Trabalho 01 - Soma perfeita

Data de entrega: 13/05/2024 até 18:30, peso 50% de Nt

Importante:

- **Não** olhe códigos de outros ou da internet. Exceto os que são fornecidos. Também não mostre ou publique o seu.
- Em caso de plágio, fraude ou tentativa de burlar os sistemas será aplicado nota 0 na disciplina aos envolvidos.
- Alguns alunos podem ser 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 trabalho pode ser feito em dupla. O método de submissão ainda será divulgado.

Considere o seguinte problema, seja $S \subset \mathbb{N}$, e $k \in \mathbb{N}$, encontrar quantos pares $\{i, j\}$: $i, j \in S, i \neq j$ e $i + j = k$, existem em S.

Um programador implementou o seguinte código para sortear S e resolver o problema:

```
import random

tamanho = int(input())
random.seed(tamanho)
l = []
while len(l) < tamanho:
      num = random.randint(0, 2**17)
      if num not in l:
            l.append(num)

alvo = random.randint(0, 2**17)
while alvo % 2 == 0:
      alvo = random.randint(0, 2**17)

contador = 0
for i in l:
      compl = alvo - i
      if compl in l:
            contador = contador + 1

print(int(contador/2))
```

Entretanto o código feito é bastante ineficiente, nesse trabalho você deverá entender o porque esse código é ineficiente e desenvolver 3 alternativas (ainda em Python) para melhorá-lo, chegando ao mesmo resultado. Sujeito às seguintes restrições:

- Uma das alternativas obrigatoriamente deve usar Tabela Hash **implementada por você**.
- Você não deve usar bibliotecas muito elaboradas (ex: NumPy, itertools, etc). Em caso de dúvida se algo pode ser utilizado consulte o professor.
- Suas alternativas devem seguir ideias diferentes (e não apenas implementações diferentes da mesma ideia.)