Universidade Federal de Ouro Preto - UFOP
Disciplina: BCC 201 - Introdução à Programação
Professor: Guillermo Cámara-Chávez

Aluno: _____ **No.** _____

A cola não será tolerada. Se alguém for pego colando, será reprovado com Zero. É considerado cola: olhar/copiar da prova de outro ou deixar outro aluno olhar sua prova. Desligar os celulares, quem for pego olhando o celular, será considerado cola. Implementar cada questão utilizando funções e/ou procedimentos.

2da. Avaliação - Grupo B

1. (2 pts) Escreva um(a) procedimento/função que responda se um par de números são amigos ou não.Dois números A e B são amigos se a soma dos divisores de A excluindo A é igual a B e a soma dos divisores de B excluindo B é igual a A.
   Exemplo:
   220 e 284 são amigos, pois
   220: 1+2+4+5+10+11+20+22+44+55+110=284
   284: 1+2+4+71+142=220
   1184 e 1210 também são amigos.

```
int amigos(int num1, int num2)
{
    int i, j, soma1 = 0, soma2 = 0;
    for (i = 1; i < num1; i++)
    {
        if (num1 % i == 0)
            soma1 += i;
    }
    for (i = 1; i < num2; i++)
    {
        if (num2 % i == 0)
            soma2 += i;
    }
    if (soma1 == num2 && soma2 == num1)
        return 1;
    else
        return 0;
}
```

2. (2 pts) Dado um vetor de 15 números inteiros, faça um(a) procedimento/função para comprimir o vetor suprimindo as repetições de números vizinhos através da contagem do número de repetições de cada um da seguinte forma:
   Vetor de entrada:

   1 1 1 4 1 1 4 4 25 67 67 67 67 2 2

   Vetor de saída:

3 1 1 4 2 1 2 4 1 25 4 67 2 2 0

```
int Comprime(int* x, int tamX, int* c)
{
    int num, i, cont = 0, j = 0;
    num = x[0];
    i = 0;
    while (i < tamX)
    {
        if (num == x[i])
        {
            cont++;
            i++;
        }
        else
        {
            c[j++] = cont;
            c[j++] = num;
            cont = 0;
            num = x[i];
        }
        if (i == tamX)
        {
            c[j++] = cont;
            c[j++] = num;
        }
    }
    return j;
}
```

3. (2 pts) Construir um programa que seja capaz de embaralhar uma string $s1$ com uma string $s2$ e colocar o resultado em uma string $s3$. Para embaralhar $s1$ com $s2$ é necessário preencher os índices pares de $s3$ com os elementos de $s1$ e os ímpares com os elementos de $s2$ até que os elementos de uma das duas strings termine e os demais elementos de $S3$ serão preenchidos com os elementos da string restante. Considere o índice 0 (zero) como sendo par. Por exemplo:

   - s1 = "local"
   - s2 = "misterio"
   - Nova string s3 = "lmoicsatlerio"

```
void embaralha(char* cad1, char* cad2, char* cadFinal)
{
    int tam1 = strlen(cad1), tam2 = strlen(cad2), i, j;
    for (i = 0, j = 0; j < tam1 && j < tam2; i+=2, j++)
    {
        cadFinal[i] = cad1[j];
        cadFinal[i+1] = cad2[j];
    }
    for (; j < tam1; j++, i++)
        cadFinal[i] = cad1[j];
    for (; j < tam2; j++, i++)
        cadFinal[i] = cad2[j];
    cadFinal[i] = '\0';
}
```

4. (2 pts) Faça um(a) procedimento/função que receba uma matriz $m \times n$ e normalize seus valores dividindo cada elemento de uma coluna pelo maior valor da coluna da

matriz. Exemplo: matriz de entrada $\begin{matrix} 2 & 3 & 4 & 5 \\ 3 & 6 & 7 & 9 \\ 1 & 5 & 2 & 7 \end{matrix}$

Matriz resultante $\begin{matrix} \frac{2}{3} & \frac{3}{6} & \frac{4}{7} & \frac{5}{9} \\ \frac{3}{3} & \frac{6}{6} & \frac{7}{7} & \frac{9}{9} \\ \frac{1}{3} & \frac{5}{6} & \frac{2}{7} & \frac{7}{9} \end{matrix}$

```
#define L 5
#define C 6
void Normaliza(float M[L][C])
{
    float maxElem;
    int i, j;
    for (j = 0; j < C; j++)
    {
        maxElem = M[0][j];
        for (i = 1; i < L; i++)
        {
            if (M[i][j] > maxElem)
                maxElem = M[i][j];
        }
        for (i = 0; i < L; i++)
        {
            M[i][j] /= maxElem;
        }
    }
}
```

5. (2 pts) Escreva um programa que leia um vetor de 10 elementos reais, e um valor qualquer X. Determine o vetor resultante da multiplicação de X pelo vetor lido.

```
void Multiplica(int* V, int n, int num)
{
    int i;
    for (i = 0; i < n; i++)
    {
        V[i] *= num;
    }
}
```