```java
public final class Packet {
    public static final int HEADER_LEN = 13;//packet包头协议长度

    public static final byte FLAG_CRYPTO = 0x01;//packet包启用加密
    public static final byte FLAG_COMPRESS = 0x02;//packet包启用压缩
    public static final byte FLAG_BIZ_ACK = 0x04;
    public static final byte FLAG_AUTO_ACK = 0x08;

    public static final byte HB_PACKET_BYTE = -33;
    public static final Packet HB_PACKET = new Packet(Command.HEARTBEAT);

    public byte cmd; //命令
    public short cc; //校验码 暂时没有用到
    public byte flags; //特性，如是否加密，是否压缩等
    public int sessionId; // 会话id
    public byte lrc; // 校验，纵向冗余校验。只校验header
    public byte[] body;

    public Packet(byte cmd) {
        this.cmd = cmd;
    }

    public Packet(byte cmd, int sessionId) {
        this.cmd = cmd;
        this.sessionId = sessionId;
    }

    public Packet(Command cmd) {
        this.cmd = cmd.cmd;
    }

    public Packet(Command cmd, int sessionId) {
        this.cmd = cmd.cmd;
        this.sessionId = sessionId;
    }

    public int getBodyLength() {
        return body == null ? 0 : body.length;
    }

    public void addFlag(byte flag) {
        this.flags |= flag;
    }

    public boolean hasFlag(byte flag) {
        return (flags & flag) == flag;
    }

    public short calcCheckCode() {
        short checkCode = 0;
        if (body != null) {
            for (int i = 0; i < body.length; i++) {
                checkCode += (body[i] & 0x0ff);
```

```
            }
        }
        return checkCode;
    }

    public byte calcLrc() {
        byte[] data = ByteBuffer.allocate(HEADER_LEN - 1)
                .putInt(getBodyLength())
                .put(cmd)
                .putShort(cc)
                .put(flags)
                .putInt(sessionId)
                .array();
        byte lrc = 0;
        for (int i = 0; i < data.length; i++) {
            lrc ^= data[i];
        }
        return lrc;
    }

    public boolean validCheckCode() {
        return calcCheckCode() == cc;
    }

    public boolean validLrc() {
        return (lrc ^ calcLrc()) == 0;
    }

    @Override
    public String toString() {
        return "Packet{" +
                "cmd=" + cmd +
                ", cc=" + cc +
                ", flags=" + flags +
                ", sessionId=" + sessionId +
                ", lrc=" + lrc +
                ", body=" + (body == null ? 0 : body.length) +
                '}';
    }
}
```

```java
public class HelloWorld {
    public static final byte FLAG_CRYPTO = 0x01;//packet包启用加密
    public static final byte FLAG_COMPRESS = 0x02;//packet包启用压缩
    public static final byte FLAG_BIZ_ACK = 0x04;
    public static final byte FLAG_AUTO_ACK = 0x08;
    public byte flags;
    public void addFlag(byte flag) {//将flag对应位的值设为1
        this.flags |= flag;
    }
    public boolean hasFlag(byte flag) { //判断相应位置是否为1
        return (flags & flag) == flag;
    }
    public byte getFlag(){
        return this.flags;
    }
    public static void main(String []args) {
        HelloWorld hh=new HelloWorld();
        hh.addFlag(FLAG_COMPRESS);
        hh.addFlag(FLAG_CRYPTO);
        hh.addFlag(FLAG_AUTO_ACK);
        System.out.println(hh.getFlag());
        System.out.println(hh.hasFlag(FLAG_COMPRESS));
        System.out.println(hh.hasFlag(FLAG_BIZ_ACK));
    }
}
```

上面运行的结果，返回：

11

true

false