In [1]:
```
import numpy as np
from scipy.stats import uniform, bernoulli
import matplotlib.pyplot as plt
```

## Άσκηση 1

In [2]:
```
theta = 10
distr_u = uniform(0, theta)
```

In [4]:
```
x = distr_u.rvs(1)
print(x)
```

```
[2.75146071]
```

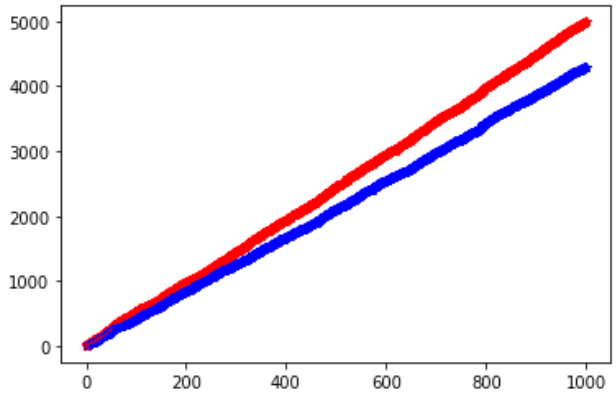
In [5]:
```
def d(a, x):
    return a * x
```

In [6]:
```
a = np.sqrt(2)
d(a, x)
```

Out[6]:
```
array([3.89115306])
```

In [7]:
```
def L(theta, d):
    return np.abs(theta - d)
```

In [17]:
```
N = 1000
a = np.sqrt(2)
a2 = 2
# plt.figure()
loss = 0
loss2 = 0
x = distr_u.rvs(N)
for i in range(N):
    # x = distr_u.rvs(1)
    loss += L(theta, d(a, x[i]))
    loss2 += L(theta, d(a2, x[i]))
    plt.plot(i, loss, '*', c = 'b')
    plt.plot(i, loss2, '*', c = 'r')
```



### Άσκηση 2

In [18]:
```
n = 2
```

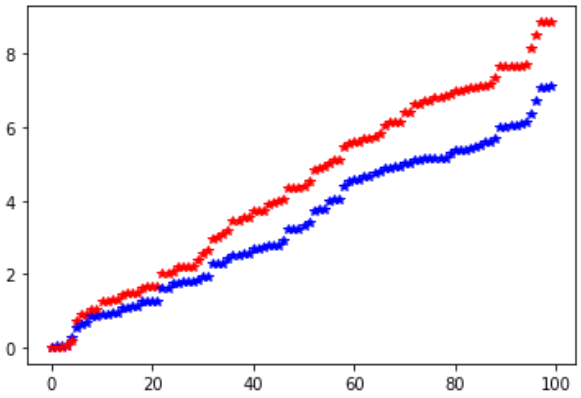
In [19]:
```
def d(a, x):
    return a * np.mean(x)
```

In [31]:
```
def L(theta, d):
    return (theta - d)**2
```

In [32]:
```
a = 2*n / (2*n + 1)
a2 = 1
```

In [33]:
```
N = 100
distr_u = uniform(0,1)
p = distr_u.rvs(N)
```

In [34]:
```
plt.figure()
loss = 0
loss2 = 0
for i in range(N):
    distr_b = bernoulli(p[i])
    x = distr_b.rvs(n)
    loss += L(p[i], d(a, x))
    loss2 += L(p[i], d(a2, x))
    plt.plot(i, loss, '*', c='b')
    plt.plot(i, loss2, '*', c='r')
```




In [ ]:
```
```