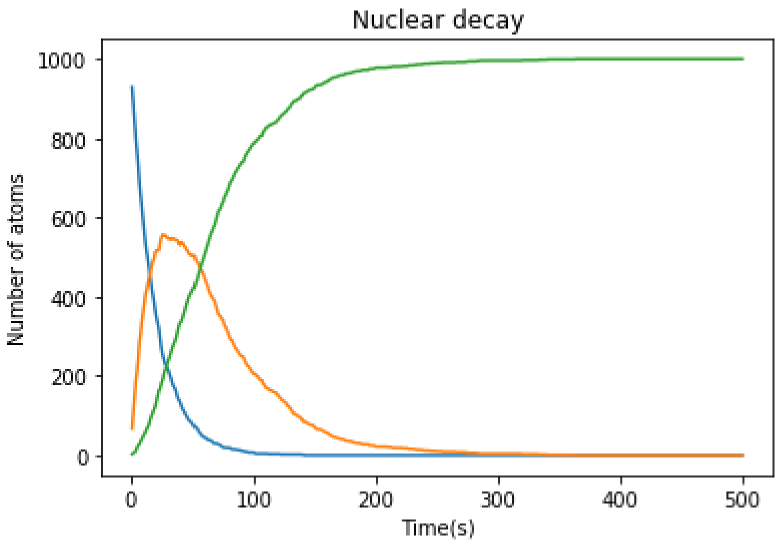
The first cell answers questions 1 and 2. I set up empty arrays to collect the information for time, and the number of atoms and its products. I used a while loop to represent the total time with for loops inside to run the monte carlo method for each relevant atom at each time step. The data is appended to an array to create a graph describing the number of atoms and its products over time.

In [27]:

```
import numpy as np
import matplotlib.pyplot as plt

t_data = []
N_data = []
D_data = []
G_data = []
N = 1000
D = 0
G = 0
t = 0
dt = 1

while t < 500:

    for i in range(N):
        if np.random.rand() <= 0.05:
            N += -1
            D += 1

    for i in range(D):
        if np.random.rand() <= 0.02:
            D += -1
            G += 1

    t += dt
    t_data.append(t)
    N_data.append(N)
    D_data.append(D)
    G_data.append(G)

plt.plot(t_data, N_data)
plt.plot(t_data, D_data)
plt.plot(t_data, G_data)
plt.title("Nuclear decay")
plt.xlabel("Time(s)")
plt.ylabel("Number of atoms")
print("The half life is approximately 15 seconds from the graph and the daughter nucleu
s peaks at 30 seconds.\nThe theoretical value for the half life is 13.5 seconds ")
```

```
The half life is approximately 15 seconds from the graph and the daughter
nucleus peaks at 30 seconds.
The theoretical value for the half life is 13.5 seconds
```



The second cell answers question 3. A while loop runs for as long as the number of transmitted atoms are higher than 50 percent of the initial flux. At each timestep a for loop runs for every undecayed neutron and puts them through a monte carlo method. For every iteration of the for loop the thickness of the material is increased by one atom.

In [25]:

```
N_0 = 5000
N2 = N_0
p_absorb = 0.03
b = 0

while N2 > N_0/2:
    for i in range(N2):
        if np.random.rand() <= p_absorb:
            N2 += -1
    b += 1

print("Thickness in atoms: ", b)
```

```
Thickness in atoms:  23
```

Part 4 uses the same setup and varies the while loop conditions so it runs while the number of nuetrons is greater than or equal to 1 percent of the intitial flux. The number of neutrons and the thickness of the shield are stored in arrays for each iteration. It then plots the fraction of transmitted rays vs the thickness of the shield

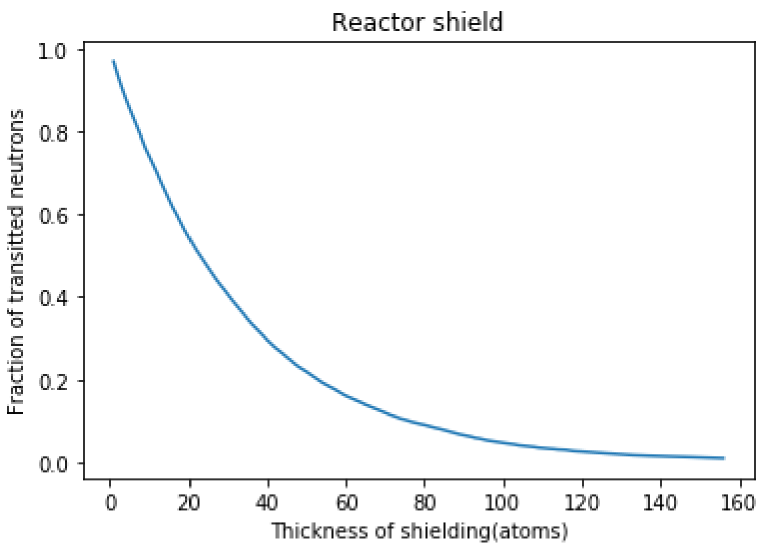
In [24]:

```
b_data = []
N3_data = []
b2 = 0
N0 = 10000
N3 = N0

while N3 >= N0 * 0.01:
    for i in range(N3):
        if np.random.rand() <= p_absorb:
            N3 += -1
    b2 += 1
    N3_data.append(N3/N0)
    b_data.append(b2)

print("Number of atoms: ", b2)
plt.plot(b_data, N3_data)
plt.title("Reactor shield")
plt.xlabel("Thickness of shielding(atoms)")
plt.ylabel("Fraction of transitted neutrons")
```

```
Number of atoms:  156
```

Out[24]:

```
Text(0,0.5,'Fraction of transitted neutrons')
```


Reactor shield
Fraction of transitted neutrons
Thickness of shielding(atoms)