

同济大学计算机系

专业实习报告



学 号 2251745

姓 名 张宇

指导老师 李文根

完成日期 2025 年 8 月 9 日

目 录

1 实习内容简介.....	3
1.1 课程背景.....	3
1.1.1 项目目标与定位.....	3
1.1.2 课程形式与内容.....	3
1.1.3 平台与工具.....	3
1.2 实习具体安排.....	3
1.2.1 课程时长与比赛安排.....	4
1.2.2 每周比赛内容.....	4
1.2.3 复盘与反馈.....	4
1.2.4 学习与提升.....	4
2 实习中遇到的难点问题分析.....	5
2.1 编程挑战的难点.....	5
2.1.1 每周比赛的难点.....	5
2.1.2 编程中的常见问题.....	5
2.1.3 比赛压力下的策略.....	6
2.2 技术难点.....	6
2.2.1 高级算法的应用.....	6
2.2.2 代码优化与调试.....	6
3 实习体会与收获.....	8
3.1 技能收获.....	8
3.1.1 编程能力的提升.....	8
3.1.2 解题思路的拓展.....	8
3.2 时间管理与自我调节.....	8
3.2.1 比赛中的时间分配与压力管理.....	8
3.2.2 自我调节与心理素质提升.....	9
4 总结与展望.....	20
4.1 对百度松果菁英班的评价.....	10
4.1.1 课程优势.....	10
4.1.2 课程不足.....	10
4.2 未来学习与发展规划.....	10
4.2.1 下一步学习方向.....	10
4.2.2 职业发展规划.....	11

1 实习内容简介

1.1 课程背景

百度松果菁英班是由百度公司推出的高水平编程训练项目，旨在培养具备扎实算法基础和解决复杂问题能力的技术型人才。该课程是为了响应人工智能、数据科学等前沿技术领域对高水平人才的需求，提供一个集中的、系统化的学习平台，帮助学员提升编程能力和算法水平。

1.1.1 项目目标与定位

百度松果菁英班旨在为学员提供一个全面的编程能力训练，通过理论与实践相结合的方式，使学员能够掌握解决复杂编程问题的技巧。该项目的核心目标是提升学员的算法设计与优化能力，帮助他们在极短的时间内解决实际问题。课程不仅注重学术理论，还强调在实际应用场景中如何快速迭代和解决问题。

1.1.2 课程形式与内容

课程的主要形式是每周的在线编程比赛，类似于国际著名的 ACM 竞赛模式。学员通过参与每周的比赛，解决一系列具有挑战性的编程题目。这些题目涵盖了算法、数据结构、数学建模等多个方面，要求学员在限定时间内提交解决方案并接受评测。通过这些比赛，学员能够不断挑战自我，锤炼技术，并积累编程经验。

比赛内容的难度逐步递进，从基础的算法和数据结构题目，到涉及复杂计算和优化的高阶题目，逐步提高学员的思维能力和解题速度。比赛题目的种类也非常广泛，包括但不限于：

- a. 排序与查找；
- b. 动态规划与贪心算法；
- c. 图论与网络流；
- d. 数学建模与模拟问题；
- e. 字符串匹配与正则表达式；

1.1.3 平台与工具

课程通过一个线上平台进行，所有的编程比赛、题目发布、代码提交和评测都在平台上完成。学员通过平台提交自己的代码，平台自动评测并给出反馈。这种线上竞赛的形式让学员可以随时随地参与，突破了传统线下比赛的时间和地点限制，极大地方便了学员的参与。

学员可以使用 C++ 编程语言，该语言由于其高效的执行速度和对内存的精细控制，常被用于解决需要优化性能的编程问题。通过 C++ 的学习与应用，学员能够更好地理解计算机系统的底层实现，提高编程效率和代码的优化能力。

1.2 实习具体安排

百度松果菁英班的实习安排主要围绕每周的编程比赛进行，结合系统化的训练和项目实战，确保学员能够在实践中不断提升编程能力、算法能力和解题技巧。以下是实习的具体安排内容：

1.2.1 课程时长与比赛安排

该课程的总体时长为一学年，分为若干个学习阶段，每个阶段以每周一次的编程比赛为核心。每周比赛的时间固定，通常安排在每周的指定工作日，比赛持续时间为 3 小时，学员在这一时间内需要独立完成题目的编程与提交。在每场比赛结束后，系统会自动评估学员提交的代码，并提供详细的评分和反馈。

1.2.2 每周比赛内容

每周的编程比赛内容都有不同的主题和挑战，比赛题目将涵盖算法、数据结构、数学建模等多个领域。题目的难度从基础到高级逐步提升，保证学员能够在每周比赛中不断突破自我。具体安排如下：

a. 第一阶段（基础训练）：主要聚焦基础算法和数据结构的应用，例如排序算法、查找算法、栈与队列、链表、树、图等常见数据结构。题目较为简单，旨在帮助学员熟悉编程语言和基本算法。

b. 第二阶段（算法与优化）：此阶段内容逐渐过渡到较为复杂的算法题目，涉及动态规划、回溯算法、贪心算法等常用的解题策略。比赛的难度逐步加大，重点锻炼学员的思维能力和代码优化能力。

c. 第三阶段（复杂题目与综合应用）：此阶段的题目会加入更多的数学建模、图论、网络流等内容，需要学员在解决问题时综合运用多种算法和策略。题目复杂且具有一定的挑战性，要求学员具有较强的算法分析和优化能力。

1.2.3 反馈与复盘

每场比赛结束后，平台会提供详细的评分报告和反馈，帮助学员了解自己的解题过程和代码质量。学员可以通过分析评测报告中的时间消耗、空间消耗和性能瓶颈，找到自己需要改进的地方。课程还会组织定期的复盘环节，学员可以与其他参赛者一起讨论比赛中的解题思路，交流技术经验。

1.2.4 学习与提升

除了每周的编程比赛，课程还会组织一些针对性的技术讲座和学习材料推荐，帮助学员深入了解一些高阶的算法和编程技巧。这些材料和讲座的内容紧贴比赛题目的难点，帮助学员在实际编程过程中解决问题，提升解题能力。

2 实习中遇到的难点问题分析

2.1 编程挑战的难点

在每周的编程比赛中，题目的难度逐渐递增，涉及到的算法和数据结构的应用也越来越复杂。以下是几个典型的难题以及如何通过 C++ 解决这些问题的经验。

2.1.1 每周比赛的难点

a. 图的遍历与最短路径问题

比赛中，我曾遇到过一个经典的图遍历问题：给定一个无向图，求出从某个节点到其他所有节点的最短路径。初始阶段，我使用了简单的广度优先搜索（BFS）来解决这个问题。然而，由于节点数量非常庞大，BFS 的时间复杂度较高，尤其是在图的规模达到数万时，容易超时。

通过分析图的特点，我选择了 Dijkstra 算法来优化最短路径的计算。使用优先队列（priority_queue）来优化算法，使得时间复杂度从 $O(n^2)$ 降低到了 $O(n \log n)$ 。通过合理的数据结构和算法选择，我成功避免了超时问题。

b. 动态规划的优化

另一个常见的难题是动态规划问题。例如，在一个背包问题的变种中，题目要求求解最大价值的背包，并且容量和物品数非常庞大。在使用动态规划时，常规的二维数组会导致空间复杂度非常高。为了解决这个问题，我通过空间优化，将二维数组压缩为一维数组，极大地降低了内存消耗，并保持了算法的时间复杂度。

具体实现时，我使用了“滚动数组”的方式，只保存当前状态和上一轮的状态，从而避免了冗余的计算。这种优化不仅减少了空间复杂度，还提高了程序运行的效率。

c. 递归与记忆化搜索

在处理一些递归问题时，往往会遇到重复计算导致性能瓶颈的问题。比如在求解某些斐波那契数列类的动态规划问题时，直接递归会导致大量重复的子问题计算。为此，我使用了记忆化搜索（Memoization）技术，将已经计算过的结果缓存起来，从而避免了重复计算，显著提升了算法效率。

例如在解决背包问题时，使用递归加缓存的方式可以快速计算子问题的解，并通过回溯逐步合成最终解，从而将时间复杂度从 $O(2^n)$ 降低到 $O(n \cdot W)$ （其中 W 为背包容量）。

2.1.2 编程中的常见问题

在编写代码时，最常见的问题之一是时间复杂度和空间复杂度的控制。特别是在比赛的高压环境下，如何在保证程序正确性的同时，优化代码的执行效率是一个重要的挑战。

比赛中，我常常面临着如何在短时间内处理大量数据的问题。以一个图的最短路径问题为例，如果直接使用暴力方法（例如广度优先搜索或者深度优先搜索），其时间复杂度可能达到 $O(n^2)$ 或更高，这很容易超出比赛给定的时间限制。为了解决这个问题，我会根据题目的特点选择合适的算法，并使用高效的数据结构（如堆、优先队列、哈希表等）来优化性能。

对于一些数据量较大的题目，我还通过数据分块、分治法等方式减少计算量，并结合递归与迭代优化，尽可能地控制空间复杂度，避免内存溢出。

2.1.3 比赛压力下的策略

在编程比赛中，时间的压力通常非常大，因此如何在有限的时间内高效地解决问题，合理分配时间是非常重要的。为了在比赛中取得更好的成绩，我总结了一些应对策略。比赛开始前，我会快速浏览所有题目，并初步评估每道题目的难度和解题思路。对于那些看似难度较大的题目，我会先尝试做一些简单的题目，以积累分数并建立信心。遇到难题时，我会选择先解决一些简单的部分，逐步逼近解答，而不是一开始就对复杂问题进行深入分析。

在比赛过程中，我会优先选择自己最擅长的题目进行解决。通常，比赛中会有一些基础的排序、查找等算法题，这些题目不仅能够帮助我在较短时间内积累分数，还能帮助我迅速找到解题的思路并在比赛中保持良好的节奏。如果遇到难度较大的题目，我会采取逐步推进的方式，尽量不陷入困境，避免浪费时间。

在高压环境下，保持冷静和专注是非常关键的。我通过反复的练习和比赛，逐渐培养了在紧张环境中高效思维和快速决策的能力。每当遇到卡壳时，我会深呼吸，暂停片刻，调整思路，重新审视问题，并迅速做出下一步的判断。这些策略帮助我在比赛中保持高效的工作状态，并有效地提升了我的解决问题的速度和质量。

2.2 技术难点

在百度松果菁英班的编程比赛中，我遇到了许多技术上的难点，尤其是在解决复杂算法问题时。以下是我在解决这些问题过程中遇到的主要技术难点及其应对策略：

2.2.1 高级算法的应用

比赛中，涉及了很多高难度的算法问题，尤其是动态规划、图算法和贪心算法的应用。在这些问题上，我必须确保算法不仅能正确解决问题，还能满足时间和空间复杂度的要求。

例如，在解决动态规划问题时，常常需要通过状态转移方程来优化计算过程。然而，动态规划的核心难点在于如何合理划分状态空间并设计合适的递推关系。在比赛中，我多次遇到需要处理大规模输入的动态规划题目，为了避免重复计算，我使用了记忆化搜索和滚动数组等技巧，优化了算法的空间复杂度。

在图算法方面，我在比赛中遇到了复杂的最短路径问题，例如 Dijkstra 算法和 Floyd-Warshall 算法。这些算法本身已经有一定的时间复杂度，但在处理较大规模图时，仍然可能会超时。因此，我通过使用优先队列来优化 Dijkstra 算法，将时间复杂度从 $O(n^2)$ 优化为 $O(n \log n)$ ，大大提高了算法效率。

在贪心算法的应用中，许多问题的解法依赖于对问题的深入理解和良好的贪心策略。例如，在处理某些区间问题时，通过分析问题的结构，我能够提出合适的贪心选择，使得算法的时间复杂度保持在可接受的范围内。

2.2.2 代码优化与调试

比赛中的时间压力非常大，因此，我必须在有限的时间内编写出高效且无错误的代码。为了提高代码的效率，我常常会对代码进行优化，减少冗余的计算和不必要的内存分配。

在比赛过程中，我使用了一些常见的优化技巧，例如提前返回、剪枝等，避免了在某些问题中进行无谓的计算。例如，在解决一些涉及大量计算的递归问题时，我通过引入递归深度限制，提前判断某些不可能的情况，从而提高了程序运行速度。

在调试方面，比赛中的时间压力让我没有太多时间进行传统的调试。因此，我通常会提前做

好单元测试，确保代码的正确性。在遇到程序出现错误时，我会快速定位问题，使用断点调试和输出调试相结合的方法，尽可能快速地找到并修复代码中的漏洞。此外，我还学会了如何通过性能分析工具（如 `gprof`）来分析程序的瓶颈，并进行针对性的优化。

3 实习体会与收获

3.1 技能收获

在参与百度松果菁英班的实习过程中，通过每周的编程比赛，我获得了许多宝贵的技能收获。这些收获不仅限于编程能力的提升，还包括解题思路的拓展以及对 C++ 语言应用的深入理解。以下是我在实习过程中最大的技能收获：

3.1.1 编程能力的提升

通过每周比赛的反复练习，我在算法设计和数据结构应用方面得到了显著提升。在刚开始接触竞赛时，我常常会遇到一些复杂的题目，无法快速找到解题的思路。然而，随着练习的积累，我逐渐掌握了不同类型题目的解法和优化技巧，尤其是在排序、查找、图算法、动态规划等方面。

通过不断实践，我逐步提高了对问题的快速分析与解决能力，能够在短时间内快速找出合适的算法并进行实现。随着时间的推移，我开始能够将这些算法进行优化，提高代码执行效率，尤其是在复杂度较高的题目中，通过优化减少了算法的时间和空间开销。

在每次比赛结束后，我都会复盘总结自己的得失，不断总结自己的优缺点。逐渐地，我学会了如何在比赛中做出最优的时间分配，如何在有限的时间内解决最大量的题目。这不仅提高了我的编程能力，也让我在面对高强度的编程任务时更加从容。

3.1.2 解题思路的拓展

每周的比赛不仅考验我对算法和数据结构的掌握，还考验我解题的思维方式和思考深度。初期，我常常陷入一些问题的表面，无法深度理解问题的本质。随着比赛的进行，我逐渐意识到，解题不仅仅是完成一个算法的实现，更重要的是如何从问题出发，分解问题，构建合适的解题框架。

我开始逐步形成了一种系统化的解题思路：首先，通过分析问题的输入输出要求，理解题目中的隐含条件；然后，选定合适的算法和数据结构，进行思路推导和算法设计；最后，通过调试、优化和验证，确保代码的正确性和效率。这种系统的思维模式使我在比赛中能够更加高效地解决问题。

例如，在一些动态规划问题中，最开始我难以理解如何设计状态转移方程。通过不断练习并学习其他优秀选手的解法，我逐步掌握了如何合理划分状态，并将状态转移的公式简化成易于实现的步骤。这不仅增强了我解决动态规划问题的信心，也让我对算法的设计和优化有了更深刻的理解。

3.2 时间管理与自我调节

在参加百度松果菁英班的编程比赛过程中，时间管理与自我调节成为了我提升效率、应对压力的关键因素。每场比赛不仅要求快速解决问题，还需要在限定时间内保持高效的思维与执行力。以下是我在比赛中关于时间管理和自我调节的具体经验与心得：

3.2.1 比赛中的时间分配与压力管理

每次比赛开始时，面对时间的压力，我都会首先进行题目快速评估，对所有题目的难度进行初步判断。在这一阶段，我不会深入分析每个题目的解法，而是根据自己的优势与比赛经验，决定先解决哪些题目。例如，我会优先选择那些自己熟悉且较为简单的算法题目，这样不仅能积累时间，还能帮助我进入比赛状态，逐渐提升自信心。

在时间紧迫时，我会采取分阶段的解决方法。一开始，我尽量在短时间内解决简单的题目，积累一定的分数后，再挑战难度较大的问题。对于复杂题目，我会通过分步走的策略，将题目拆解成若干个子任务，逐步推进。这样的做法能够确保我在有限的时间内有效解决问题，同时避免被一个难题困住而浪费过多时间。

此外，在比赛过程中，我会随时调整解决问题的优先级。例如，当我遇到一个难题时，若无法立即解决，我会暂时跳过，先做其他题目，等到比赛剩余时间充裕时再回头解决难题。这种灵活的应对策略帮助我在比赛中合理分配时间，避免了因时间压力过大而导致的情绪波动。

3.2.2 自我调节与心理素质提升

比赛中的压力不可避免，如何在高强度的环境下保持冷静和专注，往往决定了比赛的成绩。在面对时间的紧迫与不断出现的问题时，我学会了如何进行自我调节，保持高效的心态。

首先，我深知保持冷静是应对压力的关键。在比赛过程中，若遇到卡壳或未能立即解决的问题，我不会急于做出反应，而是深呼吸，放松自己，短暂地停下来调整心态。这有助于我保持清醒的思维，避免陷入焦虑或压力中，从而提高了在压力下的决策能力和执行力。

其次，我学会了通过细分任务来应对比赛中的挑战。当比赛时间有限，且题目复杂时，单纯的思考问题本身可能会让我感到压迫。此时，我会将问题分解成小步骤，逐一攻克。这不仅能减轻心里的负担，还能够在比赛过程中逐步看到进展，从而增强信心和动力。

在比赛过程中，我也学会了适时调整自己的工作节奏。例如，在解决较为复杂的题目时，我会暂停片刻，重新审视自己的解题思路，避免陷入死胡同；对于较为简单的题目，我则会迅速动手，通过快速思考并实践解决问题，这样既能提高效率，又能避免浪费时间。

通过不断参加比赛，我逐步提升了应对压力的心理素质，并学会了如何保持冷静、合理分配时间、调整节奏，这使我能够在高强度的编程比赛中保持清晰的头脑，迅速做出决策，提升了整体表现。

4 总结与展望

4.1 对百度松果菁英班的评价

4.1.1 课程优势

百度松果菁英班的课程内容非常贴近实际，尤其是在培养学员独立解决问题的能力方面，体现了非常高的水平。每周的编程比赛不仅让学员积累了大量的编程经验，还帮助我们提升了算法分析、代码优化和高效解题的能力。比赛形式模拟了真实的技术挑战，具有较强的实战性和针对性，能有效提高学员在实际工作中遇到技术问题时的应变能力。

课程中涉及的题目种类繁多，从基础的排序和查找到高阶的动态规划、图算法和数学建模，涵盖了算法和数据结构的方方面面。通过这些题目的训练，我不仅学到了许多新的解题技巧，还掌握了如何在短时间内高效选择合适的算法解决问题。此外，比赛的实时反馈和评估机制让学员能迅速了解自己代码的优缺点，并及时进行调整和优化，这对于我的成长非常有帮助。

课程还鼓励学员进行复盘总结，每周比赛后都有机会回顾自己的思路和解法，发现自己的优缺点并加以改进。这种培养独立思考和总结的模式，非常适合培养编程能力和解决问题的综合能力。

4.1.2 课程不足

尽管百度松果菁英班在课程设计和比赛形式上有许多优势，但也存在一些改进空间。首先，课程内容的深度可能还不够。虽然课程涵盖了广泛的算法和数据结构，但在某些高级主题（例如深度学习、机器学习优化等）方面的内容较为薄弱。如果课程能够增加一些更深入的专题讲座，帮助学员深入理解一些高阶算法或新兴技术领域，可能会更有助于学员的全面成长。

其次，比赛时间和准备时间的平衡也是一个问题。尽管每周的比赛都非常具有挑战性，但比赛题目的难度有时会超出学员的准备范围，导致在比赛中无法充分发挥。一些较为复杂的题目需要更多的预备知识和深入的练习。如果能提供更详细的预习资料或提前揭示比赛题目的大致方向，学员在比赛前能有更充分的准备，可能会提高比赛的整体效果和参与感。

最后，课程可以考虑在比赛之外增加更多的实际项目或团队合作项目。尽管每周的编程比赛能有效提高学员的技术能力，但缺少团队合作和实际项目经验，可能会让学员对如何将编程应用于实际工作场景的理解有所欠缺。如果能够通过项目实战或团队合作的形式，帮助学员更好地理解如何在实际项目中应用算法，可能会使课程内容更贴近职业需求。

4.2 未来学习与发展规划

4.2.1 下一步学习方向

通过在百度松果菁英班的实习，我不仅提升了编程能力，还进一步明确了自己未来学习的方向。接下来，我计划在几个关键领域深入学习，并继续加强自己的技术水平。

首先，深度学习和人工智能是我未来学习的重点领域。随着人工智能的快速发展，深度学习已经成为该领域的核心技术之一。我计划深入学习深度学习的基础与进阶知识，特别是卷积神经网络（CNN）、循环神经网络（RNN）以及生成对抗网络（GAN）等前沿技术。通过结合理论学习和实践项目，我希望能更好地掌握这些技术，并在计算机视觉、自然语言处理等领域探索其实际应用。为此，我将通过在线课程、书籍学习以及参与开源项目来进一步提升自己在深度学习方面的应用能力。

另外，算法优化和数据结构的学习仍然是我的重点。我已经掌握了许多常见的算法，但在面

对更复杂的问题和大数据时，我意识到我仍然需要进一步提升自己在算法优化方面的能力。特别是在分治算法、动态规划和图算法等方面的优化，这些知识是解决大规模数据处理和复杂问题的关键。我计划通过参与更多的算法竞赛，学习优化技巧，并阅读一些经典的算法书籍，来不断增强自己的解题能力，提升解决复杂问题的效率。

随着数据量的不断增长，大数据处理和高性能计算已成为未来技术发展的重要方向。因此，我也计划学习和掌握分布式计算、并行计算等技术，了解如何处理海量数据，并利用现代计算平台高效地进行数据分析。通过实际项目的参与，我希望将这些知识应用于真实的工作场景中，进一步提升自己在大数据和高性能计算领域的竞争力。

这些学习方向不仅是我在百度松果菁英班实习期间的反思总结，也是在未来工作中不断追求技术深度和广度的规划。我相信，通过不断的学习和实践，我能不断提高自己的技术水平，为将来在技术领域的发展打下更加坚实的基础。

4.2.2 职业发展规划

通过提升编程能力和算法水平，我希望为自己未来的职业发展打下坚实的基础。短期内，我计划继续参加各种编程比赛和技术活动，以进一步锤炼我的问题解决能力和编程技巧。这些比赛不仅能够提高我的算法水平，也为我进入技术岗位做好充分准备。我相信，通过不断地实践，我将在求职市场中获得更强的竞争力，并能够应对未来技术岗位上的挑战。

我未来的职业目标是成为一名算法工程师，并专注于人工智能、深度学习和大数据分析领域的研究与应用。我希望能够在技术领先的公司中工作，参与技术创新，并将所学的算法和数据结构应用到实际的项目开发中。通过解决实际问题并应用先进技术，我希望能够推动技术的进步，并为行业的发展做出贡献。

为了达到这个职业目标，我将继续加强对编程语言（如 C++、Python）和软件工程的掌握，提升自己的代码实现能力和系统设计能力。随着技术的快速发展，我计划学习更多领域的知识，如云计算、数据库管理等，为将来可能涉及的大规模数据处理和分布式系统开发做好准备。我还打算通过研究生深造或参加技术培训，进一步扩展我的技术视野，使自己在算法和计算机科学的前沿技术领域始终保持领先。

在未来的工作中，我希望能够逐步成长为技术团队的核心成员，参与到复杂项目的设计与开发中。随着工作经验的积累，我将不断挑战更高难度的技术问题，提升自己的技术深度，并争取早日晋升为技术负责人。我期望能够在技术领域不断积累经验，并最终向管理岗位发展，成为团队和项目的领导者。

通过这次实习，我对自己的职业方向有了更加明确的规划，并希望通过不断的学习和实践，逐步实现自己的职业目标。我将持续关注前沿技术的发展，保持对技术的热情，不断提升自己的综合能力，为未来的工作和研究打下坚实的基础。