|  |  |  |
| --- | --- | --- |
| **Problem Chosen**  X | **2024**  **MCM/ICM**  **Summary Sheet** | **Team Control Number**  1234567 |

**Title**

**Summary**

Your summary

三段式写作手法

第一部分（虎头）3-5行

第二部分（猪肚）

针对问题一：

针对问题二：

第三部分（豹尾）

**灵敏度分析**

**Keywords:** keyword1; keyword2; keyword3; keyword4

Contents

[1 Introduction 3](#_Toc19220)

[1.1 Background 3](#_Toc17950)

[1.2 Restatement of the Problem 3](#_Toc8602)

[1.3 Our Work 3](#_Toc26505)

[2 Assumptions and Justifications 3](#_Toc4614)

[3 Notations 3](#_Toc19813)

[4 Problem 1: Model establishment and solution 3](#_Toc7089)

[4.1 Abc 3](#_Toc23744)

[4.1.1 Abc 3](#_Toc20010)

[5 Problem 2: Model Establishment and Solution 4](#_Toc29401)

[6 Problem 3: Model Establishment and Solution 4](#_Toc5619)

[7 Sensitivity Analysis 4](#_Toc17766)

[8 Strengths and Weaknesses 4](#_Toc18492)

[8.1 Strengths 4](#_Toc14811)

[8.2 Weaknesses 4](#_Toc11657)

[References 5](#_Toc5129)

[Appendices 6](#_Toc2655)

[Report on Use of AI 7](#_Toc7820)

# Introduction

## Background

## Restatement of the Problem

## Our Work

# Assumptions and Justifications



# Notations

The primary notations used in this paper are listed in Table 1.

Table 1: Notations

|  |  |  |
| --- | --- | --- |
| **Symbol** | **Description** | **Unit** |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |
|  |  |  |

# Problem 1: Model establishment and solution

## Abc

### Abc

# Problem 2: Model Establishment and Solution

# Problem 3: Model Establishment and Solution

# Sensitivity Analysis

# Strengths and Weaknesses

## Strengths



## Weaknesses



# **References**

[1]

[2]

[3]

[4]

# **Appendices**

|  |
| --- |
| Appendix 1 |
| Introduce: 简介 |
| 代码 |
| Appendix 2 |
| Introduce: 简介 |
| 代码 |

**Report on Use of AI**

1. **Bing AI**

**Query1: write a sort algorithm**

Output: Sure, I can help you with that. Here is a simple implementation of the selection sort algorithm in Python:

|  |
| --- |
| **def selection\_ sort (arr) :**  **for i in range (1en(arr)) :**  **min\_ idx - i**  **for j in range(i + 1，len(arr)) :**  **if arr[min\_ idx] > arr[j] :**  **min\_ \_idx = j**  **arr[i]，arr [min\_ idx] = arr[min\_ idx]，arr[i]**  **return arr** |

This algorithm sorts an array by repeatedly finding the minimum element from the unsorted part of the array and putting it at the beginning. The time complexity of this algorithm is O(n2).

There are many other sorting algorithms available such as bubble sort, insertion sort, merge sort, quick sort, heap sort, and more. Each of these algorithms has its own advantages and disadvantages. You can learn more about them on GeeksforGeeks and Programiz.

I hope this helps!

1. **OpenAl Enie (Nov 5, 2023 version, Ermie 4.0)**

Queryl: <insert the exact wording of any subsequent input into the Al tool>

Output: <insert the complete output from the second query>

1. **Github CoPilot (Feb 3, 2024 version)**

Queryl: <insert the exact wording you input into the Al tool>

Output: <insert the complete output fron the Al tool>

1. **Google Bard (Feb 2,2024 version)**

Query: <insert the exact wording of your query>

Output: <irsert the complete output from the Al tool>