1) sum(theta)/n

theta is a vector of n values drawn from p(theta|y)

g(theta) and p(theta) are not required in this case

In [1]:
```
using Distributions;
using Gadfly;
```

```
2

a)  p(x) =  unif(1,7) = 1/6;
    g(x) = 6 f(x)  = 6 x^2 e^-x
```

b)

1)p(x) >= 0;

2) $\int p(x)\,dx = 1$

3) g(x) is a factor of x such that g(x

c)

1) Initialize x1,.....,xn to 0's

2) for i = 1...n times

3) Draw xi ~ unif(1,7)

4)end

5) sn = 6 * sum(x1,....xn)/n

6) return sn

d)

In [2]:
```
n=10000;
delta=6;
f(x) = (x^2.7)*(e^-x);
x = rand(Uniform(1, 7), n);
S = sum(delta.*f.(x))/n
```

Out[2]:
```
3.797650977086129
```

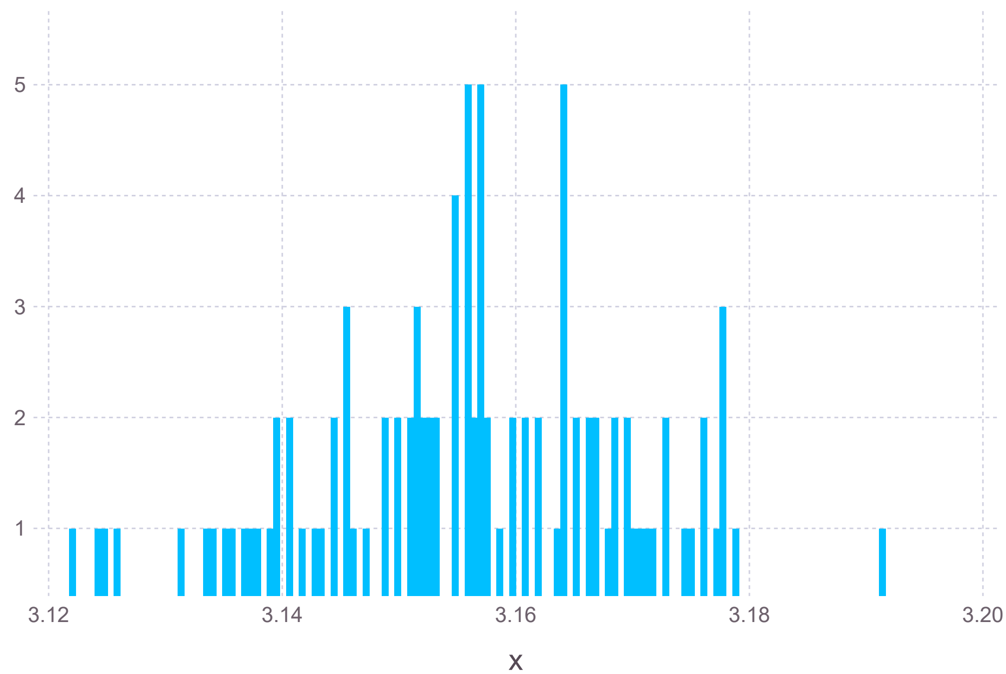
e)

In [14]:
```
n = 10000;
delta = 5;
S = zeros(100);
for i= 1:100
    x = rand(Uniform(1, 7), n);
    S[i] = sum(delta.*f.(x))/n;
end
mean(S), var(S)
```

Out[14]:
```
(3.1556872756238556, 0.00015674075078885727)
```

In [5]:
```
plot(layer(x=S, Geom.histogram))
```

Out[5]:

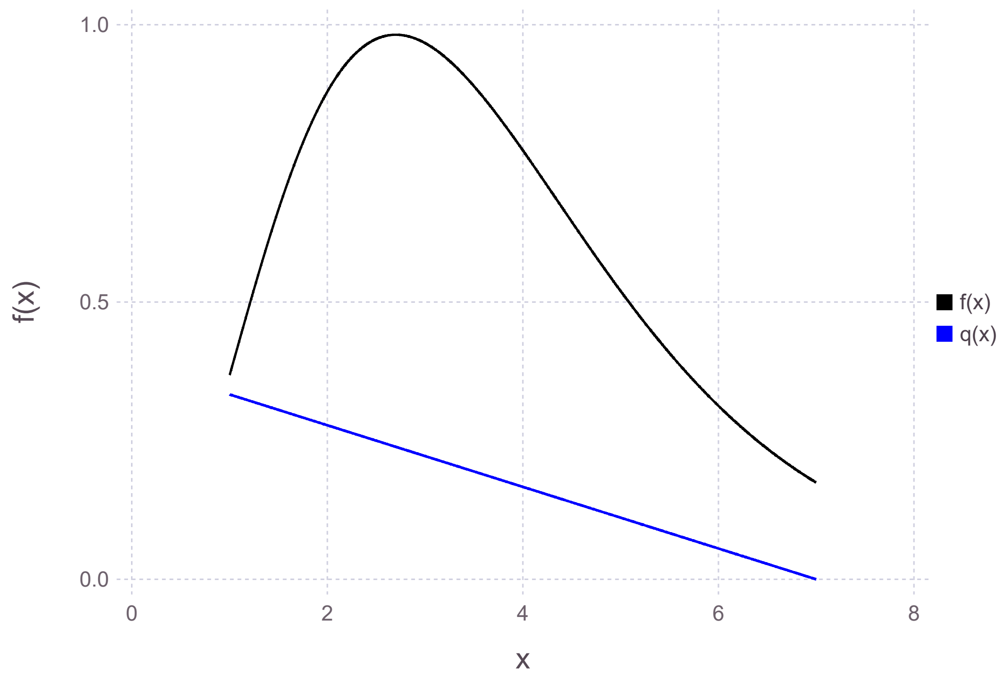


3) a)

In [6]:
```
x = collect(1:0.01:7);
f(x) = (x^2.7)*(e^-x);
q(x) = (7-x)/18;
```

In [15]:
```
plot(
layer(x=x,y=f.(x),Geom.line,Theme(default_color=colorant"black")),
layer(x=x,y=q.(x),Geom.line,Theme(default_color=colorant"blue")),
Guide.ylabel("f(x)"),Guide.xlabel("x"),Coord.Cartesian(xmin=0, xmax=8),
Guide.manual_color_key("", ["f(x)", "q(x)"], ["black","blue"]))
```

Out[15]:



b)

1) Initialize x1,.....,xn to 0's

2) for i = 1...n times

3) Draw xi ~ q(x)

4)end

5) sn = sum(f(xi)/q(xi))/n

6) return sn

f(x) = (x^2.7)*(e^-x);

q(x) = (7-x)/18;

c)

In [10]:
```
function accept_reject_method(n)
    x = 1:0.01:7;
    f(x) = (7-x)/18;
    g(x) = pdf(Uniform(1,7),x);
    M = maximum(f.(x)./g.(x));
    count = 0;
    samples = [];
    while(count<n)
        y = rand(Uniform(1,7));
        u = rand(Uniform(0,1));
        if(u<f(y)/(M*g(y)))
            samples = [samples; y];
            count +=1;
            end
        end
    return samples;
end
```

Out[10]:
```
accept_reject_method (generic function with 1 method)
```

In [11]:
```
f(x) = (x^2.7)*(e^-x);
q(x) = (7-x)/18;
n = 10000;
x = accept_reject_method(n);
S = sum(f.(x)./(q.(x)))/length(x)
```

Out[11]:
```
3.776183033867579
```

d)

In [13]:
```
n = 10000;
f(x) = (x^2.7)*(e^-x);
p(x) = (7-x)/18;
S = zeros(100);
for i= 1:100
    x = accept_reject_method(n);
    S[i] = sum(f.(x)./(p.(x)))/n;
end
mean(S), var(S)
```

Out[13]:
```
(3.7841925791854165, 0.00041929977839883889)
```

4)

Ordinary Monte Carlo Integration has less variance compared to estimate generated by importance sampling because shaoe factor is not solely resposible for reducing variance.