

МГУ 27 июня 2005

Информатика-21. ИТ-образование и национальные



## Произведение успешного юного “профессионала”

```
PROCEDURE generate*(obj:setGuider);
  VAR exit,reset:BOOLEAN;i:INTEGER;
  BEGIN
    exit:=FALSE;reset:=FALSE;i:=LEN(obj.content)-1;
    WHILE exit=FALSE
      DO
        IF i>=0 THEN
          obj.content[i].checkFull();
          IF obj.content[i].full=FALSE THEN
            obj.content[i].add();
            exit:=TRUE;
          ELSE
            reset:=TRUE;
            obj.content[i].reset();
            DEC(i);
          END;
          IF (reset=TRUE) & (exit=TRUE) THEN
            obj.setMax(i);
          END;
        ELSE
          exit:=TRUE;
          obj.end:=TRUE;
        END;
      END;
    END;
  END generate;
```

**три IF,**  
**две** логич.  
переменных,  
**запутанная** логика  
— потребовала  
“отладки”

МГУ 27 июня 2005

Информатика-21. ИТ-образование и национальные



## Эквивалентная программа

получена формальным преобразованием за несколько минут;  
**один IF, ни одной** логич. переменной, **стандартная** логика

```
PROCEDURE Next* ( sg: SetGuider );  
  VAR i: INTEGER; c: POINTER TO ARRAY OF Set;  
BEGIN  
  c := sg.content;  
  i := LEN( c ) - 1;  
  WHILE ( i >= 0 ) & c[ i ].IsFull() DO  
    c[ i ].Reset(); DEC( i )  
  END;  
  
  IF i >= 0 THEN ( * поиск успешен *)  
    c[ i ].Inc();  
    IF i < LEN( c ) - 1 THEN sg.SetMax( i ) END  
  ELSE  
    sg.end := TRUE  
  END  
END Next;
```

элементарная схема  
линейного поиска

стандартная обработка  
окончания цикла

Программист с правильной **базовой** подготовкой должен **сразу** писать такой цикл, **не тратя ни секунды** на “отладку” логики...

**Примерно такого и нужно добиться.**

## Приветствие Никлауса Вирта

Мне приятно написать несколько слов по поводу введения моего языка Оберон в курсы программирования в Московском государственном университете.

В 1970 г. я опубликовал "Сообщение о языке Паскаль" и представил первый компилятор. Паскаль был основан на концепции структурного программирования с использованием структурированного языка, а также строгой статической типизации. В дальнейшем он был широко принят для обучения программирования по всему миру. Ряд реализаций Паскаля сделали его популярным средством программирования. В 1979 г. за ним последовал язык Модула-2, спроектированный в том же духе, но с дополнительными средствами для создания больших программных систем. В частности, это концепция модулей с четко определенными интерфейсами и возможность раздельной компиляции с полной проверкой типов. Последний член семейства, [Оберон](#), был спроектирован и реализован в 1988. Он включает в себя средства, необходимые для объектно-ориентированного программирования, сохраняя стиль Паскаля, и является результатом моего стремления к простоте без потери выразительности. В этом должна состоять сущность языка, равно пригодного как для учебной аудитории, так и для профессиональной деятельности.

Оберон ... является результатом стремления к простоте без потери выразительности. В этом должна состоять сущность языка, равно пригодного как для учебной аудитории, так и для профессиональной деятельности.

Счастлив добавить, что Паскаль был весьма популярен в России, как и Модула-2, поддерживаемая активной группой энтузиастов Модулы в Москве. Это и не удивительно, ибо структурированный, точно и кратко определенный язык, конечно, привлекателен для людей, получающих образование в контексте столь сильной математической традиции как российская. (И, может быть, они оценят язык программирования, грамматика которого настолько проще, чем у русского языка :-) Математически подготовленный ум приветствует язык, управляемый немногими, но общими правилами. В результате компиляторы и другие программы поддержки тоже просты и компактны. Уже нет нужды в целых горах программного обеспечения.

Целью компетентного инженера должно быть создание надежных и эффективных конструкций, в данном случае программного обеспечения. Чем сложнее требования, тем лучше нужно понимать задачу и инструменты. Чем лаконичней и компактней программы, тем меньше шансов, что они содержат ошибки. Программы должны быть интеллектуально постижимы, и только их формулировка на структурированном языке делает это возможным. Точные рассуждения следует предпочесть методу проб и ошибок.

Программы должны быть интеллектуально постижимы ... Точные рассуждения следует предпочесть методу проб и ошибок.

Паскаль, Модула и Оберон были спроектированы, когда я преподавал в Швейцарском федеральном техническом институте (ETH) в Цюрихе, с намерением обеспечить надлежащие инструменты для надлежащего обучения и надлежащего профессионального программирования. Мои бывшие студенты основали компанию с целью переноса Системы Оберон на коммерческие вычислительные платформы. В результате язык получил имя Компонентный Паскаль, чтобы подчеркнуть родство по прямой линии с широко известным Паскалем.

Искренне надеюсь, что Оберон поможет в обучении высококлассных программистов и ученых в России, и что его оценят в стране, где всегда придавали большое значение основательному образованию.

Никлаус Вирт  
21 сентября 2001, Цюрих

**Грушин № 673 28-02-2006**

... Я пришёл на спецкурс, читаемый уважаемым Ф.В.Ткачёвым на физфаке МГУ, в самом начале, это была пилотная версия, так сказать. Тогда мне было просто интересно, много нового, причём нового не потому, что недавно открыли, а потому, что стандартный курс программирования на факультете фактически ничего не давал - всё сводилось к освоению Ворда, Интернета и нескольким семинарам по C/C++ и MatLab (говорю за первый поток, на втором потоке преподавал Задков, и, по слухам, подавался предмет на порядок лучше). Всё, что мне из этого потом пригодилось - это MatLab, действительно полезное приобретение, но для успешной работы всё же недостаточно. Соответственно структура курса - от самого простого, базового (грамотное написание программного кода, пред- и пост-условия, основные типы данных, объекты+методы), последовательно, к сложному (полиморфизм, инкапсуляция, модульность) - пришлось очень кстати и даже не самым умным студентам (т.е. мне :) ) удалось овладеть хотя бы азами программирования, причём правильного программирования (на мой взгляд, об этом - чуть ниже). Сразу оговорюсь - освоить удалось отнюдь не всё, что давал нам Ф.В.Ткачёв и в этом виноват только я сам (хех, глупый был), но уже по мне можно провести нижнюю грань того, что можно вынести из читаемого курса (верхнюю грань сможет, наверное, провести сам Фёдор Васильевич, взяв наиболее успешных студентов). Во что всё это вылилось?

В работе сейчас использую Fortran - традиция-с ? + объективная необходимость работы на многопроцессорных суперЭВМ, работающих под \*nix'ами. Уже при первом знакомстве с этим языком программирования стало понятно, что "здесь вам не тут" и многое из того, к чему я привык в BlackBox'e и считал разумным, необходимым и достаточным минимумом возможностей, в Fortran'e отсутствует, несмотря на регулярные "обновления" языка. Автоматическое управление памятью, наследование (не нашёл, но может плохо искал), методы - нет, но можно и обойтись. Плюс трудно находимые и не всегда понятные причины различий результатов от сборок Debug-Release. Непонятно, также, как бороться с модулями - не слишком разумное решение имена всех процедур и глобальных переменных из всех используемых модулей сливать в одно множество и потом барахтаться с пересечениями, куда логичней подход ИмяМодуля(delimiter)ИмяПроцедуры. Но в принципе, и с этим можно смириться. Так вот, если взять множества средств и возможностей этих двух языков (извините за дилетантство) и выделить пересечение, то сухим остатком вполне можно пользоваться! Типы в Fortran'e есть, указатели похожи на BlackBox'овские, можно скрывать что угодно внутри модуля от шаловливого доступа извне, со скрипом использовать модули. Что ещё нужно? Добавляем сюда технику осторожного программирования, отказываемся сразу от всех остальных возможностей Fortran'a "and we ready to go". За последние 2 года интенсивного самостоятельного программирования приходилось не раз и не два довольно серьёзно корректировать алгоритм, изменять множество уравнений модели, корректировать исходные предположения, потом менять всё назад, опять корректировать, вычленять из этих действий знаменатель и удалять всё лишнее. И ни разу за всё это время у меня не возникало серьёзных проблем в плане программирования, были скорее проблемы с пониманием физики описываемых моделью процессов и математикой самой модели, и я их довольно успешно решал. Не в последнюю очередь потому что, за спиной стоял надёжный как гвоздь код, в котором я был уверен, были чёткие типы данных, структурировано собранное в себе основные физические сущности, которые по мере необходимости расширялись или лишались каких-либо полей. Всему этому я научился у info21 и за что давно хотел сказать ему огромное спасибо.

А потом я взглянул, как работали раньше, и был сражён наповал одним COMMON блоком и кучей goto, которые официально до сих пор даже не номинировались на исключение из стандарта языка. Кто видел старый код, тот поймёт, что я имею виду. Особенно старый код, состоящий из кусков, написанных разными людьми, и на скорую руку сшитый воедино. Нет, всё понятно - аргумент "Раньше по-другому было нельзя" многое объясняет, нельзя, так нельзя, но на дворе 21-й век и машины давно уже лишены многих ограничений своих почтенных предков. И переучить старых программистов действительно тяжело, да что говорить - порой объяснить преимущество тех же типов в совокупности с грамотным разбиением программы на процедуры над COMMON блоками тяжело. Понятие модуля тоже где-то на Камчатке восприятия (use copy-paste, stupid). Мне пока удаётся лишь частично доказать собственным примером, что это вообще можно с успехом применять, об остальном уже и мечтать перестал. Представим теперь, что кто-то будет учиться программированию на Fortran'e, причём с нуля и имея перед глазами старый, наследственный код. А учиться будут, можно не сомневаться. Я уже наблюдаю одно дарование у нас в отделе, твёрдо уверенное, что старый стиль - он верный самый. И вот ещё зарисовка:

Приходят к нам в отдел студенты из МИФИ. Физику, судя по всему, знают довольно хорошо, даже отлично. С математикой, вроде, тоже проблем нет. Спрашиваю, а как вас учили программированию? Отвечают, что мол, прочитали курс численных методов с уклоном на практическую реализацию, т.е., грубо говоря, самому программированию учили никак, но показали несколько алгоритмов решения стандартных задач, вроде решения СЛАУ или ОДУ различными методами и т.д. Кажется, я их правильно понял. Соответственно диплом-то они защитят, благо для этого в отделе имеется достаточное количество рабочих программных блоков, используя которые можно получать определённые результаты. Но ведь надо и дальше двигаться - после поступления в аспирантуру придётся писать свой код, лично проверять свои предположения, искать ошибки, перерабатывать программы, взаимодействовать с чужими программами. Не скажу ничего нового, если замечу, что основным рабочим инструментом для теоретика давно стал компьютер, и почему студентов-естественников не учат работать с этим универсальным вычислителем - мне лично не понятно (теперь придётся либо отсылать их в МГУ на прослушивание курса info21, либо самому впрягаться и пытаться объяснить им то, что в своё время худо-бедно смог усвоить).

Так вот, не могу с уверенностью предсказать, что будет, если учиться программированию на языке типа Fortran без выделения упомянутого выше минимума относительно безопасных возможностей языка и отбрасывания остального, но есть мнение, что ничего хорошего из этого не выйдет. Как мне кажется, в самом лучшем случае человек начнёт самостоятельно отбрасывать лишнее и небезопасное, но на это уйдёт время; в худшем - смирится с необходимостью сидеть сутками в Debugger'e и будет считать это вполне нормальной частью рабочего процесса.

Понимаю, что срок моей самостоятельной "деятельности" маленький (2 года - Sic!), опыта ничтожно мало, но у меня и цель другая - показать, что как минимум одна цель проекта жизненно необходима и имеет под собой реальную почву, а именно: сейчас во многих естественнонаучных Институтах программированию практически не обучают. Вообще и никак.

Резюмируя:

- 1) я всеми конечностями за проект, особенно в ВУЗах для, как минимум, физических специальностей (про школу ничего конкретного сказать не могу, но аргументация в предыдущих ~700 постах представляется вполне разумной);
- 2) Искренне рад успеху спецкурса на физфаке МГУ. ...

Успехов и удачи!  
Александр Грушин