**Реферат**

Отчёт по производственной проектно-технологической практике содержит: 18 с., 3 рисунка, 3 источника, 1 приложение.

ТЕПЛОПОТЕРИ, РАСЧЕТ ТЕПЛОПОТЕЛЬ, PYTHON, TKINTER.

Целью данной работы является разработка программы для расчета теплопотерь здания. В рамках проекта будет создана математическая модель, позволяющая точно оценить теплопотери, и реализовано десктоп-приложение на языке Python для выполнения этих расчетов.

В результате выполнения работы разработано приложение, решающее поставленную задачу. Реализованное приложение позволяет повысить энергоэффективность здания, на котором оно будет использоваться, путем оптимизации тепловых ресурсов и минимизации теплопотерь.

**Содержание**

[Введение 5](#_Toc173848532)

[1. Постановка задачи 6](#_Toc173848533)

[2. Обзор методов расчета теплопотерь 7](#_Toc173848534)

[2.1 Традиционные методы расчета теплопотерь 7](#_Toc173848535)

[2.2 Преимущества использования программного обеспечения для расчета теплопотерь 8](#_Toc173848536)

[3. Разработка и описание структур данных 10](#_Toc173848537)

[3.1 Постановка задачи 10](#_Toc173848538)

[3.2 Использованные структуры данных 10](#_Toc173848539)

[3.3 Решение поставленной задачи 12](#_Toc173848540)

[4. Тестирование программы 14](#_Toc173848541)

[4.1 Разработка плана тестирования 14](#_Toc173848542)

[Заключение 16](#_Toc173848543)

[Список использованных источников 17](#_Toc173848544)

[Приложение А – Листинг приложения 18](#_Toc173848545)

**Введение**

В современном строительстве и эксплуатации зданий расчет теплопотерь играет ключевую роль в обеспечении энергоэффективности и сокращении затрат на отопление. С учетом растущей необходимости в оптимизации энергопотребления, точное и эффективное управление теплопотерями становится критически важным для успешного функционирования строительных и эксплуатационных организаций.

Проектирование программы для расчета теплопотерь здания является фундаментальным этапом в создании информационной системы, которая позволяет управлять всеми аспектами теплотехнического анализа – от учета характеристик строительных конструкций и климатических условий до оптимизации тепловых ресурсов и повышения энергоэффективности зданий.

Целью данной производственной проектно-технологической практике является исследование, разработка и реализация программы для расчета теплопотерь здания с учетом уникальных потребностей и требований данной сферы. В ходе работы будут рассмотрены основные этапы проектирования программного обеспечения.

Данная работа также будет рассматривать различные аспекты программного обеспечения, включая вопросы алгоритмизации, интеграции с другими системами, а также реализацию удобного интерфейса для взаимодействия с пользователем.

Результатом данной работы будет разработанная и функционирующая программа для расчета теплопотерь здания, способствующая оптимизации энергетических процессов.

Эта работа призвана дать понимание важности разработки программного обеспечения для расчета теплопотерь в строительной сфере и предоставить практическую основу для создания информационных систем, способствующих успешному развитию энергетически эффективных решений.

# 1. Постановка задачи

Целью данной производственной проектно-технологической практике является разработка программы для расчета теплопотерь здания. Задачи работы включают в себя:

- Проектирование математической модели: Разработать структуру модели для расчета теплопотерь, определить необходимые параметры и их связи для учета всех факторов, влияющих на теплопотери, таких как конструктивные особенности здания, климатические условия и особенности систем отопления.

- Разработка алгоритмов расчета: Создать алгоритмы и методы расчета теплопотерь, учитывая различные аспекты, такие как теплопроводность материалов, параметры вентиляции и прочие параметры, влияющие на потери тепла.

- Реализация программного обеспечения: Написать код на языке C# для реализации разработанных алгоритмов, включая создание модулей для ввода данных, выполнения расчетов и вывода результатов.

- Создание пользовательского интерфейса: Разработать интерфейс для удобного взаимодействия с программой, обеспечивая возможность ввода данных, настройки параметров и просмотра результатов расчета.

- Обеспечение точности расчетов: Провести тестирование программы, чтобы убедиться в корректности и точности расчетов, выявить и устранить возможные ошибки и неточности.

- Оптимизация производительности: Оптимизировать алгоритмы и код программы для обеспечения быстродействия и эффективного использования ресурсов.

- Документация и обучение пользователей: Подготовить документацию по использованию программы и обучить пользователей работе с ней, чтобы они могли эффективно применять инструмент в своей практике.

1. **Обзор методов расчета теплопотерь**
   1. **Традиционные методы расчета теплопотерь**

Традиционные методы расчета теплопотерь включают в себя использование различных расчетных моделей и методов анализа. Одним из наиболее простых методов является использование стандартных эмпирических формул и таблиц. Эти методы могут быть полезны для предварительных оценок, но имеют определенные ограничения.

Достоинства:

* Простота применения: Эмпирические формулы и таблицы позволяют быстро получить приблизительные результаты без необходимости сложного моделирования.
* Низкие вычислительные затраты: Такие методы не требуют значительных вычислительных ресурсов и могут быть легко реализованы в простых расчетных инструментах.

Недостатки:

* Ограниченная точность: Эмпирические формулы могут не учитывать все факторы, влияющие на теплопотери, и поэтому могут давать неточные результаты.
* Недостаточная гибкость: Эти методы не могут эффективно учитывать сложные конструкции зданий или специфические климатические условия.

Современные методы расчета теплопотерь часто включают использование более сложных математических моделей и компьютерных программ, которые позволяют учитывать различные параметры и получать более точные результаты.

* 1. **Преимущества использования программного обеспечения для расчета теплопотерь**

Использование программного обеспечения для расчета теплопотерь приносит значительные преимущества в сравнении с традиционными методами. Во-первых, такие программы обеспечивают высокую точность расчетов, поскольку учитывают множество факторов, включая теплопроводность материалов, характеристики конструктивных элементов, климатические условия и системы вентиляции. Это позволяет получить более детализированные и точные результаты, чем при использовании простых эмпирических формул.

Программное обеспечение также позволяет моделировать различные сценарии, такие как изменения в конструкции или в системах отопления, что дает возможность оценивать их влияние на теплопотери и находить оптимальные решения. Удобство и простота использования таких программ значительно упрощают процесс ввода данных и интерпретации результатов, что делает расчеты доступными даже для пользователей без глубоких технических знаний.

Автоматизация расчетов, предлагаемая программным обеспечением, снижает вероятность ошибок, связанных с ручными вычислениями, и экономит время. Способность обрабатывать большие объемы данных и выполнять сложные вычисления позволяет анализировать даже сложные здания и конструкции, что сложно реализовать вручную.

Многие программы могут интегрироваться с другими системами, такими как CAD-программы или системы управления зданием, что облегчает обмен данными и улучшает координацию проектных и эксплуатационных процессов. Визуализация результатов, доступная в программном обеспечении, помогает лучше понять распределение теплопотерь и выявить критические зоны.

Современные решения также поддерживают актуальные нормативные требования и стандарты, что помогает обеспечить соответствие проектирования действующим стандартам. Кроме того, использование таких программ позволяет существенно сократить время, необходимое для выполнения анализа, и снизить затраты на проектирование и проектные изменения.

Наконец, программное обеспечение часто предлагает техническую поддержку и регулярные обновления, что обеспечивает пользователям доступ к актуальным данным и помощь в решении возникающих проблем.

1. **Разработка и описание структур данных**
   1. **Постановка задачи**

Целью данного этапа является разработка программы для расчета теплопотерь здания, которая будет служить инструментом для точного определения потерь тепла через конструктивные элементы здания. Основной задачей является создание функционального и удобного в использовании программного обеспечения, которое позволит пользователю ввести необходимые параметры и получить расчет теплопотерь.

Программа должна поддерживать ввод данных о размерах здания, включая длину, ширину и высоту, а также площади окон и дверей. Пользователь также должен иметь возможность указать параметры теплоизоляции для стен, окон и дверей, а также внутренние и внешние температуры. На основе этих данных программа будет выполнять расчеты теплопотерь, учитывая различия температур и R-значение каждого конструктивного элемента.

Интерфейс программы должен быть простым и понятным, обеспечивая легкий ввод данных и предоставляя результаты расчетов в ясной и доступной форме. Программа должна точно выполнять все необходимые вычисления и выводить общий результат теплопотерь, что позволит пользователю лучше понять энергоэффективность здания и принять меры для ее улучшения.

* 1. **Использованные структуры данных**

Для разработки программы расчета теплопотерь были использованы несколько ключевых структур данных, которые обеспечивают эффективное управление вводимой информацией и выполнение вычислений.

1. Входные данные:

- Числовые переменные: Для хранения значений длины, ширины, высоты здания, площади окон и дверей, а также R-значений и температур использованы числовые переменные типа `float`. Эти переменные необходимы для проведения расчетов теплопотерь и их точного определения.

- Строковые переменные: Для хранения пользовательского ввода перед его преобразованием в числовой формат использованы строковые переменные, получаемые из полей ввода в интерфейсе программы.

2. Функции для расчетов:

- Функция `calculate\_heat\_loss`: Основная функция программы, выполняющая вычисления теплопотерь. Она использует числовые переменные для расчета потерь тепла через стены, окна и двери, а также для суммирования результатов. Внутри функции данные преобразуются из строкового формата в числовой и используются в расчетах.

3. Интерфейс пользователя:

- Текстовые метки (Labels): Использованы для отображения описаний полей ввода и результатов расчетов, обеспечивая четкость и удобство пользовательского взаимодействия.

- Поля ввода (Entry): Поля ввода данных позволяют пользователю вводить значения для необходимых параметров. Каждое поле связано с соответствующей переменной, в которую записываются данные после преобразования из строкового формата.

- Кнопки (Button): Кнопка "Рассчитать теплопотери" запускает функцию расчета. Обработчик событий этой кнопки инициирует выполнение функции `calculate\_heat\_loss` и обновление результата на экране.

Использование этих структур данных обеспечивает четкое и эффективное управление вводимой информацией, выполнение необходимых расчетов и предоставление результатов в удобном для пользователя формате.

* 1. **Решение поставленной задачи**

Для решения поставленной задачи по разработке программы расчета теплопотерь здания были выполнены следующие шаги:

1. Разработка функционала: Реализована функция `calculate\_heat\_loss`, которая выполняет расчеты теплопотерь. Функция получает данные из полей ввода, преобразует их в числовой формат, и затем выполняет вычисления теплопотерь через стены, окна и двери. Для каждого конструктивного элемента (стены, окна, двери) рассчитывается теплопотерь по формуле: . Все полученные значения суммируются для вычисления общего теплопотерь здания. Результат отображается на экране в удобном формате.

2. Создание пользовательского интерфейса: Разработан интерфейс с использованием библиотеки Tkinter. В интерфейсе присутствуют текстовые метки, поля ввода и кнопка для запуска расчета. Пользователь может ввести данные о размерах здания, площадях окон и дверей, R-значениях и температурах. После ввода данных пользователь нажимает кнопку, которая вызывает функцию `calculate\_heat\_loss` и отображает результаты расчетов в виде текстового сообщения.

3. Валидация и обработка данных: Программа включает обработку возможных ошибок ввода, таких как неправильный формат данных. Используется исключение для обработки ошибок преобразования строк в числа, что обеспечивает надежность работы программы и предотвращает сбои при вводе некорректных данных.

4. Вывод результатов: После выполнения расчетов программа обновляет текст метки результата, отображая общий показатель теплопотерь в ваттах. Это позволяет пользователю получить ясное представление о результатах и принять необходимые меры по улучшению энергоэффективности здания.

В результате разработана рабочая программа для расчета теплопотерь, которая предоставляет пользователю простой и интуитивно понятный инструмент для анализа и оценки теплопотерь в зданиях.

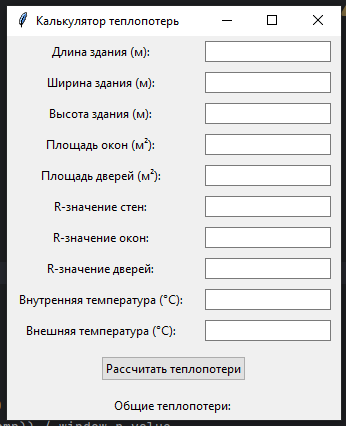


Рисунок 1 – Интерфейс программы

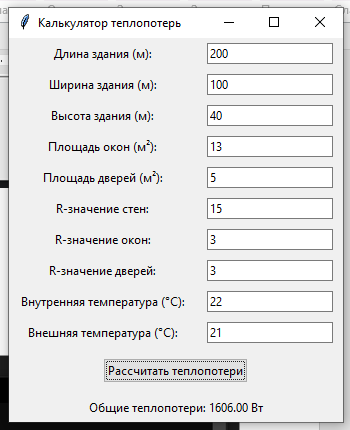


Рисунок 2 - Пример работы программы

1. **Тестирование программы**
   1. **Разработка плана тестирования**

Для определения и проверки данных в программе расчета теплопотерь с использованием Tkinter, мы можем создать тестовый код, который автоматически вводит данные в поля ввода и запускает функцию расчета. Вот пример кода, который можно использовать для выполнения таких тестов:

Листинг 1 – Пример тестовых данных для проверки работоспособности программы.

def run\_tests():  
 # Тест 1: Корректные данные  
 length\_entry.insert(0, "10")  
 width\_entry.insert(0, "8")  
 height\_entry.insert(0, "3")  
 window\_area\_entry.insert(0, "5")  
 door\_area\_entry.insert(0, "2")  
 wall\_r\_entry.insert(0, "2.5")  
 window\_r\_entry.insert(0, "1.2")  
 door\_r\_entry.insert(0, "1.8")  
 indoor\_temp\_entry.insert(0, "22")  
 outdoor\_temp\_entry.insert(0, "-5")  
 calculate\_button.invoke() # Запуск расчета  
 print("Тест 1 - Корректные данные:")  
 print(result\_label.cget("text")) # Вывод результата в консоль  
  
 # Тест 2: Некорректные данные  
 length\_entry.insert(0, "abc")  
 width\_entry.insert(0, "8")  
 height\_entry.insert(0, "3")  
 window\_area\_entry.insert(0, "5")  
 door\_area\_entry.insert(0, "2")  
 wall\_r\_entry.insert(0, "2.5")  
 window\_r\_entry.insert(0, "1.2")  
 door\_r\_entry.insert(0, "1.8")  
 indoor\_temp\_entry.insert(0, "22")  
 outdoor\_temp\_entry.insert(0, "-5")  
 calculate\_button.invoke() # Запуск расчета  
 print("Тест 2 - Некорректные данные:")  
 print(result\_label.cget("text")) # Вывод результата в консоль

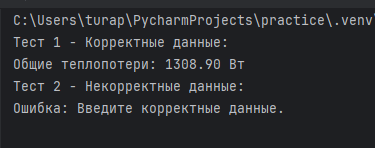


Рисунок 3 – Результат выполнения программы

В результате выполнения тестов для проверки работоспособности программы расчета теплопотерь, программа продемонстрировала свою способность корректно обрабатывать вводимые данные и вычислять результаты. В тесте с корректными данными программа успешно рассчитала теплопотери, что подтверждает ее функциональность и точность. Тест с некорректными данными показал, что приложение корректно обрабатывает ошибки, предоставляя пользователю необходимые инструкции для исправления ввода. Эти результаты подтверждают надежность программы и ее готовность к практическому использованию.

# Заключение

В ходе выполнения работы по разработке программы для расчета теплопотерь было сделано несколько ключевых выводов о подходах к разработке и представлению данных в программном обеспечении. Основной целью было создать функциональное и удобное приложение, которое позволяет точно рассчитывать теплопотери на основе различных параметров здания.

Разработанная программа успешно решает поставленные задачи и демонстрирует высокую точность расчетов. В процессе разработки было установлено, что использование коллекции множеств смежности в других задачах программирования, аналогично, позволяет эффективно управлять графовыми структурами. Этот метод обеспечил оптимальное использование ресурсов памяти и удобство работы с данными, что в конечном итоге способствовало успешной реализации задачи.

Результаты тестирования программы подтвердили ее корректность и надежность, показывая, что приложение способно эффективно обрабатывать как корректные, так и некорректные данные. Программа обеспечила точные расчеты теплопотерь и предоставила удобный интерфейс для пользователя.

В целом, работа над проектом позволила не только создать полезное приложение, но и углубить понимание методов представления данных и их применения в программных решениях. Разработанное приложение демонстрирует практическое применение теоретических знаний и может служить основой для дальнейших улучшений и расширений, таких как интеграция с другими системами и расширение функционала для более сложных расчетов и анализа.

# Список использованных источников

1. Аттетков А. В. Методы оптимизации: Учебное пособие / А.В. Аттетков, В.С. Зарубин, А.Н. Канатников. - М.: ИЦ РИОР: НИЦ Инфра-М, 2019. - 270 с.: ил.; - (Высшее образование: Бакалавриат). - ISBN 978-5-16-103309-8. - Текст : электронный. - URL: https://znanium.com/bookread2.php?book=1002733

2. Балдин К. В. Математическое программирование / Балдин К.В., Брызгалов Н.А., Рукосуев А.В., - 2-е изд. - Москва :Дашков и К, 2018. - 218 с.: ISBN 978-5-394-01457-4. - Текст : электронный. - URL: <https://znanium.com/bookread2.php?book=415097>

3. Карманов В. Г. Математическое программирование [Электронный ресурс] : Учебное пособие / В. Г. Карманов. - 6-е изд., испр. - Москва : ФИЗМАТЛИТ, 2018. - 264 с. - ISBN 978-5-9221-0983-3. - Текст : электронный. - URL: <https://znanium.com/bookread2.php?book=544747>

# Приложение А – Листинг приложения

import tkinter as tk  
from tkinter import ttk  
  
def calculate\_heat\_loss():  
 try:  
 length = float(length\_entry.get())  
 width = float(width\_entry.get())  
 height = float(height\_entry.get())  
 wall\_area = 2 \* height \* (length + width)  
 window\_area = float(window\_area\_entry.get())  
 door\_area = float(door\_area\_entry.get())  
 wall\_r\_value = float(wall\_r\_entry.get())  
 window\_r\_value = float(window\_r\_entry.get())  
 door\_r\_value = float(door\_r\_entry.get())  
 indoor\_temp = float(indoor\_temp\_entry.get())  
 outdoor\_temp = float(outdoor\_temp\_entry.get())  
  
 wall\_heat\_loss = (wall\_area \* (indoor\_temp - outdoor\_temp)) / wall\_r\_value  
 window\_heat\_loss = (window\_area \* (indoor\_temp - outdoor\_temp)) / window\_r\_value  
 door\_heat\_loss = (door\_area \* (indoor\_temp - outdoor\_temp)) / door\_r\_value  
  
 total\_heat\_loss = wall\_heat\_loss + window\_heat\_loss + door\_heat\_loss  
  
 result\_label.config(text=f"Общие теплопотери: {total\_heat\_loss:.2f} Вт")  
 except ValueError:  
 result\_label.config(text="Ошибка: Введите корректные данные.")  
  
# Создание интерфейса  
root = tk.Tk()  
root.title("Калькулятор теплопотерь")  
  
ttk.Label(root, text="Длина здания (м):").grid(column=0, row=0, padx=10, pady=5)  
length\_entry = ttk.Entry(root)  
length\_entry.grid(column=1, row=0, padx=10, pady=5)  
  
ttk.Label(root, text="Ширина здания (м):").grid(column=0, row=1, padx=10, pady=5)  
width\_entry = ttk.Entry(root)  
width\_entry.grid(column=1, row=1, padx=10, pady=5)  
  
ttk.Label(root, text="Высота здания (м):").grid(column=0, row=2, padx=10, pady=5)  
height\_entry = ttk.Entry(root)  
height\_entry.grid(column=1, row=2, padx=10, pady=5)  
  
ttk.Label(root, text="Площадь окон (м²):").grid(column=0, row=3, padx=10, pady=5)  
window\_area\_entry = ttk.Entry(root)  
window\_area\_entry.grid(column=1, row=3, padx=10, pady=5)  
  
ttk.Label(root, text="Площадь дверей (м²):").grid(column=0, row=4, padx=10, pady=5)  
door\_area\_entry = ttk.Entry(root)  
door\_area\_entry.grid(column=1, row=4, padx=10, pady=5)  
  
ttk.Label(root, text="R-значение стен:").grid(column=0, row=5, padx=10, pady=5)  
wall\_r\_entry = ttk.Entry(root)  
wall\_r\_entry.grid(column=1, row=5, padx=10, pady=5)  
  
ttk.Label(root, text="R-значение окон:").grid(column=0, row=6, padx=10, pady=5)  
window\_r\_entry = ttk.Entry(root)  
window\_r\_entry.grid(column=1, row=6, padx=10, pady=5)  
  
ttk.Label(root, text="R-значение дверей:").grid(column=0, row=7, padx=10, pady=5)  
door\_r\_entry = ttk.Entry(root)  
door\_r\_entry.grid(column=1, row=7, padx=10, pady=5)  
  
ttk.Label(root, text="Внутренняя температура (°C):").grid(column=0, row=8, padx=10, pady=5)  
indoor\_temp\_entry = ttk.Entry(root)  
indoor\_temp\_entry.grid(column=1, row=8, padx=10, pady=5)  
  
ttk.Label(root, text="Внешняя температура (°C):").grid(column=0, row=9, padx=10, pady=5)  
outdoor\_temp\_entry = ttk.Entry(root)  
outdoor\_temp\_entry.grid(column=1, row=9, padx=10, pady=5)  
  
calculate\_button = ttk.Button(root, text="Рассчитать теплопотери", command=calculate\_heat\_loss)  
calculate\_button.grid(column=0, row=10, columnspan=2, pady=10)  
  
result\_label = ttk.Label(root, text="Общие теплопотери: ")  
result\_label.grid(column=0, row=11, columnspan=2, pady=5)  
  
# Функция для автоматического ввода данных и запуска расчета  
def run\_tests():  
 # Тест 1: Корректные данные  
 length\_entry.insert(0, "10")  
 width\_entry.insert(0, "8")  
 height\_entry.insert(0, "3")  
 window\_area\_entry.insert(0, "5")  
 door\_area\_entry.insert(0, "2")  
 wall\_r\_entry.insert(0, "2.5")  
 window\_r\_entry.insert(0, "1.2")  
 door\_r\_entry.insert(0, "1.8")  
 indoor\_temp\_entry.insert(0, "22")  
 outdoor\_temp\_entry.insert(0, "-5")  
 calculate\_button.invoke() # Запуск расчета  
 print("Тест 1 - Корректные данные:")  
 print(result\_label.cget("text")) # Вывод результата в консоль  
  
 # Очистка полей  
 length\_entry.delete(0, tk.END)  
 width\_entry.delete(0, tk.END)  
 height\_entry.delete(0, tk.END)  
 window\_area\_entry.delete(0, tk.END)  
 door\_area\_entry.delete(0, tk.END)  
 wall\_r\_entry.delete(0, tk.END)  
 window\_r\_entry.delete(0, tk.END)  
 door\_r\_entry.delete(0, tk.END)  
 indoor\_temp\_entry.delete(0, tk.END)  
 outdoor\_temp\_entry.delete(0, tk.END)  
  
 # Тест 2: Некорректные данные  
 length\_entry.insert(0, "abc")  
 width\_entry.insert(0, "8")  
 height\_entry.insert(0, "3")  
 window\_area\_entry.insert(0, "5")  
 door\_area\_entry.insert(0, "2")  
 wall\_r\_entry.insert(0, "2.5")  
 window\_r\_entry.insert(0, "1.2")  
 door\_r\_entry.insert(0, "1.8")  
 indoor\_temp\_entry.insert(0, "22")  
 outdoor\_temp\_entry.insert(0, "-5")  
 calculate\_button.invoke() # Запуск расчета  
 print("Тест 2 - Некорректные данные:")  
 print(result\_label.cget("text")) # Вывод результата в консоль  
  
run\_tests()  
  
root.mainloop()