Programação II – IC/UFF, Prof. Raphael Machado
Lista de Exercícios

Exercícios de Revisão (Exercícios básicos de programação em Linguagem C)

1. Faça um programa que leia um número inteiro e exiba no monitor o seu antecedente (inteiro anterior) e o seu sucessor (inteiro posterior).

2. Faça um programa que leia 3 números reais e exiba no monitor a média aritmética dos mesmos.

3. Faça um programa que leia dois números inteiros representando, respectivamente, o valor das horas e dos minutos, e exiba no monitor quantos minutos se passaram desde o início do dia.

4. Faça um programa que leia 2 números inteiros e exiba no monitor uma mensagem dizendo se eles são iguais ou diferentes.

5. Faça um programa que leia 2 números inteiros e exiba no monitor o maior deles. Assuma que os números digitados são diferentes.

6. Faça um programa que leia um número inteiro e exiba no monitor uma mensagem dizendo se ele é par ou ímpar.

7. Faça um programa que leia 2 números inteiros e os exiba no monitor em ordem crescente.

8. Faça um programa que exiba no monitor os números inteiros de 1 a 100.

9. Faça um programa que exiba no monitor os números inteiros de 23 a 578.

10. Faça um programa que leia um número da entrada e exiba no monitor os números inteiros de 0 até este número.

11. Faça um programa que leia um número inteiro e exiba no monitor os números pares entre 2 e o número lido.

12. Faça um programa que leia um número inteiro e exiba no monitor os números pares entre o número lido e 100.

13. Faça um programa que leia um número inteiro (limite), um incremento (incr, inteiro) e exiba no monitor os números inteiros de 0 até limite, com incremento de incr. Suponha que incr seja maior do que zero.

Questões de Prova

Questões básicas de programação em Linguagem C

Questão. Explique o que o programa abaixo faz. Quais as saídas do programa para a entrada `4<enter>`? E para a entrada `5<enter>`?

```
#include <stdio.h>
int main(){
  int r, i, j, n= 1;
  scanf("%d",&r);
  for(i=r; i >= 1; i--){
    for(j=1; j <= i; ++j)
      printf("%d ",r+j-i);
    printf("\n");
  }
  return 0;
}
```

Questão. O que o programa abaixo faz? Qual a saída do programa abaixo para a entrada `4 6 <enter>`? E qual a saída para a entrada `2 3 <enter>`? E qual a saída para a entrada `5 7 <enter>`?

```
#include <stdio.h>
int main(){
  int n1, n2, M, cond=1;
  printf("Digite dois inteiros: ");
  scanf("%d %d", &n1, &n2);
  if (n1<n2) M = n1;
  else M = n2;
  while(cond){
    if( n1%M==0 && n2%M==0 ){
      printf("Resposta: %d.\n",M);
      cond=0;
      }
      M--;
    }
  return 0;
}
```

Questão. Escreva um programa em C que imprima um "triângulo" numérico com as seguintes propriedades:
- o triângulo é um conjunto de linhas impressas pelo seu programa onde cada linha tem um número a mais que a linha imediatamente inferior;
- o a quantidade de números na primeira linha é igual ao total de linhas e deve ser lida a partir da entrada de usuário;
- o último elemento impresso pelo seu programa deve ser 1 e será o único elemento da última linha;
- os elementos apresentam incrementos unitários da esquerda para a direita e de baixo para cima;
- não há necessidade de ajustar formatação em caso de números com mais de um algarismo.

Veja, abaixo exemplos de execução:

```
Air-de-Raphael:!!Prova1 raphaelmachado$ ./a.out
5
5 6 7 8 9
4 5 6 7
3 4 5
2 3
1
Air-de-Raphael:!!Prova1 raphaelmachado$ ./a.out
10
10 11 12 13 14 15 16 17 18 19
9 10 11 12 13 14 15 16 17
8 9 10 11 12 13 14 15
7 8 9 10 11 12 13
6 7 8 9 10 11
5 6 7 8 9
4 5 6 7
3 4 5
2 3
1
Air-de-Raphael:!!Prova1 raphaelmachado$
```

Questões envolvendo Funções e Ponteiros.

Questão. Escreva o código da função troca que deverá trocar os conteúdos das variáveis a e b, no programa em C abaixo:

```
#include <stdio.h>

// AQUI ENTRA O SEU CÓDIGO PARA A FUNÇÃO troca

int main(){
int a = 1, b = 2;
printf("a,b: %d,%d\n", a, b);
troca(&a, &b);
printf("a,b: %d,%d\n", a, b);
}
```

A saída da execução do seu programa deve ser:

```
$ ./a.out
a,b: 1,2
a,b: 2,1
```

Questão. Complete o código do programa nos dois lugares indicados, de modo que ele receba, como entrada, uma palavra e realize uma rotação da palavra, conforme indicado mais abaixo.

Código do programa:

```
#include <stdio.h>

int calculaTamanho(char*palavra){
  int tam=0;
  // COMPLETAR AQUI
  return tam;
}

void rotPalavra(char*palavra){
  int tamanho;
  tamanho = calculaTamanho(palavra);
  palavra[tamanho]=palavra[0];
  //COMPLETAR AQUI
  palavra[tamanho]=0;
}

int main(){
  char palavra[50];
  int tamanho;

  scanf("%s",palavra);
  rotPalavra(palavra);
  printf("%s\n",palavra);
}
```

A saída da execução do seu programa deve ser:

```
$ ./a.out
RaphaelMachado
aphaelMachadoR
```

Dicas:

- Cada local pode ser completado com menos que 100 (cem) caracteres de código.
- O caractere “nulo” – indicado pelo zero – representa o fim da string.

Questão. Qual a saída impressa pelo programa a seguir?

```
#include<stdio.h>

int main() {
  int a, b;
  int *p, *q;
  b = 33;
  p  =  &a;
  q = p;
  p  =  &b;
  *q  =  11;
  *p  =  39;
  *p = a*b;
  printf("a = %d\n", a);
  printf("b = %d\n", b);
  return 0;
}
```

Questão. Explique o que o programa abaixo faz. Qual a saída do programa para a entrada `1 <enter> 1 <enter>`. E para a entrada `2 <enter> 3 <enter> 4 <enter>` ?

```
#include <stdio.h>
#include <stdlib.h>
int main()
{
    int n, i, *ptr, sum = 0;
    scanf("%d", &n);
    ptr = (int*) calloc(n, sizeof(int));


    for(i = 0; i < n; ++i)
    {
        scanf("%d", ptr + i);
        sum += *(ptr + i);
    }
    printf("%d", sum);

    free(ptr);
    return 0;
}
```

Questão. Em relação ao anterior, qual seria o comando `malloc` a ser usado usado no lugar de `ptr = (int*) calloc(n, sizeof(int));` ?

Questões envolvendo Vetores

Questão. Escreva um programa em C que faça o seguinte:

1) leia 10 números inteiros,
2) armazene estes números em um vetor,
3) calcule a média dos valores do vetor,
4) divida cada elemento do vetor pela média calculada na etapa 3, e
5) imprima cada um dos elementos, separados por vírgula, e com ponto final após o último elemento (exemplo: “2,5,4,7,9.”).

Questão. Escreva um programa em C que faça o seguinte:

1) leia 5 números do tipo float,
2) armazene estes números em um vetor,
3) identifique o maior dos números vetor,
4) divida cada elemento do vetor pelo valor máximo encontrado na etapa 3, e
5) imprima cada um dos elementos, entre parêntesis e separados por vírgula, conforme o exemplo: (0.2,0.3,0.4,0.1,1).

Questões envolvendo alocação dinâmica.

Questão. Escreva um programa em C que faça o seguinte:

6) Receba do usuário um número inteiro positivo que será armazenado na variável `tamanho`.
7) Encerre o programa caso o número digitado pelo usuário seja menor que ou igual a zero.
8) Aloque dinamicamente "espaços de memória" que serão acessados por meio de variáveis `entrada` e `saida`.
9) Leia (entrada do usuário) `tamanho` números `int` e armazene no espaço locado dinamicamente e acessível por meio da variável `entrada`.
10) Imprima os números armazenados no espaço acessível por meio de `entrada`.
11) Preencha cada posição do espaço locado dinamicamente e acessível por meio da variável `saida` da seguinte forma: o valor de cada posição será `1`, caso o número na posição correspondente do espaço acessível a partir de `entrada` seja par, e será `0`, caso contrário.
12) Imprima os números acessíveis por meio de `saida`.

O seu programa deve usar uma função `Paridade` que recebe como entrada um valor `int` e retorna `1`, caso esse valor seja par, e `0`, caso contrário.
Seu programa deve se comportar conforme apresentado abaixo.

```
MacBook-Air-de-Raphael:Prova 1 raphaelmachado$ ./a.out
Entre com o número para a posição 0.
34
Entre com o número para a posição 1.
27
Entre com o número para a posição 2.
9
Entre com o número para a posição 3.
100
Entre com o número para a posição 4.
1
Números da sua lista: 34, 27, 9, 100, 1.
Lista de Paridades: 1, 0, 0, 1, 0.
MacBook-Air-de-Raphael:Prova 1 raphaelmachado$
```

Exercício. Escreva um programa em C que faça o seguinte:

1. Instancie dois vetores `entrada` e `saida`, cada um com `TAM` posições de `int`, onde `TAM` é uma constante (definida com `#define`) de valor `5`.
2. Leia (entrada do usuário) `TAM` números `int` e armazene em `entrada`.
3. Imprima os números armazenados em `entrada`.
4. Armazene em cada posição de `saida` o valor `1`, se o número da posição correspondente em `entrada` for par, e o valor `0`, caso contrário.
5. Imprima os números armazenados em `saida`.

O seu programa deve usar uma função `Paridade` que recebe como entrada um valor `int` e retorna `1`, caso esse valor seja par, e `0`, caso contrário.

Seu programa deve se comportar conforme apresentado abaixo.

```
MacBook-Air-de-Raphael:Prova 1 raphaelmachado$ ./a.out
Entre com o número para a posição 0.
34
Entre com o número para a posição 1.
27
Entre com o número para a posição 2.
9
Entre com o número para a posição 3.
100
Entre com o número para a posição 4.
1
Números da sua lista: 34, 27, 9, 100, 1.
Lista de Paridades: 1, 0, 0, 1, 0.
MacBook-Air-de-Raphael:Prova 1 raphaelmachado$
```

Questões envolvendo Modularização e Tipos de Dados Definidos pelo Usuário

Questão. Explique por que, no programa abaixo, `struct fracao` não pode ser considerado um Tipo Abstrato de Dados. Como você resolveria a "violação à abstração de dados" no código abaixo (explique ou escreva código).

```
#include<stdio.h>

#define NUM 0
#define DEN 1

struct fracao{
  int valor[2];
};

int main(void){
  struct fracao f;
  printf("Digite o numerador (inteiro): ");
  scanf("%d",&f.valor[NUM]);
  printf("Digite o denominador (inteiro): ");
  scanf("%d",&f.valor[DEN]);
  printf("\n Sua fração é %d/%d\n\n",f.valor[NUM],f.valor[DEN]);
}
```

Questão. Considere os códigos dos arquivos `main.c`, `matriz.h` e `matriz.c` apresentados abaixo:

main.c

```
#include <stdio.h>
#include "matriz.h"

int main() {
	char op;

	printf("Digite a operacao que deseja realizar:\n");
	printf("  C - Multiplicação de uma matriz por uma constante\n");
	printf("  A - Adicao de matrizes\n");
	printf("  U - Subtracao de matrizes\n");
	printf("  M - Multiplicacao de matrizes\n");
	printf("  T - Transposta de uma matriz\n");
	printf("  S - Checar se uma matriz e simetrica\n");
	printf("  Q - Checar se uma matriz e quadrada\n");
	printf("  E - Checar se uma matriz e esparsa\n");
	scanf("%c", &op);

	// Multiplicação de uma matriz por uma constante
	if (op == 'C') {
		// leitura da matriz
		Matriz A;
		printf("Digite as informações da matriz:\n");
		leMatriz(&A);
		// leitura da constante
		int c;
		printf("Digite a constante:\n");
		scanf("%d", &c);
		// calculo da multiplicacao
		Matriz B;
		B = MultiplicacaoConstante(A, c);
		// impressao do resultado
		printf("O resultado e:\n");
		imprimeMatriz(B);
	}

	// Adicao de matrizes
	if (op == 'A') {
		// leitura da primeira matriz
		Matriz A;
		printf("Digite as informações da primeira matriz:\n");
		leMatriz(&A);
		// leitura da segunda matriz
		Matriz B;
		printf("Digite as informações da segunda matriz:\n");
		leMatriz(&B);
		// calculo da adicao
		Matriz C;
		C = AdicaoMatriz(A, B);
		// impressao do resultado
		printf("O resultado e:\n");
		imprimeMatriz(C);
	}

	// Subtracao de matrizes
	if (op == 'U') {
		// leitura da primeira matriz
		Matriz A;
		printf("Digite as informações da primeira matriz:\n");
		leMatriz(&A);
		// leitura da segunda matriz
		Matriz B;
		printf("Digite as informações da segunda matriz:\n");
		leMatriz(&B);
		// calculo da subtracao
		Matriz C;
```

```
        C = SubtracaoMatriz(A, B);
        // impressao do resultado
        printf("O resultado e:\n");
        imprimeMatriz(C);
}

// Multiplicacao de matrizes
if (op == 'M') {
        // leitura da primeira matriz
        Matriz A;
        printf("Digite as informações da primeira matriz:\n");
        leMatriz(&A);
        // leitura da segunda matriz
        Matriz B;
        printf("Digite as informações da segunda matriz:\n");
        leMatriz(&B);
        // calculo da multiplicacao
        Matriz C;
        C = MultiplicacaoMatriz(A, B);
        // impressao do resultado
        printf("O resultado e:\n");
        imprimeMatriz(C);
}

// Transposta da matriz
if (op == 'T') {
        // leitura da matriz
        Matriz A;
        printf("Digite as informações da matriz:\n");
        leMatriz(&A);
        // calculo da transposta
        Matriz B;
        B = MatrizTransposta(A);
        // impressao do resultado
        printf("O resultado e:\n");
        imprimeMatriz(B);
}

// Checa de matriz é simétrica
if (op == 'S') {
        // leitura da matriz
        Matriz A;
        printf("Digite as informações da matriz:\n");
        leMatriz(&A);
        // calculo da transposta
        if (MatrizSimetrica(A)) printf("A matriz e simetrica\n");
        else printf("A matriz nao e simetrica\n");
}

// Checa de matriz é quadrada
if (op == 'Q') {
        // leitura da matriz
        Matriz A;
        printf("Digite as informações da matriz:\n");
        leMatriz(&A);
        // calculo da quadrada
        if (MatrizQuadrada(A)) printf("A matriz e quadrada\n");
        else printf("A matriz nao e quadrada\n");
}

// Checa de matriz é esparsa
if (op == 'E') {
        // leitura da matriz
        Matriz A;
        printf("Digite as informações da matriz:\n");
        leMatriz(&A);
```

```
                // calculo da esparsa
                if (MatrizEsparsa(A)) printf("A matriz e esparsa\n");
                else printf("A matriz nao e esparsa\n");
        }

        return 0;
}
```

matriz.h

```
#include <stdio.h>
#include <stdlib.h>

typedef struct matriz {
        int n;
        int m;
        int mat[100][100];
} Matriz;

// leitura da matriz
void leMatriz(Matriz *A);
// imprime uma matriz
void imprimeMatriz(Matriz A);
// calcula a multiplicacao de uma matriz por uma constante
Matriz MultiplicacaoConstante(Matriz A, int c);
// calcula a adicao de duas matrizes
Matriz AdicaoMatriz(Matriz A, Matriz B);
// calcula a subtracao de duas matrizes
Matriz SubtracaoMatriz(Matriz A, Matriz B);
// calcula a multiplicacao de duas matrizes
Matriz MultiplicacaoMatriz(Matriz A, Matriz B);
// calcula a transposta de uma matriz
Matriz MatrizTransposta(Matriz A);
// checa se uma matriz é simétrica
int MatrizSimetrica(Matriz A);
// checa se uma matriz é quadrada
int MatrizQuadrada(Matriz A);
// checa se uma matriz é esparsa
int MatrizEsparsa(Matriz A);
```

matriz.c

```
#include "matriz.h"

// leitura da matriz
void leMatriz(Matriz *A) {
        /* COMPLETE AQUI COM SEU CODIGO */
}

// imprime uma matriz
void imprimeMatriz(Matriz A) {
        /* COMPLETE AQUI COM SEU CODIGO */
}

// calcula a multiplicacao de uma matriz por uma constante
Matriz MultiplicacaoConstante(Matriz A, int c) {
        /* COMPLETE AQUI COM SEU CODIGO */
}

// calcula a adicao de duas matrizes
Matriz AdicaoMatriz(Matriz A, Matriz B) {
        /* COMPLETE AQUI COM SEU CODIGO */
}

// calcula a subtracao de duas matrizes
Matriz SubtracaoMatriz(Matriz A, Matriz B) {
```

```
	/* COMPLETE AQUI COM SEU CODIGO */
}

// calcula a multiplicacao de duas matrizes
Matriz MultiplicacaoMatriz(Matriz A, Matriz B) {
	/* COMPLETE AQUI COM SEU CODIGO */
}

// calcula a transposta de uma matriz
Matriz MatrizTransposta(Matriz A) {
	/* COMPLETE AQUI COM SEU CODIGO */
}

// checa se uma matriz é simétrica
int MatrizSimetrica(Matriz A) {
	/* COMPLETE AQUI COM SEU CODIGO */
}

// checa se uma matriz é quadrada
int MatrizQuadrada(Matriz A) {
	/* COMPLETE AQUI COM SEU CODIGO */
}

// checa se uma matriz é esparsa
int MatrizEsparsa(Matriz A) {
	/* COMPLETE AQUI COM SEU CODIGO */
}
```

Sua tarefa será implementar os algoritmos das funções `leMatriz` e `imprimeMatriz` e mais uma função à escolha dentre as funções declaradas em `matriz.h`. Você deverá escrever os códigos correspondentes dos arquivos `main.c`, `matriz.c` e `matriz.h`.

Observe que o registro `struct matriz` em `matriz.h` está preenchido – seus algoritmos devem ser consistentes com a estrutura de dados proposta.