Задание 1.

Исходные данные:

Смоделируем стократное подбрасывание монетки. Рассмотрим случайную величину, равную числу выпаданий орла.

Решение:

```
Python 3.8.5 (default, May 27 2021, 13:30:53)

[GCC 9.3.0] on linux

Type "help", "copyright", "credits" or "license" for more information.

>>> import numpy as np

>>> a = np.random.randint(0, 2, size=100).sum()

>>> print(a)

48

>>> b = np.random.binomial(n=100, p=0.5)

>>> print(b)

53

>>> c = np.random.binomial(n=100, p=0.5, size=50)

>>> print(c)

[58 48 54 56 56 60 40 48 47 43 45 45 56 47 51 37 46 45 54 47 57 51 47 43

 56 53 46 52 49 54 46 50 34 52 53 57 46 52 55 57 58 59 49 54 48 54 58 49

 53 49]

>>>
```

Задание 2.

Исходные данные:

усть X — сумма значений двух подбрасываемых игральных кубиков. Вот её закон распределения:

Решение:

2==1/36

3==2/36

4==3/36

5==4/36

6==5/36

7==6/36

8==5/36

9==4/36

10==3/36

11==2/36

12==1/36

Задание 3.

Исходные данные:

В урне 8 шаров, из которых 5 белых, остальные — чёрные. Наудачу вынимают 3 шара. Найти закон распределения количества белых шаров в выборке.

Решение:

```
Python 3.8.5 (default, May 27 2021, 13:30:53)

[GCC 9.3.0] on linux

Type "help", "copyright", "credits" or "license" for more information.

>>> import math

>>> from math import factorial

>>> m = 8

>>> k = 3
>>> def combinations(m, k):
...     try:
...         m = int(input("Введите число: "))
...         k = int(input("Введите число: "))
...     except zerodivisionerror:
...         return
...     s = int(factorial(m) / (factorial(k) * factorial(m - k)))
...     return s
...
>>> print(combinations(m, k))
Введите число: 8
Введите число: 3
56
>>>


Python 3.8.5 (default, May 27 2021, 13:30:53)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import math
>>> from math import factorial
>>> a = 0
>>> b = 3
>>> m = 8
>>> k = 0
>>> def combinations(m, k):
```

```
...     try:
...         m = int(input("Введите число: "))
...         k = int(input("Введите число: "))
...     except zerodivisionerror:
...         return
...     s = int(factorial(m) / (factorial(k) * factorial(m - k)))
...     return s
...
>>> print(combinations(m, k))
Введите число: 8
Введите число: 0
1
>>>
```

```
Python 3.8.5 (default, May 27 2021, 13:30:53)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import math
>>> from math import factorial
>>> m = 5
>>> k = 1
>>> def combinations(m, k):
...     try:
...         m = int(input("Введите число: "))
...         k = int(input("Введите число: "))
...     except zerodivisionerror:
...         return
...     s = int(factorial(m) / (factorial(k) * factorial(m - k)))
...     return s
...
>>> print(combinations(m, k))
Введите число: 5
Введите число: 1
5
>>> import math
>>> from math import factorial
>>> a = 3
>>> b = 2
>>> def combinations(a, b):
...     try:
...         a = int(input("Введите число: "))
...         b = int(input("Введите число: "))
...     except zerodivisionerror:
...         return
...     s = int(factorial(a) / (factorial(b) * factorial(a - b)))
...     return s
...
>>> print(combinations(a, b))
Введите число: 3
Введите число: 2
```

```
3
>>> c = 5 * 3
>>> print(c)
15
>>>

vik@vik-Z580:~$ command python3
Python 3.8.5 (default, May 27 2021, 13:30:53)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import math
>>> from math import factorial
>>> m = 5
>>> k = 2
>>> def combinations(m, k):
...     try:
...         m = int(input("Введите число: "))
...         k = int(input("Введите число: "))
...     except zerodivisionerror:
...         return
...     s = int(factorial(m) / (factorial(k) * factorial(m - k)))
...     return s
...
>>> print(combinations(m, k))
Введите число: 5
```

```
Введите число: 2
10
>>> import math
>>> from math import factorial
>>> a = 3
>>> b = 1
>>> def combinations(a, b):
...     try:
...         a = int(input("Введите число: "))
...         b = int(input("Введите число: "))
...     except zerodivisionerror:
...         return
...     s = int(factorial(a) / (factorial(b) * factorial(a - b)))
...     return s
...
>>> print(combinations(a, b))
Введите число: 3
Введите число: 1
3
>>> f = 10 * 3
>>> print(f)
30
>>>
```

```
Python 3.8.5 (default, May 27 2021, 13:30:53)
```

```
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import math
>>> from math import factorial
>>> m = 5
>>> k = 3
>>> def combinations(m, k):
...     try:
...         m = int(input("Введите число: "))
...         k = int(input("Введите число: "))
...     except zerodivisionerror:
...         return
...     s = int(factorial(m) / (factorial(k) * factorial(m - k)))
...     return s
...
>>> print(combinations(m, k))
Введите число: 5
Введите число: 3
10
>>> import math
>>> from math import factorial
>>> a = 3
>>> b = 0
>>> def combinations(a, b):
...     try:
```

```
...         a = int(input("Введите число: "))
...         b = int(input("Введите число: "))
...     except zerodivisionerror:
...         return
...     s = int(factorial(a) / (factorial(b) * factorial(a - b)))
...     return s
...
>>> print(combinations(a, b))
Введите число: 3
Введите число: 0
1
>>>
>>> f = 10 * 1
>>> print(f)
10
>>>
```

Задание 4.

Исходные данные:

В задании 2 мы записали закон распределения суммы значений при броске двух игральных кубиков. Сделаем то же самое с использованием свойства суммы случайных величин.

Распределение значений при броске одного кубика имеет т.н. *равномерное распределение*: каждое из значений от 1 до 6 имеет одинаковую вероятность $P(X = k) = 1 / 6$.

Решение:

```
Python 3.8.5 (default, May 27 2021, 13:30:53)
[GCC 9.3.0] on linux
```

```
Type "help", "copyright", "credits" or "license" for more information.

>>> import itertools

>>> from itertools import product

>>> pairs = list(product(range(1, 7), repeat=2))

>>> pairs

[(1, 1), (1, 2), (1, 3), (1, 4), (1, 5), (1, 6), (2, 1), (2, 2), (2, 3), (2, 4), (2, 5), (2, 6), (3, 1), (3, 2), (3, 3), (3, 4),
(3, 5), (3, 6), (4, 1), (4, 2), (4, 3), (4, 4), (4, 5), (4, 6), (5, 1), (5, 2), (5, 3), (5, 4), (5, 5), (5, 6), (6, 1), (6, 2),
(6, 3), (6, 4), (6, 5), (6, 6)]

>>>

>>> import numpy as np

>>> probabilities = np.ones(len(pairs)) / 36

>>> probabilities

array([0.02777778, 0.02777778, 0.02777778, 0.02777778, 0.02777778,

       0.02777778, 0.02777778, 0.02777778, 0.02777778, 0.02777778,

       0.02777778, 0.02777778, 0.02777778, 0.02777778, 0.02777778,

       0.02777778, 0.02777778, 0.02777778, 0.02777778, 0.02777778,

       0.02777778, 0.02777778, 0.02777778, 0.02777778, 0.02777778,

       0.02777778, 0.02777778, 0.02777778, 0.02777778, 0.02777778,

       0.02777778, 0.02777778, 0.02777778, 0.02777778, 0.02777778,

       0.02777778])

>>>

>>> values = list(map(sum, pairs))

>>> values

[2, 3, 4, 5, 6, 7, 3, 4, 5, 6, 7, 8, 4, 5, 6, 7, 8, 9, 5, 6, 7, 8, 9, 10, 6, 7, 8, 9, 10, 11, 7, 8, 9, 10, 11, 12]

>>>

>>> import pandas as pd
>>> z = pd.DataFrame({'value': values, 'probability': probabilities})
>>> z
    value  probability
0       2     0.027778
1       3     0.027778
2       4     0.027778
3       5     0.027778
4       6     0.027778
5       7     0.027778
6       3     0.027778
7       4     0.027778
8       5     0.027778
9       6     0.027778
10      7     0.027778
11      8     0.027778
12      4     0.027778
13      5     0.027778
14      6     0.027778
15      7     0.027778
16      8     0.027778
17      9     0.027778
18      5     0.027778
19      6     0.027778
20      7     0.027778
21      8     0.027778
```

```
22     9    0.027778
23    10     0.027778
24     6    0.027778
25     7    0.027778
26     8    0.027778
27     9    0.027778
28    10     0.027778
29    11     0.027778
30     7    0.027778
31     8    0.027778
32     9    0.027778
33    10     0.027778
34    11     0.027778
35    12     0.027778
>>>
>>> z_probabilities = z.groupby('value')['probability'].sum()
>>> z_probabilities
value
2    0.027778
3    0.055556
4    0.083333
5    0.111111
6    0.138889
7    0.166667
8    0.138889
```

```
9     0.111111
10    0.083333
11    0.055556
12    0.027778
Name: probability, dtype: float64
>>>
```

Задание 5.

Исходные данные:

Посчитаем математическое ожидание случайной величины из примера 2.

Решение:

```
Python 3.8.5 (default, May 27 2021, 13:30:53)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy as np
>>> x_values = np.arange(2, 13)
>>> print(x_values)
[ 2  3  4  5  6  7  8  9 10 11 12]
>>>
>>> x_probabilities = np.array([1, 2, 3, 4, 5, 6, 5, 4, 3, 2, 1]) /36
>>> print(x_probabilities)
[0.02777778 0.05555556 0.08333333 0.11111111 0.13888889 0.16666667
 0.13888889 0.11111111 0.08333333 0.05555556 0.02777778]
>>>
>>>
```

```
>>> m = (x_values * x_probabilities).sum()
>>> print(m)
6.999999999999998
>>>
```

Задание 6.

Исходные данные:

Посчитаем дисперсию случайной величины из примера 2. Её математическое ожидание мы уже считали.

Решение:

```
vik@vik-Z580:~$ command python3
Python 3.8.5 (default, May 27 2021, 13:30:53)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy as np
>>> x_values = np.arange(2, 13)
>>> print(x_values)
[ 2  3  4  5  6  7  8  9 10 11 12]
>>>
>>> m = 7
>>> m
7
>>>
>>> y_values = x_values - m
>>> print(y_values)
[-5 -4 -3 -2 -1  0  1  2  3  4  5]
>>>
>>> z_values = y_values **2
>>> print(z_values)
[25 16  9  4  1  0  1  4  9 16 25]
>>>
>>> x_probabilities = np.array([1, 2, 3, 4, 5, 6, 5, 4, 3, 2, 1]) / 36
>>> print(x_probabilities)
[0.02777778 0.05555556 0.08333333 0.11111111 0.13888889 0.16666667
 0.13888889 0.11111111 0.08333333 0.05555556 0.02777778]
>>>
>>> d = (z_values * x_probabilities).sum()
>>> print(d)
5.833333333333334
>>>
```

Задание 7.

Исходные данные:

при трёхкратном подбрасывании монеты. Возможные значения такой случайной величины: $x_1 = 0$, $x_2 = 1$, $x_3 = 2$, $x_4 = 3$.

Решение:

```
Python 3.8.5 (default, May 27 2021, 13:30:53)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
```

```
>>> x0 = 0
>>> a = 3 ** 0 / 0.5 ** 0 / 0.5 ** 3
>>>
>>> 3 ** 0
1
>>>
>>> 0.5 ** 0
1.0
>>>
>>> 0.5 ** 3
0.125
>>> x0 = 0.125
>>>
```

```
Python 3.8.5 (default, May 27 2021, 13:30:53)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> x1 = 1
>>> a = 3 ** 1 / 0.5 ** 1 / 0.5 ** 2
>>> 3 ** 1
3
>>>
>>> 0.5 ** 1
0.5
>>>
```

```
>>> 0.5 ** 2
0.25
>>> x1 = 0.375
>>>
```

```
Python 3.8.5 (default, May 27 2021, 13:30:53)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> x2 = 2
>>> a = 3 ** 2 / 0.5 ** 2/ 0.5 ** 1
>>> 3 ** 2
9
>>> 0.5 ** 2
0.25
>>> 0.5 ** 1
0.5
>>> x2 = 0.375
>>>
```

```
Python 3.8.5 (default, May 27 2021, 13:30:53)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> x3 = 3
>>> a = 3 ** 3 / 0.5 ** 3 / 0.5 ** 0
>>> 3 ** 3
```

```
27

>>> 0.5 ** 3

0.125

>>>

>>> 0.5 ** 0

1.0

>>> x3 = 0.125

>>>
```

Задание 8.

Исходные данные:

Посчитаем математическое ожидание распределения из предыдущего примера.

Решение:

m = 3 * 0.5

m = 0.5

d = 3 * 0.5 * (1 — 0.5)

d = 3. * 0.5 * 0.5

d = 0.75

Задание 9.

Исходные данные:

среднем за час мимо автобусной остановки проезжают 30 автобусов. Какова вероятность, что за час мимо остановки проедут: а) 30 автобусов? б) не более 15 автобусов? в) более 50 автобусов?

Решение:

а) 30 автобусов?

```
Python 3.8.5 (default, May 27 2021, 13:30:53)
```

```
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy as np
>>> import math
>>> from math import factorial
>>> np.exp([1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27,
28, 29, 30])
array([2.71828183e+00, 7.38905610e+00, 2.00855369e+01, 5.45981500e+01,
       1.48413159e+02, 4.03428793e+02, 1.09663316e+03, 2.98095799e+03,
       8.10308393e+03, 2.20264658e+04, 5.98741417e+04, 1.62754791e+05,
       4.42413392e+05, 1.20260428e+06, 3.26901737e+06, 8.88611052e+06,
       2.41549528e+07, 6.56599691e+07, 1.78482301e+08, 4.85165195e+08,
       1.31881573e+09, 3.58491285e+09, 9.74480345e+09, 2.64891221e+10,
       7.20048993e+10, 1.95729609e+11, 5.32048241e+11, 1.44625706e+12,
       3.93133430e+12, 1.06864746e+13])
>>>
>>> k = 30
>>> lambda_ = 30
>>> lambda_ = lambda_
>>> def poisson_proba(k, lambda_):
...     try:
...         k = int(input("Введите число: "))
...         lambda_ = float(input("Введите число: "))
...     except zerodivisionerror:
...         return
...     s = (lambda_ ** k) * (np.exp(-lambda_)) / np.math.factorial(k)

...     return s

...

>>> print(poisson_proba(k, lambda_))

Введите число: 30

Введите число: 30

0.07263452647159149

>>>
```

б) не более 15 автобусов?

```
Python 3.8.5 (default, May 27 2021, 13:30:53)

[GCC 9.3.0] on linux

Type "help", "copyright", "credits" or "license" for more information.

>>> import numpy as np

>>> import math

>>> from math import factorial

>>> k = 15

>>> lambda_ = 15

>>> lambda_ = lambda_

>>> np.exp([1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15])

array([2.71828183e+00, 7.38905610e+00, 2.00855369e+01, 5.45981500e+01,

       1.48413159e+02, 4.03428793e+02, 1.09663316e+03, 2.98095799e+03,

       8.10308393e+03, 2.20264658e+04, 5.98741417e+04, 1.62754791e+05,

       4.42413392e+05, 1.20260428e+06, 3.26901737e+06])

>>>
```

```
>>> def poisson_proba(k, lambda_):
...     try:
...         k = int(input("Введите число: "))
...         lambda_ = float(input("Введите число: "))
...     except zerodivisionerror:
...         return
...     s = (lambda_ ** k) * (np.exp(-lambda_)) / np.math.factorial(k)
...     return s
...
>>> print(poisson_proba(k, lambda_))
Введите число: 15
Введите число: 15
0.10243586666453419
>>>
```

в) более 50 автобусов?

```
Python 3.8.5 (default, May 27 2021, 13:30:53)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> import numpy as np
>>> import math
>>> from math import factorial
>>> k = 51
>>> lambda_ = 51
>>> lambda_ = lambda_
>>> np.exp([1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27,
28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43, 44, 45, 46, 47, 48, 49, 50, 51])

array([2.71828183e+00, 7.38905610e+00, 2.00855369e+01, 5.45981500e+01,

       1.48413159e+02, 4.03428793e+02, 1.09663316e+03, 2.98095799e+03,

       8.10308393e+03, 2.20264658e+04, 5.98741417e+04, 1.62754791e+05,

       4.42413392e+05, 1.20260428e+06, 3.26901737e+06, 8.88611052e+06,

       2.41549528e+07, 6.56599691e+07, 1.78482301e+08, 4.85165195e+08,

       1.31881573e+09, 3.58491285e+09, 9.74480345e+09, 2.64891221e+10,

       7.20048993e+10, 1.95729609e+11, 5.32048241e+11, 1.44625706e+12,

       3.93133430e+12, 1.06864746e+13, 2.90488497e+13, 7.89629602e+13,

       2.14643580e+14, 5.83461743e+14, 1.58601345e+15, 4.31123155e+15,

       1.17191424e+16, 3.18559318e+16, 8.65934004e+16, 2.35385267e+17,

       6.39843494e+17, 1.73927494e+18, 4.72783947e+18, 1.28516001e+19,

       3.49342711e+19, 9.49611942e+19, 2.58131289e+20, 7.01673591e+20,

       1.90734657e+21, 5.18470553e+21, 1.40934908e+22])

>>> def poisson_proba(k, lambda_):

...     try:

...         k = int(input("Введите число: "))

...         lambda_ = float(input("Введите число: "))

...     except zerodivisionerror:

...         return

...     s = (lambda_ ** k) * (np.exp(-lambda_)) / np.math.factorial(k)

...     return s

...

>>> print(poisson_proba(k, lambda_))
```

```
Введите число: 51

Введите число: 51

0.055771889130539744

>>>
```