Министерство науки и образования РФ

Федеральное государственное бюджетное учреждение

высшего образования

**«Тверской государственный технический университет»**

(ТвГТУ)

Кафедра программного обеспечения

**Отчет по практической работе №3**

по дисциплине: «Динамическое программирование»

Тема: «Практическое применение метода динамического программирования на примере реализации игры BlackJack»

|  |
| --- |
| Выполнил:  студент группы  Б.ПИН.РИС – 18.05  Федотов Ярослав |
| Проверила:  ассистент кафедры ПО  Корнеева Е.И. |

Тверь 2019

**Постановка задачи**

## Порядок выполнения первой части задания

* 1. Определить структуры данных для игры в программе (классы, перечисления, массивы и т.д.);
  2. Создать консольное приложение для реализации игры. В консольном приложении реализуем ручное управление игрой. Пример интерфейса (возможна другая реализация): “Ваши карты: Q♠ 9♥ (19). Ваше решение: 1 - hit, любая другая клавиша - stay”

В решение входит:

А) Генерация колоды для игры;

Б) Раздача карт игроку и подсчет очков;

В) Моделирование игрового процесса.

Г) Решение задачи оптимизации. Создать проект с тестами (можно посмотреть существующее решение на языке C# и переписать его на язык Python 3.\*). Реализовать функцию для решения подзадач BJ(i) и решить исходную задачу, вычислив BJ(0) на сгенерированной колоде размера N.

## Порядок выполнения второй части задания

1. Создать визуальный интерфейс, описанный в теории, для простой реализации игры BlackJack (первая часть задания) на PyQt5.

Интерфейс игры сделать по своему усмотрению.

Для визуализации колоды в игре необходимо сделать следующее. Создать две папки с картинками для карт. Выбрать оптимальный размер для отображения карт (все картинки должны растягиваться без потерь для выбранного размера и лучше, если карты будут одинковыми или почти одинаковыми).

А) В первой папке разместить картинки для карт лицевой стороной. Название картинкам сделать с учетом числовой кодировки их масти (0=♠, 1=♣, 2=♦, 3=♥). Например:

cards/...jpg

2\_0.jpg // 2♠

2\_1.jpg // 2♣

2\_2.jpg // 2♦

2\_3.jpg // 2♥

3\_0.jpg // 3♠

…

10\_3.jpg // 10♥

(далее для краткости x - это масть)

11\_x.jpg // A = Ace (туз)

12\_x.jpg // J = Jack (валет)

13\_x.jpg // Q = Queen (королева)

14\_x.jpg // K = King (король)

Б) Вторая папка должна содержать отдельно “рубашки” карт (того же оптимального размера, что и карты) – минимум две разные для смены колоды в программе.

Например:

cover/...jpg

1.jpg // изображение рубашки того же размера, что и карта

2.jpg // альтернативная рубашка

1. \* Добавить в программную реализацию игры расширение (по заданию преподавателя и после реализации первой части задания и пункта 1).

**Алгоритм решения первой части задания:**

импортируем random

money = int(0)

bstavkab = bool(false)

класс card(object):

функция \_\_init\_\_(self, rank, suit):

self.rank = rank

self.suit = suit

функция card\_value(self):

#возращает количество очков которое дает карта

если self.rank in "tjqk":

# по 10 за десятку, валета, даму и короля

return 10

иначе :

# возвращает нужное число очков за любую другую карту

# туз изначально дает одно очко.

return " a23456789".index(self.rank)

функция get\_rank(self):

return self.rank

функция \_\_str\_\_(self):

return "%s%s" % (self.rank, self.suit)

класс hand(object):

функция \_\_init\_\_(self, name):

# имя игрока

self.name = name

# изначально рука пустая

self.cards = []

функция add\_card(self, card):

""" добавляет карту на руку """

self.cards.append(card)

функция get\_value(self):

""" метод получения числа очков на руке """

result = 0

# количество тузов на руке.

aces = 0

для card in self.cards:

result += card.card\_value()

# если на руке есть туз - увеличиваем количество тузов

если card.get\_rank() == "a":

aces += 1

# решаем считать тузы за 1 очко или за 11

если result + aces \* 10 <= 21:

result += aces \* 10

return result

функция \_\_str\_\_(self):

text = "%s содержит:\n" % self.name

для card in self.cards:

text += str(card) + " "

text += "\nколичество очков: " + str(self.get\_value())

return text

класс deck(object):

функция \_\_init\_\_(self):

# ранги

ranks = "23456789tjqka"

# масти

suits = "dchs"

# генератор списков создающий колоду из 52 карт

self.cards = [card(r,s) для r in ranks для s in suits]

# перетасовываем колоду. не забудьте импортировать функцию shuffle из модуля random

random.shuffle(self.cards)

функция deal\_card(self):

""" функция сдачи карты """

return self.cards.pop()

функция home\_money():

# генерируем количество денег у игрока

money = random.randint(1,5)

return money

функция new\_game(money,bstavkab):

# создаем колоду

d = deck()

# задаем "руки" для игрока и дилера

player\_hand = hand("игрок")

dealer\_hand = hand("дилер")

# сдаем две карты игроку

player\_hand.add\_card(d.deal\_card())

player\_hand.add\_card(d.deal\_card())

# сдаем одну карту дилеру

dealer\_hand.add\_card(d.deal\_card())

печать (dealer\_hand)

печать ("="\*20)

печать (player\_hand)

ans = input("решить bj(0) исходную задачу? (y/n)")

Если (ans == "y"):

h = player\_hand.get\_value()

печать (str(h))

печать ("продожим игру...:)")

# флаг проверки необходимости продолжать игру

in\_game = истина

# набирать карты игроку имеет смысл только если у него на руке меньше 21 очка

пока player\_hand.get\_value() < 21:

Если (money > 0):

печать ("ваше текущее количество денег: " + str(money))

Если (bstavkab == false):

stavka = int(input("ваша ставка: "))

bstavkab = истина

Иначе :

ans = input("удвоить ставку?: (y/n)")

Если (ans == "y"):

stavka \*=2

Если (stavka > money and stavka < 0):

печать ("не может быть больше количества ваших денег! или равна 0!")

new\_game(money,bstavkab)

Иначе :

ans = input("взять карту или хватит? (h/s) ")

если ans == "h":

player\_hand.add\_card(d.deal\_card())

печать (player\_hand)

# если у игрока перебор - дилеру нет смысла набирать карты

если player\_hand.get\_value() > 21:

печать ("вы проиграли!")

money -= stavka

in\_game = ложь

печать ("ваше текущее количество денег: " + str(money))

Если (money > 0):

ans = input("дилер: ещё сыграем? (y/n)")

if(ans == "y"):

new\_game(money,bstavkab)

Иначе :

выйти

Иначе :

печать ("вы сказали достаточно!")

печать ("=" \* 20)

если in\_game:

# по правилам дилер обязан набирать карты пока его счет меньше 17

пока dealer\_hand.get\_value() < 17:

dealer\_hand.add\_card(d.deal\_card())

печать (dealer\_hand)

# если у дилера перебор играть дальше нет смысла - игрок выиграл

если dealer\_hand.get\_value() > 21:

печать ("перебор у дилера!")

in\_game = ложь

если in\_game:

# ни у кого не было перебора - сравниваем количество очков у игрока и дилера.

# в нашей версии если у дилера и игрока равное количество очков - выигрывает казино

если player\_hand.get\_value() > dealer\_hand.get\_value():

печать ("вы выйграли!")

money += stavka\*1.5

печать ("ваше текущее количество денег: " + str(money))

Если (money > 0):

ans = input("дилер: ещё сыграем? (y/n)")

Если (ans == "y"):

new\_game(money,bstavkab)

Иначе :

выйти

Иначе :

печать ("дилер выйграл вас!")

money -= stavka

печать ("ваше текущее количество денег: " + str(money))

Если (money > 0):

ans = input("дилер: ещё сыграем? (y/n)")

Если (ans == "y"):

new\_game(money,bstavkab)

Иначе :

выйти

Иначе :

печать ("вы програли все деньги! пора домой за новой партией!")

money = home\_money()

new\_game(money,bstavkab)

money = home\_money()

new\_game(money,bstavkab)

**Программа первой части задания:**

import random

money = int(0)

bstavkab = bool(False)

class Card(object):

def \_\_init\_\_(self, rank, suit):

self.rank = rank

self.suit = suit

def card\_value(self):

#Возращает количество очков которое дает карта

if self.rank in "TJQK":

# По 10 за десятку, валета, даму и короля

return 10

else:

# Возвращает нужное число очков за любую другую карту

# Туз изначально дает одно очко.

return " A23456789".index(self.rank)

def get\_rank(self):

return self.rank

def \_\_str\_\_(self):

return "%s%s" % (self.rank, self.suit)

class Hand(object):

def \_\_init\_\_(self, name):

# имя игрока

self.name = name

# Изначально рука пустая

self.cards = []

def add\_card(self, card):

""" Добавляет карту на руку """

self.cards.append(card)

def get\_value(self):

""" Метод получения числа очков на руке """

result = 0

# Количество тузов на руке.

aces = 0

for card in self.cards:

result += card.card\_value()

# Если на руке есть туз - увеличиваем количество тузов

if card.get\_rank() == "A":

aces += 1

# Решаем считать тузы за 1 очко или за 11

if result + aces \* 10 <= 21:

result += aces \* 10

return result

def \_\_str\_\_(self):

text = "%s Содержит:\n" % self.name

for card in self.cards:

text += str(card) + " "

text += "\nКоличество очков: " + str(self.get\_value())

return text

class Deck(object):

def \_\_init\_\_(self):

# ранги

ranks = "23456789TJQKA"

# масти

suits = "DCHS"

# генератор списков создающий колоду из 52 карт

self.cards = [Card(r,s) for r in ranks for s in suits]

# перетасовываем колоду. Не забудьте импортировать функцию shuffle из модуля random

random.shuffle(self.cards)

def deal\_card(self):

""" Функция сдачи карты """

return self.cards.pop()

def home\_money():

# генерируем количество денег у игрока

money = random.randint(1,5)

return money

def new\_game(money,bstavkab):

# создаем колоду

d = Deck()

# задаем "руки" для игрока и дилера

player\_hand = Hand("Игрок")

dealer\_hand = Hand("Дилер")

# сдаем две карты игроку

player\_hand.add\_card(d.deal\_card())

player\_hand.add\_card(d.deal\_card())

# сдаем одну карту дилеру

dealer\_hand.add\_card(d.deal\_card())

print (dealer\_hand)

print ("="\*20)

print (player\_hand)

ans = input("Решить BJ(0) исходную задачу? (y/n)")

if(ans == "y"):

h = player\_hand.get\_value()

print(str(h))

print("Продожим игру...:)")

# Флаг проверки необходимости продолжать игру

in\_game = True

# набирать карты игроку имеет смысл только если у него на руке меньше 21 очка

while player\_hand.get\_value() < 21:

if(money > 0):

print("Ваше текущее количество денег: " + str(money))

if(bstavkab == False):

stavka = int(input("Ваша ставка: "))

bstavkab = True

else:

ans = input("Удвоить ставку?: (y/n)")

if(ans == "y"):

stavka \*=2

if(stavka > money and stavka < 0):

print("Не может быть больше количества ваших денег! или равна 0!")

new\_game(money,bstavkab)

else:

ans = input("Взять карту или хватит? (h/s) ")

if ans == "h":

player\_hand.add\_card(d.deal\_card())

print (player\_hand)

# Если у игрока перебор - дилеру нет смысла набирать карты

if player\_hand.get\_value() > 21:

print ("Вы проиграли!")

money -= stavka

in\_game = False

print("Ваше текущее количество денег: " + str(money))

if(money > 0):

ans = input("Дилер: Ещё сыграем? (y/n)")

if(ans == "y"):

new\_game(money,bstavkab)

else:

break

else:

print ("Вы сказали достаточно!")

print ("=" \* 20)

if in\_game:

# По правилам дилер обязан набирать карты пока его счет меньше 17

while dealer\_hand.get\_value() < 17:

dealer\_hand.add\_card(d.deal\_card())

print (dealer\_hand)

# Если у дилера перебор играть дальше нет смысла - игрок выиграл

if dealer\_hand.get\_value() > 21:

print ("Перебор у дилера!")

in\_game = False

if in\_game:

# Ни у кого не было перебора - сравниваем количество очков у игрока и дилера.

# В нашей версии если у дилера и игрока равное количество очков - выигрывает казино

if player\_hand.get\_value() > dealer\_hand.get\_value():

print ("Вы выйграли!")

money += stavka\*1.5

print("Ваше текущее количество денег: " + str(money))

if(money > 0):

ans = input("Дилер: Ещё сыграем? (y/n)")

if(ans == "y"):

new\_game(money,bstavkab)

else:

break

else:

print ("Дилер выйграл вас!")

money -= stavka

print("Ваше текущее количество денег: " + str(money))

if(money > 0):

ans = input("Дилер: Ещё сыграем? (y/n)")

if(ans == "y"):

new\_game(money,bstavkab)

else:

break

else:

print("Вы програли все деньги! Пора домой за новой партией!")

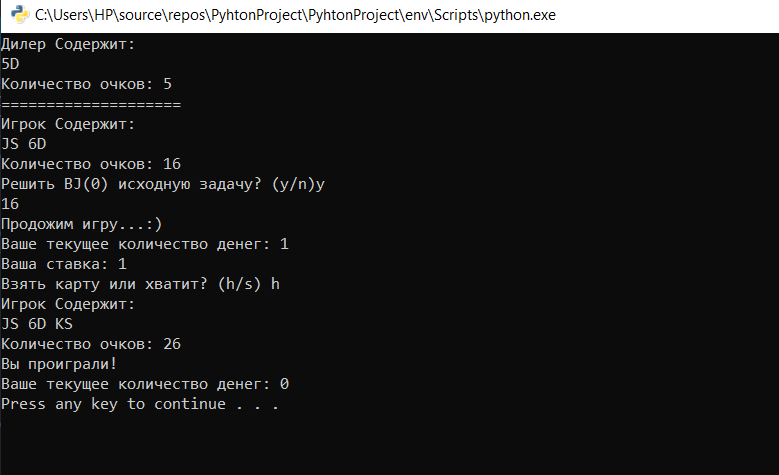
money = home\_money()

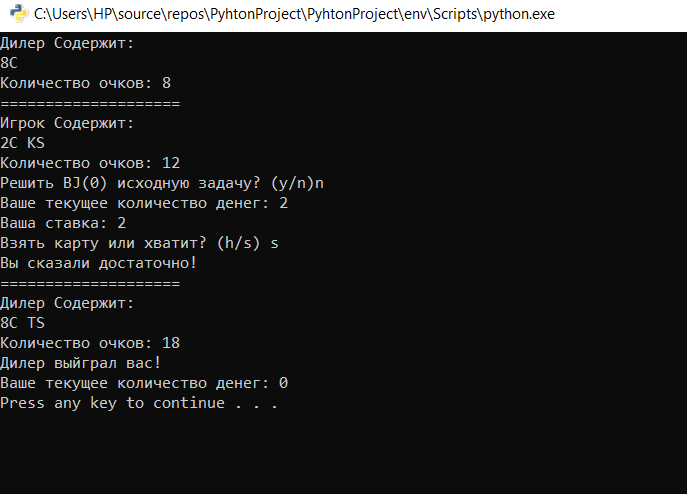
new\_game(money,bstavkab)

money = home\_money()

new\_game(money,bstavkab)

Тестирование первой части программы





**Алгоритм решения второй части задания:**

#!/usr/bin/python

# импортируем библиотеки

Импортируем pygame

из pygame.locals Импортируем \*

Импортируем random

Импортируем copy

# загрузка изображений

icon = pygame.image.load('resources/icon.png')

cback = pygame.image.load('resources/cards/cardback.png')

diamonda = pygame.image.load('resources/cards/ad.png')

cluba = pygame.image.load('resources/cards/ac.png')

hearta = pygame.image.load('resources/cards/ah.png')

spadea = pygame.image.load('resources/cards/as.png')

diamond2 = pygame.image.load('resources/cards/2d.png')

club2 = pygame.image.load('resources/cards/2c.png')

heart2 = pygame.image.load('resources/cards/2h.png')

spade2 = pygame.image.load('resources/cards/2s.png')

diamond3 = pygame.image.load('resources/cards/3d.png')

club3 = pygame.image.load('resources/cards/3c.png')

heart3 = pygame.image.load('resources/cards/3h.png')

spade3 = pygame.image.load('resources/cards/3s.png')

diamond4 = pygame.image.load('resources/cards/4d.png')

club4 = pygame.image.load('resources/cards/4c.png')

heart4 = pygame.image.load('resources/cards/4h.png')

spade4 = pygame.image.load('resources/cards/4s.png')

diamond5 = pygame.image.load('resources/cards/5d.png')

club5 = pygame.image.load('resources/cards/5c.png')

heart5 = pygame.image.load('resources/cards/5h.png')

spade5 = pygame.image.load('resources/cards/5s.png')

diamond6 = pygame.image.load('resources/cards/6d.png')

club6 = pygame.image.load('resources/cards/6c.png')

heart6 = pygame.image.load('resources/cards/6h.png')

spade6 = pygame.image.load('resources/cards/6s.png')

diamond7 = pygame.image.load('resources/cards/7d.png')

club7 = pygame.image.load('resources/cards/7c.png')

heart7 = pygame.image.load('resources/cards/7h.png')

spade7 = pygame.image.load('resources/cards/7s.png')

diamond8 = pygame.image.load('resources/cards/8d.png')

club8 = pygame.image.load('resources/cards/8c.png')

heart8 = pygame.image.load('resources/cards/8h.png')

spade8 = pygame.image.load('resources/cards/8s.png')

diamond9 = pygame.image.load('resources/cards/9d.png')

club9 = pygame.image.load('resources/cards/9c.png')

heart9 = pygame.image.load('resources/cards/9h.png')

spade9 = pygame.image.load('resources/cards/9s.png')

diamond10 = pygame.image.load('resources/cards/10d.png')

club10 = pygame.image.load('resources/cards/10c.png')

heart10 = pygame.image.load('resources/cards/10h.png')

spade10 = pygame.image.load('resources/cards/10s.png')

diamondj = pygame.image.load('resources/cards/jd.png')

clubj = pygame.image.load('resources/cards/jc.png')

heartj = pygame.image.load('resources/cards/jh.png')

spadej = pygame.image.load('resources/cards/js.png')

diamondq = pygame.image.load('resources/cards/qd.png')

clubq = pygame.image.load('resources/cards/qc.png')

heartq = pygame.image.load('resources/cards/qh.png')

spadeq = pygame.image.load('resources/cards/qs.png')

diamondk = pygame.image.load('resources/cards/kd.png')

clubk = pygame.image.load('resources/cards/kc.png')

heartk = pygame.image.load('resources/cards/kh.png')

spadek = pygame.image.load('resources/cards/ks.png')

#установка иконки на окно игры

pygame.display.set\_icon(icon)

#глобальные константы

black = (0,0,0)

white = (255,255,255)

gray = (192,192,192)

cards = [ diamonda, cluba, hearta, spadea, \

diamond2, club2, heart2, spade2, \

diamond3, club3, heart3, spade3, \

diamond4, club4, heart4, spade4, \

diamond5, club5, heart5, spade5, \

diamond6, club6, heart6, spade6, \

diamond7, club7, heart7, spade7, \

diamond8, club8, heart8, spade8, \

diamond9, club9, heart9, spade9, \

diamond10, club10, heart10, spade10, \

diamondj, clubj, heartj, spadej, \

diamondq, clubq, heartq, spadeq, \

diamondk, clubk, heartk, spadek ]

carda = [ diamonda, cluba, hearta, spadea ]

card2 = [ diamond2, club2, heart2, spade2 ]

card3 = [ diamond3, club3, heart3, spade3 ]

card4 = [ diamond4, club4, heart4, spade4 ]

card5 = [ diamond5, club5, heart5, spade5 ]

card6 = [ diamond6, club6, heart6, spade6 ]

card7 = [ diamond7, club7, heart7, spade7 ]

card8 = [ diamond8, club8, heart8, spade8 ]

card9 = [ diamond9, club9, heart9, spade9 ]

card10 = [ diamond10, club10, heart10, spade10, \

diamondj, clubj, heartj, spadej, \

diamondq, clubq, heartq, spadeq, \

diamondk, clubk, heartk, spadek ]

Функция getamt(card):

''' возвращает сумму, которую стоит карта.

например, ace по умолчанию 11. 10/валет/дама/король 10.'''

Если card in carda:

return 11

Всё\_если

Иначе если card in card2:

return 2

Всё\_иначе

Иначе если card in card3:

return 3

Всё\_иначе

Иначе если card in card4:

return 4

Всё\_иначе

Иначе если card in card5:

return 5

Всё\_иначе

Иначе если card in card6:

return 6

Всё\_иначе

Иначе если card in card7:

return 7

Всё\_иначе

Иначе если card in card8:

return 8

Всё\_иначе

Иначе если card in card9:

return 9

Всё\_иначе

Иначе если card in card10:

return 10

Всё\_иначе

Иначе :

печать ("генерация провалена...")

exit()

Функция gencard(clist, xlist):

'''создает карту из clist, удаляет ее из clist и добавляет ее в xlist.

возвращает, если карта-туз и просто 2 3 4 карта.'''

ca = 0

card = random.choice(clist)

clist.remove(card)

xlist.append(card)

Если card in carda:

ca = 1

Всё\_если

return card, ca

Функция initgame(clist, ulist, dlist):

'''генерирует две карты для дилера и пользователя, по одной за раз для каждого.

возвращает, если карта является тузом и общее количество карт на человека.'''

usera = 0

deala = 0

card1, ca = gencard(clist, ulist)

usera += ca

card2, ca = gencard(clist, dlist)

deala += ca

card3, ca = gencard(clist, ulist)

usera += ca

card4, ca = gencard(clist, dlist)

deala += ca

return getamt(card1) + getamt(card3), usera, getamt(card2) + getamt(card4), deala

Функция main():

ccards = copy.copy(cards)

stand = ложь

usercard = []

dealcard = []

winnum = 0

losenum = 0

moneynum = random.randint(1,5)

stavka = 1

#инициализация игры

pygame.init()

screen = pygame.display.set\_mode((640, 480))

pygame.display.set\_caption('игра blackjack')

font = pygame.font.sysfont('arial', 15)

hittxt = font.render('взять ещё', 0, black)

standtxt = font.render('достаточно', 0, black)

doubletxt = font.render('удвоить ставку',0,black)

restarttxt = font.render('перезапустить', 0, black)

gameovertxt = font.render('вы проиграли!', 0, white)

moneynotxt = font.render('у вас закончились деньги! пора домой за новой партией!',0,white)

usersum, usera, dealsum, deala = initgame(ccards, usercard, dealcard)

#fill background

background = pygame.surface(screen.get\_size())

background = background.convert()

background.fill((40, 150, 15))

hitb = pygame.draw.rect(background, gray, (28, 445, 75, 25))

standb = pygame.draw.rect(background, gray, (110, 445, 75, 25))

ratiob = pygame.draw.rect(background, gray, (535, 395, 95, 70))

doubleb = pygame.draw.rect(background,gray, (198, 445, 85, 25))

#бесконечный цикл

Пока true:

#проверяем если игрок проиграл,то

gameover = истина Если (usersum >= 21 and usera == 0) or len(usercard) == 5 Иначе ложь

Если len(usercard) == 2 and usersum == 21:

gameover = истина

Всё\_если

Иначе если len(dealcard) == 2 and dealsum == 21:

gameover = истина

Всё\_если

Иначе если moneynum <= 0:

gameover = истина

Всё\_если

#обновление фона под тексты денег игрока количества выйграшей и проиграшей.

scoredillertxt = font.render('текущее количество очков у диллера: %i' % dealsum,1,black)

scoreplayertxt = font.render('текущее количество очков у игрока: %i' % usersum,1,black)

stavkatxt = font.render('текущая ставка игрока: %i' % stavka,1,black)

moneytxt = font.render('денег у игрока: %i' % moneynum,1, black)

wintxt = font.render('побед: %i' % winnum, 1, black)

losetxt = font.render('поражений: %i' % losenum, 1, black)

#проверки нажатий по кнопкам взять ещё или достаточно

Для event in pygame.event.get():

Если event.type == quit:

pygame.quit()

Всё\_если

Иначе если event.type == pygame.mousebuttondown and not (gameover or stand) and hitb.collidepoint(pygame.mouse.get\_pos()):

#диллер дает игроку карту, если он не нарушает правила блэкджека.

card, ca = gencard(ccards, usercard)

usera += ca

usersum += getamt(card)

печать ("игрок: %i" % usersum)

Пока usersum > 21 and usera > 0:

usera -= 1

usersum -= 10

Всё\_иначе

Иначе если event.type == pygame.mousebuttondown and not gameover and doubleb.collidepoint(pygame.mouse.get\_pos()):

Если (stavka < moneynum):

stavka \*=2

Всё\_если

Иначе если event.type == pygame.mousebuttondown and not gameover and standb.collidepoint(pygame.mouse.get\_pos()):

#when player stands, the dealer plays

stand = истина

Пока dealsum <= usersum and dealsum < 17:

card, ca = gencard(ccards, dealcard)

deala += ca

dealsum += getamt(card)

печать ("диллер: %i" % dealsum)

Пока dealsum > 21 and deala > 0:

deala -= 1

dealsum -= 10

Всё\_пока

Всё\_пока

Иначе если event.type == pygame.mousebuttondown and (gameover or stand) and restartb.collidepoint(pygame.mouse.get\_pos()):

#restarts the game, updating scores

Если usersum == dealsum:

pass

Всё\_если

Иначе если usersum <= 21 and len(usercard) == 5:

winnum += 1

moneynum += stavka \* 1.5

Всё\_иначе

Иначе если usersum <= 21 and dealsum < usersum or dealsum > 21:

winnum += 1

Всё\_иначе

Иначе :

Если (moneynum > 0):

moneynum -=stavka

losenum += 1

stavka = 1

Всё\_если

Иначе :

moneynum = random.randint(1,5)

losenum = 0

winnum = 0

stavka = 1

Всё\_иначе

gameover = ложь

stand = ложь

usercard = []

dealcard = []

ccards = copy.copy(cards)

usersum, usera, dealsum, deala = initgame(ccards, usercard, dealcard)

restartb = pygame.draw.rect(background, (40, 150, 15), (260, 225, 105, 25))

screen.blit(background, (0, 0))

screen.blit(hittxt, (39, 448))

screen.blit(standtxt, (116, 448))

screen.blit(doubletxt,(198,448))

screen.blit(scoredillertxt,(5,158))

screen.blit(stavkatxt,(5,238))

screen.blit(scoreplayertxt,(5,258))

screen.blit(moneytxt, (535,398))

screen.blit(wintxt, (535, 423))

screen.blit(losetxt, (535, 448))

Всё\_для

#отображение карт диллера.

Для card in dealcard:

x = 10 + dealcard.index(card) \* 110

screen.blit(card, (x, 10))

screen.blit(cback, (120, 10))

Всё\_для

#отображение карт игрока.

Для card in usercard:

x = 10 + usercard.index(card) \* 110

screen.blit(card, (x, 295))

Всё\_для

#когда игра закончена, рисует кнопку перезапуска и текст, и показывает вторую карту дилера.

Если gameover or stand:

Если(moneynum > 0):

screen.blit(gameovertxt, (270, 200))

Всё\_если

Иначе:

screen.blit(moneynotxt, (150, 200))

Всё\_иначе

restartb = pygame.draw.rect(background, gray, (260, 225, 105, 25))

screen.blit(restarttxt, (270, 228))

screen.blit(dealcard[1], (120, 10))

pygame.display.update()

Если \_\_name\_\_ == '\_\_main\_\_':

main()

**Программа второй части задания**

#!/usr/bin/python

# Импортируем библиотеки

import pygame

from pygame.locals import \*

import random

import copy

# Загрузка изображений

icon =  pygame.image.load('resources/icon.png')

cBack = pygame.image.load('resources/cards/cardback.png')

diamondA = pygame.image.load('resources/cards/ad.png')

clubA =  pygame.image.load('resources/cards/ac.png')

heartA = pygame.image.load('resources/cards/ah.png')

spadeA = pygame.image.load('resources/cards/as.png')

diamond2 = pygame.image.load('resources/cards/2d.png')

club2 = pygame.image.load('resources/cards/2c.png')

heart2 = pygame.image.load('resources/cards/2h.png')

spade2 = pygame.image.load('resources/cards/2s.png')

diamond3 = pygame.image.load('resources/cards/3d.png')

club3 = pygame.image.load('resources/cards/3c.png')

heart3 = pygame.image.load('resources/cards/3h.png')

spade3 = pygame.image.load('resources/cards/3s.png')

diamond4 = pygame.image.load('resources/cards/4d.png')

club4 = pygame.image.load('resources/cards/4c.png')

heart4 = pygame.image.load('resources/cards/4h.png')

spade4 = pygame.image.load('resources/cards/4s.png')

diamond5 = pygame.image.load('resources/cards/5d.png')

club5 = pygame.image.load('resources/cards/5c.png')

heart5 = pygame.image.load('resources/cards/5h.png')

spade5 = pygame.image.load('resources/cards/5s.png')

diamond6 = pygame.image.load('resources/cards/6d.png')

club6 = pygame.image.load('resources/cards/6c.png')

heart6 = pygame.image.load('resources/cards/6h.png')

spade6 = pygame.image.load('resources/cards/6s.png')

diamond7 = pygame.image.load('resources/cards/7d.png')

club7 = pygame.image.load('resources/cards/7c.png')

heart7 = pygame.image.load('resources/cards/7h.png')

spade7 = pygame.image.load('resources/cards/7s.png')

diamond8 = pygame.image.load('resources/cards/8d.png')

club8 = pygame.image.load('resources/cards/8c.png')

heart8 = pygame.image.load('resources/cards/8h.png')

spade8 = pygame.image.load('resources/cards/8s.png')

diamond9 = pygame.image.load('resources/cards/9d.png')

club9 = pygame.image.load('resources/cards/9c.png')

heart9 = pygame.image.load('resources/cards/9h.png')

spade9 = pygame.image.load('resources/cards/9s.png')

diamond10 = pygame.image.load('resources/cards/10d.png')

club10 = pygame.image.load('resources/cards/10c.png')

heart10 = pygame.image.load('resources/cards/10h.png')

spade10 = pygame.image.load('resources/cards/10s.png')

diamondJ = pygame.image.load('resources/cards/jd.png')

clubJ = pygame.image.load('resources/cards/jc.png')

heartJ = pygame.image.load('resources/cards/jh.png')

spadeJ = pygame.image.load('resources/cards/js.png')

diamondQ = pygame.image.load('resources/cards/qd.png')

clubQ = pygame.image.load('resources/cards/qc.png')

heartQ = pygame.image.load('resources/cards/qh.png')

spadeQ = pygame.image.load('resources/cards/qs.png')

diamondK = pygame.image.load('resources/cards/kd.png')

clubK = pygame.image.load('resources/cards/kc.png')

heartK = pygame.image.load('resources/cards/kh.png')

spadeK = pygame.image.load('resources/cards/ks.png')

#Установка иконки на окно игры

pygame.display.set\_icon(icon)

#Глобальные константы

black = (0,0,0)

white = (255,255,255)

gray = (192,192,192)

cards = [ diamondA, clubA, heartA, spadeA, \

          diamond2, club2, heart2, spade2, \

          diamond3, club3, heart3, spade3, \

          diamond4, club4, heart4, spade4, \

          diamond5, club5, heart5, spade5, \

          diamond6, club6, heart6, spade6, \

          diamond7, club7, heart7, spade7, \

          diamond8, club8, heart8, spade8, \

          diamond9, club9, heart9, spade9, \

          diamond10, club10, heart10, spade10, \

          diamondJ, clubJ, heartJ, spadeJ, \

          diamondQ, clubQ, heartQ, spadeQ, \

          diamondK, clubK, heartK, spadeK ]

cardA = [ diamondA, clubA, heartA, spadeA ]

card2 = [ diamond2, club2, heart2, spade2 ]

card3 = [ diamond3, club3, heart3, spade3 ]

card4 = [ diamond4, club4, heart4, spade4 ]

card5 = [ diamond5, club5, heart5, spade5 ]

card6 = [ diamond6, club6, heart6, spade6 ]

card7 = [ diamond7, club7, heart7, spade7 ]

card8 = [ diamond8, club8, heart8, spade8 ]

card9 = [ diamond9, club9, heart9, spade9 ]

card10 = [ diamond10, club10, heart10, spade10, \

            diamondJ, clubJ, heartJ, spadeJ, \

            diamondQ, clubQ, heartQ, spadeQ, \

            diamondK, clubK, heartK, spadeK ]

def getAmt(card):

    ''' Возвращает сумму, которую стоит карта.

Например, Ace по умолчанию 11. 10/Валет/Дама/Король 10.'''

    if card in cardA:

        return 11

    elif card in card2:

        return 2

    elif card in card3:

        return 3

    elif card in card4:

        return 4

    elif card in card5:

        return 5

    elif card in card6:

        return 6

    elif card in card7:

        return 7

    elif card in card8:

        return 8

    elif card in card9:

        return 9

    elif card in card10:

        return 10

    else:

        print ("Генерация провалена...")

        exit()

def genCard(cList, xList):

    '''Создает карту из cList, удаляет ее из cList и добавляет ее в xList.

Возвращает, если карта-Туз и просто 2 3 4 карта.'''

    cA = 0

    card = random.choice(cList)

    cList.remove(card)

    xList.append(card)

    if card in cardA:

        cA = 1

    return card, cA

def initGame(cList, uList, dList):

    '''Генерирует две карты для дилера и пользователя, по одной за раз для каждого.

Возвращает, если карта является тузом и общее количество карт на человека.'''

    userA = 0

    dealA = 0

    card1, cA = genCard(cList, uList)

    userA += cA

    card2, cA = genCard(cList, dList)

    dealA += cA

    card3, cA = genCard(cList, uList)

    userA += cA

    card4, cA = genCard(cList, dList)

    dealA += cA

    return getAmt(card1) + getAmt(card3), userA, getAmt(card2) + getAmt(card4), dealA

def main():

    ccards = copy.copy(cards)

    stand = False

    userCard = []

    dealCard = []

    winNum = 0

    loseNum = 0

    moneyNum = random.randint(1,5)

    stavka = 1

    #Инициализация игры

    pygame.init()

    screen = pygame.display.set\_mode((640, 480))

    pygame.display.set\_caption('Игра BlackJack')

    font = pygame.font.SysFont('arial', 15)

    hitTxt = font.render('Взять ещё', 0, black)

    standTxt = font.render('Достаточно', 0, black)

    doubleTxt = font.render('Удвоить ставку',0,black)

    restartTxt = font.render('Перезапустить', 0, black)

    gameoverTxt = font.render('Вы проиграли!', 0, white)

    moneyNoTxt = font.render('У вас закончились деньги! Пора домой за новой партией!',0,white)

    userSum, userA, dealSum, dealA = initGame(ccards, userCard, dealCard)

    #Fill Background

    background = pygame.Surface(screen.get\_size())

    background = background.convert()

    background.fill((40, 150, 15))

    hitB = pygame.draw.rect(background, gray, (28, 445, 75, 25))

    standB = pygame.draw.rect(background, gray, (110, 445, 75, 25))

    ratioB = pygame.draw.rect(background, gray, (535, 395, 95, 70))

    doubleB = pygame.draw.rect(background,gray, (198, 445, 85, 25))

    #Бесконечный цикл

    while True:

        #Проверяем если игрок проиграл,то

        gameover = True if (userSum >= 21 and userA == 0) or len(userCard) == 5 else False

        if len(userCard) == 2 and userSum == 21:

            gameover = True

        elif len(dealCard) == 2 and dealSum == 21:

            gameover = True

        elif moneyNum <= 0:

            gameover = True

        #Обновление фона под тексты денег игрока количества выйграшей и проиграшей.

        scoreDillerTxt = font.render('Текущее количество очков у Диллера: %i' % dealSum,1,black)

        scorePlayerTxt = font.render('Текущее количество очков у Игрока: %i' % userSum,1,black)

        stavkaTxt = font.render('Текущая ставка игрока: %i' % stavka,1,black)

        moneyTxt = font.render('Денег у игрока: %i' % moneyNum,1, black)

        winTxt = font.render('Побед: %i' % winNum, 1, black)

        loseTxt = font.render('Поражений: %i' % loseNum, 1, black)

        #Проверки нажатий по кнопкам Взять ещё или достаточно

        for event in pygame.event.get():

            if event.type == quit:

                pygame.quit()

            elif event.type == pygame.MOUSEBUTTONDOWN and not (gameover or stand) and hitB.collidepoint(pygame.mouse.get\_pos()):

                #Диллер дает игроку карту, если он не нарушает правила блэкджека.

                card, cA = genCard(ccards, userCard)

                userA += cA

                userSum += getAmt(card)

                print ("Игрок: %i" % userSum)

                while userSum > 21 and userA > 0:

                    userA -= 1

                    userSum -= 10

            elif event.type == pygame.MOUSEBUTTONDOWN and not gameover and doubleB.collidepoint(pygame.mouse.get\_pos()):

                if(stavka < moneyNum):

                    stavka \*=2

            elif event.type == pygame.MOUSEBUTTONDOWN and not gameover and standB.collidepoint(pygame.mouse.get\_pos()):

                #when player stands, the dealer plays

                stand = True

                while dealSum <= userSum and dealSum < 17:

                    card, cA = genCard(ccards, dealCard)

                    dealA += cA

                    dealSum += getAmt(card)

                    print ("Диллер: %i" % dealSum)

                    while dealSum > 21 and dealA > 0:

                        dealA -= 1

                        dealSum -= 10

            elif event.type == pygame.MOUSEBUTTONDOWN and (gameover or stand) and restartB.collidepoint(pygame.mouse.get\_pos()):

                #restarts the game, updating scores

                if userSum == dealSum:

                    pass

                elif userSum <= 21 and len(userCard) == 5:

                    winNum += 1

                    moneyNum += stavka \* 1.5

                elif userSum <= 21 and dealSum < userSum or dealSum > 21:

                    winNum += 1

                else:

                    if(moneyNum > 0):

                        moneyNum -=stavka

                        loseNum += 1

                        stavka = 1

                    else:

                        moneyNum = random.randint(1,5)

                        loseNum = 0

                        winNum = 0

                        stavka = 1

                gameover = False

                stand = False

                userCard = []

                dealCard = []

                ccards = copy.copy(cards)

                userSum, userA, dealSum, dealA = initGame(ccards, userCard, dealCard)

                restartB = pygame.draw.rect(background, (40, 150, 15), (260, 225, 105, 25))

        screen.blit(background, (0, 0))

        screen.blit(hitTxt, (39, 448))

        screen.blit(standTxt, (116, 448))

        screen.blit(doubleTxt,(198,448))

        screen.blit(scoreDillerTxt,(5,158))

        screen.blit(stavkaTxt,(5,238))

        screen.blit(scorePlayerTxt,(5,258))

        screen.blit(moneyTxt, (535,398))

        screen.blit(winTxt, (535, 423))

        screen.blit(loseTxt, (535, 448))

        #Отображение карт Диллера.

        for card in dealCard:

            x = 10 + dealCard.index(card) \* 110

            screen.blit(card, (x, 10))

        screen.blit(cBack, (120, 10))

        #Отображение карт Игрока.

        for card in userCard:

            x = 10 + userCard.index(card) \* 110

            screen.blit(card, (x, 295))

        #когда игра закончена, рисует кнопку перезапуска и текст, и показывает вторую карту дилера.

        if gameover or stand:

            if(moneyNum > 0):

                screen.blit(gameoverTxt, (270, 200))

            else:

                screen.blit(moneyNoTxt, (150, 200))

            restartB = pygame.draw.rect(background, gray, (260, 225, 105, 25))

            screen.blit(restartTxt, (270, 228))

            screen.blit(dealCard[1], (120, 10))

        pygame.display.update()

if \_\_name\_\_ == '\_\_main\_\_':

    main()

**Тестирование второй части программы**

