УДК 378.147 + 519.682

ОСОБЕННОСТИ ОБУЧЕНИЯ ПРОГРАММИРОВАНИЮ НА ОСНОВЕ ОБЩНОСТИ И РАЗЛИЧИЯ ПРИНЦИПОВ

Широкова О. А.

ФГАОУ ВПО «Казанский (Приволжский) федеральный университет», Казань, Россия (420008, Казань, ул. Кремлевская, 18), e-mail: oshirokova@mail.ru

В статье рассматриваются особенности обучения студентов различным принципам программирования. Это позволяет сформировать умение моделировать и проектировать предметную область разными стилями, основанными на разных парадигмах программирования: процедурной, объектно-ориентированной, функциональной, логической. Объекты используются при моделировании, проектировании и программировании в различных областях. Понятие объекта является общим в каждой из парадигм. Однако механизмы, описывающие структуру объекта и действия с ним, реализуются в разных парадигмах по-разному. Каждый стиль программирования требует своего подхода к решению задач. Для решения вычислительных задач лучше пользоваться принципами процедурного и объектно-ориентированного программирования. Для проектирования баз знаний и экспертных систем используются принципы логического и функционального программирования. При обучении программированию нужно использовать систему специально подобранных задач, при решении которых студенты изучают особенности использования различных принципов и методов обработки информации.

Ключевые слова: парадигмы программирования: процедурная, объектно-ориентированная, функциональная, логическая; общность и различия принципов программирования; объекты.

TRAINING IN PROGRAMMING ON THE BASIS OF THE COMMUNITY AND DISTINCTION OF THE PRINCIPLES

Shirokova O.A.

Kazan (Volga region) federal university , Kazan, Russia (420008, Kazan, street Kremlyovskaya, 8), e-mail: oshirokova@mail.ru

The article discusses the features of teaching students to different principles of programming. This allows to create the ability to simulate and design subject area different styles based on different programming paradigms: procedural, object-oriented, functional, logical. Objects used in the modeling, design, and programming in different areas.The concept of object is common to each of paradigms. However the mechanisms describing structure of object and action with it are realized in different paradigms differently. Each style of programming requires a different approach to solving problems. For solving computational problems is better to use the principles of procedural and object-oriented programming. For the design of knowledge bases and expert systems used the principles of logic and functional programming. When teaching programming is necessary to use a system of specially selected tasks, at which decision students study features of use of various principles and methods of information processing.

Keywords: programming paradigms: procedural, object-oriented, functional, logical; community and distinctions of the principles of programming , objects.

Программирование занимает важное место в системе подготовки учителей математики и информатики. Система обучения программированию основана на интеграции различных принципов, разных парадигм программирования. Выделяют следующие парадигмы программирования:

- процедурная;
- объектно-ориентированная;
- функциональная;
- логическая.

В каждой из парадигм программирования вводится понятие объекта. Объекты используются при моделировании, проектировании и программировании в различных областях. Понятие объекта является общим в каждой из парадигм. Однако механизмы, описывающие структуру объекта и действия с ним, реализуются в разных парадигмах по-разному. Например, в объектно-ориентированном программировании программу можно рассматривать как набор взаимодействующих объектов, а в логическом – между объектами моделируются логические отношения при помощи совокупности утверждений на формальном логическом языке.

Цель исследования: формирование в процессе обучения программированию умений и навыков проектировать предметную область разными стилями на основе общности и различия принципов программирования.

Объекты исследования: особенности обучения различным принципам программирования.

Каждый стиль программирования требует своего подхода к решению задач. Для решения вычислительных задач лучше пользоваться принципами процедурного и объектно-ориентированного программирования. Для проектирования баз знаний и экспертных систем используются принципы логического и функционального программирования.

В объектно-ориентированном программировании (ООП) программу можно рассматривать как набор взаимодействующих объектов, поведение которых реализуется с помощью методов, представленных в виде процедур. Процедуры лежат в основе процедурного программирования. Именно при изучении процедурного программирования вводится принцип разбиения задачи на подзадачи, определения их иерархии, механизмы передачи параметров в процедуры, рассматривается структура модулей, в которых размещаются процедуры. Знания структуры модулей и их реализации важны на этапе изучения объектно-ориентированного стиля, поскольку именно в модули помещаются определения классов и объектов.

Объектно-ориентированный стиль применяется при разработке широкого круга приложений. Моделирование реальных объектов с помощью классов объектно-ориентированного языка программирования часто является более эффективным и естественным, чем при процедурном программировании. С помощью объектов могут быть представлены системы самой разной природы. Объектный подход применяется в проектировании интерфейса пользователя, баз данных, архитектуры компьютеров.

Важнейшими компонентами объектного подхода являются абстрагирование, инкапсуляция, модульность и иерархия[2,5]. Абстрагирование – это процесс выделения абстракций в изучаемой предметной области. Инкапсуляция – объединение всех свойств объекта, составляющих его состояние и поведение, в единую абстракцию и ограничение

доступа к реализации этих свойств. Модульность – принцип разработки программы, предполагающий реализацию ее в виде отдельных частей (модулей). Иерархия – ранжированная или упорядоченная система абстракций.

Решение задач на основе объектно-ориентированной парадигмы можно разбить на следующие этапы:

- постановка задачи, выделение объектов на основе анализа классов (объектно-ориентированный анализ);
- определение и реализация связей между объектами, эволюция взаимодействий и наследования классов, модификация классов и их распределение по модулям (объектно-ориентированное проектирование);
- реализация на объектно-ориентированном языке (объектно-ориентированное программирование).

При обучении студентов объектно-ориентированному стилю программирования необходимо выработать умение формализовать задачу, выделять абстракции и объекты данной предметной области, структурировать их и реализовать их с помощью объектно-ориентированной технологии программирования. Например, описание класса TStudent объектов типа студент в Delphiможет иметь вид:

```
Type TStudent=class
        fam:string;
        gr:integer;
        procedureSetfam(x:string);
        functionGetfam:string;
        procedureSetgr(y:integer);
        functionGetgr:integer;
end;
```

Таким образом, объекты типа TStudent будут иметь два поля данных: фамилия и номер группы, а также методы чтения и записи каждого из полей.

При подборе учебных задач нужно учитывать развитие объектно-ориентированной технологии программирования, демонстрируя различие подходов при решении одной и той же задачи. Студентов нужно обучать применению знаний в реальных ситуациях, расширять сферу возможного применения ООП. Для этого рекомендуется решать задачи, имеющие объекты, прототипами которых являются реально существующие математические объекты и структуры. Базовыми понятиями линейной алгебры и аналитической геометрии являются вектор и матрица. Их моделью в алгоритмических языках являются массивы. Для разработки проекта решения задач аналитической геометрии можно создать класс Tmas для объектов типа «массив» с описанием полей, методов и свойств массивов, необходимых для решения задач.

```
typeTMas=class
protected
Orig: pointer;  jMin, jMax:integer;
functionElemP(j:integer):RealP; {определяетадрес j элемента}
public
constructor Create(jMin_,jMax_:integer);
destructor Destroy; override;
procedure Clearance; {методдлясозданиянулевогомассива}
procedureAdd(x:TMas); {метод сложения элементов массивов}
procedureSub(x:TMas); {метод вычитания элементов}
procedureMul(x:TMas); {метод умножения элементов}
procedureMulx(x:real); {метод умножения элемента на число}
procedureDivx(x:real); { метод деления элемента на число }
FunctionSum:real; {метод сложения элементов массива}
FunctionPMn(x:TMas;y:TMas):real;  {метод вычисления площади
ориентированного многоугольника}
end;
```

Таким образом, возможности ООП можно эффективно использовать при реализации алгоритмов вычислительного типа.

Способность студентов мыслить «объектно» формируется при разработке визуальных приложений с использованием стандартных объектов (компонентов) системы объектно-ориентированного программирования, например систем Delphi и C++ [2,3]. Учащихся привлекает возможность создания графического интерфейса приложения из визуальных компонентов этих систем. Например, при изучении курса технологии объектно-ориентированного программирования студентам в числе многих других проектов предлагается создать проект для решения задач аналитической геометрии. Интерфейс проекта создается с помощью меню, размещенного на форме. Меню диалогового окна позволяет решать поставленные задачи (Рис.1а,1б):

Рис.1а.

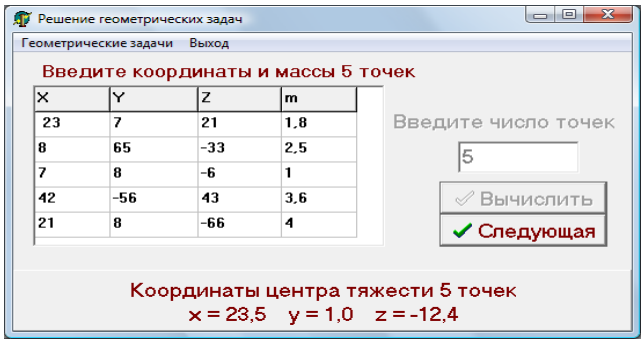
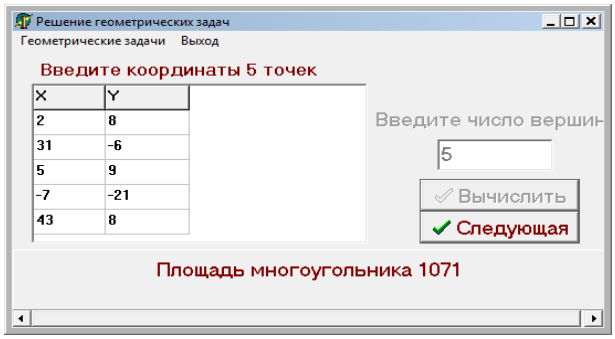


Рис.1б.



Системы обучения, построенные на основе функциональной и логической парадигм, исследованы недостаточно [1] по сравнению с системами обучения на основе парадигм процедурного и объектно-ориентированного программирования.

Логическое программирование относится к области искусственного интеллекта. Основой логического подхода является булева алгебра. Появление нечетной логики повысило выразительность логического подхода. Нечетная логика широко применяется при моделировании [6].

Язык Пролог предназначен для решения не вычислительных, а логических задач, для моделирования процесса логического умозаключения человека [1]. Факты и правила программы на Прологе являются описанием отношений и связей между объектами некоторой предметной области, т.е. записью условия некой логической задачи, которую предстоит решить. Описанные отношения и связи рассматриваются статически. Такой подход к программе называется декларативным. Все это позволяет отнести Пролог к языкам сверхвысокого уровня. Пролог особенно хорошо приспособлен для решения задач, основу которых составляют объекты и отношения между ними [1,5]. Рассмотрим это на примере программы:

```
любит(оля,чтение).
любит(света,бадминтон).
любит(оля,бадминтон).
любит(лена,плавание).
любит(лена,чтение).
?- любит(X,чтение), любит(X,плавание).
```

Запрос означает: есть ли люди, которым нравится и чтение, и плавание? Сначала Пролог ищет факт, сопоставимый с первой частью вопроса: любит(X, чтение). Подходит первый же факт программы

любит(оля,чтение).

и переменная X связывается значением «оля». В то же время Пролог фиксирует в списке фактов указатель, показывающий состояние процедуры поиска. Далее Пролог пытается согласовать вторую часть запроса при условии X = оля, т.е. ищет с самого начала программы факт «любит(оля, плавание)». Такого факта в программе нет. Тогда Пролог возвращается к первой части запроса: любит(X,чтение) , «развязывает» переменную X и продолжает поиск подходящих фактов, начиная с ранее установленного в списке фактов указателя Подходит факт «любит(лена,чтение)», переменная X конкретизируется значением «лена», и далее вторая часть вопроса успешно согласуется с фактом «любит(лена, плавание)». Пролог выполнил в данном примере поиск с возвратом. Графически процесс выполнения программы представляется в виде обхода бинарного дерева – дерева вывода.

В программировании, помимо процедурного и объектно-ориентированного подходов, а также логического подхода, представленного языком Пролог, существует еще одно направление — функциональное. Оно возникло вместе с созданием языка программирования Лисп. Подавляющее большинство программ искусственного интеллекта составлено на языке Лисп [4].Списки в языке Лисп — основное средство представления знаний, так же как объекты в Delphi. Например, с помощью вложенных списков может быть представлена характеристика человека:

```
(сотрудник
(имя Петр)
(отчество Петрович)
(фамилия Иванов)
( образование (среднее (с 1980 по 1990))
                (высшее (КГУ г. Казань (с 1990 по 1995)
                (МГУ г. Москва (с 1995 по 2014)) (специальность
                (техническая кибернетика)
                (программирование)
(стаж (с 1995 по 2014)
)
```

Используемый в языке Лисп функциональный подход к программированию основывается на той идее, что вся обработка информации и получение искомого результата могут быть представлены в виде вложенных и/или рекурсивных вызовов функций, выполняющих некоторые действия, так что значение одной функции используется как аргумент другой. Значение этой функции становится аргументом следующей, и так далее, пока не будет получен конечный результат — решение задачи. Программы строятся из

логически расчлененных определений функций. Определения состоят из управляющих структур, организующих вычисления, и из вложенных вызовов функций. Основными методами функционального программирования являются композиция и рекурсия. Все это представляет собой реализацию идей теории рекурсивных функций.

Таким образом, при обучении программированию нужно использовать систему специально подобранных задач, при решении которых студенты изучают особенности использования различных принципов и методов обработки информации. Следует отметить, что программирование является одной из дисциплин, для успешного овладения которой необходимо не только применение приобретенных знаний и умений, но и обладание абстрактным и логическим мышлением и исследовательскими способностями. В свою очередь обучение программированию на основе интеграции парадигм программирования способствует развитию таких способностей.

Заключение

Обучение студентов различным принципам программирования позволяет сформировать умение моделировать и проектировать предметную область разными стилями, основанными на разных парадигмах программирования. При обучении программированию нужно использовать систему задач, показывающих различие подходов при их решении. При изучении различных принципов программирования студенты познают особенности использования различных методов и принципов структуризации и обработки информации.

Список литературы

1. Братко И. Программирование на языке ПРОЛОГ для искусственного интеллекта. – М.: Мир, 1990. – 559 с.
2. Буч Г. Объектно-ориентированный анализ и проектирование с примерами приложений на C++. –СПб.: Невский диалект, 2000. – 560 с.
3. Дарахвелидзе П. Г., Марков Е. П. Программирование в Delphi 7. – СПб.: БХВ — Петербург, 2005. – 784 с.
4. Могилев А. В., Пак Н. И., Хеннер Е. К. Информатика: Учебное пособие для студентов педвузов. – М.: ACADEMIA, 2004. – 848 с.
5. Халитова З. Р. Развитие абстрактно-логического мышления будущих учителей информатики при обучении программированию на основе интеграции различных парадигм //Филология и культура. PhilologyandCulture. Вестник ТГГПУ, Казань. – 2012. – № 1(27), –С. 273–277.

6. Шустова Е. П. Изучение нечеткого моделирования с использованием Mathematica 8 при подготовке специалистов на кафедре прикладной информатики КФУ // Международный электронный журнал «Образовательные технологии и общество (EducationalTechnology&Society)» – 2012. – Т. 15, № 4. – С. 536–549.

Рецензенты:

Шакирова Л.Р., д.п.н., профессор, зав. кафедрой теории и технологий преподавания математики и информатики, Институт математики и механики им. Н.И. Лобачевского, ФГАОУ ВПО «Казанский (Приволжский) федеральный университет», г. Казань;

Салехова Л.Л., д.п.н., профессор, зав. кафедрой математической лингвистики и информационных систем в филологии, Институтфилологии и межкультурной коммуникации, ФГАОУ ВПО «Казанский (Приволжский) федеральный университет», г. Казань.