Опыт применения игровых технологий при обучении программированию

Нестеров М.В.
nesteroffmaksim@gmail.com
Саратовский государственный университет имени Н.Г.Чернышевского
МОУ «СОШ №83», Саратов, Россия

Знакомство со «взрослым» программированием, как правило, начинается в 8-9 классах, именно в это время формируется отношение детей к данному творческому процессу. Но очень часто можно увидеть, что ученикам не интересно программирование, и ему посвящают время лишь единицы. Эти дети готовы учиться программировать «не благодаря, а вопреки». В данной статье рассмотрена методика обучения программированию И.Р. Дединского. Описана частичная апробация данной методики на учениках 8 класса МОУ «СОШ №83» г. Саратов, а также её итоги.

Ключевые слова: обучение программированию, методика И.Р. Дединского, геймификация на уроках информатики.

Введение

Программирование – раздел изучения информатики, проблемы с которым начинаются в 8-9 классах, когда учащиеся переходят с «детских» языков и сред программирования (Лого, КуМир, PencilCode, Scratch и пр.) на языки «взрослого» программирования (Pascal, C++, Python и пр.). Именно в этот момент можно научить детей «плохому», отбить желание изучать

программирование из-за того, что всё обучение сводится не к решению проблем с помощью программирования, не разработке программ, а лишь к изучения языка программирования как таковому: к изучению конструкций языка и решения задач на использование изученных конструкций.

Вопрос возвращения интереса обучающихся к изучению информатики не нов. [1] Одним из способов является применение современных образовательных технологий. [2]

Не так давно в мире появился термин геймификация – процесс, который стал пронизывать многие сферы жизни общества от образования до бизнеса [3], от программ лояльностей в супермаркетах до получения нового статуса на рабочем месте [4]. Ввиду этого рассмотрение возможности геймификации процесса обучения программированию возникает сам собой.

1. Методика обучения программированию И.Р. Дединского

Один из способов решения названной выше проблемы – методика преподавания Ильи Рудольфовича Дединского – старшего преподавателя МФТИ, а также учителя в физико-математической школе в г. Москва [5]. Его методика в корни отличается от часто преподаваемой: уже к третьему занятию дети знакомятся с функциями (правда, без параметров), к концу 9-го класса у детей уже есть представление о стеках, списках, деревьях, а к концу 10 класса учащиеся уже могут решать задачи промышленного уровня, реализуемого на базе уровня учебной модели:

– моделирование вычислительных систем – nanoCPU – стеки;
– функциональное программирование – nanoLisp – списки;
– сжатие и передача данных – архиватор Хаффмана – деревья;
– трансляция и оптимизация, символьные преобразования – nanoGCC – деревья. [6]

Исследование, описанное в данной статье, полностью не дублирует методику Дединского, при этом используется один из её основных компонентов – геймификация процесса обучения программированию. Апробация данной методики проводилась на учениках 8 класса (изучение информатики 1 час в неделю), перед которыми была поставлена задача в создании консольной игры.

2. Описание апробации методики И.Р. Дединского

Перед тем как перейти к созданию игры, учащиеся в течение половины четверти знакомились с языком программирования C++, с его основными конструкция. В середине четверти была сделана пауза в изучении теоретического материала на деловую игру, в которой все учащиеся превращались в сотрудников компании по программированию, название которой могли выбрать сами. Учитель выступал в качестве заказчика — руководителя квест-комнаты, которому была необходима игра «Дом с привидениями».

За круглым столом заказчик рассказал информацию о себе, а также о самом заказе. На этом этапе ребята могли задать интересующие их вопросы. Таким образом, было сформировано первое впечатление о техническом задании. После беседы с учителем у учащихся было время на проработку плана работы по данному «заказу», после чего каждый мог высказаться о возможных вариантах реализации поставленной задачи.

Стоит отметить, что ввиду проведения данной игры в середине четверти, учащиеся не знали всех основных конструкций, в частности циклов, они имели лишь представление о среде КуМир, в которой так же есть циклы, поэтому данная игра на всём своем протяжении подпитывала интерес учащихся к дальнейшему изучению языка программирования.

Когда был обсужден план работы, когда были намечены ключевые дедлайны (сроки сдачи определённых этапов игры), когда были изучены необходимые для реализации своих планов конструкции языка, учащиеся приступили к написанию программы.

В среднем потребовалось 2 недели на написание данного проекта всеми учениками.

Чтобы детям было проще выполнить свой первый проект по программированию было принято решение о необходимости создания некого шаблона. Части кодов будут приведены ниже.

В листинге 1 приводится описание сюжетной линии, которое встречает разработчика, для более полного погружения в работу.

Листинг 1 – Описание сюжетной линии

```
Описание игры

  В пригороде городка Хэлсвилль уже десяток лет пустует один дом. Ходит легенда, что когда-то там жила семья Митчелл. Маленькая Молли Митчелл очень любила кукол, она собирала их и складывала на полки в своей спальне. На её седьмой день рождения мама пообещала ей купить любую игрушку, какую она пожелает. Молли сразу повела её в магазин подержанных игрушек, где она давно видела одну очень старую куклу, которую никто не покупал, но девочке она очень нравилась.

  Когда мама положила игрушку на прилавок, владелец магазина сказал ей что, к сожалению, эта кукла не продаётся. Мать посмотрела вниз и увидела, что на глазах её дочери наворачиваются слёзы. Она поняла, что во что бы то ни стало должна сдержать обещание и начала торговаться с продавцом. Она предлагала всё более и более выгодную цену, но владелец магазина оставался
```

Продолжение листинга 1

```
  непреклонен. Тогда мама Молли сказала, что заплатит 100 долларов и протянула деньги. Глаза продавца округлились от настолько щедрого предложения, он замялся, но через секунду с жадностью схватил деньги и согласился.

  Молли с мамой были так довольны покупкой, что не обратили никакого внимания на предостережение продавца никогда не оставлять девочку с куклой наедине… Через неделю после дня рождения Молли мама съехала из дома... Дом опустел... Конечно, находились смельчаки, которые хотели раскрыть секреты дома Митчел, но, к сожалению, это никому не удавалось…
```

```
  Сможете ли Вы покинуть дом Митчелл? Преодолеете ли Вы 100 дверей этого дома, чтобы раскрыть тайну куклы Молли?
```

После описания сюжета начинается блок служебной информации, в котором более конкретно описывается учебная задача, а также критерии оценивания. В данной работе были предусмотрены, помимо выполнения основной задачи, дополнительные задания, которые могли не только повысить оценку учащегося, но также и добавить дополнительные оценки.

Критерии оценивания были определены следующим образом:

– «отлично» – работоспособность программы, а также выполнение двух самостоятельных заданий;

– «хорошо» – работоспособность программы, а также выполнение одного самостоятельного задания;

– «удовлетворительно» – работоспособность программы;

– «неудовлетворительно» – неработоспособность программы.

Каждый этап программы был полностью описан на естественном языке, от учащегося требовалось только перевести эту информацию на формальный язык (см. листинг 2, 3, 4).

Листинг 2 – Первый этап игры

```
/*

  Как и в любой игре, предложим представиться пользователю. Для этого нам нужно будет запросить имя пользователя*, которое сохраним в строковой переменной. Для работы со строковыми переменными необходимо подключить библиотеку string.


  Пример работы со строковой переменной:


     string str = ""; // создаем пустую переменную-строку

     cin >> str;      // предположим, что пользователь ввёл имя Vasya

     cout << str;     // на экране увидим: Vasya


  Предложив ввести имя пользователя, а также получив это самое имя, поприветствуем игрока.


     Пример входных данных:  Вася

     Пример выходных данных: Здравствуй, Вася!
```

```
  В ходе игры нам будет необходимо генерировать (спаунить от англ.
spawn - рождение) привидений. Для этого будем пользоваться
генератором случайных чисел rand(), который находится в

стандартной библиотеке C++, поэтому никаких дополнительных
библиотек подключать не нужно.

---------------

* под именем пользователя будем понимать набор символов, кроме
пробельного и символа перехода на новую строку, необязательно
осмысленный

*/
```

Листинг 3 – Второй этап игры

```
  int main() {

  /********************************** ВТОРОЙ ЭТАП ИГРЫ
************************************/

  /************************ Подготовка параметров (аргументов) игры
************************/

  /*

  В ходе игры от пользователя нам необходимо узнавать, какую дверь
он выбирает, а также вести подсчёт верных ответов, который будет
соответствовать уровню в игре: 1 верная дверь - 1 новый уровень.


  Как вы знаете, существует один целочисленный тип данных int. У
него же существует  четыре

  спецификатора signed (знаковый), unsigned (беззнаковый), short,
long. Спецификаторы short

  и long могут использоваться для описания короткого или длинного
целого типов данных. При

  этом название типа int может быть опущено. То есть, short int
соответствует short, а long int — long. Объем памяти, выделяемый
под целочисленные типы, различается в зависимости от компьютера и
компилятора, но можем точно утверждать, что выгоднее использовать
спецификаторы, зная
```

Продолжение листинга 3

```
  примерный диапазон входных данных. К примеру, если мы знаем, что
значение чисел будет невелико, то не имеет смысла использовать
полный тип int, выгоднее использовать short int.
```

```
  Спецификаторы signed (знаковый) и unsigned (беззнаковый) можно
добавлять к любому имени типа. По умолчанию все целочисленные типы
являются знаковыми, поэтому спецификатор signed можно опускать.

  Игра может завершится в одном из двух случаев:

     1) пользователь прошел 100 уровней, следовательно он ПОБЕДИЛ

     2) пользователь ошибся при выборе двери, следовательно он
ПРОИГРАЛ

  Если завершение игры в ходе достижения 100-го уровня понятно
(счётчик достиг значения 100), то завершение игры из-за поражения
вызывает некоторое недоумение. Для решения данной проблемы
создадим так называемый флаг, т.е. переменную которая может иметь
всего два значения: true или false. Пока в переменной хранится
true, мы даём возможность пользователю играть, но как только он
ошибается в выборе двери, мы меняем значение с true на false и
завершаем игру.

  Переменная-флаг будет иметь логический тип bool

  */

    bool bCheck = true; //т.к. в начале игры пользователь ещё ни
разу не ошибся, инициализируем

                  //переменную-флаг как истинное значение

  /*

  Примечание:

  Рекомендую ознакомиться с Венгерской нотацией - правилами
именования переменных и пр.:
https://ru.wikipedia.org/wiki/Венгерская_нотация

  */
```

Стоит отметить, что уже в 8 классе, как и советует И.Р. Дединский, учащиеся знакомятся с нормами «хорошего» программирования, т.ч. учатся правильно давать имена переменным.

Листинг 4 – Третий этап игры

```
  /********************************** ТРЕТИЙ ЭТАП ИГРЫ
************************************/

  /********************************** Функциональная часть
************************************/

  /*

  Игра будет продолжаться, пока пользователь не ошибётся или не
пройдёт 100 уровень, т.е. не
```

```
  достигнет 101 уровень. Это и должно являться основным условие
игры, внутри которого будет

  располагаться остальной функционал


  На каждой итерации пользователю нужно предлагать сделать выбор:
1 из 3 дверей. При этом

  нужно учитывать, что пользователь может ошибиться и выбрать №
такой двери, которой не

  существует, например, №10. В этом случае мы не должны учитывать
его ход, мы должны сообщить

  о неверности выбора и предложить выбрать дверь ещё раз.


  Как только пользователь вводит нормальный номер двери, мы должны
проверить, а угадал ли он

  дверь без приведений, но для начала нам нужно определить, где
будут приведения. Есть 2

  варианта:

     1) случайно заспаунить привидений за 2 двери

     2) определить дверь, которая будет "чистой"

  Конечно же, легче будет организовать 2-ой вариант.


  Для определения "чистой" двери воспользуемся генератором
случайных чисел rand(). Сразу стоит отметить, что ни один
генератор случайных чисел в мире не является генератором
СЛУЧАЙНОГО числа, все они работают по некоторым правилам, но об
этом подробнее поговорим в более старших классах.

  */

     short iClearDoor = rand() % 3 + 1; //rand() задаёт случайное
число от 0 до 32767, но для

                                //нас это большой диапазон. Нам нужен
диапазон [1; 3]

  /*

  Осталось дело за малым:

     - если № двери, которую ввёл пользователь, совпадает со
случайной чистой дверью, то поздравляем пользователя, увеличиваем
количество очков на единицу, и вновь просим ввести дверь, не
забывая про проверку верности ввода
```

```
    - если № двери, которую ввёл пользователь, не совпадает со
случайной чистой дверью, то сообщаем пользователю о поражении,
выводим на экран его счёт, завершаем игру


  Перед началом каждого уровня очищайте экран. Для этого
воспользуйтесь командой system("cls")


  Если при проверке игры, Вам не удаётся прочитать на экране всё,
что выводится, используйте команду system("pause"), вписав её в
необходимую часть кода
  */
  }
```

Хочется отметить, что с целью экономии места в листинге 4 опущена часть, в которой поясняется как добиться смещения диапазона. С полным кодом-шаблоном, а также с некоторыми работами детей, можно ознакомиться на репозитории. [7]

После основной части программы следуют задания на повышение оценки:

1. исправить код таким образом, чтобы в начале каждого уровня демонстрировался его номер;
2. исправить код таким образом, чтобы в начале игры пользователю предлагалось выбрать уровень сложности, согласно которому меняются условия победы и поражения.

Затем следовали дополнительные задания. Под дополнительными заданиями понималось усовершенствование программного кода с помощью конструкций, которые не изучались на уроках, учащимся было необходимо самим найти способы решения поставленных задач:

1. изменить код таким образом, чтобы на протяжении всей игры звучала нагнетающая музыка
2. изменить код таким образом, чтобы в зависимости от ситуации менялся цвет фона
3. изменить код таким образом, чтобы при запуске игры демонстрировался сюжет: звук + изображение/видеоролик

Заключение

С данным проектом справились 23 из 26 учеников, около половины заинтересовались программированием, а примерно четверть изъявила желание подготовиться и участвовать в олимпиаде по программированию, при это делать серьёзные выводы о такой методике преподавания программирования на основе данной апробации нельзя, но анализируя работы учеников и студентов И.Р. Дединского, анализируя опыт других преподавателей, взявшихся также апробировать данную методику, можно заявить, что это большой прогресс в области обучения детей программированию. Но, как и любой прогресс, данный способ обучения имеет свои недостатки, главными из которых являются высокие

требования к преподавателю: преподаватель должен обладать опытом практического программирования и уметь быстро разбираться в коде учеников, оценить его состояние и последствия его развития, представить себе, мог ли подобный фрагмент кода быть использован в реальном проекте.

Помимо данного негативного аспекта, также можно выделить и слабое техническое оснащение школ, что не позволяет создавать крупные проекты. Решить данную проблему может сотрудничество школ с ведущими ВУЗами региона, которые бы могли предоставлять возможность детям реализовать свои проекты с использованием более мощных средств, имеющихся на балансе данных высших учебных заведений, а также выстроить преемственность довузовского и вузовского обучения программированию.

Список литературы

[1] *Храмова М.В., Чабан М.А.* Как вернуть мотивацию к изучению информатики посредством современных образовательных технологий? // Материалы Шестнадцатой открытой Всероссийской конференции: Преподавание информационных технологий в Российской Федерации. - М.: Московский государственный технический университет имени Н.Э. Баумана (национальный исследовательский университет), 2018. - С. 320-322.

[2] *Чабан М.А.* Применение современных образовательных технологий на уроках информатики // Материалы VIII Международной научно-практической конференции: Информационные технологии в образовании "ИТО-2016". - Саратов: ООО "Издательский центр "Наука", 2016. - С. 122-125.

[3] Геймификация в бизнесе: как пробиться сквозь шум и завладеть вниманием сотрудников и клиентов / Гейб Зикерманн, Джоселин Линдер; пер. с англ. Иделии Айзятуловой. - М.: Манн, Иванов и Фербер, 2014. - 272 с.

[4] Профессия: геймификатор. «Это искусство на грани аналитической работы и чистого креатива» // www.properm.ru / URL: https://properm.ru/news/society/119702 (дата обращения: 25.10.2020).

[5] Опыт обучения школьников программированию // www.habr.com / URL: https://habr.com/ru/post/179307 (дата обращения: 12.01.2020).

[6] Аналитический подход к довузовскому преподаванию программирования // www.ded32.ru / URL: http://storage.ded32.net.ru/Lib/Doc/Analytic Approach2010.pdf (дата обращения: 12.01.2020)

[7] Репозиторий MaximNesterov // www.github.com / URL: https://github.com/MaximNesterov?tab=repositories (дата обращения: 30.09.2020)

[8] Сайт методики довузовского обучения программированию и проектной деятельности в информатике // http://ded32.net.ru (дата обращения: 12.01.2020)