Министерство образования и науки Российской Федерации

ФБГОУ ВПО Волгоградский государственный технический университет

Кафедра «Программное обеспечение автоматизированных систем»

ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

Карточная игра «Покер»

Листов 13

Руководитель работы:

 Нормоконтролер:

к.т.н., доц. каф. ПОАС

 \_\_\_\_\_\_\_\_\_\_\_\_ Кульцова М.Б.

(подпись)

 «\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2014г.

к.т.н., доц. каф. ПОАС

\_\_\_\_\_\_\_\_\_\_\_\_ Кульцова М.Б.

(подпись)

«\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_ 2014г.

Разработчики:

студенты группы ИВТ-460

\_\_\_\_\_\_\_\_\_\_ Козлов Д.В.

 (подпись)

\_\_\_\_\_\_\_\_\_\_ Терехов Г.В.

 (подпись)

«\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_2014 г.

Волгоград 2014

Содержание

[1 Введение 2](#_Toc375340442)

[2 Основание для разработки 2](#_Toc375340443)

[3 Постановка задачи 2](#_Toc375340444)

[3.1 Цель работы 2](#_Toc375340445)

[3.2 Требования к программе 3](#_Toc375340446)

[4 Условия игры 3](#_Toc375340447)

[4.1 Общие сведения 3](#_Toc375340448)

[4.2 Состав игры 3](#_Toc375340449)

[4.3 Ход игры 3](#_Toc375340450)

[4.3.1 Начало игры 3](#_Toc375340451)

[4.3.2 Условия победы 4](#_Toc375340452)

[5 Экранные формы 5](#_Toc375340453)

[6 Управление игроком-человеком 5](#_Toc375340454)

[7 Структура данных 6](#_Toc375340455)

[8 Алгоритм арбитра 6](#_Toc375340456)

[8.1 Начало игры 6](#_Toc375340457)

[8.2 Ходы игроков 6](#_Toc375340458)

[8.3 Конец игры 7](#_Toc375340459)

[9 Стратегия компьютерного игрока 7](#_Toc375340460)

[9.1 Приоритеты комбинаций карт. 7](#_Toc375340461)

[9.2 Начало хода компьютерного игрока 7](#_Toc375340462)

[9.3 После смены карт 9](#_Toc375340463)

[9.4 Предикаты компьютерного противника 9](#_Toc375340464)

[10 Листинг программы 10](#_Toc375340465)

# Введение

Наименование программного продукта – «Карточная игра Покер»

Краткое обозначение наименования программного продукта – «Poker».

Программа предназначена для развития навыков логического мышления.

# Основание для разработки

Основанием для проведения разработки является задание на курсовую работу по дисциплине «Системы Искусственного Интеллекта», выданного доцентом кафедры ПОАС Кульцовой М.Б. 6 ноября 2013 года.

# Постановка задачи

# 3.1 Цель работы

Реализовать карточную игру «Покер» на языке Prolog с искусственным интеллектом.

Для достижения цели был поставлен следующий ряд задач:

- реализовать режим игры человека с компьютером;

- компьютерный игрок должен принимать решения в соответствии с текущей обстановкой в игре;

- арбитр должен следить за игрой и предоставлять ход человеку игроку.

# Требования к программе

Требования к арбитру - должен реализовать следующие условия игры: играют 4 игрока: один игрок управляется человеком, остальные компьютером.

Требования к компьютерному игроку --- должен принимать правильные решения, учитывая следующие критерии:

* оставшееся кол-во игроков;
* состав карт на руках у себя;
* вероятность появления группы карт;
* количество карт, обменянных другими игроками;
* размер ставки и денег у игроков.

# Условия игры

# Общие сведения

Цель игры состоит в том, что бы комбинация карт на руках у игрока имела наивысший ранг.

# Состав игры

В игре принимает участие 4 игрока, из них 3 компьютерных игрока и 1 человек. Игра ведётся колодой из 52 карт.

# Ход игры

# Начало игры

Раздаётся по пять карт на каждого игрока. Игрок делает ставку - кладёт деньги в банк, как бы заключая пари на то, что его комбинация окажется самой старшей. Кто не рискует уравнять ставку партнёра, тот выбывает из борьбы за банк.

Затем происходит смена карт. Причём игрок вправе, как не скидывать карты совсем, так и сменить все. После этого снова происходит торг, по правилам описанным выше.

После этого вскрываются карты, и определяется выигравший игрок, который и берёт себе весь банк. Причем если играющий скинул карты, то они не смотрятся. Если все кроме одного игрока скинули карты, то карты оставшегося участника так же не вскрываются.

# Условия победы

Для определения проигравших игроков определяют приоритет карт, находящихся на руках игроков.

Карты в покере имеют следующую классификацию: покер, четвёрка, флеш-роял, масть, стрит, фул-хаус, тройка, две пары, пара, нулёвка. Наивысший приоритет имеет покер, а наименьший нулёвка.

Покер (royal flush) - карты одной масти, идущие по порядку, начинающиеся с туза, то есть туз, король, дама, валет, десятка.

Четвёрка (four of a kind) - четыре карты одинакового достоинства.

Стрит-флеш (straight flush) - карты одной масти, идущие по порядку. Например, дама, валет, десятка, девятка, восьмёрка.

Масть (flush) - карты одной масти.

Стрит (straight) - карты идущие по порядку. Например, дама, валет, десятка, девятка, восьмёрка.

Фул-хаус (full house) - пара и тройка.

Тройка (three of a kind) - три карты одного достоинства.

Две пары (two pairs) - две пары.

Пара (pair) - две карты одного достоинства.

Нулёвка (highest card) - карты разных мастей, разного достоинства, идущие не по порядку. Если ранг у карт оказывается равным, то расставляют карты по старшинству, учитывая их принадлежность, какому-либо выше перечисленному сочетанию.

# Экранные формы

Графический интерфейс пользователя представлен на рисунке 1.

Рис. 1 Макет графического интерфейса приложения.

# Управление игроком-человеком

Управление игроком-человеком осуществляется путем нажатия мышкой на область игрового поля, на котором нарисованы открытые карты, это карты человека игрока. В случае нажатия на другие зоны игрового поля никаких действий не происходит.

# Структура данных

В программе на прологе имеются следующие структуры данных:

1. card(suit, value);
2. game\_card(card).

В программе на С++ имеются соответствующие структуры данных. Карты имеют достоинство от 2 до 14. Номера (11…14) - валет, дама, король и туз соответственно.

1. enum Suit {

 CLUBS = 0,

 DIAMONDS = 1,

 SPADES = 2,

 HEARTS = 3

 };

1. enum Value {

 TWO = 2,

 THREE = 3,

 FOUR = 4,

 FIVE = 5,

 SIX = 6,

 SEVEN = 7,

 EIGHT = 8,

 NINE = 9,

 TEN = 10,

 JACK = 11,

 QUEEN = 12,

 KING = 13,

 ACE = 14

 };

# Алгоритм арбитра

# Начало игры

Игра начинается при нажатии на кнопку «New game». Игра длится несколько раундов.

# Ходы игроков

Ходы игроков имеют несколько вариантов:

1. игроки делают ставки;
2. игроки меняют карты;
3. игроки открывают карты.

В начале раунда всем игрокам, раздается по пять карт на руки. Игроки делают ставки, заключая пари на то, что их комбинация - самая старшая. Кто не поддерживает ставку - выбывает из борьбы за банк в этом раунде. Игроки меняют карты, при этом могут как не сменить ни одной карты, так и сменить все. Происходит второй тур ставок, по тем же самым правилам, что и первый. Игроки открывают свои карты, игрок с лучшей комбинацией выигрывает банк, на этом раунд заканчивается. Если у игрока еще есть деньги, то начинается новый раунд с шага 1.

# Конец игры

Выигрывает игрок, который по прошествии нескольких раундов выигрывает все деньги остальных игроков. Игрок, потерявший все свои деньги, проигрывает.

# Стратегия компьютерного игрока

# Приоритеты комбинаций карт.

Комбинации карт имеют следующие приоритеты:

10. Покер - карты одной масти, идущие по порядку, начинающиеся с туза.

9. Четверка - 4-е карты одинакового достоинства

8. Стрит-флеш - карты одной масти, идущие по порядку

7. Масть - карты одной масти

6. Стрит - карты, идущие по порядку

5. Фул - пара и тройка

4. Тройка - 3 одного достоинства

3. Две пары

2. Пара - 2-е одного достоинства

1. Нулёвка - карты разных мастей, разного достоинства, идущие не по порядку.

# Начало хода компьютерного игрока

Компьютерный игрок получает на руки 5 карт. Происходит проверка на наличие каких-либо комбинаций на руках. Если на руках нет ни одной из указанных комбинаций, то компьютерный игрок ищет наиболее близкую комбинацию (комбинацию, для получения которой не хватает минимального количества карт).

При нахождении комбинации из ее приоритета вычитается количество карт, которого не хватает для ее получения, при этом карты, необходимые для получения данной комбинации запоминаются.

После выполнения этого шага все комбинации будут отсортированы по приоритетам, которые зависят от легкости получения этой комбинации и от ее "цены".

Потом приоритеты для комбинаций, для получения которых необходимо сменить одни и те же карты, увеличиваются на значение пропорциональное их изначальной стоимости.

Затем выбирается комбинация с наибольшим и наименьшим приоритетом.

Размер ставки также зависит от уровня готовности к риску компьютерного игрока, игрок может сбросить карты, если максимальный приоритет последовательности ниже определенного уровня.

Компьютерный игрок меняет карты, смена которых может привести к нескольким комбинациям с высоким приоритетом.

# После смены карт

Компьютерный игрок проверяет находящиеся у него на руках карты и ищет выигрышные комбинации.

Найдя все возможные комбинации, приносящие очки, игрок выбирает комбинацию, приносящую наибольшее их количество.

Компьютерный игрок знает все карты, которые у него есть и которые были у него на руках, исходя из этого, производится оценка возможных карт, находящихся на руках соперников с использованием колоды, не содержащей использованные карты. Проверяется возможность наличия у соперников комбинаций карт с большей ценностью, чем текущая комбинация на руках у игрока, с учетом того, сколько карт поменял противник.

• Противник поменял три карты и более карт - уменьшается вероятность нахождения на руках у противника всех комбинаций с ценой больше 4.

• Противник поменял две карты - уменьшается вероятность нахождения у противника на руках карт комбинации ценнее 7

• Противник поменял одну или менее карт - рассчитанная вероятность не меняется.

Если приоритет текущей комбинации мал, а вероятность нахождения у противников более ценных комбинаций велика, то игрок может сбросить карты. Если приоритет комбинации велик, то игрок может повысить ставку. Если комбинация имеет средний приоритет, то игрок может уравнять ставку противника.

# Предикаты компьютерного противника

Предикат и его назначение:

1. isOnePair(Hand) - определения наличия пары;
2. isTwoPair(Hand) - определение наличия двух пар;
3. isThreeOfAKind(Hand) - определение наличия тройки;
4. isStraight(Hand) - определение наличия стрита;
5. isFlush(Hand) - определение наличия флеша;
6. isFullHouse(Hand) - попределение наличия фула;
7. isFourOfAKind(Hand) - определения наличия четверки;
8. isStraightFlush(Hand) - определение наличия стрит-флеша;
9. isRoyalFlush(Hand) - определение наличия покера;
10. straight(Hand) - проверка, что карты идут по порядку;
11. which\_rank (Hand,Rank) - ранжирование комбинаий карт;
12. isPairB(Hand1,Hand2) - сравнение пар между собой;
13. isTwoPairB(Hand1,Hand2) - сравнение двух пар между собой;
14. isHigherCardB (Hand1,Hand2) - сравнение двух наибольших карт между собой;
15. isFlushCardB(Hand1,Hand2) - сравнение двух флешей меду собой;
16. isKickerBesideOnePairB(Hand1,Hand2) - сравнение старших карт в парах;
17. isKickerBesideTwoPairB(Hand1,Hand2) - сравнение старших карт в двух парах;
18. isKickerBesideThreeOfAKindB(Hand1,Hand2) - сравнение старших карт в тройках;
19. isThreeOfAKindCardB(Hand1,Hand2) – сравнение двух троек;
20. isFourOfAKindCardB(Hand1,Hand2) – сравнение двух четверок;
21. isStraightEndCardB(Hand1,Hand2) - сравнение самых старших карт в стрите;
22. isStraightFlushEndCardB(Hand1,Hand2) - сравнение самых старших карт в флеше;
23. isHandB(Hand1,Hand2) - сравнение двух комбинаций карт;
24. isHandE(Hand1,Hand2) - проверка колод двух комбинаций карт на равенство.

# Выполняемые задачи

Таблица 1. Выполняемые задачи.

|  |  |  |
| --- | --- | --- |
| Содержание работ на данной стадии | Ответственные за выполнение | Семестр |
| API для Игроков. | Козлов Д.В. | 1 |
| API для Арбитра. | Терехов Г.В. | 1 |
| Игра начинается при нажатии кнопки «Новая игра». | Терехов Г.В. | 1 |
| Создание колоды карт на прологе | Козлов Д.В. | 1 |
| Получение карт из колоды | Терехов Г.В. | 1 |
| В начале раунда всем игрокам раздаётся по 5 карт. | Терехов Г.В. | 1 |
| Игроки делают ставки. | Козлов Д.В. | 1 |
| Кто не поддерживает ставку, выбывает из борьбы за банк в этом раунде. | Терехов Г.В. | 2 |
| Смена карт игроками. | Козлов Д.В. | 2 |
| Второй тур ставок. | Терехов Г.В. | 2 |
| Игроки открывают свои карты. | Терехов Г.В. | 2 |
| Игрок с лучшей комбинацией выигрывает банк, раунд заканчивается. | Терехов Г.В. | 1 |
| Если у игрока ещё есть деньги, то начинается новый раунд. | Терехов Г.В. | 2 |
| Выигрывает игрок, который по прошествии нескольких раундов выигрывает все деньги остальных игроков. | Терехов Г.В. | 2 |
| Игрок, потерявший все свои деньги, проигрывает. | Терехов Г.В. | 2 |
| Проверка на наличие каких-либо комбинаций на руках. | Козлов Д.В. | 1 |
| Если на руках нет ни одной из указанных комбинаций, то компьютерный игрок ищет наиболее близкую комбинацию. | Козлов Д.В. | 2 |
| Определение принятия решения пас/ставка. | Козлов Д.В. | 2 |
| Определение смены карт. | Козлов Д.В. | 2 |
| Действия после смены карт. | Терехов Г.В. | 2 |
| Реализация графического интерфейса. | Терехов Г.В. | 1 |
| Подключение пролога к С++. | Терехов Г.В. | 1 |
| Написание тестов к коду пролога. | Козлов Д.В. | 2 |

# Листинг программы

C++:

Новая игра:

void MainWindow::newGame() {

 setEnabledHumanButtons(true);

 // Удаление текущих игроков

 foreach (Player \* player, players)

 delete player;

 players.clear();

 Player \* hplayer = new HumanPlayer("Human", 100, &scene, PlayerView::BOTTOM);

 Player \* bot1 = new BotPlayer("Bot1", 100, &scene, PlayerView::LEFT);

 Player \* bot2 = new BotPlayer("Bot2", 100, &scene, PlayerView::RIGHT);

 Player \* bot3 = new BotPlayer("Bot3", 100, &scene, PlayerView::TOP);

 players.append(hplayer);

 players.append(bot1);

 players.append(bot2);

 players.append(bot3);

 connect(ui->betSpinBox, SIGNAL(valueChanged(int)), hplayer, SLOT(SetBet(int)));

 connect(ui->betButton, SIGNAL(clicked()), this, SLOT(makeBet()));

 connect(bot1, SIGNAL(BetChanged(int)), ui->cpu1Bet, SLOT(setNum(int)));

 connect(bot2, SIGNAL(BetChanged(int)), ui->cpu2Bet, SLOT(setNum(int)));

 connect(bot3, SIGNAL(BetChanged(int)), ui->cpu3Bet, SLOT(setNum(int)));

 connect(hplayer, SIGNAL(MoneyChanged(int)), ui->manMoney, SLOT(setNum(int)));

 connect(bot1, SIGNAL(MoneyChanged(int)), ui->botMoney\_1, SLOT(setNum(int)));

 connect(bot2, SIGNAL(MoneyChanged(int)), ui->botMoney\_2, SLOT(setNum(int)));

 connect(bot3, SIGNAL(MoneyChanged(int)), ui->botMoney\_3, SLOT(setNum(int)));

 connect(hplayer, SIGNAL(GameOver(Player\*)), this, SLOT(playerPassed(Player\*)));

 connect(bot1, SIGNAL(GameOver(Player\*)), this, SLOT(playerPassed(Player\*)));

 connect(bot2, SIGNAL(GameOver(Player\*)), this, SLOT(playerPassed(Player\*)));

 connect(bot3, SIGNAL(GameOver(Player\*)), this, SLOT(playerPassed(Player\*)));

 connect(ui->passButton, SIGNAL(clicked()), hplayer, SLOT(Pass()));

 connect(this, SIGNAL(humanPassed()), this, SLOT(botGameAfterPass()));

 newRound();

}

Новый раунд:

void MainWindow::newRound() {

 stateSpace.stop();

 defaultOptions();

 ui->betSpinBox->setMaximum(10);

 ui->betSpinBox->setMinimum(0);

 try {

 PlTerm deckID(1l);

 PlCall("create\_poker\_table", deckID);

 } catch (PlException e) {

 QTextStream(stderr) << e.name();

 QMessageBox::warning(NULL, QString("Ошибка!"), QString((char\*)e));

 }

 foreach (Player \* player, players) {

 if (!player->HadLost()) {

 player->ClearTurn();

 for (int i = 0; i < 5; ++i) {

 player->GetHand()->TakeCardFromDeck();

 }

 }

 }

 QTimer::singleShot(120, &stateSpace, SLOT(start()));

}

Определение победителя + победитель забирает банк

void MainWindow::determineWinner() {

 bank = 0;

 foreach (Player \* ply, players)

 if (!ply->HadLost())

 bank += ply->GetBet();

 Player \* best = players.at(0);

 foreach (Player \* ply, players)

 if (!ply->HadPassed()) {

 best = ply;

 break;

 }

 foreach (Player \* ply, players)

 if (!ply->HadLost() && ply->GetHand()->BetterThan(best->GetHand()->GetCards()) && !ply->HadPassed())

 best = ply;

 foreach (Player \* ply, players) {

 if (!ply->HadLost() && !ply->HadPassed() && (ply != best || best->HadPassed())) {

 ui->resultLabel->setText(QString("Потеряно %1$ с комбинацией %2").arg(ply->GetBet()).arg(ply->GetHand()->GetRankingName()));

 ply->LoseBet();

 if (ply->HadLost()) {

 ui->resultLabel->setText("Вы проиграли");

 if (ply == players.at(0)) {

 QMessageBox::information(NULL, "Игра окончена!", "Вы проиграли.");

 }

 }

 }

 }

 if (!best->HadPassed()) {

 if(!best->IsNpc())

 ui->resultLabel->setText(QString("Выиграно %1$ с комбинацией %2").arg(bank - best->GetBet()).arg(best->GetHand()->GetRankingName()));

 else

 ui->resultLabel->setText(QString("Банк %1$ был разыгран с комбинацией %2").arg(bank - best->GetBet()).arg(best->GetHand()->GetRankingName()));

 best->WinBet(bank);

 }

}

Получение карты игроком:
bool Hand::TakeCardFromDeck() {

 QString cardResult;

 QRegExp reg("(hearts|diamonds|clubs|spades)\\s\*,\\s\*([2-9jqka]|10)");

 try {

 PlTermv av(3);

 PlTerm card;

 PlTerm id(1l);

 double randN = double(qrand()) / RAND\_MAX;

 PlTerm randNum(randN);

 av[0] = id;

 av[1] = card;

 av[2] = randNum;

 PlCall("take\_card\_from\_pack", av);

 cardResult = QString((const char \*)card);

 if (cardResult.contains(reg)) {

 QStringList texts = reg.capturedTexts();

 QString suit = texts.at(1);

 QString val = texts.at(2);

 GameCard \* gameCard = GameCard::createFromString(suit, val);

 if (gameCard) {

 AddCard(gameCard);

 }

 return true;

 }

 } catch (PlException &e) {

 QTextStream(stderr) << QString((char \*) e) << "\n";

 }

 return false;

}

Prolog:

Пример инициализации карт в колоде:

game\_card( card(hearts, 6) ).

game\_card( card(diamonds, 3) ).

game\_card( card(clubs, q) ).

game\_card( card(spades, k) ).

Создание игрового поля:

%Создание колоды карт

create\_poker\_table(PackID) :-

 findall( Card, game\_card(Card), Deck ),

 retractall( db\_game\_deck(PackID, \_) ),

 assert( db\_game\_deck(PackID, Deck) ),

 true.

%Вычитание карт из текущей колоды

remove\_cards\_from\_pack(ListCards, Result) :-

 findall(Card, game\_card(Card), Deck),

 subtract(Deck, ListCards, Result),

 true.

take\_card\_from\_pack(PackID, Card, CardNumber) :-

 db\_game\_deck(PackID, Deck),

 length(Deck, DeckLength),

 Index is floor(CardNumber\*DeckLength),

 nth0(Index, Deck, Card),

 select(Card, Deck, RestDeck),

 retractall( db\_game\_deck(PackID, \_) ),

 assert( db\_game\_deck(PackID, RestDeck) ),

 true.

Примеры основных предикатов предикаты:

is\_straight(ListCards):-

 select(Card1, ListCards, RestListCards1),

 select(Card2, RestListCards1, RestListCards2),

 select(Card3, RestListCards2, RestListCards3),

 select(Card4, RestListCards3, RestListCards4),

 select(Card5, RestListCards4, \_),

 straight([Card1, Card2, Card3, Card4, Card5]),

 true.

is\_flush(ListCards):-

 select(card(Colour, \_), ListCards, RestListCards1),

 select(card(Colour, \_), RestListCards1, RestListCards2),

 select(card(Colour, \_), RestListCards2, RestListCards3),

 select(card(Colour, \_), RestListCards3, RestListCards4),

 select(card(Colour, \_), RestListCards4, \_),

 true.

is\_full\_house(ListCards):-

 select(card(\_, Value), ListCards, RestListCards1),

 select(card(\_, Value), RestListCards1, RestListCards2),

 select(card(\_, Value), RestListCards2, RestListCards3),

 select(card(\_, Value2), RestListCards3, RestListCards4),

 select(card(\_, Value2), RestListCards4, \_),

 true.

is\_four\_of\_a\_kind(ListCards):-

 select(card(\_, Value), ListCards, RestListCards1),

 select(card(\_, Value), RestListCards1, RestListCards2),

 select(card(\_, Value), RestListCards2, RestListCards3),

 select(card(\_, Value), RestListCards3, \_),

 true.

Пример тестов:

test\_data(

isOnePair( [

card(hearts,10),

card(spades,5),

card(diamonds,10),

card(clubs,j),

card(diamonds,3),

card(diamonds,a),

card(spades,7)

 ]),

 true

).

test\_data(

isOnePair( [

card(diamonds,10),

 card(spades,5),

card(spades,10),

card(clubs,j),

card(diamonds,3),

card(hearts,10),

card(spades,7)

 ]),

True

 ).

Тестирование:

silentTest :-

 test\_data(X, Boolean), %Возращает полученное и ожидамаемое значения

 testTry(X,Boolean),

 fail.

%Завершить тестирование

silentTest:-

 true.

testTry(X,true) :-

 %write('Ok: '), write(X), nl,

 X, !, true.

testTry(X,true) :- write('Error: '), write(X), nl, true.

testTry(X,fail) :- X, !, write('Error: '), write(X), nl, true.

testTry(\_,fail) :- true.