

Trabalho 04 - Um Jogo muito Nerd

Data de entrega: 04/12/2025

Data de entrega atrasada sem penalidade: 07/12/2025

Importante:

- **Não** olhem códigos de outros grupos ou da internet, exceto os que são fornecidos. Não utilize LLMs.
- Os trabalhos DEVEM ser feitos com o mesmo grupo do trabalho 01.
- TODOS os membros do grupo devem participar e compreender completamente a implementação.
- Em caso de plágio, fraude ou tentativa de burlar o sistema será aplicado nota 0 na disciplina aos envolvidos.
- Alguns alunos serão solicitados para explicar com detalhes a implementação.
- Passar em todos os testes não é garantia de tirar a nota máxima. Sua nota ainda depende do cumprimento das especificações do trabalho, qualidade do código, clareza dos comentários, boas práticas de programação e entendimento da matéria demonstrada em possível reunião.
- O líder do grupo deverá submeter, até a data de entrega, o seu código na plataforma runcodes.hokama.com.br.
- As dúvidas podem ser encaminhadas primeiramente ao monitor da disciplina.
- É esperado e faz parte do aprendizado vocês terem algumas dificuldades como Problemas com Falha de Segmentação, Loops Infinitos, etc. Então não deixe para o último dia!

Este trabalho é em dupla com as mesmas duplas registradas e deverá ser implementado em linguagem C.

Considere o seguinte jogo, um mestre irá te falar uma sequência de números. A cada número que ele falar, você precisa responder dizendo a mediana¹ dos números que foram dados até aquele momento. Considere o seguinte exemplo:

Mestre: 5

Você: Mediana: 5.0

Mestre: 10

Você: Mediana: 7.5

Mestre: 1

Você: Mediana: 5

O jogo continua até o mestre passar o número -1 (um negativo) e tendo acertado todas as medianas você ganha o jogo. Você então decide fazer um programa, uma ideia bastante óbvia é manter um vértice ordenado, e assim que um novo número é dito pelo mestre você coloca no vetor, e pega a mediana.

¹ [https://pt.wikipedia.org/wiki/Mediana_\(estat%C3%ADstica\)](https://pt.wikipedia.org/wiki/Mediana_(estat%C3%ADstica))

Porém essa solução começa a ficar muito custosa quando a quantidade de números começa a crescer demais. Então você precisa de uma solução mais rápida.

Ideia: Manter 2 conjuntos, um com os maiores elementos, e outro com os menores elementos, de forma que cada um tem metade dos números originais (se houver um número ímpar de elementos, um desses conjuntos tem 1 elemento a mais).

Nessa solução quando você vai encontrar a mediana você precisa encontrar o menor do conjunto dos Maiores elementos, e o maior do conjunto dos Menores elementos. E por isso parece uma boa solução você implementar esses conjuntos como Heaps!

O conjunto dos menores elementos deve utilizar um heap de máximo (Max Heap), enquanto o conjunto dos maiores elementos deve utilizar um heap de mínimo (Min Heap). Você poderá basear sua solução no Max Heap feito em aula, e adaptar para criar o seu próprio Min Heap.

Seu programa deverá ler linha por linha da entrada padrão do sistema, e a cada linha seu programa deve imprimir a mediana, com 1 casa decimal, na saída padrão do sistema, precedido pela palavra "Mediana:". Considere a seguinte entrada:

```
5
10
1
-1
```

Ao final do programa seu programa deverá ter impresso ao longo da execução as seguintes linhas:

```
Mediana: 5.0
Mediana: 7.5
Mediana: 5.0
```

Note que como você deve imprimir a mediana a cada entrada, se você estiver executando seu programa normalmente no terminal, as linhas ficam alternadas:

```
5
Mediana: 5.0
10
Mediana: 7.5
1
Mediana: 5.0
-1
```

Você deverá submeter seu trabalho em um arquivo zip contendo todos os seus arquivos ".c" (não é necessário colocar um Makefile).

Bom trabalho!