1. Resumo: Você deverá implementar em C++ dois algoritmos de busca por subsequências de caracteres, o algoritmo Knuth-Morris-Pratt e o algoritmo "força-bruta", e deverá avaliar o desempenho obtido por eles. Cada aluno deverá, individualmente, combinar com o professor a data e o horário para apresentação do trabalho.

Importante: lembre que o objetivo da disciplina é cada aluno aprimorar suas habilidades de programação e aprender os algoritmos ensinados, e que o caminho indicado pelo professor para que isso aconteça passa por cada aluno escrever o seu próprio código. Portanto, por favor colabore com o bom andamento da disciplina, não copiando códigos da internet ou de outros alunos, e leve em consideração que o professor está à disposição para ajudar. Obrigado.

2. Introdução: Os dois algoritmos de busca por subsequências deverão ter a forma a seguir:

void buscar (**const char** ∗texto, **const char** ∗padrao, **int** ∗saida)

Explicações sobre os parâmetros:

- Os dois primeiros parâmetros são strings ao estilo de C: sequências de caracteres terminadas por um '\0'. Observe que os tamanhos dessas strings não são parâmetros da função, pois o fim de cada string pode ser detectado pelo '\0'. Apesar disso, os algoritmos podem partir da seguinte pré-condição: se n é o tamanho do texto e m o do padrão (excluindo o '\0' da contagem), então $1 \leq m \leq n$.
- O último parâmetro aponta para um vetor onde deve ser gravada a saída do algoritmo. Mais especificamente, se houver k ocorrências do padrão, as quais comecem respectivamente nos índices $i_0, \ldots, i_{k-1}$ do texto, então o algoritmo deve gravar i_0 em saida[0], i_1 em saida[1], ..., i_{k-1} em saida[k−1], e finalmente -1 em saida[k], para indicar o fim do conteúdo escrito no vetor; as posições saida[i] com $i > k$ não devem ser utilizadas pelo algoritmo. O tamanho do vetor onde a saída será gravada não é conhecido pelo algoritmo, mas é pré-condição que o vetor seja grande o suficiente[1] para armazenar a saída.

Com relação às instâncias a serem fornecidas para os algoritmos, o seu programa deve possuir pelo menos os seguintes 4 tipos:

- **Aleatórias:** o texto e o padrão devem possuir caracteres cujos códigos sejam obtidos pseudoaleatoriamente. Os caracteres utilizados deverão ser as l primeiras letras minúsculas do alfabeto, com $1 \leq l \leq 26$ e sendo l informado pelo usuário.
- **Pior Caso 1:** o texto consiste em n ocorrências da letra "a" ("aaa...aa"), e o padrão em $m-1$ ocorrências de "a" seguidas de 1 ocorrência da letra "b" ("aaa...ab").
- **Pior Caso 2:** o texto consiste em n ocorrências da letra "a" ("aaa...a"), e o padrão em m ocorrências de "a" ("aaa...a").
- **Textos Reais:** a serem fornecidas pelo professor via SIGAA.

3. Requisitos: Segue abaixo uma especificação do restante do trabalho. Em caso de dúvida, por favor entre em contato com o professor rapidamente.

(a) Você deve apresentar um programa que comece perguntando ao usuário o tipo de instância a ser utilizada.

[1] A sua implementação pode sempre fornecer aos algoritmos um vetor com $n+1$ posições, por exemplo.

- Se a instância não for de texto real (isto é, se a instância for aleatória ou de pior caso), então o programa deve perguntar ao usuário o tamanho n do texto e o tamanho m do padrão, respeitando $1 \leq m \leq n$, bem como o valor de l, respeitando $1 \leq l \leq 26$, no caso de instâncias aleatórias.
- Se a instância for de texto real, então devem ser seguidas as instruções a serem fornecidas pelo professor para essas instâncias.

(b) Após o usuário fornecer os dados necessários para a construção da instância, a instância deve ser gerada e então fornecida para cada um dos dois algoritmos em questão. O tempo utilizado por cada algoritmo deve ser registrado e então impresso na tela.

(c) O programa também deve checar se os dois algoritmos geraram a mesma saída, e imprimir na tela o resultado dessa checagem.

(d) Por fim, o programa deve facultar ao usuário a realização de outro teste.