Universidade de Itaúna - Ciência da Computação
Laboratório de Algoritmos e Estruturas de Dados I
Professor: Thiago Silva Vilela
Avaliação Final – Valor: 30 pontos

Instruções:

- **Resolva as questões na folha de papel almaço, devidamente enumeradas.**
- **A interpretação da prova é de responsabilidade do aluno. Caso necessário, escreva na solução da questão as considerações feitas.**
- **As questões não precisam ser feitas em ordem, desde que estejam claramente enumeradas no papel almaço.**

1) (3 pontos) Considere o programa em C abaixo:

```
#include <stdio.h>
#include <stdlib.h>
#define MUL(x, y) x * y

int main(int argc, char *argv[]) {
  int a = MUL(2+2, 5+5);
  printf("%d\n", a);
  return 0;
}
```

Qual será o valor impresso para a variável `a`? Justifique sua resposta.

2) (6 pontos) Em certo programa em C deseja-se armazenar dados relativos aos funcionário de uma empresa. Cada funcionário tem um nome de no máximo 80 caracteres, um inteiro com seu número de filhos e um float que representa seu salário. Cada funcionário tem também duas datas, uma que representa sua data de nascimento e outra que representa a data em que foi contratado. Finalmente, cada funcionário pode possuir um certo número de dependentes (no máximo 10). Cada dependente tem um nome de no máximo 80 caracteres e uma idade. As datas devem ser armazenadas da seguinte forma: dia e ano no formato inteiro e mês no formato string, com até 10 caracteres.

Com base no enunciado, faça o que se pede:

a. (4 pontos) Faça as declarações de dados necessárias para se atender ao enunciado desta questão da forma mais clara possível, o que inclui bons nomes para tipos de dados e membros. O uso de typedef é obrigatório.
b. (2 pontos) Faça as declarações de todas as variáveis necessárias (de acordo com os tipos de dados criados), considerando que o programa irá armazenar no máximo 500 funcionários.

3) (8 pontos) Considere o seguinte programa na linguagem C:

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

typedef struct {
  int dia;
  char mes[10];
  int ano;
} Data;
typedef struct {
  char nome[81];
  int num_matricula;
  Data nascimento;
} Aluno;
typedef struct {
  char nome[81];
  Aluno alunos[300];
  int num_alunos;
} Curso;

void imprime_curso(Curso);

int main(int argc, char *argv[]) {
  Curso x;
  /*Preenchimento da variável x. O código não está sendo mostrado, pois é
irrelevante para o exercício.*/
  imprime_curso(x);
  return 0;
}
```

Esse programa realiza a impressão dos dados de um certo curso. Note que, na estrutura Curso, existe uma variável (`num_alunos`) que conta quantos alunos um curso possui (quantas posições do vetor `alunos` estão inicializadas). A inserção de dados na variável `x`, declarada no método main(), não foi mostrada, mas considere que esse variável foi inicializada com uma série de valores.
Você deve agora Implementar a função `imprime_curso()`, cujo protótipo foi mostrado no código acima. Essa função deve imprimir as informações de um curso passado como parâmetro, da seguinte forma:

```
Curso: Ciencia Computacao
Numero de alunos: 3

=======Lista de Alunos=======
Matricula | Nome     | Nascimento
123       | João     | 12 de dezembro de 1989
456       | José     | 1 de abril de 1990
988       | Carlos   | 3 de julho de 1994
```

Os dados sublinhados estão armazenados no curso passado como parâmetro.

4) (8 pontos) Um programador está trabalhando com um programa que faz uso de matrizes quadradas (tamanho `n x n`) de números inteiros. No entanto, para esse programa, matrizes válidas são somente aquelas cujas linhas não tenham elementos repetidos. Por exemplo, a matriz:

```
1 2 3
3 2 1
2 1 3
```

é válida. Já a matriz:

```
1 2 3 4
4 3 2 1
1 1 2 3
8 6 5 3
```

é inválida, já que na terceira linha o número 1 aparece repetido. Com base nessas informações:

a. (4 pontos) Escreva uma função de nome `verifica_linha()`, que recebe como parâmetros um vetor de inteiros e um número `n`, que representa o tamanho desse vetor. Sua função deve retornar o valor 1 caso esse vetor não tenha números repetidos, e 0 caso contrário.

b. (4 pontos) Utilizando a função definida anteriormente, complete a função `main()` do programa em C abaixo, que deve verificar se a matriz `m` é válida. O programa deve imprimir "`Matriz valida!`" caso a matriz seja válida, e "`Matriz inválida!`" caso contrário. **Seu código deve obrigatoriamente utilizar a função definida anteriormente, além de um comando de repetição.**

```
int main(int argc, char *argv[]) {
  int m[5][5] = {{1, 2, 3, 4, 5}, {1, 2, 5, 5, 3}, {5, 4, 3, 2, 1}, {5,
2, 3, 4, 1}, {7, 4, 1, 8, 1}};
  /* Seu codigo aqui */
  return 0;
}
```

5) (5 pontos) Em matemática, uma matriz transposta é obtida ao realizarmos a troca de linhas por colunas em uma determinada matriz. Por exemplo, a matriz:

```
1 2 3
4 5 6
7 8 9
```

possui, como matriz transposta:

```
1 4 7
2 5 8
3 6 9
```

Imagine um programa em C que possua uma matriz de inteiros chamada **matriz** com 10 linhas e 10 colunas, com todas as células já preenchidas com determinados valores. Imagine também que, nesse mesmo programa, temos uma outra matriz de inteiros chamada **matriz_t**, também com 10 linhas e 10 colunas, ainda não inicializada. Faça **um trecho de código** para preencher **matriz_t**, de forma que **matriz_t** seja a matriz transposta de **matriz**.