

✓ 1: initial_state() → tuple (playerlist, boardlist)
+ player: list(1, 2, 3,...) + board: 1Dlist (board_cell stored in it, length = total number of grids in the board)
+ user_initialized (hp+gold+location) + board_initialized (1stD: initialized to be 0 2ndD: randomized number from ? 25 to 40)

player - nextround_order
↓
move_one_step
↓
check_valid_action
↓
perform_steps

string_of_board_index

string_of_board()
- board: list
+ board: string
+ print_out_board_in_string_type()

(特效) print out grids' index
e.g. grid=28, then print "00"->"21"

✓ 2: drop_lucky_box (state: tuple) → List board list

- board: list
+ board: list
+ choose_n_spot_to_drop (? randomized from #of grids / 5 to #of grids / 8)
+ update_board()

✓ 3: move_one_step()

- board: list
+ roll_dice_in_random_order_for_first_round() + roll_dice_based_on_last_round_dice_# (the user got the larger dice # last round, go earlier) + check_game_over()

4. perform_steps()

- board: list
+ iterate until game over while(!game_over()){ move_one_step() }

✓ game_over (state: tuple) → bool

if game over: True, else: False
(特效) rule: 0 hp, out immediately
(no matter how much gold left)

- user_gold: int
+ game_over: bool
+ check_if_is_game_over (1. if any user has 0 hp & cant pay tolls, then, game over immediately 2. if any user cant pay the tolls after pay_tolls(), game also over)

settlement()

- user_gold: int
+ transfer_hp_to_gold (hp:gold != 1:15) + update_player_attribute

winner()

- player: list
+ player: int
+ print_out_winner()

- problems
- ☐ the type of board
 - ☐ number of users (先2个)
 - ☐ create a user object
 - ☐ output type of the board
 - ☐ check valid action return type
 - ☒ how to iterate to play

✓ BoardCell

+ luckybox: tuple(int, int, int) (0-no lucky box dropped here or 1-lucky box dropped here) (0-no one visited yet. 1-hp, 2-gold) (amount)
+ occupied: tuple(int, int, int) (0-no one visited yet, 1,2,3-corresponding player) (0-no one visited yet. 1-hp, 2-gold) (amount)
print_cell_in

✓ Player

+ gold: int (?= 50) + hp: int (?= 5) + location: int (=0)
check_valid_action() move_one_step()

4. check_valid_action()

- board: list
+ check_what_can_user_do_now (open lucky box / occupy grid / pay rolls)

open_lucky_box()

- board: list
+ board: list
+ randomized_gold_in_box() + add_gold_to_the_player_stepped_here (randomized number from ? 0 to 10) + update_board() (remove current spot box + regenerate a new box elsewhere)

grid_occupy()

- board: list
+ board: list
+ decide_to_take_hp_or_gold_from_others_pass_here()

pay_tolls()

- board: list
+ user_gold: int + user_hp: int
+ exchange_hp_to_gold_pay_tolls() + deduct_hp_or_gold_to_current_user() + add_hp_or_gold_to_the_owner_of_the_grid()

新增:

player - nextround_order (roundnum: int) → tuple

rule: 谁点数大, 谁下一轮先走
• 点数一样时, 默认按 index
从小到大走

(__, __)
↑ ↓
roundnum 本轮 roll
下一轮顺序 的点数

e.g. roundnum = 0. → ([0, 1], []) . roundnum = 1 → ([1, 0], [4, 2])
随机的