


## Relatório Projeto Símios

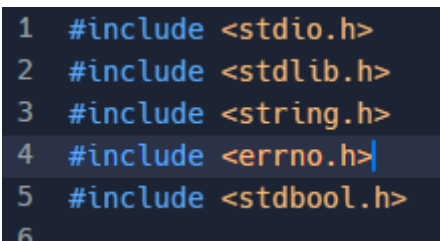
O projeto tem como objetivo receber uma matriz que será o DNA de um humano ou símio, o tamanho da matriz é predefinida pelo usuário através de um segundo arquivo .txt, onde ele deve digitar a matrix de que forma ela seja, podendo ser de qualquer tamanho apenas possuindo caracteres que sejam C,T,G ou A de formas maiúsculas.



A screenshot of a text editor window titled 'dna.txt'. It displays a 13x13 matrix of DNA sequences. Each row is numbered from 1 to 13 on the left. The sequences are as follows:

Line	Sequence
1	ATCGCTGAGCAAT
2	TCCGATGCAATTC
3	GCTAATGGCTAGT
4	GGTCTCCATGCAT
5	CTAGAGTTGCAGA
6	GCTAATGACTAGT
7	ACTAGTGAGCAAT
8	TCCGATCCATGCA
9	ACCATGATGCAAT
10	TCCGATCAATTCC
11	GAGCAATACCATG
12	GATGCAAACCATG
13	CCGATGCAAGTAC

Acima pode se encontrar o exemplo do arquivo de envio de DNA, que possui um tamanho de 13 por 13, onde possui apenas caracteres A,C,T ou G



A screenshot of C code showing the first six lines of a program, which are standard library includes:

```
1 #include <stdio.h>
2 #include <stdlib.h>
3 #include <string.h>
4 #include <errno.h>
5 #include <stdbool.h>
6
```

Foram importadas bibliotecas padrão e algumas extras como a <string.h> que foi utilizada para a ser feito a comparação de strings de uma forma correta, algumas outras foram adicionadas para erros, e para a manipulação de variáveis.

```

7 char **leMatriz(size_t *, size_t *, char *);
8 void imprimeMatriz(char**, size_t, size_t);
9 void liberaMatriz(char**, size_t);
10 int analisaMatriz(char**, size_t, size_t);
11 int analisaLinhasSubMatriz(char**, int, int, size_t, size_t);
12 int analisaColunasSubMatriz(char**, int, int, size_t, size_t);
13 int analisaDiagonalPrincipalSubMatriz(char**, int, int, size_t, size_t);
14 int analisaDiagonalSecundariaSubMatriz(char**, int, int, size_t, size_t);
15 void simioDetectado(char**, int, int, int, size_t, size_t);
16 void linhaDetectada(char**, int, int, size_t, size_t);
17 void colunaDetectada(char**, int, int, size_t, size_t);
18 void diagonalPrincipalDetectada(char**, int, int, size_t, size_t);
19 void diagonalSecundariaDetectada(char**, int, int, size_t, size_t);
20

```

A prototipação foi utilizada para um melhor entendimento do código e também para uma usabilidade melhor durante o projeto, sendo assim mais fácil acesso as variáveis.

```

char arquivo[256];

printf("Digite o nome do arquivo contendo o DNA (Ex: dna.txt): ");
scanf("%s", arquivo);

char** matriz = leMatriz(&col, &lin, arquivo);

if(matriz == NULL) {
    fprintf(stderr, "nao foi possivel ler metriz\n");
    return 1;
}

```

Nessa parte do código é onde se encontra a leitura do arquivo externo do DNA, onde se deve receber um arquivo, que pode possuir qualquer nome apenas sendo extensão .txt, mas se o arquivo que for lido e estiver vazio retorna um erro que não foi possível ler a matriz.

```

char **leMatriz(size_t *linhas, size_t *cols, char *filename) {
    if (linhas == NULL || cols == NULL || filename == NULL)
        return NULL;

    *linhas = 0;
    *cols = 0;

    FILE *fp = fopen(filename, "r");

    if (fp == NULL) {
        fprintf(stderr, "nao foi possivel abrir %s: %s\n", filename, strerror(errno));
        return NULL;
    }
}

```

Aqui é onde se encontra a principal parte da leitura da matriz, onde será recebido os valores do arquivo externo e transferido seus valores, onde possui uma mensagem de erro se não for possível abrir o arquivo de texto.

```

int analisaMatriz (char** matriz, size_t lin, size_t col) {
    int i, j;
    int isSimian = 0;
    for (i = 0; i < lin - 3; i++) {
        for (j = 0; j < col - 3; j++) {
            isSimian = analisaLinhasSubMatriz(matriz, i, j, lin, col);
            if (isSimian == 1) return 1;

            isSimian = analisaColunasSubMatriz(matriz, i, j, lin, col);
            if (isSimian == 1) return 1;

            isSimian = analisaDiagonalPrincipalSubMatriz(matriz, i, j, lin, col);
            if (isSimian == 1) return 1;

            isSimian = analisaDiagonalSecundariaSubMatriz(matriz, i, j, lin, col);
            if (isSimian == 1) return 1;
        }
    }
    return isSimian;
}

```

Na função analisa matriz =, está sendo recebido a matriz é feito a comparação com cada parte do código, assim como as linhas, as colunas, e as diagonais principal e secundária.

```

void simioDetectado(char** mat, int i, int j, int identificador, size_t lin, size_t col) {
    switch (identificador) {
        case 0:
            linhaDetectada(mat, i, j, lin, col);
            break;

        case 1:
            colunaDetectada(mat, i, j, lin, col);
            break;

        case 2:
            diagonalPrincipalDetectada(mat, i, j, lin, col);
            break;

        case 3:
            diagonalSecundariaDetectada(mat, i, j, lin, col);
            break;
    }
}

```

em Símio Detectado, é aonde ele vai dizer onde que a sequência genética repetida com 4 caracteres se foi na linha, coluna ou diagonal principal, ou na diagonal secundária, e se for encontrado o sequência em qualquer uma estando na ordem da imagem o código já para e acusa que é símio, exibindo a matriz, com as letras coloridas da repetição.

```

void diagonalSecundariaDetectada(char** mat, int i, int j, size_t lin, size_t col) {
    printf("Simio detectado na diagonal secundaria, comecando na linha %d e na coluna %d\n", (i+1),
(j+4));
    for (int k = 0; k < lin; k++) {
        for (int l = 0; l < col; l++) {
            if (((k == i) && (l == j+3)) || ((k == i+1) && (l == j+2)) ||
((k == i+2) && (l == j+1)) || ((k == i+3) && (l == j))) {
                printf("\x1B[31m" "%-3c" "\x1B[0m",mat[k][l]);
            } else {
                printf("%-3c ", mat[k][l]);
            }
        }
        printf("\n");
    }
}

```

Aqui se encontra o exemplo de detecção da diagonal secundária, onde a função está recebendo a matriz, as linhas, as colunas e tudo, ele vai fazer a comparação para descobrir se há a repetição do mesmo carácter, imprimindo as letras coloridas da diagonal onde ela foi encontrada.

```

int isSimian = analisaMatriz(matriz, lin, col);
if (isSimian == 0) {
    printf("DNA pertence a um humano!");
}

liberaMatriz(matriz, lin);

return 0;
}

```

Caso não ocorra nenhuma detecção no DNA ele apenas retorna uma mensagem dizendo que o DNA pertence a um humano, e libera a memória da matriz, para não sobrecarregar a memória do código ficando pesado.