Министерство науки и образования РФ

Федеральное государственное бюджетное учреждение

высшего образования

**«Тверской государственный технический университет»**

(ТвГТУ)

Кафедра программного обеспечения

**Отчет по лабораторной работе №1**

по дисциплине: «Компьютерные системы моделирования»

Тема: «Построение аналитической модели по текстовому описанию задачи и по результатам экспериментов»

|  |
| --- |
| Выполнил:  студент группы  Б.ПИН.РИС-22.06  Иванов А.М. |
| Проверила:  старший преподаватель кафедры ПО  Корнеева Е.И. |

Тверь 2025

**Отчет по задаче 1**

**Постановка задачи**

На швейной фабрике для изготовления четырех видов изделий используется ткань трех артикулов. Для каждого вида изделия известны:

* нормы расхода ткани (м²),
* общее количество ткани каждого артикула на складе (м²),
* стоимость одного изделия данного вида (руб.).

Цель задачи: определить, сколько изделий каждого вида необходимо произвести, чтобы стоимость изготовленной продукции была максимальной.

**Данные:**



**Построение математической модели**

**Целевая функция:**  
Максимизация общей стоимости продуфкции:

F(x) = 9x\_1 + 6x\_2 + 4x\_3 + 7x\_4

**Ограничения:**

1. Ограничение по ткани I:

x\_1 + 2x\_3 + x\_4 ≤ 180

1. Ограничение по ткани II:

x\_2 + 3x\_3 + 2x\_4 ≤ 210

1. Ограничение по ткани III:

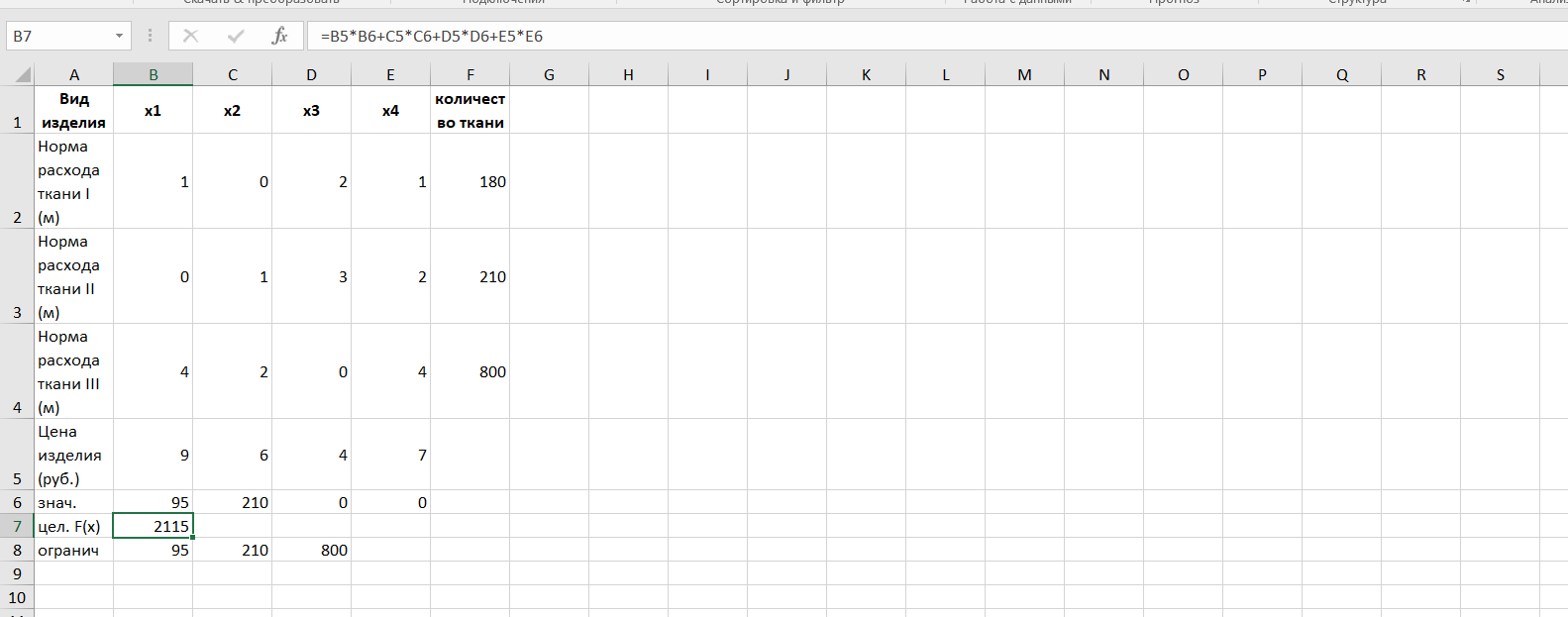
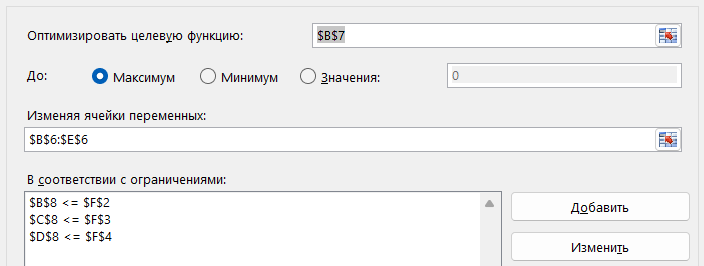
4x\_1 + 2x\_2 + 4x\_4 ≤ 800

**Переменные:**  
x\_1, x\_2, x\_3, x\_4 ≥ 0, целочисленные.

**Решение задачи**

**Решение в MS Excel**

**Модель задачи в Excel:**

**Результат решения:**



Т.е. максимальная прибыль составит 2115 руб. при выпуске 95 изделий типа 1, и 210 изделий типа 2

**Решение с помощью Python**

Для решения задачи использовалась библиотека pulp.

Код:

from pulp import LpMaximize, LpProblem, LpVariable

model = LpProblem(name="maximize-production-cost", sense=LpMaximize)

x1 = LpVariable(name="x1", lowBound=0, cat="Integer")

x2 = LpVariable(name="x2", lowBound=0, cat="Integer")

x3 = LpVariable(name="x3", lowBound=0, cat="Integer")

x4 = LpVariable(name="x4", lowBound=0, cat="Integer")

model += 9 \* x1 + 6 \* x2 + 4 \* x3 + 7 \* x4, "Total cost"

model += x1 + 2 \* x3 + x4 <= 180, "Constraint for fabric I"

model += x2 + 3 \* x3 + 2 \* x4 <= 210, "Constraint for fabric II"

model += 4 \* x1 + 2 \* x2 + 4 \* x4 <= 800, "Constraint for fabric III"

status = model.solve()

print(f"Status: {model.status}")

print(f"Optimal production:")

print(f"x1 (item 1): {x1.value()}")

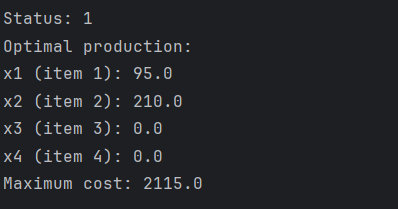
print(f"x2 (item 2): {x2.value()}")

print(f"x3 (item 3): {x3.value()}")

print(f"x4 (item 4): {x4.value()}")

print(f"Maximum cost: {model.objective.value()}")

**Результат:**



* x\_1 = 95 (изделия вида 1),
* x\_2 = 210 (изделия вида 2),
* x\_3 = 0 (изделия вида 3),
* x\_4 = 0 (изделия вида 4),
* Максимальная стоимость продукции: F(x) = 2115 руб.

**Основные результаты**

1. **Оптимальная стратегия производства:**
   * Произвести 95 изделий вида 1,
   * Произвести 210 изделий вида 2,
   * Изделия видов 3 и 4 не производить.
2. **Максимальная стоимость продукции:** 2115 руб.

**Вывод**

Поставленная задача успешно решена двумя способами: с использованием Excel и Python. Результаты совпадают, что подтверждает корректность модели. Использование Python обеспечивает автоматизацию и масштабируемость для решения более сложных задач.

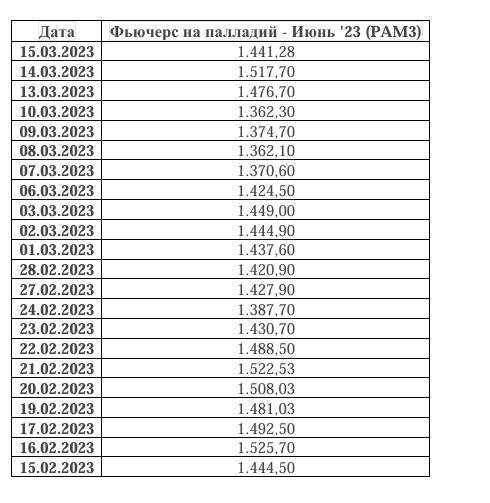
**Отчет по задаче 2.**

**Постановка задачи**

По данным о котировках с сайта **Investing.com** за 15.03.2023 необходимо:

1. Построить аналитическую модель на основе данных.
2. Создать таблицу в MS Excel или Google Sheets и построить линии тренда: линейную, полиномиальную, степенную и логарифмическую. Вывести на графике уравнение и коэффициент детерминации R^2.
3. Построить аналогичные графики с использованием Python и библиотек matplotlib/seaborn. Сравнить результаты и сделать выводы.

**Данные**:



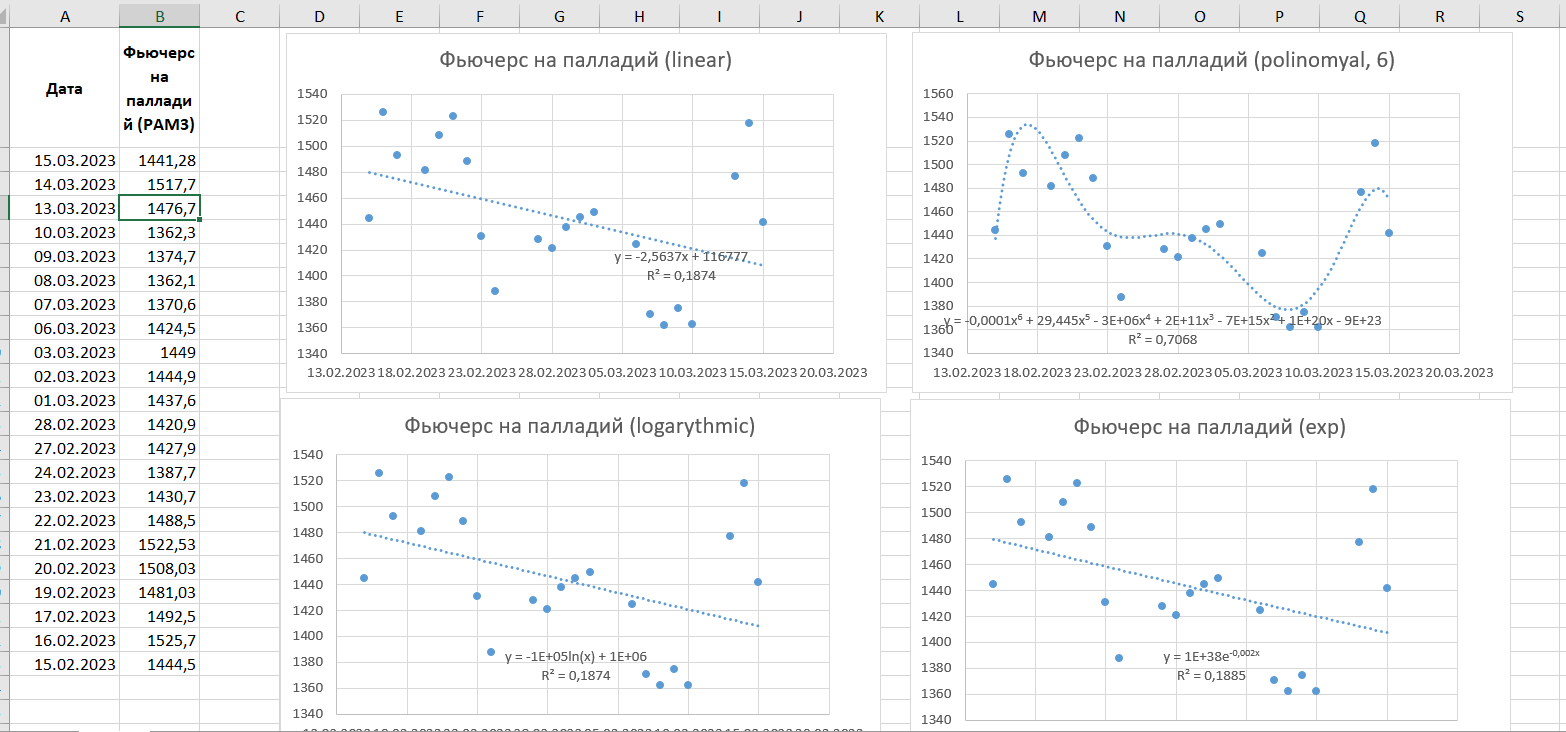
**Основные результаты**

**Excel**

В Excel были построены линии тренда с использованием встроенных функций:

* Линейная, полиномиальная, степенная и логарифмическая аппроксимации.
* Для каждой модели отображены уравнение тренда и R^2.

Скриншоты графиков и уравнений:



**Python**

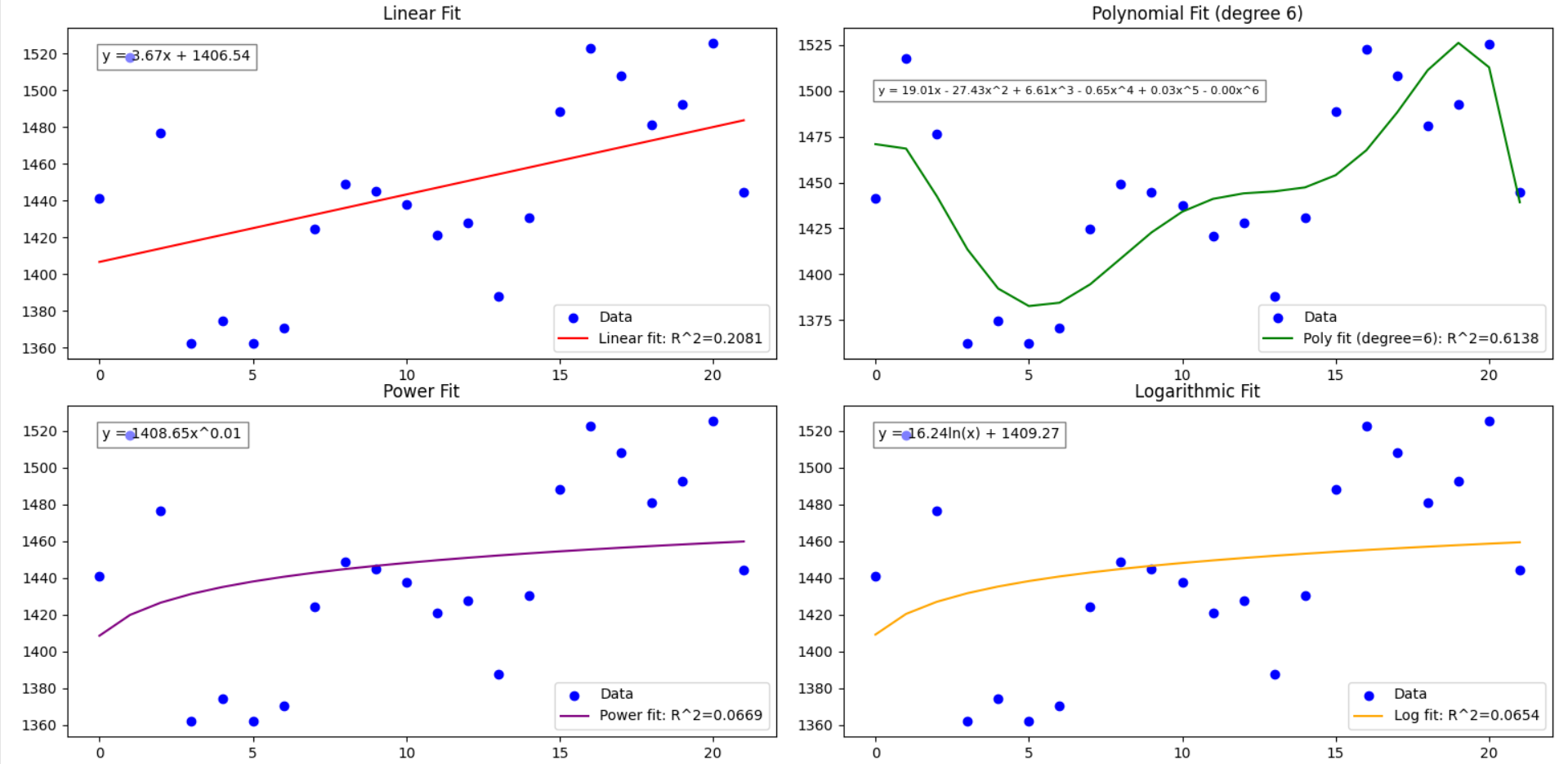
В Python для анализа использовались библиотеки numpy, scikit-learn, matplotlib и функции:

* Линейная регрессия (LinearRegression).
* Полиномиальная регрессия с использованием степени k = 6.
* Степенная модель с функцией y=a⋅x.
* Логарифмическая модель y=a⋅ln(x)+b.

**Код**

def plot\_second\_task():  
  
 def polynomial\_model(x, degree):  
 poly = PolynomialFeatures(degree=degree)  
 x\_poly = poly.fit\_transform(x.reshape(-1, 1))  
 model = LinearRegression().fit(x\_poly, prices)  
 return model.predict(x\_poly), model  
  
 def power\_law(x, a, b):  
 return a \* x \*\* b  
  
 def log\_model(x, a, b):  
 return a \* np.log(x) + b  
  
 # Линейная модель  
 linear\_model = LinearRegression().fit(x.reshape(-1, 1), prices)  
 linear\_pred = linear\_model.predict(x.reshape(-1, 1))  
  
 # Полиномиальная модель  
 degree = 6  
 poly\_pred, poly\_model = polynomial\_model(x, degree)  
  
 # Степенная модель  
 params\_power, \_ = curve\_fit(power\_law, x + 1, prices, maxfev=10000)  
 power\_pred = power\_law(x + 1, \*params\_power)  
  
 # Логарифмическая модель  
 params\_log, \_ = curve\_fit(log\_model, x + 1, prices)  
 log\_pred = log\_model(x + 1, \*params\_log)  
  
 r2\_linear = r2\_score(prices, linear\_pred)  
 r2\_poly = r2\_score(prices, poly\_pred)  
 r2\_power = r2\_score(prices, power\_pred)  
 r2\_log = r2\_score(prices, log\_pred)  
  
 fig, axs = plt.subplots(2, 2, figsize=(14, 10))  
  
 # Линейная модель  
 axs[0, 0].scatter(x, prices, color='blue', label='Data')  
 axs[0, 0].plot(x, linear\_pred, color='red', label=f'Linear fit: R^2={r2\_linear:.4f}')  
 axs[0, 0].set\_title("Linear Fit")  
 axs[0, 0].legend()  
  
 # Полиномиальная модель  
 axs[0, 1].scatter(x, prices, color='blue', label='Data')  
 axs[0, 1].plot(x, poly\_pred, color='green', label=f'Poly fit (degree={degree}): R^2={r2\_poly:.4f}')  
 axs[0, 1].set\_title(f"Polynomial Fit (degree {degree})")  
 axs[0, 1].legend()  
  
 # Степенная модель  
 axs[1, 0].scatter(x, prices, color='blue', label='Data')  
 axs[1, 0].plot(x, power\_pred, color='purple', label=f'Power fit: R^2={r2\_power:.4f}')  
 axs[1, 0].set\_title("Power Fit")  
 axs[1, 0].legend()  
  
 # Логарифмическая модель  
 axs[1, 1].scatter(x, prices, color='blue', label='Data')  
 axs[1, 1].plot(x, log\_pred, color='orange', label=f'Log fit: R^2={r2\_log:.4f}')  
 axs[1, 1].set\_title("Logarithmic Fit")  
 axs[1, 1].legend()  
  
 axs[0, 0].text(0.05, 0.9, f"y = {linear\_model.coef\_[0]:.2f}x + {linear\_model.intercept\_:.2f}",  
 transform=axs[0, 0].transAxes, fontsize=10, bbox=dict(facecolor='white', alpha=0.5))  
  
 def format\_polynomial(coefficients):  
 terms = []  
 for i, coef in enumerate(coefficients):  
 if abs(coef) > 1e-6:  
 if i == 0:  
 terms.append(f"{coef:.2f}")  
 elif i == 1:  
 terms.append(f"{coef:.2f}x")  
 else:  
 terms.append(f"{coef:.2f}x^{i}")  
 return " + ".join(terms).replace("+ -", "- ")  
  
 poly\_coefficients = poly\_model.coef\_  
  
 poly\_equation = format\_polynomial(poly\_coefficients)  
  
 axs[0, 1].text(0.05, 0.8, f"y = {poly\_equation}",  
 transform=axs[0, 1].transAxes, fontsize=8, bbox=dict(facecolor='white', alpha=0.5))  
  
 axs[1, 0].text(0.05, 0.9, f"y = {params\_power[0]:.2f}x^{params\_power[1]:.2f}",  
 transform=axs[1, 0].transAxes, fontsize=10, bbox=dict(facecolor='white', alpha=0.5))  
 axs[1, 1].text(0.05, 0.9, f"y = {params\_log[0]:.2f}ln(x) + {params\_log[1]:.2f}",  
 transform=axs[1, 1].transAxes, fontsize=10, bbox=dict(facecolor='white', alpha=0.5))  
  
 plt.tight\_layout()  
 plt.show()

Итоговые графики:



**Выводы**

**Различия между Excel и Python**  
Графики в Excel и Python могут отличаться из-за следующих факторов:

1. **Методы аппроксимации**: Excel использует собственные алгоритмы для расчета параметров тренда, которые могут отличаться от подходов в Python
2. **Числовая стабильность**: При расчетах коэффициентов уравнений и R^2 в Python могут использоваться более точные алгоритмы или настройки, влияющие на результаты.
3. **Преобразование данных**: В Python, чтобы избежать ошибок при работе со степенной и логарифмической аппроксимацией, к данным добавляется смещение (x+1). Это может влиять на параметры модели.

**Коэффициент детерминации (R^2)**

R^2 показывает, насколько хорошо модель объясняет вариацию данных. Значения R^2 для различных моделей из Python и Excel близки, но могут отличаться из-за разницы в расчетах.

**Вывод по моделям**

На основе коэффициента R^2, наиболее подходящей для данных является полиномиальная модель со степенью 6.

Линейная модель в Excel имеет наименьшее R^2, что говорит о её низкой точности для описания данных. В python же логарифмическая модель показала еще менее точный результат.

**Общий вывод по лабораторной работе**

В ходе выполнения лабораторной работы были решены две задачи с использованием различных методов анализа и моделирования.

**Первая задача** касалась оптимизации производства на швейной фабрике. Были успешно построены и решены математические модели для максимизации прибыли.

Решение задачи выполнено двумя способами: с использованием MS Excel и библиотеки PuLP в Python.

Результаты обоих подходов совпали, что подтверждает корректность математической модели и её реализации.

Оптимальная стратегия производства определена, а максимальная стоимость продукции составила 2115 руб.

**Вторая задача** была направлена на анализ котировок и построение аналитических моделей с помощью Excel и Python.

Построены линии тренда с использованием различных типов аппроксимации (линейная, полиномиальная, степенная, логарифмическая).

В Python и Excel получены близкие, но не идентичные результаты из-за различий в методах расчёта и обработки данных.

Полиномиальная модель шестой степени оказалась наиболее подходящей для описания данных по коэффициенту детерминации R^2.

Различия в точности между платформами подчёркивают важность выбора инструмента в зависимости от задачи.

**Итог:**  
Работа показала, что применение математических моделей и современных инструментов анализа данных позволяет эффективно решать задачи оптимизации и анализа. Использование Python предоставляет более широкие возможности для автоматизации, обработки данных и построения сложных моделей.

**Ссылка на реализацию (github):**

https://github.com/ghimik/Computer-Simulating-Systems-Labs