

Р. Р. Мухаметзянов,

Набережночелнинский государственный педагогический университет, Республика Татарстан

ОБУЧЕНИЕ ОБЪЕКТНО-ОРИЕНТИРОВАННОМУ ПРОГРАММИРОВАНИЮ

Аннотация

Растущий рынок информационных и коммуникационных технологий требует повышения качества подготовки ИТ-специалистов. Одним из приоритетных направлений подготовки ИТ-специалистов является подготовка квалифицированных программистов по объектно-ориентированному программированию (ООП). Подготовку в области объектно-ориентированного программирования необходимо начинать уже со школьной скамьи и продолжать далее в ССУЗе и вузе.

Ключевые слова: объектно-ориентированное программирование, язык программирования, C#, Java, федеральный государственный образовательный стандарт, преемственность в обучении объектно-ориентированному программированию.

В нашей стране изучению информатики и информационно-коммуникационных технологий в последнее время уделяется большое внимание, так как в Российской Федерации по-прежнему не хватает квалифицированных ИТ-специалистов. Растущий ИТ-рынок требует все больше и больше разработчиков, веб-программистов, специалистов по робототехнике и т. д. Министерство образования и науки РФ пытается по-своему решить данную проблему, увеличивая количество бюджетных мест на ИТ-направления подготовки высшего образования и специальности среднего специального образования, а также реализуя различные гранты по информационным технологиям. Все это, конечно, благоприятно влияет на подготовку ИТ-специалистов в нашей стране. Но только административными мерами кардинально решить данную проблему невозможно. Необходимо, прежде всего, изменить содержание подготовки будущих ИТ-специалистов и, наверно, самое главное — обеспечить преемственность данной подготовки. Сегодня, к сожалению, пока еще отсутствует четко выстроенная преемственность в этой подготовке в системе «школа — ССУЗ — вуз». В своей работе мы хотели бы рассмотреть всего лишь одну небольшую часть модели преемственности, которая, на наш взгляд, должна существовать и должна быть четко выстроена. *Среди огромного количества направлений в подготовке будущих*

ИТ-специалистов можно и нужно выделить подготовку будущих программистов по объектно-ориентированному программированию (ООП), так как именно эта парадигма программирования является в последние годы самой востребованной и популярной в мире вообще и в нашей стране в частности.

Существует огромное количество языков программирования. Basic, Pascal, Python, C, C++... Этот список можно продолжать очень долго. Среди этих языков программирования наибольшей популярностью сегодня пользуются именно объектно-ориентированные языки. Достаточно взглянуть на статистику сайта tiobe.com (см. рис.) и можно увидеть, что в первой десятке такие объектно-ориентированные языки программирования, как Java, C#, C++, Visual Basic.NET [5]. Причем популярность их из года в год только растет.

Такие языки программирования, как Delphi, Python, PHP или JavaScript, также имеют возможности объектно-ориентированного программирования. Однако назвать их типичными объектно-ориентированными языками по большому счету нельзя, они лишь имеют определенные объектно-ориентированные возможности. Несомненно, эти языки программирования тоже могут способствовать формированию объектно-ориентированного стиля программирования, и их нельзя сбрасывать со счета. Постараемся

Контактная информация

Мухаметзянов Рамиль Рафаилович, канд. пед. наук, декан факультета математики и информатики, доцент кафедры информатики и вычислительной математики Набережночелнинского государственного педагогического университета, Республика Татарстан; *адрес:* 423806, Республика Татарстан, г. Набережные Челны, пер. Железнодорожников, д. 9А; *телефон:* (8552) 46-71-15; *e-mail:* mrr-nisptr@mail.ru

R. R. Mukhametzyanov,

Naberezhnye Chelny State Pedagogical University, The Republic of Tatarstan

TRAINING IN OBJECT ORIENTED PROGRAMMING

Abstract

The growing market of information and communication technologies requires improving the quality of training of its specialists. One of the priority directions of training of IT-specialists is the training of skilled programmers in object oriented programming (OOP). Training in the field of object oriented programming is needed to start from school and continue in colleges and universities.

Keywords: object oriented programming, programming language, C#, Java, Federal State Educational Standard, continuity in learning object oriented programming.

Feb 2017	Feb 2016	Change	Programming Language	Ratings	Change
1	1		Java	16.676%	-4.47%
2	2		C	8.445%	-7.15%
3	3		C++	5.429%	-1.48%
4	4		C#	4.902%	+0.50%
5	5		Python	4.043%	-0.14%
6	6		PHP	3.072%	+0.30%
7	9	^	JavaScript	2.872%	+0.67%
8	7	v	Visual Basic .NET	2.824%	+0.37%
9	10	^	Delphi/Object Pascal	2.479%	+0.32%
10	8	v	Perl	2.171%	-0.08%

Рис. Статистика сайта tiobe.com на февраль 2017 года

выстроить модель обучения объектно-ориентированному программированию, начиная со средней общеобразовательной школы и до высшей школы.

Современное состояние преподавания информатики и ИКТ характеризуется переходом на федеральные государственные образовательные стандарты. ФГОС основного общего образования в качестве предметных результатов обучения определяет:

- развитие алгоритмического мышления, необходимого для профессиональной деятельности в современном обществе;
- развитие умений составлять и записывать алгоритм для конкретного исполнителя;
- формирование знаний об основных алгоритмических конструкциях;
- знакомство с одним из языков программирования и основными алгоритмическими структурами (линейной, условной и циклической) [3].

Стандарт среднего общего образования ставит целью:

- владение умением понимать программы, написанные на выбранном для изучения универсальном алгоритмическом языке высокого уровня;
- знание основных конструкций программирования;
- умение анализировать алгоритмы с использованием таблиц;
- владение стандартными приемами написания на алгоритмическом языке программы для решения стандартной задачи с использованием основных конструкций программирования и отладки таких программ [4].

Конечно, в школьных стандартах об изучении объектно-ориентированного программирования явно ничего не сказано. В то же время и не оговорено четко, какой именно язык программирования следует использовать для достижения предметных результатов обучения по информатике. Если рассмотреть задания ОГЭ и ЕГЭ по информатике, таких языков на сегодня пять: алгоритмический язык, языки Basic, Pascal, C и Python. С точки зрения пропедевтики для

дальнейшего изучения объектно-ориентированных языков C++, C# или Java оптимальным вариантом является, конечно, изучение в *средней школе* языка программирования C. На уровне средней общеобразовательной школы можно изучить синтаксис языка и реализацию на нем основных алгоритмических конструкций. Можно рассмотреть также темы, связанные с созданием функций и массивов в этом языке программирования. Изучение принципов работы с файловой системой компьютера на этом языке просто необходимо для учащихся, участвующих в олимпиадах по информатике (программированию). Изучение языка программирования C будет являться прочным фундаментом для изучения ООП на языках C++, C# и Java, так как синтаксис этих языков фактически одинаков и все они относятся к одному семейству так называемых C-подобных языков программирования. В ходе дальнейшего обучения в вузе уже можно рассматривать темы, связанные с разработкой пользовательских классов и использованием встроенных классов языка.

В качестве объектно-ориентированного языка для обучения в *вузе* оптимальным, на наш взгляд, является использование языков C# и (или) Java. Причин этому несколько.

- Во-первых, именно на этих языках программируют большинство современных разработчиков.
- Во-вторых, языки C# и Java постоянно совершенствуются и обновляются.
- В-третьих, существуют удобные, а самое главное, бесплатные, интегрированные среды разработки на этих языках. Например, для языка Java это Eclipse, Net Beans. Для языка C# это, конечно же, MS Visual Studio, точнее, ее бесплатная учебная версия Community.
- В-четвертых, на сегодняшний день существует огромное количество технологий разработки не только десктопных, но и мобильных веб-приложений на этих языках.

Достаточно упомянуть приложения под Android, веб-технологии JSP, ASP.Net и др., и вы поймете,

насколько распространенными и удобными являются языки программирования C# и Java. Можно сказать, что эти два языка программирования образуют свою экосистему, особенно язык Java, на котором сегодня программируют и роботы, и бытовую технику, и т. д. Не представляется возможным отдать явное предпочтение какому-то одному из этих двух языков, и тот и другой заслуживают уважения. Java появился исторически раньше и занимает в рейтинге tiobe.com первое место вот уже несколько лет, но у языка C# имеются свои неоспоримые плюсы.

Чтобы оценить объектно-ориентированные возможности, преимущества и недостатки этих двух языков программирования, постараемся дать им сравнительный анализ на конкретных примерах. Примеры будут достаточно простыми, из обыденной жизни, так, чтобы их могли понять не только преподаватели и студенты, но даже школьники. Уверен, что многих заинтересует объектно-ориентированная парадигма программирования и наша публикация даст очередной толчок как преподавателям, так и учащимся в изучении ООП.

Опыт работы и внедрения объектно-ориентированной парадигмы программирования позволяет нам предложить следующую **схему преемственности изучения ООП**:

Школа	ССУЗ, вуз (1-й курс)	Вуз (2-й курс)	Вуз (3-й курс)
Изучение языка программирования C	Изучение ООП на языке C# или Java	Изучение разработки веб-приложений на ASP.Net или JSP	Изучение разработки мобильных приложений на C# или Java

Объектно-ориентированное программирование, как говорят, держится на трех китах: наследование, инкапсуляция и полиморфизм. Только когда обучаемый поймет сущность этих понятий и научится реализовывать их на примерах, можно сказать, что он владеет методикой ООП.

В качестве основных понятий ООП рассматриваются «объект» и «класс».

Объект — это некоторая сущность, которая обладает свойствами и каким-то поведением. Например, машина отца имеет модель, цвет. Она может ездить, перевозить людей и грузы. Береза, которая растет во дворе школы, имеет возраст, высоту, она растет, распускает листья и т. д. Полезно рассмотреть также программные объекты на примере какой-либо компьютерной программы, с которыми учащиеся часто сталкиваются, например, в компьютерных играх или в офисном пакете. Нам часто приходится видеть такие программные объекты, как «кнопка», «радиокнопка», «текстовое поле ввода» и т. д. Они имеют такие свойства, как «длина», «ширина», «цвет».

Понятие **класса** является более абстрактным и позволяет объединить множество объектов в одну группу согласно их общим свойствам и поведению. Говорят, что класс — это шаблон, с помощью которого описываются объекты — экземпляры класса.

Легко представить себе такие классы, как «Книга», «Автомобиль», «Учитель» и др. Если в обыденной жизни классы позволяют нам абстрагироваться от несущественных свойств и методов объектов, в программном коде благодаря использованию классов разработка приложений упрощается и сокращается объем программного кода [2].

Для создания класса и в языке C#, и в языке Java используется ключевое слово *class*.

Рассмотрим всем знакомый класс «Книга» со свойствами: название, автор, количество страниц.

Реализация на языке C#	Реализация на языке Java
<pre>class Book { public string title; public string author; public int pages; }</pre>	<pre>class Book { public String title; public String author; public int pages; }</pre>

Как видите, отличие только в написании типа *string*, так как на самом деле в языке Java не существует встроенного базового типа *string*, как в языке C#, — строка реализуется через класс *String*. Во всем остальном все идентично, и тело класса записывается в фигурных скобках. Оба языка являются регистрозависимыми, поэтому необходимо учитывать регистр символов при написании программного кода.

На самом деле, если быть более точными, в нашем классе описаны три поля, а не три свойства. То, что мы обычно называем свойствами класса, в обыденной жизни в терминологии ООП принято называть полями класса [1]. Свойства класса в ООП имеют немного иное предназначение и реализуются по-другому.

Кроме полей класса практически никогда мы не можем обойтись без методов класса, которые задают поведение объектов класса. Машины могут ездить, люди — есть, деревья — расти и т. д. В качестве очередного примера рассмотрим класс «Человек» со свойствами: имя, возраст, рост.

Реализация на языке C#	Реализация на языке Java
<pre>class Person { public string name; public int age; public double height; }</pre>	<pre>class Person { public String name; public int age; public double height; }</pre>

Человек обладает следующим поведением: ест, ходит, бегает, спит, читает книги и т. д. Реализовать каждый из этих методов в виде программного кода не представляется возможным да и не требуется. Как мы сможем реализовать метод «Есть»? В этом процессе задействовано слишком много сложных биологических и физиологических процессов. Можно, конечно, реализовать этот метод таким образом, что прием пищи приводит к увеличению массы человека или просто выводит на экран сообщение вида: «Я поел. Спасибо!» Тогда у нас получится, например, следующий вариант класса «Человек» с методом *Info()*:

Реализация на языке C#	Реализация на языке Java
<pre>class Person { public string name; public int age; public double height; public void Info(){ Console.Write ("Я поел. Спасибо!"); } }</pre>	<pre>class Person { public String name; public int age; public double height; public void Info(){ System.out.print ("Я поел. Спасибо!"); } }</pre>

На самом деле и в таком методе нет особой необходимости. Часто приходится выводить информацию об объектах класса на экран. Поэтому, чтобы избежать дублирования однотипного программного кода, добавим в наш класс метод, который выводит на экран полную информацию о человеке. Метод называется *ShowInfo()*.

Реализация на языке C#	Реализация на языке Java
<pre>class Person { public string name; public int age; public double height; public void ShowInfo(){ Console. Write("Имя: {0}, возраст: {1}, рост: {2} ", name, age, height) } }</pre>	<pre>class Person { public String name; public int age; public double height; public void ShowInfo(){ System.out. printf("Имя: %s, возраст: %d, рост: %f ", name, age, height); } }</pre>

Отличие двух реализаций метода заключается только в способе вывода информации на экран. В языке C# для этого используется встроенный класс *Console*, а в языке Java — класс *System* и их соответствующие методы.

Классы разрабатываются с одной главной целью: чтобы на их основе можно было бы создавать объекты — **экземпляры класса**. Например, класс «Человек» нужен, чтобы описывать множество людей с разными именами, ростом и возрастом и др. Поэтому следующим важным моментом является создание экземпляра класса. В этом отношении языки также идентичны. Для создания объекта класса используется запись вида:

```
Person brother = new Person();
```

Зададим значения полям объекта. Для этого используется точечная нотация:

```
brother.name="Андрей";
brother.age=23;
brother.height=1.86;
```

Каждый раз обращаться к полям класса для создания объекта неудобно, особенно если требуется создать много экземпляров класса. В этом случае удобно использовать **конструктор класса**. Это следующее важное понятие при изучении ООП. Конструктор класса — это специальный метод класса, который можно вызвать при создании его объекта. Добавим в наш класс «Человек» конструктор:

Реализация на языке C#	Реализация на языке Java
<pre>class Person { public string name; public int age; public double height; public Person(string name, int age, double height) { this.name=name; this.age=age; this.height= height; } }</pre>	<pre>class Person { public String name; public int age; public double height; public Person(String name, int age, double height) { this.name=name; this.age=age; this.height= height; } }</pre>

Как мы видим, реализация конструктора практически идентична в обоих языках программирования. Самое главное, благодаря конструктору можно создать экземпляр класса следующим образом:

```
Person sister = new Person("Марина", 18, 1.68)
```

Говоря об объектно-ориентированном программировании, обязательно необходимо рассказать о **принципе наследования**. Это, наверное, главный и самый известный принцип ООП, когда один класс может наследовать поля и методы другого — родительского класса (суперкласса). Реализация наследования классов в языках C# и Java отличается:

Реализация на языке C#	Реализация на языке Java
<pre>class Student: Person { public string faculty; public int group; }</pre>	<pre>class Student extends Person { public String faculty; public int group; }</pre>

Чтобы объявить один класс наследником другого, в языке Java после имени класса-наследника используется ключевое слово *extends*, после которого идет имя родительского класса, а в языке C# просто ставится двоеточие.

Для разработки веб-приложений на языке C# используется объектно-ориентированная технология ASP.NET. Хотя сегодня существует не так много сайтов на ASP.NET (по крайней мере, если сравнивать с языком PHP), технология заслуживает пристального внимания, так как сам подход является исключительно объектно-ориентированным и построен на веб-формах. Все основные элементы, размещаемые на веб-страницах, реализованы через классы языка C#. Изучение технологии ASP.NET позволит усилить объектно-ориентированную подготовку студентов и помочь им в понимании основных принципов ООП. Аналогичная технология разработки веб-приложений на языке Java называется JSP. Смысл их одинаков.

В мире мобильных приложений сегодня выделяются две основные операционные системы — Android и iOS. Кроме того, в последнее время динамично развивается ОС Windows Phone. С точки зрения разработки мобильных приложений язык C# пока сильно уступает языку Java. Большинство приложений для ОС Android пишут именно на языке Java. Компания Microsoft, пытаясь восполнить данный

пробел, разработала платформу Xamarin, которая набирает из года в год все большую популярность. Главным преимуществом Xamarin является то, что это единый инструмент, который позволяет создавать приложения сразу для нескольких платформ.

Объектно-ориентированный подход в программировании на сегодняшний день является самым популярным и перспективным. Необходимо выстроить правильную модель изучения данной парадигмы программирования во всех ее аспектах. Не стоит забывать и о том, что ООП используется сегодня для разработки и оконных приложений, и веб-приложений, и мобильных приложений. Правильно выстроенная модель обучения ООП позволит сформировать объектно-ориентированный стиль программирования и объектно-ориентированный тип мышления современного программиста.

Список использованных источников

1. Мухаметзянов Р. Р. Объектно-ориентированный подход для изучения массивов // Информатика и образование. 2013. № 3.
2. Мухаметзянов Р. Р., Минегалиева И. Д. Основы программирования на Java: учебное пособие. Набережные Челны: НГПУ, 2016.
3. Федеральный государственный образовательный стандарт основного общего образования (в ред. Приказа Минобрнауки России от 29.12.2014 № 1644). http://минобрнауки.рф/документы/938/файл/749/приказ_Об_утверждении_1897.pdf
4. Федеральный государственный образовательный стандарт среднего общего образования (в ред. Приказа Минобрнауки России от 29.12.2014 № 1644). http://минобрнауки.рф/документы/2365/файл/736/12.05.17-Приказ_413.pdf
5. TIOBE Index. <http://www.tiobe.com/tiobe-index/>

НОВОСТИ

Три новых языка программирования

Сегодня в мире существуют уже сотни языков программирования, так зачем же прикладывать серьезные усилия к созданию еще одного нового? Разработчики трех новомодных языков с открытым исходным кодом — Coconut, Crystal и Oden — отвечают на этот вопрос очень просто: все дело в особенностях программирования. Регулярно появляются новые ниши, требующие заполнения, и новые потребности, которые нужно удовлетворять.

Не желая мириться с ограничениями уже существующих инструментов и преследуя вполне конкретные цели, разработчики создают собственные языки. Язык Coconut, компилируемый в код Python, привносит в него средства функционального программирования. Crystal объединяет в себе преимущества сразу нескольких языков. А причиной разработки Oden стало отсутствие некоторых возможностей в популярном языке Go, продвигаемом Google.

Каждый из разработчиков трех новых языков уже видит для себя конкретные сценарии их использования.

Oden хорош при решении задач «на территории» Go. В качестве примера здесь можно привести веб-серверы, различные сервисы и инструменты командной строки. Он подходит для создания библиотек с обобщенными структурами данных, которые определяются пользователями, типовых алгоритмов и абстракций управления потоками.

Язык Crystal благодаря неблокируемому вводу-выводу и облегченным процессам подходит для веб-сервисов. На нем уже написан ряд приложений командной строки, эмуляторов, веб-сайтов и ботов IRC. Crystal можно использовать также при создании конкурентных программ, когда нужно быстро получить рабочий прототип, обладающий приемлемой производительностью. И наконец, он может быть задействован для проектирования компиляторов, в том числе и компиляторов самого Crystal.

Язык Coconut подходит для написания тех же приложений, что и Python. Весьма обширный спектр этих приложений и обеспечил Python его сегодняшнюю популярность. Поскольку программы на Coconut компилируются в код Python, новый язык обладает всеми возможностями Python. В то время как Crystal и Oden находятся еще на начальной стадии разработки, у Coconut в июне уже была выпущена версия 1.0, а в июле появилась и версия 1.1.

Первоначально Crystal был написан на Ruby, но затем его компилятор переписали непосредственно на Crystal. Компилятор Coconut написан на Python. Первый компилятор Oden был создан на Racket, а процедура приведения типов — на MiniKanren, реляционном языке программирования, встроенном в Scheme. Впоследствии компилятор переписали на Haskell.

(По материалам международного компьютерного еженедельника «Computerworld Россия»)