**РОССИЙСКИЙ УНИВЕРСИТЕТ ДРУЖБЫ НАРОДОВ**

**Факультет физико-математических и естественных наук**

**Кафедра прикладной информатики и теории вероятностей**

**ОТЧЕТ**

**по домашней работе № 1**

*дисциплина: Теория автоматов и формальных языков*

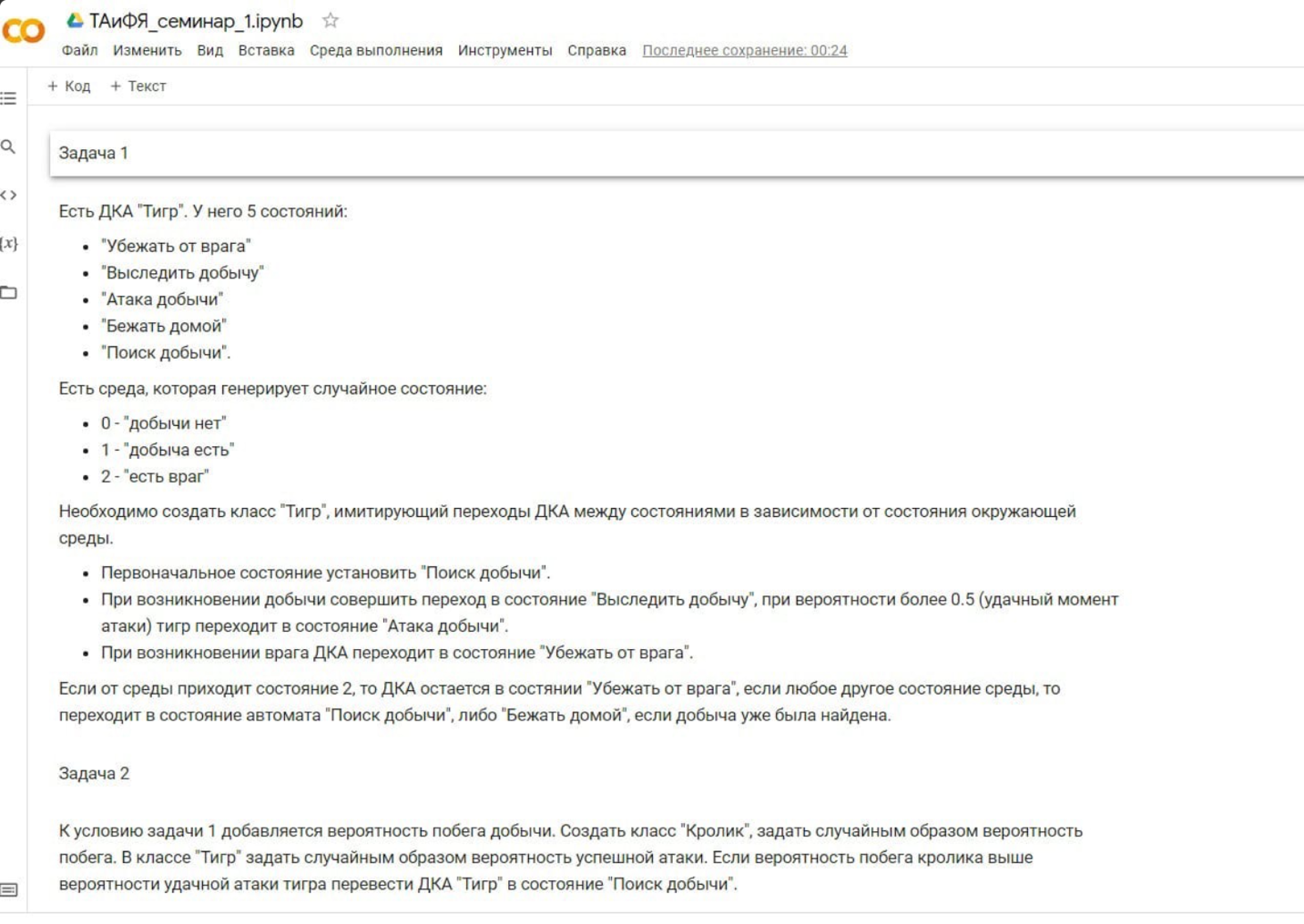
Студент: Шутенко Виктория Михайловна

Группа: НФИ-бд-03-19

**МОСКВА**

2021 г.

Задание



Решение

Для решения первой задачи я создала класс Tiger, включающий в себя две функции: \_\_init\_\_, conditions. Первая функция инициализирует атрибуты state и tiger. Где

state – среда, а tiger – Тигр. Вторая функция определяет работу всей программы. Здесь указывается первоначальное значение переменной self.tiger = “Поиск добычы”, в соответствии с заданием. Так же здесь есть два списка self.state и self.opportunity\_tiger. В первом списке указываются в состояния среды, в соответствии с заданием. Второй список указывает возможную вероятность выследить добычу. Поскольку вероятность не может быть больше единицы, то она находится на интервале от нуля до единицы. Также в соответствии с заданием я использовала функции рандом для выбора значений из этих списков. Далее с помощью if-elif-else Я осуществляла перевод в разные состояния.

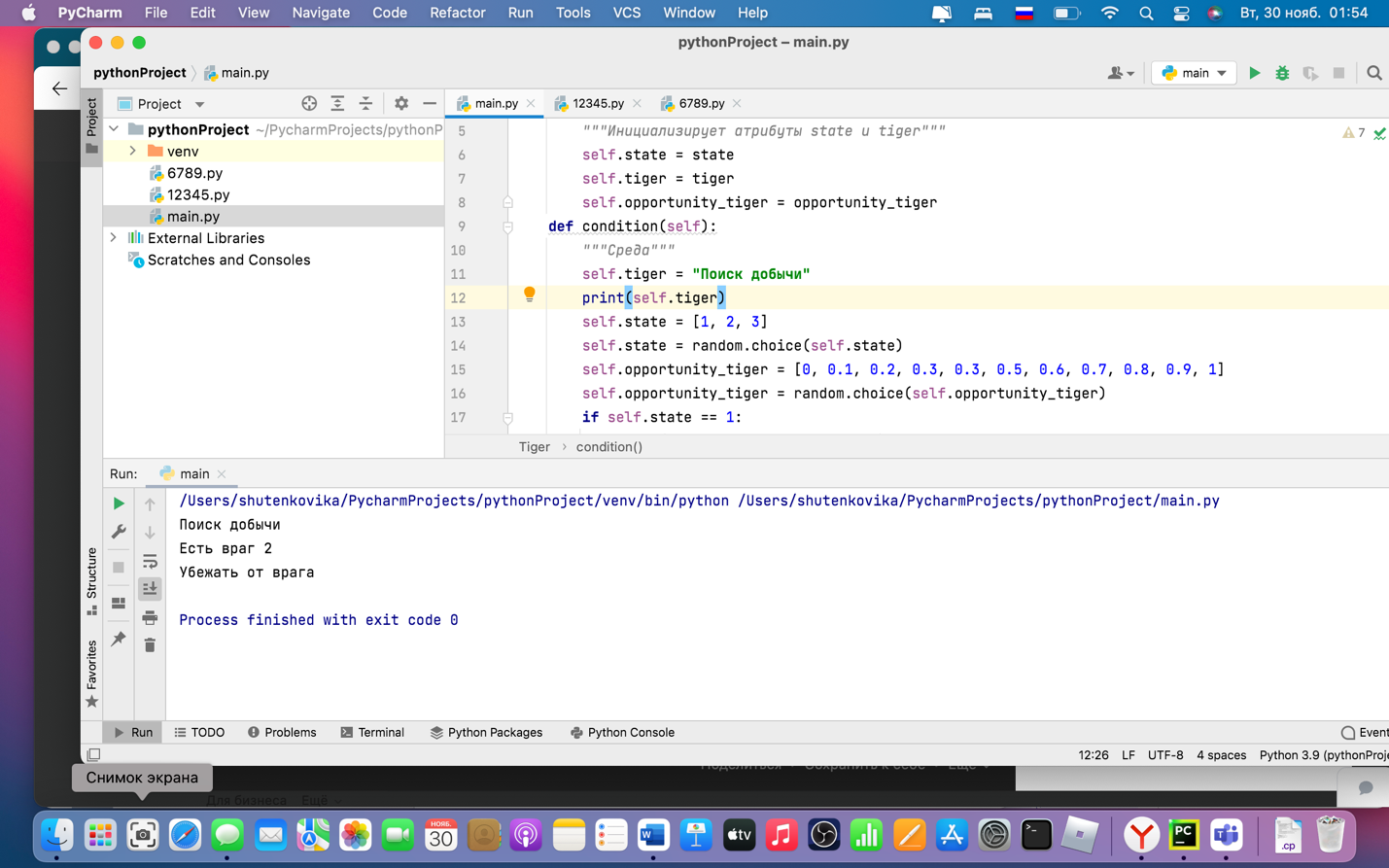
По условию задачи при 1 добыча есть, при 2 есть враг, а при нуле добычи нет. Важным следует заметить, что при 1, у нас появляется действие «выследить добычу». При условии, что вероятность данного действия больше, чем 0,5. В каждом условии прописывается вывод состояния и действие тигра соответственно. В конце программы вызывается класс.

Программный код

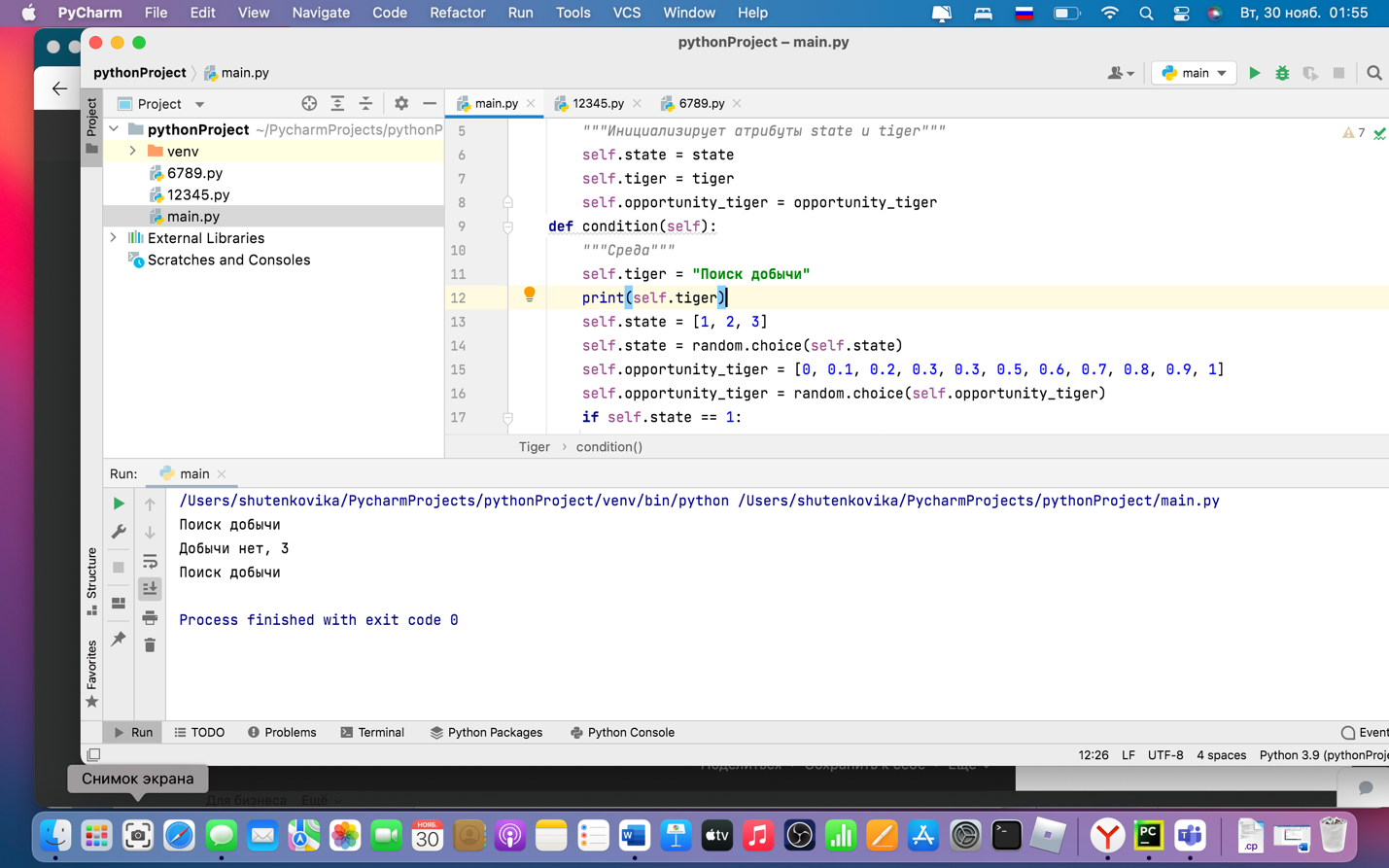
**import** random  
**class** Tiger(object):  
 *"""Описание среды"""* **def** \_\_init\_\_(self, state, tiger, opportunity\_tiger):  
 *"""Инициализирует атрибуты state и tiger"""* self.state = state  
 self.tiger = tiger  
 self.opportunity\_tiger = opportunity\_tiger  
 **def** condition(self):  
 *"""Среда"""* self.tiger = **"Поиск добычи"** print(self.tiger)  
 self.state = [1, 2, 3]  
 self.state = random.choice(self.state)  
 self.opportunity\_tiger = [0, 0.1, 0.2, 0.3, 0.3, 0.5, 0.6, 0.7, 0.8, 0.9, 1]  
 self.opportunity\_tiger = random.choice(self.opportunity\_tiger)  
 **if** self.state == 1:  
 print(**'Добыча есть,'**,self.state)  
 self.tiger = **"Выследить добычу"** print(self.tiger)  
 **if** self.opportunity\_tiger > 0.5:  
 print(**"Вероятность = "**,self.opportunity\_tiger)  
 self.tiger = **"Атака добычи"  
 else**:  
 print(**"Вероятность = "**, self.opportunity\_tiger)  
 self.tiger = **"Поиск добычи"  
 elif** self.state == 2:  
 print(**'Есть враг,'**, self.state)  
 self.tiger = **"Убежать от врага"  
 else**:  
 print(**'Добычи нет,'**, self.state)  
 self.tiger = **"Поиск добычи"  
 return** self.tiger  
my\_tiger = Tiger(**'savana'**, **'behavior'**, **'op'**)  
print(my\_tiger.condition())

Скриншоты выводов 1 задачи

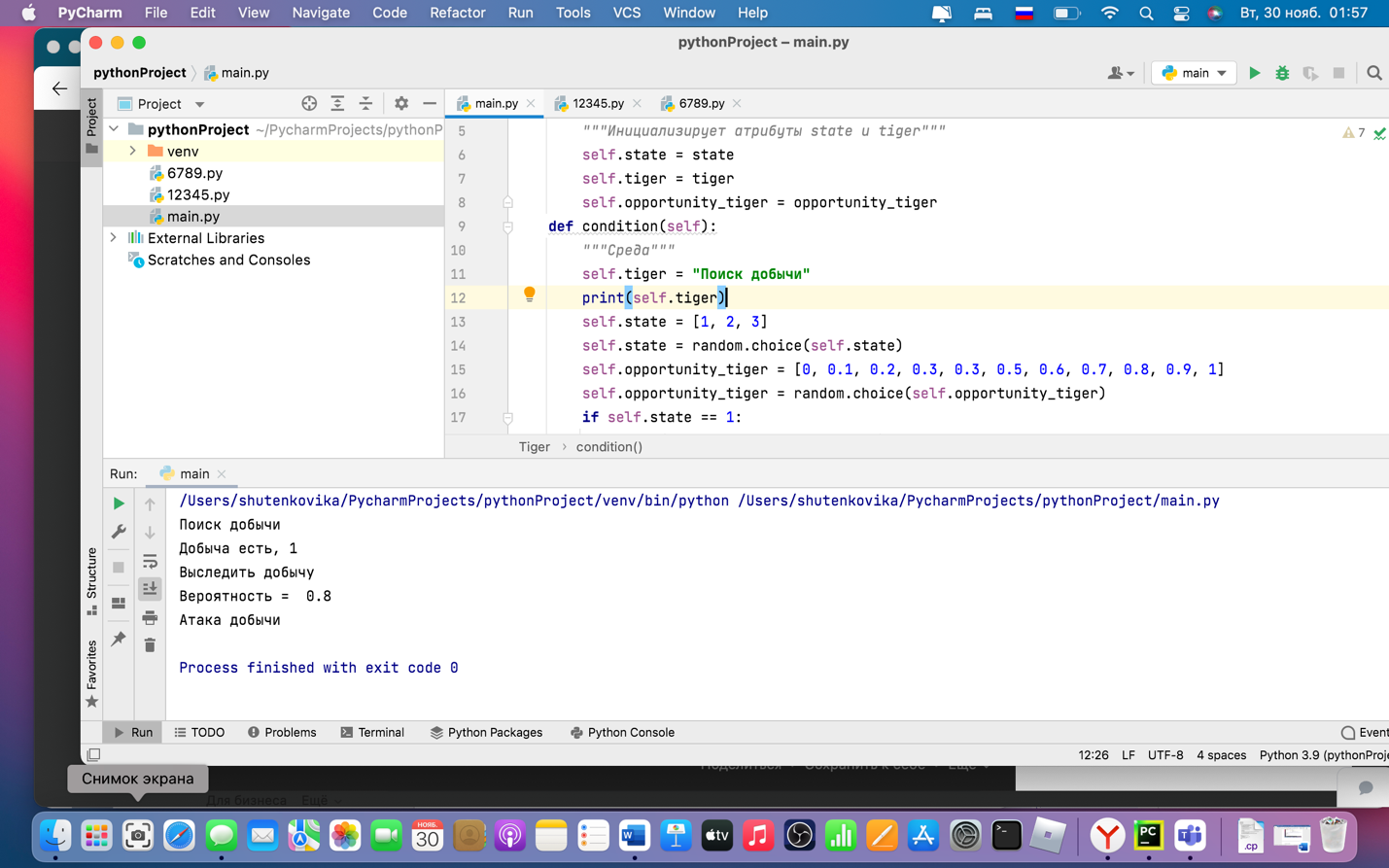
При условии, что есть враг:



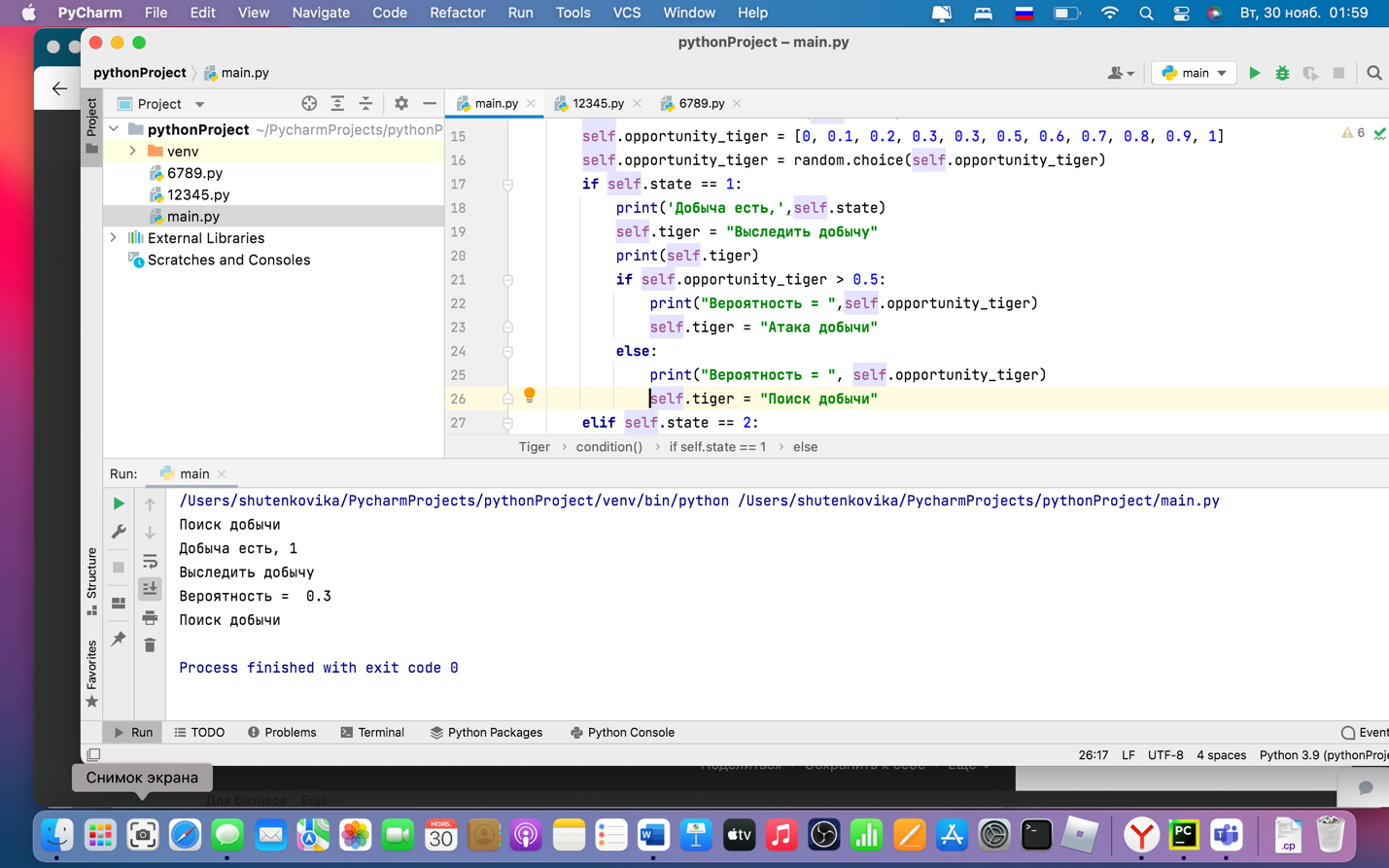
При условии, что нет добычи:



При условии, что есть добыча (вероятность больше, чем 0,5):



При условии, что есть добыча (вероятность меньше, чем 0,5):



Решение задачи 2:

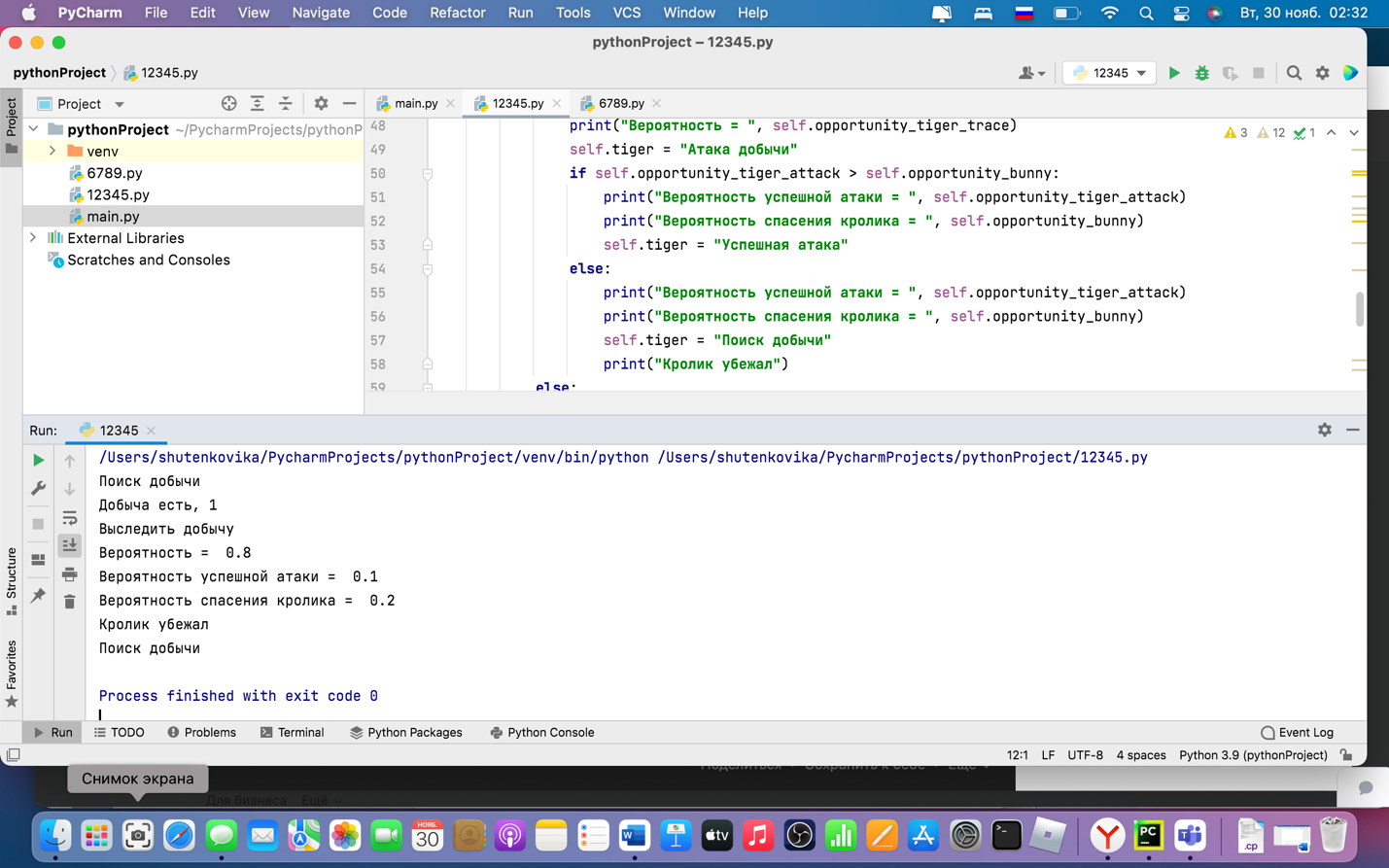
Во второй задаче нужно было добавить ещё один класс кролик. У меня этот класс главный, но в нём только две функции. Первая функция \_\_init\_\_ –инициализирует атрибуты state, bunny и opportunity\_bunny. Вторая функция condition\_bunny считает вероятность побега кролика. Класс Tiger стал дочерним, хотя в нём происходит основное описание кода. Поскольку удобнее наследовать функцию из главного класса в дочерней с использованием функции super(). Далее, в классе Tiger добавила ещё одно условие if-else – вероятность успешный атаки и вероятность спасения кролика. Обе эти вероятности задаются рандомно, так, как делалось в 1 задаче с использованием списков вероятностей и функции random. Потом в этом условии они сравниваются и от результата сравнения получается переход в состояние.

Программный код:

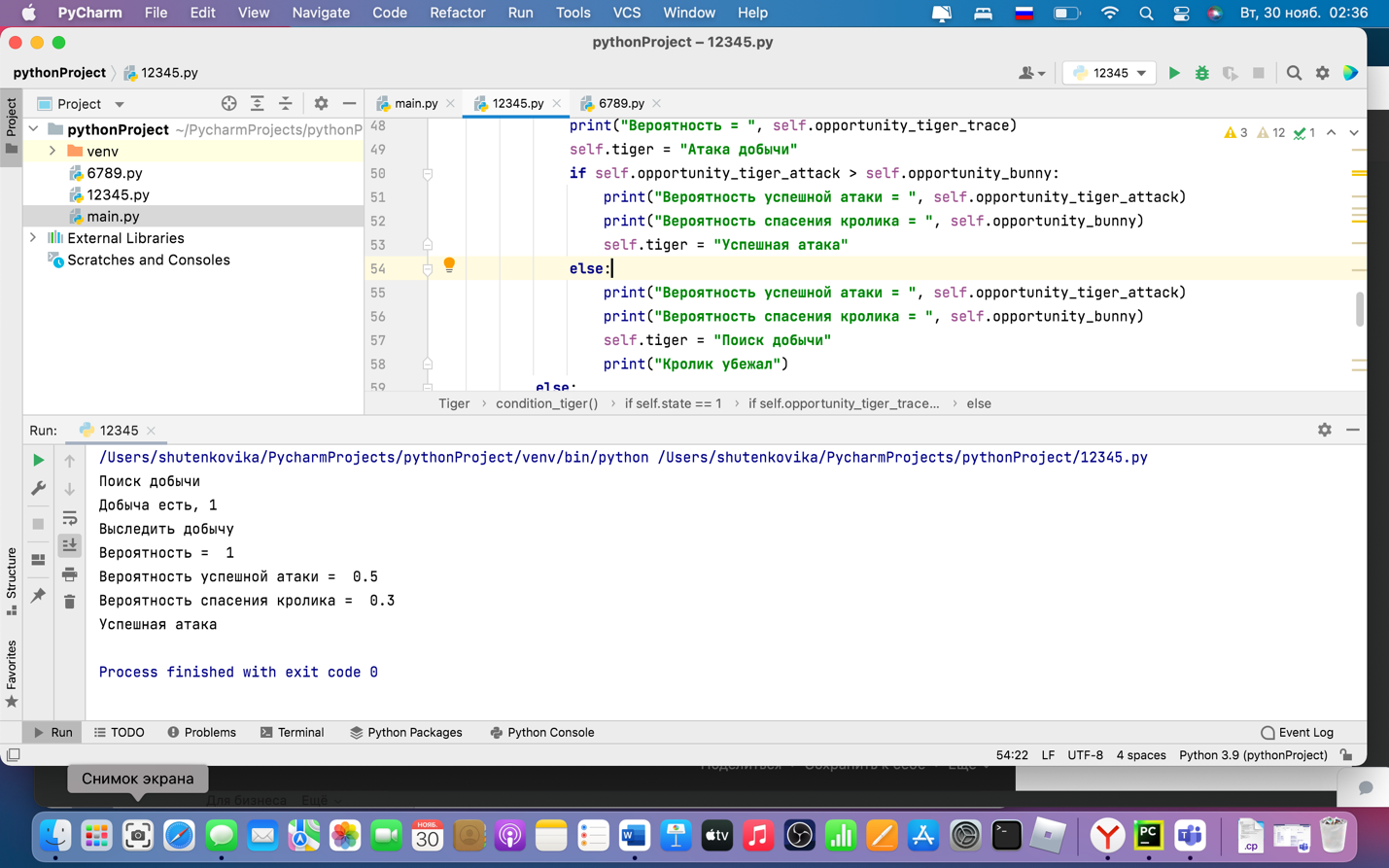
**import** random  
**class** Bunny(object):  
 *"""Описание кролика"""* **def** \_\_init\_\_(self, state, bunny, opportunity\_bunny):  
 *"""Инициализирует атрибуты state, bunny и opportunity\_bunny"""* self.state = state  
 self.bunny = bunny  
 self.opportunity\_bunny = opportunity\_bunny  
 **def** condition\_bunny(self):  
 *"""Среда"""* self.opportunity\_bunny = [0, 0.1, 0.2, 0.3, 0.3, 0.5, 0.6, 0.7, 0.8, 0.9, 1]  
 self.opportunity\_bunny = random.choice(self.opportunity\_bunny)  
 **return** self.opportunity\_bunny  
  
**class** Tiger(Bunny):  
 *"""Описание среды"""* **def** \_\_init\_\_(self, state, tiger, opportunity\_tiger\_trace, opportunity\_tiger\_attack):  
 *"""Инициализирует атрибуты state и tiger"""* self.state = state  
 self.tiger = tiger  
 self.opportunity\_tiger\_trace = opportunity\_tiger\_trace  
 self.opportunity\_tiger\_attack = opportunity\_tiger\_attack  
 super(Tiger, self).condition\_bunny() *# передача из род. класса в дочерний* **def** condition\_tiger(self):  
 *"""Среда"""* self.tiger = **"Поиск добычи"** print(self.tiger)  
 self.state = [1, 2, 3]  
 self.state = random.choice(self.state)  
 self.opportunity\_tiger\_trace = [0, 0.1, 0.2, 0.3, 0.3, 0.5, 0.6, 0.7, 0.8, 0.9, 1]  
 self.opportunity\_tiger\_trace = random.choice(self.opportunity\_tiger\_trace)  
 self.opportunity\_tiger\_attack = [0, 0.1, 0.2, 0.3, 0.3, 0.5, 0.6, 0.7, 0.8, 0.9, 1]  
 self.opportunity\_tiger\_attack= random.choice(self.opportunity\_tiger\_attack)  
  
 **if** self.state == 1:  
 print(**'Добыча есть,'**, self.state)  
 self.tiger = **"Выследить добычу"** print(self.tiger)  
 **if** self.opportunity\_tiger\_trace > 0.5:  
 print(**"Вероятность = "**, self.opportunity\_tiger\_trace)  
 self.tiger = **"Атака добычи"  
 if** self.opportunity\_tiger\_attack > self.opportunity\_bunny:  
 print(**"Вероятность успешной атаки = "**, self.opportunity\_tiger\_attack)  
 print(**"Вероятность спасения кролика = "**, self.opportunity\_bunny)  
 self.tiger = **"Успешная атака"  
 else**:  
 print(**"Вероятность успешной атаки = "**, self.opportunity\_tiger\_attack)  
 print(**"Вероятность спасения кролика = "**, self.opportunity\_bunny)  
 self.tiger = **"Поиск добычи"** print(**"Кролик убежал"**)  
 **else**:  
 print(**"Вероятность = "**, self.opportunity\_tiger\_trace)  
 self.tiger = **"Поиск добычи"  
 elif** self.state == 2:  
 print(**'Есть враг,'**, self.state)  
 self.tiger = **"Убежать от врага"  
 else**:  
 print(**'Добычи нет,'**, self.state)  
 self.tiger = **"Поиск добычи"  
 return** self.tiger  
  
my\_tiger = Tiger(**'savana'**, **'behavior'**, **'op1'**, **'op2'**)  
print(my\_tiger.condition\_tiger())

Скриншоты выводов 2 задачи (новые условия)

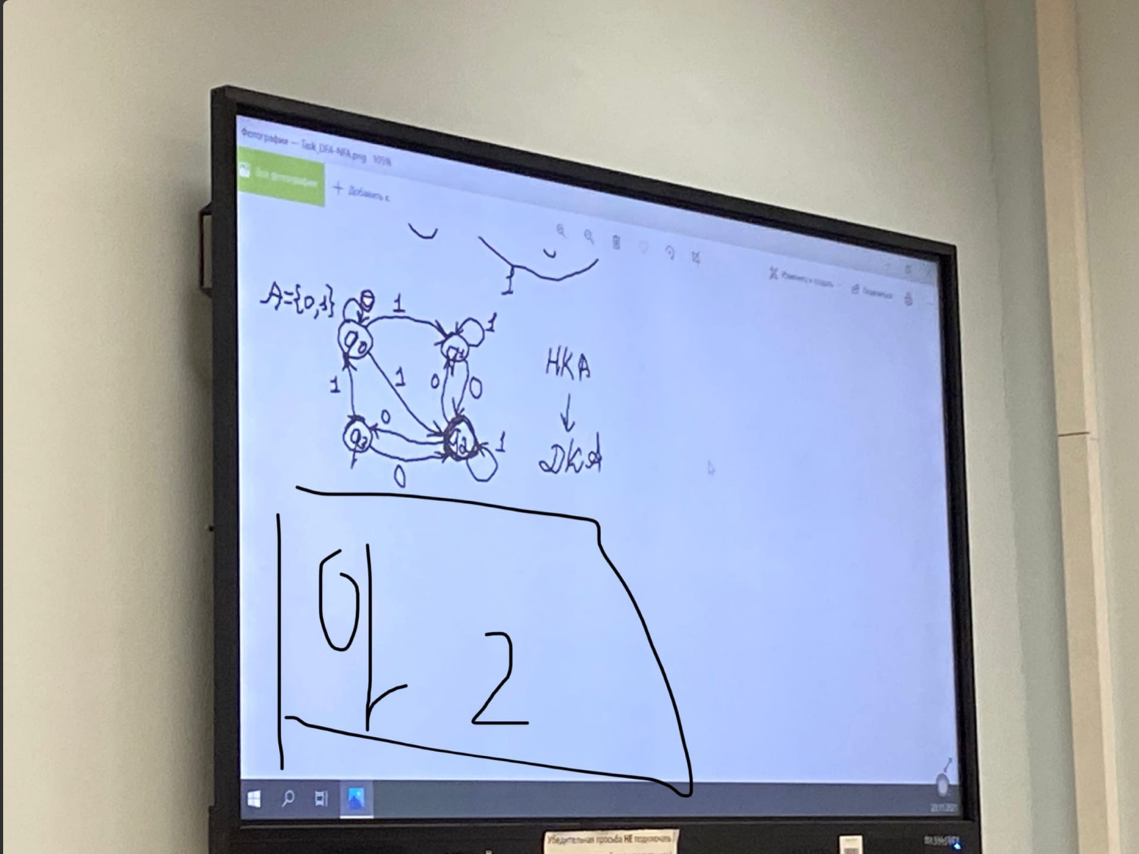
Вероятность спасения кролика больше чем, вероятность успешной атаки:



Вероятность успешной атаки больше чем, вероятность спасения кролика:



Задание №3



Решение

{Q}: = Q0, Q1 Q2, Q1 Q2 Q3, Q0 Q1 Q2, Q0 Q1 Q2 Q3,

1

0

0

1

1

0

0

1

1

0

|  |  |
| --- | --- |
| Qi | Вершины |
| Q0 | 0: Q0  1: Q1 Q2 |
| Q1 Q2 | 0: Q1 Q2 Q3  1: Q1 Q2 |
| Q1 Q2 Q3 | 0: Q1 Q2 Q3  1: Q0 Q1 Q2 |
| Q0 Q1 Q2 | 0: Q0 Q1 Q2 Q3  1: Q1 Q2 |
| Q0 Q1 Q2 Q3 | 0: Q0 Q1 Q2 Q3  1: Q0 Q1 Q2 |