А. СПИВАКОВСКИЙ
г. Херсон

Педагогические программные средства: объектно-ориентированный подход

На нынешнем этапе развития программного и методического обеспечения компьютерного обучения особо актуальным является поиск новых подходов — как на уровне идей, так и на уровне написания программ. Ведь большинство существующих и создающихся педагогических программных средств (ППС) реализуют в основе своей идеи программированного обучения.

Здесь будет изложен новый подход (обозначенный в заголовке) к созданию обучающих программ и описанию адекватной программной реализации в конкретной предметной среде.

На наш взгляд, основу объектно-ориентированного подхода к созданию ППС должна составить следующая система требований.

1. Пользователь должен работать с реальными объектами предметной области (уравнениями, многочленами, матрицами, системами линейных уравнений и т. п.), а не с текстами и вопросами, как это реализуется в программированном обучении.

2. Пользователь должен работать в реальной операционной среде, однозначно определяемой предметной областью (например, для матриц: сложи две строки, домножь строку на число, переставь две строки местами и т. п.).

3. Интерфейс пользователя практически не должен отличаться от традиционного (лист бумаги заменяется экраном, последовательность листов бумаги — последовательностью кадров на экране, которые можно передвигать как вперед, так и назад, карандаш — курсором, резинка — забоем). При этом пользователь в рамках дидактических целей может использовать вычислительные ресурсы, предоставляемые ППС.

4. ППС должно предоставлять пользователю свободу, ограниченную только рамками предметной области (например, с матрицей можно делать любые элементарные преобразования в любой последовательности, главное — найти ее ранг), т. е. пользователь не должен находиться под прессом алгоритма решения, определенного на стадии написания ППС. Причем пользователь должен иметь возможность «передвигаться» по своим действиям, вставляя между ними новые. Более того, пользователь должен иметь возможность вообще отказаться от операционной среды ППС и строить произвольно новый объект, а дело программы — оценить правильность его действий.

5. Пользователь должен всегда иметь выход из тупиковых ситуаций, для чего операционную среду необходимо дополнить, например, приказом «Очередной шаг решения сделай, машина, сама».

6. История работы пользователя должна представляться в виде последовательности его действий с соответствующей диагностикой, что резко облегчит работу проверяю-

щего, а также позволит пользователю иметь оперативную информацию о правильности своих действий.

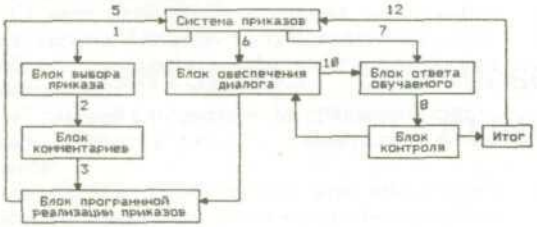
7. ППС должно иметь два режима:

а) «пассивное наблюдение» за работой пользователя с выдачей информации о правильности его действий;

б) «параллельная работа» (обучаемый сделал, затем программа показала, как бы сделала она).

Очевидно, ППС, удовлетворяющее вышеперечисленным свойствам, интегрирует в себе функции демонстратора, тренажера и контролера в естественной для обучаемого форме, позволяя при этом преподавателю проследить не только диагностику, но и логику действий обучаемого.

Диаграмма схематично отражает структуру программной реализации объектно-ориентированного подхода.



Опишем вкратце технологию производства ППС, в основе которой лежит рассматриваемый подход, на примере созданного программного средства «Системы линейных уравнений» для ПЭВМ «Ямаха», написанного на Паскале.

1. Определяем систему объектов, с которыми будет работать обучаемый (в данном случае — системы линейных уравнений над полем рациональных чисел, без ограничения на совместность и количество решений).

2. Определяем операционную среду пользователя (см. табл.).

3. Определяем интерфейс пользователя. В данном случае он представлен четырьмя окнами.

Первое окно, расположенное в строках 0—2, столбцах 0—80 экрана, предназначено для организации управления учебным процессом. Именно здесь обучаемый имеет возможность вызвать исполнение приказов 1—17.

Второе окно, расположенное в строках 3—12, столбцах 0—40, организует работу с последовательностью действий обучаемого.

Третье окно, расположенное в строках 3—12, столбцах 40—80, позволяет видеть пользователю параллельную работу ППС над матрицей.

№ п/п	Приказы	Действие приказа
1	+	Производит сложение строк, указанных пользователем
2	—	Производит вычитание строк, указанных пользователем
3	*	Производит домножение строки системы на множитель
4	/	Производит сокращение строки системы на множитель
5	p	Производит перестановку двух строк системы
6	r	Производит перестановку двух столбцов системы
7	b	После приведения системы к диагональному виду организует определение ранга основной и расширенной матриц системы, совместности системы, числа решений, частного и фундаментальных решений системы линейных уравнений
8	o	Организует работу с коэффициентом основной матрицы
9	c/o	Возвращает систему в форме, предшествующей приказу «o»
10	del	Организует удаление нулевой строки системы
11	x	Предоставляет пользователю перейти к произвольной системе. ППС оценит эквивалентность введенной и исходной систем. Данный приказ предоставляет пользователю полную свободу в рамках предметной области
12	←	Позволяет пользователю возвратиться к любой системе, полученной им ранее, и, если надо, продолжить работу с этого места (откатка назад)
13	→	То же, что и откатка назад, но вперед
14	⏎	В случае затруднения очередное преобразование выполнит компьютер с кратким пояснением
15	cntr/f	Вывод на экран полного решения системы машиной или распечатка на принтере всех выполненных действий
16	cntr/q	Настоящее ППС предоставляет возможность моделировать две схемы взаимодействия обучаемого и педагога: а) обучаемый выполнил действие — педагог прокомментировал его правильность; б) обучаемый выполнил действие — педагог прокомментировал его и предложил свой шаг в решении задачи Настоящий приказ позволяет переключать эти режимы.
17	h	Вызов окна помощи, в котором непосредственно можно выбрать один из приказов 1—16

Четвертое окно организует работу по приказу h.

Интерфейс может в каждом конкретном случае иметь свою структуру, однако должен настраиваться (в разумных пределах) самим обучаемым.

4. *Определяем алгоритм решения задачи,* которая будет задаваться обучаемому, таким образом, чтобы соответствующая процедура «сумела оценить» состояние изменяемого объекта и найти из системы приказов нужный (см. приказ 14).

5. Создаем процедуры, реализующие приказы 1—13 и 15—17.

6. На основе диаграммы строим управляющую программу, отражающую структуру дерева обучения.

В заключение необходимо сказать, что технология конкретного наполнения пунктов 4—6 является предметом отдельного исследования, имеющего значительный объем, и потому в рамках настоящей статьи не описывается.