WinGPSS

Die ersten Stunden der Simulationsausbildung

Richard Born

Northern Illinois University

Ingolf Ståhl

Handelhochschule Stockholm

Henry Herper

Otto-von-Guericke-Universität Magdeburg

Kapitel 1

Dieses Arbeitsmaterial gibt eine grundlegende Einführung in die Simulationstechnik unter Verwendung der Simulationssprache WinGPSS. In diesen sieben Kapiteln erstellen wir schrittweise ein Simulationsmodell eines einfachen Friseurladens, in dem nur ein Friseur arbeitet und ein Kundentyp bedient wird. Die **Kunden** werden für dieses Modell als **temporäre Modellelemente** bezeichnet, da sie den Friseurladen betreten und nach der Bedienung wieder verlassen. Der Friseur, auch als **Bedieneinrichtung** bezeichnet, ist in diesem Modell ein **permanentes Modellelement**, da er für die gesamte Zeit der Simulation im System ist. Die Zeitpunkte, zu denen Kunden den Laden betreten, und die notwendigen Zeiten für das Schneiden der Haare sind in diesem Modell zufällig gewählt. Das Ziel unseres Simulationsexperimentes besteht darin, zu ermitteln, wie lang die Warteschlangen werden und wie lange Kunden auf einen Haarschnitt warten müssen.

Wir erstellen das Simulationsmodell schrittweise. Unsere ersten beiden Programme beschreiben nur den einfachen Fall, dass die Kunden das System betreten und sofort wieder verlassen. Dies kommt z.B. an einem Drehkreuz vor, welches zur Eingangs- oder Ausgangskontrolle genutzt wird. Im ersten Programm gehen wir zur weiteren Vereinfachung davon aus, dass die Kunden nicht zufällig ankommen, sondern dass die Zeit zwischen der Ankunft zweier Kunden, die auch als **Zwischenankunftszeit** bezeichnet wird, konstant ist. In diesem Programm beenden wir die Simulation, wenn 50 Kunden das System betreten und wieder verlassen haben. Das System, welches mit Programm PRO1 nachgebildet, wird im Bild 1 beschrieben.

- Das System ist ein Kiosk mit einem Drehkreuz. Am Kiosk werden Marken verkauft, die das Drehkreuz freigeben.
- Genau alle 18 Minuten kommt ein Kunde an und kauft genau eine Marke.
- Am Morgen eines Tages sind genau 50 Marken vorhanden.
- Sind alle 50 Marken verkauft, so muss der Kiosk für diesen Tag schließen.

Bild 1: Das System für Programm PRO1

Alle 18 Minuten erreichen Kunden den Kiosk um eine Marke zu kaufen. Jeder Kunde kauft genau eine Marke. Wenn alle 50 Marken verkauft sind, muss der Kiosk schließen, da keine Marken mehr verkauft werden können.

Dieses Modell wird nun mit der Simulationssprache WinGPSS erstellt. Dazu wird die Datei wingpss.exe gestartet.

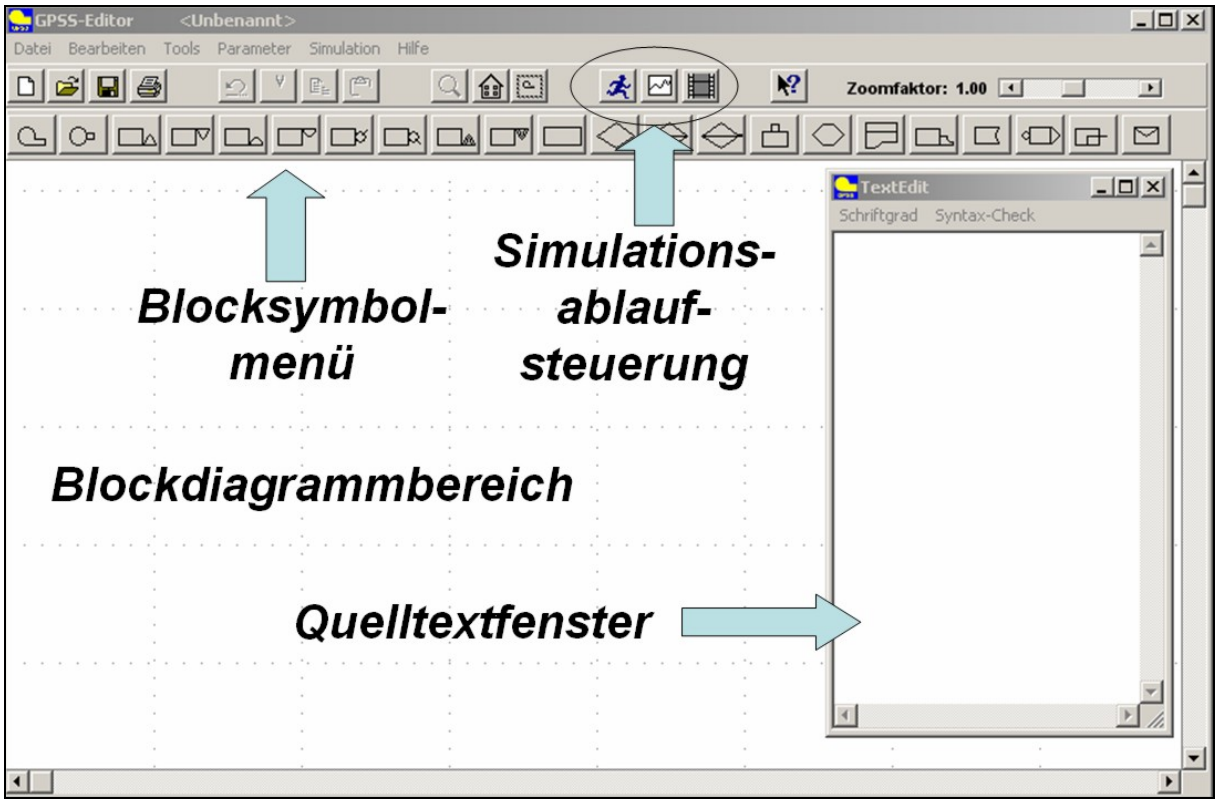


Bild 2: Graphische Oberfläche des WinGPSS

Der Bildschirm der WinGPSS-Nutzeroberfläche ist in mehrere Bereiche aufgeteilt. Im Blocksymbolmenü, welches Symbole für 22 GPSS-Blöcke enthält, werden die gewünschten Blocksymbole ausgewählt und im Blockdiagrammbereich platziert. Wird der Mauszeiger auf ein Blocksymbol positioniert, so erscheint der Name des Blockes. Der aus dem Blockdiagramm erzeugte Quelltext wird im Quelltextfenster dargestellt.

Als erstes Symbol finden wir im Blockmenü den GENERATE-Block. Dieser Block wird in jedem GPSS-Programm benötigt. Mit ihm können wir die Kunden (Forderungen) in das System bringen. Durch Anklicken des Blocksymbols mit der rechten Maustaste verändert der Mauszeiger seine Form und zeigt das gewählte Symbol an. Anschließend kann es im Blockdiagrammbereich platziert werden. Dazu steht jede zweite Spalte zur Verfügung. Wird das Blocksymbol mit der linken Maustaste angeklickt, so erfolgt eine sofortige Positionierung auf die aktuelle Position im Blockdiagrammbereich. Die aktuelle Position ist durch einen roten Punkt dargestellt und kann durch Klicken auf den Blockdiagrammbereich festgelegt werden. Beim Setzen eines Blockes wird die aktuelle Position um einen Block weiter gesetzt.

Für dieses Programm benötigen wir weiterhin einen TERMINATE-Block um die Kunden aus dem System zu entfernen. Wir klicken auf das Blocksymbol (2. von links) und positionieren dieses unter dem GENERATE-Block.

Damit haben wir ein erstes Segment von Blöcken erstellt, das im Bild 3 dargestellt ist. Ein Pfeil zeigt die Bewegungsrichtung der Kunden im Blockdiagramm an und verbindet die Blöcke GENERATE und TERMINATE.

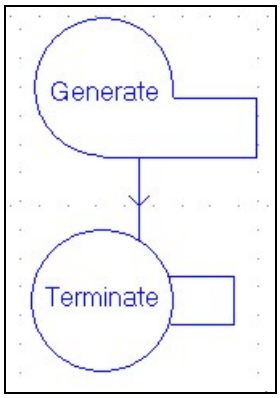


Bild 3: Blockdiagrammdarstellung der Programmlogik PRO01

Mit dem Blockdiagramm wird die Struktur des Programms vorgegeben. Um die speziellen Eingabewerte zu übertragen müssen die Blöcke parametrisiert werden. Dazu wird ein Doppelklick auf das entsprechende Blocksymbol durchgeführt und das entsprechende Dialogfenster öffnet sich. Der GENERATE-Block kann, wie im Bild 4 dargestellt, parametrisiert werden.

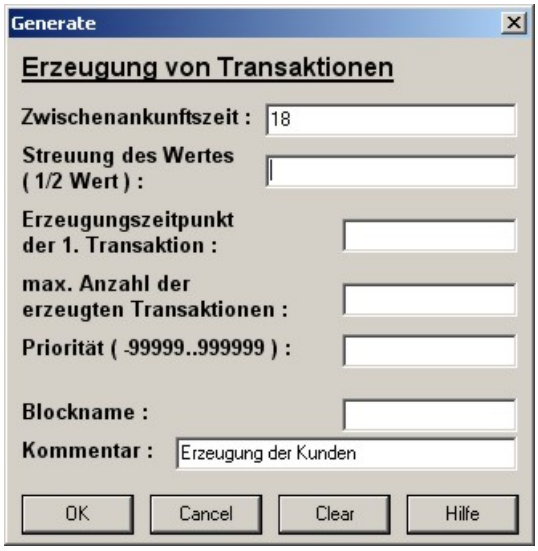


Bild 4: Parameterfenster des GENERATE-Blockes

Insgesamt verfügt der GENERATE-Block über 5 Parameter. Für dieses Programm benötigen wir nur den ersten Parameter, der den Mittelwert der Zwischenankunftszeit festlegt. Die Zwischenankunftszeit (ZAZ) (IAT – Inter Arrival Time) beschreibt den mittleren Abstand zwischen der Ankunft zweier Kunden (Forderungen). In WinGPSS zu Beginn nur der Ankunftszeitpunkt für den ersten Kunden festgelegt. Ist ein Kunde angekommen (erzeugt worden), so wird mit der Zwischenankunftszeit der Erzeugungszeitpunkt für den nächsten Kunden berechnet. Der Erzeugungszeitpunkt für den nächsten Kunden ergibt sich aus der Addition der Zwischenankunftszeit zum Erzeugungszeitpunkt des vorhergehenden Kunden.

Da in unserem Modell alle Kunden genau in einem Abstand von 18 Minuten kommen, wird dieser Wert in das Parameterfenster eingetragen. Durch Klicken des OK-Buttons wird die Parametrisierung des Blockes abgeschlossen. Die eingegebenen Parameter werden in das Blockdiagramm übernommen (Bild 5). Bekommt nur der erste Operand einen Wert, so ist die Ankunftszeit des ersten Kunden genau die Zwischenankunftszeit. Dies ist eine Standardfestlegung. Daraus resultiert, dass der erste Kunde genau zum Zeitpunkt 18 ankommt.

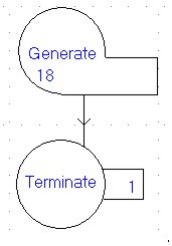


Bild 5: Vollständiges Blockdiagramm für PRO01

Im nächsten Schritt wird der TERMINATE-Block parametrisiert. Wie bei allen Blöcken wird auch hier das Parametrisierungsfenster über einen Doppelklick geöffnet (Bild 6). Der TERMINATE-Block verfügt nur über einen Operanden. Dieser legt den Wert fest, um den der Startzähler reduziert wird, wenn ein Kunde den Block betritt. In unserem Modell ist der Wert 1 eingetragen. Das bedeutet, jeder Kunde reduziert den Startzähler um 1, was der Entnahme einer Marke entspricht. Nach Abschluss der Parametrisierung mit OK wird die 1 in das Blockdiagramm übernommen.

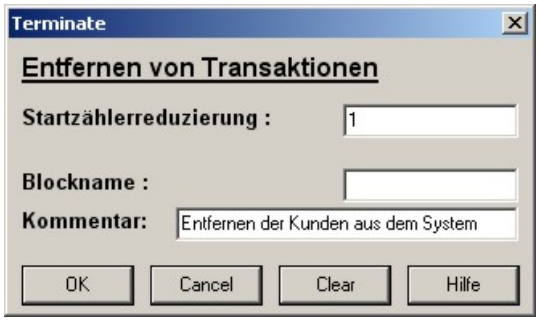


Bild 6: Parametrisierung des TERMINATE-Blockes

Damit ist das Blockdiagramm für unser erstes Modell vollständig parametrisiert. Dies ist auch an der schwarzen Farbe der Blocksymbole zu erkennen. Gleichzeitig wurde das Programm im Textformat generiert. Das wird im Quelltextfenster angezeigt.

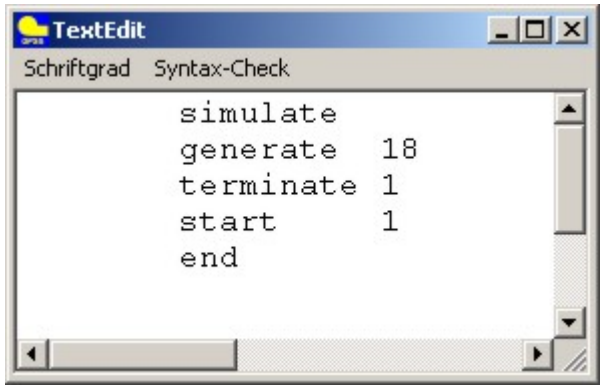


Bild 7: Quelltext von PRO01

Neben den beiden Blockanweisungen sind im Quelltext noch drei Steueranweisungen enthalten. Die Blockanweisungen beziehen sich auf Aktionen, die durch Kunden ausgelöst werden. Deren Ausführung startet, wenn ein Kunde die Blockanweisung betritt. Im Gegensatz dazu beziehen sich die Steueranweisungen im allgemeinen auf Aktionen, die bei Start oder Ende der Simulation ausgeführt werden. In jedem WinGPSS-Programm sind mindestens die 3 Steueranweisungen SIMULATE, START und END vorhanden.

SIMULATE ist die erste Anweisung in einem Programm. Der Operand gibt an, wie oft das Simulationsprogramm abgearbeitet werden soll. Wird kein Operand angegeben, so wird das Programm einmal abgearbeitet. Der Wert kann im Textedit-Fenster oder über das Menü **`Parameter>SIMULATE`** gesetzt werden. Das entsprechende Parameterfenster ist im Bild 8 dargestellt.

Bild 8: Parameterfenster zum Setzen des SIMULATE-Zählers und des Startzählers

Die START-Definitionsanweisung steht immer hinter allen Blockanweisungen und hat als Operand den Wert des Startzählers. In unserem Beispiel entspricht das der Anzahl der Marken, die für den Tag verfügbar sind. Der Operand entscheidet über die Beendigung der Simulation. Der Standardwert für den Startzähler ist 1. Für den ersten Simulationslauf lassen wir den Startzähler unverändert. Damit haben wir für den Startzähler den Wert 1. Für unser Beispiel bedeut das, dass nur eine Marke zur Verfügung steht. Das Programm wird schon durch den ersten Kunden beendet. Es sei nochmals darauf hingewiesen, dass die START-Definitionsanweisung immer hinter dem letzten Block steht.

Die END-Definitionsanweisung besitzt keine Operanden. Sie wird automatisch als letzte Anweisung gesetzt und kennzeichnet das Ende des Programms.

Als nächstes starten wir den Simulationslauf. Dazu gibt es zwei Möglichkeiten. Es kann im Menü **`Simulation>Start`** aufgerufen werden oder der Start-Button gedrückt werden. Der Start-Button ist der linke Button im Bereich der Simulationsablaufsteuerung.

Nach dem Start des Simulationslaufes öffnet sich das Resultatfenster. Dieses zeigt mehrere Karteikarten, die die Ergebnisse enthalten. Für unser Beispiel sind das das Listing, die Blockstatistik und die Zusammenfassung aller Ergebnisse im Lis-File.

```
        simulate  1
        generate  18           !Erzeugung der Kunden
        terminate 1            !Entfernen der Kunden aus dem System
        start     1
        end
```

```
Aufbereitete Quelltextausgabe


 Block                                                              Zeile
 Nr.  *Adr.  Operation A,B,C,D,E,F,G,H        Kommentar             Nr.
             simulate  1                                              1
   1         generate  18                   !Erzeugung der Kunden     2
   2         terminate 1                     !Entfernen der Kunden aus de  3
             start     1                                              4
             end                                                      5
```

```
Zeit         18.00

 Blockstatistik
 Nummer Name   Oper.   Aktuell Gesamt
     1         GENERA             1
     2         TERMIN             1
```

Bild 9: Quelltext, aufbereiteter Quelltext und Blockstatistik für Programm 1

Das Listing enthält ein aufbereitetes Listing des Programms. „Aufbereitetes Listing“ bedeutet, dass das GPSS-System einige Informationen zum Quelltext hinzugefügt hat. Das Programm wird formatiert dargestellt im Gegensatz zur formatfreien Darstellung des Quelltextes. Weiterhin sind die Anweisungen auf der rechten Seite nummeriert. Auf der linken Seite finden wir ebenfalls eine Nummerierung, die sich ausschließlich auf die Blockanweisungen bezieht. In diesem Programmbeispiel gibt es nur zwei Blöcke.

Anschließend klicken wir auf die Karteikarte *Blockstatistik*, um die einzige Statistik zu bekommen, die dieses Programm erzeugt hat. Als erstes erhalten wir eine Information über den Wert der Simulationsuhr, zu dem Zeitpunkt als die Simulation beendet wurde, in diesem Fall 18. GENERATE 18 bedeutet, dass wir annehmen, dass der erste Kunde zum Zeitpunkt 18 ankommt. Da dieser Kunde die einzige Marke entnimmt, endet die Simulation zum Zeitpunkt 18. Wir sehen weiterhin unter **`Gesamt,`** dass genau ein Kunde durch den GENERATE-Block und genau ein Kunde durch den TERMINATE-Block gelaufen ist, genau wie wir es erwartet haben.
Nun verändern wir das Programm so, dass 50 Kunden durch das System laufen. Dazu setzen wir den Startzähler auf den Wert 50. Wir erhalten das Programm, welches in den WinGPSS-Beispielen unter dem Namen PRO01 gespeichert ist. Wir drücken erneut den START-Button und erhalten ein Resultatfenster, wie es im Bild 10 dargestellt ist. Im aufbereiteten Quelltext sehen wir, dass die Start-Anweisung den Wert 50 hat. Das bedeutet, dass 50 Marken entnommen werden müssen, bevor die Simulation beendet wird.

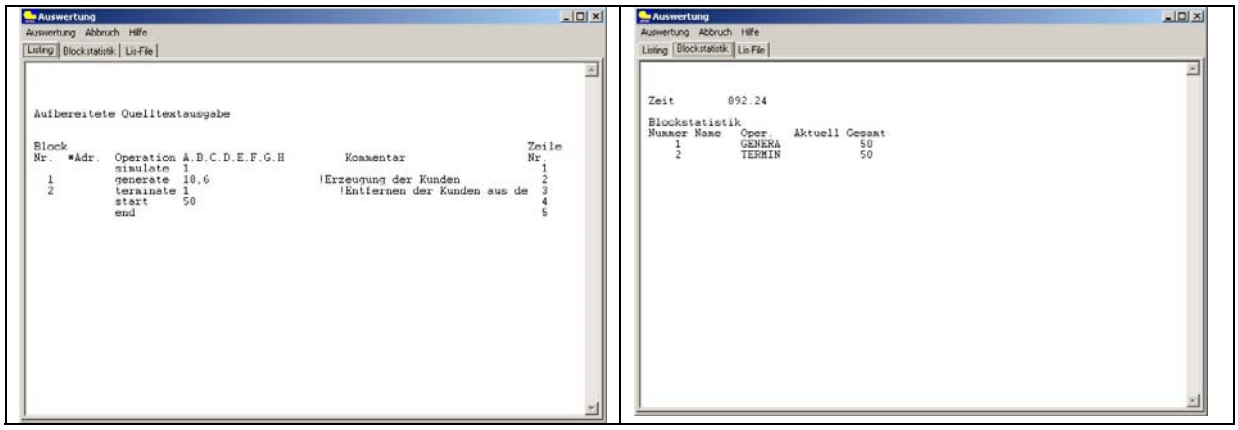


Bild 10: Resultatfenster beim Start mit 50 Marken

Wie angenommen, betritt Kunde 1 genau nach 18 Minuten das Modell. Wann wird der nächste Kunde ankommen? Der GENERATE-Block kann jede gewünschte Anzahl von Kunden erzeugen, alle mit den Zwischenankunftszeiten entsprechend der Verteilung, die durch die Operanden, in diesem Fall den ersten Operanden, beschrieben wird. Daher berechnet sich der Ankunftszeitpunkt für Kunde 2 nach: T(2) = T(1) + ZAZ(2) = 18 + 18 = 36. Kunde 3 wird dementsprechend weitere 18 Zeiteinheiten (Minuten) später ankommen, in diesem Fall 36 + 18 = 54. Entsprechend dieser Vorgehensweise wird der GENERATE-Block weitere Kunden erzeugen und in das System bringen. Als Kunde 1 zum Zeitpunkt 18 aktiv wurde, berechnete der GENERATE-Block den Ankunftszeitpunkt für den Kunden 2 für den Zeitpunkt 36. Wenn Kunde 2 das System zum Zeitpunkt 36 betritt, wird vom GENERATE-Block der Ankunftszeitpunkt für den Kunden 3 berechnet.

Auf diese Weise würde die Simulation unendlich laufen, wenn es nicht den TERMINATE-Block geben würde. Jeder Kunde entnimmt eine Marke. Nach dem ersten Kunden sind noch 49 Marken, nach dem 2. Kunden noch 48 Marken vorhanden und nach 49 Kunden ist genau noch eine Marke vorhanden. Der 50. Kunde, der das Modell zum Zeitpunkt 50*18=900 betreten hat, entnimmt die letzte Marke. Die Simulation wird beendet, da keine Marken mehr vorhanden sind. Dies ist in der Blockstatistik ersichtlich, da die Simulationsuhr zum Zeitpunkt des Simulationsendes den Wert 900 anzeigt. Unter der Überschrift **`Gesamt`** können wir sehen, dass 50 Kunden sowohl den GENERATE als auch den TERMINATE-Block durchlaufen haben.

Zusammenfassung Kapitel 1

- Der GENERATE-Block ist der erste Block, der in jedem GPSS-Programm verwendet wird.
- Er wird verwendet, um Kunden, oder vergleichbare temporäre Elemente (Forderungen) zu erzeugen.
- Der erste Operand ist die **Z**wischen**A**nkunfts**Z**eit (ZAZ) (InterArrivalTime IAT), die Zeit zwischen der Erzeugung von zwei Forderungen. Wird nur die ZAZ angegeben, so wird der erste Kunde zum Zeitpunkt 0 + die erste berechnete ZAZ erzeugt.
- Der TERMINATE-Block entfernt Kunden aus dem Simulationssystem. Sein einziger Operand ist die Anzahl um die der Startzähler von jedem Kunden reduziert wird, der den Block betritt.
- Die Start-Steueranweisung hat als Operanden den Wert des Startzählers zum Beginn der Simulation.
- Die Simulation wird beendet, wenn der Startzähler den Wert 0 hat.

Fragen Kapitel 1

- Welche Änderung müssen Sie im Programm PRO01 vornehmen, wenn die Kunden alle 24.6 Minuten ankommen?
- Welche Änderung müssen Sie im Programm PRO01 vornehmen, wenn die Kunden alle 10 Stunden ankommen?
- Wie verändern sich die Ausgabedaten im Programm PRO01, wenn 100 Kunden ankommen?
- Was passiert, wenn der A-Operand des TERMINATE-Blockes von 1 auf 2 verändert wird?

Übung 1:

Kunden erreichen das Drehkreuz genau in einem Abstand von 22 Minuten. Der erste Kunde kommt genau 22 Minuten nach Öffnung des Drehkreuzes. Alle Kunden bewegen sich direkt (zeitlos) durch das Drehkreuz. Das Drehkreuz wird nach dem Passieren von 150 Kunden geschlossen. Erstellen Sie ein GPSS-Programm, welches dieses System nachbildet.

Kapitel 2

Im Programm PRO01 war nichts vom Zufall abhängig. Die Zeit zwischen der Ankunft eines Kunden und seines Nachfolgers betrug genau 18 Minuten. Da in diesem Modell nichts unbestimmt war, bekomme ich immer die gleichen Resultate, unabhängig davon, wie oft ich das Programm starte. Dies kann man erkennen, wenn man das Programm PRO01 zum Beispiel dreimal ablaufen lässt.

Wenn Sie mit diesem Kapitel unmittelbar im Anschluss an Kapitel 1 fortsetzen, ohne Übung 1 zu bearbeiten, so haben Sie noch das Programm PRO01 im WinGPSS geladen. Anderenfalls müssen Sie Programm PRO01 erneut laden. Dazu gehen zum Menü **`Datei>öffnen.`** Anschließend wählen Sie über den Windows-Dialog die Datei im entsprechenden Verzeichnis aus. Anschließend erscheinen das Block-Diagramm und der Quelltext von PRO01.

Wenn Sie im nächsten Schritt den Simulate-Parameter auf 3 setzen (**`Parameter>Simulate`**) und anschließend die Simulation starten, so wird das Programm dreimal abgearbeitet. In der ersten Zeile des Programmlistings steht jetzt simulate 3. Das bedeutet, dass das Programm dreimal abgearbeitet wird. In der Blockstatistik werden jetzt die Resultate aller 3 Simulationsläufe angezeigt. In diesem Fall sind die Resultate von Lauf 2 und Lauf 3 identisch mit denen des ersten Laufes: 50 Kunden haben das System betreten und zum Zeitpunkt 900 wieder verlassen. Das ist nicht überraschend, da alle 3 Simulationsläufe mit den gleichen Zwischenankunftszeiten für die Kunden abgearbeitet werden. Haben wir ein deterministisches Modell, **ohne zufälligen Größen**, so ist es offensichtlich klar, das wir dieses Modell nur **einmal abarbeiten** müssen.

Wir werden im Folgenden das Modell PRO01 so modifizieren, das **zufällige Größen** enthalten sind. Wir werden die Zwischenankunftszeiten der Kunden zufällig so verändern, dass diese nicht mehr alle 18 Minuten ankommen, sondern stattdessen zwischen 12 und 14 Minuten, so wie im Bild 11 dargestellt.

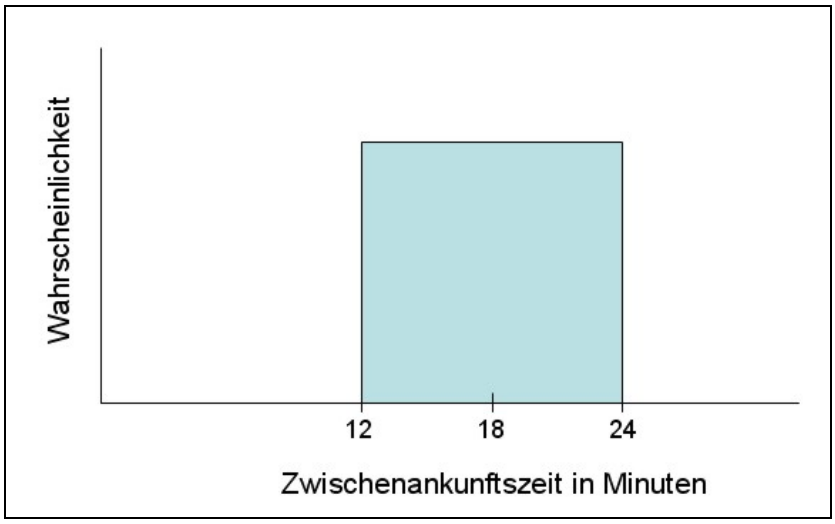


Bild 11: Gleichverteilte Zwischenankunftszeiten

Der Mittelwert der Zwischenankunftszeit liegt immer noch bei 18 Minuten, aber die Kunden treffen in einem Abstand von 18 ± 6 Minuten ein. Die „halbe Abweichung (Streuung)“ ist 6 Minuten, während die gesamte Streuung 24-12 = 12 Minuten ist. Alle Zeiten sind daher **gleichverteilt**, das bedeutet, dass alle Zeiten zwischen den beiden Extremwerten von 12 und 24 gleich häufig auftreten können. Das ist die einfachste Annahme bezüglich der Zufälligkeit und anwendbar für unsere ersten Programme. An dieser Stelle soll jedoch schon darauf hingewiesen werden, dass viele andere Verteilungen in GPSS verfügbar sind, die in den späteren Kapiteln eingeführt werden.

Um die Ankunft der Kunden entsprechend dieser Gleichverteilung zu verändern ist es erforderlich, die Parameter des GENERATE-Blockes zu verändern. Dazu führen wir einen Doppelklick auf den GENERATE-Block im Blockdiagramm aus. Es öffnet sich ein Parameterfenster. Wir tragen in das Feld „Streuung des Wertes (1/2 Wert):“ die 6 ein so wie im Bild 12 dargestellt.

Die Eintragung des zweiten Operanden bewirkt Folgendes: Es ist nicht möglich, dass Kunde 2 vor dem Kunden 1 das System betritt, d.h. die Zwischenankunftszeit ZAZ(2) zwischen der Ankunft des zweiten und des ersten Kunden kann nicht negativ sein. Da der kleinste Wert der Zwischenankunftszeit nicht negativ werden darf, darf der **zweite Operand**, der als **B-Operand** bezeichnet wird, nicht größer sein als der **erste Operand**, der als **A-Operand** bezeichnet wird. (Aus der Tatsache, dass die untere Grenze A-B ≥ 0 sein muss, folgt das A ≥ 0 sein muss.) Wird für den B-Operanden kein Wert angegeben, so wie in PRO01, so wird ein impliziter „Standardwert“ 0 für den B-Operanden angenommen, der eine konstante Zwischenankunftszeit erzeugt.

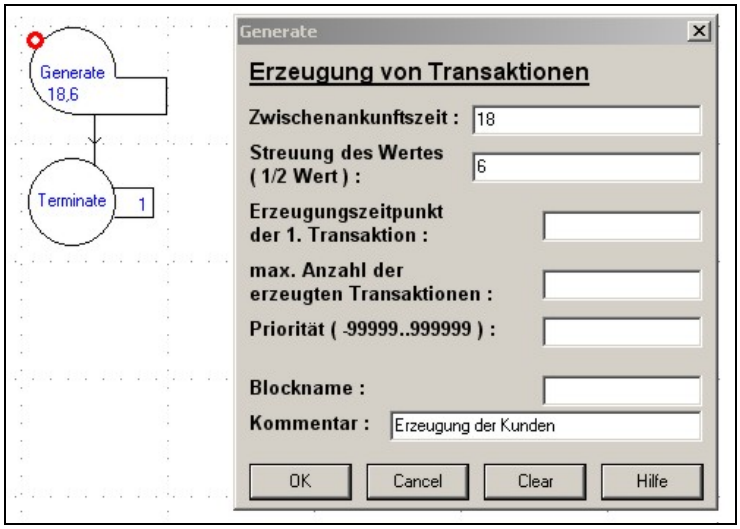
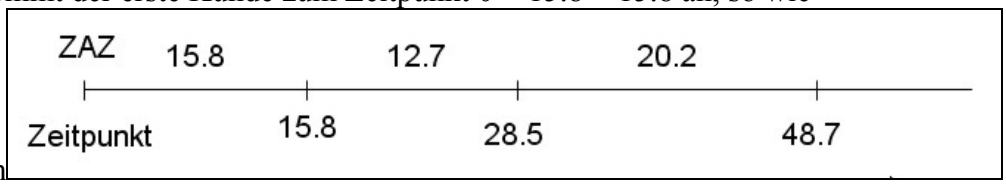


Bild 12: Einführung zufälliger Zwischenankunftszeiten

Nun schließen wir das Parameterfenster mit dem OK-Button und sehen, dass 18,6 als Text im GENERATE-Block erscheint. Die beiden Operanden sind durch ein **Komma** getrennt, so wie im Bild 12 dargestellt. Das Komma ist das **Trennzeichen** für die Operanden. Die Kunden werden jetzt mit Zwischenankunftszeiten zwischen 12 und 24 Minuten erzeugt, wobei alle Werte die gleiche Wahrscheinlichkeit haben.

Im Folgenden betrachten wir, wie dieser GENERATE-Block das zufällige Eintreffen der Kunden realisiert. Für den Ankunftszeitpunkt des ersten Kunden bestimmen wir die erste Zwischenankunftszeit ZAZ(1). Da nur der A- und B-Operand des GENERATE-Blockes festgelegt sind, haben wir den Standardfall, dass der erste Kunde zum Zeitpunkt T(1) = 0 + ZAZ(1) = ZAZ(1) ankommt. Wenn wir annehmen das ZAZ(1) = 15.8 ist, so kommt der erste Kunde zum Zeitpunkt 0 + 15.8 = 15.8 an, so wie



im Bild 13 dargestellt. (Zur Vereinfachung wird nur 1 Nachkommastelle dargestellt. Für die Bestimmung des Ankunftszeitpunktes des 2. Kunden muss zuerst die 2. Zwischenankunftszeit, ZAZ(2), festgelegt werden. Im nächsten Schritt berechnen wir den Ankunftszeitpunkt des zweiten Kunden T(2) als Summe von T(1) und ZAZ(2). Das bedeutet, dass die Zwischenan-

kunftszeit 2 zum Ankunftszeitpunkt des ersten Kunden addiert wird. Wenn ZAZ(2) = 12.7 ist, so erhalten wir in unserem Beispiel in

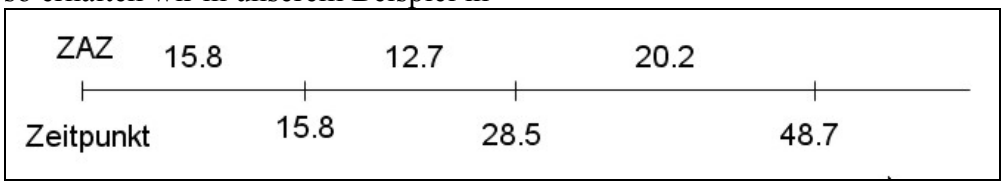


Bild 13 einen Wert für T(2) = 15.8 + 12.7 = 25.5. Analog wird die Zwischenankunftszeit des 3. Kunden berechnet.
T(3) = T(2) + ZAZ(3), d.h. 28.5 + 20.2 = 48.7.

Bild 13: Beziehungen zwischen Zwischenankunftszeit und Ankunftszeitpunkt

Da wir nun zufällige Zwischenankunftszeiten haben, ist es nicht mehr ausreichend, das neue Programm, welches wir als **PRO02** bezeichnen wollen, nur einmal abzuarbeiten. Jeder Simulationslauf verwendet eine andere Folge von Zufallszahlen. Da SIMULATE noch den Wert 3 hat klicken wir auf den Start-Button und lassen dieses Programm dreimal abarbeiten. Im Resultatfenster sehen wir im Programm-Listing, dass wir SIMULATE 3 und GENERATE 18,6 haben. In der Blockstatistik im Bild 14 sehen wir die Resultate von allen drei Läufen. Der erste war zum Zeitpunkt 892.24, der zweite zum Zeitpunkt 859.92 und der dritte Zeitpunkt 900.63 beendet.

```
Zeit        892.24

 Blockstatistik
 Nummer Name   Oper.   Aktuell Gesamt
     1         GENERA            50
     2         TERMIN            50
------------------------------------------------------------
Lauf Nr.:  2

 Zeit        859.92

 Blockstatistik
 Nummer Name   Oper.   Aktuell Gesamt
     1         GENERA            50
     2         TERMIN            50

------------------------------------------------------------
Lauf Nr.:  3
Zeit        900.63

 Blockstatistik
 Nummer Name   Oper.   Aktuell Gesamt
     1         GENERA            50
     2         TERMIN            50
```

Bild 14: Blockstatistik für drei Simulationsläufe PRO02

Jeder der dieses Programm startet erhält dieselben Resultate. Das liegt daran, dass GPSS wie alle anderen Simulationssysteme, keine echten Zufallszahlen benutzt, sondern Pseudozufallszahlen. Diese „künstlichen Zufallszahlen“ sehen zwar aus wie Zufallszahlen, werden jedoch durch ein Programm berechnet. Lassen Sie uns den Unterschied etwas näher betrachten.

Im täglichen Leben kennen wir die Erzeugung von Zufallszahlen. Das Werfen einer Münze ist ein Beispiel für einen Prozess der echte Zufallszahlen erzeugt. Wenn wir vollkommen unvoreingenommen würfeln, so erzeugen wir die Zufallszahlen 1,2,3,4,5 oder 6, wobei jede dieser 6 Zahlen gleich wahrscheinlich ist.

Bei der Ziehung mancher Lotterie-Gewinnzahlen wird die Gewinn-Nummer Ziffer für Ziffer ermittelt, indem für jede Ziffer einer von zehn Bällen gezogen wird. Für diese echten Zufallszahlen kann niemand, auch nicht mit der Unterstützung leistungsfähigster Computer bestimmen, welche Ziffer als nächste gezogen wird. Weiterhin ist jede Ziffer die erzeugt wird eine unabhängige Ziehung. Das bedeutet, dass die gezogene Ziffer vollständig unabhängig von den vorher gezogenen Ziffern ist. Die Wahrscheinlichkeit, eine 6 zu würfeln beträgt immer 1/6, unabhängig davon ob zuvor eine 6 gewürfelt wurde oder nicht.

Bei der Simulation stochastischer Ereignisse werden keine echten Zufallszahlen benutzt, sondern eine Folge von *Pseudozufallszahlen*. Diese Pseudozufallszahlen haben im Allgemeinen die gleichen Eigenschaften wie echte Zufallszahlen. Ein Anwender sollte Pseudozufallszahlen nicht von echten Zufallszahlen unterscheiden können. Falls jemand, der nicht mit der Erzeugung von Pseudozufallszahlen vertraut ist, eine Folge solcher Zufallszahlen bekommt, so sollte er nicht unterscheiden können, ob diese Zufallszahlen echt sind, z.B. mit einem 10-flächigen Würfel gewürfelt, oder ob sie pseudozufällig, durch einen speziellen Computeralgorithmus erzeugt wurden. Dies sollte sogar zutreffen, wenn unser Anwender ein Statistiker ist und verschiedene statistische Tests anwendet.

Jedoch unterscheiden sich Pseudozufallszahlen von echten Zufallszahlen in verschiedener Hinsicht. Alle diese Unterschiede beruhen auf der Tatsache, dass die Folge von Pseudozufallszahlen durch einen Algorithmus berechnet wird, z.B. ein Computerprogramm. Dieses startet immer mit einer Zahl, *r(0)*, die auch als **Startwert** bezeichnet wird.

Die nächste Zahl in der Zufallszahlenfolge, *r(1)*, wird durch eine Berechnung erzeugt, die häufig auf einer komplizierten Transformation von *r(0)* beruht. Die nächste Nummer in der Zufallszahlenfolge ist *r(2)*, die nach der gleichen Transformation aus *r(1)* berechnet wird. Für *r(3)* und die weiteren Zahlen der Zufallszahlenfolge gilt das gleiche Vorgehen. In einem kurzen Anhang an das Kapitel 2 wird für technisch Interessierte detailliert erklärt, wie die Zufallszahlenerzeugung in GPSS realisiert wird. Für den allgemeinen Leser ist es ausreichend, folgende 3 Dinge zu kennen:

1. Die Zufallszahlen in der Folge liegen zwischen 0 und 1 und sind gleichverteilt. Etwas genauer, die 0 ist der kleinste mögliche Wert und 0.99999999999… ist der größte mögliche Wert. (Die Begründung, weshalb keine Zufallszahl genau den Wert annehmen kann, wird im Anhang gegeben.) Zufallszahlen dieses Typs werden genutzt um eine Gleichverteilung zu erzeugen, wie sie z.B. im Programm PRO02 benötigt wird. Ist in diesem Beispiel der Wert der Zufallszahl in der Nähe von 0, so ist der berechnete Wert nahe am unteren Limit von 12; ist der Wert der Zufallszahl nahe 1, so ist der berechnete Wert nahe dem oberen Limit von 24. Hat die Zufallszahl den Wert 0.5, so ist der berechnete Wert genau der Mittelwert von 18.
2. Die Zufallszahlenfolge kann leicht wiederholt werden. Immer wenn wir den gleichen Startwert *r(0)* und den gleichen Transformationsalgorithmus verwenden, wird auch die gleiche Zufallszahlenfolge *r(1), r(2), r(3)* usw. erzeugt. Solch eine Wiederholung einer Zufallszahlenfolge ist offensichtlich mit echten Zufallszahlen nicht möglich. In den folgenden Kapiteln wird sich noch zeigen, dass die Verwendung gleicher Zufallszahlenfolgen für die Durchführung von Simulationsexperimenten große Vorteile hat.
3. Wie schon erwähnt initialisiert der Startwert die Zufallszahlenfolge. Für jeden Simulationslauf wird ein spezieller Startwert genutzt, der dementsprechend eine spezifische Folge

von Zufallszahlen erzeugt. Daher ist die Startzahl des 1. Simulationslaufes verschieden von der Startzahl des 2. Simulationslaufes und so sind natürlich auch die beiden Zufallszahlenfolgen verschieden. Daraus folgt, wenn wir mehr als einen Simulationslauf durchführen, so können wir für jeden Lauf **unterschiedliche** Ergebnisse erhalten. Jedes dieser Ergebnisse repräsentiert ein Beispiel aus einer großen Anzahl von möglichen Lösungen. Wollen wir aus den Simulationsergebnissen die richtigen Schlussfolgerungen ableiten, so darf man **niemals** nur einen Simulationslauf durchführen, denn man kann nicht vorhersagen, welche Auswirkungen die Unterschiede in den Zufallszahlenfolgen haben.

Zusammenfassung Kapitel 2

- Der A-Operand des GENERATE-Blockes enthält den Mittelwert der Zwischenankunftszeit, der B-Operand enthält die halbe Abweichung der Gleichverteilung.
- GENERATE A,B erzeugt Zufallszahlen mit der Zwischenankunftszeit A ± B Zeiteinheiten, dass heißt, das alle Zeiten im Intervall von A – B bis A + B gleich wahrscheinlich sind.
- Wenn kein Wert für den A- oder B-Operanden im GENERATE-Dialogfenster angegeben wird, so wird der Standardwert 0 für diese Operanden angenommen.
- Ein Programm, welches stochastische Größen beinhaltet, sollte mehrfach abgearbeitet werden, wenn man gesicherte Ergebnisse erhalten will. Die Anzahl der Simulationsläufe wird mit dem SIMULATE-Parameter bestimmt.
- Für jeden Simulationslauf wird eine andere Zufallszahlenfolge benutzt. Daraus folgt, dass Programme mit zufälligen Variablen in der Regel für jeden Simulationslauf unterschiedliche Ergebnisse erzeugen.

Fragen Kapitel 2

1. Wie muss der GENERATE-Block im Beispiel PRO02 verändert werden, wenn die Kunden im Mittel alle 18 Minuten eintreffen? Der kleinste Abstand zwischen zwei Kunden ist 12.5 Minuten, der größte Abstand 23.5 Minuten. Alle Werte sind gleichverteilt.
2. Wie muss das Programm PRO02 verändert werden, wenn sichergestellt werden soll, dass die Kunden alle 18.5 Minuten eintreffen, wobei die Zeiten gleichverteilt von 12 bis 25 Minuten schwanken?
3. Wie muss das Programm PRO02 verändert werden, wenn sichergestellt werden soll, dass die Kunden alle 10 Minuten eintreffen, wobei die Zeiten gleichverteilt von 5 bis 15 Minuten schwanken?

Übung 2:

An einem Messestand kommen Kunden in einem mittleren Abstand von 10 Minuten an. Manchmal liegen nur 5 Minuten zwischen der Ankunft zweier Kunden, manchmal liegt eine viertel Stunde zwischen der Ankunft zweier Kunden. Alle Zeiten zwischen diesen beiden Extremwerten können gleich häufig auftreten. Jeder Kunde nimmt einen Prospekt. Am Morgen eines jeden Tages sind 80 Prospekte vorhanden. Zu welchem Zeitpunkt sind alle Prospekte vergriffen? Lassen Sie das Programm zweimal laufen.

Bekommen Sie als Ergebnis ca. 795 und 756 (Minuten)? Verändern Sie das Programm nun dahingehend, das jeder Kunde 2 Prospekte entnimmt. Wann sind unter diesen Bedingungen alle Broschüren vergriffen und wie viele Kunden waren an dem Stand?

Erhalten Sie ca. 396 und 374 (Minuten) und 40 Kunden als Ergebnis?

Lassen Sie uns abschließend untersuchen, wie sich dieses Ergebnis verändert, wenn wir am Morgen des Tages nur 79 Prospekte haben. Begründen Sie das Ergebnis. (Wie viele Prospekte bekam der letzte Kunde?)

Anhang:

Für den technisch interessierten Anwender erläutern wir im folgenden Abschnitt detaillierter die Zufallszahlenerzeugung. WinGPSS verwendet die multiplikative Kongruenzmethode nach Lehmer. Das bedeutet, dass wir eine Folge von ganzen Zahlen in einer Serie von mehreren Schritten berechnen das *i(j+1) = ai(j) modulo (m)* ist, wobei m der Maximalwert z.B. 2147483648 und a der Faktor z.B. 69069 ist. Wir können so jeden Schritt, ohne die Modulorechnung wie folgt erklären: Als erstes multiplizieren wir *i(j)* mit dem Faktor a und erhalten das Produkt p. Dieses Produkt wird anschließend durch m dividiert und wir erhalten den Quotienten q. Trennen wir den ganzzahligen Anteil von diesem Quotienten ab, so erhalten wir den Nachkommaanteil q – TRUNC(q). Multiplizieren wir diesen Nachkommaanteil mit m so erhalten wir die nächste Zufallszahl unserer Folge *i(j+1)*. Da *i(0)* durch den Startwert festgelegt ist, können wir nach diesem Algorithmus ganzzahlige Zufallszahlen von 0 bis zur größten Zahl m – 1 erzeugen. Es ist klar, dass der Wert von m nicht erreicht werden kann, da dazu der Nachkommaanteil gleich 1 sein müsste. Der Nachkommaanteil ist aber immer < 1 definiert. Durch eine abschließende Division jeder Zufallszahl *i(j)* durch den Maximalwert m so erhalten wir eine Zufallszahl r(j) mit einem Wert von 0 bis 0.999999999…

Kapitel 3

In den beiden bisher betrachteten Programmen bestimmte die Anzahl der Kunden, die das Modell durchlaufen haben, wann die Simulation beendet wurde. Im nächsten Programm soll stattdessen die **Simulation nach einer festgelegten Zeit beendet werden,** genau nach 8 Stunden. Wobei die Kunden weiterhin in einem Abstand von 18 ± 6 Minuten ankommen. Wir verändern das Programm PRO02 entsprechend diesen Bedingungen. Falls das Programm nicht im WinGPSS geladen ist, so öffnen wir es dazu jetzt.

Die erste Veränderung die wir vornehmen besteht darin, dass die Kunden nicht mehr über das Ende der Simulation entscheiden. Dies wird dadurch realisiert, dass sie keine Marken mehr entnehmen. Im Programm erreichen wir das, indem das Dialogfenster des TERMINATE-Blockes geöffnet und der Wert 1 im Feld Startzählerreduzierung gelöscht wird. Das leere Feld wird mit dem Standardwert 0 belegt.

Nach Klicken des OK-Buttons verändert sich die Bildschirmdarstellung so, wie im

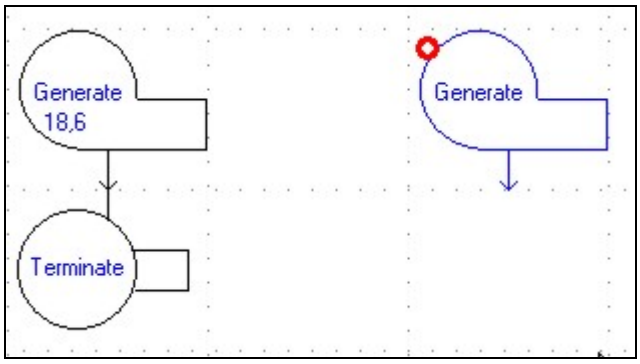
Bild 15 dargestellt. Diese Anpassung bewirkt, dass der TERMINATE-Block zwar die Kunden aus dem System entfernt, den Startzähler aber nicht herunterzählt.

Bild 15: Kundensegment ohne Veränderung des Startzählers

Wir müssen jetzt ein zweites Segment einführen, dass den Simulationslauf dieses Programms nach 8 Stunden beendet. (Als Segment wird eine Folge von Blöcken bezeichnet, die mit einem GENERATE-Block beginnt und üblicherweise mit einem TERMINATE-Block beendet wird.) Wenn wir kein solches Stopp-Segment haben würde die Simulation endlos laufen, da die Kunden den Startzähler nicht herunterzählen und jeder erzeugte Kunde den Erzeugungszeitpunkt seines Nachfolgers festlegt. Das Stopp-Segment können wir uns als Hausmeister vorstellen, der nach 8 Stunden ankommt und das Drehkreuz abschließt.

Zur Erzeugung des Hausmeisters wird ein neues Segment erstellt. Dazu wird mit dem Mauszeiger das Symbol des GENERATE-Blockes im Blockmenü (Symbol am linken Rand) akti-

viert und anschließend durch einen Klick in die erste Zeile der übernächsten Spalte positio-



niert, so wie im Bild 16 dargestellt.

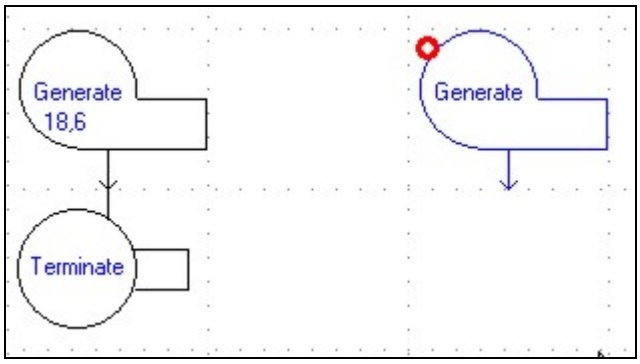
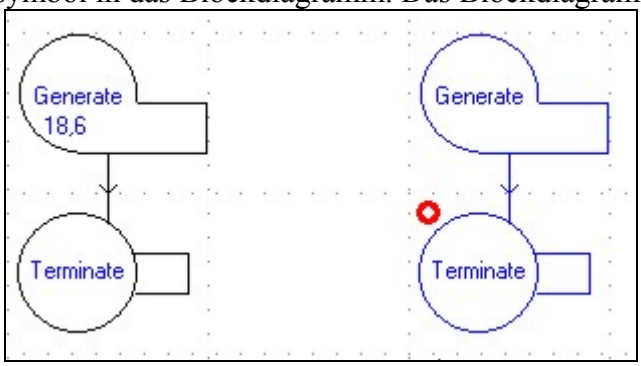


Bild 16: Einfügen eines neuen GENERATE-Blockes

Der Hausmeister muss das System auch wieder verlassen. Daher positionieren wir auf die gleiche Weise das TERMINATE-Blocksymbol in das Blockdiagramm. Das Blockdiagramm der Kunden- und Stopp-Segments ist im Bild 17 dargestellt.



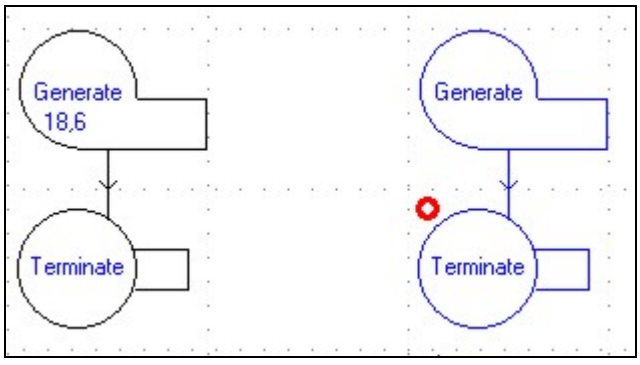


Bild 17: Kunden- und Stopp-Segment

Da es verwirrend ist, im Stopp-Segment von Kunden oder Forderungen zu sprechen werden diese im GPSS allgemein als **Transaktionen** bezeichnet. Transaktionen sind die **temporären**

Modellelemente, die durch einen GENERATE-Block erzeugt werden. In unserem Beispielprogramm haben wir zwei Arten von Transaktionen, Kunden und einen Hausmeister.

Als nächstes parametrisieren wir die Blöcke im Stopp-Segment. Da wir im Kundensegment Minuten als Zeiteinheit verwenden, müssen wir auch im Stopp-Segment Minuten als Zeiteinheit verwenden. In GPSS ist nicht festgelegt, welche Zeitdauer einer Zeiteinheit entspricht. Einer Zeiteinheit kann jede beliebige Zeitdauer zugeordnet werden, jedoch muss beachtet werden, dass diese **Zuordnung für das gesamte Modell gilt**. 8 Stunden müssen demzufolge als 480 Minuten angegeben werden. Durch Doppelklick auf den neuen GENERATE-Block öffnet sich das Parameterfenster und wir tragen den Wert 480 für die Zwischenankunftszeit ein. Da in diesem Fall der Hausmeister exakt nach 480 Minuten kommt, braucht keine Abweichung eingetragen werden und der B-Operand dieses GENERATE-Blockes bleibt leer. Vom System wird als Wert für die Abweichung der Standardwert 0 verwendet. Wenn kein anderer Parameter angegeben wurde, so erreicht der Hausmeister entsprechend der eingegebenen Zwischenankunftszeit das System exakt zum Zeitpunkt 480 nach Simulationsbeginn.

Da der Hausmeister das System nach 480 Minuten schließen soll, folgt daraus, dass der zu diesem Segment gehörende TERMINATE-Block den Startzähler herunterzählen muss (bzw. Marken entnehmen). Steht noch der Wert 50 im Startzähler, so muss der Hausmeister alle 50 Marken entnehmen, d.h. den Startzähler auf den Wert 0 herunterzählen. Wir öffnen das Parameterfenster des TERMINATE-Blockes, tragen 50 in das Feld „Startzählerreduzierung“ ein und schließen das Parameterfenster mit OK. Wir starten das Programm, PRO03, und erhalten eine Blockstatistik, wie im Bild 18 angegeben.

```
Zeit        480.00

 Blockstatistik
 Nummer Name   Oper.   Aktuell Gesamt
     1         GENERA            27
     2         TERMIN            27
     3         GENERA             1
     4         TERMIN             1
---------------------------------------
Lauf Nr.:  2
Zeit        480.00

 Blockstatistik
 Nummer Name   Oper.   Aktuell Gesamt
     1         GENERA            27
     2         TERMIN            27
     3         GENERA             1
     4         TERMIN             1
-------------------------------------
Lauf Nr.:  3
Zeit        480.00

 Blockstatistik
 Nummer Name   Oper.   Aktuell Gesamt
     1         GENERA            26
     2         TERMIN            26
     3         GENERA             1
     4         TERMIN             1
```

Bild 18: Blockstatistik von PRO03

Wir sehen, dass die Simulation in allen drei Simulationsläufen genau zum Zeitpunkt 480 beendet wurde. In den ersten beiden Simulationsläufen sind 27 Kunden angekommen, jedoch im dritten Simulationslauf nur 26 Kunden.

Im Folgenden soll darauf hingewiesen werden, dass wir dieses Modell vereinfachen können, indem wir den Startzähler nur den Wert 1 geben. Wenn der Hausmeister jetzt den Startzähler um 1 reduziert, wird die Simulation beendet. Dazu öffnen wir als erstes das Parameterfenster des TERMINATE-Blockes und setzen die Startzählerreduzierung auf 1. Nach Schließen des Parameterfensters haben wir das Blockdiagramm des Programms PRO04, wie im Bild 19 dargestellt.

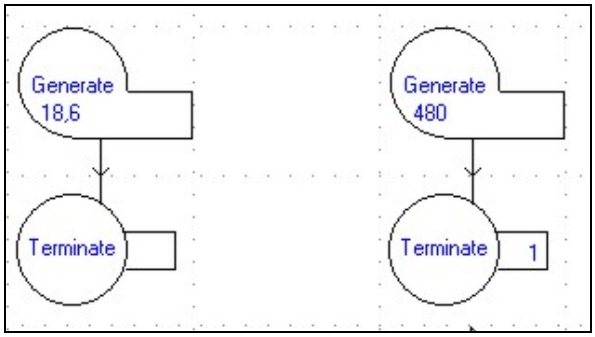


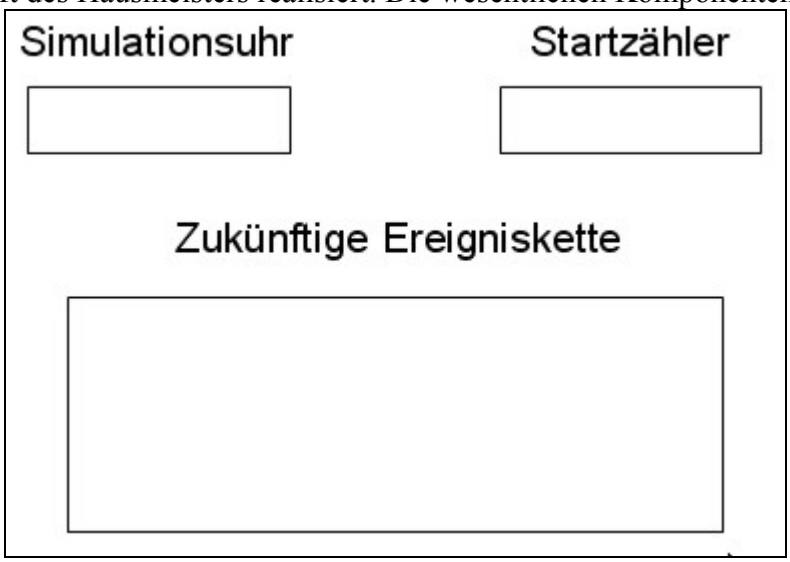
Bild 19: Blockdiagramm von PRO04

Als nächstes setzen wird als Parameter für den Startzähler den Wert 1 und starten das Programm erneut. Wir erhalten exakt die gleichen Resultate wie für das Programm PRO03 im
Zeit 480.00

```
 Blockstatistik
 Nummer Name   Oper.   Aktuell Gesamt
     1         GENERA            27
     2         TERMIN            27
     3         GENERA             1
     4         TERMIN             1
---------------------------------------
Lauf Nr.:  2
Zeit        480.00

 Blockstatistik
 Nummer Name   Oper.   Aktuell Gesamt
     1         GENERA            27
     2         TERMIN            27
     3         GENERA             1
     4         TERMIN             1
-----------------------------------
Lauf Nr.:  3
Zeit        480.00

 Blockstatistik
 Nummer Name   Oper.   Aktuell Gesamt
     1         GENERA            26
     2         TERMIN            26
     3         GENERA             1
     4         TERMIN             1
```

Bild 18. Die Simulation stoppt, unabhängig davon, ob der Startzähler mit dem Anfangswert 50 um 50 reduziert oder der Startzähler mit dem Anfangswert um 1 reduziert wurde.

Abschließend betrachten wir etwas genauer, wie das GPSS-System intern die Ankunft der Kunden und die Ankunft des Hausmeisters realisiert. Die wesentlichen Komponenten des GPSS-Systems sind im Bild 20 dargestellt.



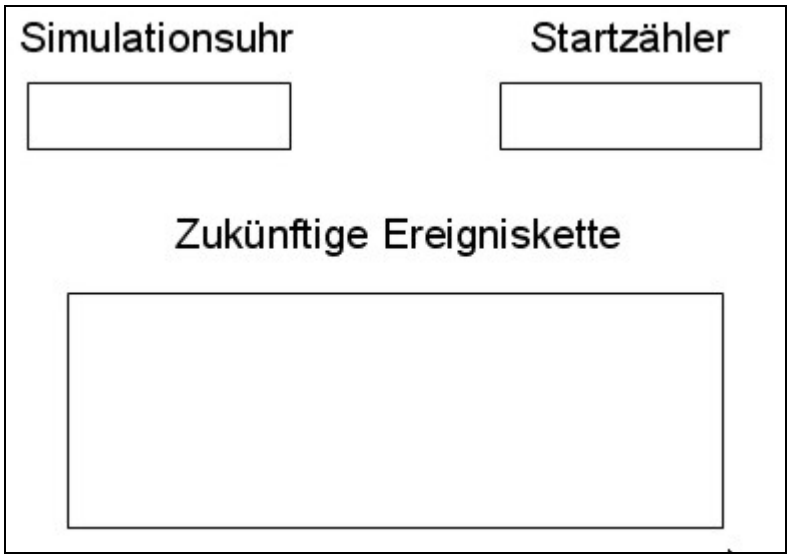


Bild 20: Ein erster Blick in das interne GPSS-System

Die Ausführung und die Ereignisverwaltung kann wie folgt beschrieben werden:

1. Das GPSS-Systsem liest die SIMULATE und die START-Anweisung und setzt den Startzähler auf den Wert, der als A-Operand der START-Definitionsanweisung angegeben ist, (z. B. die Anzahl der Marken, die beim Start vorhanden sind).
2. Der erste Ankunftszeitpunkt für die erste Transaktion eines jeden GENERATE-Blockes wird berechnet und in eine zukünftige Ereigniskette (ZEK) eingetragen, so wie in Phase 1 von Bild 21 dargestellt.
3. Die zukünftige Ereignisliste ist nach den nächsten Ereignissen sortiert, in diesem Fall die Ankunft von Kunde 1 zum Zeitpunkt 15.8.
4. Die Simulationsuhr wird auf diesen Zeitpunkt gesetzt, hier 15.8.
5. Wenn das Ereignis ein GENERATE-Block-Ereignis ist, so wie in diesem Fall:
   - Eine neue Transaktion wird in das Modell eingeschleust.
   - Das zugehörige Ereignis wird aus der ZEK entnommen.
   - Der nächste Ankunftszeitpunkt wird berechnet und in die Liste eingefügt. In diesem Beispiel wird für den zweiten Kunden die Ankunft für den Zeitpunkt 28.5 berechnet, so wie als Ereignis 2 im Bild 21 dargestellt.
6. Die neu eingeschleuste Transaktion wird so weit wie möglich durch die Blöcke bewegt.
7. Wenn die neu eingeschleuste Transaktion angehalten wurde, geht das GPSS-System zurück zur ZEK und bestimmt das nächste Ereignis. Auf diese Weise setzt das GPSS-System die Abarbeitung bei Ereignis 3 fort. In diesem Fall ist das das Entfernen eines weiteren Kunden aus der ZEK und eine entsprechende Ankunftszeitberechnung für den nachfolgenden Kunden.
8. Wenn eine Transaktion durch einen TERMINATE-Block läuft und dabei den Wert des Startzählers reduziert, so dass der Startzähler einen Wert ≤ 0 bekommt, wird die Simulation gestoppt. Im Programm PRO04 erfolgt dies zum Zeitpunkt 480. Wir sehen im Bild 21 das zum Zeitpunkt 472.9 die Ankunft für den 28. Kunden zum Zeitpunkt 496.1 berechnet wird. Zum Zeitpunkt 472.9 ist die Ankunft des Hausmeisters zum Zeitpunkt 480 das nächste Ereignis. Somit kommt der Hausmeister an und beendet den Simulationslauf.

```
Ereignis Ereigniszeitpunkt
  1        0.0          Kunde 1     Ankunftszeit        15.8
                        Hausmeister Ankunftszeitpunkt  480
  2       15.8          Kunde 2     Ankunftszeit        28.5
                        Hausmeister Ankunftszeitpunkt  480
-------------------------------------
  27     472.9          Kunde 28    Ankunftszeit       496.1
                        Hausmeister Ankunftszeitpunkt  480
```

Bild 21: Die zukünftige Ereigniskette in unterschiedlichen Phasen für PRO04

Zusammenfassung von Kapitel 3

- Soll der Simulationslauf zu einem festgelegten Zeitpunkt beendet werden, so muss ein Stopp-Ereignis erzeugt werden, welches den Startzähler auf den Wert 0 setzt.
- Eine Zeiteinheit hat in allen Blöcken eines Simulationsprogramms die gleiche Dauer. Wird eine Minute für eine Zeiteinheit in einem Block genutzt, so gilt auch für alle anderen Blöcke, dass Minute für Zeiteinheit steht.

Fragen zum Kapitel 3

1. Was würde passieren, wenn wir den Wert 8 (für Stunden) an Stelle von 480 (für Minuten) in Stopp-Segment des Programms PRO04 verwenden?
2. Wie muss Programm PRO04 verändert werden, so dass die Simulation nach einem Tag mit 24 Stunden stoppt?
3. Entfernen Sie vollständig das Stopp-Segment vom Programm PRO04. Dies wird möglich, indem man die zu löschenden Blöcke mit gedrückter Maustaste markiert (Blocksymbole werden rot) und anschließend über das Kontextmenü der rechten Maustaste löscht. Wie geht WinGPSS mit der nun entstandenen Endlosschleife um?
   (Antwort: Es wird eine Fehlermitteilung ausgegeben: „Kein TERMINATE-Block A-Operanden > 0“)

Übung zum Kapitel 3

Personen kommen am Drehkreuz im Mittel alle 12 Minuten an, wobei die Zwischenankunftszeit zwischen 6 und 18 Minuten schwanken kann. Alle Zeiten in diesem Intervall treten gleich häufig auf. Das Drehkreuz wird nach 8 Stunden oder nach 40 Minuten geschlossen, abhängig davon, welches der beiden Ereignisse zuerst eintritt. Führen Sie drei Simulationsläufe durch und bewerten Sie die Simulationsresultate. Wann wurde die Simulation beendet und wie viele Kunden haben das Drehkreuz passiert?

Anleitung: Wie sollte das Programm aussehen, das die Simulation nach dem Eintreffen von 40 Kunden beendet? Was ist die einfachste Veränderung die Sie durchführen müssen, damit der Simulationslauf nach 8 Stunden beendet wird, auch wenn das wesentlich früher ist als das der 40. Kunde das System betreten hat?

In den ersten beiden Simulationsläufen kommen 40 Kunden an und die Simulation wird vor dem Zeitpunkt 480 beendet (475.02 und 449.05). Im dritten Simulationslauf kommen nur 39 Kunden an und die Simulation wird zum Zeitpunkt 480 beendet.

Kapitel 4

Wir öffnen das Programm PRO04, welches ein Drehkreuz nachbildet. Dieses Drehkreuz wollen wir nun in ein Museum umwandeln. Die Besucher kommen immer noch zufällig alle 12 bis 24 Minuten an. Die Zwischenankunftszeiten sind gleichverteilt und die Simulation wird nach 480 Minuten beendet. Das bedeutet, dass das Museum eine Öffnungszeit von 8 Stunden hat. Bei dem Modell des Drehkreuzes verlassen die Kunden das System zu dem Zeitpunkt, zu dem sie es betreten haben. Im Museum erlauben wir den Besuchern sich eine gewisse Zeit aufzuhalten. Diese schwankt zwischen 20 und 30 Minuten, wobei alle Zeiten in diesem Intervall gleich häufig auftreten können, so wie im Bild 22 dargestellt. Wir bezeichnen dies als Museum, da wir keine Beschränkung der Anzahl der Besucher festgelegt haben, die sich gleichzeitig im Museum aufhalten können.

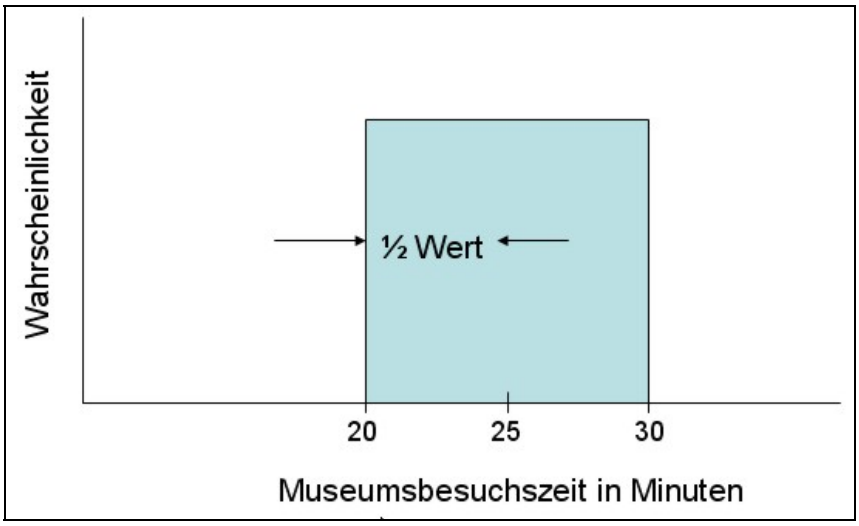


Bild 22: Gleichverteilte Museumsbesuchszeit

Um den Aufenthalt der Besucher im Museum nachbilden zu können müssen wir unser Programm verändern. Es entsteht das Programm **PRO05**, das einen neuen Block, den **ADVANCE-Block** enthält. Das Symbol dieses Blockes ist ein Rechteck. Dieser Block soll zwischen die Blöcke GENERATE und TERMINATE des Kundensegmentes eingefügt werden. Damit ermöglichen wir den Besuchern eine Verweilzeit zwischen Ankunft im und Verlassen des Systems. Um diesen Block ins Blockdiagramm einzufügen, aktivieren wir das Blocksymbol im Blockmenü und fügen es im Blockdiagramm über dem Nachfolgeblock ein. Eine weitere Möglichkeit des Einfügens besteht darin, die Position, vor der der neue Block eingefügt werden soll durch einen Mausklick zu markieren. Der markierte Block ist durch einen roten Kreis in der linken oberen Ecke gekennzeichnet. Anschließend wird der Block durch Klicken mit der rechten Maustaste auf das entsprechende Symbol im Blockmenü eingefügt. Die Darstellung vor und nach dem Einfügen ist im Bild 23 dargestellt.

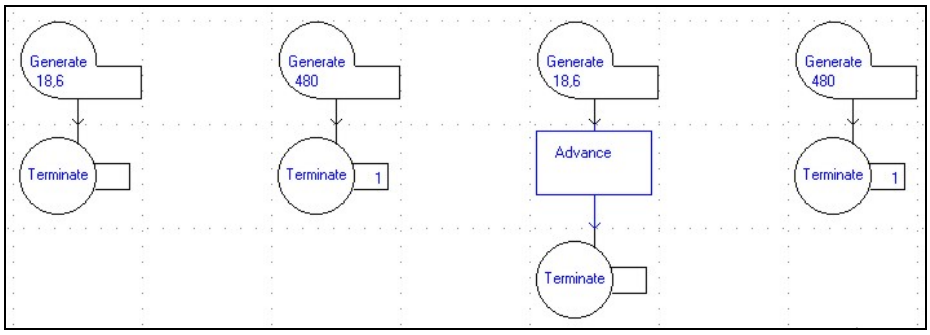


Bild 23: Einfügen des Advance-Blockes im Kundensegment

Als nächstes parametrisieren wir den ADVANCE-Block so dass gleichverteilte Verzögerungen erzeugt werden, wie in Bild 22 dargestellt. Durch einen Doppelklick auf das das Blocksymbol wird das Parameterfenster geöffnet. Wir sehen die beiden Parameter „Aufenthaltszeit (Mittelwert)“ und „Streuung der Aufenthaltszeit (1/2 des Wertes)“. Der Wert 25 wird als Aufenthaltszeit und der Wert 5 als Streuung der Aufenthaltszeit eingetragen, so wie im Bild 24 dargestellt. Nach Schließen des Parameterfensters mit OK sehen wir die 25,5 innerhalb des Blocksymbols des ADVANCE-Blockes. Das bedeutet, dass alle Zeiten zwischen 20 und 30 Minuten gleich häufig auftreten können. Der A- und B- Operand des ADVANCE-Blockes haben die gleiche Bedeutung wie die des GENERATE-Blockes. Die Standardwerte für beide Parameter sind ebenso 0.

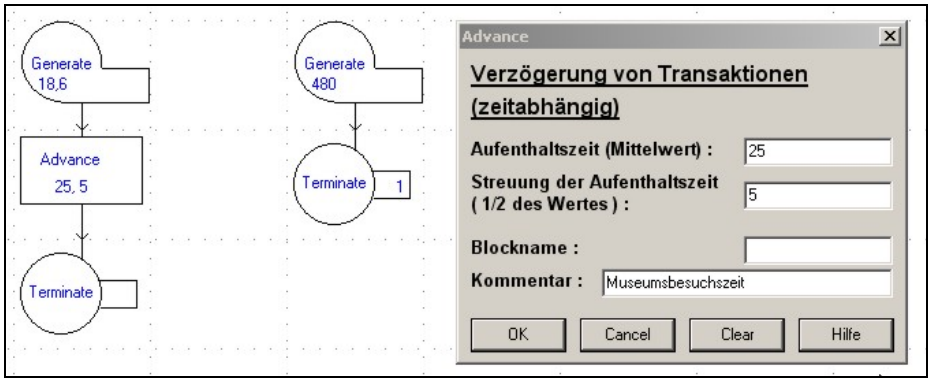


Bild 24: Eintragen der zufälligen Museumsbesuchszeit

Wir starten das Programm. Das Programmlisting enthält keine Überraschungen. Neu ist die Blockstatistik, wie sie im Bild 25 dargestellt ist. In den früheren Simulationsläufen hatten wir Werte ausschließlich in der rechten Spalte, *Gesamt*, in der angezeigt wird, wie viele Transaktionen diesen Block durchlaufen haben. Jetzt haben wir eine Eintragung in der mittleren Spalte mit der der Überschrift *Aktuell*. Für den ersten Simulationslauf sehen wir im Block 1 (GENERATE) unter Gesamt, dass 27 Besucher erzeugt wurden, im Block 3 (TERMINATE) dass nur 26 Besucher das Museum verlassen haben. Wir sehen im Block 2 (ADVANCE), unter Aktuell, dass sich noch ein Besucher im Museum befindet, wenn es geschlossen wird. Das ist ein Besucher, der geplant hat, das Museum nach dem Zeitpunkt 480 zu verlassen, jetzt aber früher gehen muss.

```
Zeit         480.00

 Blockstatistik
 Nummer Name    Oper.    Aktuell Gesamt
     1          GENERA             27
     2          ADVANC      1      27
     3          TERMIN             26
     4          GENERA              1
     5          TERMIN              1
```

Bild 25: Blockstatistik für den ersten Lauf des Museumsprogramms

Wir werden jetzt dieses Programm, PRO05, noch ein wenig verändern. Nehmen wir an, dass die Besucher zwischen ein und zwei Stunden im Museum bleiben wollen. Da alle anderen Zeiten in Minuten angegeben sind (eine Zeiteinheit entspricht in unserem Modell einer Minute) so bedeutet das für die Besucher, dass sie zwischen 60 und 120 Minuten bleiben. Für die Parameter ergibt sich ein Mittelwert der Aufenthaltszeit von 90 und eine Streuung der Aufenthaltszeit von 30 Minuten. Daher werden 90 und 30 als Parameter in den ADVANCE-Block eingetragen. Zur Veränderung der Parameter ist es nicht unbedingt erforderlich, das Dialogfenster zu öffnen. Wir können die Werte auch direkt im TextEdit verändern. Dazu wird der Block angeklickt, durch Wechsel der Farbe auf rot erkennbar. Im TextEdit-Fenster wird die entsprechende Zeile blau markiert und kann geändert werden. Die Änderung der Parameter ist auch möglich, wenn kein Block ausgewählt wurde. Quelltextänderungen werden sofort in das Blockdiagramm übertragen. Das TextEdit-Fenster ist im Bild 26 dargestellt.

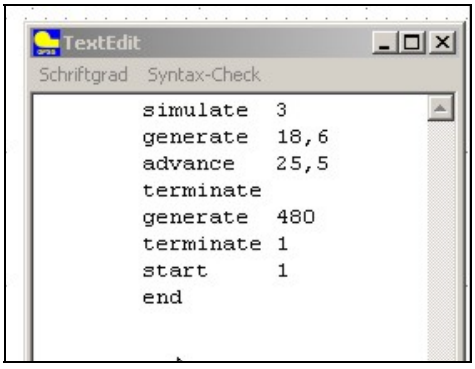


Bild 26: Direkte Parametereingabe im TextEdit-Fenster

Jetzt werden die Werte 90,30 innerhalb des ADVANCE-Blockes dargestellt. Als nächstes starten wir dieses neue Programm, **PRO06**, und sehen uns die Blockstatistik des ersten Simulationslaufes an, so wie sie im Bild 27 dargestellt ist. Wir sehen im Block 2 (ADVANCE) unter Aktuell, das sich zum Zeitpunkt des Schließens noch 5 Besucher im Museum aufhalten. Das zeigt, dass es kein Limit gibt, wie viele Besucher sich gleichzeitig im ADVANCE-Block aufhalten können.

```
Blockstatistik
 Nummer Name    Oper.    Aktuell Gesamt
     1          GENERA             27
     2          ADVANC      5      27
     3          TERMIN             22
     4          GENERA              1
     5          TERMIN              1
```

Bild 27: Auszug aus der Blockstatistik von Programm PRO06

Abschließend sollen noch einige Bemerkungen zur internen Verwaltung des ADVANCE-Blockes im WinGPSS-System gemacht werden. Wenn der erste Besucher ankommt, z.B. zum Zeitpunkt 15.8 im zweiten Simulationslauf von PRO05, so wird eine Zeitdauer aus dem im ADVANCE-Block angegebenen Intervall entsprechend der vorgegebenen Verteilung bestimmt, z.B. 28.6. Das ist die Zeitdauer, die der Besucher im ADVANCE-Block verweilen muss, bevor er den TERMINATE-Block betritt. Daher wird der Zeitpunkt des Verlassens des ADVANCE-Blockes in die ZEK eingetragen, in diesem Fall 15.8 + 26.8 = 42.6, so wie im Bild 28 ersichtlich. Das Ereignis, das aktuell vom ADVANCE-Block gesteuert wird, ist der früheste mögliche Zeitpunkt, zu dem die Transaktion den ADVANCE-Block verlassen kann. In den folgenden Kapiteln wird darauf eingegangen, dass es der Transaktion nicht immer möglich ist, den ADVANCE-Block zur berechneten Zeit zu verlassen. Abschließend sehen wir, dass zum Zeitpunkt 479.5, wenn in diesem Modell der 27. Besucher ankommt, das WinGPSS den Zeitpunkt für das Verlassen des ADVANCE-Blockes für diesen Besucher auf den Zeitpunkt 502.8 festgelegt hat, also wesentlich nachdem der Hausmeister ankommt und den Simulationslauf beendet.

```
Ereignis Ereigniszeitpunkt
  1        0.0          Besucher 1  Ankunftszeit         15.8
                        Hausmeister Ankunftszeitpunkt  480
  2       15.8          Besucher 2  Ankunftszeit         28.5
                        Besucher 1  Verlassen            42.6
                        Hausmeister Ankunftszeitpunkt  480
-------------------------------------
  n       15.8          Besucher28  Ankunftszeit       496.1
                        Hausmeister Ankunftszeitpunkt  480
                        Besucher27  Verlassen          502.8
```

Bild 28: Die zukünftige Ereigniskette zu verschiedenen Ereigniszeitpunkten im Programm PRO06

Zusammenfassung von Kapitel 4

- Der ADVANCE-Block wird genutzt um eine Transaktion zu verzögern. Der früheste mögliche Zeitpunkt, den ADVANCE-Block zu verlassen wird in die ZEK eingetragen.
- Der A- und B-Operand des ADVANCE-Blockes sind denen des GENERATE-Blockes sehr ähnlich. Mit dem A-Operanden wird der Mittelwert der Aufenthaltszeit und mit dem B-Operanden wird die Streuung der Aufenthaltszeit (½ des Wertes) angegeben. Die Aufenthaltszeiten sind in diesem Fall gleichverteilt und es muss $A \geq B$ sein. Der Standardwert für beide Operanden ist 0.

Fragen zum Kapitel 4

1. Wie viele Besucher betreten das Museum im ersten Simulationslauf von PRO06 und wie viele verlassen es, bevor es zum Zeitpunkt 480 geschlossen wird?
2. Wie müssen Sie Programm PRO05 verändern, wenn die Besucher im Mittel 25 Minuten bleiben, wobei einige Besucher das Museum sofort wieder verlassen, andere 50 Minuten bleiben? Besteht der Unterschied zwischen den Simulationsmodellen darin, dass die Anzahl der Besucher, die sich zum Zeitpunkt des Schließens des Museums noch darin befinden größer ist als die im Originalprogramm PRO05, wenn genügend Simulationsläufe, z.B. 20, durchgeführt werden?
3. Angenommen, Sie wollen PRO05 so modifizieren, dass das Museum bereits nach 7 Stunden geschlossen wird. Sie können den Mittelwert der Zwischenankunftszeit im GENERATE-Block des Stoppsegments von 480 auf 420 verändern. Können Sie die Zwischenankunftszeit auch mit 7*60 angeben? Kann WinGPSS arithmetische Ausdrücke in Operanden verarbeiten?
4. Wie reagiert WinGPSS, wenn Sie PRO05 starten und versehentlich die Operanden A und B des ADVANCE-Blockes vertauscht wurden?

Übung 4

Besucher treffen im Mittel mit einem Zeitabstand von 10 Minuten in einem Museum ein. Die Zeiten variieren zwischen 5 und 15 Minuten, wobei alle Zeiten aus diesem Intervall gleich häufig auftreten können. Der Besuch dauert im Mittel eine halbe Stunde. Einige bleiben eine viertel Stunde, andere bleiben eine dreiviertel Stunde. Simulieren Sie einen Tag mit einer Öffnungszeit von 5 Stunden und stellen Sie fest, wie viele Besucher sich zum Zeitpunkt des Schließens noch im Museum befinden. Lassen Sie dieses Programm dreimal mit unterschiedlichen Zufallszahlen laufen. (Die Anzahl solcher Besucher ist 4, 3 und 3.)

Kapitel 5

Bis jetzt lag der Schwerpunkt unserer Betrachtungen auf den temporären Modellelementen, den Transaktionen, z.B. Kunden. Jetzt führen wir eine weitere wichtige Modellkomponente, die Bedieneinrichtungen, ein. Während die Transaktionen temporäre Modellelemente sind, gehören die Bedieneinrichtungen zu den permanenten Modellelementen. Sie sind während des gesamten Simulationslaufes vorhanden. In den letzten Kapiteln haben wir gelernt, dass die Transaktionen erzeugt und durch das System bewegt werden. Die Bedieneinrichtungen sind jedoch vom Start bis zum Ende der Simulation verfügbar. Bild 29 zeigt einige Beispiele für reale Systeme, in denen Transaktionen und Bedieneinrichtungen vorkommen.

reales System	Transaktionen	Bedieneinrichtung
Telefonanlage	Anrufe	Telefonzentrale
Sekretariat	Aufgaben	Sekretärin
Einkaufsmarkt	Kunden	Verkäufer
Bank	Kunden	Bankangestellter, Geld
Transportfirma	Aufträge	Fahrzeuge
Computersystem	Programme	Festplattenlaufwerke
Lagerhaltungssystem	Waren	Lagerkapazität
Fertigungssystem	Teile	Bearbeitungsmaschinen
Hafen	Schiffe	Ankerplätze, Kräne, Schlepper
Krankenhaus	Patienten	Ärzte, Krankenschwestern
Verkehrssystem	Autos	Straßen, Ampeln

Bild 29: Beispiele für Systeme mit Transaktionen und Bediensystemen

In diesem Kapitel wird die einfachste Form einer Bedieneinrichtung betrachtet, die wir als **Einrichtung** bezeichnen. Diese kann nur **eine Transaktion** maximal gleichzeitig bedienen. Aus dem Programm PRO05, unserem Museumsprogramm, entwickeln wir ein Programm, das einen kleinen Friseurladen mit einem Friseur, Beni, nachbildet. Beni kann immer nur einem Kunden gleichzeitig die Haare schneiden. Dazu müssen wir das Programm PRO05 laden. Die Kunden kommen in den gleichen Abständen an wie in PRO05. Der Abstand zwischen 2 Kunden liegt zwischen 12 und 24 Minuten. Die Zeit für das Schneiden der Haare variiert zwischen 20 und 30 Minuten mit einem Mittelwert von 25 Minuten, analog zur Aufenthaltszeit der Besucher im Museum. Die Simulation wird nach einer Arbeitszeit von 8 Stunden beendet. Es wird eine Situation angenommen, wie sie Bild 30 im dargestellt ist.

- Das System ist ein kleiner Friseurladen, mit einem Friseur, dessen Name Beni ist.
- Die Kunden kommen zufällig in einem Abstand zwischen 12 und 24 Minuten gleichverteilt an.
- Die Zeit für einen Haarschnitt schwankt gleichverteilt zwischen 20 und 30 Minuten.
- Beni kann nur einen Kunden gleichzeitig bedienen. Daraus folgt, dass sich eine Warteschlange bildet, wenn Beni die Kunden nicht so schnell bedienen kann, wie sie ankommen.
- Der Friseurladen schließt nach 8 Stunden und die Simulation wird beendet.

Bild 30: Beschreibung des Systems für Programm PRO07

Damit die Transaktionen eine Einrichtung nutzen können, muss ein neuer Block, der SEIZE-Block, eingeführt werden. Wir müssen die Blöcke so verwenden, dass die Transaktionen zuerst den SEIZE und anschließend den ADVANCE-Block betreten. Das Blocksymbol für den SEIZE-Block finden wir als drittes Symbol im Blockmenü. Es besteht aus einem Rechteck, mit der Beschriftung SEIZE und einem Dreieck auf der rechten, unteren Seite des Rechtecks. Zum Einfügen zwischen GENERATE und ADVANCE wird der SEIZE-Block im Blockmenü durch Anklicken ausgewählt und anschließend über dem Block platziert, vor dem er eingefügt werden soll.

Wenn ein Kunde diesen SEIZE-Block betritt, kennzeichnet er den Friseur als belegt. Nachdem die Transaktion die vorgegebene Zeit im ADVANCE-Block verbracht hat, die das Schneiden der Haare nachbildet, so muss der Kunde den Friseur wieder freigeben. Dies wird durch Betreten eines weiteren neuen Blockes, des RELEASE-Blockes, bewirkt. Im Blockmenü befindet sich das Symbol des RELEASE-Blockes unmittelbar neben dem SEIZE-Block. Die Blocksymbole von SEIZE und RELEASE sehen wie gespiegelt aus. Der SEIZE-Block hat an der rechten Seite ein Dreieck, welches mit der Spitze nach oben zeigt, während beim RELEASE-Block die Dreiecksspitze nach unten zeigt. Wie man im Blockmenü erkennen kann, gibt es in WinGPSS mehrere solcher Blockpaare. Das macht es einfacher, die Logik des Blockdiagramms zu verfolgen.

Nun fügen wir den RELEASE-Block hinter dem ADVANCE-Block auf die gleiche Weise ein, wie sie beim SEIZE-Block beschrieben wurde. Wir erhalten dann das Blockdiagramm des neuen Programms PRO07, das im Bild 31 dargestellt ist.

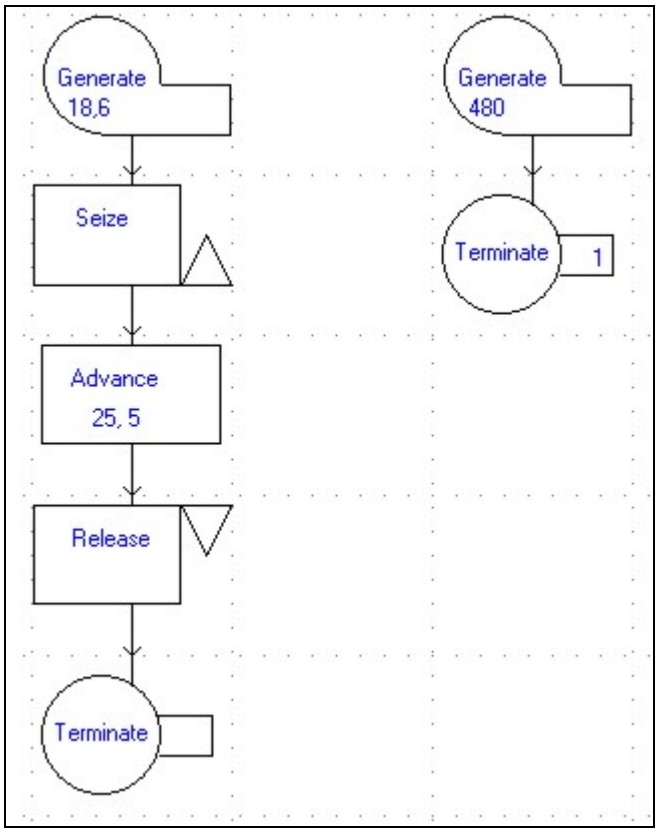


Bild 31: Blockdiagramm zur Beschreibung der Logik von PRO07

Als nächstes werden die beiden eingefügten Blöcke parametrisiert. Zuerst öffnen wir das Parameterfenster des SEIZE-Blockes durch einen Doppelklick auf das Blocksymbol. Der erste Parameter ist der „*Name der Einrichtung*“, wie im Bild 32 dargestellt. In unserem Beispiel wird hier der Name des Friseurs, Beni, eingetragen und das Parameterfenster mit OK geschlossen. Wir sehen jetzt den Namen Beni im Dreieck des SEIZE-Blocksymbols eingetragen, so wie in Bild 33 dargestellt.

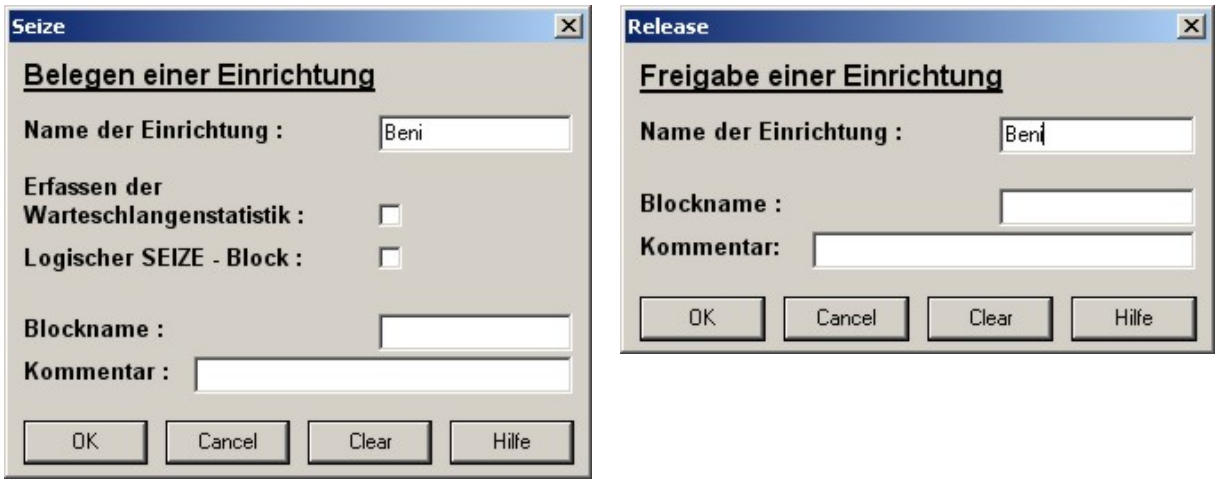


Bild 32: Parameterfenster des SEIZE- und RELEASE-Blocks

In diesem Zusammenhang wollen wir die Regeln für die Vergabe von symbolischen Namen in WinGPSS vorstellen, wie sie z.B. für die Bezeichnung von Einrichtungen, Warteschlangen oder Blöcken verwendet werden. Ein Name besteht immer aus 3 bis 6 Zeichen. Die ersten 3 Zeichen müssen Buchstaben sein, gefolgt von 0 bis 3 Ziffern oder Buchstaben. Buchstaben sind aber nur die kleinen und großen Buchstaben des englischen Alphabetes, deutsche Umlau-

te oder Sonderzeichen anderer Sprachen, wie z.B. das Å von STÅHL sind nicht zulässig. Auch die Verwendung von Sonderzeichen ist nicht erlaubt. So sind z.B. JOE, BENY1a, KASSE und KASSE1 zulässige Namen, AB, JOE+1, AB3C keine zulässigen Namen. In WinGPSS werden in Namen keine Klein- und Großbuchstaben unterschieden.

Abschließend muss der RELEASE-Block parametrisiert werden. Wir öffnen das Parameterfenster des RELEASE-Blocks und finden ebenfalls den Parameter „Name der Einrichtung“, so wie im Bild 32 dargestellt. Wir tragen hier den gleichen Namen ein, den wir auch beim SEIZE-Block verwendet haben, z.B. Beni. Nach dem Schließen des Festers mit OK, sehen wir, dass der Name Beni in das Dreieck des Blocksymbols des RELEASE-Blocks eingetragen wurde, so wie im Bild 33 dargestellt. An dieser Stelle muss eindringlich darauf hingewiesen werden, dass es sehr wichtig ist, genau den gleichen Namen im SEIZE- und im RELEASE-Block zu verwendet, da der Kunde die gleiche Einrichtung wieder freigeben muss, die er auch betreten hat. Würden wir im RELEASE-Block versehentlich anstatt Beni den Namen Benni eintragen so führt das zu einem Laufzeitfehler, da eine Einrichtung freigegeben werden soll, die nicht belegt wurde.

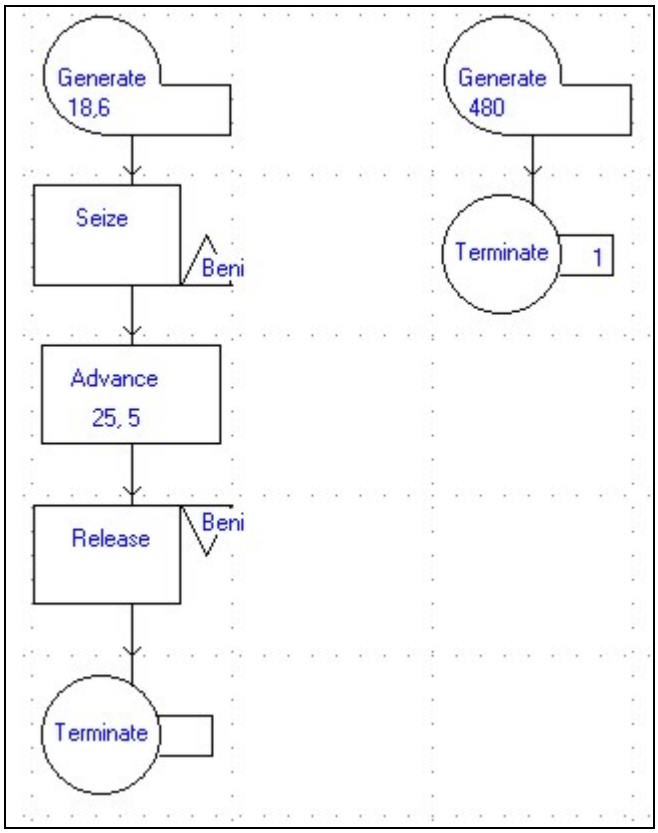


Bild 33: Blockdiagramm von Programm PRO07

Jetzt können wir dieses einfache Modell unseres Friseurladens starten. Obwohl dieses Programm sehr einfach ist, ist es das erste Programm, welches nicht sofort zu überschauen ist und bei dem Resultate ermittelt werden, die nicht mehr offensichtlich sind. Wenn die Kunden den Laden im Mittel alle 18 Minuten erreichen und das Schneiden der Haare im Mittel 25 Minuten dauert, so ist zu erwarten, dass sie Warteschlange im Laufe des Tages immer länger wird. Es ist aber nicht offensichtlich, wie lang die Warteschlange im Laufe des Tages maximal wird. Nehmen wir an, dass sich in unserem Friseurladen vier Stühle für wartende Kunden befinden. Der Friseur möchte nun wissen, ob diese vier Stühle ausreichend sind.

Aus didaktischen Gründen arbeiten wir das Programm nur einmal ab, um aber aus den Resultaten dieses Programms irgendwelche Schlussfolgerungen ziehen zu können, ist es notwendig das Programm mehrfach abzuarbeiten. Darauf wurde in den vorangegangenen Kapiteln schon

mehrfach eingegangen. Diese Aussage bezieht sich auch auf alle noch folgenden Programme. Als erstes sehen wir uns den aufbereiteten Quelltext von PRO07 an, der im Bild 34 dargestellt ist.

```
Block                                                                  Zeile
 Nr.  *Adr.  Operation A,B,C,D,E,F,G,H        Kommentar                 Nr.
             simulate  1                                                  1
   1         generate  18,6               !Erzeugung der Kunden          2
   2         seize     Beni               !Beginn der Bedienung          3
   3         advance   25,5                                              4
   4         release   Beni               !Ende der Bedienung            5
   5         terminate           !Entfernen der Kunden aus dem System     6
   6         generate  8*60                                               7
   7         terminate 1                                                  8
             start     1                                                  9
             end                                                         10
```

Bild 34: Aufbereiteter Quelltext für PRO07

In der Blockstatistik, dargestellt im Bild 35, sehen wir unter *Gesamt* in den Blöcken 1 und 2, dass 27 Kunden den Friseurladen betreten haben, aber nur 19 mit dem Schneiden der Haare in 480 Minuten begonnen haben. Unter der Eintragung *Aktuell* sehen wir im Block 1, dass sich dort eine Warteschlange von 8 Personen gebildet hat, die auf das Schneiden der Haare warten. Daher schlussfolgern wir für dieses Modell, dass für die wartenden Kunden vier Stühle nicht ausreichend sind. Wir erkennen, dass die Bildung einer Warteschlange durch die bloße Verwendung eines SEIZE-Blockes hervorgerufen wurde. Weiterhin erkennen wir, dass unter Aktuell für Block 3 noch ein Kunde bedient wurde, als die Simulation zum Zeitpunkt 480 beendet wurde.

```
Zeit        480.00

 Blockstatistik
 Nummer Name   Oper.   Aktuell Gesamt
     1         GENERA     8      27
     2         SEIZE             19
     3         ADVANC     1      19
     4         RELEAS            18
     5         TERMIN            18
     6         GENERA             1
     7         TERMIN             1
```

Bild 35: Blockstatistik für PRO07

Die 8 Kunden, die auf die Bedienung durch den Friseur warten, werden in der Blockstatistik als aktuell in dem Block angezeigt, der sich **vor dem** SEIZE-Block befindet. Intern werden sie in einer Warteschlange verwaltet, die für den Nutzer nicht sichtbar ist. Während Kunde 1 seine Haare geschnitten bekommt ohne zu warten, ist Friseur Beni beschäftigt, wenn Kunde 2 ankommt und versucht, von ihm bedient zu werden. Kunde 2 hat deshalb zu warten. In der Blockstatistik wird ihm nicht erlaubt den SEIZE-Block zu betreten, so dass er im vorhergehenden Block, dem GENERATE-Block warten muss.

Dies zeigt, wie der SEIZE-Block arbeitet. SEIZE bedeutet sinngemäß, „Versuche von der Einrichtung bedient zu werden“. Ist die Einrichtung, in diesem Beispiel Friseur Beni, frei, so wird ein ankommender Kunde sofort bedient. Damit wechselt die Einrichtung in den Status „belegt“ und es kann kein anderer Kunde bedient werden. Der Kunde, dessen Haare jetzt geschnitten werden bewegt sich zeitlos durch den SEIZE-Block in den nachfolgenden Block. Ist die Einrichtung jedoch belegt, wenn ein Kunde eintrifft um bedient zu werden, d.h. er versucht den SEIZE-Block zu betreten, so wird der Eintritt in den SEIZE-Block verweigert. Der Kunde hat zu warten, bis die Einrichtung wieder frei ist.

Ein RELEASE-Block hat für eine Transaktion keine blockierende Wirkung. Ein Kunde, der einen RELEASE-Block zu betreten will, kann dies ohne Zeitverzögerung tun und anschließend, zum gleichen Zeitpunkt versuchen den nachfolgenden Block zu betreten. Wartet ein Kunde auf die Bedienung in dieser Einrichtung, wie Kunde 2 in unserem Beispiel, so kann er nun die Warteschlange verlassen, den SEIZE-Block durchlaufen und diesen wieder in den Zustand belegt versetzen. WinGPSS entscheidet automatisch, wer zu warten hat und wer bedient wird auf Grundlage des fifo-Prinzips (first come- first serve ::= wer zuerst kommt wird auch als erster bedient), wenn keine andere Strategie festgelegt wurde.

Befindet sich in einem Programm ein SEIZE-Block, so werden zusätzliche Statistiken für Einrichtungen erzeugt. So wie im Bild 36 dargstellt, erhalten wir eine zusätzliche Tabelle auf der Karteikarte Speicher/Einrichtungen mit 3 Eintragungen:

Einrichtung	(1) mittlere Auslastung	(2) Anzahl der Eintritte	(3) mittlere Zeit/Trans
BENI	97.41	19	24.61

Bild 36: Einrichtungsstatistik für Programm PRO07

1. mittlere Auslastung, 97.41, gibt den prozentualen Anteil der Zeit an in der Beni während der gesamten Simulationszeit beschäftigt war
2. Anzahl der Eintritte, 19, gibt die Anzahl der Kunden an, die mit der Bedienung begonnen haben
3. mittlere Zeit/Transaktion, 24.61, gibt die Zeit an, die jeder Kunde im Mittel bedient wurde. Das entspricht der gesamten Arbeitszeit von Beni geteilt durch die Anzahl der Kunden, 19.

Zusammenfassung von Kapitel 5

- Die einfachste GPSS Bedieneinrichtung, mit der genau **eine Transaktion zur selben Zeit** bedient werden kann, ist die **Einrichtung**.
- Die Transaktionen belegen die Bedieneinrichtung durch den **SEIZE**-Block und geben die Bedieneinrichtung durch den **RELEASE**-Block wieder frei.
- Der Name einer Einrichtung darf zwischen 3 und 6 Zeichen lang sein. Die ersten 3 Zeichen müssen Buchstaben sein, die folgenden Zeichen können Buchstaben oder Ziffern sein.
- Versucht eine Transaktion den SEIZE-Block einer Einrichtung zu betreten die belegt ist, so muss diese Transaktion warten. Die Transaktion wartet in dem Block, der sich unmittelbar vor dem zu betretenden SEIZE-Block befindet.
- Die Transaktion ist zusätzlich in einer speziellen Warteschlange für diese Einrichtung eingekettet. Die Warteschlange ist normalerweise nach dem first-come-first-serve-Prinzip organisiert.
- Einer Transaktion wird niemals der Eintritt in einen RELEASE-Block verweigert.
- Enthält ein Programm einen SEIZE-Block, so wird automatisch eine Statistik erzeugt, die neben anderen Daten auch die mittlere Auslastung der Einrichtung enthält.

Fragen zum Kapitel 5

1. Wie stark kann die Länge der Warteschlange am Ende eines Simulationslaufes durch stochastische Einflüsse variieren? Lassen Sie das Programm PRO07 zehn oder zwanzig Mal abarbeiten und finden Sie es heraus.

2. Wie verändert sich die Länge der Warteschlange am Ende des Simulationslaufes, wenn die Öffnungszeit des Friseurladens von 8 auf 12 Stunden erhöht wird?
3. Warum erreicht die mittlere Auslastung des Friseurs niemals 100%, unabhängig davon, wie lange der Friseurladen geöffnet ist und wie oft wir das Programm abarbeiten lassen?
4. Erinnern Sie sich, dass die mittlere Auslastung/ Transaktion die Arbeitszeit geteilt durch die Anzahl der Eintritte ist. Wie beeinflusst das die Berechnung der mittleren Zeit, die Beni für einen Haarschnitt benötigt?

Übung 5

Kunden erreichen ein Geschäft im Mittel alle 16 Minuten, mit 10 Minuten als kürzestem und 22 Minuten als längstem Abstand zwischen der Ankunft zweier Kunden. Alle Zeiten in diesem Intervall können gleich häufig auftreten. Der Eigentümer des Geschäftes arbeitet allein und hat das Geschäft 8 Stunden am Tag geöffnet. Die Bedienung eines Kunden dauert zwischen 10 und 20 Minuten. Als Ergebnis einer Werbekampagne wird angenommen, dass sich die Anzahl der Kunden verdoppelt. Das bedeutet, dass die Kunden jetzt mit einem Mittelwert von 8 Minuten in einem Intervall zwischen 4 und 12 Minuten gleichverteilt ankommen.

Die zu beantwortende Frage ist: Schafft es der Geschäftsinhaber, alle Kunden zu bedienen, ohne das die Warteschlange zu lang wird? Schreiben Sie ein Programm für das System ohne Werbung. Lassen Sie es zwei Mal abarbeiten. Verändern Sie das Programm so, das die Ergebnisse der Werbekampagne berücksichtigt werden. Lassen Sie auch dieses Modell zwei Mal abarbeiten. Ist es zweckmäßig für den Ladenbesitzer, die Werbekampagne durchzuführen?

(Nein, während des Normalbetriebes gibt es keine Warteschlangen, in dem Modell mit der Werbekampagne ergibt sich eine Warteschlange von 26 und 29 Kunden.)

Kapitel 6

Die Ergebnisse von PRO07 im Kapitel 5 zeigten die Länge der Warteschlange am **Ende des Simulationslaufes**. Es stehen keine Informationen zur Verfügung, wie sich die Länge der Warteschlange während der Simulation entwickelt. Um diese detailliertere Warteschlangenstatistik zu bekommen laden wir Programm PRO07 erneut und öffnen das Parameterfenster des SEIZE-Blockes. So wie im Bild 37 dargestellt, gibt es eine Checkbox mit der Bezeichnung „Erfassen der Warteschlangenstatistik“. Durch Anklicken wird diese Checkbox aktiviert. Nach Schließen dieses Parameterfensters mit OK erscheint das q als Operand im Blocksymbol des SEIZE-Blockes in der rechten, unteren Ecke.

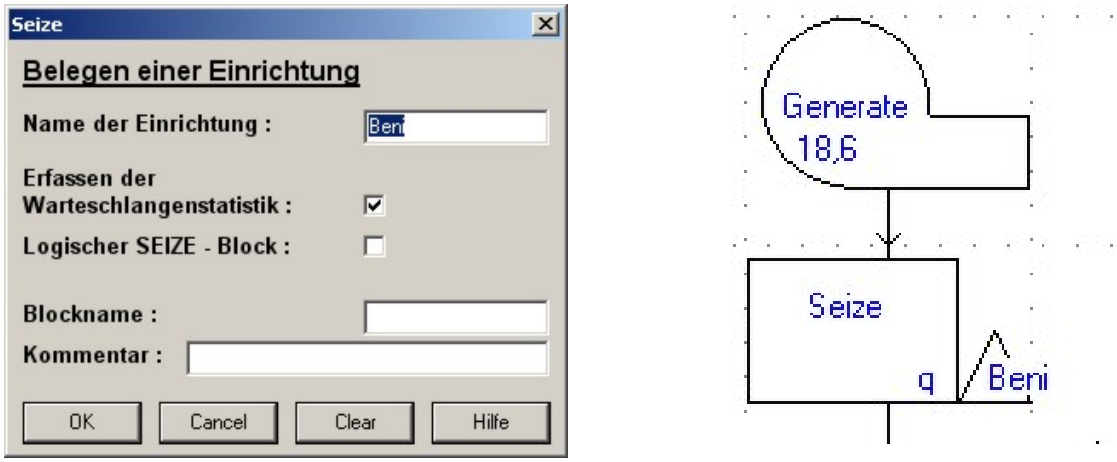


Bild 37: SEIZE-Parameterfenster und SEIZE-Blocksymbol mit Warteschlangenkennzeichen

Nun starten wir das neue Programm, PRO08. Im aufbereiteten Quelltext erkennen wir, dass im Vergleich zu PRO07 der zweite Block jetzt SEIZE Beny,Q ist.

```
Block                                                                  Zeile
 Nr.   *Adr.  Operation A,B,C,D,E,F,G,H         Kommentar               Nr.
              simulate  1                                                 1
   1          generate  18,6               !Erzeugung der Kunden        2
   2          seize     Beni,q             !Beginn der Bedienung        3
   3          advance   25,5                                            4
   4          release   Beni               !Ende der Bedienung          5
   5          terminate           !Entfernen der Kunden aus dem System   6
   6          generate  8*60                                            7
   7          terminate 1                                               8
              start     1                                               9
              end                                                      10
```

Bild 38: Aufbereiteter Quelltext von PRO08

Die Blockstatistik und die Statistik für die Einrichtung sind bei den Programmen PRO07 und PRO08 identisch. Das Hinzufügen des Operanden Q zum SEIZE-Block hat keinen Einfluss auf den Simulationslauf. Mit dem Operanden Q wird nur eine weitere Statistik erzeugt. Im Resultatfenster wird eine neue Karteikarte mit der Bezeichnung „Warteschlangen“ eingefügt. Die erzeugte Statistik für Programm PRO08 ist im Bild 39 dargestellt.

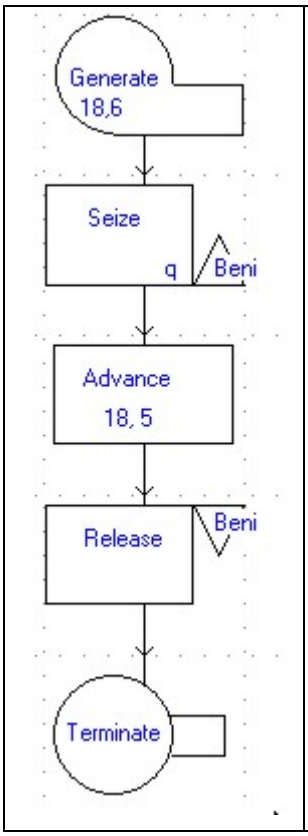
```
                (1)          (2)          (3)          (4)          (5)
Schlange     Maximum      Mittel       Gesamt      Zeitlos     Prozent
             Inhalt       Inhalt     Eintritte    Eintritte    zeitlos
BENI            8          3.78         27            1          3.70


                (6)          (7)          (8)
Schlange    mittlere     $mittlere    aktueller
          Zeit/Trans   Zeit/Trans     Inhalt
BENI         67.25        69.83          8
$mittlere Zeit/Trans ::= mittlere Zeit/Trans ohne zeitlose Eintritte
```

Bild 39: Warteschlangenstatistik vom Programm PRO08

Hier werden acht Informationen angezeigt, die zur Warteschlange von Beni gehören:

1. **Maximaler Inhalt (Maximum contents)**, 8, gibt an, wie viele Kunden sich, gemessen über die gesamte Simulationszeit, maximal gleichzeitig in der Warteschlange befunden haben.
2. **Mittlerer Inhalt (Average contents)**, 3.78, ist die mittlere Anzahl der Kunden die während des Tages warteten. Sie berechnet sich aus der Summe der Wartezeiten aller Kunden in der Warteschlange geteilt durch die Simulationszeit, 480.
3. **Gesamte Eintritte (Total entries)**, 27, gibt an, wie viele Kunden die Warteschlange betreten haben, also mit dem Warten begonnen haben. In diesem Fall ist es die gleiche Anzahl die den Block 1 betreten hat, wie in Bild 35 ersichtlich.
4. **Zeitlose Eintritte (Zero entries)**, 1, gibt an, wie viele Kunden nicht in der Warteschlange warten mussten, weil die Einrichtung Beni frei war, als sie ankamen. Das Bedeutet, dass der Zeitpunkt des Betretens gleich dem Zeitpunkt des Verlassens der Warteschlange ist. In diesem Modell musste nur der erste Kunde nicht warten.
5. **Prozent zeitlos (Percent zeroes)**, 3.70, gibt das Verhältnis der zeitlosen Eintritte, 1, zu den gesamten Eintritten, 27, in Prozent an.
6. **Mittlere Zeit/Trans (Average time per Transaktion)**, Mittlere Wartezeit der Transaktion in der Warteschlange, 67.25, berechnet sich aus der Summe aller Aufenthaltszeiten der Transaktionen dividiert durch die Anzahl der Eintritte, 27.
7. **$Mittlere Zeit/Trans ($Average time per Transaktion)**, 69.83, gibt die durchschnittliche Wartezeit je Transaktion unter Berücksichtigung der zeitlosen Eintritte an. Dabei wird die Wartezeit nur durch die Anzahl der wirklich wartenden Transaktionen dividiert (27 - 1 = 26). Dies ist die mittlere Wartezeit der Transaktionen, die wirklich warten mussten, weil die Einrichtung belegt war
8. **Aktueller Inhalt (Current contents)**, 8, gibt die Anzahl der Kunden oder Transaktionen in der Warteschlange am Ende der Simulation an und hat damit den gleichen Wert wie der aktuelle Inhalt des Blockes 1 in der Blockstatistik, wie in Bild 35 ersichtlich.

Da die Kunden mit einer mittleren Zwischenankunftszeit von 18 Minuten ankommen, das Schneiden der Haare im Mittel jedoch 25 Minuten dauert, ist es klar, dass die Warteschlange ständig länger wird. Es stellt sich jedoch die Frage: „Liegt es ausschließlich am Unterschied dieser beiden Zeiten, dass sich eine Warteschlange bildet?“ Um diese Frage zu beantworten, öffnen wir das Parameterfenster des ADVANCE-Blockes und verändern den Mittelwert der Dauer des Haareschneidens auf 18 Minuten. Damit haben Zwischenankunftszeit und Bedienzeit den gleichen Mittelwert. Das Blockdiagramm ist im linken Teil von Bild 40 dargestellt. Starten wir dieses Programm, so erhalten wir die Blockstatistik und die Warteschlangenstatistik, wie im rechten Teil von Bild 40 dargestellt.

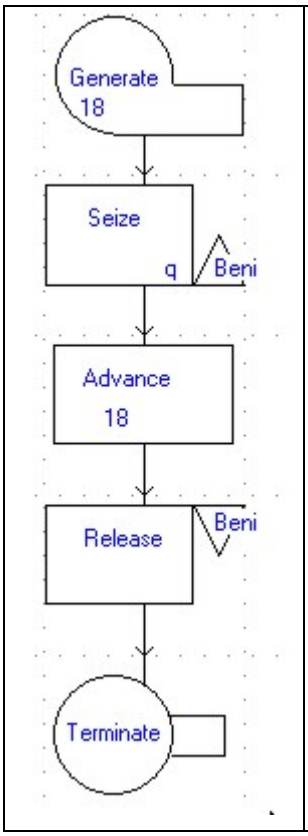



```
Blockstatistik
--------------

 Zeit        480.00

 Blockstatistik
 Nummer Name   Oper.   Aktuell Gesamt
     1         GENERA     1      27
     2         SEIZE              26
     3         ADVANC     1      26
     4         RELEAS             25
     5         TERMIN             25
     6         GENERA              1
     7         TERMIN              1

------------------------------------------------------------

Warteschlangen
--------------

                (1)         (2)         (3)         (4)         (5)
 Schlange     Maximum      Mittel      Gesamt     Zeitlos    Prozent
              Inhalt       Inhalt    Eintritte   Eintritte   zeitlos
 BENI            2          0.76        27           2        7.41

                (6)          (7)          (8)
 Schlange   mittlere     $mittlere    aktueller
          Zeit/Trans    Zeit/Trans    Inhalt
 BENI         13.43         14.50          1
 $mittelere Zeit/Trans ::= mittlere Zeit/Trans ohne zeitlose Ein-
tritte
```

Bild 40: Kundensegment, Block und Warteschlangenstatistik von PRO09

Wir sehen schon in der Blockstatistik, unter aktuell im Block 1, dass es eine Warteschlange gibt, die zwar wesentlich kürzer ist als im PRO08, aber zu diesem Zeitpunkt eine Person beinhaltet. Weiterhin sehen wir in der Warteschlangenstatistik, dass die mittlere Wartezeit wesentlich kürzer geworden ist, aber immer noch etwa eine Viertelstunde beträgt. Die Anzahl der Kunden, die nicht warten mussten hat sich auf 2 erhöht. Damit haben immer noch 93% der Kunden (25 von 27) gewartet.

Nun stellt sich die Frage: „Ist die Ursache für die Entstehung der Warteschlangen vielleicht die zufällige Streuung der Zwischenankunfts- und Bedienzeiten?“. Um diese Frage zu beantworten, verändern wir das Programm so, dass wir keine zufälligen Abweichungen mehr haben. Im GENERATE-Block des Kundensegments löschen wir die 6 als halbe Abweichung der Zwischenankunftszeit und im ADVANCE-Block löschen wir die 5 als halbe Abweichung. Damit haben wir GENERATE 18 und ADVANCE 18, so wie im Bild 41 dargestellt.



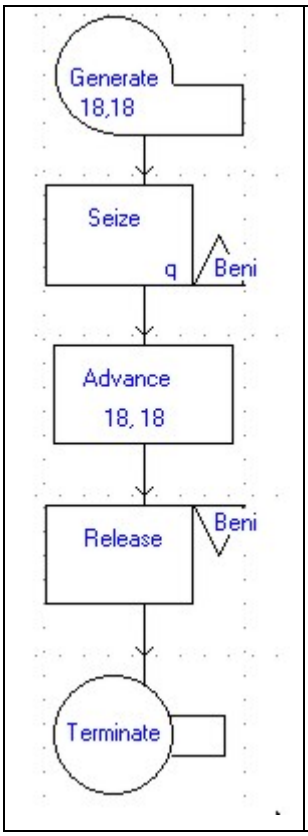

```
Blockstatistik
--------------

 Zeit        480.00

 Blockstatistik
 Nummer Name   Oper.   Aktuell Gesamt
     1         GENERA            26
     2         SEIZE             26
     3         ADVANC     1      26
     4         RELEAS            25
     5         TERMIN            25
     6         GENERA             1
     7         TERMIN             1

------------------------------------------------------------------------

Warteschlangen
--------------

                (1)         (2)         (3)         (4)         (5)
 Schlange     Maximum      Mittel      Gesamt     Zeitlos     Prozent
              Inhalt       Inhalt    Eintritte   Eintritte   zeitlos
 BENI            0          0.00        26          26       100.00


                (6)           (7)          (8)
 Schlange   mittlere      $mittlere    aktueller
           Zeit/Trans    Zeit/Trans    Inhalt
 BENI          0.00          0.00          0
 $mittelere Zeit/Trans ::= mittlere Zeit/Trans ohne zeitlose Ein-
tritte
```

Bild 41: Kundensegment, Block und Warteschlangenstatistik von PRO10

Nach dem Start von Programm PRO10 sehen wir in der Blockstatistik, dass kein Kunde im Block 1 wartet. In der Warteschlangenstatistik erkennen wir, dass die Wartezeit 0 ist und der Anteil der Kunden die nicht warten mussten beträgt 100%. Die Warteschlange im Programm PRO09 wurde demzufolge von der zufälligen Streuung der Zeiten hervorgerufen.

Das unterstreicht die Bedeutung der Verwendung von Zufallsvariablen für Systeme mit erkennbar zufälligen Einflussgrößen. Um die Bedeutung großer Streuungen in Zwischenankunfts- und Bedienzeiten zu demonstrieren, verändern wir im GENRATE- und im ADVANCE-Block den halben Wert der Streuung jeweils auf 18, so dass wir GENERATE 18,18 und ADVANCE 18,18 erhalten, so wie im Bild 42 dargestellt.

Starten wir das Programm PRO10b, so erhalten wir eine Warteschlangenlänge am Ende der Simulation von 5 und die Warteschlangenstatistik weist eine mittlere Wartezeit von etwa einer Stunde aus.




```
Blockstatistik
--------------

 Zeit        480.00

 Blockstatistik
 Nummer Name   Oper.   Aktuell Gesamt
     1         GENERA     5      29
     2         SEIZE             24
     3         ADVANC     1      24
     4         RELEAS            23
     5         TERMIN            23
     6         GENERA             1
     7         TERMIN             1

------------------------------------------------------------------------

Warteschlangen
--------------


                (1)         (2)         (3)         (4)         (5)
 Schlange     Maximum      Mittel      Gesamt     Zeitlos     Prozent
              Inhalt       Inhalt    Eintritte   Eintritte   zeitlos
 BENI           6          3.25        29          2         6.90


                (6)           (7)          (8)
 Schlange   mittlere      $mittlere    aktueller
          Zeit/Trans    Zeit/Trans     Inhalt
 BENI        53.76         57.75          5
 $mittelere Zeit/Trans ::= mittlere Zeit/Trans ohne zeitlose Ein-
tritte
```

Bild 42: Kundensegment, Block und Warteschlangenstatistik von PRO10b

Zusammenfassung Kapitel 6

- Wird die Checkbox für Warteschlangenstatistiken im Parameterfenster des SEIZE-Blocks aktiviert, so erhält man eine Warteschlangenstatistik, die sich auf die Warteschlange bezieht, die in diesem SEIZE-Block angegeben ist.
- Die Warteschlangenstatistik enthält neben anderen Ergebnissen Angaben über die maximale Anzahl von Kunden, die während dieses Simulationslaufes gleichzeitig vor dieser Einrichtung gewartet hat, die mittlere Anzahl die während dieses Tages gewartet hat, die Anzahl der Kunden, die nicht gewartet hat und die mittlere Zeit, die jeder Kunde gewartet hat.
- Durch Veränderung der Parameter für die Zwischenankunftszeit und die Bedienzeit der Kunden haben wir gezeigt, dass die Bildung von Warteschlangen nicht nur davon abhängig ist, dass die Zwischenankunftszeit kürzer ist der Mittelwert der Bedienzeit. Die Bildung von Warteschlangen ist auch von der Streuung der Zeiten abhängig, die in diesen Beispielen durch den B-Operanden des GENERATE- und ADVANCE-Blockes im Kundensegment angegeben wird.

Fragen zum Kapitel 6

1. Wie stark kann die Wartezeit in der Warteschlangenstatistik durch statistische Einflüsse variieren? Arbeiten Sie das Programm PRO10b 20-mal ab um es herauszufinden.
2. Wie verändert sich die mittlere Länge der Warteschlange im Programm PRO08, wenn der Friseurladen täglich 12 anstatt 8 Stunden geöffnet ist?
3. Erinnern wir uns daran, dass sich die mittlere Zeit/Transaktion in der Warteschlangenstatistik berechnet, indem die gesamte Wartezeit durch die Anzahl der Eintritte dividiert wird. Eintritte bedeutet, dass alle Kunden erfasst werden, die mit dem Warten vor Ende des Simulationslaufes begonnen haben. Wie ändert sich dieser Wert für Programm

PRO08, wenn die Berechnung statt dessen mit den Kunden durchgeführt wird, die das Warten bereits beendet haben? Das sind die Kunden, die in den simulierten 8 Stunden mit der Bedienung begonnen haben.

Aufgabe 6:

An einem Informationsstand, an dem eine Person arbeitet, kommen durchschnittlich 10 Touristen pro Stunde an. Die Zwischenankunftszeit schwankt zwischen 0 und 12 Minuten. Die Zeit für die Bedienung eines Kunden schwankt zwischen 0 und 8 Minuten. Für beide Zeiten wird angenommen, dass sie gleichverteilt sind. Im ersten Experiment untersuchen wir, wie sich das System verhält, wenn der Informationsstand nur 100 Minuten geöffnet ist. Wir arbeiten das Programm 5-mal ab und notieren die mittleren Wartezeiten für diese 5 Simulationsläufe. Nun untersuchen wir den Fall, das der Informationsstand 1000 Minuten geöffnet ist. Wir arbeiten das Programm wieder 5-mal ab und notieren die mittleren Wartezeiten. Wir vergleichen diese Werte mit denen der 5 Läufe des ersten Experiments.

Wir erhalten:

7.65, 4.37, 1.57, 0.31, 1.34 im ersten Experiment verglichen mit

2.47, 2.68, 1.89, 2.47, 1.71 im zweiten Fall.

Kapitel 7

Die Warteschlangenstatistiken, die wir bisher betrachtet haben, sind in vielen Fällen unzureichend, da sie sich nur auf Mittelwerte beziehen. Es gibt Anwendungen, für die es wichtig ist, extremwerte zu kennen, z.B. wie viele Kunden länger als zwei Stunden warten. Wir benötigen dazu eine Tabelle für die Wartezeiten, eingeteilt in unterschiedliche Zeitklassen, z.B. 10-20 Minuten,…,110-120 Minuten, usw.. In GPSS werden solche Statistiken in Tabellen (Q-Tables) erfasst. Wir werden dazu Programm PRO08 ausbauen. Durch Hinzufügen einer Tabelle erhalten wir Programm PRO11. Dazu wählen wir im Menü `Parameter` den Punkt `Tabellen` aus. Dann erhalten wir das im Bild 43 dargestellte Dialogfenster.

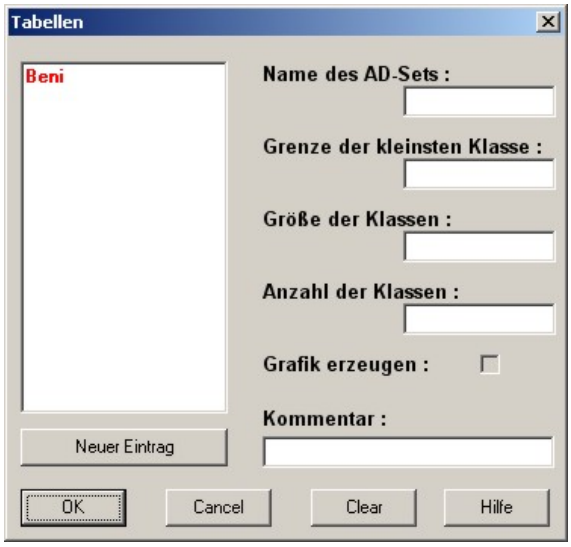


Bild 43: Tabellen-Dialogfenster für Warteschlangentabellen

Im linken Teil des Dialogfensters werden alle verfügbaren Warteschlangen angezeigt. Durch Anklicken des entsprechenden Namens wird dieser in das Feld „Name des AD-Sets“ übernommen. In diesem Fall haben wir nur eine Warteschlange, mit dem Namen Beni, dargestellt im Bild 44.

Bild 44: Einfügen des Namens in das Tabellen-Dialogfenster für Warteschlangentabellen

Es stehen drei weitere Felder für Eingabeparameter zur Verfügung:

Grenze der kleinsten Klasse legt die obere Grenze für die Klasse 1 fest. Wir setzen hier 0 ein. Das bedeutet, dass alle Kunden, die nicht gewartet haben, in dieser Klasse angezeigt werden.

Größe der Klassen legt die Anzahl der Zeiteinheiten für jede Klasse (ausgenommen die kleinste und die größte Klasse) fest. Da wir für unser Beispiel die Klassen 0-10, 10-20 Minuten usw. haben wollen, tragen wir den Wert 10 ein.

Anzahl der Klassen legt die maximale Anzahl der Klassen fest, die wir in unserer Tabelle haben wollen. Diesen Wert setzen wir auf 20, das ist der höchste Wert, den WinGPSS zulässt.

Die entsprechenden Klassen sind im Bild 45 dargestellt. Danach hat Klasse 1 das Intervall von minus unendlich bis 0 (einschließlich). Klasse 2 erfasst alle Werte, die größer als 0 sind, aber kleiner oder gleich 10. Die letzte Klasse, die zwanzigste, erfasst alle Werte, die größer als 180 sind.

..0]	(0..10]	(10..20]	(20..30]	…	(170..180]	(180..
1.Klasse	2.Klasse	3.Klasse	4.Klasse	…	19.Klasse	20.Klasse

Bild 45: Klassen der Warteschlangentabelle

Nachdem wir diese Parameter eingetragen haben schließen wir das Fenster mit dem OK-Button.

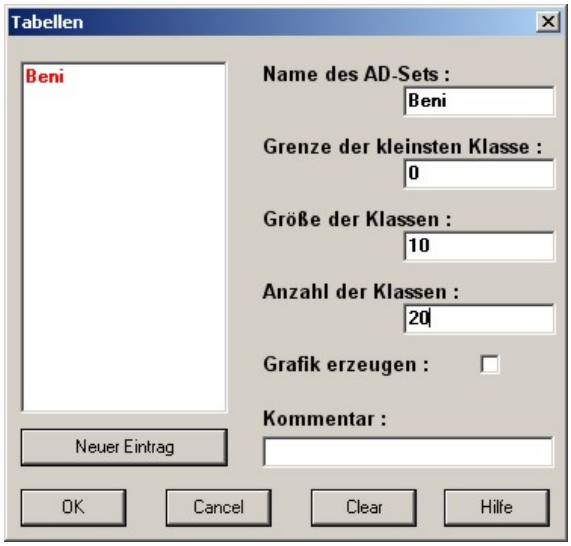


Bild 46: Parametrisierung des Tabellen-Dialogfenster für Warteschlangentabellen

Anschließend starten wir das Programm. Im Quelltext erkennen wir, dass zwischen SIMULATE und GENERATE die Anweisung

```
qtable    Beni,0,10,20
```

eingefügt wurde. Das ist der einzige Unterschied zum Programm PRO08. Es soll erwähnt werden, dass der Name der Warteschlange, Beni, der A-Operand ist, die Grenze der kleinsten Klasse der B-Operand, die Größer der Klassen der C-Operand und die Anzahl der Klassen der D-Operand ist.

Weiterhin erkennen wir, dass die Statistiken für die Blöcke, Einrichtungen und Warteschlangen genau mit denen von Programm PRO08 übereinstimmen. Den Unterschied sehen wir im Resultatfenster. Es wurde die Karteikarte „Tabellen“ eingefügt. Der Inhalt ist im Bild 47 dargestellt.

```
Tabelle: BENI
     (1)         (2)         (3)         (4)         (5)          (6)
 Eintritte  mitt. Zeit   Standardabw. Summe Zeit    Minimum     Maximum
    19         68.85        39.97      1308.15        0.00       126.25


       Klasse           Anzahl der   prozentualer   kumulativer   kumulativer
                       Eintragungen     Anteil      Prozentsatz       Rest
           -     0            1           5.26          5.26          94.74
     0.01 -     10            1           5.26         10.53          89.47
    10.01 -     20            1           5.26         15.79          84.21
    20.01 -     30            1           5.26         21.05          78.95
    30.01 -     40            2          10.53         31.58          68.42
    40.01 -     50            0           0.00         31.58          68.42
    50.01 -     60            2          10.53         42.11          57.89
    60.01 -     70            1           5.26         47.37          52.63
    70.01 -     80            1           5.26         52.63          47.37
    80.01 -     90            3          15.79         68.42          31.58
    90.01 -    100            1           5.26         73.68          26.32
   100.01 -    110            1           5.26         78.95          21.05
   110.01 -    120            2          10.53         89.47          10.53
   120.01 -    130            2          10.53        100.00           0.00
 Alle weiteren Klassen haben keine Eintragungen
```

Bild 47: Warteschlangentabelle im Resultatfenster

Als erstes haben wir eine Zeile mit 6 Resultatdaten:

1. **Eintritte (Entries)**, In der Tabelle erfasste Eintritte, 19. Die durch QTABLE erzeugte Tabelle erfasst nur die Kunden, die die Warteschlange bereits **verlassen** haben. Diese Anzahl unterscheidet sich von der Anzahl der Kunden, die durch die einfache Warteschlangenstatistik erfasst werden, 27. Die einfache Warteschlangenstatistik erfasst die Kunden, die die Warteschlange **betreten** haben.
2. **mitt. Zeit (Mean queue time)**, Mittelwert der Aufenthaltszeit in der Warteschlange, 68.85, gibt die durchschnittliche Wartezeit der Kunden an, die die Warteschlange bereits wieder verlassen haben.
3. **Standardabw. (St. dev.)**, 39.97, ist ein Maß für die Beschreibung der Streuung der Wartezeiten in dieser Tabelle.
4. **Summe Zeit (Total time),**1308.15 berechnet die Summe aller Zeiten, die die 19 in der Tabelle erfassten Kunden gewartet haben.
5. **Minimum (Minimum)**, 0, gibt die kürzeste Wartezeit eines Kunden, in diesem Fall des ersten, an.
6. **Maximum (Maximum),**126.25, gibt die längste Wartezeit an.

Nach dieser Kopfzeile folgt die Ausgabe der Tabelle mit fünf Spalten:

Spalte 1: **Klasse (Range)** gibt an, dass die erste Klasse nur die 0 enthält, die zweite Klasse den Wertebereich von >0 - 10 Minuten und die dritte Klasse den Wertebereich >10-20 Minuten, usw. umfasst.

Spalte 2: **Anzahl der Eintragungen (Observed frequency),** wird die Anzahl der Kunden ausgegeben, die in dieser Klasse erfasst wurden.

Spalte 3: **Prozentualer Anteil (Percent of total),** wird der prozentuale Anteil der Kunden dieser Zeile an den insgesamt erfassten Kunden, 19, ausgegeben.

Spalte 4: **Kumulativer Prozentsatz (Cumulative percentage)** werden die prozentualen Anteile der Spalte 3 kumulativ erfasst. Damit wird es leichter, die Frage zu beantworten „Welcher Anteil der Kunden (in Prozent) wartetet **höchstens** 20 Minuten?“. (In unserem Beispiel ist die Antwort 15.79%.)

Spalte 5: **Kumulativer Rest (Cummulative remainder)** enthält die Differenz zwischen 100 und dem Wert in Spalte 4, zum Beispiel um die Frage zu beantworten, „Welcher Anteil der Kunden (in Prozent) wartete **länger** als eine Stunde?“ (57.89%)

Obwohl bei der Parametrisierung 20 Klassen festgelegt wurden, enthält unsere Tabelle nur 14. Die höchste Klasse hat den Wertebereich von >120 bis 130. Die restlichen Klassen, die keine Eintragungen haben, werden nicht angezeigt. Daher ist es kein Fehler, zu viele Klassen anzugeben.

Die Werte der Tabelle können auch in einem Histogramm dargestellt werden. Dazu öffnen wir erneut das Parameterfenster der Tabellen. Wir wählen durch Anklicken die Tabelle für Beni aus. Die eingegebenen Parameter werden übernommen. Zur Erzeugung des Histogramms wird die Checkbox „Graphik erzeugen“ aktiviert, so wie im Bild 48 dargestellt.

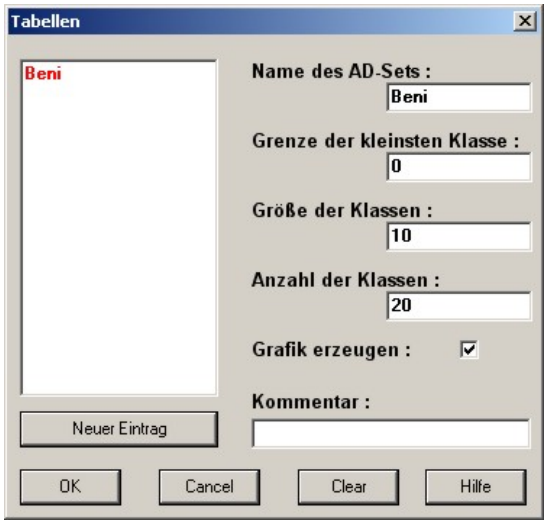


Bild 48: Parameterfenster für die Warteschlangentabelle – Anforderung Histogramm

Anschließend starten wir das Programm erneut. Im Resultatfenster ist die Karteikarte „Tabellengraph“ hinzugekommen. Wenn wir diese Karteikarte aktivieren, sehen wir die im Bild 49 dargestellte Graphik. Die Werte an der Abzissenachse geben die oberen Grenzen der Klassen an.

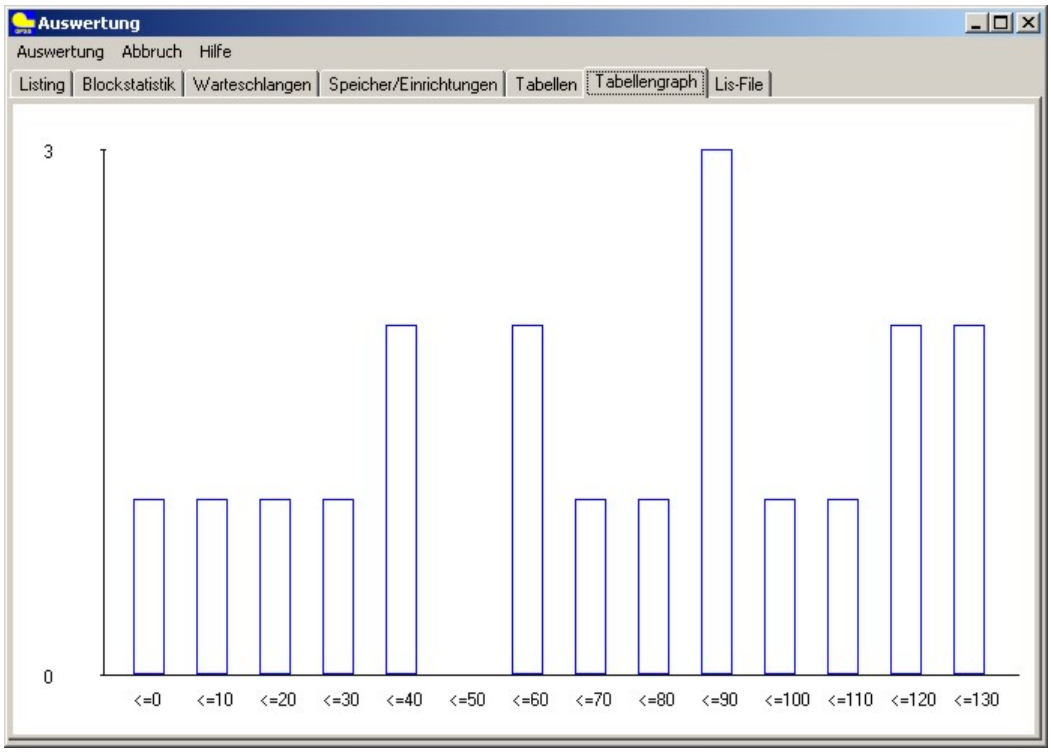


Bild 49: Histogramm für die Warteschlangentabelle

Wir sehen, dass es vorkommen kann, dass wir Klassen ohne eine Eintragung haben. Die Ursache kann sein, dass wir zu viele Klassen deklariert haben, z.B. 20. Als nächstes sehen wir uns an, was passiert, wenn wir zu wenige Klassen deklarieren. Wir öffnen erneut das Parameterfenster für Tabellen, wählen Beni aus und reduzieren die Anzahl der Klassen auf 5. Mit dem OK-Button schließen wir das Parameterfenster. Anschließend starten wir das Programm, PRO12.

Im Bild 50 sehen wir, dass wir nur noch 5 Klassen haben. Die höchste, die 5. Klasse enthält die Kunden, die mehr als 30 und höchstens 40 Minuten gewartet haben. Es gibt aber viele Kunden, die länger warten müssen. Diese werden in einer Klasse zusammengefasst, die als Überlauf bezeichnet wird. In unserem Beispiel sind hier 13 Kunden oder 68.42% enthalten, die mehr als 40 Minuten warten. In der letzten Zeile wird angezeigt, dass diese 13 Kunden im Durchschnitt 90.9 Minuten warten. Obwohl die Überlaufklasse in diesem Fall die Auswertung beeinträchtigt, kann es bei anderen Anwendungen, die später vorgestellt werden, vorteilhaft sein, mit dieser Überlaufklasse zu arbeiten.

```
Tabelle: BENI
     (1)          (2)          (3)          (4)          (5)          (6)
 Eintritte   mitt. Zeit   Standardabw. Summe Zeit    Minimum     Maximum
    19          68.85        39.97       1308.15        0.00       126.25


       Klasse           Anzahl der   prozentualer   kumulativer   kumulativer
                       Eintragungen     Anteil      Prozentsatz       Rest
           -     0            1          5.26          5.26          94.74
     0.01 -    10            1          5.26         10.53          89.47
    10.01 -    20            1          5.26         15.79          84.21
    20.01 -    30            1          5.26         21.05          78.95
    30.01 -    40            2         10.53         31.58          68.42
        Überlauf             13         68.42        100.00           0.00
     (7) Mittelwert des Überlaufes          90.90
```

Bild 50: Warteschlangentabelle mit Überlauf

Das aus diesen Daten erzeugte Diagramm sehen wir in im Bild 51. Es enthält nur die 5 Balken für die definierten Klassen. Die Restklasse wird nicht angezeigt.

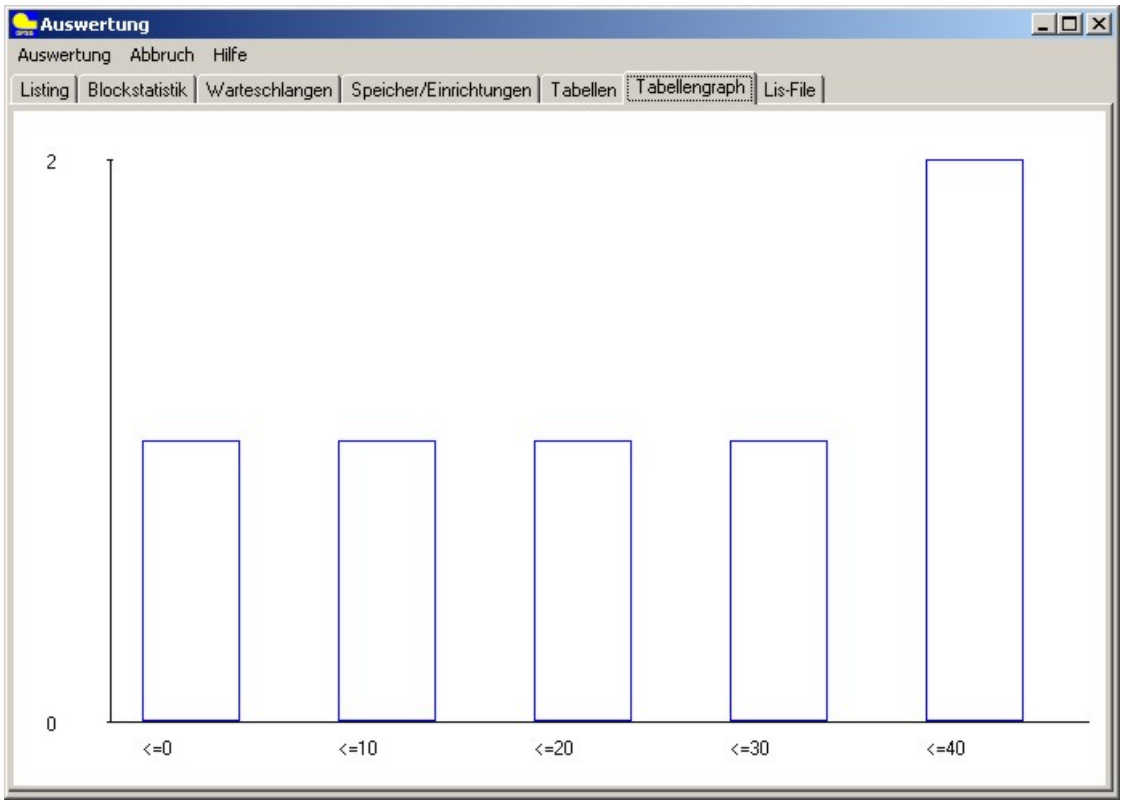


Bild 51: Histogramm mit Restklasse

Zusammenfassung Kapitel 7

- Warteschlangentabellen können zur Erfassung der Wartezeiten in unterschiedlichen Zeitklassen eingesetzt werden. Es kann z.B. gezeigt werden, wie viele Kunden zwischen 10 und 20 Minuten gewartet haben.
- Die Parametrisierung der Warteschlangentabellen erfolgt über den Menüpunkt `Parameter>Tabellen`. Der Name wird aus der Liste der verfügbaren Warteschlangenstatistiken ausgewählt.
- Zur Parametrisierung der Warteschlangentabelle werden im Dialogfenster die Grenze der kleinsten Klasse, die Größe der Klassen und die Anzahl der Klassen festgelegt.
- Warteschlangentabellen enthalten neben weiteren Daten auch Angaben über die mittlere Wartezeit, die Standardabweichung von dieser Zeit, die längste und kürzeste Wartezeit, die Anzahl der Kunden und ihr prozentulaer Anteil in einer Klasse. Die Anzahl der Kunden in den Klassen kann zusätzlich graphisch in Diagrammform ausgegeben werden.

Fragen zum Kapitel 7

1. Angenommen, jemand möchte herausfinden, wie viele Kunden mit einer Wartezeit über einer Stunde zu erwarten sind. Was ist ein geeignetes Mittel eine solche Statistik zu erzeugen, wenn jemand die Simulation mehrfach abarbeiten möchte?
2. Wenn wir Programm PRO11 so verändern, dass die Kunden alle 10 Minuten ankommen, so sollten wir einen Überlauf in der Warteschlangentabelle bekommen. Was muss zweckmäßigerweise verändert werden, um diesen Überlauf zu vermeiden?

Übung 7

Wir untersuchen einen weiteren Friseurladen. In diesem treffen durchschnittlich 3 Kunden pro Stunde ein. Die Zwischenankunftszeit schwankt zwischen 10 und 30 Minuten. Der Friseur,

der allein in seinem Laden arbeitet, benötigt durchschnittlich 20 Minuten um einen Kunden zu bedienen. Allerdings benötigt er niemals weniger als 15 Minuten und niemals mehr als 25 Minuten. Der Laden wird nach 10 Stunden geschlossen. Wir starten das Programm und erzeugen ein Histogramm, welches zeigt, wie lange die Kunden warten mussten. Die Klassen sind so zu wählen das z.B. angezeigt wird, wie viele Kunden zwischen 10 und 20 Minuten, 20 und 30 Minuten usw. warten mussten.

(12 Kunden warten 10-20 Minuten, 7 Kunden 20-30 Minuten und 4 Kunden 30-40 Minuten).