Введение

Программирование можно рассматривать как искусство, науку, ремесло. Программирование — это искусство получения ответов от машины. Для этого в узком смысле нужно составить специальный код для технического устройства, а в широком — разработать программы на языках программирования, т. е. не просто составить код, а выполнить интеллектуальную работу по составлению высокоразумных программ для решения различных задач во всех сферах человеческой деятельности.

Программирование — процесс описания последовательности действий решения задачи средствами конкретного языка программирования и оформление результатов описания в виде программы. Эта работа требует точности, аккуратности и терпения. Команды машине должны формулироваться абсолютно четко и полно, не должны содержать никакой двусмысленности.

На начальном этапе составлением программ для ЭВМ занимались сами изготовители вычислительных машин. Постепенно, с развитием техники, этот процесс из рутинной работы превратился в интеллектуальную деятельность, сравнимую с искусством, т. к. трудоемкое, ручное составление программ было подобно решению сложных комбинационных задач, которое требовало научных знаний и мастерства. Возникла потребность в людях со специальной подготовкой и особым складом ума, которых называют программистами. Овладев необходимыми знаниями, научившись грамотно и творчески применять их в повседневной работе, программист может стать незаменимым специалистом в своей области деятельности. Отмечается, что «программист должен обладать способностью первоклассного математика к абстрактному и логическому мышлению в сочетании с эдисоновским талантом сооружать все что угодно из О и 1. Он должен сочетать аккуратность бухгалтера с проницательностью разведчика, фантазию автора детективов с трезвой практичностью экономиста».

Программист — одна из самых востребованных специальностей в современном обществе.

С 1970—1980-х гг. программирование как новая научная дисциплина занимается методами разработки программных продуктов. Оно включает комплекс вопросов, связанных с написанием специ­фикаций, проектированием, кодированием, тестированием и функ­ционированием программ для ЭВМ. Для разработки программного обеспечения применяются следующие методы: математические, ин­женерных расчетов и управления.

Уровень программирования определяется четырьмя взаимосвя­занными факторами развития: возможностями компьютеров, теори­ей и языками, искусством и технологией программирования.

ля непосредственного решения задач программист должен:

* осознать задачу;
* составить план общего решения;
* выполнить план, т. е. преобразовать его в определенную по­следовательность действий;
* проверить результат решения, убедиться в его правильности. Чтобы все это выполнить, специалист должен многое знать и уметь.

История и этапы развития программирования

С глубокой древности известны попытки создать устройства, ус­коряющие и облегчающие процесс вычислений. Еще древние греки и римляне применяли приспособление, подобное счетам, — абак. Такие устройства были известны и в странах Древнего Востока. В XVM в. немецкие ученые В. Шиккард (1623), Г.Лейбниц (1673) и французский ученый Б. Паскаль (1642) создали механические вы­числительные устройства — предшественники всем известного арифмометра. Вычислительные машины совершенствовались в те­чение нескольких веков. Но при этом не применялось понятие «программа и программирование».

В 1854 г. английский математик Джордж Буль опубликовал кни­гу «Законы мышления», в которой развил алгебру высказываний —Булеву алгебру. На ее основе в начале 80-х гг. XIX в. построена тео­рия релейно-контактных схем и конструирования сложных дискрет­ных автоматов. Алгебра логики оказала многогранное влияние на развитие вычислительной техники, являясь инструментом разработ­ки и анализа сложных схем, инструментом оптимизации большого числа логических элементов, из многих тысяч которых состоит со­временная ЭВМ.

В СССР первыми авторами ЭВМ, изобретенной в декабре 1948 г., являются И. С. Брук и Б. И. Рамеев. А первая советская ЭВМ с сохраняющейся программой создана в 1951 г. под руково­дством С. А Лебедева (МЭСМ — малая электронная счетная маши­на). В 1953 г. в Советском Союзе начался серийный выпуск машин, первыми их которых были БЭСМ-1, «Стрела».

До конца 1950-х гг. ЭВМ основным элементом конструкции были электронные лампы (1-е поколение). В этот период развитие идеологии и техники программирования шло за счет достижений американских ученых Дж. фон Неймана, сформулировавшего ос­новные принципы построения ЭВМ, и Дж. Бэкуса, под руково­дством которого в 1954 г. был создан Fortran (Formula Translation) — первый язык программирования высокого уровня, используемый до настоящего времени в разных модификациях. Так, в 1965 г. в Дартмутском колледже Д. Кэмэни и Т. Куртцем была разработана упро­щенная версия Фортрана — Basic. В 1966 г. комиссия при Амери­канской ассоциации стандартов (ASA) разработала два стандарта языка: Фортран и Базисный Фортран. Используются также даль­нейшие модификации языка (например 1970, 1990 гг.).

В США в 1954 г. стал применяться алгебраический подход, совпадающий, по существу, с операторным методом. В 1956 г. корпора­цией IBM разработана универсальная ПП Фортран для автоматического программирования на ЭВМ IBM/704.

ООП основано на трех важнейших принципах (инкапсуляция, наследование, полиморфизм), придающих объектам новые свойства. Инкапсуляция — объединение в единое целое данных и алгоритмов их обработки. Данные здесь — поля объекта, а алгоритмы — объектные методы. Наследование — свойство объектов порождать своих потомков. Объект-потомок автоматически наследует все поля и методы, может дополнять объекты новыми полями, заменять и дополнять методы. Полиморфизм — свойство родственных объектов решать схожие по смыслу проблемы разными способами.

На современном этапе развиваются инструментальные среды и системы визуального программирования для создания программ на языках высокого уровня: (Turbo Pascal, Delphi, Visual Basic, C++Builder и др.).

В КП компонент рассматривается как хранилище (в виде DLL-или ЕХЕ файлов) для одного или нескольких классов. Классы рас­пространяются в бинарном виде, а не в виде исходного кода. Пре­доставление доступа к методам класса осуществляется через строго определенные интерфейсы по протоколу. Это снимает проблему несовместимости компиляторов, обеспечивая без перекомпиляции смену версий классов в разных приложениях. Интерфейсы задают содержание сервиса и являются посредником между клиентом и сервером.

Вместо этого эволюция пошла по пути машинного языка. Он был удобен для компьютеров, но не очень удобен для программистов. Это была буквально работа с железом вручную: были лишь те команды, которые были зашиты в процессор, все остальные операции приходилось реализовывать вручную.

Прямым развитием машинного языка стал язык ассемблера. Это был первый размен скорости на удобство: ассемблер был чуть менее нативным, но гораздо более дружелюбным к программисту. До языков высокого уровня, комфортных для программиста, было еще далеко.

Качественный скачок произошёл после изобретения транзистора и первых доступных компьютеров. Это побудило в период с 1954-1957 году компанию IBM активно работать над разработкой популярного коммерчески языка Fortran (от англ. Formula и Translation).

Fortran — язык для научных и инженерных вычислений. Считается, что именно он стал первым реализованным языком высокого уровня. Благодаря большому количеству математических библиотек Fortran был и остается важным языком для разного рода научных изысканий.

Далее началось бурное развитие программирования: практически одновременно появились Algol, алгоритмический высокоуровневый язык, активно использовавшийся в СССР и Европе, LISP и COBOL.

Уже сейчас появляются языки программирования, которые можно назвать языками сверхвысокого уровня. Они предназначены не для написания прикладных программ, а для обработки больших массивов данных и статистики. В их числе R, MATLAB и Julia.

Можно предполагать, что в будущем появятся человекоориентированные языки, на которых написать программу будет не сложнее, чем пост в ЖЖ. С другой стороны, квантовые компьютеры потребуют свой особый, квантовый ассемблер.

Классификация языков программирования и сред

разработки

Языки программирования высокого уровня

В языках высокого уровня особенности конкретных компьютерных архитектур не учитываются, поэтому написанные программы легко могут быть перенесены на другой компьютер. Зачастую достаточным является компиляция программы под определенную архитектурную и операционную систему. Разработка программ на языках высокого уровня значительно проще и ошибок намного меньше. К тому же время разработки программы значительно уменьшается, что является особенно важным фактором при работе над сложными программными проектами.

Недостаток некоторых языков высокого уровня состоит в большом размере программ по сравнению с программами на языках низкого уровня. В то же время текст программ на языке высокого уровня гораздо меньше, но в байтах код, написанный на ассемблере, будет более компактным.

Поэтому языки высокого уровня преимущественно используют для создания программного обеспечения для компьютеров и вычислительных устройств с большим объемом памяти. Языки же низкого уровня используются для написания программ к устройств, для которых критичным является размер программы.

Объектно-ориентированные языки

Объектно-ориентированные языки стали дальнейшим уровнем развития процедурных языков, основной концепцией которых есть совокупность программных объектов. Написание программы на языке представляется в виде последовательности создания экземпляров объектов и использование их методов. К ним относятся из первых языков Simula и SmallTalk, далее C++, Java.

Декларативные языки программирования

В декларативном программировании задается спецификация решения задачи, то есть дается описание того, что представляет собой проблема и какой ожидается результат. Программы, созданные с помощью декларативного языка, не содержат переменные и операторы присваивания.

К декларативным языкам можно отнести SQL и HTML. К подвидам декларативного программирования относится функциональное и логическое программирование.

Функциональные языки программирования

Функциональные языки являются языками искусственного интеллекта. Программа, написанная на функциональном языке, состоит из последовательности функций и выражений, которые необходимо вычислить. Основной структурой данных является связный список. Функциональное программирование принципиально отличается от процедурного. Основными функциональными языками являются Lisp, Miranda, Haskel.

Логические языки программирования

Языки, ориентированные на решение задач без описания алгоритмов, языки искусственного интеллекта. Представителем логического программирования является Prolog, которым написано большинство экспертных систем.

сред разработки и сопровождения ПС

* инструментальные среды программирования,
* рабочие места компьютерной технологии,
* инструментальные системы технологии программирования

Инструментальная среда программирования предназначена в основном для поддержки процессов программирования (кодирования), тестирования и отладки ПС. Она не обладает рассмотренными выше свойствами комплексности, ориентированности на конкретную технологию программирования, ориентированности на коллективную разработку и, как правило, свойством интегрированности, хотя имеется некоторая тенденция к созданию интегрированных сред программирования (в этом случае их следовало бы называть системами программирования). Иногда среда программирования может обладать свойством специализированности. Признак же ориентированности на конкретный язык программирования может иметь разные значения, что существенно используется для дальнейшей классификации сред программирования.

Рабочее место компьютерной технологии ориентировано на поддержку ранних этапов разработки ПС (системного анализа и спецификаций) и автоматической генерации программ по спецификациям [16.1, 16.4]. Оно существенно использует свойства специализированности, ориентированности на конкретную технологию программирования и, как правило, интегрированности. Более поздние рабочие места компьютерной технологии обладают также свойством комплексности ([16.4]). Что же касается языковой ориентированности, то вместо языков программирования они ориентированы на те или иные формальные языки спецификаций. Свойством ориентированности на коллективную разработку указанные рабочие места в настоящее время, как правило, не обладают.

Инструментальная система технологии программирования предназначена для поддержки всех процессов разработки и сопровождения в течение всего жизненного цикла ПС и ориентирована на коллективную разработку больших программных систем с продолжительным жизненным циклом. Обязательными свойствами ее являются комплексность, ориентированность на коллективную разработку и интегрированность. Кроме того, она или обладает технологической определенностью или получает это свойство в процессе расширения (настройки). Значение признака языковой ориентированности может быть различным, что используется для дальнейшей классификации этих систем.

Анализ предметной области «Магазин»