

60283 Salim Ben Ali

`.gitignore` et taille du dépôt

ok : `.gitignore` adéquat ajouté au 1er commit

tests

- Reflex Agent

```
python ../pacman.py -p ReflexAgent -l testClassic -n 100 -q
```

Winrate : 100%

- Minimax Agent

```
python ../pacman.py -p MinimaxAgent -l minimaxClassic -a depth=5 -n 100 -q
```

Winrate : 61%

- AlphaBeta Agent

```
python ../pacman.py -p AlphaBetaAgent -l smallClassic -a depth=4 -n 10 --frameTime=0 -q
```

Win rate : 60%

- ExpectiMax Agent

```
python ../pacman.py -p ExpectimaxAgent -l minimaxClassic -a depth=3
```

Win Rate : 71%

source

ReflexAgent

- ko : code inutile : `if walls[new_pos[0]][new_pos[1]]:`
- s'approche de la nourriture et s'approche ou s'éloigne de fantômes selon effrayé ou pas
- avec `-p ReflexAgent -l mediumClassic -q -n 100` :

```
Average Score: 1394.13  
Win Rate:      67/100 (0.67)
```

tout à fait raisonnable

MinimaxAgent

q : profondeur adversaires ? 1 niveau == pacman + tous les fantômes

- avec `-p MinimaxAgent -l mediumClassic -q -n 10 -a depth=1` :

```
Average Score: -178.4
```

```
Win Rate:      0/10 (0.00)
```

- avec `-p MinimaxAgent -l mediumClassic -q -n 10 -a depth=2` :

```
Average Score: 241.6
```

```
Win Rate:      2/10 (0.20)
```

ko :

- action : toujours 1er maximum

ok :

- minimax bien implémenté

défense

- ReflexAgent :
 - expliquer `if walls[new_pos[0]][new_pos[1]]` : r : ok étudiant constate que c'est inutile
 - expliquer `score += 10 / (min_food_distance + 1)` r : explications ok
- MinimaxAgent :
 - q : profondeur adversaires ? r : ok
 - expliquer `return minimax(0, 0, game_state)` et les 2 `0` qui y apparaissent. r : ok
 - expliquer `return best_action if depth == 0 else best_value` r : étudiant s'embrouille puis retombe sur ses pattes
 - expliquer pq pacman se bloque en haut à droite avec `-p MinimaxAgent -l mediumClassic -- frameTime=0 -a depth=1` r : pas demandé

conclusion : explications convaincantes
