

Prova 3

Algoritmos e Estruturas de Dados I

Professor: Pedro O.S. Vaz de Melo

31 de dezembro de 2016 (valor: 30 pontos)

Nome: _____

escrevendo o meu nome eu juro que seguirei o código de honra

Código de Honra para este exame:

- Não darei ajuda a outros colegas durante os exames, nem lhes pedirei ajuda;
- não copiarei nem deixarei que um colega copie de mim;
- não usarei no exame elementos de consulta não autorizados.

Referências:

Função/Operador	Descrição	Exemplo
<code>void* malloc (size_t size);</code>	aloca um bloco de memória de tamanho <code>size</code> , retornando um ponteiro para o início do bloco.	<code>int *p1 = (int*)malloc(sizeof(int));</code>
<code>FILE* fopen(const char *filename, const char *mode)</code>	abre o arquivo <code>filename</code> no modo <code>mode</code>	<code>FILE *temp = fopen("temp.txt", "w");</code>
<code>int fscanf(FILE *arq, const char *format, &variáveis);</code>	lê dados numéricos do arquivo <code>arq</code>	<code>fscanf(arq, "%f", &nota1);</code>
<code>int fprintf(FILE *arq, const char *format, vars);</code>	escreve dados no arquivo <code>arq</code>	<code>fprintf (arq, "valor de aux: %d", aux);</code>
<code>int fclose (FILE * arq)</code>	fecha o arquivo <code>arq</code>	<code>fclose(arq);</code>

1. (5 points) Escreva uma função RECURSIVA que receba por parâmetro dois valores inteiros x e y e retorne o resultado de x^y . Sua função não pode usar *loops* (`for`, `while`, `etc`) nem a função `pow`. A função deve ter o seguinte protótipo:

```
int potencia(int x, int y);
```

2. (5 points) Escreva uma função RECURSIVA que receba por parâmetro um ponteiro para uma *string* e a imprime invertida. Se a sua função receber um ponteiro para a *string* “bola”, a sua função deve imprimir “alob”. Sua função não pode usar *loops* (`for`, `while`, `etc`). Dica: lembre-se de uma propriedade de *strings* que vetores de caracteres não necessariamente possuem. A função deve ter o seguinte protótipo:

```
void strInv(char *str);
```

3. (5 points)

Escreva um procedimento RECURSIVO de protótipo `void imp2Cont(int i, int n)` que imprime a contagem crescente e, logo depois, a contagem decrescente de i até n . Sua função não pode usar *loops* (`for`, `while`, `etc`). Exemplo: `imp2Cont(2,5)` imprime

```
2
3
4
5
4
3
2
```

4. (5 points) Escreva um programa para criar um arquivo de nome `asc2.txt` e escrever neste arquivo a tabela ASCII, do código 0 ao código 127, contendo o código do caractere (em notação decimal) e o caractere em si. Abaixo uma parte do arquivo que você deve gerar a partir do seu programa:

```
63 ?
64 @
65 A
66 B
67 C
```

5. (10 points) Preencha o código abaixo de função que lê uma matriz quadrada simétrica de um arquivo e a retorna em memória alocada dinamicamente. Lembre-se que em uma matriz simétrica $M[i][j] = M[j][i]$ e que em uma matriz quadrada o número de linhas é igual ao número de colunas.

```
double ** le_matriz_simetrica(char *nome_arquivo, int tamanho)
{
    int i, j;
    FILE *fd = fopen(_____);
    if(!fd) abort(); //fecha o programa

    double **M = _____;
    if(M==NULL) abort(); //fecha o programa

    for(i = 0; _____; _____) {

        M[i] = _____;
        if(M==NULL) abort(); //fecha o programa

        for(j = 0; _____; _____) {

            fscanf(_____);
        }
    }

    _____;

    _____;
}
```

O parâmetro `nome_arquivo` indica o nome do arquivo com os dados na matriz e `tamanho` indica o número de linhas na matriz.

Como a matriz é simétrica, o arquivo de entrada contém apenas a metade inferior da matriz. Em outras palavras, o arquivo contém apenas os elementos $M[i][j]$ onde $i \geq j$. Um exemplo de arquivo de entrada para `tamanho = 4` é dado abaixo:

```
1.2
2.4 1.3
8.2 9.1 6.2
8.3 6.5 8.2 1.0
```

Importante: sua função deve alocar a menor quantidade possível de memória para armazenamento da matriz. Em particular, sua função deve alocar espaço apenas para elementos $M[i][j]$ onde $i > j$, da mesma forma como o arquivo de entrada armazena apenas a metade inferior da matriz.