Na programação orientada a objetos dispomos de métodos especiais que inicializam e configuram os objetos, preparando-os para uso, e que são invocados automaticamente sempre que uma nova instância da classe é criada. Embora tenham algumas características específicas e únicas, os construtores podem, assim como outros métodos, serem ou não parametrizados.

Agora que você aprendeu a utilizar construtores em Java, finalize a codificação da classe Celular proposta abaixo, incluindo três métodos construtores que atendam as seguintes especificações:

Construtor com IMEI e Número de Telefone: Este construtor deve receber como parâmetros o IMEI e o número de telefone, armazenando os valores recebidos nos devidos atributos. Construtor com Modelo e Marca: Este construtor deve receber como parâmetros o modelo e a marca, armazenando os valores recebidos nos devidos atributos. Construtor Padrão (Default): Este construtor não recebe parâmetros e não deve executar qualquer tarefa de preparação para o objeto.

Para submeter sua resposta, copie todo o código dado a seguir para o editor e complete a definição da classe Celular incluindo, na seção indicada, os três métodos construtores solicitados. O processo de análise, verificação e teste de corretude será executada pelo restante do código fornecido, portanto não faça alterações em outras partes do conteúdo.

```java
import java.util.Objects;
import java.util.Scanner;

// Início da seção com a definição da classe Celular //////////////////////

class Celular {
    String modelo;
    String marca;
    long imei;
    String numero;
    String cor;
    int anoFabricacao;

    // Início: Seção de definição dos métodos construtores ////////////////////



    // Fim: Seção de definição dos métodos construtores ///////////////////////

    // Método que apresenta na tela o estado do objeto
    public void mostraDados() {
        System.out.println("Modelo : " + modelo);
        System.out.println("Marca : " + marca);
        System.out.println("IMEI : " + imei);
        System.out.println("Número de Telefone : " + numero);
        System.out.println("Cor : " + cor);
        System.out.println("Ano de Fabricação : " + anoFabricacao);
    }

    // Outros métodos da classe
    public int hashCode() {
        return Objects.hash(modelo, marca, imei, numero, cor, anoFabricacao);
    }
}

// Fim da seção com a definição da classe Celular //////////////////////

public class Main {

    public static void main(String[] args) {
        Scanner input = new Scanner(System.in);

        byte code = input.nextByte();
        System.out.println(switch (code) {
            case 1 -> new Celular().hashCode();
            case 2 -> new Celular(input.nextLong(), input.next()).hashCode();
            case 3 -> new Celular(input.next(), input.next()).hashCode();
            default -> throw new IllegalArgumentException("Error");
        });
```

Entrada

Um conjunto de casos de testa que verificam individualmente a existência e funcionalidade dos construtores solicitados.

Saída

Resultados esperados para dos testes fornecidos como entrada.

Samples Input	Samples Output
3 iPhone13 Apple	-71857548
1	887503681
2 123456789012345 11987564321	-1091393369