Esercizio 1

a) Definiamo i seguenti eventi:

$$\begin{aligned}
A &= \{\text{viene scelta l'urna } A\}\\
B &= \{\text{viene scelta l'urna } B\}\\
R_i &= \{\text{alla } i\text{-esima estrazione si ottiene una pallina rossa}\}\\
N_i &= \{\text{alla } i\text{-esima estrazione si ottiene una pallina nera}\}\ .
\end{aligned}$$

Osserviamo che gli eventi A e B costituiscono una partizione dell'evento certo. Dunque la probabilità richiesta vale

$$\mathrm{P}(R_1) = \mathrm{P}(R_1 \cap A) + \mathrm{P}(R_1 \cap B) = \mathrm{P}(R_1\,|\,A)\mathrm{P}(A) + \mathrm{P}(R_1\,|\,B)\mathrm{P}(B)\ .$$

Per come il problema è stato posto è chiaro che deve essere

$$\mathrm{P}(R_1\,|\,A) = 1, \qquad \mathrm{P}(R_1\,|\,B) = \tfrac{r}{n}, \qquad \mathrm{P}(A) = \mathrm{P}(B) = \tfrac{1}{2}$$

e dunque

$$\mathrm{P}(R_1) = \tfrac{1}{2} + \tfrac{1}{2}\tfrac{r}{n} = \tfrac{1}{2}\big(1 + \tfrac{r}{n}\big)\ .$$

b) Indichiamo con C l'evento ‘‘le prime due estrazioni danno palline di colori diversi’’. La probabilità che in due estrazioni dall'urna A si ottengano una pallina rossa e una nera è chiaramente 0. Invece il numero di palline rosse estratte dall'urna B in due estrazioni segue una legge binomiale $B(2, \frac{r}{n})$. Dunque

$$\begin{aligned}
&\mathrm{P}(C\,|\,A) = 0\\
&\mathrm{P}(C\,|\,B) = \binom{2}{1}\frac{r}{n}\Big(1 - \frac{r}{n}\Big) = 2\,\frac{r(n-r)}{n^2}\\
&\mathrm{P}(C) = \mathrm{P}(C\,|\,A)\mathrm{P}(A) + \mathrm{P}(C\,|\,B)\mathrm{P}(B) = \frac{r(n-r)}{n^2}\ \cdot
\end{aligned}$$

c) Indichiamo con T la v.a. ‘‘tempo d'attesa della prima estrazione di una pallina rossa’’. Dobbiamo calcolare la speranza matematica di T e per farlo calcoliamone prima la legge. Ora, sempre con la formula delle probabilità totali (1.12),

$$\mathrm{P}(T = k) = \mathrm{P}(T = k\,|\,A)\mathrm{P}(A) + \mathrm{P}(T = k\,|\,B)\mathrm{P}(B)\ .$$

Ma, poiché l'urna A contiene solo palline rosse,

$$\mathrm{P}(T = k\,|\,A) = \begin{cases} 1 & \text{se } k = 1\\ 0 & \text{altrimenti}\end{cases}$$

mentre

$$\mathrm{P}(T = k\,|\,B) = \begin{cases} p(1-p)^{k-1} & \text{se } k = 1, 2, \ldots\\ 0 & \text{altrimenti}\end{cases}$$

dove abbiamo posto $p = \frac{r}{n}$ e dunque

$$\begin{aligned}\mathrm{E}(T) &= \sum_{k=1}^{\infty} k\mathrm{P}(T = k) = \sum_{k=1}^{\infty} k\big(\mathrm{P}(T = k\,|\,A)\mathrm{P}(A) + \mathrm{P}(T = k\,|\,B)\mathrm{P}(B)\big) = \\ &= \frac{1}{2} + \frac{1}{2}\sum_{k=1}^{\infty} kp(1-p)^{k-1} = \frac{1}{2}\Big(1 + \frac{1}{p}\Big) = \frac{1}{2}\Big(1 + \frac{n}{r}\Big)\end{aligned}$$

dove abbiamo riconosciuto nell'ultima serie la speranza matematica di una legge geometrica modificata, che vale appunto $\frac{1}{p}$.

d) Poniamo $E_k = R_1 \cap \ldots \cap R_k$. E_k è l'evento ''le prime k estrazioni hanno dato tutte palline rosse''. La probabilità richiesta non è altro che $\mathrm{P}(A\,|\,E_k)$. Per la formula di Bayes

$$\mathrm{P}(A\,|\,E_k) = \frac{\mathrm{P}(E_k\,|\,A)\mathrm{P}(A)}{\mathrm{P}(E_k)} \cdot$$

Ora $\mathrm{P}(E_k\,|\,A) = 1$ mentre $\mathrm{P}(A) = \frac{1}{2}$. Resta da calcolare $\mathrm{P}(E_k)$. Ancora la formula delle probabilità totali (1.12) dà

$$\mathrm{P}(E_k) = \underbrace{\mathrm{P}(E_k\,|\,A)}_{=1}\mathrm{P}(A) + \mathrm{P}(E_k\,|\,B)\mathrm{P}(B) \ .$$

Ma se l'urna prescelta è la B il numero di palline rosse estratte segue una legge binomiale $B(k, \frac{r}{n})$. Dunque

$$\mathrm{P}(E_k\,|\,B) = \mathrm{P}(R_1\,|\,B)\ldots\mathrm{P}(R_k\,|\,B) = (\tfrac{r}{n})^k$$

e in conclusione $\mathrm{P}(E_k) = \frac{1}{2}(1 + (\frac{r}{n})^k)$ e

$$\mathrm{P}(A\,|\,E_k) = \frac{1}{1 + (\frac{r}{n})^k} \cdot$$

Per $n = 12$, $r = 4$

$$\mathrm{P}(A\,|\,E_k) = \frac{1}{1 + 3^{-k}}$$

e dopo qualche manipolazione algebrica si vede che perché sia

$$\frac{1}{1 + 3^{-k}} \geq 0.99$$

deve essere $3^k \geq 99$ e cioè $k \geq 5$.

In [1]:
```
# Esercizio 2
```

In [2]:
```
import numpy as np
```

In [5]:
```
X = np.array([4.65, 4.7, 4.75, 4.77, 4.8, 4.95, 5., 4.75, 4.54, 4.66])
```

In [7]:
```
# Media campionaria
mu = np.mean(X)
display(mu)
```

```
4.757000000000001
```

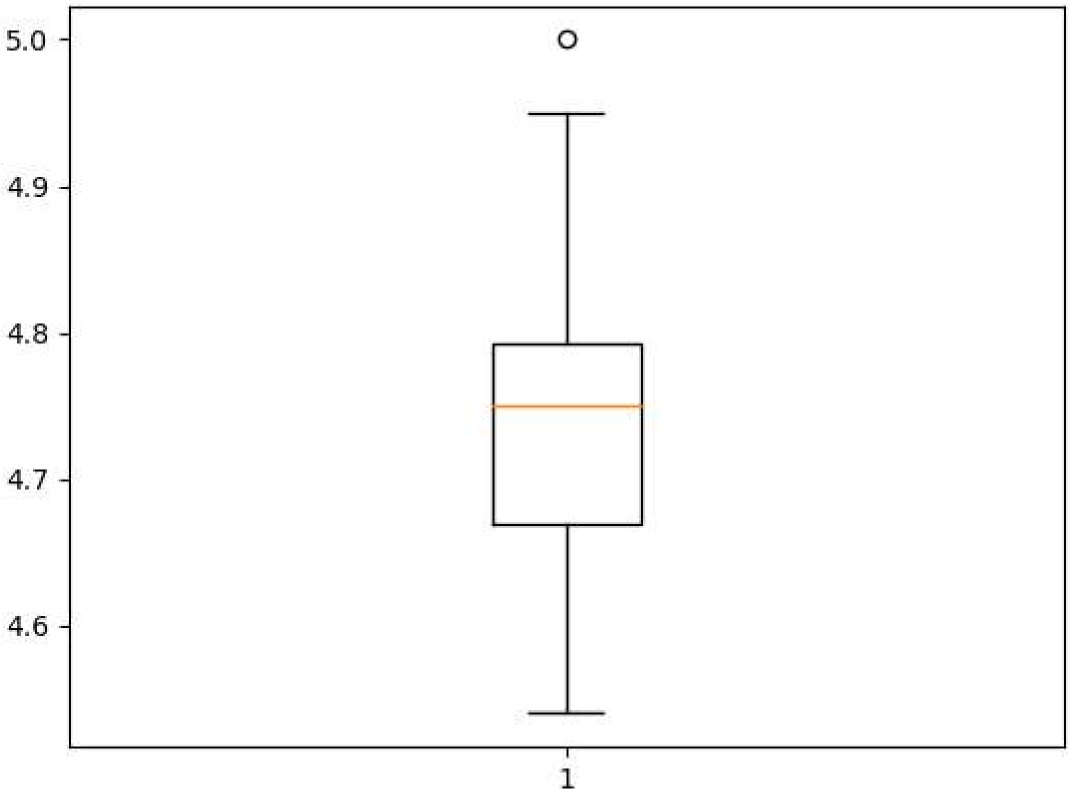
In [9]:
```
# Deviazione standard
S = np.std(X,ddof=1)
display(S)
```

```
0.13727911551126612
```

In [14]:
```
# Calcoliamo l'intervallo di confidenza al 95% per la media
from scipy.stats import t
n = X.size
df = n-1
alpha = 0.01
t_alpha = t.ppf(1.-alpha/2., df)
Il = mu - S/np.sqrt(n)*t_alpha
Ir = mu + S/np.sqrt(n)*t_alpha
display(Il,Ir)
```

```
4.615919868214308
4.898080131785693
```

In [15]:
```
# crea box_plot
import matplotlib.pyplot as plt
fig, ax = plt.subplots(1, 1)

ax.boxplot(X)
plt.show()
```



In [19]:
```
# Test sulla media
mu_0 = 4.7

T_0 = (mu-mu_0)/S*np.sqrt(n)
display(T_0)

alpha = 0.05
T = t.ppf(1.-alpha, n-1)
display(T)
```

```
1.3130171035725113
1.8331129326536335
```

In [20]:
```
# T_0 è minore di T pertanto non è possibile rigettare l'ipotesi nulla
```

In [ ]:

In [1]:
```
# Esercizio 3
```

In [2]:
```
import numpy as np
```

In [3]:
```
x = np.array([2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017])    # anno
y = np.array([10, 11, 10.5, 12, 10, 12, 11, 13])    # temperatura
display(x,y)
```

```
array([2010, 2011, 2012, 2013, 2014, 2015, 2016, 2017])
array([10. , 11. , 10.5, 12. , 10. , 12. , 11. , 13. ])
```

In [4]:
```
# Calcolo dei coefficienti di regressione y = b_0 + b_1 x
x_bar = np.mean(x)
y_bar = np.mean(y)
n = x.size
```

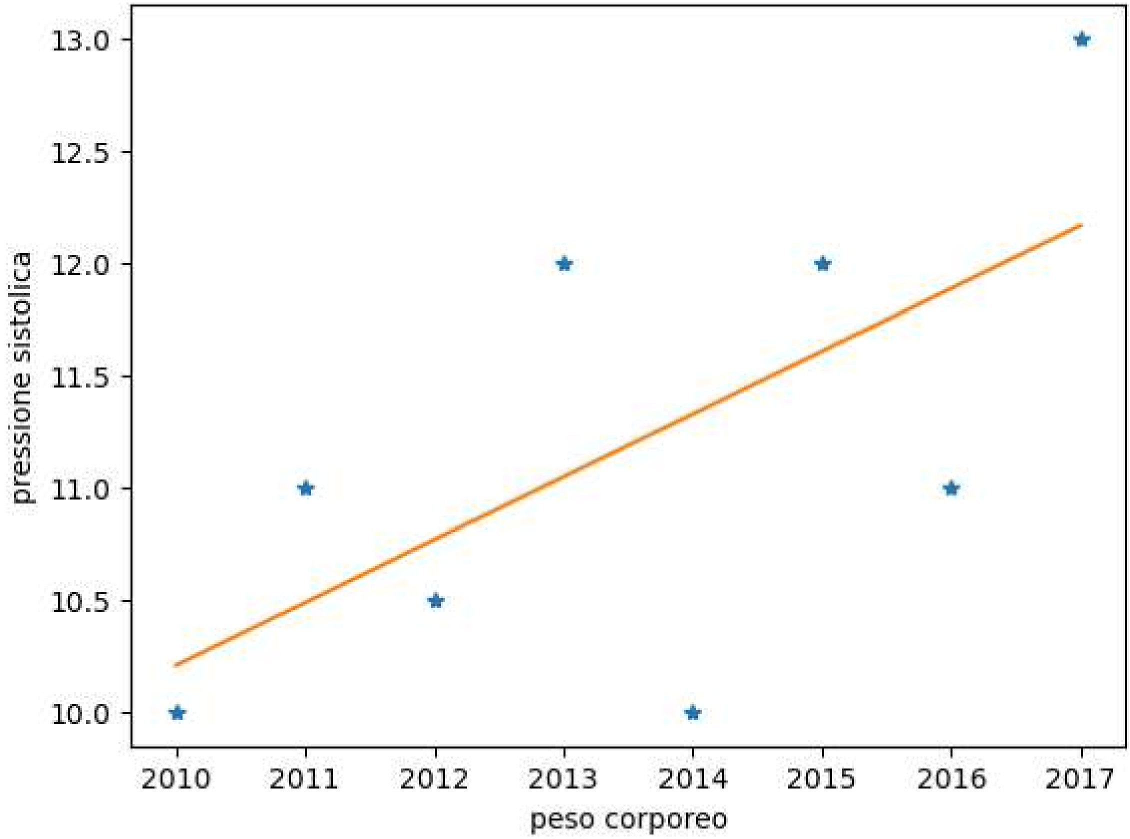
In [6]:
```
sig_xy = np.sum((x-x_bar)*(y-y_bar))/n
sig_x_2 = np.sum((x-x_bar)**2.)/n
b_0 = y_bar - sig_xy/sig_x_2*x_bar
b_1 = sig_xy/sig_x_2
display(b_0, b_1)
```

```
-552.1130952380953
0.27976190476190477
```

In [9]:
```
y_hat = b_0 + b_1*x
r = y - y_hat
s2 = np.sum(r**2.)/(n-2)

from scipy.stats import t
alpha = 0.05
T = t.ppf(1.-alpha/2.,n-2)
b_0_l = b_0 - np.sqrt(s2)*np.sqrt(1./n + x_bar**2./(n*sig_x_2))*T
b_0_r = b_0 + np.sqrt(s2)*np.sqrt(1./n + x_bar**2./(n*sig_x_2))*T

b_1_l = b_1 - np.sqrt(s2)/np.sqrt(n*sig_x_2)*T
b_1_r = b_1 + np.sqrt(s2)/np.sqrt(n*sig_x_2)*T

display(b_0)
display(b_0_l,b_0_r)

display(b_1)
display(b_1_l,b_1_r)
```

```
-552.1130952380953
-1223.6419698314085
119.41577935521775
0.27976190476190477
-0.05375110233645042
0.61327491186026
```

In [13]:
```
xx = np.linspace(2010.,2017.,100)
yy = b_0 + b_1*xx
import matplotlib.pyplot as plt
plt.plot(x, y, '*')
plt.plot(xx, yy)
plt.xlabel('peso corporeo')
plt.ylabel('pressione sistolica')
plt.show()
```



In [14]:
```
# Calcolo del coefficiente di determinazione
sig_y_2 = np.sum((y-y_bar)**2.)/n
R2 = sig_xy**2./(sig_x_2*sig_y_2)
display(R2)
```

0.4125116713352007

In [17]:
```
# Temperatura prevista per l'anno 2022
x_s = 2022.
y_s = b_0 + b_1*x_s
display(y_s)
```

13.565476190476147

In [19]:
```
# Intervallo di confidenza
from scipy.stats import t
alpha = 0.05
T = t.ppf(1.-alpha/2.,n-2)

I_l = b_0 + b_1*x_s - np.sqrt(1 + 1/n + (x_s-x_bar)**2./(n*sig_x_2))*T
I_r = b_0 + b_1*x_s + np.sqrt(1 + 1/n + (x_s-x_bar)**2./(n*sig_x_2))*T

display(I_l)
display(I_r)
```

9.438066006592889
17.692886374359404

In [ ]: