MC202GH - ESTRUTURAS DE DADOS

2º Semestre de 2017

Professor: Rafael C. S. Schouery

Monitores: Yulle Glebbyo Felipe Borges (PED)

Victor Andrietta Razoli Silva (PAD)

Lab. 03 - Manipulação de Sequências de DNA

Peso 2

1. O Problema

Para expandir o nosso entendimento sobre a evolução, cientistas estudam e comparam as diferenças dos genomas de diversas espécies. A distância evolucionária entre duas espécies pode ser dada através da distância de rearranjo de genomas. Uma mutação em grande escala pode ser representada através de um rearranjo do genoma. Assim, é útil que possamos representar computacionalmente cadeias de DNA para que possamos simular certas mutações e medir o nível de semelhança entre diferentes espécies da mesma família. Você foi escolhido para colaborar com um grupo de cientistas da Unicamp em um estudo deste tipo. Sua parte no projeto é criar uma estrutura de dados que represente uma cadeia de DNA, e que suporte algumas operações de rearranjo.

Uma cadeia de DNA **deverá ser representada como uma lista ligada** de núcleo-bases. Cada núcleo-base é representada por um caractere que pode assumir valor 'g', c', 'a' ou 't'. Ao início de uma simulação, tem-se uma cadeia vazia. A partir deste ponto, deve ser possível adicionar uma nucleo-base à cadeia. Operações de adição devem suportar um parâmetro posicional, que indica o índice que será ocupado pela nova núcleo-base na cadeia atual. Deve ser possível também, remover uma nucleo-base de uma certa posição da cadeia. Por fim, a simulação deve suportar operações de rearranjo como inversão de prefixo ou sufixo, no qual um trecho da cadeia deve ter sua ordem invertida, e transposição, no qual um trecho da cadeia deve ser movido para outra posição na mesma cadeia. Por exemplo, a inversão de prefixo de tamanho 3 na cadeia "**cgt**agctt" resultaria na cadeia "**tgc**agctt", e a transposição do trecho GC da cadeia "cgta**gc**tt" duas posições para direita resultaria em "cgtatt**gc**".

2. Entrada

As operações a serem suportadas pela simulação e suas respectivas entradas são detalhadas na tabela abaixo:

ID	Nome	Descrição
1	Inserção de uma núcleo-base	Dado um **caractere** representando a núcleo-base e um **inteiro p** representando a posição que a núcleo-base deve ser inserida na cadeia, insere o elemento. Dica: Para ler um caractere corretamente, utilize o comando `scanf(" %c", &caractere)` (com um espaço antes do `%c`)
2	Remoção de uma núcleo-base	Dado um **inteiro** representando uma posição da sequência, remove a núcleo-base que está nesta posição
3	Inversão de um prefixo do DNA	Dado um **inteiro p** representando o tamanho do prefixo, inverte as posições das p primeiras núcleo-bases da cadeia entre si. Ex: "**cgta**gctt" vira "**atgc**gctt" para `p=4`. Assume-se que o tamanho de um prefixo será sempre menor que o tamanho total da cadeia.
4	Inversão de um sufixo do DNA	Dado um **inteiro p** representando o tamanho do sufixo, inverte as posições das **p** últimas nucleo-bases da cadeia entre si. Ex: "cgtag**ctt**" vira "cgtag**ttc**" para `p=3.` Assume-se que o tamanho de um sufixo será sempre menor que o tamanho total da cadeia.
5	Transposição de um trecho do DNA	Dado **dois inteiros p e q** que descrevem um intervalo do DNA ($p \leq q$) e **um inteiro r** que representa o tamanho da transposição do trecho, move as núcleo-bases descritas de p até q para r núcleo-bases a direita (caso r seja positivo) ou a esquerda (caso r seja negativo). Ex: "cg**ta**gctt" vira "cggctt**ta"** com `p=2, q=3` e `r=4.` Assume-se que o intervalo de transposição nunca será total, e que o tamanho da transposição seja sempre válido para o intervalo dado.
6	Imprimir a cadeia de DNA	Imprime a cadeia de DNA atual com um espaço em branco separando cada núcleo-base
0	Finalização	Saia do sistema, liberando corretamente toda a memória utilizada

Exemplo de entrada:

```
1
t 0
1
c 0
1
g 0
1
t 0
1
g 0
1
c 0
1
t 6
1
a 3
3 4
4 3
5
2 3 -1
6
0
```

3. Saída

Abaixo, temos as saídas esperadas correspondentes a cada uma das operações:

ID	Saída	Comentário
1	Nucleo-base adicionada na posicao **Y** da sequencia!	**Y** é a posição passada como parâmetro para inserção
2	Nucleo-base da posicao **Y** removida!	**Y** é a posição passada como parâmetro para remoção
3 e 4	X0 X1 … Xp - Xp … X0 X1	**X0 X1** e **Xp** são as respectivas núcleo-base envolvidas na inversão. Obs: Deve existir um caractere de espaço branco ao fim da linha, antes do **\n**
5	X0 X1 … Xp S r	**X0 X1** e **Xp** são as respectivas

		núcleo-base envolvidas na transposição. **S** é '>' caso o movimento seja para direita e '<' caso o movimento seja para esquerda; e **r** é o tamanho do movimento. Caso o movimento tenha tamanho 0, **S** deve ser '>'
6	X0 X1 X2 X3 … Xn	Os valores de cada núcleo-base da sequência separadas por um espaço em branco. Obs: Deve existir um caractere de espaço branco depois da última núcleo-base, antes do `\n`
0	Sistema encerrado.	Antes de encerrar o sistema, é necessário liberar toda a memória alocada

Exemplo de saída:

```
Nucleo-base adicionada na posicao 0 da sequencia!
Nucleo-base adicionada na posicao 0 da sequencia!
Nucleo-base adicionada na posicao 0 da sequencia!
Nucleo-base adicionada na posicao 0 da sequencia!
Nucleo-base adicionada na posicao 0 da sequencia!
Nucleo-base adicionada na posicao 0 da sequencia!
Nucleo-base adicionada na posicao 6 da sequencia!
Nucleo-base adicionada na posicao 3 da sequência!
c g t a - a t g c
c t t - t t c
g c < 1
a g c t g t t c
Sistema Encerrado.
```

4. Informações

- Este laboratório possui **peso 2**.
- **Não** há um número máximo de submissões.
- No início de cada arquivo a ser submetido, insira um comentário com seu nome, RA e uma breve descrição do conteúdo do arquivo.
- Apenas comentários no formato `/* comentário */` serão aceitos. Comentários com `//` serão acusados como erros pelo SuSy.
- Os cabeçalhos permitidos são:

    - `stdio.h`
    - `stdlib.h`
    - `math.h`
- A submissão da sua solução deverá conter múltiplos arquivos:
    - **lista.h**: interface da lista ligada
    - **lista.c**: implementação da interface da lista ligada
    - **lab03.c**: cliente que utiliza a nova estrutura
- Será disponibilizado um **Makefile** para este trabalho na página do laboratório:
    - Para compilar seu projeto, basta navegar até o diretório do projeto e utilizar o comando ‘make’ no terminal do Linux.
    - Veja mais instruções na página da disciplina.

5. Critérios de Avaliação

- Seu código deverá passar por todos os casos de teste do SuSy definidos para este laboratório. Caso positivo, utilizaremos os seguintes critérios adicionais:
    - Indentação correta de todo o código. Escolha entre espaços ou tabs para indentar seu código e seja coerente em todos os arquivos. Indentações utilizando espaços e tabs intercalados, ou trechos não indentados corretamente irão resultar em uma penalização de **um ponto** na nota
    - A **não utilização da estrutura de lista ligada** para a representação da cadeia de DNA neste laboratório implicará em nota **zero**
    - Utilização de bibliotecas ou funções não permitidas pelo enunciado resultará em uma **penalização severa** na nota
    - Toda a memória alocada deve ser liberada adequadamente ao fim do programa