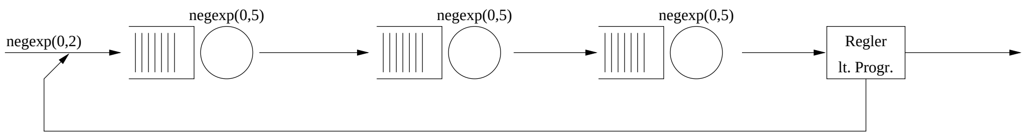
Andreas van Almsick, Marcus Völker
Lehrstuhl Informatik IV, Universität Dortmund

10. Juni 1998

# Übung 7

(18) Gegeben ist das folgende System:




Die Kunden betreten das System und durchlaufen zunächst die drei Stationen. Je nach Reglerentscheidung verlassen sie anschließend das System oder werden wieder zum Anfang zurück geschickt

Bestimmen Sie die mittlere Anzahl von Kunden im System in der stationären Phase.

(i) Starten Sie das System im leeren Zustand. Ermitteln Sie $t_c$ mit einer modifizerten "batch-means" Methode. Betrachten Sie die jeweils 10 letzten Batches. Der $i$-te Batch soll dabei die mittlere Population im Zeitintervall $(0, i \cdot t]$ erfassen. Wählen Sie $\epsilon = 0,3$. *4 Punkte*

(ii) Skizzieren Sie die Trajektorien von 10 Läufen. Starten Sie mit unterschiedlichen Seeds! Legt die graphische Darstellung die Annahme von Stationarität ab dem von Ihnen ermittelten Zeitpunkt $t_c$ nahe? Wählen Sie eine angemessene Simulationsdauer. *4 Punkte*

(iii) Ermitteln Sie die mittlere Anzahl von Kunden im eingeschwungenen System. Die Abweichung vom wirklichen Wert soll mit 90%-iger Wahrscheinlichkeit nicht größer als 10% Ihres Schätzwertes sein. Nehmen Sie eine Normalverteilung für die Schätzwerte an. *4 Punkte*

Den auf den folgenden Seiten abgedruckten Simula-Quellcode für einen Simulator des Systems finden sie unter

`~sim000/uebung7`

. Erweitern Sie diesen Simulator.

Andreas van Almsick, Marcus Völker
Lehrstuhl Informatik IV, Universität Dortmund

10. Juni 1998

```
% ***************************************************************************
% Aufgabe 18)
% ***************************************************************************

BEGIN

% ***************************************************************************
% ***************************************************************************
% PROCEDURE Simulationslauf
%
% Die Prozedur Simulationslauf startet einen Simulationslauf, der nach der
% Bedienung des Kunden mit der Nummer <Zielnummer> abgebrochen wird. Ausgegeben
% werden bei jeder Kundenabfertigung die entsprechenden Mittelwerte.
% ***************************************************************************
% ***************************************************************************

  REAL PROCEDURE Simulationslauf (Zielnummer, Seed); INTEGER Zielnummer, Seed;
  SIMULATION BEGIN

%   *************************************************************************
%   *************************************************************************
%   LINK CLASS KundenanzahlMonitorKlasse
%
%   Die KundenanzahlMonitorKlasse dient zur Berechnung der mittleren
%   Kundenanzahl im System über die gesamte bisherige Simulationszeit.
%   *************************************************************************
%   *************************************************************************

    CLASS KundenanzahlMonitorKlasse;
    BEGIN
      REAL LetzterBerechnungszeitpunkt, Verweilzeitensumme;

%     ***********************************************************************
%     PROCEDURE Verweilzeitensummation
%
%     Diese Prozedur summiert die Verweilzeiten auf, indem auf dem alten Wert
%     für jeden Kunden im System einmal die verstrichene Zeit seit der letzten
%     Summation aufaddiert wird.
%     ***********************************************************************

      PROCEDURE Verweilzeitensummation (Kundenanzahl); INTEGER Kundenanzahl;
      BEGIN
        Verweilzeitensumme:= Verweilzeitensumme + Kundenanzahl *
                                    (TIME - LetzterBerechnungszeitpunkt);
        LetzterBerechnungszeitpunkt:= TIME;
      END;
```

Andreas van Almsick, Marcus Völker
Lehrstuhl Informatik IV, Universität Dortmund

10. Juni 1998

```
%    ***********************************************************************
%    PROCEDURE MittlereKundenanzahl
%
%    Diese Prozedur berechnet die mittlere Kundenanzahl im System über die
%    gesamte bisherige Simulationszeit.
%    ***********************************************************************

     REAL PROCEDURE MittlereKundenanzahl (Kundenanzahl); INTEGER Kundenanzahl;
     BEGIN
       Verweilzeitensumme:= Verweilzeitensumme + Kundenanzahl *
                                   (TIME - LetzterBerechnungszeitpunkt);
       MittlereKundenanzahl:= Verweilzeitensumme / TIME;
       LetzterBerechnungszeitpunkt:= TIME;
     END;

%    ***********************************************************************
%    KundenanzahlMonitorKlasse-Hauptprozedur
%
%    Als Startwert für die Variable <LetzterBerechnungszeitpunkt> wird die
%    aktuelle Zeit übergeben.
%    ***********************************************************************

     Verweilzeitensumme:= 0.0;
     LetzterBerechnungszeitpunkt:= TIME;
   END;

%  *************************************************************************
%  *************************************************************************
%  LINK CLASS KundenKlasse
%
%  Die KundenKlasse beschreibt das Verhalten eines Kunden. Für jeden Kunden
%  wird die <Ankunftszeit> festgehalten. Nach dem Verlassen eines Kunden wird
%  der Mittelwert neu berechnet. Jeder Kunde verarbeitet seine eigene Ankunft
%  und seine eigene Abfertigung.
%  *************************************************************************
%  *************************************************************************

   PROCESS CLASS KundenKlasse (Nummer); INTEGER Nummer;
   BEGIN
     REAL Ankunftszeit, Verweilzeit;
```

Andreas van Almsick, Marcus Völker
Lehrstuhl Informatik IV, Universität Dortmund

10. Juni 1998

```
%     ***********************************************************************
%     PROCEDURE Ankunft
%
%     Setzt die <Ankunftszeit> für den neuen Kunden auf die aktuelle Zeit und
%     aktualisiert die Verweilzeitensumme.
%     ***********************************************************************

      PROCEDURE Ankunft;
      BEGIN
        Ankunftszeit:= TIME;
        KundenanzahlMonitor.Verweilzeitensummation (Kundenanzahl);
        Kundenanzahl:= Kundenanzahl + 1;
        OUTTEXT ("Kunde ");
        OUTINT (Nummer, 6);
        OUTTEXT (" kommt: ");
        OUTFIX (TIME, 2, 9);
        OUTIMAGE;
      END;

%     ***********************************************************************
%     PROCEDURE Bedienung
%
%     Belegt die übergebene <Station>, aktiviert ihren Prozess und fügt sich in
%     die zugehörige Kundenliste ein.
%     ***********************************************************************

      PROCEDURE Bedienung (Station); REF (BedienKlasse) Station;
      BEGIN
        ACTIVATE (Station);
        WAIT (Station.Kundenliste);
        OUT;
      END;

%     ***********************************************************************
%     PROCEDURE Durchlauf
%
%     Schickt den Kunden nacheinander durch alle drei Stationen.
%     ***********************************************************************

      PROCEDURE Durchlauf;
      BEGIN
        Bedienung (Station1);
        Bedienung (Station2);
        Bedienung (Station3);
      END;
```

Andreas van Almsick, Marcus Völker
Lehrstuhl Informatik IV, Universität Dortmund

10. Juni 1998

```
%     ************************************************************************
%     PROCEDURE Regler
%
%     Simuliert den Regler am Ende des Systems. Zurückgegeben wird ein von der
%     bisherigen mittleren Kundenanzahl abhängiger Zufallswert.
%     ************************************************************************

      BOOLEAN PROCEDURE Regler;
      BEGIN
        REAL p;

        p:= KundenanzahlMonitor.MittlereKundenanzahl (Kundenanzahl) / 40;
        IF p > 1.0
          THEN p:= 1.0;
        Regler:= DRAW (1 - p, Seed);
      END;

%     ************************************************************************
%     PROCEDURE Abfertigung
%
%     Aktualisiert die Verweilzeitensumme und den Wert für die mittlere
%     Anzahl der Kunden und gibt die mittlere Kundenanzahl mit der aktuellen
%     Simulationszeit aus.
%     ************************************************************************

      PROCEDURE Abfertigung;
      BEGIN
        KundenanzahlMonitor.Verweilzeitensummation (Kundenanzahl);
        OUTTEXT ("Kunde ");
        OUTINT (Nummer, 6);
        OUTTEXT (" geht: ");
        OUTFIX (TIME, 2, 9);
        OUTIMAGE;
        OUTTEXT ("Bis zum Zeitpunkt ");
        OUTFIX (TIME, 2, 9);
        OUTTEXT (" folgende mittlere Kundenanzahl: ");
        OUTFIX (KundenanzahlMonitor.MittlereKundenanzahl (Kundenanzahl), 2, 8);
        OUTIMAGE;
        Kundenanzahl:= Kundenanzahl - 1;
      END;

%     ************************************************************************
%     KundenKlasse-Hauptprozedur
%
%     Jeder neu erzeugte Kunde löst zunächst seine Ankunft aus. Anschließend
%     durchläuft er das System. Weitere Durchläufe werden in Abhängigkeit vom
%     durch den Regler erzeugten Zufallswert absolviert. Nach dem letzten
%     Durchlauf verläßt der Kunde das System.
%     ************************************************************************

      Ankunft;
      Durchlauf;
      WHILE (Regler) DO BEGIN
        Durchlauf;
      END;
      Abfertigung;
    END;
```

Andreas van Almsick, Marcus Völker
Lehrstuhl Informatik IV, Universität Dortmund

10. Juni 1998

```
%   ***************************************************************************
%   ***************************************************************************
%   PROCESS CLASS BedienKlasse
%
%   Die BedienKlasse beschreibt das Bedienverhalten der Bedienstationen. Solange
%   Kunden warten, werden diese für eine zufällige der <Bedienrate>
%   entsprechenden Dauer bedient und dann abgefertigt. Ansonsten wird dieser
%   Prozess passiviert. Jede Station besitzt dafür eine eigene Kundenliste.
%   ***************************************************************************
%   ***************************************************************************

    PROCESS CLASS BedienKlasse (Bedienrate, Seed); REAL Bedienrate;
                                                                 INTEGER Seed;
    BEGIN
      REF (HEAD) Kundenliste;
      REF (KundenKlasse) Kunde;

      Kundenliste:- NEW HEAD;
      WHILE (TRUE) DO BEGIN
        WHILE NOT Kundenliste.EMPTY DO BEGIN
          HOLD (NEGEXP (Bedienrate, Seed));
          Kunde:- Kundenliste.FIRST;
          ACTIVATE (Kunde);
        END;
        PASSIVATE;
      END;
    END;

%   ***************************************************************************
%   ***************************************************************************
%   PROCESS CLASS AnkunftsKlasse
%
%   Die AnkunftsKlasse generiert und aktiviert solange neue Kunden, bis der
%   Kunde mit der Nummer <Zielnummer> eingetroffen ist. Kontrolliert wird dies
%   durch die Variable <Beenden>. Die Zwischenankunftszeiten der Kunden sind
%   entsprechend der <Ankunftsrate> zufällig verteilt.
%   ***************************************************************************
%   ***************************************************************************

    PROCESS CLASS AnkunftsKlasse (Ankunftsrate, Seed); REAL Ankunftsrate;
                                                                 INTEGER Seed;
    BEGIN
      INTEGER Kundennummer;
      REF (KundenKlasse) Kunde;

      Kundennummer:=0;
      WHILE (NOT Beenden) DO BEGIN
        HOLD (NEGEXP (Ankunftsrate, Seed));
        Kundennummer:= Kundennummer + 1;
        Kunde:- NEW KundenKlasse (Kundennummer);
        ACTIVATE (Kunde);
        IF (Kundennummer = Zielnummer)
          THEN Beenden:= TRUE;
      END;
    END;
```

Andreas van Almsick, Marcus Völker
Lehrstuhl Informatik IV, Universität Dortmund

10. Juni 1998

```
%   ************************************************************************
%   ************************************************************************
%   Simulationslauf-Hauptprozedur
%
%   Simuliert wird ein Wartesystem mit drei Stationen mit einer jeweiligen
%   Bedienrate von 0,5 und eine reglergesteuerte Rückführung der Kunden zum
%   Anfang. Die Ankunftsrate beträgt 0,4. Die Übergabe unterschiedlicher Seeds
%   dient dazu, unterschiedliche Zufallswerte zu erhalten. Die Simulation dauert
%   an, bis der 1000. Kunde abgefertigt worden ist. Kontrolliert wird dies über
%   die BOOLEAN-Variable <Beenden>. Das HOLD-Statement ist notwendig, damit die
%   Simulation nicht sofort nach dem Start wieder beendet wird. Sollten 1000000
%   Zeiteinheiten nicht ausreichen, um 1000 Kunden zu bearbeiten, so muß dieser
%   Wert erhöht werden.
%   ************************************************************************
%   ************************************************************************

    BOOLEAN Beenden;
    INTEGER Kundenanzahl;
    REF (AnkunftsKlasse) Ankunft;
    REF (BedienKlasse) Station1, Station2, Station3;
    REF (KundenanzahlMonitorKlasse) KundenanzahlMonitor;

    Beenden:= FALSE;
    Kundenanzahl:= 0;
    Ankunft:- NEW AnkunftsKlasse (0.01, 10 * seed);
    Station1:- NEW BedienKlasse (0.5, seed);
    Station2:- NEW BedienKlasse (0.5, 5 * seed);
    Station3:- NEW BedienKlasse (0.5, 7 * seed);
    KundenanzahlMonitor:- NEW KundenanzahlMonitorKlasse;
    ACTIVATE (Ankunft);
    HOLD (1000000);
  END;

% **************************************************************************
% **************************************************************************
% Simulations-Hauptprozedur
%
% Simuliert wird ein Wartesystem mit drei Stationen mit einer jeweiligen
% Bedienrate von 0,5 und eine reglergesteuerte Rückführung der Kunden zum
% Anfang. Die Ankunftsrate beträgt 0,4. Ein Berechnungslauf wird so lange
% ausgeführt, bis 1000 Kunden abgefertigt worden sind.
% **************************************************************************
% **************************************************************************

  INTEGER Berechnungslauf;

  FOR Berechnungslauf:= 1 STEP 1 UNTIL 1 DO BEGIN
    Simulationslauf (1000, Berechnungslauf);
  END;
END;
```

Andreas van Almsick, Marcus Völker
Lehrstuhl Informatik IV, Universität Dortmund

10. Juni 1998

**Den kommentierten (!) Quelltext Ihrer Lösung zu dieser Aufgabe senden Sie bitte bis zum 17. Juni 1998, 12.00 Uhr per e-mail an sim000@gallium. Eine Abgabe in den Briefkasten ist für diesen Aufgabenteil nicht erforderlich. Ihre Lösung zu den anderen Aufgabenbestandteilen dieser Aufgabe können Sie bis zum 17. Juni 1998 um 12.00 Uhr in den Briefkasten 18 im Pavillon 6 einwerfen.**