Homework 1

DATA604 Simulation and Modeling

Daniel Dittenhafer

February 2, 2016

1.1

Name several entities, attributes, activities, events, and state variables for the following systems.

(a) A cafeteria

Entities

- Serving Line
- Food Servers
- Tables

Attributes

- Number of Food Servers
- Number of seats per table
- Rate of serving for Food Servers
- Time range for eating the meal

Activities

- Waiting in line
- Being served by a Food Server
- Waiting for a table to eat
- Eating at a table

Events

- Arrival of new person in serving line to be served
- Person leaving serving line
- Person waiting for seat at table to eat
- Person finishing eating and leaving table

State Variables

- Number of people eating at tables
- Number of people waiting in line to be served

(b) A grocery store

Entities

- Checkout lanes

Attributes

- Max number of items allowed in checkout lane
- Rate of checkout for cashier

Activities

- Customer shopping in the grocery store
- Customer checking out (paying for goods)

Events

- Arrival of customer at grocery store
- Arrival of customer at checkout lane
- Customer completing checkout
- Customer departing store without purchasing anything

State Variables

- Number of customers in grocery store
- Number of customers in checkout lane lines

(c) A laundromat

Entities

- Washing machines
- Drying machines

Attributes

- Washing machine run time
- Drying machine run time
- Ratio of washing machine to drying machine capacity

Activities

- Washing clothes
- Drying clothes
- Loading washing machine
- Transfering from washing to drying machine
- Unloading from drying machine

Events

- Washing maching cycle starts
- Washing machine cycle stops
- Dryer cycle starts
- Dryer cycle stops

State Variables

- Number of busy washing machines
- Number of busy dryers

(d) A fast-food restaurant

Entities

- Cashiers
- Back-cooks (i.e. burger flippers)
- Fryers

Attributes

- Burgers per burger flipper
- Orders of frys per Fryer
- Cashier busy or not

Activities

- Cooking a burger
- Making french frys
- Cashier taking order, accepting payment

Events

- Order in
- Order ready for pickup
- French fries done cooking

State Variables

- Number of orders pending
- Number of burgers being cooked
- Orders of french frys cooked/ready for serving
- Number of bugers being ready for serving

(e) A hospital emergency room

Entities

- Doctors
- Beds
- Patients
- Admitting staff

Attributes

- Patients per Doctor

Activities

- Patient admitted
- Doctor take care of patient
- Patient discharged

Events

- Patient arrives
- Patient admitted
- Doctor discharges patient

State Variables

- Beds empty
- Patients awaiting addmission
- Paitents awaiting discharge

(f) A taxicab company with 10 taxis

Entities

- Taxis
- Dispatcher
- Customers

Attributes

- Taxi has customer
- Taxi enroute to customer
- Customer waiting for taxi

Activities

- Enroute to customer
- Transporting customer

Events

- Picking up custommer
- Dropping off customer

State Variables

- Taxis with customers
- Customers waiting for available taxi

(g) An automobile assembly line

Entities

- Parts
- Assembly machines
- Workers

Attributes

- Parts inventory
- Assembly machine rate of production
- Worker rate of production

Activities

- Machine assembling car
- Worker assembling car
- Staging parts for use by Machine or Worker

Events

- Car assembly started
- Car assembly completed
- Parts depleted
- Car assembly by machine X completed
- Car assembly by worker Y completed

State Variables

- Cars on assembly line
- Parts inventory level
- Workers out sick/vacation
- Machines broken down

2.1

Consider the following continuously operating job shop. Interarrival times of jobs are distributed as follows:

Time Between Arrivals (hours)	Probability
0	0.23
1	0.37
2	0.28
3	0.12

Processing times for jobs are normally distributed, with mean 50 minutes, and standard deviation 8 minutes. Construct a simulation table and perform a simulation for 10 new customers. Assume that, when the simulation starts, there is one job being processed (scheduled to be completed in 25 minutes) and there is one job with a 50-minute processing time in the queue.

```r
# Create a data frame of the pre-existing jobs
existingJobs <- data.frame(customer=c(-2, -1),
                           iaHrs=c(0,0),
                           iaMins=c(0,0),
                           arrivalMins=c(0,0),
                           svcTimeMins=c(25, 50),
                           timeSvcBegin=c(0, 25),
                           queueWaitMins=c(0,25),
                           timeSvcEnd=c(25, 75),
                           timeInSystem=c(25,75))
# Create a data frame of the new customers and their jobs
newJobs <- data.frame(customer=seq(1, 10),
                      iaHrs=c(0, sample(seq(0, 3),
                                        size=9,
                                        prob=c(.23, .37, .28, .12),
                                        replace=TRUE)),
                      iaMins=rep(NA, 10),
                      arrivalMins=rep(0, 10),
                      svcTimeMins=rnorm(10, mean=50, sd=8),
                      timeSvcBegin=rep(0, 10),
                      queueWaitMins=rep(0, 10),
                      timeSvcEnd=rep(0, 10),
                      timeInSystem=rep(0, 10))
# Convert from interarrival hours to minutes and
# determine overall arrival times
newJobs$iaMins <- newJobs$iaHrs * 60
newJobs$arrivalMins <- cumsum(newJobs$iaMins)
# Join the existing and new jobs into one table
simTable <- rbind(existingJobs, newJobs)
# Loop over the rows the compute the various activity and clock times
for(i in seq(3, nrow(simTable)))
{
  simTable[i,]$timeSvcBegin <- max(simTable[i,]$arrivalMins, simTable[i-1,]$timeSvcEnd)
  simTable[i,]$queueWaitMins <- simTable[i,]$timeSvcBegin - simTable[i,]$arrivalMins
```

```
  simTable[i,]$timeSvcEnd <- simTable[i,]$timeSvcBegin + simTable[i,]$svcTimeMins
  simTable[i,]$timeInSystem <- simTable[i,]$timeSvcEnd - simTable[i,]$arrivalMins
}
# Show the table
kable(simTable)
```

customer	iaHrs	iaMins	arrivalMins	svcTimeMins	timeSvcBegin	queueWaitMins	timeSvcEnd	timeInSystem
-2	0	0	0	25.00000	0.0000	0.00000	25.0000	25.00000
-1	0	0	0	50.00000	25.0000	25.00000	75.0000	75.00000
1	0	0	0	49.88872	75.0000	75.00000	124.8887	124.88872
2	3	180	180	44.90948	180.0000	0.00000	224.9095	44.90948
3	0	0	180	57.70883	224.9095	44.90948	282.6183	102.61831
4	1	60	240	49.44490	282.6183	42.61831	332.0632	92.06321
5	1	60	300	55.89174	332.0632	32.06321	387.9550	87.95496
6	2	120	420	46.70007	420.0000	0.00000	466.7001	46.70007
7	1	60	480	44.79027	480.0000	0.00000	524.7903	44.79027
8	1	60	540	64.67686	540.0000	0.00000	604.6769	64.67686
9	0	0	540	35.05879	604.6769	64.67686	639.7356	99.73565
10	2	120	660	50.96674	660.0000	0.00000	710.9667	50.96674

(a) What was the average time in the queue for the 10 new jobs? The average time in the queue for the 10 new jobs is computed below:

```
mean(simTable[seq(3, 12),]$queueWaitMins)
```

```
## [1] 25.92679
```

(b) What was the average processing time of the 10 new jobs? The average processing time is computed below:

```
mean(newJobs$svcTimeMins)
```

```
## [1] 50.00364
```

(c) What was the maximum time in the system for the 10 new jobs? The maximum time in the system for the 10 new jobs is computed below:

```
max(simTable[seq(3, 12),]$timeInSystem)
```

```
## [1] 124.8887
```

2.2

A baker is trying to figure out how many dozens of bagels to bake each day. The probability distribution of the number of bagel customers is as follows:

Customer/Day	8	10	12	14
Probability	0.35	0.30	0.25	0.10

Customers order 1,2,3 or 4 dozen bagels according to the following probability distribution:

Dozen Ordered/Customer	1	2	3	4
Probability	0.4	0.3	0.2	0.1

Bagels sell for \$8.40 per dozen. They cost \$5.80 per dozen to make. All bagels not sold at the end of the day are sold at half price to a local grocery store. Based on 5 days of simulation, how many dozen (to the nearest 5 dozen) bagels should be baked each day?

```
# Function to define a simulation at a specified
# level of dozens of bagels produced.
bakersProfit <- function(bagelsMade)
{
  simDays <- 5
  revPerDoz <- 8.40
  costPerDoz <- 5.80
  simTable <- data.frame(day=seq(1, simDays),
                         customers=sample(c(8,10,12,14),
                                          size=simDays,
                                          prob=c(0.35, 0.30, 0.25, 0.10),
                                          replace=TRUE),
                         dozenOrdered=rep(NA, simDays),
                         revenue=rep(NA, simDays),
                         lostProfit=rep(NA, simDays),
                         salvage=rep(NA, simDays),
                         dailyCost=rep(NA, simDays),
                         dailyProfit=rep(NA, simDays))

  for(i in seq(1, nrow(simTable)))
  {
    bagelsOrdered <- sample(c(1,2,3,4),
                            size=simTable[i,]$customers,
                            prob=c(0.4, 0.3, 0.2, 0.1),
                            replace=TRUE)

    simTable[i,]$dozenOrdered <- sum(bagelsOrdered)

    simTable[i,]$revenue <- min(simTable[i,]$dozenOrdered, bagelsMade) * revPerDoz
    simTable[i,]$lostProfit <- max(simTable[i,]$dozenOrdered - bagelsMade, 0) * (revPerDoz - costPerDoz)
    simTable[i,]$salvage <- max(bagelsMade - simTable[i,]$dozenOrdered, 0) * (revPerDoz / 2)
```

```
    simTable[i,]$dailyCost <- bagelsMade * costPerDoz
    simTable[i,]$dailyProfit <- simTable[i,]$revenue + simTable[i,]$salvage - simTable[i,]$dailyCost
  }

  return(simTable)
}
# Loop over a range of dozens of bagels (0, 5, 10, etc)
dozens <- seq(0, 30, by=5)
profitTable <- data.frame(dozPerDay=c(), fiveDayProfit=c())
for(d in dozens)
{
  # Run the simulation for the given level of production
  simTable <- bakersProfit(d)
  profitTable <- rbind(profitTable, cbind(dozPerDay=d, fiveDayProfit=sum(simTable$dailyProfit)))
  #print(paste(d, " dozen/day: 5 day profit is ", profitTable[profitTable$dozPerDay==d,]$fiveDayProfit, ".", sep=""))
}
```

The following table shows the profit associated with various levels of production:

dozPerDay	fiveDayProfit
0	0.0
5	65.0
10	130.0
15	186.6
20	251.6
25	199.0
30	222.0

The following table shows the details of the simulation for the maximum profit shown above (20 dozen bagels/day):

day	customers	dozenOrdered	revenue	lostProfit	salvage	dailyCost	dailyProfit
1	10	19	159.6	0.0	4.2	116	47.8
2	12	28	168.0	20.8	0.0	116	52.0
3	10	15	126.0	0.0	21.0	116	31.0
4	12	26	168.0	15.6	0.0	116	52.0
5	8	19	159.6	0.0	4.2	116	47.8

2.4

Smalltown Taxi operates one vehicle during the 9:00 A.M. to 5:00 P.M. period. Currently, consideration is being given to the addition of a second vehicle to the fleet. The demand for taxis follows the distribution shown:

Time Between Calls (minutes)	15	20	25	30	35
Probability	0.14	0.22	0.43	0.17	0.04

The distribution of time to complete a service is as follows:

Service Time (minutes)	5	15	25	35	45
Probability	0.12	0.35	0.43	0.06	0.04

Simulate 5 individual days of operation of the current system and of the system with an additional taxicab. Compare the two systems with respect to the waiting times of the customers and any other measures that might shed light on the situation.

One Taxi Simulation

```
# Function to wrap a single taxi cab on a single day
singleTaxiDailyCalls <- function(callsPerDay, maxDailyMinutes)
{
  # Create a data frame of the new customers and their jobs
  simTable <- data.frame(customer=seq(1, callsPerDay),
                         iaMins=sample(seq(15, 35, by=5),
                                       size=callsPerDay,
                                       prob=c(0.14, 0.22, 0.43, 0.17, 0.04),
                                       replace=TRUE),
                         arrivalMins=rep(0, callsPerDay),
                         svcTimeMins=sample(seq(5, 45, by=10),
                                       size=callsPerDay,
                                       prob=c(0.12, 0.35, 0.43, 0.06, 0.04),
                                       replace=TRUE),
                         timeSvcBegin=rep(0, callsPerDay),
                         queueWaitMins=rep(0, callsPerDay),
                         timeSvcEnd=rep(0, callsPerDay),
                         timeInSystem=rep(0, callsPerDay))

  # Determine overall arrival times
  simTable$arrivalMins <- cumsum(simTable$iaMins)
  # Loop over the rows the compute the various activity and clock times
  for(i in seq(1, nrow(simTable)))
  {
    if(i == 1)
    {
      simTable[i,]$timeSvcBegin <- simTable[i,]$arrivalMins
```

```
    }
    else
    {
      simTable[i,]$timeSvcBegin <- max(simTable[i,]$arrivalMins, simTable[i-1,]$timeSvcEnd)
    }
    simTable[i,]$queueWaitMins <- simTable[i,]$timeSvcBegin - simTable[i,]$arrivalMins
    simTable[i,]$timeSvcEnd <- simTable[i,]$timeSvcBegin + simTable[i,]$svcTimeMins
    simTable[i,]$timeInSystem <- simTable[i,]$timeSvcEnd - simTable[i,]$arrivalMins
  }
  # Convert queue wait of zero to NA so we can aggregate only those who actually waited.
  simTable$queueWaitMins[simTable$queueWaitMins == 0] <- NA
  # subset to the max daily minutes in the business day
  simTable <- simTable[simTable$arrivalMins <= maxDailyMinutes,]

  return(simTable)
}
```

The following code segment executes the **single** taxi simulation over 5 days and aggregates the results.

```
# Run the single taxi simulation over 5 days
daysToSimulate <- 5
multiDaySimTable <- data.frame()
for(i in seq(1, daysToSimulate))
{
  oneDayOneTaxi <- singleTaxiDailyCalls(32, 480)
  multiDaySimTableOneTaxi <- rbind(multiDaySimTable, cbind(day=i, oneDayOneTaxi))
}
kable(multiDaySimTableOneTaxi)
```

day	customer	iaMins	arrivalMins	svcTimeMins	timeSvcBegin	queueWaitMins	timeSvcEnd	timeInSystem
5	1	30	30	15	30	NA	45	15
5	2	25	55	25	55	NA	80	25
5	3	30	85	5	85	NA	90	5
5	4	30	115	25	115	NA	140	25
5	5	25	140	15	140	NA	155	15
5	6	25	165	5	165	NA	170	5
5	7	25	190	15	190	NA	205	15
5	8	25	215	15	215	NA	230	15
5	9	25	240	5	240	NA	245	5
5	10	20	260	25	260	NA	285	25
5	11	25	285	15	285	NA	300	15
5	12	25	310	25	310	NA	335	25
5	13	35	345	35	345	NA	380	35

day	customer	iaMins	arrivalMins	svcTimeMins	timeSvcBegin	queueWaitMins	timeSvcEnd	timeInSystem
5	14	25	370	15	380	10	395	25
5	15	30	400	25	400	NA	425	25
5	16	25	425	35	425	NA	460	35
5	17	30	455	25	460	5	485	30
5	18	25	480	5	485	5	490	10

```
# Show the table
kable(summary(multiDaySimTableOneTaxi[,c(2,7,9)]))
```

customer	queueWaitMins	timeInSystem
Min. : 1.00	Min. : 5.000	Min. : 5.00
1st Qu.: 5.25	1st Qu.: 5.000	1st Qu.:15.00
Median : 9.50	Median : 5.000	Median :20.00
Mean : 9.50	Mean : 6.667	Mean :19.44
3rd Qu.:13.75	3rd Qu.: 7.500	3rd Qu.:25.00
Max. :18.00	Max. :10.000	Max. :35.00
NA	NA's :15	NA

2 Taxi Simulation

Now lets develop the 2 taxi simulation.

```
# Function to wrap a duel taxi cab company on a single day
duelTaxiDailyCalls <- function(callsPerDay, maxDailyMinutes)
{
  # Create a data frame of the new customers and their jobs
  simTable <- data.frame(customer=seq(1, callsPerDay),
                         iaMins=sample(seq(15, 35, by=5),
                                       size=callsPerDay,
                                       prob=c(0.14, 0.22, 0.43, 0.17, 0.04),
                                       replace=TRUE),
                         arrivalMins=rep(0, callsPerDay),
                         tc1Avail=rep(0, callsPerDay),
                         tc2Avail=rep(0, callsPerDay),
                         taxiChosen=rep(0, callsPerDay),
                         svcTimeMins=sample(seq(5, 45, by=10),
                                       size=callsPerDay,
                                       prob=c(0.12, 0.35, 0.43, 0.06, 0.04),
                                       replace=TRUE),
                         timeSvcBegin=rep(0, callsPerDay),
```

```
                         timeSvcEndTc1=rep(0, callsPerDay),
                         timeSvcEndTc2=rep(0, callsPerDay),
                         queueWaitMins=rep(0, callsPerDay),
                         timeInSystem=rep(0, callsPerDay))

  # Determine overall arrival times
  simTable$arrivalMins <- cumsum(simTable$iaMins)
  # Loop over the rows the compute the various activity and clock times
  for(i in seq(1, nrow(simTable)))
  {
    if(i == 1)
    {
      simTable[i,]$timeSvcBegin <- simTable[i,]$arrivalMins
      simTable[i,]$taxiChosen <- 1
    }
    else
    {
      # Determine availability
      simTable[i,]$tc1Avail <- max(simTable[seq(1, i),]$timeSvcEndTc1)
      simTable[i,]$tc2Avail <- max(simTable[seq(1, i),]$timeSvcEndTc2)
      # Select taxi
      simTable[i,]$taxiChosen <- if (simTable[i,]$tc1Avail <= simTable[i,]$arrivalMins ||
                                      simTable[i,]$tc1Avail <= simTable[i,]$tc2Avail) 1 else 2
      # Determine service start based on selected taxi
      simTable[i,]$timeSvcBegin <- if(simTable[i,]$taxiChosen == 1) max(simTable[i,]$arrivalMins,
                                                                            simTable[i-1,]$tc1Avail) else max(simTable[i,]$arrivalMins,
                                                                                                                  simTable[i-1,]$tc2Avail)
    }
    simTable[i,]$timeSvcEndTc1 <- if(simTable[i,]$taxiChosen == 1) simTable[i,]$timeSvcBegin + simTable[i,]$svcTimeMins else 0
    simTable[i,]$timeSvcEndTc2 <- if(simTable[i,]$taxiChosen == 2) simTable[i,]$timeSvcBegin + simTable[i,]$svcTimeMins else 0
    simTable[i,]$queueWaitMins <- simTable[i,]$timeSvcBegin - simTable[i,]$arrivalMins
    simTable[i,]$timeInSystem <- max(simTable[i,]$timeSvcEndTc1, simTable[i,]$timeSvcEndTc2) - simTable[i,]$arrivalMins
  }
  # Convert queue wait of zero to NA so we can aggregate only those who actually waited.
  simTable$queueWaitMins[simTable$queueWaitMins == 0] <- NA
  # subset to the max daily minutes in the business day
  simTable <- simTable[simTable$arrivalMins <= maxDailyMinutes,]

  return(simTable)
}
```

The following table shows the execution of a single day for the **two** taxi simulation.

customer	iaMins	arrivalMins	tc1Avail	tc2Avail	taxiChosen	svcTimeMins	timeSvcBegin	timeSvcEndTc1	timeSvcEndTc2	queueWaitMins	timeInSystem
1	25	25	0	0	1	15	25	40	0	NA	15
2	30	55	40	0	1	25	55	80	0	NA	25
3	20	75	80	0	2	5	75	0	80	NA	5
4	25	100	80	80	1	25	100	125	0	NA	25
5	25	125	125	80	1	5	125	130	0	NA	5
6	15	140	130	80	1	15	140	155	0	NA	15
7	20	160	155	80	1	25	160	185	0	NA	25
8	15	175	185	80	2	25	175	0	200	NA	25
9	25	200	185	200	1	25	200	225	0	NA	25
10	25	225	225	200	1	15	225	240	0	NA	15
11	20	245	240	200	1	15	245	260	0	NA	15
12	15	260	260	200	1	15	260	275	0	NA	15
13	25	285	275	200	1	5	285	290	0	NA	5
14	20	305	290	200	1	25	305	330	0	NA	25
15	20	325	330	200	2	45	325	0	370	NA	45
16	35	360	330	370	1	25	360	385	0	NA	25
17	20	380	385	370	2	45	380	0	425	NA	45
18	25	405	385	425	1	15	405	420	0	NA	15
19	25	430	420	425	1	25	430	455	0	NA	25
20	25	455	455	425	1	15	455	470	0	NA	15
21	15	470	470	425	1	25	470	495	0	NA	25

The following code segment executes the two taxi simulation over 5 days and aggregates the results.

```r
# Run the two taxi simulation over 5 days
daysToSimulate <- 5
multiDaySimTable <- data.frame()
for(i in seq(1, daysToSimulate))
{
  oneDayTwoTaxi <- duelTaxiDailyCalls(32, 480)
  multiDaySimTable <- rbind(multiDaySimTable, cbind(day=i, oneDayTwoTaxi))
}
kable(multiDaySimTable)
```

day	customer	iaMins	arrivalMins	tc1Avail	tc2Avail	taxiChosen	svcTimeMins	timeSvcBegin	timeSvcEndTc1	timeSvcEndTc2	queueWaitMins	timeInSys
1	1	30	30	0	0	1	5	30	35	0	NA	
1	2	15	45	35	0	1	15	45	60	0	NA	
1	3	20	65	60	0	1	15	65	80	0	NA	
1	4	25	90	80	0	1	25	90	115	0	NA	
1	5	25	115	115	0	1	5	115	120	0	NA	
1	6	25	140	120	0	1	15	140	155	0	NA	

day	customer	iaMins	arrivalMins	tc1Avail	tc2Avail	taxiChosen	svcTimeMins	timeSvcBegin	timeSvcEndTc1	timeSvcEndTc2	queueWaitMins	timeInSys
1	7	15	155	155	0	1	15	155	170	0	NA	
1	8	15	170	170	0	1	15	170	185	0	NA	
1	9	25	195	185	0	1	15	195	210	0	NA	
1	10	25	220	210	0	1	25	220	245	0	NA	
1	11	30	250	245	0	1	25	250	275	0	NA	
1	12	20	270	275	0	2	25	270	0	295	NA	
1	13	20	290	275	295	1	25	290	315	0	NA	
1	14	25	315	315	295	1	5	315	320	0	NA	
1	15	30	345	320	295	1	15	345	360	0	NA	
1	16	35	380	360	295	1	25	380	405	0	NA	
1	17	30	410	405	295	1	5	410	415	0	NA	
1	18	25	435	415	295	1	25	435	460	0	NA	
1	19	30	465	460	295	1	5	465	470	0	NA	
2	1	20	20	0	0	1	25	20	45	0	NA	
2	2	35	55	45	0	1	25	55	80	0	NA	
2	3	15	70	80	0	2	25	70	0	95	NA	
2	4	25	95	80	95	1	25	95	120	0	NA	
2	5	20	115	120	95	2	25	115	0	140	NA	
2	6	25	140	120	140	1	25	140	165	0	NA	
2	7	25	165	165	140	1	25	165	190	0	NA	
2	8	25	190	190	140	1	5	190	195	0	NA	
2	9	25	215	195	140	1	25	215	240	0	NA	
2	10	20	235	240	140	2	25	235	0	260	NA	
2	11	30	265	240	260	1	5	265	270	0	NA	
2	12	20	285	270	260	1	25	285	310	0	NA	
2	13	20	305	310	260	2	25	305	0	330	NA	
2	14	15	320	310	330	1	25	320	345	0	NA	
2	15	25	345	345	330	1	5	345	350	0	NA	
2	16	25	370	350	330	1	35	370	405	0	NA	
2	17	20	390	405	330	2	25	390	0	415	NA	
2	18	25	415	405	415	1	15	415	430	0	NA	
2	19	25	440	430	415	1	25	440	465	0	NA	
2	20	30	470	465	415	1	15	470	485	0	NA	
3	1	20	20	0	0	1	25	20	45	0	NA	
3	2	20	40	45	0	2	5	40	0	45	NA	
3	3	20	60	45	45	1	15	60	75	0	NA	
3	4	20	80	75	45	1	15	80	95	0	NA	
3	5	30	110	95	45	1	25	110	135	0	NA	
3	6	15	125	135	45	2	15	125	0	140	NA	
3	7	20	145	135	140	1	15	145	160	0	NA	
3	8	15	160	160	140	1	25	160	185	0	NA	
3	9	25	185	185	140	1	5	185	190	0	NA	
3	10	25	210	190	140	1	15	210	225	0	NA	

day	customer	iaMins	arrivalMins	tc1Avail	tc2Avail	taxiChosen	svcTimeMins	timeSvcBegin	timeSvcEndTc1	timeSvcEndTc2	queueWaitMins	timeInSys
3	11	25	235	225	140	1	25	235	260	0	NA	
3	12	30	265	260	140	1	15	265	280	0	NA	
3	13	30	295	280	140	1	25	295	320	0	NA	
3	14	20	315	320	140	2	5	315	0	320	NA	
3	15	25	340	320	320	1	25	340	365	0	NA	
3	16	35	375	365	320	1	25	375	400	0	NA	
3	17	25	400	400	320	1	25	400	425	0	NA	
3	18	25	425	425	320	1	35	425	460	0	NA	
3	19	20	445	460	320	2	25	445	0	470	NA	
3	20	20	465	460	470	1	25	465	490	0	NA	
4	1	20	20	0	0	1	15	20	35	0	NA	
4	2	25	45	35	0	1	25	45	70	0	NA	
4	3	20	65	70	0	2	5	65	0	70	NA	
4	4	30	95	70	70	1	25	95	120	0	NA	
4	5	20	115	120	70	2	5	115	0	120	NA	
4	6	25	140	120	120	1	15	140	155	0	NA	
4	7	25	165	155	120	1	15	165	180	0	NA	
4	8	25	190	180	120	1	25	190	215	0	NA	
4	9	25	215	215	120	1	15	215	230	0	NA	
4	10	20	235	230	120	1	15	235	250	0	NA	
4	11	15	250	250	120	1	25	250	275	0	NA	
4	12	20	270	275	120	2	15	270	0	285	NA	
4	13	25	295	275	285	1	25	295	320	0	NA	
4	14	25	320	320	285	1	15	320	335	0	NA	
4	15	15	335	335	285	1	25	335	360	0	NA	
4	16	25	360	360	285	1	25	360	385	0	NA	
4	17	15	375	385	285	2	25	375	0	400	NA	
4	18	30	405	385	400	1	5	405	410	0	NA	
4	19	20	425	410	400	1	15	425	440	0	NA	
4	20	25	450	440	400	1	5	450	455	0	NA	
4	21	30	480	455	400	1	25	480	505	0	NA	
5	1	30	30	0	0	1	15	30	45	0	NA	
5	2	25	55	45	0	1	5	55	60	0	NA	
5	3	30	85	60	0	1	25	85	110	0	NA	
5	4	20	105	110	0	2	15	105	0	120	NA	
5	5	30	135	110	120	1	5	135	140	0	NA	
5	6	20	155	140	120	1	15	155	170	0	NA	
5	7	25	180	170	120	1	25	180	205	0	NA	
5	8	20	200	205	120	2	15	200	0	215	NA	
5	9	20	220	205	215	1	35	220	255	0	NA	
5	10	25	245	255	215	2	15	245	0	260	NA	
5	11	20	265	255	260	1	5	265	270	0	NA	
5	12	30	295	270	260	1	35	295	330	0	NA	

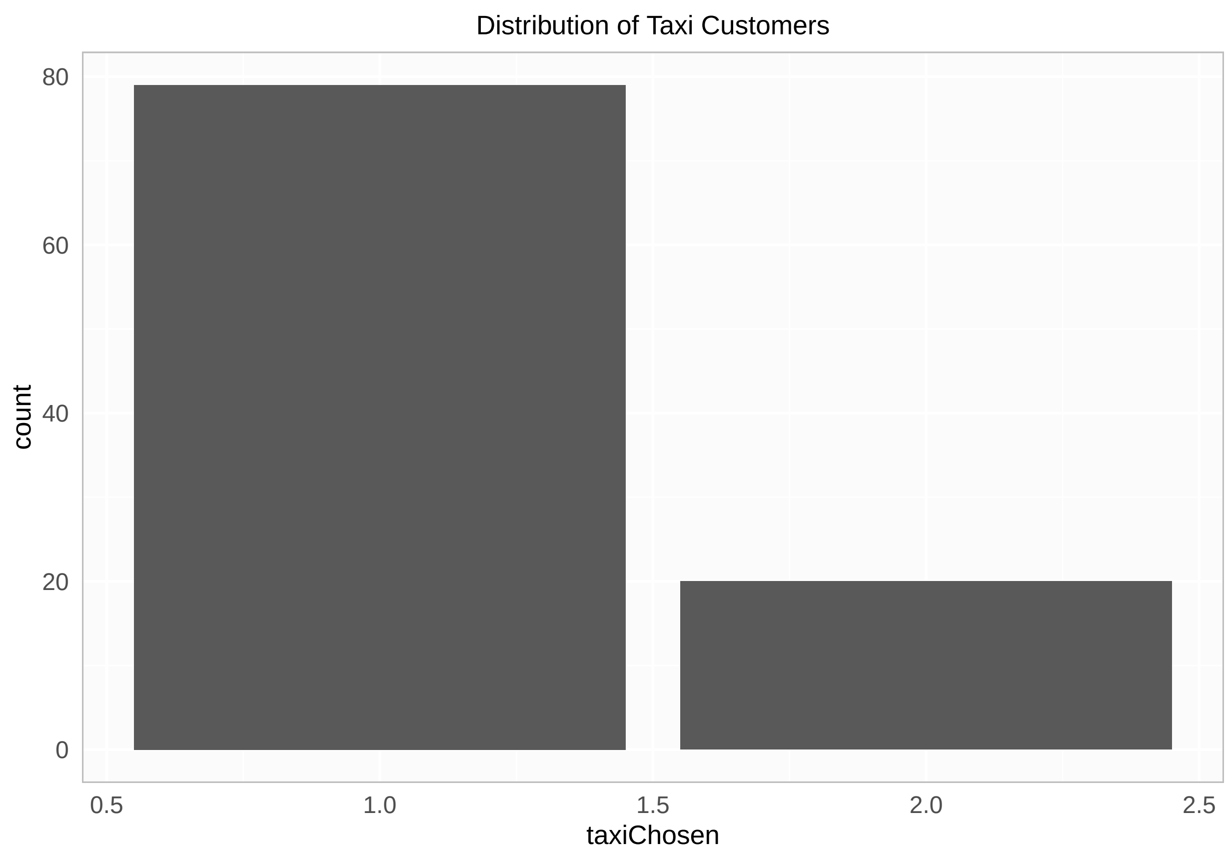
day	customer	iaMins	arrivalMins	tc1Avail	tc2Avail	taxiChosen	svcTimeMins	timeSvcBegin	timeSvcEndTc1	timeSvcEndTc2	queueWaitMins	timeInSys
5	13	15	310	330	260	2	25	310	0	335	NA	
5	14	35	345	330	335	1	45	345	390	0	NA	
5	15	25	370	390	335	2	5	370	0	375	NA	
5	16	20	390	390	375	1	15	390	405	0	NA	
5	17	20	410	405	375	1	25	410	435	0	NA	
5	18	35	445	435	375	1	45	445	490	0	NA	
5	19	30	475	490	375	2	5	475	0	480	NA	

```r
# Show the table
kable(summary(multiDaySimTable[,c(2,12,13)]))
```

customer	queueWaitMins	timeInSystem
Min. : 1.00	Min. : NA	Min. : 5.00
1st Qu.: 5.50	1st Qu.: NA	1st Qu.:15.00
Median :10.00	Median : NA	Median :25.00
Mean :10.41	Mean :NaN	Mean :18.84
3rd Qu.:15.00	3rd Qu.: NA	3rd Qu.:25.00
Max. :21.00	Max. : NA	Max. :45.00
NA	NA's :99	NA

As can be seen in the column chart below, Taxi 1 has a much larger number of customers, but Taxi 2 does pick up approximately 20% of the customers.

```r
g1 <- ggplot(multiDaySimTable) +
  geom_bar(aes(x=taxiChosen)) +
  labs(title="Distribution of Taxi Customers")  + myTheme
g1
```



2.5

The random variables X, Y, and Z are distributed as follows:

$$X \ N(\mu = 100, \sigma^2 = 100)$$

$$Y \ N(\mu = 300, \sigma^2 = 225)$$

$$Z \ N(\mu = 40, \sigma^2 = 64)$$

Simulate 50 values of the random variable

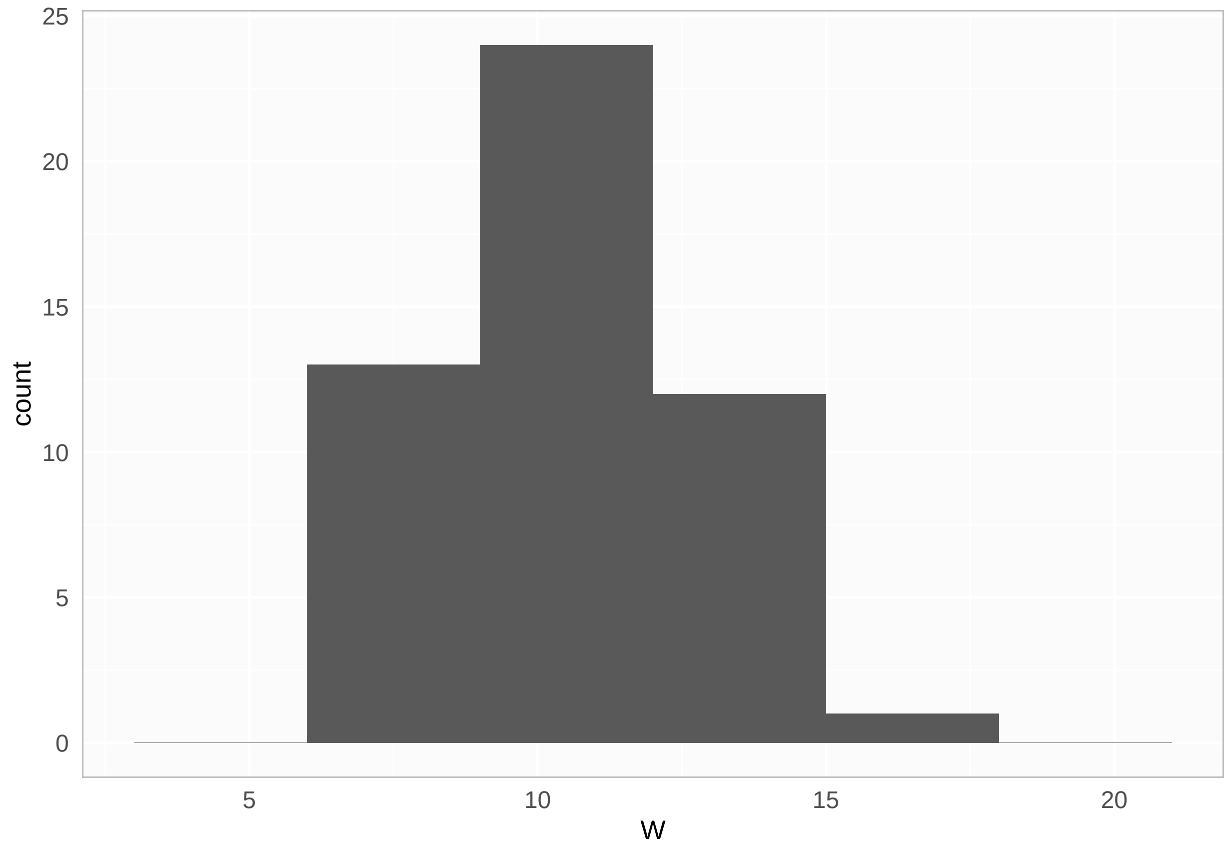
$$W = \frac{X + Y}{Z}$$

Prepare a histogram of the resulting values, using class intervals of width equal to 3.

```
fxX <- function (n)
{
  return (rnorm(n, 100, 10))
}

fxY <- function(n)
{
  return (rnorm(n, 300, 15))
}

fxZ <- function(n)
{
  return (rnorm(n, 40, 8))
}

n <- 50
dfVars <- data.frame(X=fxX(n), Y=fxY(n), Z= fxZ(n))
dfVars$W <- ( dfVars$X + dfVars$Y ) / dfVars$Z
```

The first 5 resulting random variables and the values of W are shown in the table below.

X	Y	Z	W
91.33994	309.2501	37.98667	10.545542
91.78661	300.2354	42.11673	9.307989
97.82386	272.7642	38.84069	9.541234
102.98633	305.7091	33.43615	12.223159
97.20652	294.7348	47.86196	8.188995
96.88976	292.6918	39.76631	9.796774



2.7

Estimate, by simulation, the average number of lost sales per week for an inventory system that functions as follows:

(a) Whenever the inventory level falls to or below 10 unites, an order is placed. Only one order can be outstanding at a time.

(b) The size of each order is equal to 20 - I, where I is the inventory level when the order is placed.

(c) If a demand occurs during a period when the inventory level is zero, the sale is lost.

(d) Daily demand is normally distributed, with a mean of 5 units and a standard deviation of 1.5 unites. (Round off decimals to the closest integer during the simulation and, if a negative value results, give it a demand of zero.)

(e) Lead time is distributed uniformly between zero and 5 days - integers only.

(f) The simulation will start with 18 units in inventory.

(g) For simplicity, assume that orders are placed at the close of the business day and received after the lead time has occured. Thus, if lead time is one day, the order is available for distribution on the morning of the second day of business following the placement of the order.

(h) Let the simulation run for 5 weeks.

```
inventorySim <- function(daysToSimulate, daysPerWeek)
{
  simTable <- data.frame(day=seq(0, daysToSimulate),
                         #cycle=
                         dayInCycle=c(daysPerWeek, rep(seq(1, daysPerWeek), daysToSimulate / daysPerWeek)),
                         beginInv=c(NA, rep(NA, daysToSimulate)),
                         demand=c(NA, round(rnorm(daysToSimulate, mean=5, sd=1.5))),
                         endInv=c(18, rep(NA, daysToSimulate)),
                         shortage=c(0, rep(NA, daysToSimulate)),
                         pendingOrder=c(0, rep(0, daysToSimulate)),
                         leadTime=c(NA, rep(NA, daysToSimulate)),
                         orderArriveDays=c(0, rep(0, daysToSimulate)))

  # Loop over the rows the compute the various activity and clock times
  for(i in seq(1, nrow(simTable)))
  {
    if(i == 1)
    {

    }
    else
    {
      pendingO <- 0
      if(simTable[i-1,]$orderArriveDays == 1)
      {
        pendingO <- simTable[i-1,]$pendingOrder
        simTable[i-1,]$pendingOrder <- 0
      }

      simTable[i,]$beginInv <- simTable[i-1,]$endInv + pendingO

      endI <- simTable[i,]$beginInv - simTable[i,]$demand - simTable[i-1,]$shortage
      simTable[i,]$endInv <- max(endI, 0)
      simTable[i,]$shortage <- if(endI < 0) simTable[i-1,]$shortage + simTable[i,]$demand - simTable[i,]$beginInv else 0

      # Ordering
      if(simTable[i,]$endInv <= 10 && simTable[i-1,]$pendingOrder == 0)
      {
        simTable[i,]$pendingOrder <- 20 - simTable[i,]$endInv
        simTable[i,]$leadTime <- round(runif(1, 0, 5))
```

```
        simTable[i,]$orderArriveDays <- simTable[i,]$leadTime
      }
      else
      {
        simTable[i,]$orderArriveDays <- if(simTable[i-1,]$orderArriveDays > 0) simTable[i-1,]$orderArriveDays - 1 else 0
        simTable[i,]$pendingOrder <- simTable[i-1,]$pendingOrder
      }
    }
  }

  return(simTable)
}

daysToSim <- 5 * 5
inventorySimTable <- inventorySim(daysToSim, 5)
```

day	dayInCycle	beginInv	demand	endInv	shortage	pendingOrder	leadTime	orderArriveDays
0	5	NA	NA	18	0	0	NA	0
1	1	18	3	15	0	0	NA	0
2	2	15	7	8	0	12	5	5
3	3	8	6	2	0	12	NA	4
4	4	2	5	0	3	12	NA	3
5	5	0	5	0	8	12	NA	2
6	1	0	6	0	14	0	NA	1
7	2	12	5	0	7	20	3	3
8	3	0	6	0	13	20	NA	2
9	4	0	4	0	17	0	NA	1
10	5	20	6	0	3	20	0	0
11	1	0	6	0	9	20	NA	0
12	2	0	7	0	16	20	NA	0
13	3	0	3	0	19	20	NA	0
14	4	0	2	0	21	20	NA	0
15	5	0	5	0	26	20	NA	0
16	1	0	5	0	31	20	NA	0
17	2	0	2	0	33	20	NA	0
18	3	0	3	0	36	20	NA	0
19	4	0	6	0	42	20	NA	0
20	5	0	6	0	48	20	NA	0
21	1	0	7	0	55	20	NA	0
22	2	0	3	0	58	20	NA	0
23	3	0	5	0	63	20	NA	0
24	4	0	3	0	66	20	NA	0
25	5	0	4	0	70	20	NA	0