UNIVERSITY OF ABERDEEN

阿伯丁大学

自然与计算科学学院 计算科学系

2024 - 2025

编程作业--由 4-5 名学生组成的小组合作完成

标题：JC4004 - 计算智能	注：本作业占课程总分的 30%。

截止日期： 在 *MyAberdeen* 上提交作业。中国时间 2024 年 12 月 23 日 23:00。

剽窃和串通信息： 源代码和您的报告可提交到 *MyAberdeen* 进行抄袭检查。有关避免抄袭的更多信息，请参阅 *MyAberdeen* 上的幻灯片。过度使用大型语言模型（如 ChatGPT）编写代码或报告也可能被视为抄袭。此外，与其他小组一起提交类似的作业也会被视为串通行为。

延期信息： 根据阿伯丁大学新的延期政策，教师不得再对课程作业的截止日期进行延期。可通过电子邮件向学校管理部门申请延期：uoa-ji-enquiries@abdn.ac.uk。

延期需要强有力的理由（如重病或申诉），延期申请应附有证明，如医疗证明。延期政策另见另一份文件。由于本作业是集体作业，只有在非常特殊的情况下才允许延期。

导言

在本作业中，您的任务是制作一个人工智能游戏机器人，用于玩传统棋盘游戏 "***狐狸和鹅***"。你的游戏机器人应能扮演狐狸和鹅两边玩游戏。游戏的详细规则说明如下。请注意，该游戏有不同的版本：对于本作业，您应遵循本文档中描述的规则。

狐狸和鹅是一款双人棋盘游戏。其中一名玩家扮演狐狸，试图捕获所有的鹅。另一名玩家代表大雁，试图包围狐狸，使其无法再移动。游戏在棋盘上进行，狐狸和大雁有 33 个可能的位置。开始时，棋盘上有 15 只鹅和一只狐狸，如图 1 所示。白色棋子是鹅，红色棋子是狐狸。

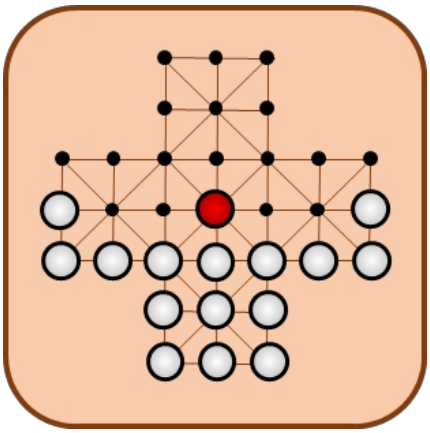

图 1.狐狸和鹅的初始位置。

游戏轮流进行。在这个版本的游戏中，狐狸和鹅都可以在自己的回合中沿着棋盘上的线水平、垂直或对角线移动一步。玩鹅的玩家可以选择棋盘上任何一只鹅移动。*请注意，只有棋盘上的线条标明的某些位置才允许对角线移动。*

您不能将棋子移动到已被其他棋子占据的位置。不过，狐狸可以跳过一只鹅，将其吃掉。被吃掉的鹅会被移出棋盘。也可以像下*跳棋*一样，通过连锁跳跃在一个回合内吃掉多只鹅。鹅不能吃掉狐狸。即使有可能吃掉狐狸，也不是一定要吃掉它，但狐狸和鹅都必须在自己的回合中走一步棋。合法走棋的例子如下图 2 所示。

鹅的目标是包围狐狸，让它无法再做出任何合法动作。狐狸的目标是捕获所有的鹅。从理论上讲，最少四只大雁就足以包围狐狸；因此，当棋盘上剩下的大雁少于四只时，狐狸就

赢了。游戏获胜的例子如图 3 所示。

由于狐狸和大雁的目标不同，遵循的规则也不同，所以游戏是*不平衡的*。因此，玩家通常会玩双数游戏，交换角色。谁是

赢得更多游戏的人就是最后的赢家。在本作业中，您的任务是实现狐狸和鹅的游戏逻辑。

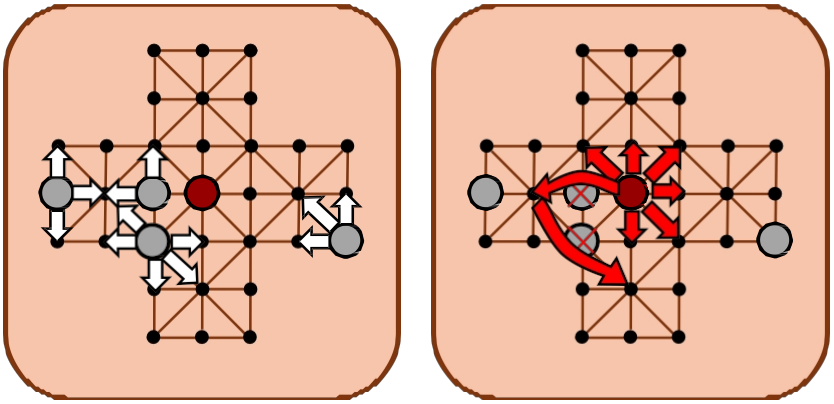

图 2.鹅（左）和狐狸（右）的合法动作示例。

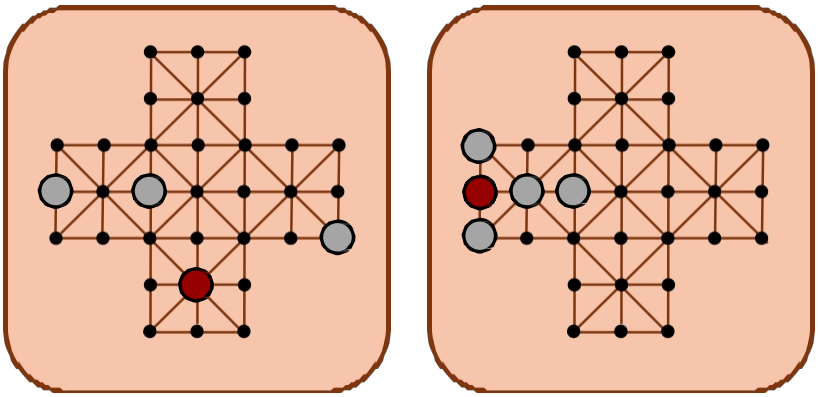

图 3.狐狸赢得游戏（左）和大雁赢得游戏（右）的例子。

一般指导和要求

在本作业中，您需要编写一个 Python 类 `Player`，该类可以通过 `play_fox()` 和 `play_goose()` 方法玩狐狸和鹅游戏。当前棋盘作为参数传递给这两个方法，这两个方法将分别以狐狸或鹅的形式返回下一步棋。将共享 Python 文件 **TestFoxAndGoose.py**，以演示游戏测试框架如何使用 `Player` 类。

棋盘是一个二维列表对象，上面有 7 × 7 个字符，代表棋局的状态。字符 "F "和 "G " 分别代表狐狸和鹅。空位置用点". "标记，空格""标记游戏区域外的位置。在文件 **TestFoxAndGoose.py** 中的 **FoxAndGoose** 类中，棋盘初始化如下：

```
self.board = [
        [' ',' ','.','.','.',' ',' '
        '],
        [' ',' ','.','.','.',' ',' '
        '],
        ['.','.','.','.','.','.','.']
        ,
        ['g'、'。'、'。'、'f'、'。'、'。
        '、'g']、
        ['g'、'g'、'g'、'g'、'g'、'g'、
        'g']、
        [' ',' ','g','g','g',' ',' ']
        、
        ['',''、'G'、'G'、'G'、''、'']
    ]
```

代码中的 play_fox() 和 play_goose()方法应将上面定义的棋盘作为输入参数。作为输出参数，该方法应返回一个包含两个或多个整数对的 list 对象，其中第一个值代表棋盘上的行，第二个值代表棋盘上的列。第一对是初始位置，第二对是目标位置。例如，返回值 [[3,2],[3,3]] 表示位于 4th 行、3rd 列的棋子将被移动到 4th 行、4th 列。注意编号是从 0 开始的：例如，位置 [0,1] 是 1st 行的 2nd 列。

play_fox()方法可以返回一个包含多个目标位置的更长列表，以防狐狸在一次移动中捕获不止一只鹅。例如，返回值 [[3,3],[3,1], [5,3]] 表示狐狸先从位置 [3,3] 跳到位置 [3,1]，捕获了位置 [3,2] 的鹅，然后继续跳到位置 [5,3]，捕获了位置 [4,2] 的鹅。

您可以自由决定使用哪种计算智能技术来实现游戏逻辑。如有必要，你还可以实现其他函数和类。不过，如上所述，Player 类只能通过 play_fox() 和 play_goose()方法

与游戏框架交互。机器人应具有合理的复杂性：在测试阶段，考虑移动的时间限制为 5 秒。如果您的实现需要耗时的初始化，例如下载深度神经网络，则初始化应该在类 `Player` 的构造函数 `init` 而不是 `play_fox()` 和 `play_goose()` 方法中完成。

您可以适度使用代码生成工具和外部来源的代码来辅助部分代码的实现，但应在项目报告中对任何来源或工具的使用进行解释，并提供参考文献。

提交要求

您应在 *MyAberdeen* 的课程页面中提交作业。您提交的作品至少应包括两个文件：**TeamXX.py** 文件，其中包含实现类 `Player` 的 Python 代码，以及 `play_fox()` 和 `play_goose()` 方法；**ReportXX.pdf**，即项目报告。在文件名中，将 **XX** 替换为团队编号，例如 **05**。作为示例，我们提供了 **Team00.py** 文件，允许您使用人类用户输入的棋步手动下棋。如果您的代码运行需要任何其他文件，例如预训练的神经网络，您也应在提交的文件中包含这些文件。

请注意，您有责任确保 **TeamXX.py** 中的代码在我们测试时能正常工作：您应使用 **TestFoxAndGoose.py** 文件导入您的类，并测试您的代码是否能在测试框架中正常工作。将 `module= import ( "Team00")` 中的模块名 `Team00` 替换为您自己的文件名，**后缀名** 不要**为 .py**。如果您的代码有需要额外安装的外部依赖项，应在项目报告或提交文件中的 **readme** 文件中明确说明。

请注意，由不同小组实现的游戏机器人将相互对战，因此必须确保兼容性。您应该使用 **Python 3**。如果您使用任何第三方软件包，如 TensorFlow 或 PyTorch，我们建议您使用最新的稳定版本，并避免使用存在已知向后兼容性问题的功能。我们建议从一个干净的环境开始，跟踪所有已安装的软件包及其版本号，并在项目报告或 **readme** 文件中报告它们。

请注意，在撰写本文时，最新的 TensorFlow 版本与最新的 Python 稳定版 3.13.0 不兼容。因此，如果您计划使用 TensorFlow，Python 版本应为 **3.12.7** 或更早。

项目报告的长度应在 1 500 字左右。建议包含图表说明，但应避免程序代码截图。如果代码实现了一些难以用其他方式解释的复杂算法，可以使用流程图或伪代码作为说明工具。报告应包括以下部分：

1. 导言：*约200 字。*
2. 理论依据，包括所使用方法和算法的描述，并简要说明为何选择这些技术：*约600 字。*
3. 实施细节，包括使用的库和 UML 图或基本方法及其参数列表等：*约300 字。*
4. 结论，包括自我反省、面临的困难、测试代码的经验以及今后改进的想法：*约300*

字。

5. 个人作用概述，包括团队成员贡献的简要说明：

   约100 字。

6. 参考资料

如果您希望在 MyAberdeen 的排行榜中公布您所在小组的成绩，请在报告中为您所在的小组命名！

评分标准

作业将根据***项目报告***（40 分）、***方法论***（40 分）和以下内容进行评分
表现（20 分）。

项目报告将根据所要求的覆盖面、表述的清晰度（包括语言和插图）、报告与提交的代码之间的一致性以及参考文献的相关性进行评分。

评估方法的依据包括：所选方法和算法对特定任务的适用性、创造性（例如，以非传统的方式结合不同的方法）以及实施（例如，源代码的清晰度、计算效率）。

为了进行性能评估，我们将安排所有提交的作业相互对战，对它们进行测试。每份作业都将与其他每份作业对弈两次，一次作为狐狸，一次作为鹅。比赛结果将汇总到联赛表中，胜一场得一分，负一场得零分。获胜者将获得 20 分，其他小组将根据公式得分：

$$m_i = \frac{20x_i}{xwiner},$$

其中，m_i 是i组的分数，x_i 是i组的总分，x_winer 是联赛冠军的总分。

请注意，如果两个博弈机器人在同一位置来回重复走棋，博弈可能会陷入僵局。为解决僵局，最大下棋步数设置为 1000 步。如果棋局因僵局而无胜负，双方都将获得零分。

联系方式

如有任何问题或疑问，请联系课程教师：人工智能课程的 Jari Korhonen 博士 (jari.korhonen@abdn.ac.uk) 和计算机科学与技术课程的黄永超博士 (yuan.wen@abdn.ac.uk)。