Universidade Tecnológica Federal do Paraná
Departamento Acadêmico de Informática
IF61C – Fundamentos de Programação 1
Prof. Bogdan Tomoyuki Nassu
Profa. Leyza Baldo Dorini

1.

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

#define N 4

int ehSimetrica (int matriz [N][N])
{
    int i, j;

    /* Note que só preciamos verificar as posições à direita e acima da diagonal
       principal. Na diagonal, os elementos seriam comparados consigo próprios
       (!?), e à direita e abaixo, os elementos já são comparados quando
       verificamos as posições [j][i]. */
    for (i = 0; i < N - 1; i++)
        for (j = i + 1; j < N; j++)
            if (matriz [i][j] != matriz [j][i])
                return (0);
    return (1);
}

int main ()
{
    int i, j;
    int matriz [N][N];

    /* Lê */
    for (i = 0; i < N; i++)
    {
        for (j = 0; j < N; j++)
        {
            printf ("Elemento em (%d, %d): ", i, j);
            scanf ("%d", &(matriz [i][j]));
        }
    }

    if (ehSimetrica (matriz))
        printf ("Simetrica!\n");
    else
        printf ("Nao eh simetrica!\n");

    return (0);
}
```

2.

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

#define N_LINHAS 4
#define N_COLUNAS 5

int temRepeticao (int matriz [N_LINHAS][N_COLUNAS])
{
    int i_elem, j_elem, i_rep, j_rep;

    /* Para cada elemento da matriz... */
    for (i_elem = 0; i_elem < N_LINHAS; i_elem++)
    {
        for (j_elem = 0; j_elem < N_COLUNAS; j_elem++)
        {
            int elemento = matriz [i_elem][j_elem];

            /* Percorre o resto dessa linha. */
            for (j_rep = j_elem + 1; j_rep < N_COLUNAS; j_rep++)
                if (matriz [i_elem][j_rep] == elemento)
                    return (1);

            /* Percorre o resto da matriz. */
            for (i_rep = i_elem + 1; i_rep < N_LINHAS; i_rep++)
                for (j_rep = 0; j_rep < N_COLUNAS; j_rep++)
                    if (matriz [i_rep][j_rep] == elemento)
                        return (1);
        }
    }

    return (0);
}

int main ()
{
    int i, j;
    int matriz [N_LINHAS][N_COLUNAS];

    srand (time (NULL));

    /* Preenche */
    for (i = 0; i < N_LINHAS; i++)
        for (j = 0; j < N_COLUNAS; j++)
            matriz [i][j] = rand () % 100;

    /* Imprime */
    for (i = 0; i < N_LINHAS; i++)
    {
        for (j = 0; j < N_COLUNAS; j++)
            printf ("%d\t", matriz [i][j]);
        printf ("\n");
    }

    /* Testa */
    if (temRepeticao (matriz))
        printf ("Tem elementos repetidos.");
    else
        printf ("Nao tem elementos repetidos.");

    return (0);
}
```

3.

```
/* O desafio aqui é enxergar que precisamos encontrar letras iguais percorrendo
  a matriz de 4 formas diferentes: na ordem normal ([i][j]), trocando as linhas
  por colunas ([j][i]), na ordem inversa ([N-1-i][N-1-j]), e na ordem inversa na
  vertical ([N-1-j][N-1-i]). */

#include <stdio.h>

#define N 5

int testaMatriz (char matriz [N][N])
{
    int i, j;

    for (i = 0; i < N; i++)
        for (j = i; j < N-i; j++)
        {
            if (matriz [i][j] != matriz [j][i] ||
                matriz [i][j] != matriz [N-1-i][N-1-j] ||
                matriz [i][j] != matriz [N-1-j][N-1-i])
                    return (0);
        }

    return (1);
}

int main ()
{
    char m [N][N] = {{'s','a','t','o','r'},
                     {'a','r','e','p','o'},
                     {'t','e','n','e','t'},
                     {'o','p','e','r','a'},
                     {'r','o','t','a','s'}};

    if (testaMatriz (m))
        printf ("eh!\n");
    else
        printf ("nao eh!\n");

    return 0;
}
```