Final test - Salla Vesterinen

Probability, Statistics and Discrete Mathematics, Spring 2019
9.5.2019
Helsinki Metropolia University of Applied Sciences

In [1]:
```
# Import necessary libraries
%pylab inline
import numpy.random as rnd
import scipy.stats as stats
import pandas as pd
from pandas import Series
from matplotlib import pyplot
from scipy.stats import norm
```

```
Populating the interactive namespace from numpy and matplotlib
```

Problem 1

Here is how I solved this problem...

In [2]:
```
#Value array
speed=np.array([199.49, 199.33, 199.23, 200.01, 200.15, 199.67, 200.25, 199.51, 200.6, 199.94])

#Average = mean
mn=speed.mean()
print("Mean:",mn)

#Variance = standard deviation
sd=speed.std()
print("Standard deviation:",sd)

#95% falls in between two standard deviations of the mean
l=mn-2*sd
print("Low:",l)
h=mn+2*sd
print("High:",h)
```

```
Mean: 199.81799999999998
Standard deviation: 0.42051872728809525
Low: 198.9769625454238
High: 200.65903745457618
```

Final answers:

- (a) 199.818
- (b) 0.421
- (c) 198.877 to 200.659

Problem 2

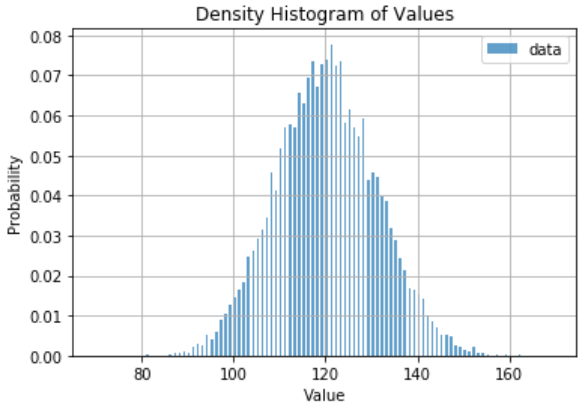
Here is how I solved this problem...

In [3]:
```
#Array of random values following Poisson distribution and with mean of 120
x = np.random.poisson(size=10000, lam=120)
#Check mean
print(x.mean())
```

```
119.9399
```

In [4]:
```
#Density histogram
hist(x, bins = arange(70, 170, 0.5), alpha = 0.7, density = True, label = "data")

xlabel('Value')
ylabel('Probability')
plt.title('Density Histogram of Values')
legend()
grid()

#Probability value<100
print("Less than 100:",(x<100).sum())

#Probability 120<value<140
print("Between 120 and 140:",(x<120).sum()+(x>140).sum())

#Quantile 25% and 75% limits
low=np.quantile(x, 0.25)
print("25% limit:",low)
high=np.quantile(x, 0.75)
print("75% limit:",high)

#Verification that 50% falls between range
print("50%:",10000*0.5)
res=(x<low).sum()+(x>high).sum()
print("Count:",res)
```

```
Less than 100: 295
Between 120 and 140: 5219
25% limit: 112.0
75% limit: 127.0
50%: 5000.0
Count: 4721
```



Final answers:

- (a) Histogram above, "Density Histogram of Values"
- (b) Between 268 and 302 (mean of 286.6), when repeated 5 times
- (c) Between 4702 and 4864 (mean of 4799.6), when repeated 5 times
- (d) 25% quantile limit: 112 or 113 (depending on run), 75% quantile limit: 127 or 128

Problem 3

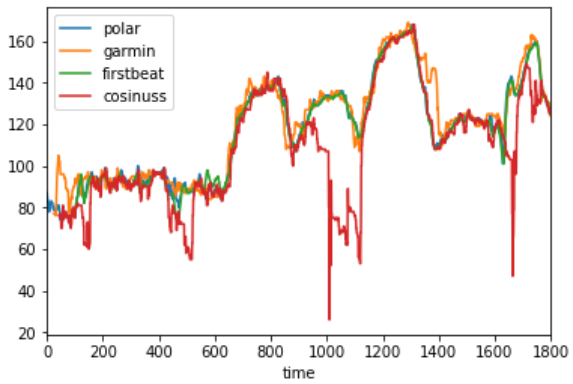
Here is how I solved this problem...

In [5]:
```
# Read the example data into Python
file=open("heartrate.csv", "r")
sep = ","
data = pd.read_csv(file, sep, index_col='time')
data.head()
```

Out[5]:

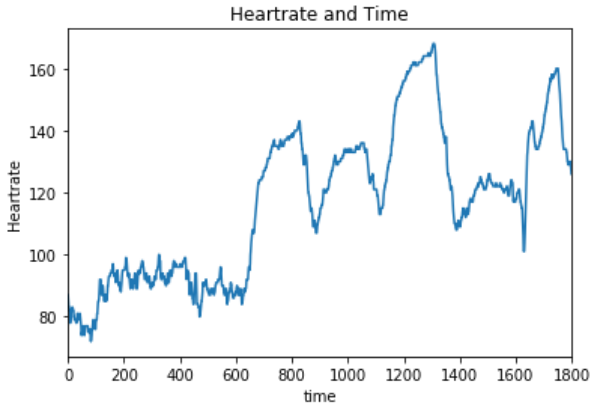
time	polar	garmin	firstbeat	cosinuss
0	87.0	NaN	NaN	NaN
1	85.0	NaN	NaN	NaN
2	84.0	NaN	NaN	NaN
3	83.0	NaN	NaN	NaN
4	83.0	NaN	NaN	NaN

In [6]:
```
#Graphical plot of heartrates against time (time on x-axis)
series = pd.read_csv('heartrate.csv', header=0, index_col='time')
series.plot()
pyplot.show()
```



In [7]:
```
#Another way of doing it
series = Series.from_csv('heartrate.csv', header=0)
ylabel('Heartrate')
plt.title('Heartrate and Time')
series.plot()
pyplot.show()
```

```
C:\Users\Salla\Anaconda3\lib\site-packages\pandas\core\series.py:4141: FutureWarning: from_csv is deprecated.
Please use read_csv(...) instead. Note that some of the default arguments are different, so please refer to t
he documentation for from_csv when changing your function calls
  infer_datetime_format=infer_datetime_format)
```



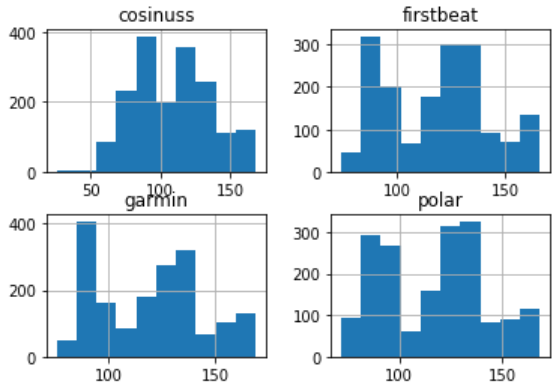
Final answers (based only on graph!):

- (a) Typical range: 80-160
- (b) Min: 75, Max: 165
- (c) Average: 120

Problem 4

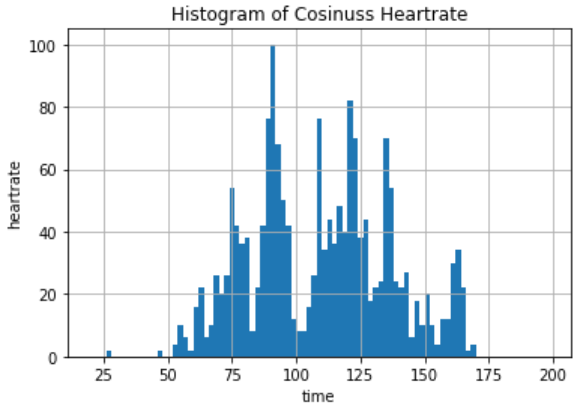
Here is how I solved this problem...

In [8]:
```
#Histogram plots of heartrates and calculate descriptive statistics of the results
#Overview of all
series = pd.read_csv('heartrate.csv', header=0, index_col='time')
series.hist()
pyplot.show()
```

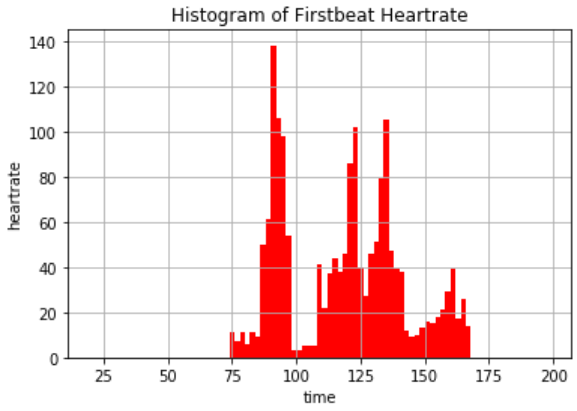


In [9]:
```
#Data
series = pd.read_csv('heartrate.csv', header=0, index_col='time')
a=data['cosinuss']
b=data['firstbeat']
c=data['garmin']
d=data['polar']
bins=np.arange(20,200,2)
```

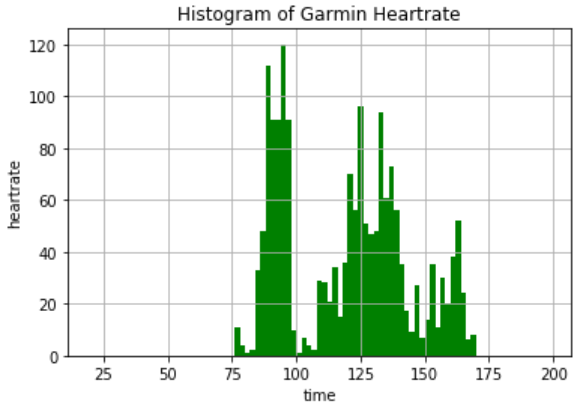
In [10]:
```
#Cosinuss
plt.hist(a, bins)
xlabel('time')
ylabel('heartrate')
plt.title('Histogram of Cosinuss Heartrate')
grid()
```



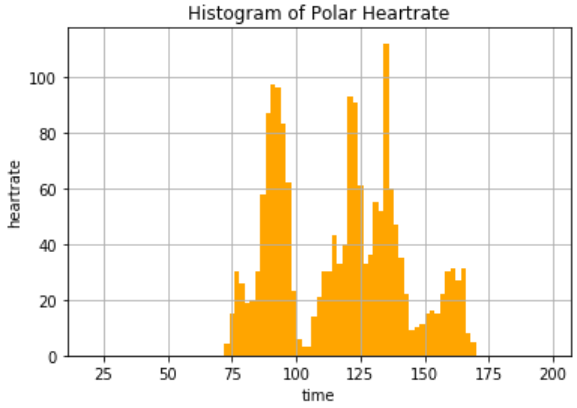
In [11]:
```
#Firstbeat
plt.hist(b, bins, color='red')
xlabel('time')
ylabel('heartrate')
plt.title('Histogram of Firstbeat Heartrate')
grid()
```



In [12]:
```
#Garmin
plt.hist(c, bins, color='green')
xlabel('time')
ylabel('heartrate')
plt.title('Histogram of Garmin Heartrate')
grid()
```



In [13]:
```
#Polar
plt.hist(d, bins, color='orange')
xlabel('time')
ylabel('heartrate')
plt.title('Histogram of Polar Heartrate')
grid()
```

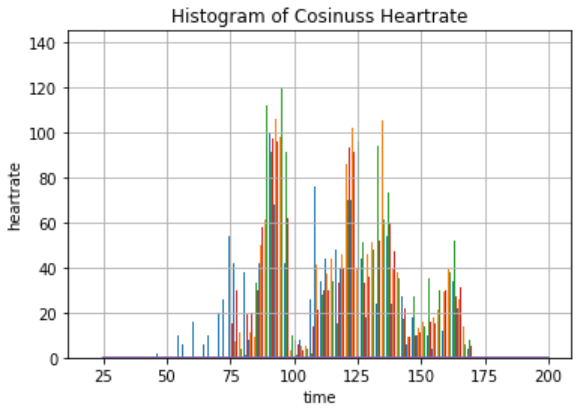


In [14]:
```
#Data
print("Cosinuss:",a.mean(), a.std())
print("Firstbeat:",b.mean(), b.std())
print("Garmin:",c.mean(), c.std())
print("Polar:",d.mean(), d.std())
```

```
Cosinuss: 109.07797381900967 27.21012387789403
Firstbeat: 118.09766081871345 23.337284370811943
Garmin: 119.2697072072072 24.159724490637693
Polar: 116.66277777777778 24.42094602676549
```

In [15]:
```
#Probability distribution that fits the histograms
plt.hist([a, b, c, d], bins)
xlabel('time')
ylabel('heartrate')
plt.title('Histogram of Cosinuss Heartrate')
grid()

# Create a normal distribution with default values loc = 1.0, and scale = 1.0
rv = norm()
# Create x-axis
x = arange(25, 200, 0.1)
# Draw the probability density function (pdf)
plot(x, rv.pdf(x))
```

Out[15]: [<matplotlib.lines.Line2D at 0x2375db4b898>]



Final answers:

- (a) Number of samples: 5
- (b) Device with highest average heartrate: Garmin
- (c) Device with largest deviation: Cosinuss
  this is because Cosinuss started when the time started (lower heartrate) unlike the others
- (d) Probability distribution that fits: From afar it looks like a normal distribution with a slope up followed by the peak and then the descent down. In reality though it goes down and up again after the initial peak therefore looking like a multimodal distribution function with three peaks.