In [155]:
```
import pandas as pd
import numpy as np
```

In [156]:
```
x = [6.45, 9.67, 8.27, 7.22, 2.65, 4.85, 1.63, 9.64, 10.16, 1.23, 6.39, 2.02, 8.17, 6.47, 8.76, 3.03, 3.69, 5.52, 1.12
```

In [157]:
```
x = np.array(x)
```

In [158]:
```
x
```

Out[158]:
```
array([ 6.45,  9.67,  8.27,  7.22,  2.65,  4.85,  1.63,  9.64, 10.16,
        1.23,  6.39,  2.02,  8.17,  6.47,  8.76,  3.03,  3.69,  5.52,
        1.12,  9.54, 10.15, 10.97,  1.59,  2.37,  1.4 ,  7.44,  3.37,
        8.08,  7.52, 10.22,  8.7 ,  9.66,  4.18,  3.61,  5.87,  3.77,
        1.92,  9.18,  5.5 ,  7.59])
```

In [159]:
```
y = [25.07, 30.48, 27.4, 25.75, 15.55, 23.01, 9.21, 25.63, 29.93, 8.01, 22.3, 13.97, 22.58, 26.55, 28.92, 18.97, 16.8,
y = np.array(y)
```

In [160]:
```


# Создание словаря
data_dict = dict(zip(x, y))

# Сортировка словаря по возрастанию ключей (x)
sorted_data = dict(sorted(data_dict.items()))

# Преобразование обратно в массивы
x = list(sorted_data.keys())
y = list(sorted_data.values())
```

In [162]:
```
x
```

Out[162]:
```
[1.12,
 1.23,
 1.4,
 1.59,
 1.63,
 1.92,
 2.02,
 2.37,
 2.65,
 3.03,
 3.37,
 3.61,
 3.69,
 3.77,
 4.18,
 4.85,
 5.5,
 5.52,
 5.87,
 6.39,
 6.45,
 6.47,
 7.22,
 7.44,
 7.52,
 7.59,
 8.08,
 8.17,
 8.27,
 8.7,
 8.76,
 9.18,
 9.54,
 9.64,
 9.66,
 9.67,
 10.15,
 10.16,
 10.22,
 10.97]
```

In [161]:
```
df = pd.DataFrame({'X': x, 'Y': y})
```

In [ ]:
```

```

In [163]:
```
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import linregress
from scipy.optimize import curve_fit



# 1) Построение диаграммы рассеяния (scatter plot)
plt.scatter(x, y)
plt.title("Диаграмма рассеяния")
plt.xlabel("X")
plt.ylabel("Y")
plt.show()

# 2) Линейная регрессия
# Вычисление коэффициентов линейной регрессии методом наименьших квадратов
A = np.vstack([x, np.ones_like(x)]).T
m, c = np.linalg.lstsq(A, y, rcond=None)[0]

# Построение диаграммы рассеяния
plt.scatter(x, y, marker='o', color='blue', label='Наблюдаемые данные')

# Построение линии регрессии
plt.plot(x, m*x + c, 'r', label=f'Линейная регрессия: y = {m:.2f}x + {c:.2f}')

# Добавление заголовка и меток осей
plt.title('Диаграмма рассеяния и линейная регрессия')
plt.xlabel('X')
plt.ylabel('Y')

# Добавление легенды
plt.legend()

# Отображение графика
plt.show()


# 3) Квадратичная регрессия

# Вычисление коэффициентов квадратичной регрессии методом наименьших квадратов
A = np.vstack([x**2, x, np.ones_like(x)]).T
```

```
a, b, c = np.linalg.lstsq(A, y, rcond=None)[0]

# Построение диаграммы рассеяния
plt.scatter(x, y, marker='o', color='blue', label='Наблюдаемые данные')

# Построение квадратичной регрессии
x_range = np.linspace(min(x), max(x), 100)
y_pred = a * x_range**2 + b * x_range + c
plt.plot(x_range, y_pred, 'r', label=f'Квадратичная регрессия: y = {a:.2f}x^2 + {b:.2f}x + {c:.2f}')

# Добавление заголовка и меток осей
plt.title('Диаграмма рассеяния и квадратичная регрессия')
plt.xlabel('X')
plt.ylabel('Y')

# Добавление легенды
plt.legend()

# Отображение графика
plt.show()




# 4) Кубическа регрессия

# Вычисление коэффициентов кубической регрессии методом наименьших квадратов
A = np.vstack([x**3, x**2, x, np.ones_like(x)]).T
a, b, c, d = np.linalg.lstsq(A, y, rcond=None)[0]

# Построение диаграммы рассеяния
plt.scatter(x, y, marker='o', color='blue', label='Наблюдаемые данные')

# Построение кубической регрессии
x_range = np.linspace(min(x), max(x), 100)
y_pred = a * x_range**3 + b * x_range**2 + c * x_range + d
plt.plot(x_range, y_pred, 'r', label=f'Кубическая регрессия: y = {a:.2f}x^3 + {b:.2f}x^2 + {c:.2f}x + {d:.2f}')

# Добавление заголовка и меток осей
plt.title('Диаграмма рассеяния и кубическая регрессия')
plt.xlabel('X')
plt.ylabel('Y')
```

```
# Добавление легенды
plt.legend()

# Отображение графика
plt.show()


# 5)Логарифмическа регрессия

# Преобразование x с учетом логарифма
x_log = np.log(x)

# Вычисление коэффициентов логарифмической регрессии методом наименьших квадратов
A = np.vstack([x_log, np.ones_like(x)]).T
a, b = np.linalg.lstsq(A, y, rcond=None)[0]

# Построение диаграммы рассеяния
plt.scatter(x, y, marker='o', color='blue', label='Наблюдаемые данные')

# Построение логарифмической регрессии
x_range = np.linspace(min(x), max(x), 100)
y_pred = a * np.log(x_range) + b
plt.plot(x_range, y_pred, 'r', label=f'Логарифмическая регрессия: y = {a:.2f}ln(x) + {b:.2f}')

# Добавление заголовка и меток осей
plt.title('Диаграмма рассеяния и логарифмическая регрессия')
plt.xlabel('X')
plt.ylabel('Y')

# Добавление легенды
plt.legend()

# Отображение графика
plt.show()

#6) Показательная регрессия

# Преобразование y с учетом логарифма
y_log = np.log(y)

# Вычисление коэффициентов показательной регрессии методом наименьших квадратов
```

```
A = np.vstack([x, np.ones_like(x)]).T
b, a = np.linalg.lstsq(A, y_log, rcond=None)[0]

# Построение диаграммы рассеяния
plt.scatter(x, y, marker='o', color='blue', label='Наблюдаемые данные')

# Построение показательной регрессии
x_range = np.linspace(min(x), max(x), 100)
y_pred = np.exp(a * x_range + b)
plt.plot(x_range, y_pred, 'r', label=f'Показательная регрессия: y = {np.exp(a):.2f}e^({b:.2f}x)')
# Настройка масштаба осей
plt.axis([min(x), max(x), min(y), max(y)])

# Добавление заголовка и меток осей
plt.title('Диаграмма рассеяния и показательная регрессия')
plt.xlabel('X')
plt.ylabel('Y')

# Добавление легенды
plt.legend()

# Отображение графика
plt.show()
```


Диаграмма рассеяния
30
25
20
15
Y

In [125]:
```
import numpy as np
import statsmodels.api as sm
import matplotlib.pyplot as plt




# Добавляем константу к x для учёта свободного члена в уравнении регрессии
X = sm.add_constant(x)

# Линейная регрессия
model_linear = sm.OLS(y, X).fit()
linear_fit = model_linear.predict(X)

# Оценки остаточной дисперсии и коэффициента детерминации
residual_variance_linear = np.var(model_linear.resid)
r_squared_linear = model_linear.rsquared

# Печать результатов для линейной регрессии
print("Линейная регрессия:")
print(f"Остаточная дисперсия: {residual_variance_linear:.4f}")
print(f"Коэффициент детерминации (R^2): {r_squared_linear:.4f}")
print(model_linear.summary())

# Продолжаем аналогично для квадратичной, логарифмической и показательной регрессии.
# Замените уравнения регрессии и повторите шаги.

# Квадратичная регрессия
X_quad = sm.add_constant(np.column_stack((x, x**2)))
model_quad = sm.OLS(y, X_quad).fit()
quadratic_fit = model_quad.predict(X_quad)

# Оценки остаточной дисперсии и коэффициента детерминации
residual_variance_quad = np.var(model_quad.resid)
r_squared_quad = model_quad.rsquared

# Печать результатов для квадратичной регрессии
print("\nКвадратичная регрессия:")
print(f"Остаточная дисперсия: {residual_variance_quad:.4f}")
print(f"Коэффициент детерминации (R^2): {r_squared_quad:.4f}")
print(model_quad.summary())
```

```
X_cubic = sm.add_constant(np.column_stack((x, x**2, x**3)))
model_cubic = sm.OLS(y, X_cubic).fit()
cubic_fit = model_cubic.predict(X_cubic)

# Оценки остаточной дисперсии и коэффициента детерминации
residual_variance_cubic = np.var(model_cubic.resid)
r_squared_cubic = model_cubic.rsquared

# Печать результатов для кубической регрессии
print("\nКубическая регрессия:")
print(f"Остаточная дисперсия: {residual_variance_cubic:.4f}")
print(f"Коэффициент детерминации (R^2): {r_squared_cubic:.4f}")
print(model_cubic.summary())

# Повторите те же шаги для логарифмической и показательной регрессии.

# Логарифмическая регрессия
X_log = sm.add_constant(np.column_stack((np.log(x), x)))
model_log = sm.OLS(y, X_log).fit()
logarithmic_fit = model_log.predict(X_log)

# Оценки остаточной дисперсии и коэффициента детерминации
residual_variance_log = np.var(model_log.resid)
r_squared_log = model_log.rsquared

# Печать результатов для логарифмической регрессии
print("\nЛогарифмическая регрессия:")
print(f"Остаточная дисперсия: {residual_variance_log:.4f}")
print(f"Коэффициент детерминации (R^2): {r_squared_log:.4f}")
print(model_log.summary())

# Показательная регрессия
X_exp = sm.add_constant(np.column_stack((np.exp(x), x)))
model_exp = sm.OLS(y, X_exp).fit()
exponential_fit = model_exp.predict(X_exp)

# Оценки остаточной дисперсии и коэффициента детерминации
residual_variance_exp = np.var(model_exp.resid)
r_squared_exp = model_exp.rsquared

# Печать результатов для показательной регрессии
print("\nПоказательная регрессия:")
```

```
print(f"Остаточная дисперсия: {residual_variance_exp:.4f}")
print(f"Коэффициент детерминации (R^2): {r_squared_exp:.4f}")
print(model_exp.summary())

# Проверка значимости модели в целом
f_stat_linear, p_value_linear, _ = model_linear.compare_f_test(model_quad)
f_stat_quad, p_value_quad, _ = model_quad.compare_f_test(model_log)
f_stat_log, p_value_log, _ = model_log.compare_f_test(model_exp)

# Печать результатов F-теста
print("\nF-тест для модели в целом:")
print(f"Линейная vs. Квадратичная: F = {f_stat_linear:.4f}, p-value = {p_value_linear:.4f}")
print(f"Квадратичная vs. Логарифмическая: F = {f_stat_quad:.4f}, p-value = {p_value_quad:.4f}")
print(f"Логарифмическая vs. Показательная: F = {f_stat_log:.4f}, p-value = {p_value_log:.4f}")
```

```
Линейная регрессия:
Остаточная дисперсия: 6.7972
Коэффициент детерминации (R^2): 0.8573
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.857
Model:                            OLS   Adj. R-squared:                  0.854
Method:                 Least Squares   F-statistic:                     228.4
Date:                Mon, 11 Dec 2023   Prob (F-statistic):           1.18e-17
Time:                        15:41:03   Log-Likelihood:                -95.088
No. Observations:                  40   AIC:                             194.2
Df Residuals:                      38   BIC:                             197.6
Df Model:                           1
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          9.5504      0.926     10.310      0.000       7.675      11.426
x1             2.0793      0.138     15.111      0.000       1.801       2.358
==============================================================================
Omnibus:                        1.957   Durbin-Watson:                   1.734
Prob(Omnibus):                  0.376   Jarque-Bera (JB):                1.327
Skew:                          -0.184   Prob(JB):                        0.515
Kurtosis:                       2.187   Cond. No.                         15.0
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Квадратичная регрессия:
Остаточная дисперсия: 3.0375
Коэффициент детерминации (R^2): 0.9362
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.936
Model:                            OLS   Adj. R-squared:                  0.933
Method:                 Least Squares   F-statistic:                     271.7
Date:                Mon, 11 Dec 2023   Prob (F-statistic):           7.65e-23
Time:                        15:41:03   Log-Likelihood:                -78.978
No. Observations:                  40   AIC:                             164.0
Df Residuals:                      37   BIC:                             169.0
```

```
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          3.2311      1.125      2.872      0.007       0.951       5.511
x1             5.1047      0.457     11.178      0.000       4.179       6.030
x2            -0.2604      0.038     -6.767      0.000      -0.338      -0.182
==============================================================================
Omnibus:                        1.937   Durbin-Watson:                   2.315
Prob(Omnibus):                  0.380   Jarque-Bera (JB):                0.989
Skew:                          -0.302   Prob(JB):                        0.610
Kurtosis:                       3.478   Cond. No.                         246.
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Кубическая регрессия:
Остаточная дисперсия: 2.4398
Коэффициент детерминации (R^2): 0.9488
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.949
Model:                            OLS   Adj. R-squared:                  0.945
Method:                 Least Squares   F-statistic:                     222.3
Date:                Mon, 11 Dec 2023   Prob (F-statistic):           2.80e-23
Time:                        15:41:03   Log-Likelihood:                -74.596
No. Observations:                  40   AIC:                             157.2
Df Residuals:                      36   BIC:                             163.9
Df Model:                           3
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const         -1.4323      1.874     -0.764      0.450      -5.232       2.368
x1             8.7785      1.305      6.728      0.000       6.132      11.425
x2            -0.9871      0.247     -3.993      0.000      -1.488      -0.486
x3             0.0409      0.014      2.970      0.005       0.013       0.069
==============================================================================
Omnibus:                        1.443   Durbin-Watson:                   2.045
Prob(Omnibus):                  0.486   Jarque-Bera (JB):                0.936
```

```
Skew:                          -0.374   Prob(JB):                        0.626
Kurtosis:                       3.051   Cond. No.                     4.74e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 4.74e+03. This might indicate that there are
strong multicollinearity or other numerical problems.

Логарифмическая регрессия:
Остаточная дисперсия: 2.4035
Коэффициент детерминации (R^2): 0.9496
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.950
Model:                            OLS   Adj. R-squared:                  0.947
Method:                 Least Squares   F-statistic:                     348.2
Date:                Mon, 11 Dec 2023   Prob (F-statistic):           1.01e-24
Time:                        15:41:03   Log-Likelihood:                -74.296
No. Observations:                  40   AIC:                             154.6
Df Residuals:                      37   BIC:                             159.7
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          5.5929      0.737      7.589      0.000       4.100       7.086
x1            11.9020      1.447      8.224      0.000       8.970      14.834
x2            -0.4452      0.318     -1.400      0.170      -1.089       0.199
==============================================================================
Omnibus:                        3.003   Durbin-Watson:                   2.091
Prob(Omnibus):                  0.223   Jarque-Bera (JB):                2.059
Skew:                          -0.539   Prob(JB):                        0.357
Kurtosis:                       3.269   Cond. No.                         43.0
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Показательная регрессия:
Остаточная дисперсия: 5.4332
Коэффициент детерминации (R^2): 0.8860
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                      y   R-squared:                       0.886
Model:                            OLS   Adj. R-squared:                  0.880
Method:                 Least Squares   F-statistic:                     143.7
Date:                Mon, 11 Dec 2023   Prob (F-statistic):           3.59e-18
Time:                        15:41:03   Log-Likelihood:                -90.608
No. Observations:                  40   AIC:                             187.2
Df Residuals:                      37   BIC:                             192.3
Df Model:                           2
Covariance Type:            nonrobust
==============================================================================
                 coef    std err          t      P>|t|      [0.025      0.975]
------------------------------------------------------------------------------
const          8.3167      0.932      8.926      0.000       6.429      10.205
x1            -0.0001   4.56e-05     -3.048      0.004      -0.000   -4.66e-05
x2             2.4249      0.169     14.388      0.000       2.083       2.766
==============================================================================
Omnibus:                        0.342   Durbin-Watson:                   2.141
Prob(Omnibus):                  0.843   Jarque-Bera (JB):                0.463
Skew:                          -0.192   Prob(JB):                        0.793
Kurtosis:                       2.639   Cond. No.                     3.16e+04
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 3.16e+04. This might indicate that there are
strong multicollinearity or other numerical problems.

F-тест для модели в целом:
Линейная vs. Квадратичная: F = 21.0187, p-value = nan
Квадратичная vs. Логарифмическая: F = -inf, p-value = nan
Логарифмическая vs. Показательная: F = inf, p-value = nan

C:\Users\Семён\AppData\Local\Packages\PythonSoftwareFoundation.Python.3.10_qbz5n2kfra8p0\LocalCache\local-packages\Py
thon310\site-packages\statsmodels\regression\linear_model.py:2283: RuntimeWarning: divide by zero encountered in doub
le_scalars
  f_value = (ssr_restr - ssr_full) / df_diff / ssr_full * df_full
```

In [164]:
```
print("Линейная регрессия:")
print(f"Остаточная дисперсия: {residual_variance_linear:.4f}")
print(f"Коэффициент детерминации (R^2): {r_squared_linear:.4f}")
print("\nКвадратичная регрессия:")
print(f"Остаточная дисперсия: {residual_variance_quad:.4f}")
print(f"Коэффициент детерминации (R^2): {r_squared_quad:.4f}")
print("\nКубическая регрессия:")
print(f"Остаточная дисперсия: {residual_variance_cubic:.4f}")
print(f"Коэффициент детерминации (R^2): {r_squared_cubic:.4f}")
print("\nЛогарифмическая регрессия:")
print(f"Остаточная дисперсия: {residual_variance_log:.4f}")
print(f"Коэффициент детерминации (R^2): {r_squared_log:.4f}")
print("\nПоказательная регрессия:")
print(f"Остаточная дисперсия: {residual_variance_exp:.4f}")
print(f"Коэффициент детерминации (R^2): {r_squared_exp:.4f}")
```

```
Линейная регрессия:
Остаточная дисперсия: 6.7972
Коэффициент детерминации (R^2): 0.8573

Квадратичная регрессия:
Остаточная дисперсия: 3.0375
Коэффициент детерминации (R^2): 0.9362

Кубическая регрессия:
Остаточная дисперсия: 2.4398
Коэффициент детерминации (R^2): 0.9488

Логарифмическая регрессия:
Остаточная дисперсия: 2.4035
Коэффициент детерминации (R^2): 0.9496

Показательная регрессия:
Остаточная дисперсия: 5.4332
Коэффициент детерминации (R^2): 0.8860
```

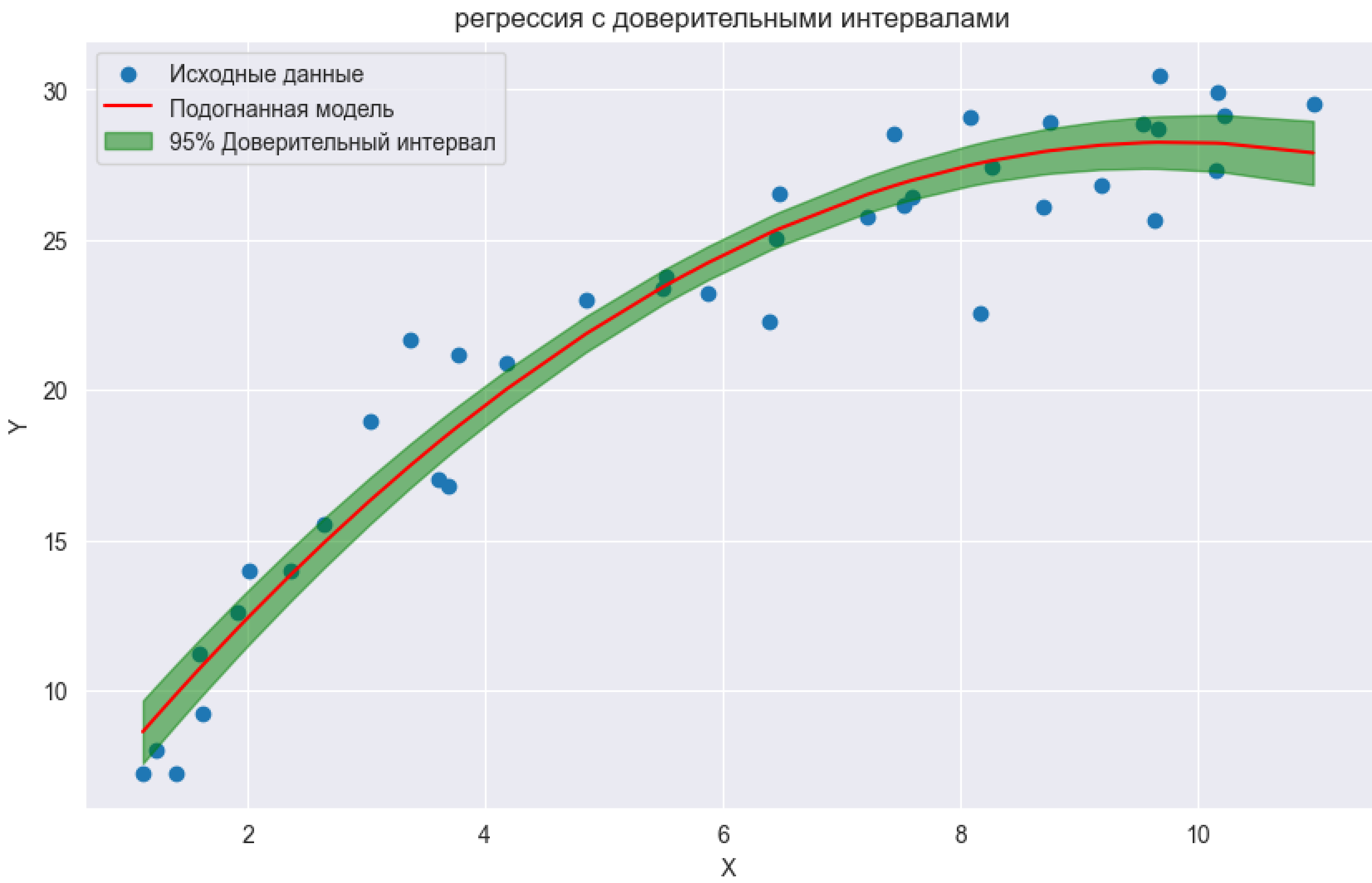
Вывод:

Коэффициент детерминации (R^2): Чем выше R^2, тем лучше модель объясняет изменчивость данных. С точки зрения R^2, логарифмическая регрессия имеет наилучшую предсказательную способность (R^2 = 0.9496), за ней следует квадратичная регрессия (R^2 = 0.9362). Линейная регрессия также хороша, но у нее R^2 чуть меньше (R^2 = 0.8573), а показательная регрессия хоть и хороша, но уступает по R^2 двум предыдущим (R^2 = 0.8860).

Остаточная дисперсия: Меньшая остаточная дисперсия указывает на то, что остатки модели более сгруппированы вокруг средней линии. С этой точки зрения, логарифмическая регрессия (2.4035) также имеет наилучшую производительность, за ней идет

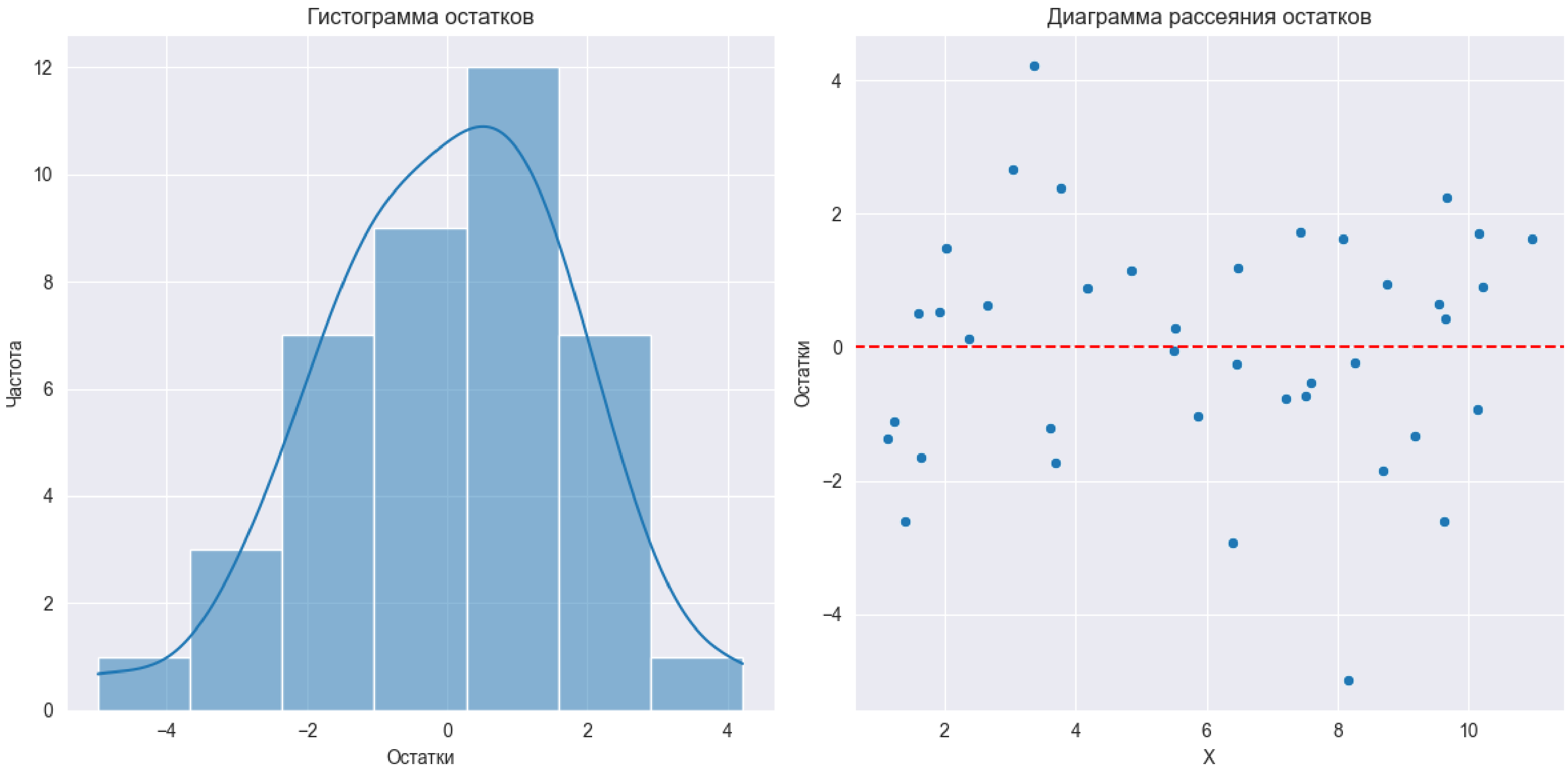
In [165]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from scipy.optimize import curve_fit #метод наименьших квадратов
from scipy import stats
from scipy.stats import t
import numpy as np
def square(x,*params):
    a=params[0]
    b=params[1]
    c=params[2]
    return a*x*x + b*x + c
```

In [166]:
```
coef, cov_matrix = curve_fit(square, df['X'], df['Y'], p0=[1,1,1,1])
residuals_cube = df['Y'] - square(df['X'], *coef) #Остатки
mse_cube = np.mean(residuals_cube**2) #Оценка остат дисперсии

se_pred = np.sqrt(mse_cube * (1/df['X'].size + (df['X'] - np.mean(df['X']))**2 / np.sum((df['X'] - np.mean(df['X']))**

alpha = 0.05
t_crit = t.ppf(1 - alpha/2, df=df['X'].size-2)

ci_lower = square(df['X'], *coef) - t_crit * se_pred
ci_upper = square(df['X'], *coef) + t_crit * se_pred

plt.figure(figsize=(10, 6))
plt.scatter(df['X'], df['Y'], label='Исходные данные')
plt.plot(df['X'], square(df['X'], *coef), color='red', label='Подогнанная модель')
plt.fill_between(df['X'], ci_lower, ci_upper, color='green', alpha=0.5, label='95% Доверительный интервал')
plt.title(' регрессия с доверительными интервалами')
plt.xlabel('X')
plt.ylabel('Y')
plt.legend()
plt.grid(True)

# Вывод графика
plt.show()
```



принимаем гипотезу о нормальности, отвергаем гипотезу о однородности

In [167]:
```
plt.figure(figsize=(12, 6))
plt.subplot(1, 2, 1)
sns.histplot(residuals_cube, kde=True)
plt.title('Гистограмма остатков')
plt.xlabel('Остатки')
plt.ylabel('Частота')

plt.subplot(1, 2, 2)
sns.scatterplot(x=df['X'], y=residuals_cube)
plt.axhline(y=0, color='r', linestyle='--')
plt.title('Диаграмма рассеяния остатков')
plt.xlabel('X')
plt.ylabel('Остатки')

plt.tight_layout()
plt.show()
# Тест на нормальность остатков (Шапиро-Уилка)
stat, p_value_shapiro = shapiro(residuals)
print(f'Шапиро-Уилка тест на нормальность: Statistic={stat:.4f}, p-value={p_value_shapiro:.4f}')

# Тест Бартлетта на гомоскедастичность
stat, p_value_bartlett = bartlett(residuals, x)
print(f'Тест Бартлетта на гомоскедастичность: Statistic={stat:.4f}, p-value={p_value_bartlett:.40f}')
```




```
Шапиро-Уилка тест на нормальность: Statistic=0.9585, p-value=0.1485
Тест Бартлетта на гомоскедастичность: Statistic=157.8211, p-value=0.0000000000000000000000000000000000033863
```

In [ ]: