

CTE206 - Homework 3

Halil Özmen

1. Monopoly Game Data Structures

The positions (squares) on the board are numbered from 0 to 39. The starting position (Go) is 0, "Mediterranean Avenue" is 1, etc., "Broadwalk" is 39.

Player data structure:

An **array of structures** will be used to hold the players' information.

The player structure will be as follows:

Data item	Data Type	Description
player_id	int	Values starting from 1. 1, 2, 3, 4, etc.
position	int	The position on the board: a value from 0 to 39.
capital	int	Amount of money the player has
wait_turns	int	Number of turns the player has to wait. Value from 0 to 3.
leave_prison	int	The number of leave prison cards that the player has. 0, 1, 2.
status	int	1= in the game; 0= quit the game (bankrupted).

When a player plays, the data structure for that player will be updated. In some cases (paying a rent to another player), other players structures may need to be updated. When a player buys a property

Property data structure:

An **array of structures** will be used to hold the information about the properties.

The property structure will be as follows:

Data item	Data Type	Description
property_name	string	Name of the property
property_type	int	1= Avenue, 2= Railroad, 3= Electric Company, 4= Water Works.
position	int	The position of the property on the board: a value from 0 to 39.
neighbourhood_no	int	For avenues, the neighborhood number: 1 to 8.
player_id (owner)	int	The id of player who owns it. Zero (0) if not owned by any player.
num_houses	int	Number of houses, 0 to 5. (5 means hotel)
mortgage_status	int	0= Not mortgaged; 1= Mortgaged.
price	int	Price to buy the property
price_per_house	int	Price to build one house on the property
rent	int	Rent when there are no house(s) or hotel.
rent_1_house	int	Rent when there is 1 house
rent_2_house	int	Rent when there is 2 house
rent_3_house	int	Rent when there is 3 house
rent_4_house	int	Rent when there is 4 house
rent_hotel	int	Rent when there is hotel
mortgage	int	The mortgage value

Chance and Community Chest Cards:

Both the chance cards and community chest cards will be "**queue**" data structures. When a card is processed (leaves the queue), it will again be inserted to the queue.

2. Monopoly Algorithms

Main Algorithm:

```

MAIN
Begin
  Do Initializations // Set number of players, initialize players' data, initialize property values, etc.

  While (Number of active players > 1)
    For p = 1 to number_of_players
      if player's status is 1 (in the game)
        Do player_play
      end if
    End for
  End While

End Main.

```

Player_play Algorithm:

```

Function player_play
Begin
  if player owns a neighborhood that are not full of hotels
    if player has enough money to put house(s)
      Input wish for puting houses
      if player wants to put houses
        Do put_houses
      end if
    end if
  end if

  if wait_turns of the player > 0
    if player has a "Leave Prison" card
      set wait_turns to 0
      Put "Leave Prison" card to chance or community cect cards queue
    end if
  end if

  if wait_turns of the player == 0
    number_of_dice_throws = 0
    Do
      number_of_dice_throws ++
      Throw dice
      if dice is a pair and number_of_dice_throws == 3
        Go to jail and set wait_turns to 3
      else
        Do advance_operations
      end if
      While dice is a pair and number_of_dice_throws < 3
    end if
  end if
End player_play.

```

Advance_operations Algorithm:

```

Function advance_operations
Begin
  Advance // position += dice_value
  if passed through or at "Go" (square 0) // position >= 40

```

```

        Add 200 to player's capital.           // also: position = position % 40
    end if
    if position is a Chance square
        Do chance_operations
    elseif position is a Community Chest square
        Do community_chest_operations
    elseif position is "Go to jail" (square 30) // position == 30
        go to jail (square 10)
        set wait_turns to 3
    elseif position is a tax square
        Do Pay_Tax (deduct tax money from player and add tax money to "collected tax money")
    elseif position is "Free Parking" (square 20) // position == 20
        Add collected tax money to capital
        Set collected tax money to 0
    elseif position is at a property belonging to another player
        Do rent_operations
    elseif position is at a property belonging to nobody
        if player has enough money to buy property
            Input wish for buying property
            if player wants to buy property
                Do buy_property
            end if
        end if
    end if
end if
End advance_operations.

```

Buy_property Algorithm:

```

Function buy_property
Begin
    Deduct property's price from player's capital
    Set player_id (owner) field of the property to player_id of the player
End buy_property

```

Rent_operations Algorithm:

```

Function rent_operations
Begin
    if player's capital >= rent value
        Do pay_rent
    else
        if player has hotels or houses
            Input decision about to mortgage property or sell houses
            if player wants to mortgage property
                Mortgage property (or properties) (set mortgage_status to 1)
                Add mortgage value(s) to player's capital
                if player's capital >= rent value
                    Do pay_rent
                else
                    Do sell_house_operations
                    if player's capital >= rent value
                        Do pay_rent
                    else
                        Do pay_all_capital
                        Set player's status to 0 (bankrupted)
                    end if
                end if
            end if
        else
            Do sell_house_operations
            if player's capital >= rent value

```

```

        Do pay_rent
    else
        Mortgage property (or properties) (set mortgage_status to 1)
        Add mortgage value(s) to player's capital
        if player's capital >= rent value
            Do pay_rent
        else
            Do pay_all_capital
            Set player's status to 0 (bankrupted)
        end if
    end if
end if
else // player do not have hotels or houses
    Mortgage property (or properties) (set mortgage_status to 1)
    Add mortgage value(s) to player's capital
    if player's capital >= rent value
        Do pay_rent
    else
        Do pay_all_capital
        Set player's status to 0 (bankrupted)
    end if
end if
end if
End rent_operations.

```

Pay_rent Algorithm:

```

Function pay_rent
Begin
    deduct rent value from player's capital
    add rent value to the property owner's capital
End pay_rent

```

Pay_all_capital Algorithm:

```

Function pay_all_capital
Begin
    Add player's capital value to the property owner's capital
    Set player's capital to 0
End pay_all_capital

```