***1. Qual a saída do algoritmo?***

***Resposta:***Ele imprime no console os números primos, dentro de um limite, paginados e ordenados.

***2. Você julga que este código é limpo? Aponte quais erros o programador cometeu que prejudicaram a qualidade do código.*** *Obs: não existe nenhum bug escondido no código.*

***Resposta:*** Não, o código não é um código limpo. A escolha dos nomes (classe, método e variáveis) não é nada descritiva, não sabemos exatamente o que cada coisa está representando no código. Está sendo definido todas as variáveis como let, até mesmo as que tem um valor constante, quando deveria ser usado const. O código está em um único bloco, tendo mais de uma responsabilidade num único método, ao mesmo tempo que ele verifica os números primos, ele personaliza e formata a saída. Poderia ter comentários no código, explicando o que está sendo feito, ou uma documentação. Não tem nenhum tipo de tratamento de erro ou validação de entradas.

***3. Refatore o código do arquivo utilizando conceitos de Clean Code, de maneira que o código se torne mais limpo, legível e de fácil manutenção.***

***Resposta:*** Código refatorado no arquivo *main.js* na raiz do repositório.

***4. Explique como o conceito de middlewares no Express.js pode ser utilizado para evitar repetição de código.***

***Resposta:***

***5. Tendo em vista duas abordagens de backend: uma utilizando um ORM (como Prisma e Sequelize) e outra utilizando apenas um query builder (como o Knex), quais as vantagens e desvantagens de cada abordagem?***

***Resposta:***

***6. Faça uma query em SQL que traga em cada linha o nome de jogadores que se enfrentaram mais de duas vezes, onde em cada partida a soma dos pontos foi maior que 30 e a duração do jogo foi maior que 90 minutos. Não pode haver resultados repetidos.***

***Resposta:*** Query no arquivo *6.query.sql* contido na raiz do repositório.