Universidade de Brasília — Faculdade UnB Gama

Estruturas de Dados 1 - Prova Teórica 1

Nome:

Matrícula: Data:

Observações:

(A) A prova é individual e sem consulta, sendo vedado o uso de calculadoras e de telefones celulares.

(B) A interpretação dos comandos das questões faz parte da avaliação.

(C) Nas questões de múltipla escolha, apenas **uma** alternativa deve ser escolhida. Questões com **múltiplas marcações** ou **sem marcação** serão desconsideradas.

---

1. Considere o código abaixo:

```
1 x = '0';
```

A constante atribuída à variável `x` é do tipo

(A) `int`
(X) `char`
(C) `float`
(D) `double`

2. Considere o código abaixo:

```
1 int x = 7 % 3;
```

Após a atribuição, a variável `x` tem valor igual a

(X) 1
(B) 2
(C) 3
(D) 4

3. Considere a condicional a seguir:

```
1 if (!x || y)
2     z = x - y;
3 else
4     z = x + y;
```

Se antes da execução da condicional temos `x = 1` e `y = 0`, após a condicional a variável `z` terá valor igual a

(A) -1
(B) 0
(X) 1
(D) 2

4. Considere o código abaixo:

```
1 for (int i = 0; i < 100; i++) {
2     if (i % 3 == 0)
3         continue;
4
5     soma += i;
6
7     if (soma > 25)
8         break;
9 }
```

Após a sua execução do laço, o valor da variável `soma` é igual a

(X) 27
(B) 30
(C) 36
(D) 38

5. Considere a atribuição abaixo:

```
1 int x = 0 ? (1 ? 1 : 2) : (1 ? 3 : 4)
```

O valor de x, após a atribuição, é igual a

(A) 1
(B) 2
(X) 3
(D) 4

6. Considere a união Comando, definida abaixo:

```
1 typedef union _Comando {
2     short handle;
3     char name[6];
4     int parameters[4];
5 } Comando;
```

Considerando que um int, um char e um short ocupem 4, 1 e 2 *bytes*, respectivamente, qual é o tamanho em *bytes* de uma variável do tipo Comando em memória?

(A) 8
(X) 16
(C) 24
(D) 32

7. Considere o trecho de código abaixo:

```
1 int v[] = {1, 0, 1, 0, 1};
2 int *p = v;
3 p++;
4 int **q = &p;
```

Considere que um int ocupe 4 *bytes* de memória e que os endereços de memória do vetor v e do ponteiro p sejam 0xABE2 e 0xCDF4, respectivamente. Ao final da execução, o ponteiro q aponta para o endereço de memória

(A) 0xABE2
(B) 0xABE6
(C) 0xCDF4
(X) 0xCDF8

8. Considere a declaração de variáveis a seguir:

```
1 int matrix[][3] = {
2     {1, 2, 3}, {4, 5, 6}, {7, 8, 9}
3 };
4
5 int (*p)[3] = matrix;
6
7 printf("%d\n", *(*(p + 2) + 2));
```

O resultado da impressão da linha 7 é igual a

(X) 9
(B) 8
(C) 5
(D) 4

9. Considere o código a seguir:

```
1 typedef struct _Point {
2     int x;
3     int y;
4     int z;
5 } Point;
6
7 Point *p = NULL;
```

Considerando uma arquitetura de 16 *bits* (isto é, um int ocupa 2 *bytes* em memória), o valor de sizeof(p) é igual a

(A) 8
(B) 6
(C) 4
(X) 2

10. Considere a função conta(), implementada abaixo:

```
1 float conta(int minutos, float vpm)
2 {
3     return minutos * vpm;
4 }
```

A declaração correta de um ponteiro para a função conta() é

(A) float *p()
(B) float *p(int, float)
(C) float *p(minutos, vpm)
(X) float (*p)(int, float)

11. Em relação à manipulação de arquivos em linguagem C, a função fseek()

(A) procura pela string indicada no arquivo
(B) retorna o número de strings no arquivo
(C) indica a posição do cursor de leitura
(X) posiciona o cursor de leitura

12. Para abrir um arquivo com permissão para leitura, o segundo parâmetro da função fopen() deve ser igual a

(A) "w"
(X) "r"
(C) "b"
(D) "a"

13. (4 pontos) A função `words()` declarada abaixo retorna o número de palavras no arquivo `in`:

```
int words(const char *in);
```

Assuma que, caso exista, o arquivo `in` contenha apenas caracteres alfabéticos, maiúsculos e minúsculos, e espaços em branco. Uma palavra é uma sequência contígua de um ou mais caracteres alfabéticos.

Implemente, em C ou C++, a função `words()`. O código abaixo ilustra o uso da função e seus respectivos retornos:

```
1 #include <stdio.h>
2 #include "words.h"                    // Contém a declaração da função words()
3
4 int main()
5 {
6     int x = words("one.txt");         // x = 1
7     int y = words("two.txt");         // y = 2
8     int z = words("five.txt");        // z = 5
9
10    return 0;
11 }
```

Abaixo estão os conteúdos dos arquivos citados no exemplo. O número 1 à esquerda refere-se à linha e é meramente ilustrativo, não fazendo parte do conteúdo do arquivo.

Arquivo `one.txt`

```
1 abcde
```

Arquivo `two.txt`

```
1 A a
```

Arquivo `five.txt`

```
1 Exemplo de arquivo de texto
```

Use a próxima folha e siga as linhas conforme a numeração indicada. Escreva com letra legível, de preferência em letras de forma, e utilize um lápis. A implementação deve começar com a assinatura da função e deve terminar com o fim do bloco da função. Não é necessário incluir os cabeçalhos `stdio.h` e `fstream`.

O código deve ter, no máximo, 35 linhas. Trate os possíveis erros que possam ocorrer retornando, para cada erro, um número negativo. Não use o mesmo número para dois erros distintos.

Solução: Implementação em C:

```
1 #include <stdio.h>
2
3 int words(const char *in)
4 {
5     FILE *f = NULL;
6
7     if (!in)
8         return -1;
9
10    f = fopen(in, "r");
11
12    if (!f)
13        return -2;
14
15    int total = 0;
16    char word[4096];
17
18    while (fscanf(f, "%s", word) == 1)
19        total += 1;
20
```

```
    fclose(f);

    return total;
}
```

Implementação em C, sem `fscanf()`:

```c
#include <stdio.h>

int words(const char *in)
{
    FILE *f = NULL;

    if (!in)
        return -1;

    f = fopen(in, "r");

    if (!f)
        return -2;

    int total = 0, lidos = 0;
    char c;

    do {
        c = fgetc(f);

        if (('a' <= c && c <= 'z') || ('A' <= c && c <= 'Z'))
            ++lidos;
        else {
            if (lidos)
                total += 1;
            lidos = 0;
        }
    } while (c != EOF);

    fclose(f);

    return total;
}
```

Implementação em C++:

```cpp
#include <fstream>

using namespace std;

int words(const char *in)
{
    if (!in)
        return -1;

    ifstrem f(in);

    if (!f)
        return -2;

    int total = 0;
    string word;

    while (cin >> word)
        ++total;

    f.close();
```

```
22 }
```