ИЗ ИСТОРИИ И АРХИВОВ

УДК 519.6

50 ЛЕТ ФОРТРАНУ: ИСТОРИЯ, СОВРЕМЕННОЕ СОСТОЯНИЕ И БУДУЩЕЕ

В. П. Соколов
(РФЯЦ-ВНИИТФ)

Статья подготовлена по материалам доклада, сделанного на математической конференции РФЯЦ-ВНИИТФ в г. Снежинске в 2007 г.

О Фортране можно писать или очень кратко, или бесконечно много, поэтому данную статью следует рассматривать лишь как авторские наброски. Автор надеется, что она привлечет внимание к Фортрану и по крайней мере уменьшит недоверие и пренебрежение к нему, которые возникли совершенно незаслуженно в среде программистов за последние два десятилетия. При подготовке статьи использовались как официально опубликованные материалы, так и сведения, помещенные в Интернете. За возможные неточности автор приносит свои извинения.

Здравствуй, дружок мой, Фортрану обученный,
Бедный, несчастный, отшельник замученный,
Будет тебе горевать!

(*Из компьютерной поэзии в Интернете* [1])

Введение

Сначала несколько слов о том, как появилась эта статья. В конце 2006 г. на сайте Британского компьютерного общества (British Computer Society, http://www.fortran.bcs.org/) было сделано сообщение о том, что 25 января 2007 г. состоится заседание, посвященное пятидесятилетнему юбилею с момента официального появления языка Фортран, и была представлена программа этого торжественного собрания. В нашей стране тоже принято отмечать различные юбилейные даты, но это событие прошло совершенно незаметно (по крайней мере в Интернете об этом информации не было). А уже в марте на многочисленных сайтах сообщалось о кончине Джона Бэкуса, создателя первого в мире языка программирования высокого уровня — Фортрана. Тогда и появилась идея подготовить обзорный материал, посвященный истории и развитию языка Фортран.

Программисты старшего поколения, на плечи которых в 70—80-е годы легла основная тяжесть по созданию многочисленных прикладных программ и комплексов научного назначения на языке Фортран, с ностальгией вспоминают эти годы, но, как правило, недостаточно осведомлены о современных возможностях Фортрана. Молодое поколение программистов, обученное в университетах новым современным языкам, и прежде всего C++, имеет смутное представление о Фортране вообще и практически его не использует. Скептическое отношение к Фортрану в нашей стране можно почувствовать по содержанию различных программистских форумов и отсутствию профессиональной информации на отечественных сайтах в Интернете. Достаточно упомянуть, что в международном каталоге Интернет-ресурсов (Open Derictory Project, http://www.dmoz.org), имеется более семи сотен ссылок на англоязычные сайты, посвященные Фортрану, и лишь незначительное число ссылок на сайты на русском языке, многие из которых давно устарели и не поддерживаются. В отличие от посвященных другим языкам программирования сайты Фортрана в основном касаются практической стороны программирования: исходных кодов программ в физике, математике, химии, микробиологии, экономике. В последнее время стало заметно некоторое оживление в этой области: появились новые книги, возрождается интерес к Фортрану в университетах, так что слухи

о кончине Фортрана, которые упорно ходили последние пятнадцать лет, скорее всего, окажутся преждевременными.

1957 и 2007 годы в отношении Фортрана оказались тесно связанными между собой:

- в феврале 1957 г. вычислительному миру предъявлен первый язык программирования высокого уровня — Фортран. Это произошло на Западной объединенной компьютерной конференции, проведенной в Лос-Анджелесе (штат Калифорния);
- также в феврале 1957 г. скончался Джон фон Нейман, один из крупнейших ученых XX века, внесший большой вклад в создание первых ЭВМ и разработку методов их применения;
- 25 января 2007 г. состоялась встреча, посвященная 50-летию языка Фортран. Встреча организована совместно Группой специалистов по Фортрану (Fortran Specialist Group) и Обществом защиты компьютеров (Computer Conservation Society). Заседание состоялось в лондонском офисе Британского компьютерного общества;
- 17 марта 2007 г. не стало Джона Бэкуса, человека, под руководством которого был разработан первый в мире высокоуровневый язык программирования — Фортран.

Достоинства Фортрана

Фортран занимает свою нишу: это решение научно-технических задач и математическая обработка числовых данных. Среди ученых, например, ходила такая присказка, что любая математическая задача уже имеет решение на Фортране. На протяжении десятилетий были созданы тысячи фортрановских прикладных программ и всевозможных библиотек, которые остаются популярными (главным образом в научной среде) и по сей день. Одна из наиболее известных таких библиотек — IMSL фирмы Visual Numerics — включает свыше тысячи процедур математической и статистической обработки данных и по существу является стандартом на самых различных компьютерных платформах. Большинство таких библиотек является фактически достоянием человечества: они доступны в исходных кодах, хорошо документированы, отлажены и весьма эффективны, поэтому изменять, а тем более переписывать их на другие современные языки программирования нецелесообразно, несмотря на то, что такие попытки регулярно предпринимаются.

Можно отметить следующие основные достоинства языка Фортран:

- межплатформенная совместимость, основанная на жестком стандарте языка, которого стараются придерживаться все разработчики компиляторов с Фортрана для любых ЭВМ;
- высокая эффективность исполняемого кода. Этому способствовала структура самого языка, а также специальные усилия, которые прилагали разработчики компиляторов с Фортрана. По этому показателю Фортран всегда превосходил другие языки, в том числе и C++;
- большой набор всевозможных библиотек численных методов, которые можно использовать в программах на Фортране (LAPACK, IMSL, BLAS, NAG и др.);
- простота изучения языка. Это относится и к последнему используемому стандарту 1995 года;
- межъязыковое программирование. Многие широко используемые системы программирования позволяют объединять фортрановские объектные коды с объектными кодами, полученными от компиляторов с других языков. Это позволяет создавать более гибкие и многофункциональные программы;
- возможность использования большого количества библиотек различных приложений для организации распределенных вычислений (например, MPI и PVM), построения графических интерфейсов (Quickwin, FORTRAN/TK), доступа к СУБД (например, Oracle).

Краткая предыстория создания языков высокого уровня

Нельзя сказать, что язык Фортран появился внезапно. У него, как и у всего другого, была своя предыстория, направленная на упрощение процесса программирования [2].

Первым практическим шагом в этом направлении стала замена двоичных кодов восьмеричными или шестнадцатеричными кодами команд и адресов. Считается, что именно с этого началось создание языков первого поколения.

В начале 50-х годов появились языки ассемблера (второе поколение), которые позволили записывать машинные команды в текстовой фор-

ме. Они переводили символическую запись в двоичную форму и брали на себя задачу распределения памяти.

Автором самого первого ассемблера для машины EDSAC (Великобритания) стал англичанин Дэвид Уиллер. Ему же принадлежала идея создания программной системы с помощью инструмента вызова закрытых подпрограмм, которая позволяет собирать программу из отдельных модулей, разработанных независимо. В 1951 г. Уиллер написал, вероятно, первую работу по программированию "Подготовка программ для электронных цифровых компьютеров". Языки ассемблеров совершенствовались параллельно с развитием архитектуры компьютеров, вскоре появились программы-загрузчики, макроассемблеры.

Программирование на ассемблере было чрезвычайно эффективным с точки зрения использования машинных ресурсов, но качество программ в значительной степени зависело от квалификации программиста. Кроме того, программы были привязаны к определенной архитектуре, следовательно, не было речи ни о какой мобильности. Понятно, что эти проблемы могли быть решены только средствами языков программирования высокого уровня (третье поколение). И первым среди таких языков стал Фортран.

Корпорация IBM

Многие выдающиеся инновации в области вычислительной техники в то время родились в корпорации IBM [3]. Одним из таких достижений стал язык Фортран. Этому событию способствовало то, что в начале 50-х, в разгар Корейской войны, армии США требовались вычислительные машины. Спрос был, но на тот момент IBM их еще не выпускала. Однако после прихода к руководству корпорацией Томаса Уотсона-младшего IBM активно включилась в борьбу на этом поприще с компанией UNIVAC. Первый серийный ламповый компьютер IBM 701, появившийся в 1952 г., выполнял до 2200 операций умножения в секунду. Основными заказчиками были военные (компьютер так и назывался — Defense Calculator — военный калькулятор), а затем правительственные и крупные научные организации. Модель оказалась на редкость удачной и была выпущена серией почти в 200 экземпляров, а ее "наследница" для гражданского применения, модель IBM 704 (рис. 1), — в еще большем количестве. Появление большого парка компьютеров породило новую проблему — проблему программирования. Пока компьютеры были единичными, о трудоемкости программирования особенно не задумывались, но когда этот вид деятельности стал массовым, потребовались средства автоматизации.

Джон фон Нейман

Говоря о первых шагах вычислительной техники, нельзя не вспомнить еще об одном человеке, который внес заметный вклад в ее становление. Это Джон фон Нейман (John von Neumann) — американский ученый венгерского происхожде-

Рис. 1. Компьютер IBM 704, для которого был разработан первый компилятор с Фортрана

ния, работавший в области математики, физики, химии, астрономии, биологии и экономики [4, 5].

Джон фон Нейман (рис. 2) родился 28 декабря 1903 г. в Будапеште в семье банкира Макса фон Неймана. В шесть лет он свободно изъяснялся на древнегреческом языке, а в восьмилетнем возрасте уже владел основами высшей математики.

По окончании лютеранской гимназии фон Нейман поступает в Федеральную высшую техническую школу в Цюрихе и одновременно на математический факультет Будапештского университета. В 1925 г. он получает диплом инженера-химика в Цюрихе и успешно защищает диссертацию по теме "Аксиоматическое построение теории множеств" на звание доктора философии в Будапештском университете.

Далее он совершенствует свои знания в Геттингенском университете, где вслед за статьей "Об основаниях квантовой механики", написанной совместно с Д. Гильбертом и Л. Нордгеймом, публикует серию работ "Математическое обоснование квантовой механики", "Теоретико-вероятностное построение квантовой механики" и "Термодинамика квантово-механических систем". В 1928 г. вышла его работа "К теории стратегических игр", ставшая краеугольным камнем возникшей гораздо позже теории игр. В этот же период им опубликованы работы по теории функций действительной переменной, общей теории меры, абстрактной теории автоматов. В 1927 г. фон Нейман становится приват-доцентом Берлинского, а с 1929 г. — Гамбургского университета.

В 1930 г. Джон фон Нейман эмигрировал из Европы в США и начал преподавать в Принстонском институте перспективных исследований. Позднее в Лос-Аламосе (штат Нью-Мексико) он был привлечен к участию в сверхсекретном Манхэттенском проекте по созданию атомной бомбы. Фон Нейман математически доказал осуществимость взрывного способа детонации атомной бомбы, а в 1946 г. вместе с Клаусом Фуксом запатентовал термоядерное взрывное устройство, приводимое в действие атомным взрывом, т. е. саму идею водородной бомбы.

В 1946 г. фон Нейман в Институте перспективных исследований (IAS) сформировал команду разработчиков для создания новой вычислительной машины (рис. 3). Он предложил использовать в качестве элементов памяти не линии задержки, а электронно-лучевые трубки. Задуманная им машина была построена под руководством Дж. Бигелоу [6]. Весной 1951 г. машину, названную так же, как институт, IAS, отдали в распоряжение институтских и "пришлых" программистов. Летом 1951 г. группа ученых из Лос-Аламоса выполнила на IAS большие вычисления, связанные с термоядерными процессами. IAS работала непрерывно, по 24 часа в сутки, в течение почти 60 дней, многие промежуточные результаты проверялись повторной прогонкой программ, и за все это время было обнаружено не более полдюжины ошибок.

Рис. 2. Джон фон Нейман

IAS (рис. 4) была одноадресной машиной с естественным управлением операциями; она содержала примерно 2500 электронных ламп и оперировала 39-разрядными двоичными числами с фиксированной запятой. Командный набор состоял из 44 команд. Машина весила около 400 кг и потребляла 28 кВт. Официальное представле-

Рис. 3. Разработчики ЭВМ IAS (слева направо: Нейман, Бигелоу, Померен, Голдстайн)

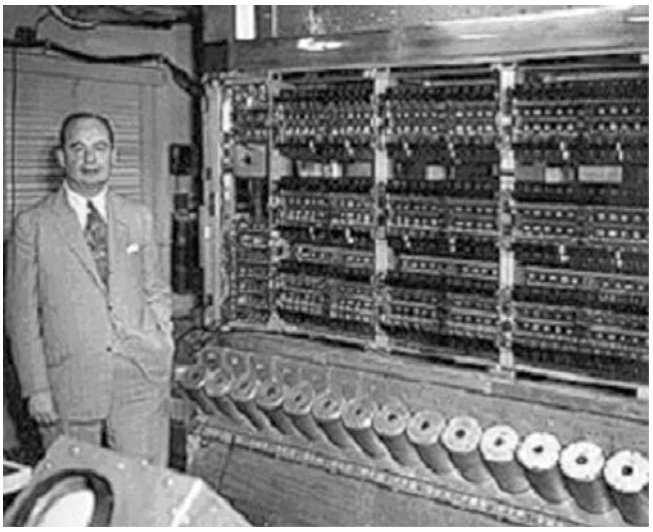

Рис. 4. Фон Нейман у ЭВМ IAS

ние IAS состоялось 10 июня 1952 г. Ее логические и схемные решения послужили прототипами при создании вычислительных машин в Лос-Аламосской национальной лаборатории, Иллинойсском университете, Окриджской и Аргоннской национальных лабораториях и корпорации RAND.

"Клоны" IAS появились затем в Швеции, Израиле, ФРГ, Дании, Австралии, Японии. На этих машинах была не только опробована новая концепция построения ЭВМ, они дали мощный толчок развитию национальных компьютерных индустрий в ряде стран западного мира. Машины приобрели широкую известность под названием JOHNIAC — в честь фон Неймана. Именно JOHNIAC позволил осуществить важные расчеты при создании водородной бомбы, превосходившие по своему объему все, что когда-либо было сосчитано человечеством.

Компьютер фон Неймана

Первая электронная вычислительная машина была построена в 1943—1946 гг. в школе инженеров-электриков Мура Пенсильванского университета под руководством Джона Мочли и Преспера Эккерта и получила название ENIAC. В создании следующей машины EDVAC, построенной также в школе Мура, принял активное участие фон Нейман. Он разработал подробную логическую схему машины, в которой структурными единицами были не физические элементы цепей, а идеализированные вычислительные элементы, и предложил ряд инженерных решений.

В июле 1954 г. фон Нейман подготовил отчет "Предварительный доклад о машине EDVAC". Этот отчет представлял собой прекрасное описание не только самой машины, но и ее логических свойств. Присутствовавший на докладе военный представитель Голдстейн размножил отчет и разослал ученым США и Великобритании. Таким образом, "Доклад" стал первой работой по цифровым электронным компьютерам, с которой познакомились широкие круги научной общественности. Его передавали из рук в руки, из лаборатории в лабораторию, из университета в университет, из одной страны в другую.

В этом отчете были изложены основные принципы функционирования универсальных вычислительных устройств (компьютеров). Фон Нейман описал, каким должен быть компьютер, чтобы он стал универсальным и удобным средством для обработки информации. По его мнению, компьютер должен иметь (рис. 5):

- *арифметическо-логическое устройство*, которое выполняет арифметические и логические операции;
- *устройство управления,* которое организует процесс выполнения программ;
- *запоминающее утройство* для хранения программ и данных;
- *внешние устройства* для ввода/вывода информации.

Свои предложения Джон фон Нейман сформулировал в виде нескольких принципов:

- программного управления;
- однородности памяти;
- адресности.

Принцип программного управления обеспечивает автоматизацию процессов вычислений на ЭВМ. Программа состоит из набора команд, которые выполняются процессором автоматически друг за другом в определенной последовательности.

Рис. 5. Устройство компьютера согласно принципам фон Неймана

Программы и данные хранятся в одной и той же памяти. Поэтому компьютер не различает, что хранится в данной ячейке, — число, текст или команда. Над командами можно выполнять такие же действия, как над данными. Это открывает целый ряд возможностей. Например, программа в процессе своего выполнения также может подвергаться модификации, что позволяет задавать в ней самой правила получения некоторых ее частей (так в программе организуется выполнение циклов и подпрограмм).

Структурно основная память состоит из перенумерованных ячеек. Процессору в произвольный момент времени доступна любая ячейка. Отсюда следует возможность давать имена областям памяти так, чтобы к запомненным в них значениям можно было впоследствии обращаться или менять их в процессе выполнения программы с использованием присвоенных имен.

Компьютеры, построенные на основе указанных принципов, относят к типу фон-неймановских. До сих пор эти принципы не устарели, и большинство существующих сегодня компьютеров реализовано по схеме, предложенной фон Нейманом.

Умер Джон фон Нейман 8 февраля 1957 г. в возрасте 54 лет в Вашингтоне — погиб от рака (саркомы), вызванного, как предполагают, облучением при испытаниях водородного оружия на атолле Бикини (Маршалловы острова).

Появление Фортрана: хроника событий

Попытки автоматизировать программирование [7] были и до Фортрана: язык A-0 для компьютера UNIVAC (1952 г.), экспериментальный *транслятор формул*, созданный в Массачусетском технологическом институте (Massachusetts Institute of Technology — MIT) (1954 г.). Но эти реализации показывали чудовищную неэффективность сгенерированного машинного кода, что породило стереотип "никакая автоматизация не сможет заменить человека-программиста, пишущего на ассемблере". Фортран этот стереотип разрушил.

В конце 1953 г. молодой научный сотрудник Джон Бэкус обратился к руководству корпорации IBM с предложением о том, как можно решить возникшую проблему повышения эффективности работ на серийных вычислительных машинах. В команде, собранной Бэкусом, не было профессиональных специалистов по программному обеспечению. В своей работе он и его коллеги большей частью ориентировались на интуицию. Тем не менее 10 ноября 1954 г. был выпущен отчет ("Preliminary report. Specifications for the IBM Mathematical FORmula TRANslation System, FORTRAN"), являющийся, по-видимому, самым ранним документом, посвященным Фортрану. Первое предложение этого отчета гласит: "Система перевода математических формул, или кратко FORTRAN, будет содержать большое количество программ, которые позволят вычислительной машине IBM 704 воспринимать описание задачи в терминах математических обозначений и автоматически создавать высокоэффективную программу для ее решения".

Так группой из восьми человек под руководством Джона Бэкуса был создан первый язык программирования высокого уровня для машин, построенных по классической схеме фон Неймана. При этом удалось сделать две замечательные вещи: во-первых, совместить возможность записи алгоритмов с выразительными способностями естественного языка; во-вторых, реализовать эффективную технологию компиляции, которая позволяла получать коды, не слишком громоздкие по сравнению с кодами, написанными на ассемблере. Сейчас это кажется вполне логичным, но воспитанное на ассемблерах первое поколение программистов с трудом воспринимало отсутствие видимого соответствия между операторами языка и машинными командами. А экспериментальные компиляторы, создаваемые в MIT, уступали Фортрану по эффективности кода на порядок.

С самого начала деятельность группы проходила в атмосфере скептицизма и недоверия. Тем не менее работа над языком шла быстро. Бэкус понимал, что развеять сомнения в возможности автоматического программирования можно лишь тогда, когда программы на Фортране будут такими же быстродействующими и надежными, как и написанные в машинных кодах на языке ассемблера. Создать эффективный компилятор оказалось трудной задачей. Работа над ним заняла около двух лет. Фортран признавали неохотно. Однако по сравнению со своими предшественниками Фортран был достаточно прост в использовании и изучении. Он на долгие годы стал самым популярным языком программирования, особенно среди научных работников.

Впервые Фортран был предъявлен вычислительному миру на Западной объединенной компьютерной конференции, проведенной в

Лос-Анджелесе (штат Калифорния) в феврале 1957 г., через два с половиной года после начала работ над ним. В середине апреля 1957 г. была осуществлена первая зарегистрированная поставка компилятора Фортрана для IBM 704, установленной в Westinghouse-Bettis[1]. Компилятор Фортран I состоял из 25 тыс. строк кода и поставлялся с каждой машиной IBM 704.

В июне 1958 г. появилась новая, усовершенствованная версия языка — Фортран-II. Основное отличие от первой версии заключалось во введении понятия подпрограмм. В 1962 г. было выпущено предварительное описание версии языка, которую теперь называют Фортран-IV (в нашей стране эта версия была реализована на БЭСМ-6 и ЕС ЭВМ). По сравнению с Фортраном-II было добавлено значительное число возможностей, включая операторы описания типа, логический оператор `IF`, имена функций и подпрограмм в качестве параметров процедур, операторы задания начальных значений переменных. В то же время в новую версию не вошли некоторые имевшиеся ранее машинно-ориентированные операторы.

Джон Бэкус — создатель языков программирования Фортран и Алгол

Говоря о языке Фортран, нельзя не сказать о его создателе Джоне Бэкусе (рис. 6). Он родился в 1924 г., в 1943 г. поступил в университет штата Вирджиния. В том же году был отчислен из университета и призван на военную службу. После окончания службы в армии во время второй мировой войны попал в Нью-Йорк, где стал учиться в радиотехнической школе. Один из преподавателей убедил Бэкуса продолжить образование, и он поступил в Колумбийский университет.

В 1950 г. Джон Бэкус, получив степень магистра математики, пришел на работу в фирму IBM. Вскоре он возглавил группу, разрабатывавшую интерпретатор "Быстрый кодировщик" для компьютера IBM 701, а позже принял участие в создании усовершенствованного варианта этой машины, модели IBM 704.

С 1953 по 1957 г. Бэкус возглавлял работы над языком и компилятором Фортран. В конце 1950-х годов он участвовал в создании языка программирования Алгол (ALGOrithmic Language). Алгол оказал существенное влияние на формирование языков высокого уровня, а языки Паскаль, Си и Ада даже называют алголоподобными. В процессе работ над Алголом Бэкус предложил специальную систему определений для языков программирования. Такой способ записи назвали нормальной формой Бэкуса, или БНФ. Позднее датский астроном Петер Наур внес уточнения в методику Бэкуса и эту форму стали называть формой Бэкуса—Наура, при этом сокращение осталось прежним — БНФ. БНФ долго употреблялась при описании языков высокого уровня.

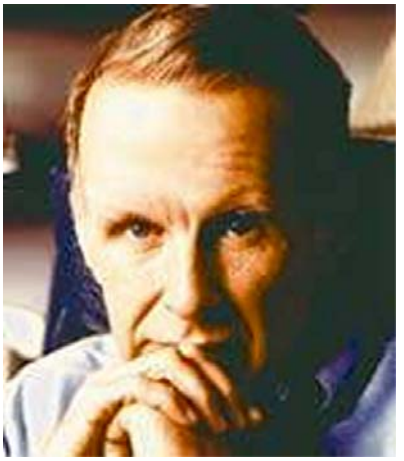

Рис. 6. Джон Бэкус

В 1975 г. Джон У. Бэкус за разработку Фортрана был удостоен ряда наград, в том числе Национальной медали США за заслуги в области науки, а в 1977 г. получил премию Тьюринга за создание языков программирования и разработку формальных процедур спецификации языков программирования. Скончался Джон Бэкус 17 марта 2007 г.

Стандартизация языка Фортран

Отсутствие стандарта языка мешало переносить программы с машин одного типа на машины других типов. В мае 1962 г. был сформирован комитет ASA X3.4.3 с целью разработки стандарта языка Фортран. Результатом его деятельности явились два стандарта с официальными названиями Фортран и Основной (Basic) Фортран. Окончательно ситуация изменилась в 1966 г., когда была завершена разработка американского стандарта на язык ANSI (American National Standards Institute), известного как Фортран 66.

Стандартизация языков программирования создает предпосылки для повышения мобиль-

[1] В лаборатории Westinghouse положено начало развитию мирной ядерной энергетики.

ности программного обеспечения. Высокая мобильность облегчает адаптацию программы для работы в различных окружениях, что позволяет использовать ранее созданные прикладные программы и облегчает перенос программы с одной платформы на другую.

Международные стандарты языка являются результатом совместной деятельности экспертов многих стран. Стандартизацией языка Фортран занимаются Американский технический комитет J3 ANSI и эксперты рабочей группы ISO/IEC JTC1/SC22/WG5[2] (WG5). Членами WG5 являются специалисты ряда стран, в том числе нашей страны. В их числе — представители компьютерных фирм, крупных университетов. Многие из тех, кто ответственен за разработку коммерческих Фортран-компиляторов, являются членами J3 и/или WG5.

Язык Фортран подвергался стандартизации в рамках ANSI и ISO пять раз [8]. Принятые стандарты языка:

Фортран IV — он же Фортран 66 (1966 г.).

Фортран 77 (1978 г.). Язык получил множество улучшений: строковый тип данных и функции для его обработки; блочные операторы `IF, ELSE IF, ELSE, END IF`; оператор включения фрагмента программы `INCLUDE` и т. д.

Фортран 90 (1991 г.) [9, 10]. Значительно переработанная версия языка. Введен свободный формат написания кода. Появились дополнительные описания `IMPLICIT NONE, TYPE, ALLOCATABLE, POINTER, TARGET, NAMELIST`; управляющие конструкции `DO ... END DO, DO WHILE, CYCLE, SELECT CASE, WHERE`; возможность работы с динамической памятью (`ALLOCATE, DEALLOCATE, NULLIFY`); программные компоненты `MODULE, PRIVATE, PUBLIC, CONTAINS, INTERFACE, USE, INTENT`. Появились новые встроенные функции, в первую очередь для работы с массивами. В языке появились элементы объектно-ориентированного программирования (ООП). Отдельно объявлен список устаревших черт языка, предназначенных для удаления в будущем.

Фортран 95 (1997 г.) [11—13]. Коррекция предыдущего стандарта.

Фортран 2003 (2004 г.) [14]. Дальнейшее развитие поддержки ООП в языке. Средства взаимодействия с языком Си и операционной системой. Новые средства ввода/вывода.

Международный стандарт языка Фортран 95 (ISO/IEC 1539-1:1997), принятый в 1997 г., реализован во всех коммерческих компиляторах. Многие из них уже включают некоторые новые средства Фортрана 2003 (ISO/IEC 1539-1:2004(E)). В настоящее время опубликован проект будущего стандарта, неформально названного Фортраном 2008 (ISO/IEC 1539-1:2004(E)), который предусматривает весьма существенные нововведения.

Современные стандарты Фортрана представляют собой семейства стандартов, состоящие из нескольких частей. Например, для Фортрана 95 это:

- основной (базовый) язык ISO/IEC 1539-1:1997;
- описание средств для работы с символьными строками переменной длины ISO/IEC 1539-2:2000(E);
- описание языка условной компиляции ISO/IEC 1539-3:1998.

Фортран I

Язык Фортран был предназначен для научных вычислений [15, 16]. В нем отсутствовали многие привычные языковые конструкции и атрибуты. Компилятор не проверял синтаксически правильную программу с точки зрения семантической корректности (соответствие типов и проч.). В нем не было современных способов структурирования кода и данных. Это осознавали и сами разработчики. По признанию самого Бэкуса, перед ними стояла задача разработки скорее компилятора, чем языка.

[2] ISO (International Organization for Standardization — Международная организация по стандартизации, http://www.iso.org/) занимается выпуском стандартов. Сфера деятельности ISO касается стандартизации во всех областях, кроме электротехники и электроники, относящихся к компетенции IEC (International Electrotechnical Commission — Международная электротехническая комиссия). Некоторые виды работ выполняются совместными усилиями этих организаций (совместные документы обозначаются ISO/IEC).

Разработкой языка Фортран занимается рабочая группа WG5 (Working Group 5) подкомитета SC22 (subcommitees 22) Объединенного технического комитета JTC1 (Joint Technical Committee 1). Публикуемые материалы этого комитета имеют обозначение ISO/IEC JTC1/SC22/WG5.

Исходный вариант Фортрана для IBM 704 содержал 32 оператора, включая:

- операторы `DIMENSION` и `EQUIVALENCE`;
- арифметический оператор `IF`;
- оператор `IF` для контроля исключения (`ACCUMULATOR OVERFLOW`, `QUOTIENT OVERFLOW`, `DIVIDE CHECK`) и логический оператор `IF`;
- оператор присваивания ;
- `DO`-циклы;
- форматный ввод/вывод: `FORMAT`, `READ`, `READ INPUT TAPE`, `WRITE`, `WRIT OUTPUT TAPE`, `PRINT`, и `PUNCH`;
- бесформатный ввод/вывод: `READ TAPE`, `READ DRUM`, `WRITE TAPE`, и `WRITE DRUM`;
- другие операторы ввода/вывода: `END FILE`, `REWIND`, и `BACKSPACE`;
- `PAUSE`, `STOP` и `CONTINUE`;
- оператор `FREQUENCY` (подсказки компилятору для выполнения оптимизации).

Из приведенного списка операторов видно, что, несмотря на машинную независимость языка, в нем явно проступают черты, отражающие устройства универсальной машины фон Неймана и даже конкретные внешние устройства (АЦПУ, магнитные ленты, ввод с перфокарт).

Несмотря на приближенность к машине, появление Фортрана было встречено еще более яростной критикой, чем внедрение ассемблера. Программистов пугало снижение эффективности программ за счет использования промежуточного звена в виде компилятора. Хороший программист при решении какой-либо небольшой задачи, как правило, вручную мог написать код, работающий быстрее, чем код, полученный в результате компиляции. Позднее пришло понимание того, что реализация больших проектов невозможна без применения языков высокого уровня. Мощность вычислительных машин росла, и падение эффективности, которое раньше считалось угрожающим, перестало пугать программистов.

Некоторые проблемы программирования на раннем Фортране

Несмотря на многие позитивные черты языка, его выразительные средства изначально были весьма бедны, поскольку Фортран был одним из первых языков высокого уровня.

Структура программ была ориентирована на ввод с перфокарт и имела ряд свойств, удобных именно для этого случая. Так, 1-я колонка служила для маркировки текста как комментария, с 1-й по 5-ю располагалась область меток, а с 7-й по 72-ю — собственно текст оператора или комментария. Колонки с 73-й по 80-ю могли служить для нумерации карт (чтобы восстановить случайно рассыпавшуюся колоду), транслятором они игнорировались. Если текст оператора не вписывался в отведенное пространство (с 7-й по 72-ю колонку), в 6-й колонке следующей карты ставился признак продолжения, затем оператор продолжался на ней. Такой позиционный (похожий на таблицы) формат записи операторов позволял заводить специальные бланки, используемые для записи текстов программ перед набивкой их на перфокартах (рис. 7). Расположить два или более оператора в одной строке (карте) было нельзя. Когда перфокарты ушли в прошлое, эти достоинства превратились в серьезные неудобства.

Своего рода визитной карточкой старого Фортрана является огромное количество меток, которые использовались в операторах безусловного перехода `GOTO`, операторах циклов и операторах описания форматного ввода/вывода `FORMAT`. Большое количество меток и операторов `GOTO` часто делало программы на Фортране трудными для понимания.

В литературе описывается легенда (скорее похожая на правду), когда в операторе цикла

$$\texttt{DO 10 I} = 1,100$$

вместо запятой была набита точка, и в результате получился оператор присваивания

$$\texttt{DO 10 I} = 1.100$$

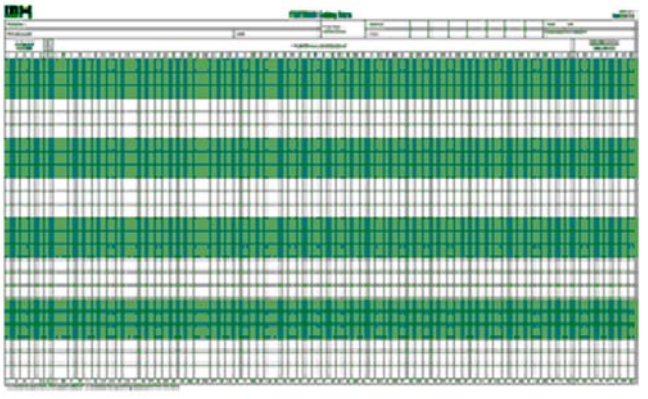

Рис. 7. Бланк для записи программ на Фортране, созданный в корпорации IBM

Программа с такой ошибкой управляла полетом ракеты. Ракета сбилась с курса и была взорвана сразу после старта. Именно такой негативный опыт стал причиной, по которой в ряде современных языков программирования метки и связанные с ними операторы безусловного перехода вообще отсутствуют.

В современных стандартах Фортрана, начиная с Фортрана 90, в дополнение к фиксированному формату исходного текста появился свободный формат записи операторов. А использование таких операторов, как `DO ... END DO`, `DO WHILE`, `SELECT CASE` позволяет обходиться без меток.

Этапы развития языка Фортран

Нет смысла описывать подробно особенности стандартов языка Фортран, ранее широко использовавшихся, а сейчас безвозвратно устаревших, — Фортрана 66 и Фортрана 77. Тем не менее краткое изложение их особенностей поможет лучше представить приспособляемость Фортрана к изменяющимся внешним условиям — требованиям программистов, совершенствованию вычислительной техники.

Фортран 66. Основные средства Фортрана 66 [15, 16]:

- головная программа, `SUBROUTINE`, `FUNCTION` и `BLOCK DATA`;
- типы данных `INTEGER`, `REAL`, `DOUBLE PRECISION`, `COMPLEX` и `LOGICAL`;
- операторы `COMMON`, `DIMENSION` и `EQUIVALENCE`;
- оператор `DATA` для определения начальных величин;
- встроенные и `EXTERNAL` функции;
- оператор присваивания;
- `GOTO`, `GOTO` по предписанию, вычисляемый оператор `GOTO`;
- логический и арифметический операторы `IF`;
- `DO`-циклы;
- операторы для последовательного ввода/вывода `READ`, `WRITE`, `BACKSPACE`, `REWIND`, `ENDFILE`;
- оператор `FORMAT`;
- операторы `CALL`, `RETURN`, `PAUSE`, `STOP`;
- текстовые константы в операторах `DATA` и `FORMAT`;
- идентификаторы до шести символов в длину.

Фортран 77. Стандарт Фортран 77 включил в себя много изменений [17]: расширение возможностей имеющихся операторов; новые средства, поддерживающие структурный подход в программировании, а также расширяющие возможности такого подхода:

- оператор объявления имени главной программы `PROGRAM`;
- данные текстового типа и операции для их обработки;
- новые операторы (блочные операторы `IF`, `ELSE IF`, `ELSE`, `END IF`, оператор объявления типа `CHARACTER`, вспомогательные операторы ввода/вывода `OPEN`, `CLOSE`, `INQUIRE`, оператор объявления стандартных функций `INTRINSIC`, оператор объявления имени константы `PARAMETER`, оператор включения фрагмента программы `INCLUDE`, оператор сохранения значений объектов из подпрограмм `SAVE`);
- отрицательные и нулевые значения нижней границы измерения массива;
- выражения в списке вывода;
- ввод/вывод во внутренние файлы;
- именованный модуль `BLOCK DATA`;
- обобщенные имена для стандартных функций;
- логические операции `.EQV.` и `.NEQV.` и др.

Фортран 90/95. После длительного перерыва появился новый стандарт языка — Фортран 90 [18] (несколько позднее — его незначительное обновление, Фортран 95), который явился выражением решительного шага вперед в развитии языка. В нем появились:

- свободный формат записи исходного текста;
- встроенные и производные типы данных (операция присваивания, перегрузка операций);
- операции над массивами, секциями массивов (встроенные процедуры);
- операторы и конструкции `WHERE` и `FORALL`;
- механизмы динамического размещения массивов;
- указатели;
- модули;
- поддержка структурного программирования;
- средства поддержки параллельности.

Данный стандарт по своим возможностям приблизил Фортран к современным языкам программирования, а по некоторым средствам (операции над массивами и секциями, встроенные матричные функции, операторы и конструкции `WHERE` и `FORALL`) позволил их превзойти. Компиляторы, реализующие данный стандарт, широко используются в нашей стране. На русском языке имеется достаточно много изданий [8, 19—27], детально и качественно описывающих особенности программирования в рамках данного стандарта. Тем не менее для полноты изложения приведем несколько примеров, иллюстрирующих мощь и удобства языка.

Каждый объект данных имеет тип, ранг (размерность), а также может иметь некоторое количество дополнительных свойств — атрибутов. Эти свойства определяют характеристики соответствующего объекта и особенности его использования.

Пример описания массива констант:

```
INTEGER, DIMENSION (5), PARAMETER ::  &
        ARRAY = (/ 1,2,3,4,5/)
```

Пример двух способов описания величин, обеспечивающих точность представления не менее 10 знаков и диапазон десятичного порядка не менее чем $(-30, +30)$:

```
INTEGER, PARAMETER ::  &
    PARAM = SELECTED_REAL_KIND(10,30)
REAL (KIND = PARAM) X, Y
    !  PARAM — параметр типа
REAL (SELECTED_REAL_KIND(10,30)) ::  P,Q
```

Примеры, иллюстрирующие описания секций массивов:

```
INTEGER, DIMENSION :: Z(M,N), V(3), A(3,3)
```

`Z (3, :)` — третья строка матрицы Z;

`Z (:, J)` — J-й столбец матрицы Z;

`Z (k1:k2, l1:l2:2)` — прямоугольная секция массива;

`V = (/ 2, 1, 3 /)` — конструктор массива;

`A (3, V)` — секция массива, с элементами $A(3,2)$, $A(3,1)$, $A(3,3)$.

Оператор присваивания по маске и конструкция присваивания по маске (`WHERE` и `FORALL`) позволяют в зависимости от некоторого условия выполнить вычисление выражения и присваивание значений не для всех элементов массива, т. е. выполнить присваивание под управлением логической маски.

Чтобы присвоить нуль всему массиву, достаточно написать

```
A=0.0
```

Обнуление только отрицательных элементов выполняется с помощью оператора

```
WHERE (A<0.0) A=0.0
```

Присваивание диагонали матрицы A осуществляется оператором

```
FORALL (I=1:N) A(I,I)=B(I)
```

`FORALL` похож на `DO`-циклы, но выполняется иначе — сначала вычисляются правые части для всех значений индексов, а затем производятся присваивания. Выполнение (для различных значений индексов) может быть параллельным, но результат не зависит от того, поддерживает ли компилятор параллельное выполнение `FORALL`.

Оценивая возможности языка Фортран в сравнении с C++, часто можно услышать упреки в отсутствии средств поддержки ООП. Это справедливо по отношению к более ранним стандартам языка. Современный Фортран поддерживает значительную часть методологии ООП.

Основная идея ООП заключается в том, чтобы обеспечить разработчику возможность создавать типы данных, соответствующие понятиям прикладной задачи, вводить новые операции и структурировать программу таким образом, чтобы эти типы и операции были доступны извне, но детали реализации были бы скрыты от пользователя.

Основные концепции ООП:

- расширяемость типов и операций;
- механизм инкапсуляции для реализации абстрактных типов данных;
- наследование;
- статический и динамический полиморфизм.

В Фортране 90/95 имеется полная поддержка инкапсуляции и абстракции данных за счет программных единиц-модулей, производных типов, операций, определяемых пользователем, и атрибутов `PUBLIC` и `PRIVATE`. Кроме того, такую поддержку обеспечивают различные механизмы динамического размещения объектов.

Наследование реализуется частично посредством производного типа и модулей. Полное наследование в явном виде не поддерживается, но может быть смоделировано.

Статический полиморфизм поддерживается с помощью явного и неявного преобразова-

ний типов, перегрузки операций и процедур, т. е. использования одного и того же имени для нескольких разных объектов в общей области видимости. Динамический полиморфизм в явном виде отсутствует, но может быть смоделирован. В стандарте Фортран 2003 предусмотрена уже полная поддержка ООП.

Средства поддержки параллельности в Фортране 95. Автоматическая векторизация требует довольно сложного анализа исходной программы. Фортран 95 не ориентирован на многопроцессорные системы, но в то же время содержит богатый набор средств, ориентированных на архитектуру с векторными операциями:

- средства для работы с массивами и секциями массивов как с целыми объектами;
- векторные операции;
- стандартные процедуры для работы с массивами;
- поэлементные процедуры;
- условное присваивание массиву под управлением логической маски;
- оператор и конструкция `FORALL`, которые позволяют специфицировать параллельный цикл;
- спецификацию `PURE` для функции без побочного эффекта. Вызов такой функции можно использовать в тех случаях, где возможна параллельная обработка без таких нежелательных последствий, как недетерминизм.

Эти средства позволяют компилятору сгенерировать эффективный код с учетом особенностей аппаратуры.

Фортран 2003. Стандарт Фортран 2003 официально был представлен в ноябре 2004 г. Фортран 2003 является серьезным расширением Фортрана 95. В полном объеме этот стандарт еще не реализован, но некоторые из его возможностей уже внедрены в компиляторах для языка Фортран, разрабатываемых компаниями Intel и Lahey Computer Systems. В изданиях на русском языке практически нет сведений об этом стандарте: в основном это различные переложения официальных документов, выпущенных рабочей группой WG5 по стандартизации Фортрана [28—30]. В новом стандарте учтены пожелания прикладных программистов в соответствии с современными взглядами на программирование. В результате язык Фортран стал сложнее, хотя сохранил все свои прежние возможности. В этом полностью проявляется преемственность языка по отношению к своим предыдущим стандартам.

Приведем перечень основных дополнений, вводимых новым стандартом Фортрана:

- полный набор средств ООП (расширение типа и наследование, динамический полиморфизм, динамическое размещение типа и процедуры, привязанные к типу);
- расширенная поддержка смешанного программирования на Фортране и Си;
- параметризованные производные типы, улучшенное управление доступом к компонентам типа, завершающие подпрограммы;
- новые средства ввода/вывода (асинхронный ввод/вывод, потоковый доступ к файлам, новые спецификаторы формата, управление округлением во время преобразований формата, средства работы с диагностическими сообщениями);
- средства обработки исключительных ситуаций для операций с плавающей точкой стандарта IEEE 1989;
- средства взаимодействия с операционной системой — доступ к аргументам командной строки, переменным окружения и сообщениям об ошибках процессора;
- процедурные указатели;
- работа с символами 32-разрядной кодировки ISO 10646, выбор десятичной точки или запятой при форматном вводе/выводе числовых данных.

Фортран 2008. Несколько слов о новом готовящемся стандарте языка. В качестве стандарта Фортран 2008 взят язык Co-Array Fortran (CAF) — расширение Фортрана 95 для параллельного программирования [31]. Рабочей группой по стандартизации опубликован рабочий проект стандарта языка [32]. Кроме того, опубликовано много материалов как по CAF, так и по результатам исследования его применения. Эти исследования показывают, что по эффективности CAF несколько превосходит средства MPI, а по наглядности программ — вне всякой конкуренции.

В последние годы некоторыми фирмами предпринимаются усилия по разработке новых языков программирования, ориентированных на

многопроцессорные машины. Такие работы [33] при поддержке DARPA (Defense Advanced Research Projects Agency — Агентство передовых оборонных исследовательских проектов, http://www.darpa.mil/body/legacy/index.html) ведутся в фирмах Cray (язык Chapel), IBM (язык X10), Sun Microsystems (язык Fortress). Работы планируется завершить в 2010 г. Но независимо от того, каков будет конечный результат, Фортран еще долгое время будет оставаться востребованным, и прежде всего из-за своего накопленного "богатства" и удивительной способности выживать, приспосабливаясь к изменяющимся условиям существования.

В заключение хочется поздравить всех поклонников языка Фортран со славным юбилеем и пожелать стойко нести свою веру в него всем, кто в этом нуждается.

Список литературы

1. Компьютерная поэзия. http://home.od.ua/watman/song.html.
2. *Черняк Л.* Транзистор программирования // Computerworld. 2006. № 11 (http://www.osp.ru/cw/2006/11/).
3. *Дубова Н.* Великая империя // Там же. 2001. № 10 (http://www.osp.ru/cw/2001/10/).
4. *Данилов Ю. А.* Джон фон Нейман. М.: Знание, 1981.
5. Знакомьтесь: компьютер. М.: Мир, 1989.
6. *Полуянов Ю.* Воплощение идей. http://www.bytmarket.ru.
7. *Шепелев В.* Генеральная линия: от Fortran до C# // Компьютерра. 2007. № 8 (http://offline.computerra.ru/2007/676/309134/).
8. *Горелик А. М.* Программирование на современном Фортране. М.: Финансы и статистика, 2006.
9. Information Technology. Programming Languages. Fortran. ISO/IEC 1539:1991(E). Second Edition, 1991-07-01. ISO Publications Department.
10. Information Technology. Programming Languages. Fortran. Part 2: Varying Length Characater Strings. ISO/IEC 1539-2:1994. ISO Publications Department.
11. Information Technology. Programming Languages. Fortran. Part 1: Base Language. ISO/IEC 1539-1:1997. ISO Publications Department.
12. Information Technology. Programming Languages. Fortran. Part 2: Varying Length Character Strings. ISO/IEC 1539-2:2000(E). ISO Publications Department.
13. Information Technology. Programming Languages. Fortran. Part 3: Conditional Compilation. ISO/IEC 1539-3:1998. ISO Publications Department.
14. Information Technology. Programming Languages. Fortran. Part 1. ISO/IEC 1539-1:2004. ISO Publications Department.
15. Открытая энциклопедия Википедия на английском языке. http://en.wikipedia.org/wiki/.
16. Открытая энциклопедия Википедия на русском языке. http://ru.wikipedia.org/wiki/.
17. Вычислительная техника социалистических стран. Вып. 24. М.: Финансы и статистика, 1988. С. 134.
18. Фортран 90. Международный стандарт: Пер. с англ. М.: Финансы и статистика, 1998.
19. *Меткалф М., Рид Дж.* Описание языка программирования Фортран 90: Пер. с англ. М.: Мир, 1995.
20. *Бартеньев О. В.* Visual Fortran: новые возможности. М.: Диалог-МИФИ, 1999.
21. *Бартеньев О. В.* Fortran для студентов. М.: Диалог-МИФИ, 1999.
22. *Рыжиков Ю. И.* Программирование на Fortran PowerStation для инженеров. М.: КОРОНА принт, 1999.
23. *Бартеньев О. В.* Современный Fortran (3-е издание). М.: Диалог-МИФИ, 2002.
24. *Штыков В. В.* Фортран & WIN32 API: создание программного интерфейса для Windows средствами современного Фортрана. М.: Диалог-МИФИ, 2001.
25. *Немнюгин С. А., Стесик О. Л.* Современный Фортран. Самоучитель. СПб.: БХВ-Санкт-Петербург, 2004.
26. *Дымов В.* Язык программирования Фортран. М.: Майор, 2003.
27. *Васильченко В. В.* Программирование Windows-приложений на языке Fortran. Элементы управления и графика Windows. М.: Диалог-МИФИ, 2006.

28. *Reid J.* WG5 Convener. The New Features of Fortran 2003. ISO/IEC JTC1/SC22/WG5 N1579.

29. *Reid J.* Fortran 2003: the Latest Fortran Standard. JKR Associates Convener, ISO Fortran Committee WG5 (ftp://ftp.numerical.rl.ac.uk/pub/jkr/f2003.pdf).

30. Final Committee Draft of Fortran 2003 Standard. ISO/IEC JTC1/SC22/WG5 N1578 (ftp://ftp.j3-fortran.org/j3/doc/standing/007/).

31. *Reid J.* Co-Arrays in the Next Fortran Standard. ISO/IEC JTC1/SC22/WG5 N1669. 2007. JKR Associates, UK.

32. Information Technology. Programming Languages. Fortran. Part 1: Base Language. International Standard Programming Language Fortran. Working Draft. J3/07-007.

33. *Антес Г.* Наследники Фортрана и Си // Computerworld. 2007. № 14 (http://www.osp.ru/cw/2007/14/).

Статья поступила в редакцию 09.07.07.