Estrutura de Dados e Algoritmos
Projeto 05: Mediana Flutuante
Ciência da Computação

Prof. Daniel Saad Nogueira Nunes

1 Contextualização

Seja $S = (s_0, \ldots, s_{n-1})$ uma sequência de inteiros. A mediana de S é definida como:

- O valor do meio da sequência S ordenada, se n é ímpar.
- A média aritmética dos valores do meio da sequência S ordenada, se n é par.

Por exemplo, para a sequência $S = (2, 1, 3)$, a mediana é 2, enquanto para a sequência $S = (2, 4, 1, 3)$, a mediana é $\frac{2+3}{2} = 2.5$.

Inicialmente, a sequência S é vazia. A cada iteração, um novo inteiro é inserido nesta sequência. Para cada iteração, o valor da mediana de S deverá ser computado.

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.

Para resolver este problema, uma estrutura de dados adequada deve ser utilizada para obtenção das medianas em tempo eficiente. A ordenação da entrada em cada iteração criará um algoritmo ineficiente.

2.1 Entrada

A primeira linha possui um inteiro n ($1 \leq n \leq 10^5$) indicando o número de inteiros a serem lidos.

A próxima linha possui n inteiros, separados por espaço, x_i ($1 \leq x_i \leq 10^9$), representando o inteiro inserido na i-ésima iteração.

2.2 Saída

Seu programa deverá imprimir, para cada iteração, uma linha com a mediana após a inserção do inteiro x_i.

Sua resposta será considerada correta se ela estiver no máximo a uma distância de 10^{-3} do gabarito. Portanto, recomenda-se a impressão dos valores com o máximo de casas decimais.

3 Exemplos

- Entrada:

```
3
2 1 3
```

- Saída:

```
2.0
1.5
2.0
```

- Entrada:

```
4
4 2 1 3
```

- Saída:

```
4.0
3.0
2.0
2.5
```

- Entrada

```
6
12 4 5 3 8 7
```

- Saída:

```
12.0
8.0
5.0
4.5
5.0
6.0
```

3.1 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, considerar-se-á que o algoritmo empregado foi ineficiente.

3.2 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).

3.3 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.

3.4 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.