Московский Авиационный Институт
(Национальный Исследовательский Университет)

Факультет информационных технологий и прикладной математики
Кафедра 806 “Вычислительная математика и программирование”

Реферат
по курсу “Архитектура компьютера”
1 семестр

«Джон Бэкус»

Студент: Старостина А.А.
Группа: М8О-108Б-22,
№ по списку: 19
Руководитель: Сахарин Н.А.
Дата: 08.01.2023
Оценка: _____

г. Москва, 2023

Содержание

Введение

Джон Бэкус

Джон Бэкус – значимая фигура в области информатики и математики. Моя задача – изучить его биографию.

Детство, учёба

Джон Бэкус родился 3 декабря 1924 года в Филадельфии. Его отец был главным инженером-химиком в компании по производству нитроглицерина. Бэкус-старший начинал свою карьеру как простой служащий, но потом продвинулся по служебной лестнице во время Первой мировой войны, когда благодаря его техническим изобретениям была предотвращена серия взрывов на заводах.

Джон рос в Уилмингтоне (Делавэр), окончил школу Хилл в Поттстауне, закончил школу в Вашингтоне, затем осенью 1942 года поступил в Виргинский университет на химический факультет. Юноша не подавал никаких надежд в плане дальнейших успехов. Педагоги сомневались, что он мог принести хоть какую-то пользу обществу. Как позднее вспоминал он сам, несмотря на то, что родители определили его в престижную школу, процесс обучения вызывал у него отвращение. Когда Бэкус поступил в университет, единственным занятием, которое он посещал раз в неделю, был урок музыки. К концу второго семестра, в 1943 году, руководство университета решило, что дальнейшее пребывание юноши в стенах учебного заведения излишне: Бэкус был отчислен.

Служба

В это время была в разгаре Вторая мировая война, и США уже полтора года вели боевые действия против Японской империи. Молодой Джон Бэкус в чине капрала

был принят в дружные ряды тихоокеанской ПВО, однако в боевых действиях Бэкус участия не принимал: при медосмотре у него обнаружили опухоль черепной кости, которая была успешно оперирована с установкой костного имплантата.

Продолжение учёбы

После службы в армии попал в Нью-Йорк. Перед ним вновь встал вопрос выбора профессии. Ничто, кроме музыки, его не привлекало. Поскольку ему сильно хотелось иметь хорошую аппаратуру для прослушивания музыки, он поступил в школу радиотехники: получив там необходимые навыки, он смог бы самостоятельно сконструировать музыкальную аппаратуру.

Однажды один преподаватель попросил Джона Бэкуса помочь ему с построением графиков частотных характеристик усилителя. Вычисления были несложными, но их обилие утомляло. Неожиданно эти повторяющиеся математические операции заинтересовали Бэкуса. Преподаватель по ремонту теле- и радиоаппаратуры пробудил в нем интерес к математике и убедил его продолжить образование в Колумбийском университете.

Весной 1949 года, за несколько месяцев до получения диплома, 25-летний Джон Бэкус по-прежнему не мог определиться со своим будущим. В поисках перспективной работы его занесло в Компьютерный центр IBM на Мэдисон Авеню. Как оказалось, это была «судьбоносная случайность».

Получив степень магистра математики в Колумбийском университете, по приглашению Рекса Сибера — одного из изобретателей машины SSEC (Selective Sequence Electronic Calculator) – Бэкус поступил на работу программистом в фирму IBM.

Работа в IBM

В 1950 году Джон был принят на работу в IBM, где позже возглавил группу, разрабатывающую интерпретатор Speedcoding. Он ускорил кодирование для компьютера IBM 701, а позже участвовал в создании улучшенной версии этой машины, модели IBM 704. В 1953 году он предложил создать язык для этого компьютера, позволяющий записывать команды почти в обычной алгебраической форме, и компилятор для него. В результате в 1954-1957 годах был создан язык Фортран, общепризнанный как первый язык программирования высокого уровня в истории. В 1963 году Джон стал почетным сотрудником IBM.

Тогда слово «компьютер» (computer) обозначало совсем не то, что оно обозначает сегодня. Selective Sequence Electronic Calculator (SSEC) был одной из первых разработок IBM в новой области электронных вычислительных устройств на вакуумных лампах. Этот, так сказать, компьютер не имел памяти, а весь ввод и

вывод происходил посредством перфолент. Так что, компьютерами были в каком-то смысле и люди, которые «помогали» машинам.

В проекте SSEC Джона Бэкуса подключили к решению задачи вычисления положения Луны в 12-часовом интервале на 200-летнем отрезке времени. Для вычисления каждой позиции Луны требовалось произвести 11 тысяч сложений и вычитаний, 9 тысяч умножений и выполнить 2 тысячи просмотров специальных таблиц. Это задача требовала напряженной работы 13 тысяч огромных вакуумных ламп SSEC. Различные узлы SSEC располагались вдоль трех стен комнаты размером 18 на 6 метров, а программисты находились как бы внутри компьютера.

Вычисления, выполненные, в том числе и Бэкусом, на SSEC использовались в космической навигации НАСА в период с 1952 по 1971 годы. Позднее ему приходилось выполнять и более экзотические задачи — например, вычисление точных орбит пяти внешних планет Солнечной системы на временном интервале с 1653 по 2060 годы.

SSEC морально устарел уже в 1952 году и был демонтирован.

Fortan

Как уже говорилось, в 1954 году компания IBM запустила новый проект – IBM 704. В отличие от ламповой ЭВМ 701, новый проект был электронно-магнитным. IBM 704 предоставила программистам универсальный набор команд для работы, в том числе с числами с плавающей запятой.

Реализовывать на языке ассемблера алгоритмы обработки чисел с плавающей запятой нелегко. А программировать в ту пору приходилось в основном только математические формулы, и никаких математических сопроцессоров не было. В конечном итоге «лентяй» Джон Бэкус (как он впоследствии шутливо, а может, и не совсем шутливо вспоминал) стал все больше задумываться над тем, как создать язык, независимый от архитектуры машины и позволяющий легко программировать математические формулы.

Джон Бэкус обратился к своему шефу Кутберту Хэрду с предложением разработать практический язык программирования высокого уровня и компилятор для машины IBM 704. Предложение Бэкуса было одобрено.

Fortran родился в штаб-квартире фирмы IBM на Мэдисон-авеню в Нью-Йорке — в офисе на 19 этаже, где расположилась группа Бэкуса. Сначала с Бэкусом работал только Ирвинг Циллер, потом присоединился Харлан Херрик с командой математиков и техников.

Бэкус собрал группу из девяти дипломированных специалистов по математике, и

они с энтузиазмом принялись за работу по созданию нового языка. Эти девять человек позднее вошли в историю программирования – Роберт Нельсон (Robert Nelson), Харлан Херрик (Harlan Herrick), Льюис Хэйт (Lois Haibt), Рой Нат (Roy Nutt), Ирвинг Циллер (Irving Ziller), Шелдон Бест (Sheldon Best), Дэвид Сэйр (David Sayre), Ричард Голдберг (Richard Goldberg) и Питер Шеридан (Peter Sheridan).
В своих воспоминаниях Бэкус пишет, что из-за свойственной ему лени он создал такую систему управления группой, что ему, собственно, и делать-то ничего не приходилось. Наибольшую сложность представляла для него только задача, как заставить членов группы не тратить столько времени на игры.

Разработчики не надеялись, что их язык программирования будет когда-либо использоваться на машинах, отличных от IBM 704, но они были уверены, что в случае успеха их работа существенно повлияет на ИТ-индустрию.

Они выделили основные понятия нового языка, в частности оператор присваивания (например, N = 100), задававший переменным определенные значения, ввели индексируемые переменные, которые сообщали компьютеру, какой элемент из списка переменных нужен (например, X(3) означает третий элемент списка, названного X), предложили очень важный оператор DO, который позволял повторять нужную последовательность операторов заданное число раз.

Текст программы стал более простым, появились конструкции IF для организации ветвлений и циклы. Невероятно упростилось программирование формул. Например, формула D=B^2-4AC, программирование которой даже на современном языке ассемблера потребует десяток строк кода, на новом языке записывалась следующим образом: D=B**2-4*A*C.

Как заявлял Бекус, люди ошибочно полагали, что основной вклад Fortran — это возможность писать программы в виде алгебраических формул, а не на машинном языке. По его мнению, Fortran в первую очередь автоматизировал организацию циклов. Важность этой задачи при разработке научных приложений сложно переоценить.

Работа над языком шла быстро, чего нельзя было сказать о разработке компилятора. Бекус понимал, что развеять сомнения в возможностях «автоматического» программирования, то есть написания программ на языках высокого уровня, нелегко. Программы на Fortran должны быть такими же быстродействующими и надежными, как и написанные в машинных кодах или на языке ассемблера.

Из трех лет, затраченных на разработку проекта в целом, более двух лет заняла работа над компилятором. Если первое сообщение о создании языка группа сделала в 1954 году, то о разработке компилятора — только в апреле 1957 года.

Разработчики долго не могли определиться с названием нового языка. Бэкус придумал несколько банальных названий, но каждый раз они не устраивали коллег. Но однажды он предложил FORTRAN (FORmula TRANSlation). Реакция была аналогичной, но это название было принято из-за отсутствия лучшего.

Пользователи признали FORTRAN очень неохотно. Как вспоминает Бэкус, программисты "весьма скептически воспринимали все наши заявки". Однако в дальнейшем многие поняли его простоту в обучении и использовании и он стал незаменимым языком для научных и инженерных приложений.

Благодаря эффективности языка и его способности переиспользовать старый код многие поколения студентов-физиков до сих пор создают программы на Fortran на основе старых программ и фрагментов кода своих научных руководителей.

Переписывать на другой язык качественный и отполированный за десятки лет код — пустая трата денег, тем более когда у всех есть проблемы с настоящим техдолгом. Поэтому многие продолжают использовать Fortran в вычислениях — там он пока незаменим.

Много лет этот язык использовали инженеры, и на нём написано множество лучших инженерных программ. Например, программы для решения задач с матрицами (matrix solvers) используются при конструировании самолётов, зданий, автомобилей и так далее. К тому же у Fortran очень лаконичный и простой встроенный синтаксис для распределённого параллельного программирования.

За 60 лет пользователи написали под Fortran множество программ, библиотек, алгоритмов и подпрограмм для любых вычислений. В арсенале «дедушки» — работа с комплексными числами, операции с матрицами и даже Coarray для параллельных вычислений.

Вот примеры важных для научного сообщества программ на Fortran:

- CHARMM — молекулярная динамика.
- Code_Saturne — вычислительная гидродинамика.
- NEMO — океанография.
- QUANTUMESPRESSO — моделирование материалов.
- SPECFEM3D — распространение сейсмических волн.

- WRF — прогнозирование погоды.

В основном Fortran используют учёные — для численного моделирования. Отчасти это связано с традицией, отчасти с тем, что синтаксис языка позволяет сделать многие операции куда быстрее и проще, чем на других языках. Например, когда дело доходит до работы с массивами, матрицами или комплексными числами. Ещё одна причина — часть самых важных библиотек линейной алгебры (LAPACK, BLAS и ARPACK) изначально были написаны на Fortran и, кажется, лучше всего оптимизированы именно под него.

Fortran — отличный язык для разработки приложений, причём не только научных. Его функции позволяют разрабатывать программное обеспечение на более высоком уровне абстракции, чем в Си-подобных языках. А значит, разработчик может фокусироваться на создании приложения, а не погружаться в низкоуровневые вопросы и бороться с ограничениями языка. При этом скорость исполнения остаётся космической.

Поэтому Fortran до сих пор используют — даже в правительстве США. Например, в 2017 году NASA провело конкурс по улучшению своего кода на Fortran.

На Fortran можно писать программы с графическим интерфейсом (GUI), но это довольно неудобный процесс, потому что почти все библиотеки для разработки GUI написаны без учёта потребностей программистов на Fortran. Предполагалось, что программисты будут писать приложения на Си, C++, C # или чем-то подобном. Соглашения о вызове процедур в Fortran сильно отличаются от таковых в Си-подобных языках, поэтому приходится вносить много ручных корректировок, чтобы вызывать процедуры в GUI-библиотеках.

Fortran — первый язык со своим стандартом ANSI. Кстати, ANSI входит в ISO (International Organization for Standardization), который регулярно обновляет спецификации языка, — а значит, он и не собирается умирать.

Fortran имеет свои особенности, которые выгодно отличают его среди современных языков программирования. Он был первой компилируемой системой высокого уровня, ориентированной на применение в области науки и техники. Более того, язык программирования Фортран продолжает успешно развиваться, что подтверждает его актуальность, даже при наличии систем, которые появились намного позже. Может показаться, что сейчас нет особых запросов по его использованию, но он все еще имеет большой список преимуществ.

Достоинства языка программирования Fortran:

- Очень простой и доступный вариант языка для обучения программированию. Система отличается понятным синтаксисом, а ее исторический опыт применения будет полезен для тех, кто только начинает знакомиться с разработкой программ.
- Освоив азы Fortran, будет не сложно разобраться в других языках.
- Система имеет обширный набор инструментов, доступных на бесплатной основе, поэтому не придется заморачиваться с лицензиями.
- Благодаря распространенности во всем мире, Фортран имеет объемную библиотеку и большой ассортимент прикладных приложений, которые созданы за его длинную историю.
- Эта система стандартизирована для разных платформ, а ее новые версия отличаются совместимостью с более ранними вариантами.
- Язык программирования Fortran имеет набор средств (трансляторов) для преобразования в машинные системы разных компьютеров.
- Практически на всех ПК присутствует компилятор Фортран, поэтому у пользователей всегда есть возможности для сложных параллельных вычислений.
- С помощью этой системы можно формировать компактные и эффективные программные коды, чем и была обусловлена ее востребованность в то время, когда ЭВМ еще не отличались особой производительностью.
- Язык программирования Fortran полезно изучать тем, кто получает высшее образование по техническим и, особенно, физико-математическим дисциплинам.

Стоит отметить, что на всех этапах своего развития Фортран, несмотря на свою надежность и востребованность, не выделялся особой продвинутостью в сфере технологий написания программ.

Основные минусы языка программирования Fortran:

- Жесткие требования к формату кода. К примеру, версия Фортран 77 имеет ограничения по длине строки, а в ее начале нужно было делать отступ. Такие условия делали неудобной работу даже при использовании перфокарт, не говоря уже о мониторе ПК. Позже, начиная со стандарта Fortran 90, эти ограничения смягчили.

- Небольшой набор команд для управления программной структурой. К примеру, очень сложно было писать программные приложения, при отсутствии оператора GOTO.
- Слабый набор средств для описания данных.

Фортран остается довольно высоким в научной и технической сферах. Этот язык программирования применяют для анализа прогнозов погоды, сейсмической активности, в молекулярной динамике и океанографии. Это эффективная система обработки и представления данных в цифровой форме, которая зарождалась еще в те времена, когда только стали появляться первые калькуляторы для массового потребителя.

Нельзя не отметить, что некоторая часть популярности Fortran обусловлена его историей. Длительное время он развивался, не испытывая острой конкуренции, что позволило сформировать масштабную клиентскую базу, обширные библиотеки и надстройки. Так как новые версии этого языка поддерживали предшествующие, то у инженеров, ученых и других потребителей языка Фортран не возникало мысли об отказе от его применения.

Algol

Другим важным достижением Бэкуса стало применение способа формальной записи — БНФ (Бэкуса — Наура форма) для описания языка Algol.

В 1958 году Джон Бэкус решил принять активное участие в обсуждении нового языка (он впоследствии и получил название Algol) в Цюрихе. Однако возникла проблема — английский язык, на котором изъяснялся Бэкус, был мало понятен швейцарским программистам. В связи с этим для описания конструкций языка были применены специальные диаграммы, которые Бэкус разработал совместно с датским астрономом и программистом Питером Науром. С тех пор форма Бэкуса-Наура (Backus-Naur Form — BNF) стала неким «эсперанто» мирового программирования. Программисту, владеющему BNF, для знакомства с новым языком не нужно изучать объемные труды с его словесным описанием: достаточно изучить BNF этого языка.

В феврале 1959 года Бэкус убедил влиятельную организацию SHARE (куда входили пользователи компьютеров фирмы IBM) сделать ставку на новый язык программирования. После этого организация настоятельно порекомендовала IBM реализовать Algol.

На словах компания пошла навстречу своим клиентам, а на деле работа в этом направлении почти не велась. В то время IBM была мировым лидером на рынке компьютеров и активно внедряла Fortran, что вполне ожидаемо. У членов SHARE, на самом деле, тоже не было ясной позиции. Когда их энтузиазм угас, они как ни в чем не бывало продолжали поддерживать Fortran.

А Бэкус, несмотря на неудачу, продолжал активно продвигать Algol.

В процессе подготовки отчёта об Алголе он разработал специальную систему определений, формально описывающую синтаксис языка программирования, которую представил на конференции ЮНЕСКО в Париже в 1959 году. Вскоре этот способ записи назвали нормальной формой Бэкуса; позже Петер Наур внёс уточнения в нотацию, и форму стали называть формой Бэкуса — Наура (при этом сокращение осталось прежним — БНФ).

FP

В 70-е годы Джон Бэкус совместно с Джоном Уильямсом и Эдвардом Уимерсом он разработал новый язык программирования FP.

FP — это так называемый чистый функциональный язык программирования, в котором программист сосредотачивается не на переменных и их значениях, а на «черных ящиках» — функциях, имеющих вход и выход.

Бэкус представил миру новый язык в 1977 году в статье под названием «Can Programming Be Liberated from the von Neumann style? A Functional Style and its Algebra of Programs» («Возможно ли освободить программирование от стиля фон Неймана? Алгебра программ в функциональном стиле»).

FP был задуман скорее как математическая модель, чем как средство разработки ПО. Он является каноническим примером языка, использующим функциональную парадигму.

В FP особое внимание уделено точному семантическому описанию и обработке абстрактных типов данных. Также вместе со своими сотрудниками Бэкус разработал оптимизирующий компилятор, использующий алгебраические преобразования.

Идеи, использованные в языке FP, были использованы при создании языка LISP.

Лекция

Джон Бэкус провел исследование парадигмы программирования на функциональном уровне, представив свои результаты в своей влиятельной лекции на премию Тьюринга 1977 года "Можно ли освободить программирование от стиля фон Неймана?"

Аннотация:

Обычные языки программирования становятся все более огромными, но не сильнее. Врожденные дефекты на самом базовом уровне делают их одновременно толстыми и слабыми: их примитивный стиль программирования "слово за слово", унаследованный от их общего предка - компьютера фон Неймана, их тесная связь семантики с переходами состояний, их разделение программирования на мир выражений и мирзаявлений, их неспособность эффективно использовать мощные комбинирующие формы для построения новых программ из существующих и отсутствие у них полезных математических свойств для рассуждений о программах. Альтернативный функциональный стиль программирования основан на использовании комбинирования форм для создания программ. Функциональные программы имеют дело со структурированными данными, часто являются неповторяющимися и нерекурсивными, построены иерархически, не называют свои аргументы и не требуют сложного механизма объявлений процедур, чтобы стать общеприменимыми. Комбинирование форм может использовать программы высокого уровня для создания программ еще более высокого уровня в стиле, невозможном на обычных языках. С функциональным стилем программирования связана алгебра программ, переменные которых варьируются по программам, а операции представляют собой комбинирование форм. Эту алгебру можно использовать для преобразования программ и решения уравнений, “неизвестными” которых являются программы, во многом таким же образом, как преобразуются уравнения в алгебре средней школы. Эти преобразования задаются алгебраическими законами и выполняются на том же языке, на котором написаны программы. Комбинирующие формы выбираются не только из-за их программной способности, но и из-за силы связанных с ними алгебраических законов. Общие теоремы алгебры дают подробные условия поведения и завершения для больших классов программ. Новый класс вычислительных систем использует стиль функционального программирования как в своем языке программирования, так и в правилах перехода состояний. В отличие от языков фон Неймана, эти системы имеют семантику, слабо связанную с состояниями — за одно основное вычисление происходит только один переход состояния.

Награды и почести

Джон назван стипендиатом IBM (1963), награжден премией У.У. Макдауэлла (1967), 1975 году награждён Национальной научной медалью США. В 1977 году за труды по созданию Фортрана и вклад по формализации специфицирования языков программирования награждён премией Тьюринга. Она была вручена человеку, который создал первый высокоуровневый язык программирования для научных и технических применений. Более официально, премия была присуждена Джону Бэкусу"…за глубокий и важный вклад в создание практических систем программирования высокого уровня, в особенности за работы по FORTRAN и частые публикации формальных процедур для спецификации языков программирования".
Также есть такие награды: Член Американской академии искусств и наук (1985), почетная степень Университета Анри Пуанкаре (1989), премия Дрейпера (1993), стипендиальная премия Музея компьютерной истории "за разработку FORTRAN, вклад в теорию компьютерных систем и управление программными проектами" (1997).
Астероид 6830 Джон Бэкус назван в его честь (1 июня 2007)

Последние годы жизни

До выхода на пенсию в 1991 году он работал в исследовательских лабораториях IBM. Подписал «Предупреждение учёных человечеству» (1992). Последние годы провёл с семьёй в Ашленде (Орегон).

Джон Бэкус ушел из жизни 17 марта 2007 года. Ему было 82. Несмотря на то, что в начале пути он не знал, что делать со своей жизнью, судьба приняла это решение за него.

Заключение

Данный учёный внес большой вклад в развитие языков программирования. Несмотря ни на что он шел вперед и добивался своего, ему будут благодарны за это еще много лет, ведь он принес пользу не только в информатику, но и в физику и математику. Разработанный им язык программирования Fortran стал незаменимым для научных и инженерных приложений. Я рада, что узнала о таком великом человеке.

Источники

1. Биография [Электронный ресурс] /. – Электронные текстовые данные – Режим доступа : https://habr.com/ru/post/317526/ свободный. Дата

посещения: 08.01.2023
2. Биография [Электронный ресурс] /. – Электронные текстовые данные – Режим доступа : http://db4.sbras.ru/elbib/data/show_page.phtml?22+294 свободный. Дата посещения: 08.01.2023
3. Биография [Электронный ресурс] /. – Электронные текстовые данные – Режим доступа : https://history.wikireading.ru/387569 свободный. Дата посещения: 08.01.2023
4. Биография [Электронный ресурс] /. – Электронные текстовые данные – Режим доступа : http://www.physics.uni-altai.ru/community/wiki/DzhonBjekus свободный. Дата посещения: 08.01.2023
5. Биография [Электронный ресурс] /. – Электронные текстовые данные – Режим доступа : https://military-history.fandom.com/wiki/Main_Page свободный. Дата посещения: 08.01.2023
6. Биография [Электронный ресурс] /. – Электронные текстовые данные – Режим доступа : https://skillbox.ru/media/code/drevneyshiy-yazyk-programmirovaniya-vosstal/ свободный. Дата посещения: 08.01.2023
7. Фортран [Электронный ресурс] /. – Электронные текстовые данные – Режим доступа : https://habr.com/ru/post/400523/ свободный. Дата посещения: 08.01.2023
8. Фортран [Электронный ресурс] /. – Электронные текстовые данные – Режим доступа : https://ru.wikipedia.org/wiki/Фортран свободный. Дата посещения: 08.01.2023
9. Фортран [Электронный ресурс] /. – Электронные текстовые данные – Режим доступа : https://gb.ru/blog/fortran/ свободный. Дата посещения: 08.01.2023