```cpp
#include <string>
#include "RatATat.h"
#include "Player.h"
#include "Pile.h"
#include "Card.h"


#define _CRTDBG_MAP_ALLOC
#include <crtdbg.h>
#ifdef _DEBUG
#ifndef DBG_NEW
#define DBG_NEW new ( _NORMAL_BLOCK , __FILE__ , __LINE__ )
#define new DBG_NEW
#endif
#endif // _DEBUG


RatATat::RatATat() : m_number_of_players(0)
{
    m_players_arr = new Player * [m_max_number_of_players]; //! "m_players_arr = new Player[m_number_of_players];" no default ctor exist for class player!!! , so - i made array of pointers to players!!.
    memset(m_players_arr, NULL, m_max_number_of_players * sizeof(Player*));

    m_Players_with_min_hand_sum = new Player * [m_max_number_of_players]; //! "m_players_arr = new Player [m_number_of_players];" no default ctor exist for class player!!! , so - i made array of pointers to players!!.
    memset(m_Players_with_min_hand_sum, NULL, m_max_number_of_players * sizeof(Player*));

    m_unused_cards_pile = new Pile;
    m_unused_cards_pile->new_pile();
    m_unused_cards_pile->shuffle();
    m_thrown_cards_pile = new Pile;

    //m_tmp_card_in_the_air = new Card;
    m_whos_turn_indx = 0;
    m_someone_shout_RatATat = false;

    //because yael ask dealer will play first , i will create him first !
    add_player("Dealer");
}

RatATat::~RatATat()
{
    //Delete all created dynamic allocations:
    for (int i = 0; i < m_number_of_players; i++) //also can be i < m_max_number_of_players
        delete m_players_arr[i];
    delete[] m_players_arr;
```

```cpp

    delete[] m_Players_with_min_hand_sum;


    delete m_unused_cards_pile;
    delete m_thrown_cards_pile;

    //delete m_tmp_card_in_the_air;
}

bool RatATat::add_player(const string& name)
{
    bool m_there_is_a_place_for_more_player = (m_number_of_players < m_max_number_of_players);
    if (m_there_is_a_place_for_more_player)
    {
        if (name == "Dealer")
            m_players_arr[m_number_of_players] = new AI_player("Dealer", this);
        else
            m_players_arr[m_number_of_players] = new Human_player(name, this);
        m_number_of_players++;
        return true;
    }
    cout << "Can't add player!" << endl;
    return false;
}

void RatATat::play()
{
    //its to make sure the game wont started with just 1 player:
    if (m_number_of_players == 1)
    {
        add_player("Dealer");
        //swap_players(Player) // swap(player[0], players[1])
    }
    cout << "Welcome to RatATat game by Or&Ido!" << endl;
    cout << "Creating a game... " << endl << endl;
    //m_someone_shout_RatATat = false;
    card_distribution();
    show_each_player_his_cards();
    //m_whos_turn_indx = 0;
    while (!m_someone_shout_RatATat)
    {
        m_whos_turn_indx = m_whos_turn_indx % m_number_of_players;
        play_turn();
        m_whos_turn_indx++;
    }
    //after someone shouted Ratatat there is 1 more play_turn to each player! (more (m_number_of_players-1) turns)
    for (int j = 0; j < (m_number_of_players - 1); j++)
    {
```

```
        m_whos_turn_indx = m_whos_turn_indx % m_number_of_players;
        play_turn();
        m_whos_turn_indx++;
    }
    //everyone show his hand
    for (int show_hand_turn = 0; show_hand_turn < m_number_of_players; show_hand_turn++)
    {
        m_players_arr[show_hand_turn]->show_player_hand_and_make_it_valid_to_calc_sum();
    }
    calc_players_hands();
    who_won();
}

void RatATat::card_distribution()
{
    int j = 0;
    for (int i = 0; i < (m_number_of_players * (Player::m_player_number_of_cards)); i++)
    {
        j = i % m_number_of_players;
        Card* tmp = get_card_from_unused_pile();
        m_players_arr[j]->put_card_to_player_hand(tmp);
        tmp = nullptr;
    }
}

void RatATat::show_each_player_his_cards()
{
    int tmp;
    for (int i = 0; i < m_number_of_players; i++)
    {
        cout << *m_players_arr[i];
        cout << endl;
    }
    //clear screen:
    //cout << "press any key to clear screen" << endl;
    //cin >> tmp;
    ////to add function to clear screen
}

void RatATat::play_turn()
{
    //if discard pile is empty so making discard pile to be valid to play with
    while (m_thrown_cards_pile->is_empty())
    {
        throw_card_to_discard_pile(get_card_from_unused_pile());
        if (m_thrown_cards_pile->back().is_special_card())
        {
            m_unused_cards_pile->push_front(m_thrown_cards_pile->pop_back());
```

```
            m_unused_cards_pile->shuffle();
        }
    }
    print_top_card_of_discard_pile();
    print_whos_turn();
    player_do_your_move();
    if (!m_someone_shout_RatATat)
        player_set_ratatat_flag();
    cout << "--------------------------" << endl;
}

void RatATat::print_top_card_of_discard_pile()
{
    cout << "==========================" << endl;
    cout << " Discard Pile:    ";
    if (m_thrown_cards_pile->is_empty())
        cout << "Discard Pile is empty" << endl;
    else
        cout << m_thrown_cards_pile->back() << endl;
    cout << "==========================" << endl;
}

void RatATat::print_whos_turn()
{
    cout << "It's " << m_players_arr[m_whos_turn_indx]->get_player_name()
      << " turn:" << endl;
}

void RatATat::player_do_your_move()
{
    int answer;
    int number_of_option = 2;
    if (m_thrown_cards_pile->back().is_special_card())
    {
        player_pick_from_unused_cards_pile();
    }
    else
    {
        asking_player_from_what_pile_to_pick();
        answer = m_players_arr[m_whos_turn_indx]-
          >player_choice_from_what_pile_to_pick(number_of_option);
        if (answer == 1)
        {
            player_pick_from_unused_cards_pile();
        }
        else
        {
            player_pick_from_thrown_cards_pile();
        }
    }
}

void RatATat::player_pick_from_unused_cards_pile()
```

```
{
    Card* m_tmp_card_in_the_air;
    int answer;
    cout << "Picking from UNUSED CARDS PILE" << endl;
    m_tmp_card_in_the_air = get_card_from_unused_pile();
    cout << m_players_arr[m_whos_turn_indx]->get_player_name() << " picked : " << *m_tmp_card_in_the_air << endl;
    m_tmp_card_in_the_air->print_card_action_menu();
    answer = m_players_arr[m_whos_turn_indx]->ChooseOption_from_card_menu(*m_tmp_card_in_the_air);
    if (answer == 1)
    {
        cout << m_players_arr[m_whos_turn_indx]->get_player_name() << " throwing " << *m_tmp_card_in_the_air << " to discard cards pile" << endl;
        throw_card_to_discard_pile(m_tmp_card_in_the_air);
    }
    else //if answer is 2.
    {
        cout << m_players_arr[m_whos_turn_indx]->get_player_name() << " using " << *m_tmp_card_in_the_air << endl;
        m_tmp_card_in_the_air->use(m_players_arr, m_whos_turn_indx, *this);
    }
}

void RatATat::player_pick_from_thrown_cards_pile()
{
    Card* m_tmp_card_in_the_air;
    m_tmp_card_in_the_air = m_thrown_cards_pile->pop_back();
    cout << "Picking from DISCARD CARDS PILE" << endl;
    cout << m_players_arr[m_whos_turn_indx]->get_player_name() << " picked : " << *m_tmp_card_in_the_air << endl;
    cout << m_players_arr[m_whos_turn_indx]->get_player_name() << " using " << *m_tmp_card_in_the_air << endl;
    m_tmp_card_in_the_air->use(m_players_arr, m_whos_turn_indx, *this);
}

//work , but with bug (if took from discard i still can throw again and its not good!.
//void RatATat::player_do_your_move()
//{
//  int answer;
//  int number_of_option = 2;
//  Card* m_tmp_card_in_the_air;
//  if (!m_thrown_cards_pile->is_empty() && !((m_thrown_cards_pile->back()).is_special_card()))
//  {
//      asking_player_from_what_pile_to_pick();
//      answer = m_players_arr[m_whos_turn_indx]->player_choice_from_what_pile_to_pick(number_of_option);
//      if (answer == 1)
//      {
```

```
//          m_tmp_card_in_the_air = get_card_from_unused_pile();
//          cout << "Picking from UNUSED CARDS PILE" << endl;
//      }
//      else
//      {
//          m_tmp_card_in_the_air = m_thrown_cards_pile->pop_back();
//          cout << "Picking from DISCARD CARDS PILE" << endl;
//      }
//  }
//  else
//  {
//      cout << "Picking from UNUSED CARDS PILE" << endl;
//      m_tmp_card_in_the_air = get_card_from_unused_pile();
//  }
//  cout << m_players_arr[m_whos_turn_indx]->get_player_name() << " picked : " << *m_tmp_card_in_the_air << endl;
//  //m_tmp_card_in_the_air->print_card_action_menu();
//  m_tmp_card_in_the_air->print_card_action_menu();
//  answer = m_players_arr[m_whos_turn_indx]->ChooseOption_from_card_menu(*m_tmp_card_in_the_air);
//  if (answer == 1)
//  {
//      cout << m_players_arr[m_whos_turn_indx]->get_player_name() << " throwing " << *m_tmp_card_in_the_air << " to discard cards pile" << endl;
//      throw_card_to_discard_pile(m_tmp_card_in_the_air);
//  }
//  else
//  {
//      cout << m_players_arr[m_whos_turn_indx]->get_player_name() << " using " << *m_tmp_card_in_the_air << endl;
//      m_tmp_card_in_the_air->use(m_players_arr, m_whos_turn_indx, *this);
//  }
//}

Card* RatATat::get_card_from_unused_pile()
{
    if (m_unused_cards_pile->is_empty())
    {
        for (int i = 0; i < m_thrown_cards_pile->size(); i++)
            m_unused_cards_pile->push_front(m_thrown_cards_pile->pop_back());
        m_unused_cards_pile->shuffle();
    }
    return m_unused_cards_pile->pop_back();
}

void RatATat::throw_card_to_discard_pile(Card* card)
{
    m_thrown_cards_pile->push_back(card);
}


```

```cpp
void RatATat::asking_player_from_what_pile_to_pick()
{
    cout << "From what pile do you want to pick a card?" << endl;
    cout << "1.From pile" << endl;
    cout << "2.From discard pile" << endl;
}

void RatATat::player_set_ratatat_flag()
{
    int number_of_options = 2;
    cout << "Call RatATat ?" << endl;
    cout << "1. No" << endl;
    cout << "2. Yes" << endl;
    m_someone_shout_RatATat = (m_players_arr[m_whos_turn_indx]->player_answer_to_call_ratatat(number_of_options) - 1);
}

void RatATat::calc_players_hands()
{
    for (int i = 0; i < m_number_of_players; i++)
        m_players_arr[i]->calc_sum_of_player_hand();
}

void RatATat::who_won()
{
    int min_hand_sum = find_min_players_hand_sum();
    int number_of_players_with_min_sum = 0;
    for (int i = 0; i < m_number_of_players; i++)
        if (m_players_arr[i]->get_player_cards_sum() == min_hand_sum)
        {
            m_Players_with_min_hand_sum[number_of_players_with_min_sum] = m_players_arr[i];
            number_of_players_with_min_sum++;
        }
    cout << "The winner is :" << endl;
    for (int i = 0; i < number_of_players_with_min_sum; i++)
    {
        //m_Players_with_min_hand_sum[i]->show_player_hand_and_make_it_valid_to_calc_sum();
        cout << m_Players_with_min_hand_sum[i]->get_player_name() << " , with sum of : " << min_hand_sum << endl;
    }
}

int RatATat::find_min_players_hand_sum()
{
    int min_sum = m_players_arr[0]->get_player_cards_sum();
    for (int i = 1; i < m_number_of_players; i++)
        if (m_players_arr[i]->get_player_cards_sum() < min_sum)
            min_sum = m_players_arr[i]->get_player_cards_sum();
    return min_sum;
}

```

```
int RatATat::get_number_of_players()
{
    return m_number_of_players;
}

int RatATat::get_whos_turn_indx()
{
    return m_whos_turn_indx;
}

////////////////////////////////////////////////////

```