электронный научно-технический журнал
ИНЖЕНЕРНЫЙ ВЕСТНИК
Издатель ФГБОУ ВПО "МГТУ им. Н.Э. Баумана". Эл No. ФС77-51036. ISSN 2307-0595

Алгоритмические и психологические аспекты при изучении дисциплины «Информатика»

 # 05, май 2016

доцент, к.т.н. Ваулин А. С.[1,*], Семин В. М.

УДК: 004.85.159

[1]Россия, МГТУ им. Н.Э. Баумана
*vaulin.1935@mail.ru

Введение

Изучение дисциплины «Информатика» студентами первого курса формирует восприятие постановки задачи в предметной области, развитие способности к формализации задачи и выбору необходимых структур данных для ее решения. В итоге это отражается на развитии таких психологических процессов как ощущение, восприятие, память, мышление, воображение, что в итоге улучшает алгоритмическое мышление и способствует дальнейшему использованию вычислительной техники в общеинженерных и специальных дисциплинах.

Для получения хороших результатов от студентов требуется мотивация, мобилизация внимания и целеустремленность в работе. Именно человеческие качества в значительной степени играет решающую роль в разработке программ, так как программы наследуют черты характера, присущие разработчикам (аккуратность, системность, непостоянство, надежность и т. д.).

В связи с этим в преподавании дисциплины возникают большие трудности, обусловленные не только недостаточным уровнем общей подготовки студентов, но и отсутствием у них практического опыта в восприятии постановки задач.

У многих студентов в начальной стадии обучения при разработке программ возникает психологический барьер. Поэтому в преподавании дисциплины необходимо каждый раз в зависимости от подготовки студентов определить круг вопросов, представляющихся наиболее сложными в усвоении, с целью устранения этого барьера.

Содержание и подготовка студентов в области информатики

Подготовка студентов по информатике осуществляется тремя видам занятий: лекции (17 час.) в первом семестре, упражнения в двух семестрах (32 час.+16 час.) и лабораторные занятия в двух семестрах (32 час.+32 час.). На лекциях рассматриваются общие вопросы, связанные с системами счисления, логическими основами ЭВМ, техническими

средствами и программным обеспечением ЭВМ, алгоритмизацией и отладкой программ, вычислительными и глобальными сетями, базами данных. Курс лекций не связан с практической подготовкой студентов.

Практическая подготовка студентов, связанная с разработкой программ, осуществляется на упражнениях (семинарах) в аудиториях и на лабораторных занятиях в дисплейных классах. Изучение материала проводится по принципу от простого к сложному. Студенты начинают осваивать простейшие конструкции языка: символы, числа, переменные, стандартные функции, выражения, а затем операторы языка. На основе полученных знаний студенты разрабатывают программы линейной, разветвляющейся, циклической и вложенной циклической структуры. Поскольку программа представляет собой последовательность операторов, соответствующих алгоритму, то перед студентом возникает проблема определить последовательность действий и расположить известные ему операторы в нужной последовательности.

Вначале объектами обработки являются данные действительного, целого типов и массивы этих типов, а затем не числовые и структурированные типы данных: символьный тип, множество, строки, записи, файлы. Подключаются новые программные структуры: подпрограммы и модули. Одновременно с разработкой программ студенты осваивают типовые приемы программирования (вычисление сумм и произведений, нахождение наибольшего и наименьшего значения в массиве, вычисление среднего арифметического и среднего геометрического значения и т. д.) и выполняют индивидуальные задания по различным разделам курса [1-8].

Первые практические занятия требуют знаний для записи простейших конструкций языка и организации программ линейной и разветвляющей структуры

В дальнейших занятиях при реализации поставленной задачи возникают проблемные ситуации. Например, у студента с одной стороны имеется достаточно знаний для понимания постановки задачи, а с другой стороны он понимает, что известными ему способами и средствами данную задачу решить нельзя. Возникает проблемная ситуация, характеризующаяся противоречием между наличием известных приемов и методов решения и поиском нового пути решения для получения результата.

Использование среды Turbo Delphi для обучения программированию позволяет на начальной стадии освоить работу в консольном режиме (принципы структурного программирования) [1], а затем перейти к режиму визуального программирования (объектно-ориентированное программирование) [9-11].

В консольном режиме не требуются дополнительные знания кроме синтаксиса языка программирования и можно сосредоточить внимание на разработке программ с простым и наглядным интерфейсом. На подготовку таких программ в целом требуется меньше времени, поэтому консольный режим удобно использовать для быстрой проверки и отладки отдельных задач.

В дальнейшем студенты осваивают работу в режиме визуального программирования с использованием компонентов языка, визуального интерфейса и компьютерной графики.

Разрабатываемая программа должна отвечать основным свойствам алгоритмов и программ, которые определяют:

- массовость – применимость программы к широкому диапазону допустимых входных данных;
- результативность – программа должна завершаться за конечное число шагов с выдачей результатов или сообщений о невозможности их получения;
- дискретность – программа должна состоять из отдельных законченных действий;
- определенность – каждое действие в программе должно быть строго определено и, не содержать неоднозначных действий.

Алгоритмические аспекты

Разработка программы является достаточно трудоемким процессом, включающим основные этапы: постановку задачи, разработку алгоритма, отладку, тестирование и получение результата. На каждом этапе разработки программы могут возникать ошибки.

К наиболее характерным ошибкам относятся синтаксические, семантические, логические и алгоритмические ошибки.

Синтаксические, связанные с нарушением синтаксиса языка, выявляются на этапе трансляции. Эти ошибки не являются сложными для студентов, но требуют знаний оформления программ, правил записи операторов и внимания и аккуратности в записи программ.

Семантические ошибки отражают смысловую сторону обработки данных программой. Это ошибки, связанные с нарушением допустимых ограничений на работу компьютера и ошибки, вызванные невозможностью выполнения арифметических операций. Оба вида ошибок приводят к прерыванию выполнения программы до ее завершения.

Логические ошибки не могут быть обнаружены транслятором, так как они связаны с пропуском операторов или фрагментов операторов, с неправильным выбором ветви при организации разветвления в программе, неправильной организацией циклических структур и другими ошибками при организации ввода-вывода данных и обработке массивов.

Алгоритмические ошибки, допускаемые при разработке алгоритма, приводят либо к решению другой задачи, либо к неточному или неэффективному решению задачи, либо совсем не обеспечивают решение.

Алгоритмические ошибки, возникающие при разработке программ, возникают от не полного понимания постановки задачи, недостаточной подготовленности и могут быть продемонстрированы фрагментами простых задач.

Одним из распространенных типовых приемов вычисления является прием накопления суммы. Фрагмент накопления суммы элементов одномерного массива имеет вид:

```
s: = 0;
for i:=1 to n do
s:= s + a[i];
```

При изменении постановки задачи на вычисление среднего арифметического значения элементов массива не возникает проблем, так как к фрагменту добавляется оператор присваивания для вычисления среднего значения sr:= s/n;

Изменение постановки задачи на вычисление среднего арифметического значения положительных элементов массива возникает проблемная ситуация, которая требует значительной переработки фрагмента программы. Во-первых – не все элементы массива могут быть положительными, а во-вторых – они могут отсутствовать. Без учета этих особенностей в постановке задачи возникает ошибка, связанная с делением на ноль. Разобравшись в сущности постановки задачи, организуя правильную последовательность операторов и учитывая свойство результативности программ, фрагмент программы принимает вид:

```
s: = 0;  k:=0;
for i:=1 to n do
if f[i] > 0 then begin s:= s + a[i]; k:= k +1 end;
if k=0 then writeln ('Положительных элементов нет') ,
        else begin sr:= s/k; writeln ('sr=', sr:6:2) end;
```

Задача усложняется при вычислении сумм элементов строк или столбцов матрицы. Студент должен решить, как организовать вложенный цикл, какой параметр цикла должен изменяться во внешнем, а какой во внутреннем цикле. Неправильный выбор параметров приводит к изменению постановки задачи. Это можно продемонстрировать следующими фрагментами программ.

При вычислении сумм элементов строк студенты часто записывают таким фрагментом:

```
s: = 0;
for i:=1 to n do
begin
for j:=1 to m do
s:= s + a[i,j];
writeln ('s=', s:6:2);
end;
```

Такое решение выдает правильный ответ только для первой строки матрицы, а затем, сумма элементов текущей строки приплюсовывается к сумме предыдущей строки. Таким образом, получен алгоритм, вычисляющий сумму элементов матрицы с выдачей промежуточных результатов.

Для такой постановки задачи правильным является фрагмент программы, в котором оператор задания начального значения счетчика суммы s: = 0 включается в тело внешнего цикла, чтобы производилось суммирование элементов каждой строки матрицы. Фрагмент программы принимает следующий вид:

```
for i:=1 to n do
begin
  s: = 0;
for j:=1 to m do
s:= s + a[i,j];
writeln ('s=', s:6:2);
end;
```

Невнимание или недостаточное понимание структуры двумерного массива может привести к вычислению не сумм элементов строк, а к вычислению сумм элементов столбцов. Следовательно, при изучении типовых приемов программирования следует уже при постановке задачи обращать внимание на возможные изменения алгоритма в случае нечеткого представления задания.

Психологические аспекты

При отборе к подготовке профессиональных программистов используют тесты, выявляющие психологические качества, определяющие склонность человека к данной профессии. Среди основных таких качеств рассматриваются способность к абстрагированию при работе со структурами данных и процессами обработки, гибкость мышления, включающее аналитическое и комплексное мышление, критичность и самокритичность, склонность к анализу, системной работе, готовность учиться и переучиваться. Психологические аспекты включают не только личностные качества, определяемые изучаемой дисциплиной, но и такие как:

- адаптация к условиям и процессу обучения в вузе;
- мотивация;
- интерес к выбранной специальности;
- умение общаться с преподавателем и др. [13,14].

Обучение студентов основам программированию отличается от подготовки профессионалов тем, что дисциплину изучают все не зависимо от наличия требуемых психологических качеств. Поэтому важным является на каждом этапе изучения определять базовые знания и понятия, непонимание которых будет психологическим барьером при дальнейшем изучении курса. Усвоение этих понятий будет способствовать развитию продуктивного мышления, и использоваться при разработке более сложных алгоритмов.

Для студентов, впервые изучающих программирование, важным является:

- понимание для чего создается программа (массовость ее применения и результативность решения);

- хранение программы и данных в памяти машины;
- взаимодействие программы и данных при выполнении программы;
- Желание, умение и возможность сделать программу.

От студента, например, сразу требуется понимание различия в применении программы при вычислениях от обычных вычислений, с которыми он имел дело раньше. Это понимание аналогично пониманию отличия арифметики, где действия выполняются над числами, от алгебры, где действия выполняются над буквами (переменными), обозначающими эти числа.

Казалось бы, непонимание выполнения простого оператора присваивания s:=s+p и его отождествление с записью S=S+P может не только снизить восприятие и дальнейшее осмысление изучаемого материала, но и способствовать появлению и развитию отрицательных качеств, таких как снижение внимания к излагаемому материалу, неуверенность в себе и др. Для исключения таких ситуаций необходимо находить методические приемы и закономерности, улучшающие восприятие и осмысливание излагаемого материала.

Так, при изучении основ языка и усвоения правил работы операторов можно использовать пошаговое выполнение программы на бумаге или использовать отладчик [12].

Такие же проблемы и психологические последствия возникают при разработке алгоритмов задач. Студент, как правило, понимает задачу, но выразить ее даже словами часто не может, не говоря уже о разбиении задачи на логически связанную последовательность действий.

Даже после изучения операторов языка типовая задача определения максимального элемента массива сводится к «сравнению чисел между собой». Это объяснимо, т.к. студент опирается на свой предшествующий опыт, где ему не приходилось разбивать обычные действия на составляющие и оценивать эффективность этих действий. Например, почему он покупает сразу белый хлеб и черный, а не по отдельности ходит за ними в магазин.

При определении максимального элемента можно еще раз проиллюстрировать важность понимания оператора присваивания. Правильный в этом случае текст фрагмента программы:

```
amax :=a[1];
for i:=2 to n do
if a[i]>amax then amax:=a[i];
```

Однако из-за механического восприятия оператора присваивания фрагмент программы часто воспроизводится студентами в виде:

```
a[1]:=amax;
for i:=2 to n do
if a[i]>amax then a[i]:=amax;
```

Такое решение приводит не только к получению неверного результата, но и изменению данных в массиве A.

Следовательно, при рассмотрении данного приема программирования оператору присваивания, задающего начальное значение, нужно еще раз уделить внимание как базовому понятию, чтобы исключить возможные ошибки.

Чтобы студент начал разрабатывать алгоритмы и писать программы необходимо выделять и концентрировать внимание на изучении и освоении основных приемов программирования при обработке числовых данных. К ним можно отнести алгоритмы:

- вычисление сумм произведений;
- нахождение максимумов, минимумов;
- формирование новых массивов;
- перестановка элементов;
- упорядочивание массивов

Усвоение материала этих разделов позволит раскрыть студенту закономерности и приемы восприятия и осмысливания дальнейшего изучаемого материала с использованием других алгоритмов и структур данных.

Следует отметить, что полученные знания, формируемые в данном предмете, имеют применение во многих других учебных предметах.

Заключение

При изучении дисциплины у студентов формируются и развиваются психологические качества, связанные со способностью к абстрагированию, пониманию отношений между объектами обработки, гибкостью мышления, критичностью к проделанной работе, склонностью к анализу.

Курсом формируется способность студента определять структуру программы, проводить детализацию решаемой задачи. В процессе практических занятий вырабатывается умение выделять однотипные участки обработки данных и оформлять их в подпрограммы. Это приводит к сокращению объема программы, улучшению ее читаемости, сокращению времени разработки.

Закрепляется умение применять типовые приемы программирования и готовые алгоритмы, успешно комбинировать ими, находить простые и рациональные решения, вырабатываются важные качества комплексного мышления, позволяющего правильно определять этапы разработки алгоритмов.

В процессе работы формируется качество, связанное с умением анализировать собственные ошибки в написании программ, систематизировать их и на этой основе вырабатывать свой стиль программирования.

Важным качеством, позволяющим успешно расширять свои познания, является способность подавлять самолюбие при неудачах. Если студент не справляется с этим, то может развиваться психологическое состояние (фрустрация), приводящее к потере веры в свои силы, тревожному чувству безысходности. Как правило, у таких студентов

проявляется агрессивное поведение в качестве самозащиты. С такими явлениями приходится сталкиваться и их приходится учитывать.

Для успешного изучения дисциплины и получения практических навыков в программировании в значительной степени способствуют личностные особенности студентов. В первую очередь к ним относятся эмоциональная устойчивость, пунктуальность, аккуратность, хорошая работоспособность. Большую роль играет фактор увлеченности работой, фактор радости творчества. Неоднократно приходилось наблюдать за поведением студентов при решении сложной задачи, когда он ощущает радость от проделанной работы при получении требуемых результатов.

Коммуникативные качества у студентов оказывают психологическую помощь при общении с преподавателем и со своими коллегами в группе, создавая творческую атмосферу в группе. Такие студенты адекватно относятся к просьбе преподавателя сделать программу более рациональной. После диалога с преподавателем, выяснив суть изменений, студент сразу настраивается на работу и доводит программу до рационального решения. В группах, в которых образовался коллектив с присущими качествами, быстрее устанавливается рабочий контакт преподавателя с большинством студентов группы и успеваемость в таких группах значительно выше.

Необходимо отметить, что алгоритмические и психологические аспекты, возникающие при изучении дисциплины «Информатика» взаимосвязаны и оказывают влияние друг на друга. В процессе изучения дисциплины необходимо уделять внимание не только вопросам, связанным с разработкой рациональных программ, но и уделять внимание формированию психологических качеств у студентов. Формирование у студентов навыков рационального подхода к разработке алгоритма автоматически вырабатывает многие психологические качества.

Список литературы

[1]. Алексеев Ю.А., Ваулин А.С., Куров А.В. Практикум по программированию. Обработка числовых данных. М.: Изд-во МГТУ им. Баумана, 2008. 285 с.

[2]. Культин Н.Б. Основы программирования в Turbo Delphi. СПб.: БХВ-Петербург, 2012 . 384 с. + CD-ROM.

[3]. Мельников С.П. Delphi и Turbo Pascal на занимательных примерах. СПб.: БХВ - Петербург, 2012. 448 с.

[4]. Флёнов М.Е. Библия Delphi. 3-е издание. СПб.: БХВ - Петербург, 2011. 673 с.

[5]. Дональд Эрвин Кнут. Искусство программирования. Том 1. Основные алгоритмы: пер. с англ. М: Вильямс, 2015. 720 с. [Knuth D.E. The Art of Computer Programming, Volumes 1-4A Boxed Set (Box Set). Addison-Wesley Professional, 2011. 3168 p.]

[6]. Ваулин А.С., Криницына Л.Ф., Мартынюк Н.Н. Практикум по программированию в среде DELPHI: Методические указания к выполнению заданий по курсу «Информатика». М.: Изд-во МГТУ им. Баумана, 2007. 41 с.

[7]. Ваулин А.С., Криницына Л.Ф., Мартынюк Н.Н. Программирование с использованием подпрограмм в среде DELPHI: электронное учебное издание // Депозитарий электронных изданий ФГУП «Информрегистр». М: МГТУ им. Н.Э. Баумана, 2011. Режим доступа: http://www.db.inforeg.ru/Inet/GetEzineByID/287698 (дата обращения: 2.03.2016)

[8]. Мельников С.П. Delphi и Turbo Pascal на занимательных примерах. СПб.: БХВ - Петербург, 2012. 448 с.

[9]. Борисов С.В., Комалов С.С., Серебрякова И.Л. Введение в среду визуального программирования Delphi. Ч.1: Метод. указания / Под. ред. Б.Г. Трусова. М.: МГТУ им. Баумана, 2005. 79 с.

[10]. Борисов С.В., Комалов С.С., Серебрякова И.Л., Степанова И.И., Пащенко О.Б. Введение в среду визуального программирования Delphi. Ч. 2: Метод. указания / Под. ред. Б.Г. Трусова. М.: МГТУ им. Баумана, 2006. 96 с.

[11]. Борисов С.В., Комалов С.С., Пащенко О.Б., Серебрякова И.Л. Введение в среду визуального программирования Delphi. Ч. 3: Метод. указания / Под. ред. Б.Г. Трусова. М.: МГТУ им. Баумана, 2007. 79 с.

[12]. Семин В.М., Русакова З.Н. Важные этапы изучения основ программирования // Инженерный вестник. Электронный журнал МГТУ им. Н.Э. Баумана, 2014. №12. Режим доступа: http://engbul.bmstu.ru/doc/742208.html (дата обращения: 4.03.2016)

[13]. Рожников В.А. Психология программирования: цели, проблемы, перспективы. // Общество, социология, психология, педагогика. Научный журнал. Краснодар: Издательский дом «ХОРС». 2014. выпуск №3.

[14]. Орел Е.А. Диагностика особенностей мыслительной деятельности специалистов в области информационных технологий (программистов): автореферат дис. … канд. психолог. наук. М: МГУ им. М.В. Ломоносова, 2007. 20 с.