) I e III, apenas.
(E) II e III, apenas.

28

Cada um dos componentes de um processador possui uma função específica, e trabalham em conjunto para executar instruções e processar dados.

A tarefa de acessar a memória principal em busca de instruções é efetuada pela(o)

(A) unidade lógica e aritmética.
(B) registrador de instruções.
(C) unidade de gerenciamento de memória.
(D) unidade de controle.
(E) gerenciador de acesso externo.

29

A memória *cache* atua como um intermediário entre a CPU e a memória principal, sendo organizada em níveis.

Sobre este tipo de memória, analise as afirmativas a seguir.

I. A memória *cache* de menor nível (L1) é a que possui o maior tempo de acesso.
II. Os dados armazenados na memória *cache* de nível L1 são mais frequentemente acessados que aqueles no nível L3.
III. O tempo de acesso à memória *cache* não influencia o desempenho do processador.

Está correto o que se afirma em

(A) I, apenas.
(B) II, apenas.
(C) III, apenas.
(D) II e III, apenas.
(E) I, II e III.

30

Os processadores em geral, no que tange à sua arquitetura, podem ser classificados em dois grandes grupos: RISC e CISC.

Os processadores com arquitetura

(A) RISC empregam a técnica de *pipelining* mais eficazmente.
(B) CISC facilitam a otimização do código por compiladores.
(C) CISC aceitam instruções apenas de um tamanho fixo pré-determinado.
(D) RISC tendem a ser mais complexos do ponto de vista do *hardware.*
(E) RISC empregam microinstruções para os comandos mais complexos.

31

As métricas de desempenho levam em consideração diferentes aspectos do desempenho computacional.

Sobre as métricas FLOPS e MIPS, analise as afirmativas a seguir.

I. FLOPS é a métrica mais adequada para ambientes cujos objetivos sejam gráficos detalhados e simulações físicas.
II. MIPS é relevante em contextos em que é necessária uma medida mais genérica e direta da execução de instruções pelo processador.
III. FLOPS pode ser imprecisa quando comparando processadores com arquiteturas distintas, ao contrário da MIPS, que fornece um indicador mais direto da capacidade de cálculo.

Está correto o que se afirma em

(A) I, apenas.
(B) II, apenas.
(C) I e II, apenas.
(D) II e III, apenas.
(E) I, II e III.

32

A métricas usada para classificar o desempenho de um processador é a chamada MIPS (*Millions of Instructions per Second*).

Nesse contexto, o valor do *score* dado pela métrica MIPS

(A) é um indicador direto e unívoco do poder de processamento.
(B) considera apenas a quantidade de instruções executadas.
(C) cresce com o aumento da complexidade das instruções.
(D) é mais preciso quando existem gráficos sendo gerados.
(E) não é afetado pela arquitetura do processador.

33

O consumo consciente de energia é importante para minimizar o impacto ambiental de tecnologias computacionais.

Assinale a opção que indica a métrica usualmente empregada para avaliar a eficiência energética de sistemas computacionais de alto desempenho.

(A) PFR/W.
(B) MIPS/W.
(C) FPS/W.
(D) (TB/s)/W.
(E) FLOPS/W.

34

Considere um computador com processador de 32 *bits* cuja memória está organizada de acordo com o endereçamento de *byte*, com o método *big-endian*.

Suponha que as palavras 25 38 94 67H (identificada por P1) e 18 31 72 46H (identificada por P2) estão armazenadas a partir dos endereços 0010H e 002CH, respectivamente.

Nesse contexto, analise as afirmativas a seguir.

I. A leitura do endereço 0014H retorna o byte 38H da palavra P1.
II. A leitura do endereço 002FH retorna o *byte* 46H da palavra P2.
III. Entre as palavras P1 e P2 podem ser armazenadas doze palavras.

Está correto o que se afirma em

(A) I, apenas.
(B) II, apenas.
(C) III, apenas.
(D) I e II, apenas.
(E) II e III, apenas.

35

Considerando o endereçamento de *byte* em arquitetura de computadores, e, os métodos *big-endian* e *little-endian* de ligação do processador à memória do computador, analise as afirmativas a seguir.

I. No método *big-endian*, o endereçamento inicia pelo byte menos significativo.
II. O acesso aos *bytes* de palavras armazenadas na memória independe do método de ligação (*big-endian* ou *little-endian)* para processadores de menos de 32 bits.
III. Em ambos os métodos de ligação (*big-endian* e *little-endian*), os *bits* menos significativos do barramento de dados estão ligados aos *bits* menos significativos da palavra armazenada na memória.

Está correto o que se afirma em

(A) I, apenas.
(B) III, apenas.
(C) I e II, apenas.
(D) II e III, apenas.
(E) I, II e III.

36

Para que um programa de computador possa ser executado, diversas etapas compõem seu desenvolvimento. Com relação ao ciclo de desenvolvimento de um programa, analise as afirmativas a seguir.

I. A etapa de compilação é responsável por gerar o arquivo executável do programa.
II. Na etapa de link-edição, módulos pré-compilados são ligados entre si.
III. Um programa compilado não causa erros de execução.

Está correto o que se afirma em

(A) I, apenas.
(B) II, apenas.
(C) III, apenas.
(D) I e II, apenas.
(E) II e III, apenas.

37

Com relação ao uso de um ambiente de programação integrado (IDE - *Integrated Development Environment*) no desenvolvimento de programas de computador, analise as afirmativas a seguir.

I. As etapas de edição e de compilação de um programa de computador podem ser integradas em um IDE.
II. O processo de link-edição de um programa não pode ser integrado em um IDE.
III. Um programa desenvolvido em um IDE não requer o processo de depuração.

Está correto o que se afirma em

(A) I, apenas.
(B) II, apenas.
(C) I e II, apenas.
(D) II e III, apenas.
(E) I, II e III.

38

Com relação ao processo de depuração de um programa de computador, analise as afirmativas a seguir.

I. Por meio da depuração, é possível investigar a ocorrência de erros no programa.
II. A depuração é realizada antes da compilação.
III. A depuração pode ser executada por meio de pontos de parada (*breakpoints*).

Está correto o que se afirma em

(A) I, apenas.
(B) II, apenas.
(C) I e II, apenas.
(D) I e III, apenas.
(E) II e III, apenas.

39

Uma linguagem de computador é uma notação que permite aos programadores expressarem instruções e algoritmos em um formato que pode ser entendido e executado por um computador. As linguagens de computador são projetadas para serem precisas, eficientes e expressivas o suficiente para descrever uma ampla gama de operações e processos computacionais.

Com relação à verificação de sintaxe do compilador, analise as afirmativas a seguir e assinale (V) para a verdadeira e (F) para a falsa.

( ) Tem como saída um arquivo executável que pode ser executado em um computador.
( ) Utiliza instruções chamadas *pseudo instruções*.
( ) Tem como saída um arquivo de objeto em linguagem de máquina ou um programa em linguagem *assembly.*
( ) É responsável por traduzir uma versão simbólica de instruções em sua versão binária.
( ) Uma de suas funções é a otimização do código.

As afirmativas são, respectivamente,

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

40

Leia o trecho a seguir.

Processo que envolve usar ferramentas ou técnicas que permitem identificar e corrigir erros de código, comportamentos inesperados ou falhas de execução em um ambiente separado ou externo ao ambiente de desenvolvimento original. É especialmente útil em sistemas complexos e distribuídos, onde pode ser difícil replicar exatamente as condições em que um erro ocorreu no ambiente de desenvolvimento.

O trecho apresenta o conceito de

(A) Logging.
(B) Depuração cruzada.
(C) Compilador.
(D) Debug.
(E) Análise estática de código.

41

A detecção e a correção de erros são fundamentais para garantir a integridade e a confiabilidade dos dados transmitidos.

Sobre a técnica de paridade no padrão RS232, assinale a afirmativa correta.

(A) É uma técnica de correção de erros.
(B) Pode detectar erros de mais de um *bit.*
(C) É uma técnica livre de sobrecarga de *bits.*
(D) Muito eficaz contra erros duplos.
(E) Pode ser acrescentado um *bit* de paridade par ou ímpar.

42

Checksum é um método utilizado para verificar a integridade de dados transmitidos em redes de computadores. Ele envolve a realização de cálculos matemáticos nos dados de forma a gerar um valor único que é adicionado aos dados antes de serem enviados. Quando os dados são recebidos, é calculado um novo *checksum* e comparado com o valor recebido. Se houver uma diferença, isso indica que os dados podem ter sido corrompidos durante a transmissão e podem precisar ser reenviados para garantir sua precisão.

Sobre esta técnica, assinale a afirmativa correta.

(A) Para obter o checksun, o remetente faz o OU exclusivo com as palavras do bloco de dados.
(B) Funciona medindo o número de *bits* que diferem entre dois códigos.
(C) Quanto maior a distância, maior a capacidade de detectar e corrigir erros em um código.
(D) Quando adicionado a cada palavra de memória, não é eficiente contra erros devidos a flutuações de tensão.
(E) Não é eficiente para detectar erros dentro de um setor de CD-ROM.

43

Sobre os códigos de detecção de erros, paridade, *checksum* e CRC, analise as afirmativas a seguir e assinale (V) para a verdadeira e (F) para a falsa.

( ) A técnica de CRC, ou *Cyclic Redundancy Check*, consiste da adição de número sequencial e um código de correção de erros chamado CRC, gerado a partir de um algoritmo que é aplicado aos dados do cabeçalho e do *payload*.
( ) Objetivam garantir a confiabilidade na transmissão e recebimento de dados.
( ) O *checksum* também recupera os dados transmitidos com erro.

As afirmativas são, respectivamente,

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

44

Leia o trecho a seguir.

É um mecanismo de software ou hardware que monitora constantemente o funcionamento de um sistema. Caso detecte que o sistema está em um estado inesperado ou travado, ele pode reiniciar o sistema automaticamente ou executar outra ação pré-determinada para garantir a continuidade operacional. Essa técnica é especialmente útil em sistemas críticos ou que precisam de alta disponibilidade, garantindo que o sistema permaneça em funcionamento mesmo em situações de falhas inesperadas.

Dentro do contexto de técnicas de tolerância a falhas, assinale a opção que apresenta a técnica discutida no texto acima.

(A) CRC.
(B) Redundância de *software*.
(C) Redundância de *hardware*.
(D) *Watchdog*.
(E) Distância de *Hamming*.

45

A técnica de "desenrolamento de *loop*" (*loop unrolling*) é uma técnica importante para obter mais desempenho de loops que acessam matrizes. Seu funcionamento envolve fazer múltiplas cópias do corpo do *loop* e executar o *loop* transformado menos vezes, o que reduz o *overhead* do *loop* e proporciona oportunidades para muitas outras otimizações.

Assinale a opção que indica a etapa da conversão de programas de alto nível para um programa executável em que esta técnica se realiza.

(A) Etapa de contrução (*builder*).
(B) Etapa de compilação (*compiler*).
(C) Etapa de montagem (*assembler*).
(D) Etapa de carga (*loader*).
(E) Etapa de vínculo (*linker*).

Realização