

**Вербин Олег**

**ВЫПУСКНАЯ КВАЛИФИКАЦИОННАЯ РАБОТА**

**Программная реализация обучающей игры на базе  
беспроводной локальной сети**

**Научный руководитель:**  
м.н.с. Н.В.Баева

# Обучающая игра на географическом факультете

Каждый игрок выбирает одну из доступных **ролей**: *Глава комиссии по ЧС, представитель медицины, представитель географии и т.д.*

Появились заболевшие люди с подозрительными симптомами, есть вероятность эпидемии. Задача игроков – **определить и вылечить болезнь**.

Игрокам доступно несколько **карточек больных** и выделено некоторое количество **монет**. Также игрокам известно о **количестве заболевших и умерших**.

# Ходы

Игра является **пошаговой**. С каждым ходом **количество больных увеличивается**

Каждой роли доступны свои **действия**, которые **стоят монет**. В течении хода игроки предлагают свои действия главе комиссии по ЧС. В конце хода глава комиссии по ЧС решает какие из предложенных действий стоит предпринять

В зависимости от действий игроков им может открываться **дополнительная информация**: *новые карточки больных, эпидемиологические карты, и т.д.*

В игре участвует **ведущий**, который может устраивать случайные события и давать подсказки

# Постановка задачи

- Необходимо разработать приложение для запуска вышеописанной игры на цифровом устройстве
- Дополнительно необходимо разработать приложение для создания и запуска других вариаций вышеописанной игры

# Выделение элементов

Первым делом необходимо определить из каких элементов состоит игра, и придумать каким образом эти элементы можно представить в программе

Элементы игры:

- **Переменные** (*число заболевших и монеты*) – это объекты, которые хранят в себе действительные значения
- **Листы** (*карточки больных, эпидемиологические карты*) – это объекты, которые представляют из себя дополнительную информацию и хранят в себе изображение и флаг видимости (доступно ли данное изображение игрокам, или нет)
- **Роли** (*глава комиссии по ЧС, представитель медицины*) – это объекты, которые определяют список возможных действий игрока

# Выделение элементов

Элементы игры:

- **Функции** – это правила по которым значение переменных изменяется с каждым ходом (*число больных увеличивается*)
- **Действия** (*купить вакцину*) – это объект, который хранит в себе условие на действие, реакцию и варианты ответов
- **Условие на действие** (*у игрока должно быть  $\geq N$  монет*) – это предикат от значений переменных, который говорит, можно ли выполнять действие
- **Реакция** (*отнять  $N$  монет, показать эпидемиологические карты*) – это правило по которому значения переменных и флаги видимости листов изменяются при активации действия

# Правила

Для того, чтобы задавать функции и действия, необходимо формализовать понятие «правило». **Правило** – это сущность, которая должна позволять изменять переменные и листы, опираясь на их нынешнее значение.

Пусть в игре есть  $n$  переменных и  $m$  листов. Составим вектор из  $n$  значений переменных и  $m$  значений флагов видимости листов.

Тогда правило – это отображение  $f : \mathbb{R}^{(n+m)} \rightarrow \mathbb{R}^{(n+m)}$

Условие на действие  $f : \mathbb{R}^n \rightarrow \{0, 1\}$

Для применения правила:

- Составляем вектор из значений переменных и значений флагов видимости листов
- Применяем отображение
- Распаковываем получившийся вектор обратно в переменные и листы

# Отображение

Далеко не все отображения можно точно представить в памяти компьютера, поэтому нужно ввести ограничения, чтобы работать только с удобными отображениями.

Будем рассматривать такие  $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$ , что для них можно разбить пространство  $\mathbb{R}^n$  на конечное число непересекающихся множеств  $R_1 \cup R_2 \cup \dots \cup R_k = \mathbb{R}^n$ , таким образом, чтобы  $\forall i \in [1, 2, \dots, k] \forall x \in R_i f(x) = \vec{f}_i(x)$ , где  $\vec{f}_i$  задаётся вектор формулой из  $\mathbb{R}^n$ , и  $\forall i \in [1, 2, \dots, k - 1] R_i$  задаётся неравенством из  $\mathbb{R}^n$

Теперь, чтобы задать подобное отображение достаточно хранить **k вектор формул** для  $\vec{f}_i$  и **k-1 неравенств** для  $R_i$



# Вектор формула

**Вектор формула**  $\vec{f} = \{f_1, f_2, \dots, f_n\}$ ,  $f_i$  - является формулой,  $\vec{f}: R^n \rightarrow R^n$

$$\vec{f}(x) = \{f_1(x), f_2(x), \dots, f_n(x)\}, \quad x \in R^n, f_i: R^n \rightarrow R$$

**Формула**  $f_i: R^n \rightarrow R$  – это функция от  $n$  вещественных переменных, состоящая из вещественных чисел, заданных функций и операций из  $op$

$op$  -  $\{+, -, /, *, \&, |\}$  математические и логические операции

$x_1 \& x_2 = (x_1 \neq 0) \text{ "логическое и" } (x_2 \neq 0), \quad x_1, x_2 \in R$

$x_1 | x_2 = (x_1 \neq 0) \text{ "логическое или" } (x_2 \neq 0), \quad x_1, x_2 \in R$

Заданные функции – это заранее заданные функции от какого-то количества вещественных переменных. *Sin, cos, max, min и т.д.*

# Примеры

**Пример формулы:**

$$f(x_1, x_2, x_3) = x_1 + x_2$$

$$f(x_1, x_2, x_3) = x_1 * x_2 + \sin(\cos(\max(25/x_1, x_3)))$$

$$f(x_1, x_2, x_3) = x_1 / x_2 \& x_3$$

**Пример задания области с помощью неравенства:**

$$e(x_1, x_2) \rightarrow x_1 * x_1 + x_2 * x_2 < 25 \text{ – внутренность круга}$$

$$e(x_1, x_2) \rightarrow 3 * x_1 > x_2 \text{ - полуплоскость}$$

$$e(x_1, x_2) \rightarrow x_1 * x_1 + x_2 * x_2 < 1 \& x_1 > 0 \& x_2 > x_1 \text{ – } 1/8 \text{ круга}$$

# Множество возможных игр

Формулы и неравенства конечны и легко представимы в памяти компьютера. Теперь можно задать игру.

Однако имея такой инструмент можно сразу задать много игр. Варьируя элементы игры (меняя значения переменных, создавая новые правила и т.д.) получим всё **множество возможных игр**, которые можно таким образом задать.

Теперь необходимо разработать **конструктор сценариев** в котором можно задать любую игру из множества возможных игр, и **приложение** на котором можно запустить любую сконструированную игру



# Разработка конструктора

Необходимо задавать большое количество элементов и набирать много текста, поэтому удобнее всего реализовать конструктор в виде компьютерного приложения.

В качестве графической библиотеки был выбран Tkinter, так как это хороший инструмент, позволяющий создавать кроссплатформенные приложения

- Операционная система – **Windows, MacOS, Linux**
- Язык программирования - **Python**
- Среда разработки - **PyCharm**
- Графическая библиотека – **Tkinter**

# Итоговый конструктор

Переменные					Добавить переменную		Вкладки		Добавить вкладку		Роли		Добавить роль		Описание действия	
Время	0	0	10	pTime	↑	Метро	t1	Хост	rhost	↑	Купить лекарство.					
Деньги	100	0	None	p0	↑	Карты больных	t2	Лидер	rleader	↑	Если правильно выбрать болезнь, то это побед					
Больные	1	0	5000	p1	↑	Климатические карты	t3	Врач	r0	↑	Условие на действие					
Tmp	0	None	None	p2	↑			Мэр	r1	↑	eq(p0,p0-200)					
Прогресс	0	0	100	p4	↑					↑	p4>=60					
Функции Больные					Добавить функцию		Листы Климатические карты		Добавить лист		Действия Врач		Добавить действие		Варианты ответа:	
eq(p1,p1*1.3)					p1_1		t3_0				Купить лекарство. 1		r0_0		Ковид an0	
eq(p1,2*p1)					p1_2						Показать карту больного 0		r0_1		Грипп an1	
															Чума an2	
															Воспаление лёгких an3	
															Реакция	
															Условие на реакцию	
															Добавить реакцию	
															Реакции	
															eq(p4,100) an=0	
															eq(p4,p4-25) an=1   an=2	
															eq(p4,p4+20) an=3	
															Редактировать действие	

Сохранить сценарий  
Условие для победы

# Разработка приложения

Так как интерфейс может легко поместиться на экране телефона – было принято решение вести разработку под мобильные устройства.

Выбрана операционная система Android, так как она занимает около 70% мирового рынка всех мобильных операционных систем

Операционная система – **Android**

Язык программирования - **Java**

Среда разработки - **Android Studio**

Графические библиотеки – **внутренние библиотеки Android**

Организация сети – **WiFi Direct (WiFi P2P)**

# Сетевое взаимодействие

Так-как все игроки будут находиться в одной комнате – организовывать сетевое взаимодействие через интернет излишне. Для реализации сети на близкой дистанции были рассмотрены Bluetooth, WiFi и WiFi Direct.

WiFi Direct – стандарт, позволяющий двум и более Wi-Fi-устройствам общаться друг с другом без маршрутизаторов и общих точек доступа

**WiFi Direct быстрее Bluetooth и работает на больших расстояниях, не требует общей точки доступа, по сравнению с обычным WiFi, а так-же тратит меньше всего зарядки, за счёт умного распределения ресурсов. По этим причинам был выбран WiFi Direct**

# Итоговое приложение

14:38

🔋 51%

ПОДКЛЮЧИТЬСЯ

СОЗДАТЬ КОМНАТУ

15:03

🔋 51%

Введите название комнаты

название комнаты

/storage/emulated/0/Download/test\_scenary/test\_scenary.rp

ВЫБРАТЬ СЦЕНАРИЙ

СОЗДАТЬ КОМНАТУ

15:07

🔋 41%

Service

Название комнаты

Игра на географическом факультете

Ильяс

Глава комиссии по ЧС

Все действия игроков отправляются лидеру

И он самолично выбирает, какое из этих действий отправить хосту

Олег

Хост

Абдул

Представитель медицины

Ильяс

Глава комиссии по ЧС

ОТПРАВИТЬ

15:12

🔋 50%

Service

Нет ограничения на время

Заболевшие

20.0

Мёртвые

2.0

Монеты

700.0

Карантин

0.0

Победа

1.0

Начало анализа

-1.0

Коэффициент заражения

1.0

Номер раунда

2.0

Купить карты с эпид. обстановкой

Цена 100 монет

Купить лекарство. Это единственный завершить игру победой

Цена 200 монет

Вы победили

ЗАВЕРШИТЬ РАУНД

15:08

🔋 51%

Service

ОТПРАВИТЬ ИГРОКАМ





# Основные результаты

- Был выработан метод по формализации конкретной игры
- Было определено множество доступных игр и разработано на Python компьютерное кроссплатформенное приложение для создания всевозможных сценариев
- Было разработано на Java мобильное сетевое приложение для реализации созданных сценариев