Multi-Agent Pacman:

Question 1: Reflex Agent:

In this question , we have to define the reflex agent evolution function so that pacman defeats ghosts.

We used “***score += successorGameState.getScore()-foodDist(newPos)*0.01***” as the **evolution function** where foodDist(newPos)returns the total distance of the path that pacman traversed in search of food and got it.

We have tested the pacman reflexive agent under few circumstances. Results are as follows-

No of ghosts	Status	Score
1	Win	1331
2	Win	1527
1(directional ghost)	Win	1332
2(directional ghost)	Win	1533
Random seed(using -f) (directional)	Loss	-368
Multiple games in a row(n=2) (directional ghost)	Win Loss	1542 117
Multiple games in a row(n=2) (not directional ghost)	Win Win	1541.0 1534.0
Multiple games in a row(n=3) (directional ghost)	Win Loss win	1716 -279 1528
Multiple games in a row(n=3) (not directional ghost)	Loss Win Win	-386.0 1513.0 1707.0

Question 2 : Minimax:

Here we implement the min_value and max_value which returns minimum and maximum value in the specified depth. As our aim is to win so maxagent is pacman.The agent index of pacman is 0.

Depth	Status	Value at root	Score
1	Loss	9	-493
2	Loss	8	-498
3	Loss	7	-492
4	Win	-492	516
5	Win	-492	514
6	Loss	-492	-492
7	Win	-493	516
8	Win	-493	516
9	Win	-493	516

When we use trapClassic then the pacman always moves to the nearest ghost. Because it got penalty .

<u>Question 3: Alpha-Beta Pruning:</u>

Alpha_ Beta_ Agent is a new agent that uses alpha-beta pruning to explore the minimax tree more efficiently . Thus, it take less time in computation in compare to minmax agent. It prunes unnecessary branches of the game tree.

For default setting:

Depth	Value at root	Status	Score
1	9	Loss	-441
2	8	Win	1314
3	7	Loss	-174
4	-492	Loss	68
5	-492	Loss	-276

We did not reorder the children.We kept it as it is in the minmax problem.

For smallClassic setting:

Depth	Value at root	Status	Score
1	9	Loss	-116
2	8	Loss	-167
3	7	Loss	70
4	-492	Loss	-136
5	-492		

Question 4 : Expectimax:

Expect_max_Agent is the reflexive agent which is useful for modeling probabilistic behavior of agents who may make suboptimal choices.
It calculates the probability with which that action can take place. Action with high probability is preferred.

Depth	Status	Score
1	Win	510
2	Win	516
3	Win	512
4	Win	516
5	Win	514
6	Win	514
7	Win	516

As we proceed according to the probability the chances of winning the game increases.
We observe a more cavalier approach in close quarters with ghosts as follows-
For **alpha beta agent** the table is as follows-

Iteration	Status	Score
1	Loss	-501
2	Loss	-501
3	Loss	-501
4	Loss	-501
5	Loss	-501
6	Loss	-501
7	Loss	-501

8	Loss	-501
9	Loss	-501
10	Loss	-501

Here, the pacman losses all the time in trappedClassic because it approaches to the nearest ghost first.

For **Expect_max_Agent** the table is as follows-

Iteration	Status	Score
1	Loss	-502
2	Win	532
3	Loss	-502
4	Win	532
5	Loss	-502
6	Win	532
7	Loss	-502
8	Loss	-502
9	Win	532
10	Win	532

The win rate of expect_max_Agent is **"Win Rate: 5/10 (0.50)"** .

It happens because alpha beta agent prune the unwanted node but Expect_max_Agent calculates the probability of all the children and then takes decision.

Question 5: Evaluation Function:

In this case, we define our evolution function as follows:

- We calculate all the ghost position on the basis of manhattan distance.
- Based on the closest ghost position, the define a variable safe for the pacman which gives the following penalty.

Distance	Penalty
<10	-0.03
<5	+0.1
<4	+0.5
<3	+1
<2	+2

- Empty spaces around each food is taken into account which allow the pacman to clear food efficiently.
- Calculate the manhattan Distance to any food. We want to maximize Inverse of the closest food distance .
- So , evolution function is defined as follows-
  "**Score+penalty**"

Where score is defined as-

"**score += (min(ghostDistance) * (1.0/min(foodDist)**2) - totalEmpty * 6.5)**"