

```

1  #include <string>
2  #include "RatATat.h"
3  #include "Player.h"
4  #include "Pile.h"
5  #include "Card.h"
6
7
8  #define _CRTDBG_MAP_ALLOC
9  #include <crtdbg.h>
10 #ifdef _DEBUG
11 #ifndef DBG_NEW
12 #define DBG_NEW new ( _NORMAL_BLOCK , __FILE__ , __LINE__ )
13 #define new DBG_NEW
14 #endif
15 #endif // _DEBUG
16
17
18 RatATat::RatATat() : m_number_of_players(0)
19 {
20     m_players_arr = new Player * [m_max_number_of_players]; //!           ↗
21     "m_players_arr = new Player[m_number_of_players];" no default ctor   ↗
22     exist for class player!!! , so - i made array of pointers to         ↗
23     players!!.
24     memset(m_players_arr, NULL, m_max_number_of_players * sizeof         ↗
25         (Player*));
26
27     m_Players_with_min_hand_sum = new Player *                             ↗
28         [m_max_number_of_players]; //! "m_players_arr = new Player       ↗
29         [m_number_of_players];" no default ctor exist for class player!!! , ↗
30         so - i made array of pointers to players!!.
31     memset(m_Players_with_min_hand_sum, NULL, m_max_number_of_players *   ↗
32         sizeof(Player*));
33
34     m_unused_cards_pile = new Pile;
35     m_unused_cards_pile->new_pile();
36     m_unused_cards_pile->shuffle();
37     m_thrown_cards_pile = new Pile;
38
39     //m_tmp_card_in_the_air = new Card;
40     m_whos_turn_indx = 0;
41     m_someone_shout_RatATat = false;
42
43     //because yael ask dealer will play first , i will create him first !
44     add_player("Dealer");
45 }
46
47 RatATat::~RatATat()
48 {
49     //Delete all created dynamic allocations:
50     for (int i = 0; i < m_number_of_players; i++) //also can be i <      ↗
51         m_max_number_of_players
52         delete m_players_arr[i];
53     delete[] m_players_arr;

```

```
45
46     delete[] m_Players_with_min_hand_sum;
47
48
49     delete m_unused_cards_pile;
50     delete m_thrown_cards_pile;
51
52     //delete m_tmp_card_in_the_air;
53 }
54
55 bool RatATat::add_player(const string& name)
56 {
57     bool m_there_is_a_place_for_more_player = (m_number_of_players <
58         m_max_number_of_players);
59     if (m_there_is_a_place_for_more_player)
60     {
61         if (name == "Dealer")
62             m_players_arr[m_number_of_players] = new AI_player("Dealer",
63                 this);
64         else
65             m_players_arr[m_number_of_players] = new Human_player(name,
66                 this);
67         m_number_of_players++;
68         return true;
69     }
70     cout << "Can't add player!" << endl;
71     return false;
72 }
73
74 void RatATat::play()
75 {
76     //its to make sure the game wont started with just 1 player:
77     if (m_number_of_players == 1)
78     {
79         add_player("Dealer");
80         //swap_players(Player) // swap(player[0], players[1])
81     }
82     cout << "Welcome to RatATat game by Or&Ido!" << endl;
83     cout << "Creating a game... " << endl << endl;
84     //m_someone_shout_RatATat = false;
85     card_distribution();
86     show_each_player_his_cards();
87     //m_whos_turn_indx = 0;
88     while (!m_someone_shout_RatATat)
89     {
90         m_whos_turn_indx = m_whos_turn_indx % m_number_of_players;
91         play_turn();
92         m_whos_turn_indx++;
93     }
94     //after someone shouted Ratat there is 1 more play_turn to each
95     player! (more (m_number_of_players-1) turns)
96     for (int j = 0; j < (m_number_of_players - 1); j++)
97     {
```

```
194         m_whos_turn_indx = m_whos_turn_indx % m_number_of_players;
195         play_turn();
196         m_whos_turn_indx++;
197     }
198     //everyone show his hand
199     for (int show_hand_turn = 0; show_hand_turn < m_number_of_players;  ↗
        show_hand_turn++)
200     {
201         m_players_arr[show_hand_turn] - ↗
            >show_player_hand_and_make_it_valid_to_calc_sum();
202     }
203     calc_players_hands();
204     who_won();
205 }
206
207 void RatATat::card_distribution()
208 {
209     int j = 0;
210     for (int i = 0; i < (m_number_of_players * ↗
        (Player::m_player_number_of_cards)); i++)
211     {
212         j = i % m_number_of_players;
213         Card* tmp = get_card_from_unused_pile();
214         m_players_arr[j]->put_card_to_player_hand(tmp);
215         tmp = nullptr;
216     }
217 }
218
219 void RatATat::show_each_player_his_cards()
220 {
221     int tmp;
222     for (int i = 0; i < m_number_of_players; i++)
223     {
224         cout << *m_players_arr[i];
225         cout << endl;
226     }
227     //clear screen:
228     //cout << "press any key to clear screen" << endl;
229     //cin >> tmp;
230     ////to add function to clear screen
231 }
232
233 void RatATat::play_turn()
234 {
235     //if discard pile is empty so making discard pile to be valid to play  ↗
        with
236     while (m_thrown_cards_pile->is_empty())
237     {
238         throw_card_to_discard_pile(get_card_from_unused_pile());
239         if (m_thrown_cards_pile->back().is_special_card())
240         {
241             m_unused_cards_pile->push_front(m_thrown_cards_pile->pop_back  ↗
                ());
```

```
142         m_unused_cards_pile->shuffle();
143     }
144 }
145 print_top_card_of_discard_pile();
146 print_whos_turn();
147 player_do_your_move();
148 if (!m_someone_shout_RatATat)
149     player_set_ratatat_flag();
150 cout << "-----" << endl;
151 }
152
153 void RatATat::print_top_card_of_discard_pile()
154 {
155     cout << "=====" << endl;
156     cout << " Discard Pile: ";
157     if (m_thrown_cards_pile->is_empty())
158         cout << "Discard Pile is empty" << endl;
159     else
160         cout << m_thrown_cards_pile->back() << endl;
161     cout << "=====" << endl;
162 }
163
164 void RatATat::print_whos_turn()
165 {
166     cout << "It's " << m_players_arr[m_whos_turn_indx]->get_player_name()
167         << " turn:" << endl;
168 }
169
170 void RatATat::player_do_your_move()
171 {
172     int answer;
173     int number_of_option = 2;
174     if (m_thrown_cards_pile->back().is_special_card())
175     {
176         player_pick_from_unused_cards_pile();
177     }
178     else
179     {
180         asking_player_from_what_pile_to_pick();
181         answer = m_players_arr[m_whos_turn_indx]-
182             >player_choice_from_what_pile_to_pick(number_of_option);
183         if (answer == 1)
184         {
185             player_pick_from_unused_cards_pile();
186         }
187         else
188         {
189             player_pick_from_thrown_cards_pile();
190         }
191     }
192 }
193
194 void RatATat::player_pick_from_unused_cards_pile()
```

```

193 {
194     Card* m_tmp_card_in_the_air;
195     int answer;
196     cout << "Picking from UNUSED CARDS PILE" << endl;
197     m_tmp_card_in_the_air = get_card_from_unused_pile();
198     cout << m_players_arr[m_whos_turn_idx]->get_player_name() << "
        picked : " << *m_tmp_card_in_the_air << endl;
199     m_tmp_card_in_the_air->print_card_action_menu();
200     answer = m_players_arr[m_whos_turn_idx]->ChooseOption_from_card_menu
        (*m_tmp_card_in_the_air);
201     if (answer == 1)
202     {
203         cout << m_players_arr[m_whos_turn_idx]->get_player_name() << "
            throwing " << *m_tmp_card_in_the_air << " to discard cards pile"
            << endl;
204         throw_card_to_discard_pile(m_tmp_card_in_the_air);
205     }
206     else //if answer is 2.
207     {
208         cout << m_players_arr[m_whos_turn_idx]->get_player_name() << "
            using " << *m_tmp_card_in_the_air << endl;
209         m_tmp_card_in_the_air->use(m_players_arr, m_whos_turn_idx,
            *this);
210     }
211 }
212
213 void RatATat::player_pick_from_thrown_cards_pile()
214 {
215     Card* m_tmp_card_in_the_air;
216     m_tmp_card_in_the_air = m_thrown_cards_pile->pop_back();
217     cout << "Picking from DISCARD CARDS PILE" << endl;
218     cout << m_players_arr[m_whos_turn_idx]->get_player_name() << "
        picked : " << *m_tmp_card_in_the_air << endl;
219     cout << m_players_arr[m_whos_turn_idx]->get_player_name() << " using
        " << *m_tmp_card_in_the_air << endl;
220     m_tmp_card_in_the_air->use(m_players_arr, m_whos_turn_idx, *this);
221 }
222
223 //work , but with bug (if took from discard i still can throw again and
    its not good!.
224 //void RatATat::player_do_your_move()
225 //{
226 //    int answer;
227 //    int number_of_option = 2;
228 //    Card* m_tmp_card_in_the_air;
229 //    if (!m_thrown_cards_pile->is_empty() && !(m_thrown_cards_pile->back
        ()).is_special_card()))
230 //    {
231 //        asking_player_from_what_pile_to_pick();
232 //        answer = m_players_arr[m_whos_turn_idx]-
        >player_choice_from_what_pile_to_pick(number_of_option);
233 //        if (answer == 1)
234 //        {

```

```

235 //          m_tmp_card_in_the_air = get_card_from_unused_pile();
236 //          cout << "Picking from UNUSED CARDS PILE" << endl;
237 //      }
238 //      else
239 //      {
240 //          m_tmp_card_in_the_air = m_thrown_cards_pile->pop_back();
241 //          cout << "Picking from DISCARD CARDS PILE" << endl;
242 //      }
243 //  }
244 //  else
245 //  {
246 //      cout << "Picking from UNUSED CARDS PILE" << endl;
247 //      m_tmp_card_in_the_air = get_card_from_unused_pile();
248 //  }
249 //  cout << m_players_arr[m_whos_turn_idx]->get_player_name() << "      ↗
    picked : " << *m_tmp_card_in_the_air << endl;
250 //  //m_tmp_card_in_the_air->print_card_action_menu();
251 //  m_tmp_card_in_the_air->print_card_action_menu();
252 //  answer = m_players_arr[m_whos_turn_idx]->ChooseOption_from_card_menu      ↗
    (*m_tmp_card_in_the_air);
253 //  if (answer == 1)
254 //  {
255 //      cout << m_players_arr[m_whos_turn_idx]->get_player_name() << "      ↗
    throwing " << *m_tmp_card_in_the_air << " to discard cards pile" << endl;
256 //      throw_card_to_discard_pile(m_tmp_card_in_the_air);
257 //  }
258 //  else
259 //  {
260 //      cout << m_players_arr[m_whos_turn_idx]->get_player_name() << "      ↗
    using " << *m_tmp_card_in_the_air << endl;
261 //      m_tmp_card_in_the_air->use(m_players_arr, m_whos_turn_idx,      ↗
    *this);
262 //  }
263 //}
264
265 Card* RatATat::get_card_from_unused_pile()
266 {
267     if (m_unused_cards_pile->is_empty())
268     {
269         for (int i = 0; i < m_thrown_cards_pile->size(); i++)
270             m_unused_cards_pile->push_front(m_thrown_cards_pile->pop_back      ↗
                ());
271         m_unused_cards_pile->shuffle();
272     }
273     return m_unused_cards_pile->pop_back();
274 }
275
276 void RatATat::throw_card_to_discard_pile(Card* card)
277 {
278     m_thrown_cards_pile->push_back(card);
279 }
280
281

```

```
282 void RatATat::asking_player_from_what_pile_to_pick()
283 {
284     cout << "From what pile do you want to pick a card?" << endl;
285     cout << "1.From pile" << endl;
286     cout << "2.From discard pile" << endl;
287 }
288
289 void RatATat::player_set_ratatat_flag()
290 {
291     int number_of_options = 2;
292     cout << "Call RatATat ?" << endl;
293     cout << "1. No" << endl;
294     cout << "2. Yes" << endl;
295     m_someone_shout_RatATat = (m_players_arr[m_whos_turn_indx] -
                                >player_answer_to_call_ratatat(number_of_options) - 1);
296 }
297
298 void RatATat::calc_players_hands()
299 {
300     for (int i = 0; i < m_number_of_players; i++)
301         m_players_arr[i]->calc_sum_of_player_hand();
302 }
303
304 void RatATat::who_won()
305 {
306     int min_hand_sum = find_min_players_hand_sum();
307     int number_of_players_with_min_sum = 0;
308     for (int i = 0; i < m_number_of_players; i++)
309         if (m_players_arr[i]->get_player_cards_sum() == min_hand_sum)
310         {
311             m_Players_with_min_hand_sum[number_of_players_with_min_sum] =
                 m_players_arr[i];
312             number_of_players_with_min_sum++;
313         }
314     cout << "The winner is : " << endl;
315     for (int i = 0; i < number_of_players_with_min_sum; i++)
316     {
317         //m_Players_with_min_hand_sum[i] -
                 >show_player_hand_and_make_it_valid_to_calc_sum();
318         cout << m_Players_with_min_hand_sum[i]->get_player_name() << " ,
                 with sum of : " << min_hand_sum << endl;
319     }
320 }
321
322 int RatATat::find_min_players_hand_sum()
323 {
324     int min_sum = m_players_arr[0]->get_player_cards_sum();
325     for (int i = 1; i < m_number_of_players; i++)
326         if (m_players_arr[i]->get_player_cards_sum() < min_sum)
327             min_sum = m_players_arr[i]->get_player_cards_sum();
328     return min_sum;
329 }
330
```

```
331 int RatATat::get_number_of_players()
332 {
333     return m_number_of_players;
334 }
335
336 int RatATat::get_whos_turn_indx()
337 {
338     return m_whos_turn_indx;
339 }
340
341 //////////////////////////////////////
342
```