Esse é um exercício que tem como objetivo praticar os conceitos de Smart Pointers. Mais do que a implementação em si, que é relativamente simples, é importante que você observe e entenda o que está acontecendo no seu código.

Você deverá criar uma classe Teste que tem um único atributo inteiro. Sua classe deve ter dois construtores: um construtor padrão que inicializa o atributo com o valor 0 e um outro construtor que recebe um valor inteiro e inicializa o atributo da classe com esse valor. O seu construtor deverá imprimir "Construtor" seguido de um espaço e o valor do atributo. Sua classe também deverá ter um destrutor que imprime "Destrutor" seguido de um espaço e o valor do atributo. (coloque um endl depois das impressões).

O seu main deverá ler um valor inteiro n.

Se n for par, você entra em um loop com um contador c variando de 1 a n. Dentro desse loop, você deverá criar um apontador tradicional para um objeto da classe Teste e alocar memória para ele inicializando-o com o valor do contador. Você também deverá criar um smart pointer do tipo unique_ptr para um outro objeto da classe Teste também inicializando-o com o valor do contador.

Se n for ímpar, você deverá criar um smart pointer do tipo shared_ptr para um objeto da classe Teste, inicializá-lo com o valor 0, e depois entrar em um loop com um contador c variando de 1 a n. Dentro desse loop, você deverá declarar um outro shared_ptr para um objeto da classe Teste e atribuir a ele o apontador criado antes do loop. Dentro do loop, você deverá também alterar o valor do atributo do objeto apontado pelo shared_ptr para o valor do contador. Ao final do loop, você deverá imprimir o número de shared_ptr que compartilham a mesma instância do um objeto, ou seja o número de shared_ptr estão atualmente apontando para o mesmo objeto (consulte o método use_count). Coloque um endl depois da impressão.