In [1]:

```
import numpy as np
from scipy.integrate import odeint
import matplotlib.pyplot as plt
%matplotlib inline
```

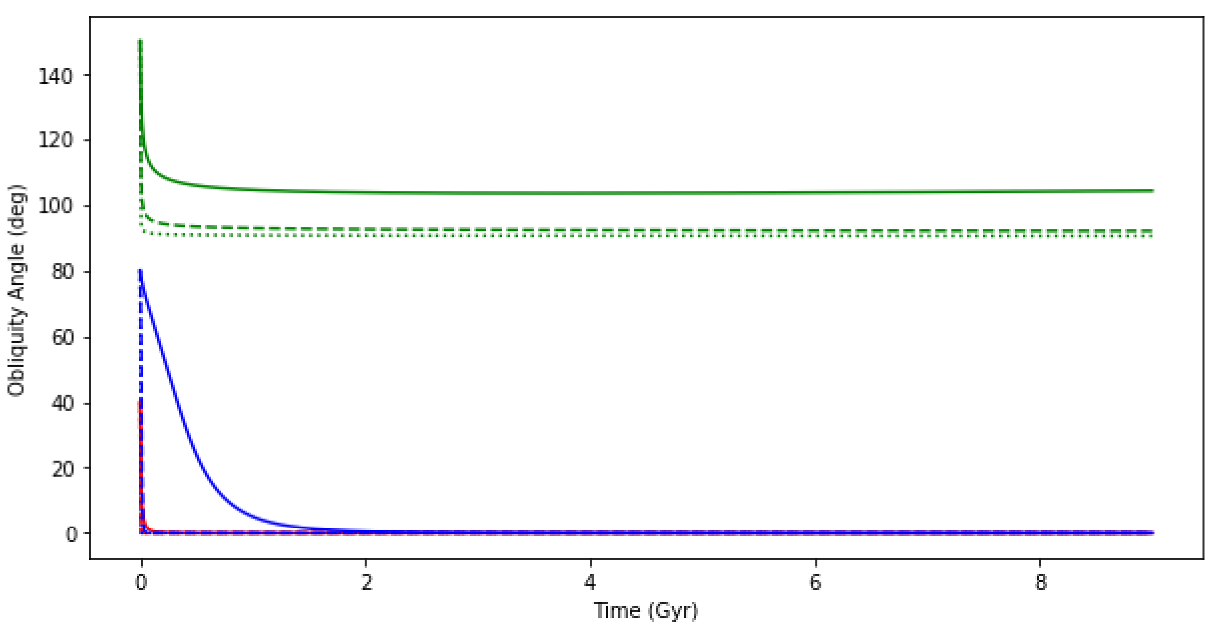
In [2]:

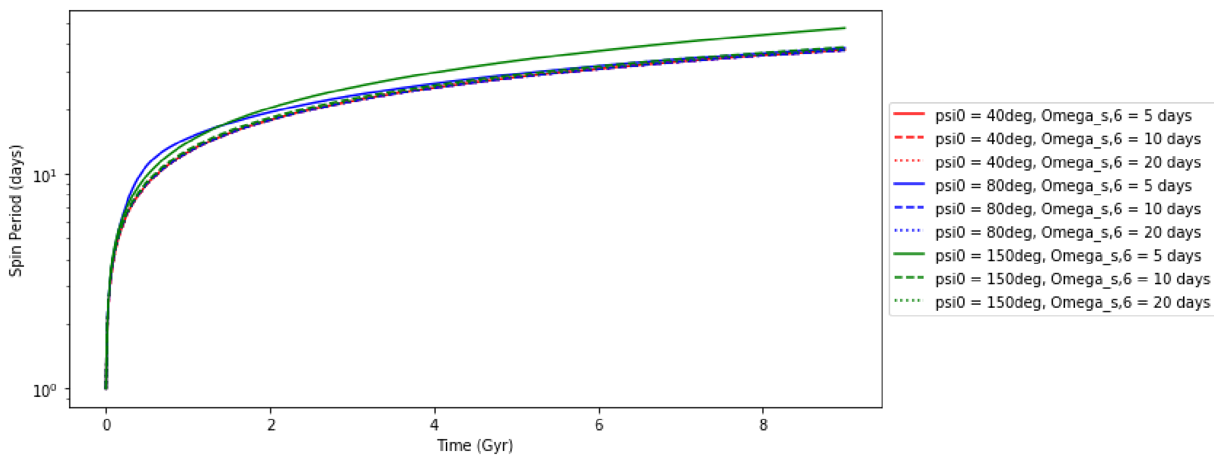
```
# Question 1

# constant parameters
G = 6.67e-11
AU = 1.5e11
alpha = 1.5e-14*3.15e7        # magnetic braking timescale, in seconds
kappa = 0.1                   # dimensionless moment of inertia constant
M_s = 2e30                    # solar mass in kg
M_p = 1e-3*2e30               # 10^-3 Jupiter masses
R_s = 7e8                     # solar radius in m
a_p = 0.015*AU                # planet semi-major axis
Omega = (G*M_s/a_p**3)**0.5   # orbital angular velocity
L_p = M_p*(G*M_s*a_p)**0.5    # planet orbital angular momentum

# inital conditions
psi0_1 = 40*(np.pi/180)
psi0_2 = 80*(np.pi/180)
psi0_3 = 150*(np.pi/180)

Period_s0 = 1*86400           # 1 day
Omega_s0 = 2*np.pi/Period_s0  # angular frequency with 1 day period
Omega_s6_1 = 2*np.pi/(5*Period_s0) # inertial wave dissipation parameter
Omega_s6_2 = 2*np.pi/(10*Period_s0)
Omega_s6_3 = 2*np.pi/(20*Period_s0)

# integration time
t = np.linspace(0,9,1000)*1e9*3.15e7  # evolve system for 9 billion years

def ObliquityTides_model(vector, t, Omega_s6):
    psi, Omega_s = vector

    S_s = kappa*M_s*R_s**2*Omega_s        # stellar angular momentum
    Q = 1e6*(Omega_s6/Omega_s)**2         # Q'_10: dimensionless, strength of t
    t_s10 = 1/((0.75/Q)*(M_p/M_s)*(R_s/a_p)**5*(L_p/S_s)*Omega)

    dpsidt = (-1/t_s10)*np.sin(psi)*np.cos(psi)**2*(np.cos(psi)+S_s/L_p)
    dOmega_sdt =(-1/t_s10)*np.sin(psi)**2*np.cos(psi)**2*Omega_s-alpha*Omega_s*

    return [dpsidt, dOmega_sdt]
```

In [3]:
```
# Solve ODEs for various free parameters

sol = odeint(ObliquityTides_model, [psi0_1, Omega_s0], t, args=(Omega_s6_1,))
psi11, Omega_s11 = sol[:,0]*(180/np.pi), sol[:,1]
Period11 = 2*np.pi/Omega_s11/86400 # convert back to days

sol = odeint(ObliquityTides_model, [psi0_1, Omega_s0], t, args=(Omega_s6_2,))
psi12, Omega_s12 = sol[:,0]*(180/np.pi), sol[:,1]
Period12 = 2*np.pi/Omega_s12/86400 # convert back to days

sol = odeint(ObliquityTides_model, [psi0_1, Omega_s0], t, args=(Omega_s6_3,))
psi13, Omega_s13 = sol[:,0]*(180/np.pi), sol[:,1]
Period13 = 2*np.pi/Omega_s13/86400 # convert back to days

sol = odeint(ObliquityTides_model, [psi0_2, Omega_s0], t, args=(Omega_s6_1,))
psi21, Omega_s21 = sol[:,0]*(180/np.pi), sol[:,1]
Period21 = 2*np.pi/Omega_s21/86400 # convert back to days

sol = odeint(ObliquityTides_model, [psi0_2, Omega_s0], t, args=(Omega_s6_2,))
psi22, Omega_s22 = sol[:,0]*(180/np.pi), sol[:,1]
Period22 = 2*np.pi/Omega_s22/86400 # convert back to days

sol = odeint(ObliquityTides_model, [psi0_2, Omega_s0], t, args=(Omega_s6_3,))
psi23, Omega_s23 = sol[:,0]*(180/np.pi), sol[:,1]
Period23 = 2*np.pi/Omega_s23/86400 # convert back to days

sol = odeint(ObliquityTides_model, [psi0_3, Omega_s0], t, args=(Omega_s6_1,))
psi31, Omega_s31 = sol[:,0]*(180/np.pi), sol[:,1]
Period31 = 2*np.pi/Omega_s31/86400 # convert back to days

sol = odeint(ObliquityTides_model, [psi0_3, Omega_s0], t, args=(Omega_s6_2,))
psi32, Omega_s32 = sol[:,0]*(180/np.pi), sol[:,1]
Period32 = 2*np.pi/Omega_s32/86400 # convert back to days

sol = odeint(ObliquityTides_model, [psi0_3, Omega_s0], t, args=(Omega_s6_3,))
psi33, Omega_s33 = sol[:,0]*(180/np.pi), sol[:,1]
Period33 = 2*np.pi/Omega_s33/86400 # convert back to days
```

In [4]:

```
# Obliquity and spin period evolution plots
plt.figure(figsize=(10,5))
plt.plot(t/(1e9*3.15e7), psi11, c='red')
plt.plot(t/(1e9*3.15e7), psi12, c='red',ls='--')
plt.plot(t/(1e9*3.15e7), psi13, c='red',ls=':')
plt.plot(t/(1e9*3.15e7), psi21, c='blue')
plt.plot(t/(1e9*3.15e7), psi22, c='blue',ls='--')
plt.plot(t/(1e9*3.15e7), psi23, c='blue',ls=':')
plt.plot(t/(1e9*3.15e7), psi31, c='green')
plt.plot(t/(1e9*3.15e7), psi32, c='green',ls='--')
plt.plot(t/(1e9*3.15e7), psi33, c='green',ls=':')
plt.xlabel('Time (Gyr)')
plt.ylabel('Obliquity Angle (deg)')
plt.savefig('q1a.png', dpi=400)
plt.show()

plt.figure(figsize=(10,5))
plt.semilogy(t/(1e9*3.15e7), Period11, c='red',label='psi0 = 40deg, Omega_s,6 =
plt.semilogy(t/(1e9*3.15e7), Period12, c='red',ls='--',label='psi0 = 40deg, Ome
plt.semilogy(t/(1e9*3.15e7), Period13, c='red',ls=':',label='psi0 = 40deg, Omeg
plt.semilogy(t/(1e9*3.15e7), Period21, c='blue',label='psi0 = 80deg, Omega_s,6
plt.semilogy(t/(1e9*3.15e7), Period22, c='blue',ls='--',label='psi0 = 80deg, Om
plt.semilogy(t/(1e9*3.15e7), Period23, c='blue',ls=':',label='psi0 = 80deg, Ome
plt.semilogy(t/(1e9*3.15e7), Period31, c='green',label='psi0 = 150deg, Omega_s,
plt.semilogy(t/(1e9*3.15e7), Period32, c='green',ls='--',label='psi0 = 150deg,
plt.semilogy(t/(1e9*3.15e7), Period33, c='green',ls=':',label='psi0 = 150deg, O
#plt.ylim(1e0, 1e1)
plt.xlabel('Time (Gyr)')
plt.ylabel('Spin Period (days)')
plt.legend(loc='center left', bbox_to_anchor=(1, 0.5))
plt.savefig('q1b.png', dpi=400)
plt.show()
```

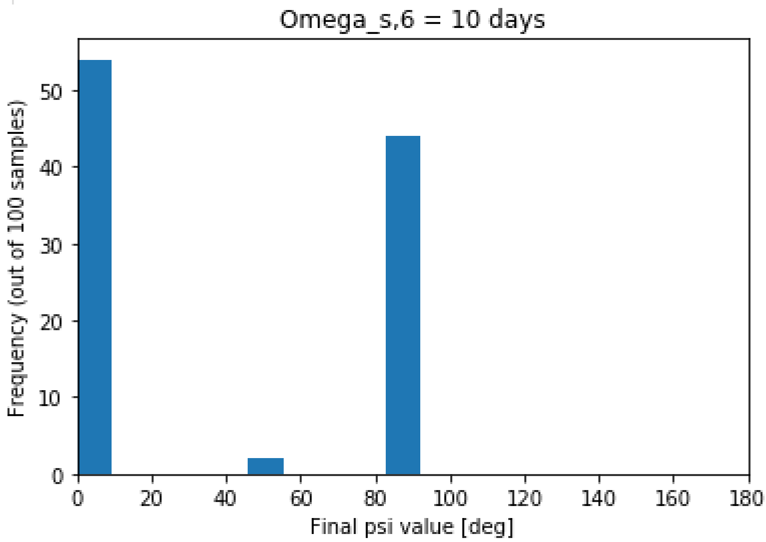


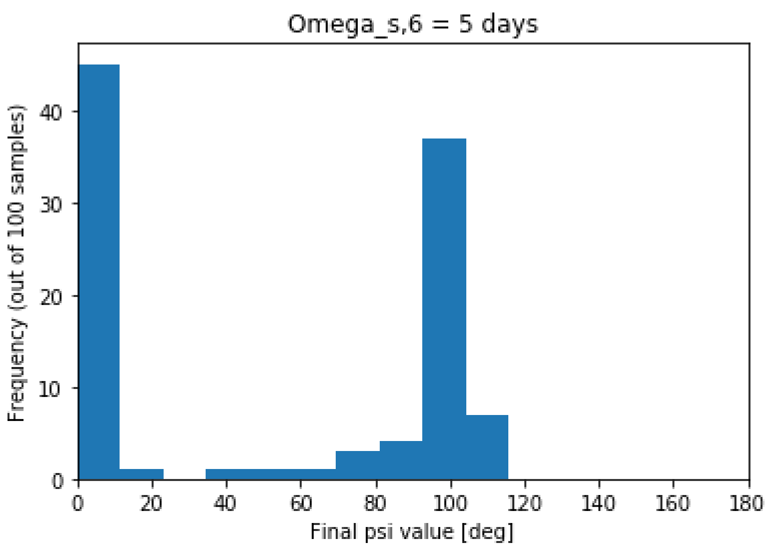


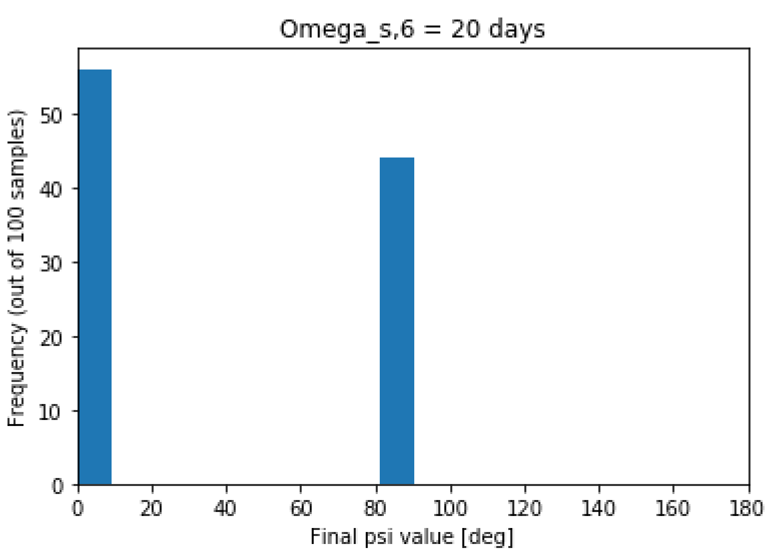
Spin Period (days)
Time (Gyr)
10^0
10^1
0
2
4
6
8
psi0 = 40deg, Omega_s,6 = 5 days
psi0 = 40deg, Omega_s,6 = 10 days
psi0 = 40deg, Omega_s,6 = 20 days
psi0 = 80deg, Omega_s,6 = 5 days
psi0 = 80deg, Omega_s,6 = 10 days
psi0 = 80deg, Omega_s,6 = 20 days
psi0 = 150deg, Omega_s,6 = 5 days
psi0 = 150deg, Omega_s,6 = 10 days
psi0 = 150deg, Omega_s,6 = 20 days

In [5]:
```
# Question 2

y = np.random.uniform(0,1,100)
psi_samples = np.arccos(1-2*y)

# 2a

Omega_s6 = 2*np.pi/(10*Period_s0) # inertial wave dissipation parameter
psi_final = []

for psi0 in psi_samples:
    sol = odeint(ObliquityTides_model, [psi0, Omega_s0], t, args=(Omega_s6,))
    psi_sol, ignored = sol[:,0]*(180/np.pi), sol[:,1]
    psi_final.append(psi_sol[-1])

count, bins, ignored = plt.hist(psi_final, 10)
plt.title('Omega_s,6 = 10 days')
plt.xlabel('Final psi value [deg]')
plt.ylabel('Frequency (out of 100 samples)')
plt.xlim(0,180)
plt.savefig('q2a.png', dpi=400)
plt.show()

# 2b

Omega_s6 = 2*np.pi/(5*Period_s0) # inertial wave dissipation parameter
psi_final = []

for psi0 in psi_samples:
    sol = odeint(ObliquityTides_model, [psi0, Omega_s0], t, args=(Omega_s6,))
    psi_sol, ignored = sol[:,0]*(180/np.pi), sol[:,1]
    psi_final.append(psi_sol[-1])

count, bins, ignored = plt.hist(psi_final, 10)
plt.title('Omega_s,6 = 5 days')
plt.xlabel('Final psi value [deg]')
plt.ylabel('Frequency (out of 100 samples)')
plt.xlim(0,180)
plt.savefig('q2b.png', dpi=400)
plt.show()

# 2c

Omega_s6 = 2*np.pi/(20*Period_s0) # inertial wave dissipation parameter
psi_final = []

for psi0 in psi_samples:
    sol = odeint(ObliquityTides_model, [psi0, Omega_s0], t, args=(Omega_s6,))
    psi_sol, ignored = sol[:,0]*(180/np.pi), sol[:,1]
    psi_final.append(psi_sol[-1])

count, bins, ignored = plt.hist(psi_final, 10)
plt.title('Omega_s,6 = 20 days')
plt.xlabel('Final psi value [deg]')
plt.ylabel('Frequency (out of 100 samples)')
plt.xlim(0,180)
```

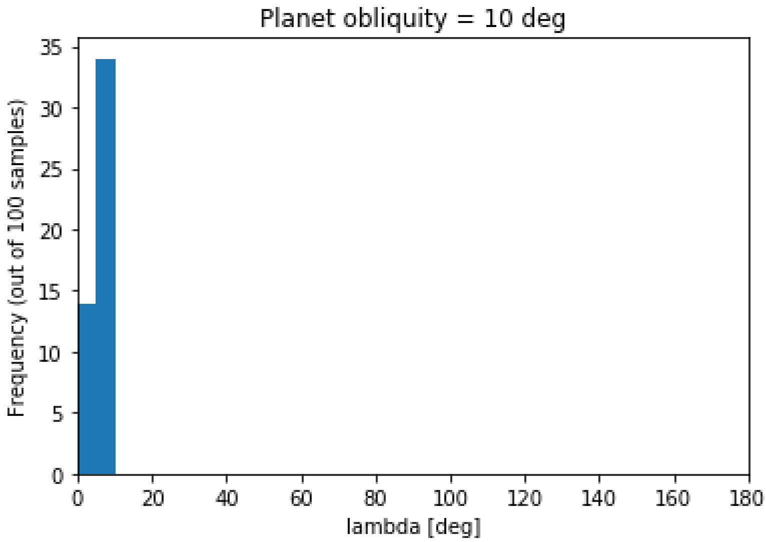
```
plt.savefig('q2c.png', dpi=400)
plt.show()
```

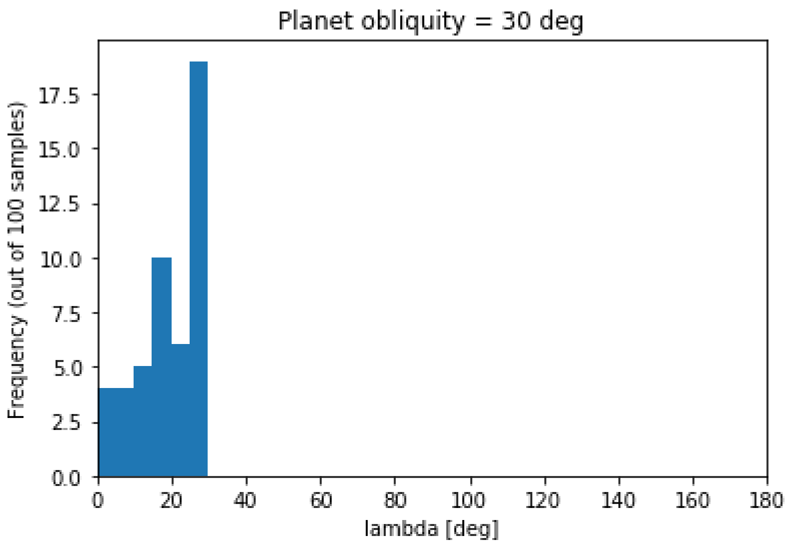


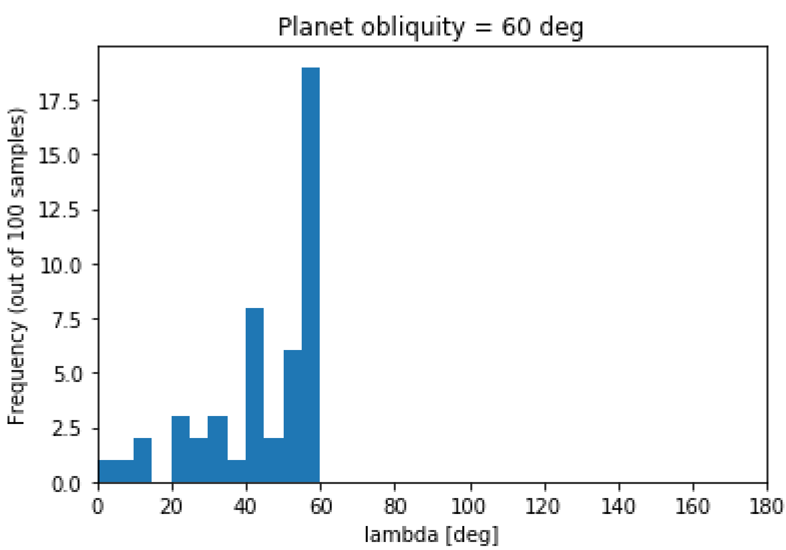


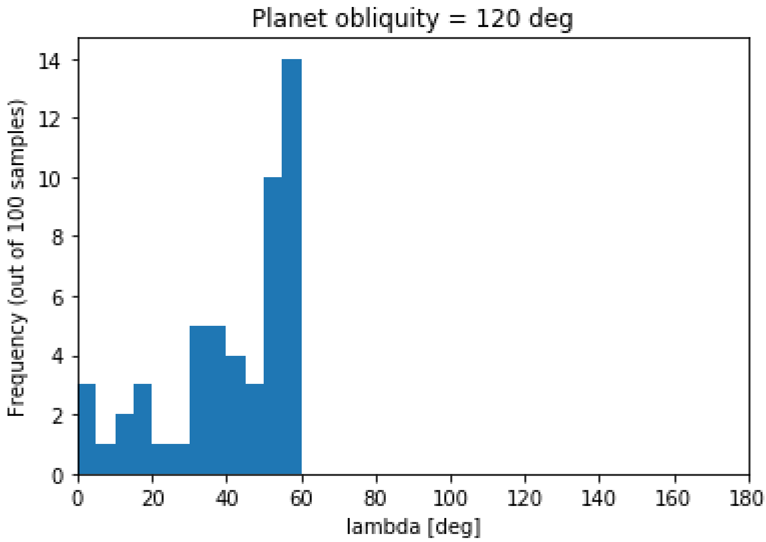


In [8]:
```
# Question 3

Omega_samples = np.random.uniform(0,2*np.pi,100)
i0 = 90*(np.pi/180)
psi10 = 10*(np.pi/180)
psi30 = 30*(np.pi/180)
psi60 = 60*(np.pi/180)
psi120 = 120*(np.pi/180)

def calculate_lambda(psi, Omega):
    return np.arctan(np.sin(psi)*np.sin(Omega)/(np.cos(psi)*np.sin(i0)+np.sin(p

lambda10 = calculate_lambda(psi10,Omega_samples) # projected obliquity
count, bins, ignored = plt.hist(lambda10*(180/np.pi), bins = np.arange(0, 185,
plt.title('Planet obliquity = 10 deg')
plt.xlabel('lambda [deg]')
plt.ylabel('Frequency (out of 100 samples)')
plt.xlim(0,180)
plt.savefig('q3a.png', dpi=400)
plt.show()

lambda30 = calculate_lambda(psi30,Omega_samples)
count, bins, ignored = plt.hist(lambda30*(180/np.pi), bins = np.arange(0, 185,
plt.title('Planet obliquity = 30 deg')
plt.xlabel('lambda [deg]')
plt.ylabel('Frequency (out of 100 samples)')
plt.xlim(0,180)
plt.savefig('q3b.png', dpi=400)
plt.show()

lambda60 = calculate_lambda(psi60,Omega_samples)
count, bins, ignored = plt.hist(lambda60*(180/np.pi), bins = np.arange(0, 185,
plt.title('Planet obliquity = 60 deg')
plt.xlabel('lambda [deg]')
plt.ylabel('Frequency (out of 100 samples)')
plt.xlim(0,180)
plt.savefig('q3c.png', dpi=400)
plt.show()

lambda120 = calculate_lambda(psi120,Omega_samples)
count, bins, ignored = plt.hist(lambda120*(180/np.pi), bins = np.arange(0, 185,
plt.title('Planet obliquity = 120 deg')
plt.xlabel('lambda [deg]')
plt.ylabel('Frequency (out of 100 samples)')
plt.xlim(0,180)
plt.savefig('q3d.png', dpi=400)
plt.show()
```


Planet obliquity = 10 deg
Frequency (out of 100 samples)
0
5
10
15
20
25
30
35
0
20
40
60
80
100
120
140
160
180
lambda [deg]


Planet obliquity = 30 deg
Frequency (out of 100 samples)
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
0
20
40
60
80
100
120
140
160
180
lambda [deg]


Planet obliquity = 60 deg
Frequency (out of 100 samples)
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5
0
20
40
60
80
100
120
140
160
180
lambda [deg]



Planet obliquity = 120 deg
Frequency (out of 100 samples)
lambda [deg]
0
2
4
6
8
10
12
14
0
20
40
60
80
100
120
140
160
180