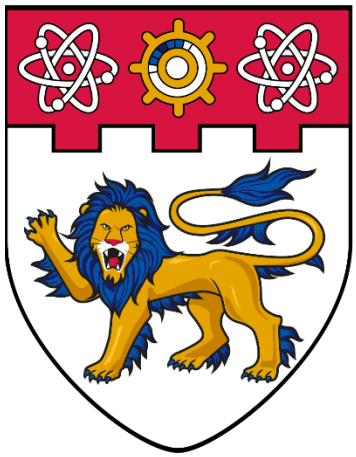

NANYANG TECHNOLOGICAL UNIVERSITY

SINGAPORE

School of Computer Science and Engineering

CE4046 Intelligent Agents

Assignment 2

Name: Lumlertluksanachai Pongpakin

Matriculation Number: U2023344C

Other Agents

In the program, there are six different types of players: NicePlayer, NastyPlayer, RandomPlayer, TolerantPlayer, FreakyPlayer, and T4TPlayer.

- NicePlayer always cooperates, so it returns 0.
- NastyPlayer always defects, so it returns 1.
- RandomPlayer randomly picks its action each time, returning 0 with a probability of 0.5 and 1 with a probability of 0.5.
- TolerantPlayer looks at its opponents' histories, and only defects if at least half of the other players' actions have been defects. It first counts the number of times its opponents have cooperated and the number of times they have defected. If the number of defections is greater than the number of cooperations, it returns 1. Otherwise, it returns 0.
- FreakyPlayer determines, at the start of the match, whether to always be nice or always be nasty. It randomly selects to cooperate (0) with a probability of 0.5 and to defect (1) with a probability of 0.5. It always returns that decision. In other words, it will be 0.5 NicePlayer or 0.5 NastyPlayer.
- T4TPlayer picks a random opponent at each play, and uses the 'tit-for-tat' strategy against them, which means it starts by cooperating (0) and then copies its opponent's previous move. If the opponent defects, it defects in the next round; if the opponent cooperates, it cooperates in the next round.

Potential Agents

There are a plethora of strategies that individuals can potentially adopt, but the predominant ones are the Adaptive Tit for Tat (TFT) strategy and Grim Trigger strategy.

The Adaptive TFT Player class effectively maintains a record of its past cooperative and defecting moves and adjusts its strategy accordingly. Specifically, if the player has defected more than it has cooperated, the player becomes more tolerant and will cooperate even if the opponent has defected. In contrast, if the player has cooperated more, it continues to utilize the TFT strategy by emulating the preceding move of its opponent. The implementation of this Adaptive TFT player can be found in the code below.

```
class AdaptiveT4TPlayer extends Player {
    int defectCount = 0;
    int cooperateCount = 0;

    int selectAction(int n, int[] myHistory, int[] oppHistory1, int[] oppHistory2) {
        if (n == 0) {
            // Cooperate on the first move
            return 0;
        }

        // Count the number of defections and cooperations in the previous round
        int myLastMove = myHistory[n-1];
        int oppLastMove1 = oppHistory1[n-1];
        int oppLastMove2 = oppHistory2[n-1];
        int numCooperations = (myLastMove == 0 ? 1 : 0) + (oppLastMove1 == 0 ? 1 : 0) + (oppLastMove2 == 0 ? 1 : 0);
        int numDefections = 3 - numCooperations;

        // Update the defect and cooperate counts
        if (myLastMove == 1) {
            defectCount++;
        } else {
            cooperateCount++;
        }

        // Check if the opponent is defecting more than cooperating
        boolean opponentDefecting = (numDefections > numCooperations);

        // Adjust strategy based on performance so far
        if (defectCount > cooperateCount) {
            // If we've been defecting more than cooperating, become more forgiving
            return (opponentDefecting ? 1 : 0);
        } else {
            // Otherwise, continue using Tit-for-Tat
            return oppLastMove1;
        }
    }
}
```

The Grim Trigger Player strategy commences by cooperating. However, if either player defects, it will persist in defecting for the remainder of the game. The implementation of this Grim Trigger player can be found in the code provided below.

```
class GrimTriggerPlayer extends Player {
    boolean triggered = false;

    int selectAction(int n, int[] myHistory, int[] oppHistory1, int[] oppHistory2) {
        if (!triggered) {
            // Cooperate until the other player defects
            if (n == 0 || oppHistory1[n-1] == 0 && oppHistory2[n-1] == 0) {
                return 0;
            } else {
                triggered = true;
                return 1;
            }
        } else {
            // Always defect once triggered
            return 1;
        }
    }
}
```

Adaptive TFT and Grim Trigger Evaluation

	Score					
	1	**2**	**3**	**4**	**5**	**Average**
Adaptive TFT	208.99	213.80	207.15	197.52	199.33	204.45
Grim Trigger	210.62	**214.35**	**220.98**	**213.55**	**216.49**	**215.20**
Nice	191.85	211.13	191.13	196.89	198.05	197.81
Nasty	165.31	158.57	164.20	174.09	170.11	166.46
Random	157.91	171.25	152.81	163.44	158.64	160.81
Tolerant	**216.17**	202.48	202.08	206.00	214.60	208.27
Freaky	181.74	182.05	188.16	171.59	177.45	180.20
T4T	197.51	201.03	204.91	204.10	207.48	203.01

Agent Implementation

The table shows that the Grim Trigger strategy produced the highest average score in the tournament. However, to achieve the highest possible score, it may be necessary to enhance the Grim Trigger approach.

The implemented player is based on the Grim Trigger strategy, which initially cooperates with the opponent. If the opponent defects, the player responds likewise, but only for a fixed number of rounds. After this period, the player forgives the opponent and returns to cooperation. The duration of the punitive phase is determined by the "forgivenAfter" variable, which is set to 5 in this case. Moreover, the implemented strategy includes a few final rounds of defection as retaliation from opponents is not feasible at that point in the game. The implementation code is presented below.

```
class Lumlertluksanachai_Pongpakin_Player extends Player {
    // Initialize variables
    boolean defected = false; // boolean variable to keep track of whether this player has defected
    int forgivenAfter = 5; // the number of rounds after which this player is forgiven for defecting
    int roundsSinceDefection = 0; // the number of rounds since this player last defected

    int selectAction(int n, int[] myHistory, int[] oppHistory1, int[] oppHistory2) {

        // Defect in the last few rounds
        if (n >= 107) return 1;

        // If this player has defected before, check if it has been enough rounds to be forgiven
        if (defected) {

            // Increment the roundsSinceDefection variable
            roundsSinceDefection++;

            // If the player has been forgiven, reset the defected variable and roundsSinceDefection variable
            if (roundsSinceDefection >= forgivenAfter) {
                defected = false;
                roundsSinceDefection = 0;
            }

            // Always return 1 if this player has defected before
            return 1;
        }

        // If this player has not defected before, check if its opponents have defected in the past
        for (int i = 0; i < n; i++) {
            if (oppHistory1[i] == 1 || oppHistory2[i] == 1) {
                // If either opponent has defected, set this player's defected variable to true
                // and reset the roundsSinceDefection variable to 0
                defected = true;
                roundsSinceDefection = 0;
                return 1;
            }
        }

        // If neither this player nor its opponents have defected, return 0
        return 0;
    }
}
```

Agent Evaluation Against Original Players

	Score					
	1	**2**	**3**	**4**	**5**	**Average**
Pongpakin	**166.83**	**168.20**	**171.23**	**172.70**	**165.67**	**169.45**
Nice	141.61	158.74	150.70	148.47	151.17	150.14
Nasty	130.20	138.76	140.07	138.95	124.97	134.59
Random	131.06	126.52	144.84	147.80	138.93	137.83
Tolerant	165.84	163.03	167.95	158.79	159.74	163.07
Freaky	139.79	135.21	158.94	154.32	136.22	144.90
T4T	152.10	157.97	165.20	167.38	154.97	159.52

Based on the evaluation, the implemented agent consistently achieved the highest scores when pitted against the original players.

```
Tournament Results
Lumlertluksanachai_Pongpakin_Player: 166.82657 points.
TolerantPlayer: 165.83577 points.
T4TPlayer: 152.09682 points.
NicePlayer: 141.61478 points.
FreakyPlayer: 139.793 points.
RandomPlayer: 131.0611 points.
NastyPlayer: 130.19734 points.
```

```
Tournament Results
Lumlertluksanachai_Pongpakin_Player: 168.1962 points.
TolerantPlayer: 163.03133 points.
NicePlayer: 158.73839 points.
T4TPlayer: 157.97232 points.
NastyPlayer: 138.7638 points.
FreakyPlayer: 135.21072 points.
RandomPlayer: 126.515724 points.
```

```
Tournament Results
Lumlertluksanachai_Pongpakin_Player: 171.22916 points.
TolerantPlayer: 167.94566 points.
T4TPlayer: 165.1987 points.
FreakyPlayer: 158.94334 points.
NicePlayer: 150.6963 points.
RandomPlayer: 144.84218 points.
NastyPlayer: 140.07054 points.
```

```
Tournament Results
Lumlertluksanachai_Pongpakin_Player: 172.7019 points.
T4TPlayer: 167.38477 points.
TolerantPlayer: 158.78854 points.
FreakyPlayer: 154.31602 points.
NicePlayer: 148.46988 points.
RandomPlayer: 147.79555 points.
NastyPlayer: 138.95335 points.
```

```
Tournament Results
Lumlertluksanachai_Pongpakin_Player: 165.67284 points.
TolerantPlayer: 159.73871 points.
T4TPlayer: 154.96613 points.
NicePlayer: 151.17392 points.
RandomPlayer: 138.93327 points.
FreakyPlayer: 136.21664 points.
NastyPlayer: 124.974785 points.
```

Agent Evaluation Against Potential Players

	Score					
	1	**2**	**3**	**4**	**5**	**Average**
Pongpakin	265.34	261.82	**272.78**	264.29	**273.99**	**267.64**
Adaptive TFT	248.77	262.79	260.57	255.98	263.08	258.24
Grim Trigger	**265.40**	262.69	267.64	**275.26**	260.69	266.34
Nice	246.21	249.97	244.24	247.35	249.21	247.40
Nasty	204.56	197.50	183.74	196.04	193.15	195.00
Random	185.50	183.82	183.87	174.94	188.50	183.33
Tolerant	256.69	**273.21**	267.94	261.54	264.44	264.76
Freaky	214.98	216.15	232.88	215.87	203.96	216.77
T4T	260.39	258.73	257.29	253.51	258.16	257.62

Based on the evaluation table, the implemented agent demonstrates a comparable winning ratio against a grim trigger player. Nevertheless, it is noteworthy that the average score of the agent is superior. Nonetheless, there are instances where other players such as a tolerant player emerge victorious. In conclusion, the implemented agent delivers a satisfactory performance against potential players. However, in future improvement, the agent should strive to achieve greater consistency in winning outcomes.

```
Tournament Results
GrimTriggerPlayer: 265.40073 points.
Lumlertluksanachai_Pongpakin_Player: 265.34195 points.
T4TPlayer: 260.38583 points.
TolerantPlayer: 256.69107 points.
AdaptiveT4TPlayer: 248.77046 points.
NicePlayer: 246.2088 points.
FreakyPlayer: 214.97682 points.
NastyPlayer: 204.56369 points.
RandomPlayer: 185.49898 points.
```

```
Tournament Results
TolerantPlayer: 273.20618 points.
AdaptiveT4TPlayer: 262.79462 points.
GrimTriggerPlayer: 262.6859 points.
Lumlertluksanachai_Pongpakin_Player: 261.82156 points.
T4TPlayer: 258.72522 points.
NicePlayer: 249.97078 points.
FreakyPlayer: 216.14584 points.
NastyPlayer: 197.49661 points.
RandomPlayer: 183.82047 points.
```

```
Tournament Results
Lumlertluksanachai_Pongpakin_Player: 272.78076 points.
TolerantPlayer: 267.94446 points.
GrimTriggerPlayer: 267.63934 points.
AdaptiveT4TPlayer: 260.56516 points.
T4TPlayer: 257.29306 points.
NicePlayer: 244.23627 points.
FreakyPlayer: 232.87819 points.
RandomPlayer: 183.86613 points.
NastyPlayer: 183.73842 points.
```

```
Tournament Results
GrimTriggerPlayer: 275.26096 points.
Lumlertluksanachai_Pongpakin_Player: 264.28635 points.
TolerantPlayer: 261.53592 points.
AdaptiveT4TPlayer: 255.97766 points.
T4TPlayer: 253.5063 points.
NicePlayer: 247.34807 points.
FreakyPlayer: 215.87453 points.
NastyPlayer: 196.04106 points.
RandomPlayer: 174.93701 points.
```

```
Tournament Results
Lumlertluksanachai_Pongpakin_Player: 273.99 points.
TolerantPlayer: 264.4394 points.
AdaptiveT4TPlayer: 263.0772 points.
GrimTriggerPlayer: 260.68762 points.
T4TPlayer: 258.1607 points.
NicePlayer: 249.20569 points.
FreakyPlayer: 203.9578 points.
NastyPlayer: 193.15334 points.
RandomPlayer: 188.49858 points.
```