Atividade Multicast - Implementação de serviço de chat P2P usando Sockets Multicast e UDP

Implementar um serviço de chat que possibilite:
- envio de mensagens para um grupo de pessoas (MulticastSocket) – receber na port 6789
- envio de mensagens individuais para as pessoas ativas (DatagramSocket) – receber na porta 6799
- interface de interação (GUI ou CLI)

Formato das Mensagens:

```
byte type;                    // especifica o tipo da mensagem [0x01 a 0x05]
byte size_source;             // tamanho do apelido
byte source[size_source];     // apelido
byte size_message;            // tamanho da mensagem
byte message[size_message];   // mensagem
```

Dica: usar a estruruta de dados `Mensagem (byte type, String source, String message)` disponível no template.

-- **JOIN [apelido]** (via MULTICAST) – **Message 0x01**
* junta-se ao grupo de conversação
* ex: JOIN [Campiolo]

```
Mensagem {
    type: 0x01
    source: "Campiolo"
    message: ""
}
```

-- **JOINACK [apelido-de-quem-recebeu-join]** – **Message 0x02**
* resposta ao JOIN para possibilitar a manutenção da lista de usuários ativos (via UDP)
* ex: JOINACK [Dracula]

```
Mensagem {
    type: 0x02
    source: "Dracula"
    message: ""
}
```

-- **MSG [apelido] "texto"** – **Message 0x03**
* mensagem enviada a todos os membros do grupo pelo IP 225.1.2.3 e porta 6789 (via MULTICAST)
* ex: MSG [Campiolo] "Bom dia pessoal."

```
Mensagem {
    type: 0x03
    source: "Campiolo"
    message: "Bom dia pessoal."
}
```

-- **MSGIDV FROM [apelido] TO [apelido] "texto"** (via UDP) – **Message 0x04**
* mensagem enviada a um membro do grupo para ser recebida na porta 6799
* ex: MSGIDV FROM [Campiolo] TO [Bicho Papão] "Hoje é sexta-feira 13."

```
Mensagem {
    type: 0x04
    source: "Campiolo"
    message: "Hoje é sexta-feira 13."
}
```

-- **LEAVE [apelido]** – **Message 0x05**
* deixa o grupo de conversação
* mensagem enviada ao grupo informando que deixou a conversação.
* ex: LEAVE [Campiolo]

```
Mensagem {
    type: 0x05
    source: Campiolo
    message: ""
}
```