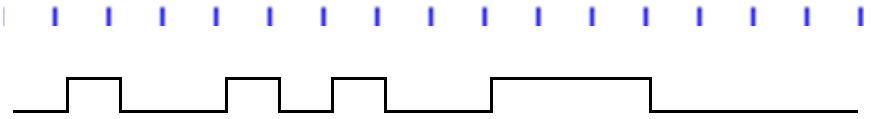
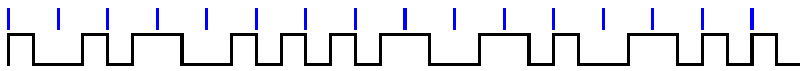
Beispielkausur Informatik SII 90’

Name: _____

1) Bitübertragungsschicht
   a. Das folgende Signal stellt eine 16 Bit lange Bit-Folge dar, die NRZI-codiert ist. Ermitteln Sie die Bitfolge an (unter der Annahme, dass der Pegel am Anfang auf Null stand).



   b. Das folgende Signal ist Manchester-codiert. Ermitteln Sie die zugehörige Bitfolge.



   c. Nennen Sie den einfachsten Code, den wir kennengelernt haben und geben Sie seinen Hauptnachteil an.

2) Berechnen Sie für die folgenden Werte die Prüfsumme und die gewichtete Prüfsumme (n=4):

   a. 11011
   b. 10101
   c. 01010

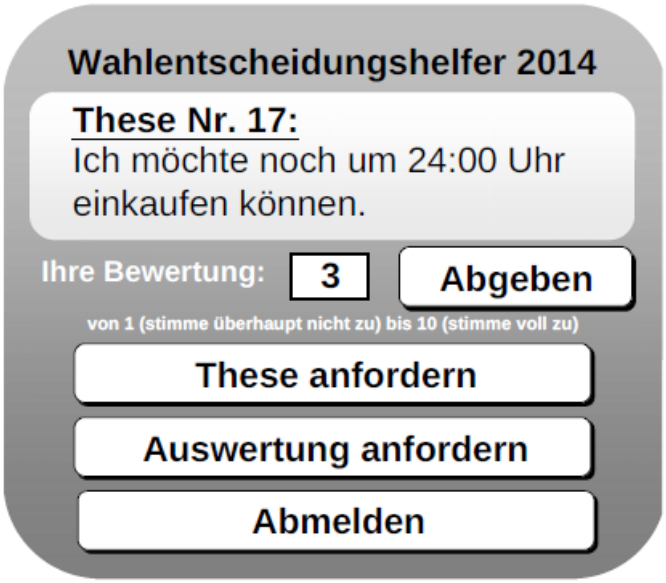
3) Gegeben sind die folgenden IP-Adressen und Netzmasken. Ermitteln Sie die dazugehörigen Netzwerkadressen, die Router-Adresse sowie die Broadcast-Adresse.

   a. 84.63.115.0 Netmask 255.255.0.0
   b. 137.226.12.118 Netmask 255.224.0.0
   c. 192.168.108.97 Netmask 255.255.255.252
   d. 85.214.248.209 Netmask 255.255.240.0

4) Netzwerkprotokolle

Der Sozialwissenschaftenleistungskurs hat im Unterricht den **Wahl-O-Mat** thematisiert. Er ist eine seit 2002 von der Bundeszentrale für politische Bildung betriebene Webpräsenz für interaktive Online-Wahlinformationen.

Nun plant der Sozialwissenschaftenkurs zusammen mit dem Informatikkurs eine eigene Version dieses Wahlentscheidungshelfers zu entwickeln. Die Schülerinnen und Schüler des Sozialwissenschaftenkurses erstellen dafür 20 relevante Thesen, welche auf einer Skala von 1 (stimme überhaupt nicht zu) bis 10 (stimme voll zu) bewertet werden können. Das Überspringen einer These soll nicht möglich sein. Der Wahlentscheidungshelfer soll sich dann nach den Wünschen des Sozialwissenschaftenkurses wie in der Abbildung präsentieren:



Der Informatikkurs übernimmt die technische Umsetzung des Projekts in Form einer Client-Server-Anwendung. Die Idee ist, dass sich ein Teilnehmer zunächst am Server anmelden kann. Danach werden dem Teilnehmer die 20 Thesen in zufälliger Reihenfolge zur Bewertung präsentiert. Zum Abschluss erhält der Teilnehmer auf Anforderung eine Auswertung in Form einer Wahlempfehlung für die beste Übereinstimmung eines Wahlprogramms mit der Bewertung der Thesen.

In einer ersten Entwicklungsphase kümmern sich die Schülerinnen und Schüler des Informatikkurses um ein Kommunikationsprotokoll und einigen sich auf das Folgende:

Client sendet an Server	Server sendet an Client
stellt eine Verbindung zum Server her...	ANGEMELDET
GIBTHESE	THESE: <Nr>: <Thesentext>
BEWERTUNG: <Zahl>	OK: These <Nr> mit <Zahl> bewertet.
ABMELDEN	ABGEMELDET ... und trennt die Verbindung.

a) Wir betrachten das folgende Szenario:

Der Schüler Hermann meldet sich beim Server an. Er fordert die erste These an, welche zufällig die These mit der Nr. 17: „Ich möchte noch um 24:00 Uhr einkaufen können.“ ist. Er findet diese Möglichkeit sehr wichtig und bewertet die These mit 10. Danach fordert er die

nächste These an, dies ist die These mit der Nr. 2 „Die Studiengebühren müssen weg!“. Hermann hat keine Lust mehr und meldet sich ab, ohne eine Bewertung für These 2 abzugeben.

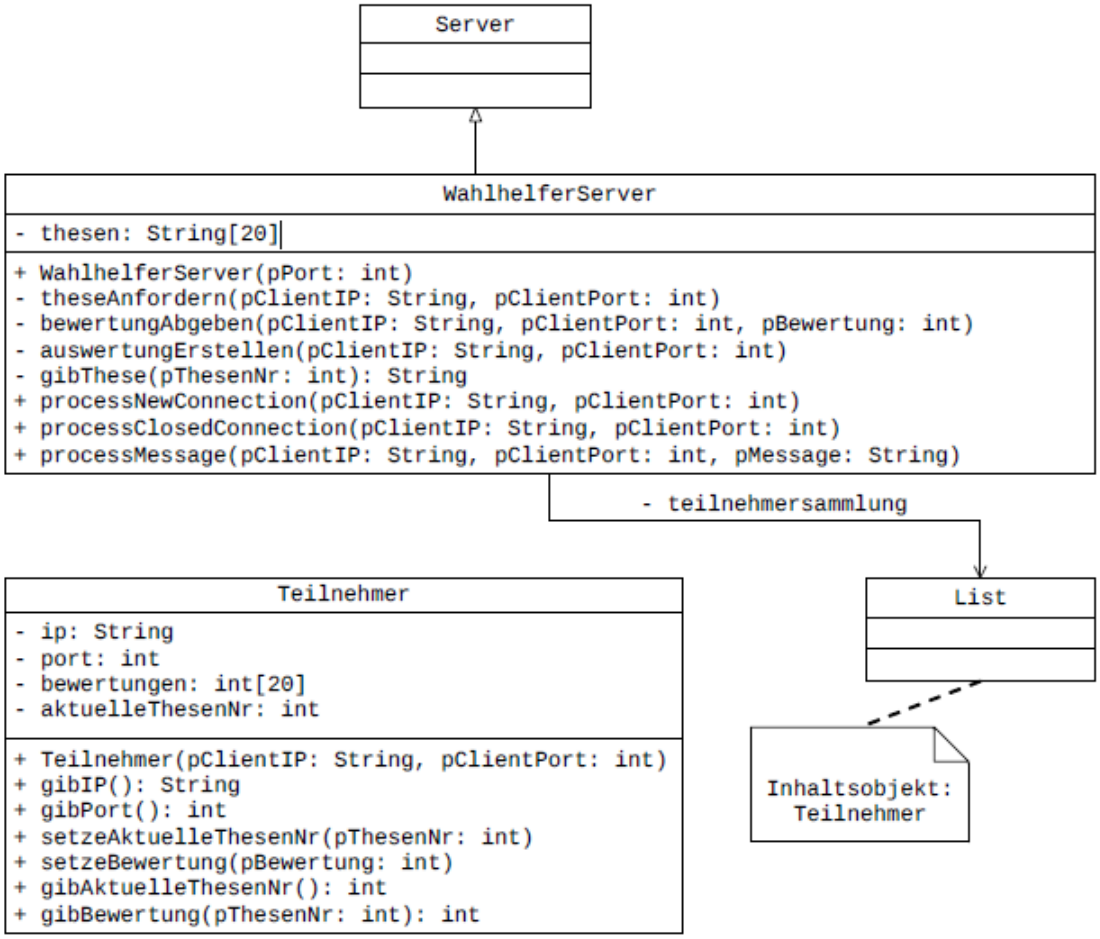
Gib tabellarisch auf der Grundlage des oben angegebenen Protokolls alle von der Anmeldung bis zur Abmeldung zwischen Client und Server ausgetauschten Nachrichten an.

Erweitern Sie das allgemeine Protokoll aus um eine Serverantwort, falls der Client

- eine Bewertung sendet, die nicht im Bereich von 1 bis 10 liegt.
- eine Bewertung sendet, obwohl er noch keine neue These angefordert hat.
- eine weitere These abruft, obwohl er schon alle Thesen bewertet hat.

Erweitern Sie das angegebene Protokoll um eine Kommunikationsregel, mit deren Hilfe ein Client die Möglichkeit hat, eine Wahlempfehlung anzufordern. Berücksichtigen Sie auch den Fall, dass der Client noch nicht alle Thesen bewertet hat und eine Auswertung somit nicht sinnvoll ist.

In einer weiteren Entwicklungsphase kümmern sich die Schülerinnen und Schüler des Informatikkurses um den Server des Wahlentscheidungshelfers. Hier das Implementationsdiagramm:



Im Anhang ist ein Auszug der Dokumentation für die Klassen *WahlhelferServer* und *Teilnehmer*.

b) Eine Schülerin hat folgende Methode für die Klasse *WahlhelferServer* erstellt:

```
1.  public void wastueich(String pClientIP, int pClientPort) {
2.    Teilnehmer tn = null;
3.    teilnehmersammlung.toFirst();
4.    while (teilnehmersammlung.hasAccess()){
5.      Teilnehmer tmpTn = (Teilnehmer)
    Teilnehmersammlung.getObject();
6.      if (pClientIP.equals(tmpTn.gibIP()) &&
                                    pClientPort == tmpTn.gibPort())
                                 {
7.        tn = tmpTn;
8.      }
9.      teilnehmersammlung.next();
10.   }
11.   if (tn != null) {
12.     int nummer = (int) (Math.random() * 20);
13.     while (tn.gibBewertung(nummer) > 0) {
14.       nummer = nummer + 1;
15.       if (nummer == 20) {
16.         nummer = 0;
17.       }
18.     }
19.     tn.setzeAktuelleThesenNr(nummer);
20.     send(pClientIP, pClientPort, "THESE: " + nummer + ": " +
21.     gibThese(nummer));
22.   }
23. }
```

*Hinweis: Der Ausdruck (int) (Math.random() * 20) in Zeile 12 liefert eine Zufallszahl im Bereich von 0 bis 19.*

Erläutern Sie die Funktionalität der Schleife in den Zeilen 4 bis 10 sowie der bedingten Anweisung in den Zeilen 11 bis 21. Beschreiben Sie dabei auch die Funktionalität der gesamten Methode.

Begründen Sie anhand des Quellcodes, dass die Methode fehlerhaft arbeitet, wenn der Teilnehmer bereits alle Thesen bewertet hat.

c) Der Client soll eine Fehlermeldung erhalten, falls er versucht, eine weitere These abzurufen, obwohl er schon alle Thesen bewertet hat, bzw. falls er versucht, eine Auswertung seiner Bewertungen anzufordern, obwohl er noch nicht alle Thesen bewertet hat. Deshalb ist eine Methode

```
private boolean alleThesenBewertet(Teilnehmer pTeilnehmer)
```

der Klasse *WahlhelferServer* gesucht, das Folgende leistet:
Falls der Teilnehmer *pTeilnehmer* bereits alle Thesen bewertet hat, so gibt die Methode den Wert *true* zurück. Ist dies nicht der Fall, so soll die Methode den Wert *false* zurückgeben.
Implementieren Sie die private Methode *alleThesenBewertet* der Klasse *WahlhelferServer*.

Klassendokumentationen

Klasse Teilnehmer (Teil)
Diese Klasse verwaltet die Netzwerkeigenschaften des Clients sowie die Gesamtheit aller Bewertungen eines Teilnehmers, der sich an dem Server angemeldet hat. Außerdem wird für einen Teilnehmer verwaltet, welche These als aktuell zu beantwortende These festgelegt ist. Die Klasse stellt u. a. folgende Methoden zur Verfügung:

- **`void setzeAktuelleThesenNr(int pThesenNr)`**
  Die These mit der Nummer `pThesenNr` wird als aktuell zu bewertende These festgelegt. Gibt es die These mit der Nummer `pThesenNr` nicht, wird keine Änderung durchgeführt.

- **`void setzeBewertung(int pBewertung)`**
  Die aktuell zu bewertende These wird mit der Zahl `pBewertung` bewertet. Der Wert für `pBewertung` muss im Intervall 1 bis 10 (einschließlich) liegen, ansonsten bleibt der zu setzende Wert unverändert.

- **`int gibAktuelleThesenNr()`**
  Die Nummer der These, welche zuvor festgelegt wurde, wird zurückgegeben. Wird die Zahl -1 zurückgegeben, so bedeutet dies, dass noch keine These als aktuell zu bewertende These festgelegt wurde.

- **`int gibBewertung(int pThesenNr)`**
  Die zur These mit der Nummer `pThesenNr` gespeicherte Bewertung wird zurückgegeben. Wird die Bewertung 0 zurückgegeben, so bedeutet dies, dass für diese These noch keine Bewertung abgegeben wurde. Eine Überprüfung, ob `pThesenNr` einen gültigen Wert enthält, erfolgt hier nicht.

Klasse WahlhelferServer (Teil)
Diese Klasse verwaltet die Thesen sowie die am Server angemeldeten Teilnehmer. Sie stellt u. a. folgende Methoden zur Verfügung:

- **`void theseAnfordern(String pClientIP, int pClientPort)`**
  Für den Teilnehmer mit der IP `pClientIP` und dem Port `pClientPort` wird eine zufällige, noch nicht bewertete Thesennummer ausgewählt und festgelegt. Der Server schickt dem Client des Teilnehmers über das Netzwerk eine Rückmeldung der Form `THESE: <Nr>: <Thesentext>`. Gibt es keine unbewertete Thesennummer, so wird entsprechend dem in Aufgabenteil a) zu entwickelnden Protokoll eine Fehlermeldung zurückgegeben.

- **`void bewertungAbgeben(String pClientIP, int pClientPort, int pBewertung)`**
  Für den Teilnehmer mit der IP `pClientIP` und dem Port `pClientPort` wird die Bewertung `pBewertung` für die aktuell festgelegte Thesennummer gespeichert. Der Server schickt dem Client des Teilnehmers über das Netzwerk eine Rückmeldung der Form `OK: These <Nr> mit <Zahl>` bewertet. Hat der Teilnehmer noch keine neue These angefordert oder eine Bewertungszahl außerhalb der Bewertungsskala eingegeben, so wird entsprechend dem in der Aufgabe ergänzten Protokoll eine Fehlermeldung zurückgegeben.

- **`void auswertungErstellen(String pClientIP, int pClientPort)`**
  Für den Teilnehmer mit der IP `pClientIP` und dem Port `pClientPort` wird eine Auswertung in Textform erstellt und über das Netzwerk entsprechend dem in der Aufgabe ergänzten Protokoll an dessen Client geschickt. Hat der Teilnehmer noch nicht alle Thesen bewertet, so wird entsprechend dem in der Aufgabe ergänzten Protokoll eine Fehlermeldung zurückgegeben.

- **`String gibThese(int pThesenNr)`**
  Die Anfrage liefert den Text der These zur übergebenen Thesennummer `pThesenNr`.

Erwartungshorizont

Aufgabe	Erwartung
1a.	Angabe der Bitfolge 0110 1111 0100 1000
1b.	Angabe der Bitfolge 0110 1111 0100 1000
1c.	Nennung NRZ, Takt als Nachteil
2	Angabe v. Prüfsummen und gew. Prüfsummen
3)	Ermittelt jeweils Netzwerkadresse, Router-Adresse und Broadcast-Adresse korrekt gemäß den Vorgaben.
4a.	Tabellarische Darstellung des Netzwerkverkehrs Erweiterung der Tabelle zum Kommunikationsprotokoll umd die geforderten Elemente.
4b.	Erläuterung • der Schleife (Sucher des Teilehmerobjektes in der Liste) • der bedingten Anweisung (Überprüfung auf geeignetes Teilnehmerobjekt, zufällige Auswahl einer These und senden einer Nachricht.) • der Funktionalität der Gesamtmethode (Übertragung einer neuen These) • Erkennen, dass die Schleife nicht terminiert, wenn alle Thesen beantwortet sind.
4.c	Implementation der Methode in JAVA: Durchzählen aller Bewertungen für einen Teilnehmer und Vergleich mit 20.