**Техническое задание:**

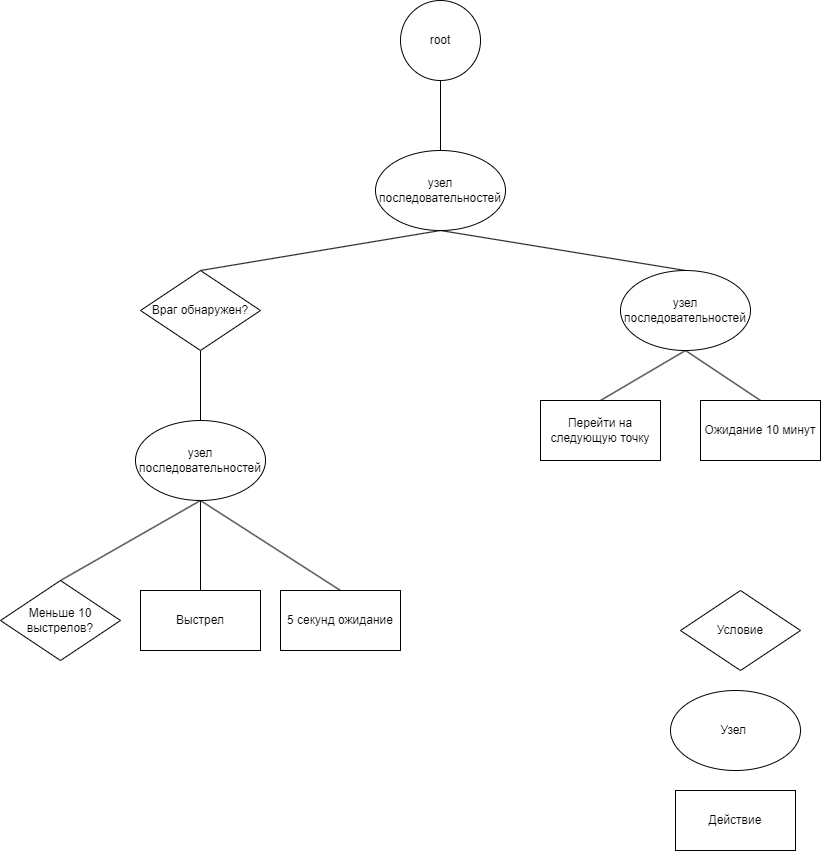
1. Реализуйте функции сериализации и десериализации двусвязного списка, заданного следующим образом:

def serialize(self, file):  
 *"""  
 Система в файле такова:  
 индекс элемента, значение элемента, (индекс элемента по случайной ссылке/или '')* ***:param*** *file:* ***:return****:  
 """* tmp1 = self.\_head  
 tmp2 = self.\_tail  
 f = open(str(file)+".txt", 'w')  
 if not tmp1.data:  
 f.write('')  
 elif tmp1 == tmp2:  
 if tmp1.rand:  
 f.write('0,' + str(tmp1.data) + ',0,')  
 else:  
 f.write('0,' + str(tmp1.data)+',')  
 else:  
 tmp\_count = 0  
 rand = {}  
 while tmp\_count < self.\_count / 2:# Заходим с двух концов, чтобы ускорить метод  
 rand[tmp1] = tmp\_count  
 rand[tmp2] = self.\_count-1-tmp\_count  
 tmp2 = tmp2.prev  
 tmp1 = tmp1.next  
 tmp\_count += 1  
  
 if self.\_count % 2 != 0:  
 rand[tmp1] = tmp\_count  
  
 for key, value in rand.items():  
 if key.rand:  
 f.write(str(value) + ',' + str(key.data) + ',' + str(rand[key.rand]) + ',\n')  
 else:  
 f.write(str(value) + ',' + str(key.data) + ',,\n')  
 f.close()

def deserialize(self, file):  
 *"""  
 Структура словаря, который заполняем, а затем по нему воссоздаём список:  
 key - индекс элемента  
 value - кортеж(экземпляр класса ListNode, индекс элемента по случайной ссылке/или '')  
 экземпляр класса ListNode в кортеже первоначально хранит только значение* ***:param*** *file:* ***:return****:  
 """* f = open(str(file)+".txt", "r")  
 rand = {}  
 for line in f:  
 elems = line.split(',')  
 rand[int(elems[0])] = (ListNode(data=elems[1]), elems[2])  
 f.close()  
 self.\_count = len(rand)  
 if len(rand) == 1:  
 self.\_head = rand[0][0]  
 if rand[0][1]:  
 self.\_head.rand = self.\_head  
 self.\_tail = self.\_head  
 elif len(rand) > 1:  
 self.\_head = rand[0][0]  
 if rand[0][1]:  
 self.\_head.rand = rand[int(rand[0][1])][0]  
 self.\_tail = rand[self.\_count-1][0]  
 if rand[self.\_count-1][1]:  
 self.\_head.rand = rand[int(rand[self.\_count-1][1])][0]  
 tmp1 = self.\_head  
 tmp2 = self.\_tail  
  
 tmp\_count = 0  
 while tmp\_count < self.\_count / 2:  
 tmp1.next = rand[tmp\_count][0]  
 if rand[tmp\_count][1]:  
 tmp1.rand = rand[int(rand[tmp\_count][1])][0]  
 tmp2.prev = rand[self.\_count-tmp\_count-1][0]  
 if rand[self.\_count-tmp\_count-1][1]:  
 tmp2.rand = rand[int(rand[tmp\_count][1])][0]  
 tmp1.next.prev = tmp1  
 tmp2.prev.next = tmp2  
 tmp1 = tmp1.next  
 tmp2 = tmp2.prev  
 tmp\_count += 1  
 if self.\_count % 2 == 0:  
 tmp1.next = tmp2  
 tmp2.prev = tmp1  
 else:  
 tmp1.next = rand[tmp\_count][0]  
 tmp2.prev = rand[tmp\_count][0]  
 rand[tmp\_count][0].next = tmp2  
 rand[tmp\_count][0].prev = tmp1  
 if rand[tmp\_count][1]:  
 rand[tmp\_count][0].rand = rand[int(rand[tmp\_count][1])][0]

**2. Напишите ИИ для противника используя BhvTree**

**https://drive.google.com/file/d/1QisR0PkvXcU-A1wYD\_-2mSXBFXVMEhJM/view?usp=sharing**

****