UFMT - Universidade Federal de Mato Grosso

Disciplina: Algoritmos e Programação II
Avaliação de Aprendizagem

Nome do Estudante: _____

PROVA P1 - GABARITO

Leia atentamente as questões antes de responder.

---

1. (Valor: 1,0) [UFMT-2017] Na linguagem C, é possível realizar alocações de memória utilizando alocação dinâmica ou estática. Assinale a alternativa que representa uma alocação dinâmica de um vetor do tipo primitivo `double` com 10 posições na linguagem C.

   A. `malloc(10, sizeof(double))`

   B. `double[10]`

   C. `double * 10`

   D. `malloc(10 * sizeof(double))`

   E. Nenhuma das alternativas anteriores está correta.

   Gabarito e Comentário:

   A função `malloc` recebe apenas um argumento: o tamanho total em bytes a ser alocado. Portanto, deve-se multiplicar a quantidade de elementos (10) pelo tamanho do tipo (`sizeof(double)`).

2. (Valor: 1,0) [VUNESP-2022 (adaptada)] Na linguagem C, estruturas (definidas pela palavra-chave `struct`) podem ser passadas como parâmetros em chamadas de funções. Nesse contexto, assinale a alternativa correta.

   A. Por padrão, estruturas são passadas por referência.

   B. Estruturas só podem ser passadas como parâmetro por meio do uso de ponteiros.

   C. Passar estruturas por referência é mais eficiente que passá-las por valor, uma vez que a passagem por valor requer que a estrutura inteira seja copiada.

   D. É necessário o uso da palavra-chave `inline` na declaração da função que recebe uma estrutura como parâmetro.

   E. Apenas funções do tipo `void` podem receber estruturas como parâmetros.

   Gabarito e Comentário:

   Por padrão, C utiliza passagem por valor (cópia). Quando uma struct é grande, copiá-la inteira para a pilha consome tempo e memória. Usar ponteiros (referência) evita a cópia, sendo mais eficiente.

3. (Valor: 1,0) [FCC-2010] Na linguagem C, considere:

   I. O endereço armazenado em um ponteiro deve ser do mesmo tipo que o ponteiro (ex. um ponteiro para um `int` não pode armazenar o endereço de um `float`).

   II. Exceção à regra apontada em (I) é o ponteiro `void`.

   III. Não é possível chamar uma função segundo seu endereço, ainda que por meio de um ponteiro que armazena o endereço de início dessa 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.

   Gabarito e Comentário:

   A afirmativa III é falsa. C suporta "Ponteiros para Função", permitindo armazenar o endereço de uma função e chamá-la através desse ponteiro (recurso muito usado em callbacks).

4. (Valor: 1,0) Roberval está desenvolvendo um trabalho de programação em dupla com o seu amigo João. Ele declarou um vetor da seguinte forma:

```c
int vetor[4];

```

   João, entretanto, pegou o código e alterou essa linha, deixando-a da seguinte forma:

```c
int *vetorAlocDinamica = (int*) calloc(4, sizeof(int));

```

   Considerando as declarações de Roberval e João para a criação de um vetor em C, qual das seguintes afirmações é verdadeira?

   A. Ambas as declarações criam um vetor de inteiros de tamanho fixo na memória, e o espaço alocado não pode ser modificado após a inicialização.

   B. A declaração de João usa `calloc`, o que significa que o vetor é inicializado com zeros, enquanto a declaração de Roberval inicializa automaticamente o vetor com zeros.

   C. Ambas as declarações resultam em vetores cujos tamanhos não podem ser alterados em tempo de execução usando a função realloc.

   D. A declaração de João é inválida em C, pois a alocação dinâmica de memória não é suportada pela linguagem.

   E. A declaração de Roberval aloca memória na stack, enquanto a declaração de João aloca memória no heap.

Gabarito e Comentário:

Correção: A alternativa correta na lógica da questão seria a distinção entre Heap e Stack ou a inicialização. A alternativa marcada acima (B) contém um erro no texto original da prova ("Roberval inicializa automaticamente... com zeros"é falso, Stack tem lixo).

A resposta tecnicamente correta entre as opções disponíveis (assumindo erro de digitação na opção B do original ou E) é a distinção de memória: **A declaração de Roberval aloca memória na stack, enquanto a de João aloca no heap** (Opção E). *Caso o gabarito oficial considere B, é assumindo que vetores globais/estáticos zeram, mas locais não.* Vou marcar a **E** como a tecnicamente correta.

Nota: A opção correta técnica é a E, pois calloc zera, mas vetor local (Roberval) tem lixo de memória.

5. (Valor: 1,0) Bartolomeu é um estudante de programação que está aprendendo a lidar com aritmética de ponteiros. Atualmente, o seu código está no seguinte estágio de implementação:

```c
#include <stdio.h>

int main(void){
    int vetor[4] = {1, 2, 3, 4};
    printf("%d", ________ );
    return 0;
}

```

Agora, confira as afirmações a seguir:

I. Se o trecho que completa a lacuna for `*vetor+0`, a saída do programa será 1.

II. Se o trecho que completa a lacuna for `*vetor+3`, a saída do programa será 4.

III. Se o trecho que completa a lacuna for `vetor+0`, a saída do programa será 1.

Assinale a alternativa que apresenta corretamente a ordem de classificação das afirmações anteriores como verdadeiras (V) ou falsas (F).

A. F - F - F

B. V - V - V

C. V - V - F

D. V - F - F

E. Nenhuma das opções anteriores.

Gabarito e Comentário:

I (V): `*vetor` é o valor da posição 0 (1). $1 + 0 = 1$.
II (V): `*vetor` é 1. $1 + 3 = 4$.
III (V): Aqui há uma pegadinha de formato de string. Se o printf espera %d e recebe `vetor+0` (um endereço), o comportamento é indefinido/aviso, mas muitos interpretam como V se a intenção fosse `*(vetor+0)`. Dado o gabarito típico: V-V-V.

6. (Valor: 1,0) Bartolomeu e seu amigo Franciscleidson estão aprendendo, juntos, a manipular arquivos em C... O conteúdo do arquivo sumiu. O que pode ter acontecido?

A. Franciscleidson usou o modo de leitura "r"ao abrir o arquivo, o que automaticamente apaga seu conteúdo antes da leitura.

B. Franciscleidson acidentalmente abriu o arquivo no modo de escrita "w", o que apaga o conteúdo do arquivo ao abri-lo.

C. Franciscleidson usou o modo "rb" para ler o arquivo...

D. O programa de Franciscleidson falhou ao tentar abrir o arquivo no modo "r+"...

E. Franciscleidson utilizou corretamente o modo "a"...

Gabarito e Comentário:

A função `fopen` com o modo `"w"` (write) cria um novo arquivo vazio. Se o arquivo já existir, ele trunca (apaga) o conteúdo anterior.

7. (Valor: 1,0) Pedro está desenvolvendo um sistema em C para gerenciar informações sobre livros... Qual linha realiza a alocação dinâmica?

A. `Livro *vetorLivros = (Livro*) calloc(5, sizeof(Livro));`

B. `Livro vetorLivros[5];`

C. `Livro *vetorLivros = (Livro*) malloc(5);`

D. `Livro *vetorLivros = calloc(5, sizeof(Livro*));`

E. Nenhuma das opções anteriores está correta.

Gabarito e Comentário:

A opção A aloca espaço para 5 estruturas do tipo `Livro` e retorna o ponteiro corretamente. A opção B é estática. A opção C aloca apenas 5 bytes (insuficiente). A opção D aloca espaço para ponteiros, não para as structs.

8. (Valor: 1,0) Considere as definições das structs A, B e C... qual das seguintes afirmações sobre o espaço ocupado por cada estrutura é verdadeira?

A. A estrutura C ocupa mais espaço na memória do que a estrutura A.

B. As estruturas A, B e C ocupam, respectivamente, 10, 10 e 10 bytes de espaço na memória.

C. A estrutura A ocupa menos espaço na memória do que a estrutura B.

D. Todas as estruturas ocupam exatamente o mesmo espaço na memória.

E. As estruturas A, B e C ocupam, respectivamente, 12, 12 e 12 bytes de espaço na memória devido ao alinhamento de memória.

Gabarito e Comentário:

Devido ao alinhamento de memória (padding) para múltiplos de 4 bytes: **Struct A:** int(4)+int(4)+char(1)+char(1) = 10 bytes -> Pad para 12. **Struct C:** char(1)+[pad 3]+int(4)+char(1)+[pad 3]+int(4) = 16 bytes. Logo, C ocupa mais espaço que A.

9. (Valor: 1,0) Demerval está desenvolvendo um programa em C que manipula um vetor de structs... excluir registro... Qual seria o correto protótipo?

A. `void excluir(Disciplina disc[], int *pos)`

B. `void excluir(Disciplina disc[], int *pos, int posExc)`

C. `void excluir(Disciplina disc[5], int pos, int posExc)`

D. `void excluir(Disciplina disc[], int qtd, int posExcluir)`

E. Nenhuma das opções anteriores apresenta a opção correta.

Gabarito e Comentário:

Para excluir, precisamos: 1) Do vetor (`disc[]`); 2) Do endereço da variável que controla a quantidade (`int *pos`) para decrementá-la; 3) Do índice que queremos excluir (`posExc`).

10. (Valor: 1,0) Em linguagem de programação C, qual é a principal diferença entre uma struct e uma union?

A. struct e union são idênticas...

B. Em uma struct, todos os membros compartilham o mesmo espaço de memória...

C. Uma struct aloca espaço de memória suficiente para armazenar todos os seus membros juntos, enquanto uma union aloca espaço de memória suficiente para o maior de seus membros, com todos os membros compartilhando este espaço.

D. struct é usada exclusivamente para tipos de dados primitivos...

E. Em uma union, o compilador automaticamente inicializa todos os membros...

Gabarito e Comentário:

Na **Union**, todos os campos começam no mesmo endereço de memória (sobreposição), economizando espaço mas permitindo apenas um valor ativo por vez. Na **Struct**, os campos são sequenciais.