REVISÃO PARA A PROVA 02 – PROGRAMAÇÃO DE SISTEMAS
CURSO DE CIÊNCIA DA COMPUTAÇÃO
UNIVERSIDADE FRANCISCANA – UFN. 2025-02. Peso:2,0.

PROFESSOR: André F. dos Santos.
Nome do aluno:_____.
Data: ___/___/_____.

Destaque em amarelo a alternativa correta nas questões de múltipla escolha. Preencha seu nome e data.

Questões de múltipla escolha (marque apenas uma alternativa).

1) Em um sistema de programação modular, considere um programa dividido em diversos módulos independentes. Qual a principal vantagem dessa abordagem em relação à manutenção do código?
a) O código se torna mais rápido independentemente do hardware utilizado.
b) Cada módulo pode ser alterado ou corrigido isoladamente, sem comprometer o restante do sistema.
c) A modularidade garante que não existirão erros de lógica durante a execução.
d) Programas modulares nunca precisam ser recompilados ao sofrer alterações.
e) A modularidade elimina completamente o uso de variáveis globais.

2) Observe o trecho em Python:

```
x = 10
def funcao():
    x = 5
    print(x)
funcao()
print(x)
```

Qual será a saída e o motivo?
a) 5 e 5, porque variáveis locais sempre substituem as globais.
b) 10 e 10, porque a função não consegue alterar a variável local.
c) 5 e 10, porque a função cria uma variável local que não altera a global.
d) 10 e 5, porque a variável global prevalece em qualquer escopo.
e) Erro de execução, pois o Python não diferencia escopos.

3) Sobre a pilha de execução em linguagens como C e Python, assinale a alternativa correta:
a) Apenas linguagens compiladas utilizam pilha de execução.
b) A pilha armazena parâmetros, variáveis locais e endereço de retorno de cada função.
c) Funções recursivas não utilizam pilha, apenas variáveis globais.
d) A pilha de execução cresce em direção aos endereços de memória mais altos em todas as arquiteturas.
e) Após o retorno de uma função, todas as variáveis globais são removidas da memória.

4) Considere as funções em Python:

```
def altera_lista(l):
    l.append(99)

def altera_num(n):
    n = n + 1

a = [1,2,3]
b = 10
altera_lista(a)
altera_num(b)
print(a, b)
```

Qual será a saída?
a) [1,2,3,99] e 11
b) [1,2,3] e 10
c) [1,2,3,99] e 10
d) [1,2,3] e 11
e) [1,2,3,99,10]

5) Em Assembly, a instrução CALL salva o endereço de retorno na pilha. O que ocorre se o endereço de retorno for corrompido/sobrescrito?
a) A CPU corrige automaticamente e retorna ao local correto.
b) O fluxo pode saltar para endereço inválido, causando falha de segmentação ou comportamento indefinido.
c) A instrução RET é ignorada e a execução continua linearmente.
d) O compilador detecta e impede a execução antes do erro ocorrer.
e) A pilha não influencia o fluxo de controle da execução.

6) Sobre "stack overflow", assinale a alternativa correta:
a) É exclusivo de linguagens compiladas e não ocorre em interpretadas.
b) Pode ocorrer em recursões muito profundas ou loops de chamadas sem liberação de frames.
c) É exatamente o mesmo que esgotamento da heap (heap overflow).
d) Ocorre quando variáveis globais têm o mesmo nome em módulos diferentes.
e) Só acontece quando o endereço de retorno é maior que o topo da pilha.

7) Sobre variáveis locais vs. globais no desenho de funções e módulos:
a) Variáveis locais permanecem válidas após o término da função.
b) Variáveis globais reduzem o acoplamento e melhoram testabilidade.
c) Variáveis locais promovem encapsulamento e reduzem efeitos colaterais.
d) Variáveis globais são automaticamente removidas após cada chamada de função.
e) Locais e globais são equivalentes em termos de segurança e manutenção.

8) Em Python, ao executar uma função recursiva com profundidade muito alta, como fatorial(1000) sem ajustes, o comportamento típico é:
a) Concluir o cálculo normalmente, pois Python expande a pilha automaticamente.
b) Entrar em loop infinito sem erro.
c) Lançar RecursionError ao atingir o limite de profundidade.
d) Encerrar o processo pelo sistema operacional sem mensagem.
e) Retornar 1 automaticamente para evitar overflow.

9) Sobre passagem de parâmetros em registradores (convenções de chamada):
a) É geralmente mais lenta que passagem por pilha.
b) É limitada pela quantidade de registradores de propósito geral definidos pela ABI.
c) Não é utilizada em arquiteturas modernas.
d) Impede passagem por valor (apenas referências).
e) Elimina a necessidade de endereço de retorno.

10) Em chamadas recursivas, o papel da pilha de execução é:
a) Reutilizar o mesmo frame para todas as chamadas para economizar memória.
b) Armazenar, para cada chamada, seu próprio frame com parâmetros/variáveis e o endereço de retorno.
c) Promover todas as variáveis locais a globais para manter o estado entre chamadas.
d) Forçar o compilador a eliminar a recursão por otimização de cauda sempre.
e) Substituir a necessidade de caso base, já que a pilha controla a parada.

11) Explique a diferença entre variáveis locais e variáveis globais, citando vantagens e riscos de cada uma.

12) Descreva passo a passo o que acontece na pilha de execução quando uma função chama outra função.