УДК 377.01

Б.К.Шаяхметова

*Карагандинский государственный университет им. Е.А.Букетова*

# Особенности преподавания процедурно-ориентированных языков

В статье рассмотрены особенности преподавания процедурно-ориентированных языков программирования, сделан анализ основных процедурных языков, раскрыты их преимущества и недостатки. Исследованы проблемы непроцедурных и ориентированных языков. Пронализированы процедурно-ориентированные языки, дано описание языков программирования Basic, Pascal, Fortran, Algol.

*Ключевые слова:* особенности преподавания, процедурно-ориентированные языки, программирование, языки программирования, программист, алгоритм, парадигма программирования, компьютер, архитектура языка программирования.

В процессе преподавания информационных дисциплин при рассмотрении особенностей процедурно-ориентированных языков необходимо дать их описание. На этих языках писались программы, которые характеризовались линейной структурой, инструкции (команды) в них выполнялись последовательно.

Процедурные языки дают возможность программисту разбить программу обработки информации на несколько программ (процедур) более низкого уровня, которые определяют структуру программы в целом. Последовательные обращения к ним управляют выполнением программы, состоящей из процедур. Программа прекращает работу тогда, когда исчерпывается список вызовов процедур. Используя эту парадигму, программисты могли писать программы до нескольких тысяч строк длиной. Эта новая парадигма программирования была прогрессивной по сравнению с парадигмой программирования на машинном языке, так как в нее было добавлено главное средство структурирования — процедуры, на которые разбивалась первоначальная программа. Более мелкие процедуры (функции) не только проще понять, но также проще отладить. Кроме того, легко понимаемый в коротких программах язык программирования становится нечитабельным, когда дело касается больших программ. Важную роль в новой парадигме программирования сыграло появление в языках средств поддержки, позволяющих оперировать программами. Идея написания подпрограмм появилась гораздо раньше, но отсутствие операционных возможностей в первых языках существенно снижало эффективность их применения. Подпрограммы можно было сохранять и использовать в других программах. В результате были созданы огромные библиотеки расчетных и служебных подпрограмм, которые по мере надобности вызывались из разрабатываемой программы.

Типичная программа того времени состояла из основной программы, области глобальных данных и набора подпрограмм (в основном библиотечных), выполняющих обработку всех данных или их части (рис. 1).

Слабым местом такой архитектуры было то, что при увеличении количества подпрограмм возрастала вероятность искажения части глобальных данных какой-либо подпрограммой. Например, подпрограмма поиска корней уравнения на заданном интервале по методу деления отрезка пополам меняет величину интервала. Если при выходе из подпрограммы не предусмотреть восстановления первоначального интервала, то в глобальной области окажется неверное значение интервала. Чтобы сократить количество таких ошибок, было предложено в подпрограммах размещать локальные данные (рис. 2).

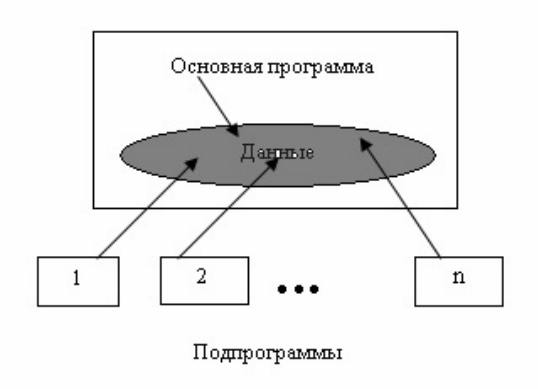


Рисунок 1. Архитектура программы с глобальной областью данных

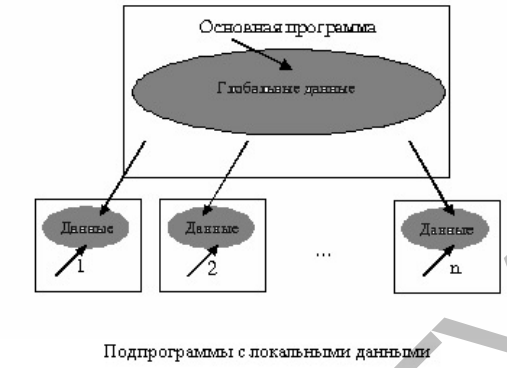


Рисунок 2. Архитектура программы, использующей подпрограммы с локальными данными

Сложность разрабатываемого программного обеспечения при использовании подпрограмм с локальными данными по-прежнему ограничивалась возможностью программиста отслеживать процессы обработки данных. На новом уровне этому способствовало появление средств поддержки подпрограмм, которые позволили осуществлять разработку программного обеспечения нескольким программистам параллельно.

Обучение разрешению этих и других вопросов — вот цель настоящего исследования. Конечно, появление их связано со сложностью программного обеспечения. Одним из путей решения поставленной проблемы обучения программированию на уровне создания программ для сложных систем является использование в учебном процессе информационных специальностей новых технологий.

Объективно все это было вызвано несовершенством технологии программирования. Прежде всего стихийно использовалась разработка «снизу-вверх» — подход, при котором вначале проектировали и реализовывали сравнительно простые подпрограммы, из которых затем пытались построить сложную программу. В отсутствие четких моделей описания подпрограмм и методов их проектирования создание каждой подпрограммы превращалось в непростую задачу. Интерфейсы подпрограмм получались сложными. При сборке программного продукта выявлялось большое количество ошибок, как правило, требовавших серьезного изменения уже разработанных подпрограмм. Это еще более осложняло ситуацию, так как в программу часто вносились новые ошибки, которые также необходимо было исправлять. В конечном итоге, процесс тестирования и отладки программ занимал более 80 % времени разработки, если вообще когда-нибудь заканчивался.

Языки программирования должны наилучшим образом соответствовать требуемым процедурам обработки данных в задании пользователя.

Из процедурно-ориентированных языков широко известны языки: Fortran, Algol, Cobol, Basic, Pascal и т.д. Спектр языков этой группы очень широк, и среди них существует определенная иерархия. Считается, что Basic предназначается для начинающих программистов, Pascal — язык студентов (язык «правильного», классического программирования). Первоначально на ЭВМ в основном решались задачи вычислительного характера, где основная часть работы машины сводилась к расчетам по формулам. Для составления программ решения задач такого типа рекомендуются языки фортран (Fortran) или алгол (Algol). Задачи экономического характера имеют уже иную специфику. Например, при расчете на ЭВМ заработной платы важно не только начислить зарплату, но и получить готовую ведомость. Для решения подобного рода проблем языки фортран и алгол не очень удобны. Особенности экономических задач были учтены при создании языка кобол (Cobol).

Вообще для каждого класса задач обычно создается один или несколько языков программирования.

Вместе с тем существуют определенные соглашения в использовании языков программирования. Так, при создании программ для собственных работ пользователь может применять любой язык, даже Basic. При разработке программного обеспечения для одного заказчика корректно использовать язык Pascal, для многих потребителей — язык Си (C). Basic популярен среди программистов. Однако для построения сложных программ он, в силу ограниченных возможностей, подходит плохо.

Общепризнано, что Pascal является наилучшим средством для обмена программами между различными типами ПЭВМ. На основе разработки языка Pascal предложен ряд новых языков, например,

язык Модула – 2 (Modula – 2), в котором особое внимание уделяется построению программы как набора независимых модулей. Также на базе языка Pascal создан язык Ада. Для разработки коммерческих программ чаще используется язык Си, который удачно сочетает в себе средства языка высокого уровня и языка Ассемблера, что позволяет разрабатывать компактные, быстродействующие, высокоэффективные программные продукты. Все описанные выше языки программирования используют так называемые пошаговые описания алгоритмов. Именно в этом и заключается источник большой трудоемкости подготовки задач к решению. К этому этапу также следует отнести и проблемно-ориентированные языки программирования (Simula, GPSS и др.). В этих языках имеется возможность описывать алгоритмы обработки информации более крупными конструкциями. Это делает программы пользователей более наглядными, так как каждая используемая конструкция соответствует вполне определенной части проблемы, исследуемой пользователем.

Следует отметить появление непроцедурных описательных языков. Несмотря на слово «непроцедурный», они также являются языками второго этапа. Примером такого языка служит ПРОЛОГ (ПРО — программирование, ЛОГ — логики), который широко применяется специалистами в области искусственного интеллекта. Конструкции языка соответствуют не математическим формулам, а определяют отношения между объектами и величинами. Язык состоит только из описаний и не имеет как таковых команд. Известно, что языков высокого уровня несколько сот, поэтому дальнейший их анализ проводить не будем.

Языки, рассмотренные выше, применяются, когда число строк программного кода до 1000, причем срок исполнения — до 1 месяца. Их может выполнить 1 программист, причем вероятность успешного завершения таких программ 100 %. Это небольшие приложения и дополнения, вносимые в готовые системы.

Рассмотренные языки процедурно-ориентированного программирования хорошо подходят для небольших приложений, состоящих из нескольких сотен строк кода, но с увеличением размера программ усложнились их управление и отладка.

Педагогические аспекты преподавания языков программирования включают следующие вопросы: класс решаемых задач; классификация языков программирования, в зависимости от класса решаемых задач; типы ЭВМ и систем программирования, на которых реализуются разработанные программы.

Конец 2-го тысячелетия характерен тем, что в мире новых информационных технологий были разработаны новые компьютерные технологии и, соответственно, новое программное обеспечение. Часть из них была реализована в различных языках программирования. В зависимости от класса решаемых задач языки программирования делятся на соответствующие группы: языки процедурного программирования (Algol, Fortran, PL/1, Кобол, Фокал, Ада, Basic, Pascal), языки системного программирования (Ассемблер, C, C++), языки логического программирования (Lisp, Multilisp, CommonLisp, Prolog, Рефал, Planner, FRL, KRL, MicroPlanner, QA4, Qlisp), языки функционального параллельного программирования (Alfl, ML, SML, APL, Fun, ParAlfl, PPL/1, Hope, Miranda) (табл.)

Таблица

**Классификация языков программирования**

| Языки процедурного программирования | Языки системного программирования | Языки логического программирования | Функциональные языки параллельного программирования |
|---|---|---|---|
| Алгол, Фор-PL/1, Кобол, Фокал, Ада, Basic, Pascal | Ассемблер, Си, C++ | Lisp, Multilisp, CommonLisp, Prolog, Рефал, Planner, FRL, KRL, OA4, Olisp, MicroPlanner | Alfl, ML, SML, APL, Fun, Hope, ParAlfl, PPL/1, Miranda |

Кроме того, существуют другие группы: языки параллельного программирования (OCCAM, PFOR, Glypnir, язык граф-схем, Actus, параллельный КОБОЛ, ОВС-ЛЯПАС, ОВС-МНЕМОКОД, ОВС-АЛГОЛ, ОВС-Фортран, язык РА (I), язык РА (GO),...); языки программирования для Интернета (HTML, Perl, Tcl/Tk, VRML); языки программирования баз данных (структурированный язык запросов SQL, PL/SQL, INFORMIK 4GL, NATURAL).

В целом алгоритмическая линия в курсе профилирующих дисциплин — это определенным образом ориентированный содержательно-методический компонент обучения, пронизывающий все обучение и получающий наибольшее развитие при изучении практических методов алгоритмизации с

использованием современных информационных технологий. В процессе обучения студенты приобретают навыки эффективного решения профессиональных задач с применением современных систем программирования и средств разработки приложений, таких как Delphi, Visual Basic, Visual C++, C++ Builder и др.

Перед педагогами, ведущими информационные дисциплины, ставятся следующие задачи:

– научить студентов классифицировать задачи и выбирать соответствующий язык программирования для представления алгоритмов решения задачи;
– изучать современные языки программирования и обучать студентов навыкам программирования в современном мире новых информационных технологий;
– дать квалифицированное образование студентам в области вычислительной техники (современные ПЭВМ, их архитектура, принципы работы);
– дать глубокие научные, теоретические и практические знания для овладения принципами работы с современными системами программирования;
– в связи с появлением новых информационных технологий изучить проблемы обучения студентов в области вычислительной техники и сформировать у студентов научное мировоззрение.

Все перечисленные выше проблемы требуют от педагогов высокого профессионализма, квалификации, научного подхода к решению проблем образования, постоянного повышения квалификации, обмена опытом работы с коллегами, просмотра периодической печати, овладения навыками работы с современными программными продуктами и компьютерными технологиями [2]. Чтобы перейти к общим вопросам процесса разработки программ, отметим некоторые недостатки процедурно-ориентированных языков: они не отличаются гибкостью и не очень откликаются на неожиданные события; взаимодействие с пользователем ограничено, а количество различных выполняемых ветвей заранее фиксировано; контроль остается за программой, а не пользователем; трудности сопровождения приложений; сложность изменения функциональных возможностей системы без внесения существенных изменений в ее структуру; низкая эффективность.

Теперь рассмотрим некоторые общие вопросы алгоритмизации.

Процесс разработки программы и вывод полученных результатов включает следующие этапы:

– разработка алгоритма решения задачи;
– разработка программы на конкретном языке программирования (например, Object Pascal);
– реализация программы в соответствующей среде программирования (например, Delphi);
– тестирование и отладка программы.

Разработка алгоритма решения задачи необходима для того, чтобы пользователь ПЭВМ мог задать формальное описание способа решения задачи в виде конечной (по времени) последовательности действий. Правильно разработанный алгоритм решения задачи устранит ошибки, которые в дальнейшем могут возникнуть в процессе разработки программы. Например, если разработчик программного обеспечения решает задачу по обработке сложных массивов данных (сложные циклы), а в алгоритме решения задачи описаны простые циклы, тогда программа, составленная для этой задачи, будет некорректной.

Для представления алгоритма в виде, понятном компьютеру, служат языки программирования. То есть алгоритм действий решения задачи записывается на одном из языков программирования. Языки программирования отличаются от естественных языков ограниченным числом «слов». Для того чтобы пользователь мог грамотно разработать программу, ему необходимо знать синтаксис и семантику языка программирования. Синтаксис языка программирования — строгие правила записи команд (операторов), семантика — смысл каждой команды и конструкций языка.

Без понимания алгоритмов и структур данных современному специалисту информационного профиля невозможно создать сколько-нибудь серьезный программный продукт. В этой связи включение в учебный процесс профессиональной подготовки студентов информационных специальностей разработанного спецкурса «Алгоритмы и структуры данных» является вполне обоснованным. Актуальность разработки и внедрения данного спецкурса объясняется тем, что структуры данных и алгоритмы служат теми материалами, из которых строятся программы. Более того, сам компьютер состоит из структур данных и алгоритмов, при этом его работа с любыми данными выполняется только в соответствии с неизменным набором алгоритмов, которые определяются системой команд центрального процессора. Следует отметить, что классической стала формула разработки программ, предложенная Н.Виртом: «Алгоритмы + структуры данных = программы» [3].

Список литературы

1 *Жангисина Г.Д.* Обучение языкам программирования в высшей школе. — Алматы: НИЦ «Ғылым», 2003. — 364 с.

2 *Егоров В.В.* Организационно-педагогические основы подготовки инженера-педагога для профессионально-трудового обучения учащихся: Автореф. дис. … д-ра пед. наук. — Алматы, 1995. — 43 с.

3 *Вирт Н.* Алгоритмы + структуры данных = программы. — М.: Мир, 1985. — 430 с.

Б.К.Шаяхметова

## Процедураға бағытталған тілдерді оқытудағы ерекшеліктер

Мақалада процедураларға бағытталған тілдерді оқытудағы ерекшеліктер зерттелініп, негізгі процедуралық тілдерге талдау жасалынуымен бірге, бұл тілдердің құндылығы мен кемшіліктері айқындалады. Сонымен қатар проблемаға бағытталған және процедуралық емес тілдер де қарастырылған. Процедуралық бағытталған тілдерді талдауда келесілер көрсетілген: Basic бастауыш бағдарлама құруға арналады; Pascal — («дұрыс» тіл, яғни студенттер тілі, классикалық бағдарлама құру үшін қажет); негізгі ДК-дың бастапқы қарастырылған есептер шешімін тауып, осы негізгі машина жұмысының уақытын формуламен қарастыру тиіс. Fortran немесе Algol тілдері бағдарламасын құру есептің шешімін табу үшін ұсынылады. Экономикалық есептің ерекшелігі, Cobol тілін құру негізінде көрінеді. Осы айтылғанның бәрі нақты мысалдарда қарастырылады.

This article studies the features of teaching of procedures — quided languages are invesligated, the analyses of the basic procedural languages was made, the advantages and lacks of these languages are marked. Besides the problem — oriented and not procedural here are considered. Analyses of procedures — quided languages show that Basic intends for beginning programmers, Pascal — is language of students («right» language the classic programming); at the firsty on PC (Personal Computer) is basically decided the tasks of computing character, where originally part of time to work by machine was redused to accounts under the formula. For drowing up of the program to decision of tasks such type were recomendated languages: Fortran or Algol. Features economic tasks were taken into account at creations of language Cobol. All of this moment are considered at this job.

УДК 800

Б.А.Абилова

*Казахская головная архитектурно-строительная академия, Алматы*

## Обучение аудированию с использованием информационно-коммуникационных технологий

В статье рассмотрены процессы обучения аудированию с использованием информационно-коммуникационных технологий при подготовке специалистов-инженеров. Отмечено, что обучение аудированию с применением информационно-коммуникационных технологий может быть значительно эффективным, намного облегчить труд учителя и повысить интерес обучаемых и мотивацию. Раскрыта сущность понятия «аудирование». Описана методика обучения языкам с применением информационно-коммуникационных технологий.

*Ключевые слова:* процессы обучения аудированию, информационно-коммуникационные технологии, подготовка специалистов-инженеров, аудирование, формирование коммуникативных умений, аутентичные материалы, технические средства обучения, методика обучения языкам, интерактивный урок, учебные компьютерные программы.

Создание искусственной иноязычной среды в процессе обучения неродному языку является одним из важных проблемных вопросов современной методики. При подготовке специалистов-инженеров обучение аудированию с применением информационно-коммуникационных технологий (ИКТ) может быть значительно эффективным, намного облегчить труд учителя и повысить интерес обучаемых и мотивацию. Упражнения и тестовые задания по аудированию изначально сопряжены с