Оглавление

[1. Текстовое описание задачи 1](#_Toc76191039)

[2.Диаграмма прецедентов 2](#_Toc76191040)

[3.Диаграмма классов 2](#_Toc76191041)

[4.Определения диаграмм по классам 4](#_Toc76191042)

[6.Отладка одного из модулей 6](#_Toc76191043)

[7.Тестовые наборы и тестирование одного из модулей 8](#_Toc76191044)

[Тестовые наборы и тестирование одного из модулей. 8](#_Toc76191045)

[8. Значение временной сложности O для одного из методов. Значение O для оптимального кода 9](#_Toc76191046)

Задание. Подготовить доклад и презентацию по заданию производственной практики «Разработка приложения «Симулятор кубика»»

# 1. Текстовое описание задачи

**Цель написания программы:**

Данная программа будет использоваться игроками DnD, во время остутствия подходящих кубиков, или в виду полного неимения таковых. Или совсем замещая физические кубики

**Полное описание задачи:**

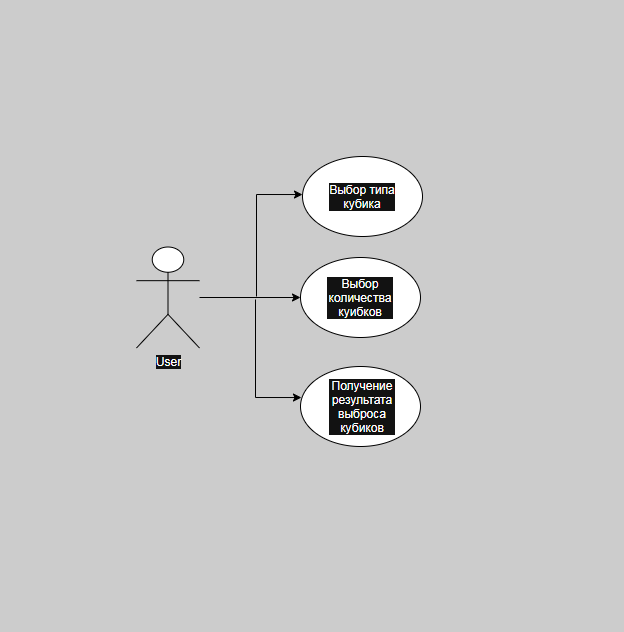
Разработать для приложения модули:

1. Панель для выбора типа кубика. Должно содержать в себе все кубики используемые для игры DnD и создавать кубик с выбранным количеством граней. Существует 5 вариаций кубиков (с четырьмя, шестью, восьмью, двенадцатью, двадцатью гранями ) После чего выкидывать случайным образом значение\я кубика\ов
2. Панель для выбора количества кубиков. т.к. По правилам игры требуются один или два кубика, в зависимости, от ситуации предусмотренной в самой игре
3. Кнопка вывода результата на экран и поля, где этот результат будет выводиться.

**Краткое описание задачи:**

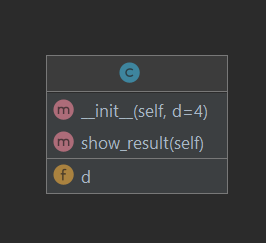
Создать приложение, которое по запросу пользователя будет выдавать результат\ты кубиков в зависимости от выбранного типа и количества

# 2.Диаграмма прецедентов

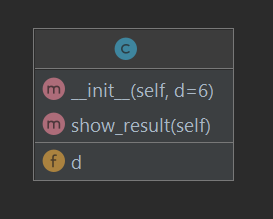


# 3.Диаграмма классов

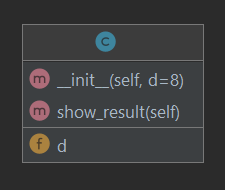
Куб д4



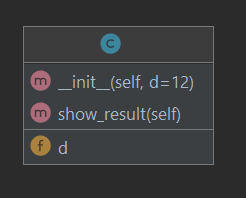
Куб д6



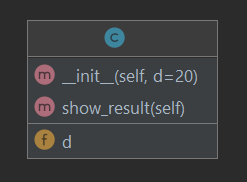
Куб д8



Куб д12



Куб д20



# 4.Определения диаграмм по классам

Листинг 1 создаём класс кубика Dice\_4d и придаём ему значение в его диапазоне

#Создаём класс для 4х гранного кубика и рандомом записываем число в результат

class Dice\_4d:

"4d dice"

def \_\_init\_\_(self, d=4):

self.d = d

def show\_result(self):

global result

result = random.randint(1, self.d)

Листинг 2 создаём класс кубика Dice\_6d и придаём ему значение в его диапазоне

#Создаём класс для 6х гранного кубика и рандомом записываем число в результат

class Dice\_6d:

"6d dice"

def \_\_init\_\_(self, d=6):

self.d = d

def show\_result(self):

global result

result = random.randint(1, self.d)

Листинг 3 создаём класс кубика Dice\_8d и придаём ему значение в его диапазоне

#Создаём класс для 8х гранного кубика и рандомом записываем число в результат

class Dice\_8d:

"8d dice"

def \_\_init\_\_(self, d=8):

self.d = d

def show\_result(self):

global result

result = random.randint(1, self.d)

Листинг 4 создаём класс кубика Dice\_12d и придаём ему значение в его диапазоне

#Создаём класс для 12х гранного кубика и рандомом записываем число в результат

class Dice\_12d:

"12d dice"

def \_\_init\_\_(self, d=12):

self.d = d

def show\_result(self):

global result

result = random.randint(1, self.d)

Листинг 5 создаём класс кубика Dice\_20d и придаём ему значение в его диапазоне

#Создаём класс для 20х гранного кубика и рандомом записываем число в результат

class Dice\_20d:

"20d dice"

def \_\_init\_\_(self, d=20):

self.d = d

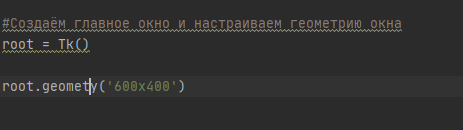
def show\_result(self):

global result

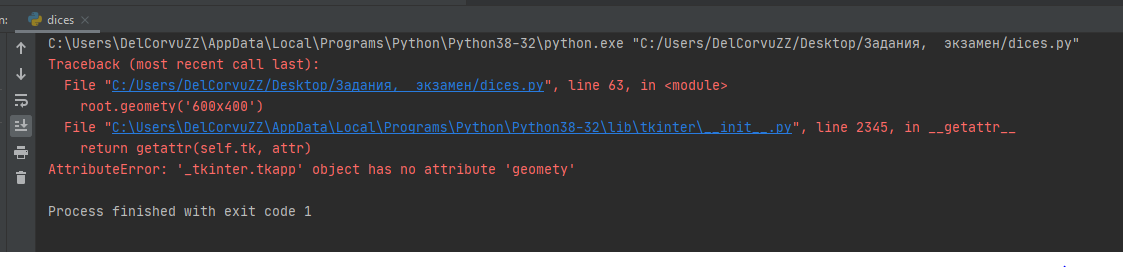
result = random.randint(1, self.d)

# 6.Отладка одного из модулей

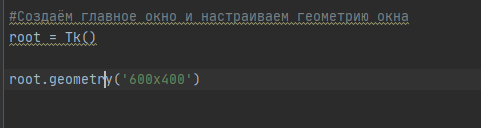
Неправильно прописываем геометрию окна в tkinter



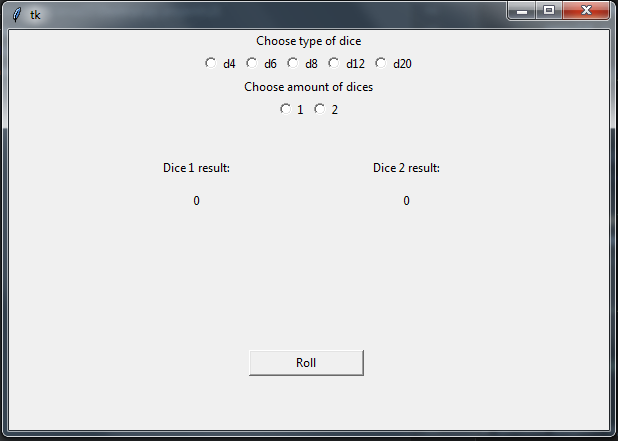
Получаем ошибку



Исправляем



Программа работает правильно

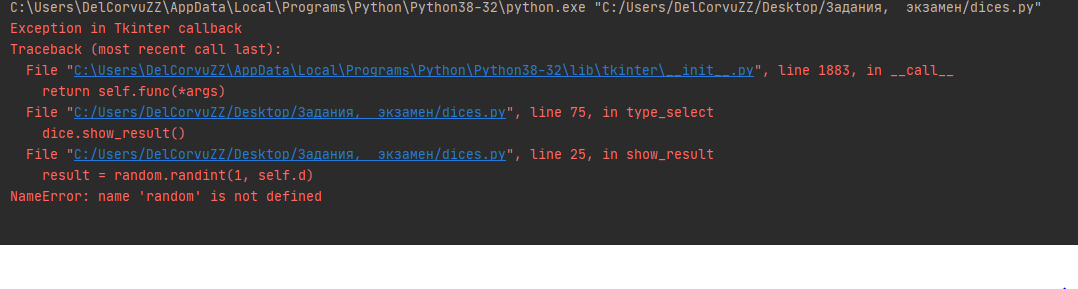


# 7.Тестовые наборы и тестирование одного из модулей

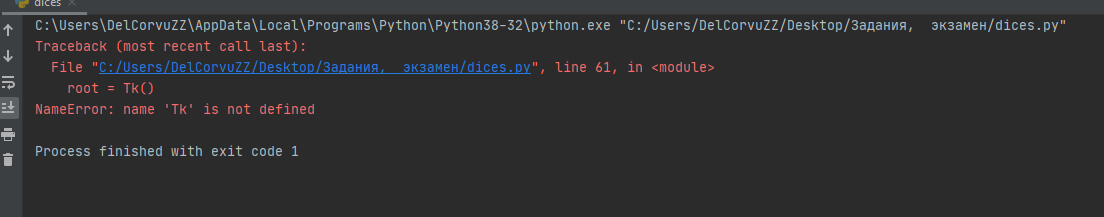
# Тестовые наборы и тестирование одного из модулей.

|  |  |
| --- | --- |
| Тест 1 Отсутствие import random | Ошибка |
| Тест 2 Отсутствие import tkinter | Ошибка |

random



tkinter



# 8. Значение временной сложности O для одного из методов. Значение O для оптимального кода

O(n) , так как алгоритм не должен пройти все строки, и данные заранее будут выбраны