Estrutura de Dados e Algoritmos
Projeto 05: a volta do k-ésimo menor
Ciência da Computação

Prof. Daniel Saad Nogueira Nunes

1 Contextualização

O k-ésimo menor é um problema bem conhecido por você. Dado uma sequência de elementos e um parâmetro k, deve-se determinar qual o k-ésimo menor elemento. O prof. Daniel, para dificultar um pouco a sua vida, resolveu propor uma versão diferente deste problema. Nesta versão, inserções e remoções de elementos são permitidas. Será que você consegue continuar resolvendo este problema?

2 Especificação

O projeto deverá ser executado através da linguagem `C`.

A entrada deve ser lida da entrada padrão (`stdin`), enquanto a saída deverá ser impressa na saída padrão (`stdout`).

O programa deverá obedecer rigorosamente o formato de saída especificada, pois parte da correção será automatizada.

Cerifique-se de utilizar uma ED eficiente para resolver o problema.

2.1 Entrada

A primeira linha da entrada possui um inteiro Q ($2 \leq Q \leq 3 \cdot 10^5$), indicando o número de operações que o seu programa deverá executar. As próximas Q linhas, descrevem cada, uma operação, que é da forma:

- 1 x: insere o inteiro x ($1 \leq x \leq 10^9$) na sequência de elementos. É garantido que x é distinto dos demais elementos da sequência.
- 2 x: remove o item x ($1 \leq x \leq 10^9$) na sequência de elementos. É garantido que x está na sequência de elementos.
- 3 k: imprime o k-ésimo menor elemento da sequência. Os valores que k podem assumir estão no intervalo $[1, n]$, em que $n \geq 1$ é o tamanho atual da sequência.É garantido haver ao menos uma operação deste tipo na entrada.

2.2 Saída

Para cada operação de impressão, seu programa deverá imprimir uma linha com o k-ésimo menor.

3 Exemplos

- Entrada :

```
10
1 1
1 2
1 3
```

```
1 4
1 5
3 1
3 2
3 3
3 4
3 5
```

- Saída:

```
1
2
3
4
5
```

- Entrada:

```
5
1 1
1 2
1 3
2 2
3 2
```

- Saída:

```
3
```

- Entrada:

```
9
1 1
1 2
1 3
3 1
2 1
3 1
2 2
3 1
2 3
```

- Saída:

```
1
2
3
```

3.1 Compilação

Um arquivo `Makefile` deve ser disponibilizado para compilação do projeto.

3.2 Limites de Tempo e Memória

Para cada caso de teste, será permitido a execução do programa por apenas 1 segundo com utilização máxima de 256 MB de memória. Caso o programa leve mais tempo ou memória do que isso, será considerado que o algoritmo empregado foi ineficiente.

3.3 Documentação

Junto do(s) código(s) necessário(s) para resolver o problema, deverá ser disponibilizado um arquivo README, identificando o autor do trabalho e especificando as instruções para compilação e execução do(s) código(s).

3.4 Critérios de Correção

Fazem partes dos critérios de correção:

- Eficiência do programa.
- Utilização de estruturas de dados adequadas.
- Documentação: além do arquivo README, o código deve estar bem documentado.
- Legibilidade.

3.5 Ambiente de Correção

Os projetos serão corrigidos em uma máquina com sistema `GNU/Linux` e compilador `gcc 10.2.0`.

Trabalhos que não compilarem não serão avaliados.

4 Considerações

- Este projeto deve ser executado individualmente.
- Os trabalhos que incidirem plágio serão avaliados automaticamente com nota 0 para os envolvidos. Medidas disciplinares também serão tomadas.
- O trabalho deve ser entregue dentro de uma pasta zipada com a devida identificação do(s) aluno(s) através da sala de aula virtual da disciplina na data estipulada no ambiente.