In [2]:
```
#NAME:ANTONY WUGOMA MWAKIACHA
#REG_NO:COM/18/17
#TASK:COM 426 TAKE AWAT CAT

#1. Write a Python program that uses a function to compute the volume of a cylin

import math
radius=float(input("Enter the radius:"))
height=float(input("Enter the height:"))
def volCylinder(r,h):
    v=math.pi*r**2*h
    return v
volume=volCylinder(radius,height)
print("The volume of the cylinder is:", volume)
```

```
Enter the radius:23
Enter the height:22
The volume of the cylinder is: 36561.85530247801
```

In [3]:
```
#2.Write a program to create a graph of the number of atoms remaining in a sampl

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

carbon_atoms = input("Submit the percentage(%) of Carbon 14 atoms:")
```

```
Submit the percentage(%) of Carbon 14 atoms:0.77
```

In [4]:
```
def age_of_carbon_atoms(C): #function to find the age of Carbon atoms
    C = float(C) / 100
    c_rem_c_atm = (C).as_integer_ratio() #function to output the result as an in
    c_rem = c_rem_c_atm[0] #assigining the first element in the tuple a variable
    c_atm = c_rem_c_atm[1]#assigining the second element in the tuple a variable
    t = -8033 * math.log(c_rem / c_atm)
    return t
```

In [5]:
```
print(age_of_carbon_atoms(carbon_atoms))
```

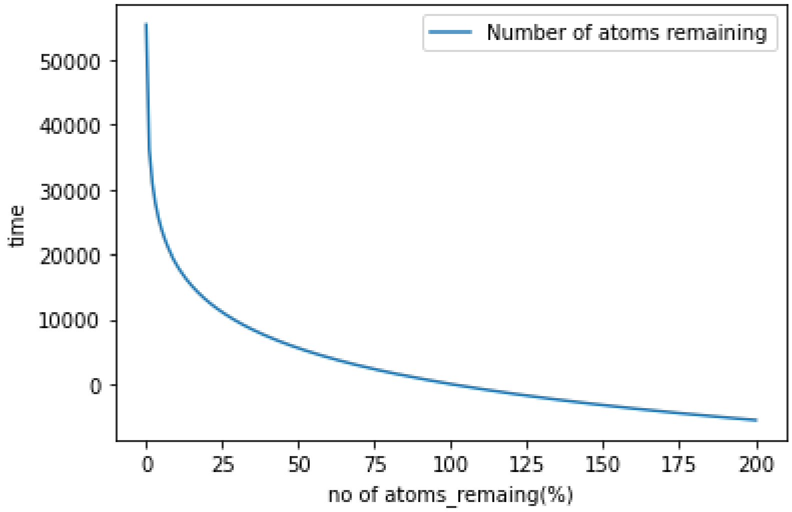
```
39092.87525433403
```

In [6]:
```
no_atoms_remaining = np.arange(0.1, 201, 1)#shows the range of the percentage of
time = [] #initialising the time independent variable into an empty array.Time w
```

In [8]:
```
for atoms in no_atoms_remaining: #for each element in the array of the range of
    time.append(age_of_carbon_atoms(atoms))
```

In [10]:
```
plt.xlabel('no of atoms_remaing(%)')
plt.ylabel('time')
plt.plot(no_atoms_remaining, time, label='Number of atoms remaining')
plt.legend()
```

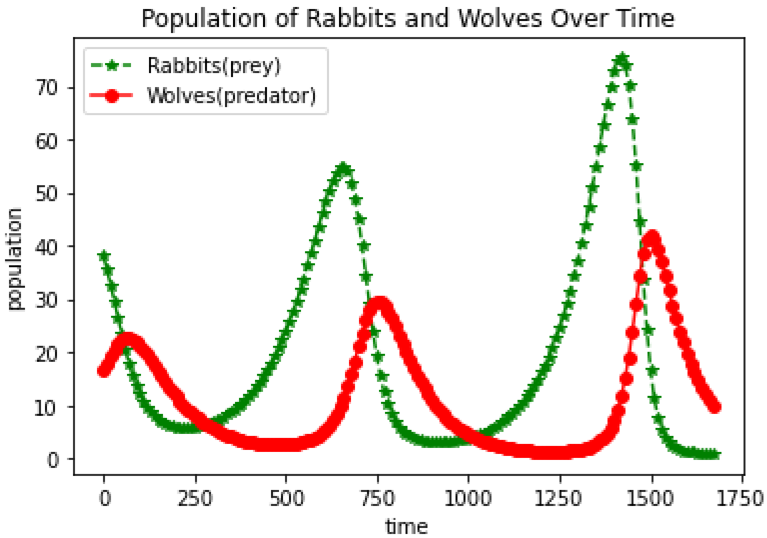
Out[10]:
```
<matplotlib.legend.Legend at 0x7f0cbc0abbb0>
```



In [12]:

```
#3.Create a model that incorporates these new factors and plot the population o
import matplotlib.pyplot as plt
import numpy as np

#Iniitialising variables with provided values
original_rabbit_population = 40
original_wolf_population = 15
rabbit_growth_rate = 0.1
wolf_growth_rate = 0.005
rabbit_death_rate = 0.01
wolf_death_rate = 0.1
time_period = 200


rabbit_populations = [] #initialising an empty array for rabbit population
wolf_populations = []  #initialising an empty array for wolf population

time = np.arange(0, 1680, 10) # within 5 years

for t in time:#for each population at a specific time
    new_rabbit_population = (original_rabbit_population + (original_rabbit_popul
    new_wolf_population = (original_wolf_population + (original_wolf_population

    rabbit_populations.append((new_rabbit_population))
    wolf_populations.append((new_wolf_population))
    original_rabbit_population = (new_rabbit_population)
    original_wolf_population = (new_wolf_population)


plt.xlabel('time')
plt.ylabel('population')
plt.title('Population of Rabbits and Wolves Over Time')
plt.plot(time, rabbit_populations, 'g*--', label='Rabbits(prey)')
plt.plot(time, wolf_populations, 'ro-', label='Wolves(predator)')
plt.legend()
```

Out[12]: <matplotlib.legend.Legend at 0x7f0cacf29d60>


Population of Rabbits and Wolves Over Time
Rabbits(prey)
Wolves(predator)
population
time
0
10
20
30
40
50
60
70
0
250
500
750
1000
1250
1500
1750

In [ ]: