УВАЖАЕМЫЕ СТУДЕНТЫ! Законспектируйте в своей рабочей тетради по дисциплине приведенную лекцию (объемом 4-5 страницы), ответьте *письменно* на контрольные вопросы.

Результаты работы, фотоотчет, предоставить преподавателю на e-mail: r.bigangel@gmail.com до 16.01.2023.

При возникновении вопросов по приведенному материалу обращаться по следующему номеру телефона: (072)111-37-59, (Viber, WhatsApp), vk.com: https://vk.com/daykini

ВНИМАНИЕ!!! При отправке работы, не забывайте указывать ФИО студента, наименование дисциплины, дата проведения занятия (по расписанию).

Лекция № 28
Тема «Системы программирования. Сравнительная характеристика, примеры использования»

План лекции:

1. Состав систем программирования
2. Языки программирования
3. Машинно-ориентированные системы программирования
4. Машинно-независимые системы программирования
5. Интерпретаторы и компиляторы

Неотъемлемой частью современных ЭВМ являются системы программного обеспечения, которые являются средствами, расширяющими возможности аппаратуры и сферу ее использования. Эти системы являются посредником между человеком и вычислительной машиной, автоматизируют выполнение определенных функций в соответствии с профилем специалистов и режимами их взаимодействия с ЭВМ. Программное обеспечение повышает эффективность труда пользователя. Программное обеспечение подразделяют на общее и специальное. Общее программное обеспечение служит для реализации функций, связанных с работой ЭВМ. Оно состоит из операционной системы, системы программирования, программ технического

обслуживания. Специальное программное обеспечение состоит из прикладных программ, проблемно ориентированных на решение определенных задач.

Состав систем программирования

Системы программирования представляют комплексы инструментальных программных средств для работы с программами на определенном языке программирования.

Используя подобные системы, программисты имеют возможность разрабатывать свои собственные компьютерные программы. Системы программирования состоят из: трансляторов с языков высокого уровня; редактирующих и компонующих средств, а также средств загрузки программ; макроассемблеров (машинно-ориентированных языков); отладчиков машинных программ.

Языки программирования

Язык программирования составляет ядро системы программирования. Они могут быть процедурными и непроцедурными.

Процедурные (или алгоритмические) программы – это системы предписаний для решения определенных задач.

Компьютер лишь механически выполняет эти предписания.

Процедурные языки могут быть представлены языками низкого и высокого уровня.

С использованием языков низкого уровня (машинно-ориентированных) создаются программы в машинных кодах. С такими языками тяжело работать, однако созданные на них программы малы по объему и быстродейственны.

Используя языки программирования низкого уровня, разрабатывают системные программы, драйвера и др.

Программы, созданные на языках высокого уровня, представляют собой наборы заданных команд, которые близки по своему звучанию к естественному (английскому) языку.

К наиболее известным процедурным системам программирования относят:

– Fortran, один из старейших и по сей день используемых в решении задач математической ориентации язык

– Basic, являющийся универсальным символическим кодом инструкций для начинающих пользователей, самый популярный среди пользователей.

– ALGOL, представляющий собой алгоритмический язык, сыгравший большую роль в теории, в настоящее время практически не используется.

– PL/1 - многоцелевой язык, который в настоящее время не используется.

– Си – широко используемый язык при создании систем программного обеспечения. Pascal – чрезвычайно популярный язык как среди новичков в программировании, так и среди профессионалов. На его основе созданы более мощные языки такие, как Ada, Delphi.

– COBOL – язык, ориентированный на общий бизнес, сейчас практически не используется.

– Delphi – очень популярный объективно-ориентированный язык визуального программирования.

– Java – платформенно независимый язык объективно-ориентированного программирования, эффективен при создании интерактивных web-страниц.

Среди непроцедурных языков программирования наиболее известны:

– Lips;

– PROLOG.

Машинно-ориентированные системы программирования

По уровню формализации входного языка, целевому назначению и структуре системы программирования делят на: машинно-ориентированные и машинно-независимые.

Машинно-ориентированные состоят из входного языка, наборов операторов и изобразительных средств.

Для систем подобного типа характерны:

– высокое качество созданных программ;

– предсказуемость заказов памяти и объектного кода;

– использование конкретных аппаратных ресурсов;

– необходимость знания системы команд и особенностей функционирования конкретной ЭВМ;

– низкая скорость программирования; трудоемкость процесса программирования;

– невозможность непосредственного использования программ, составленных на этих языках, на компьютерах других типов.

По степени автоматического программирования машинно-ориентированные системы подразделяют на классы:

Машинный язык. В системе такого типа отдельный компьютер обладает своим определенным машинным языком, которому предписывается выполнение операций над операндами. Этот язык является командным. Система символического кодирования. В системах такого типа используют языки символического кодирования, являющиеся командными. Коды операций и адреса в машинных командах в языках символьного кодирования заменены символами (идентификаторами), формы написания которых помогают легче запоминать программисту смысловое содержание операции. Это способствует существенному уменьшению числа ошибок при составлении программ.

Автокоды. Содержат все возможности языков символического кодирования через процесс расширенного введения макрокоманд. В различных программах часто встречаются некоторые используемые командные последовательности, соответствующие определенным процедурам преобразования информации. Эти последовательности оформляют в виде специальных макрокоманд, которые затем можно использовать в языке программирования при написании программ.

Макрокоманды переводятся в машинные команды 2 способами: расстановкой и генерированием.

В первом способе используются «остовы» – серии команд реализации требуемой функции, обозначенной макрокомандой. Макрокоманды передают фактические параметры, вставляемые в процессе трансляции в «остов» программы, преобразуя ее в реальную машинную программу. Системы с генерацией содержат специальные программы анализа макрокоманд, определяющие какую функцию нужно выполнить и формирующие последовательности команд, реализующих эту функцию. Обе системы используют трансляторы с языка символьного кодирования и наборы макрокоманд, являющиеся операторами автокода.

Макросы. Представляют собой более сжатую форму записи, используемую для замены последовательности символов описания выполнения требуемых действий ЭВМ. Предназначены для сокращения записи исходных программ. Компонент программного обеспечения, с помощью которого обеспечивается функционирование макросов, называют макропроцессором. На него поступает макросопределяющий и исходный тексты. Реакцией макропроцессора на вызов является выдача выходного текста.

Машинно-независимые системы программирования

Эти системы программирования являются средством описания алгоритмов решения задач и обрабатываемой информации.

Их удобно использовать широкому кругу пользователей, поскольку не требуется знаний особенностей организации функционирования ЭВМ. Машинно-независимые системы программирования подразделяют на:

Процедурно-ориентированные системы. В этих системах входные языки программирования предназначены для записи при решении задач алгоритмов обработки информации. Эти языки обеспечивают программиста средствами четкого формулирования задач и получения результатов в требуемой форме.

Проблемно-ориентированные системы используют в качестве входного языка язык программирования с проблемной ориентацией. Языки подобного типа обеспечивают программиста средствами короткой и четкой формулировки задач и средствами получения результатов в требуемой форме. Программы на этих языках программирования записываются в терминах решаемой задачи и реализуются через выполнение определенных процедур.

Диалоговые языки. Обеспечивают оперативное взаимодействие пользователя с компьютером через сохранение в его памяти копии исходной программы в машинных кодах. В процессе изменений в программе система программирования устанавливает с помощью специальных таблиц взаимосвязь между структурами исходной и объектной программ, что дает возможность в дальнейшем редактировать объектную программу.

Непроцедурные языки. Составляют группу языков, с помощью которых описывается организация обрабатываемых данных и языков связи с операционными системами. Являются табличными языками, позволяющими четко описывать как задачу, так и ее решения в наглядной форме. В одной таблице решений, описывающей некоторую ситуацию, содержатся все возможные блок-схемы реализаций алгоритмов решения.

Интерпретаторы и компиляторы

Компилятор прежде, чем запустить программу на выполнение полностью обрабатывает ее текст:

- выполняет поиск синтаксических ошибок;
- делает смысловой анализ;
- автоматически генерирует машинный код.

Далее сгенерированный объектный код обрабатывается специальной программой — сборщиком или редактором связей. В результате текст программы преобразовывается в готовый к исполнению файл, он сохраняется в памяти компьютера или на диске.

Этот файл может самостоятельно работать под управлением операционной системы. Интерпретатор используется для анализа очередного оператора языка из текста программы и запуска его на исполнение.

Перейти к выполнению следующего оператора интерпретатор может только после успешного выполнения текущего. При многократном выполнении одного и того же оператора интерпретатор каждый раз выполняет его так, будто впервые. В результате программы, содержащие большие объемы повторяющихся вычислений, работают медленно.

К основным недостаткам компиляторов можно отнести трудоемкость трансляции языков программирования, ориентированных на обработку данных сложной структуры. Используя интерпретатор, наоборот, можно остановить работу программы в любой момент, организовать диалог с пользователем, исследовать содержимое памяти, выполнить любые сложные преобразования данных и при этом постоянно осуществлять контроль за состоянием окружающей программно-аппаратной среды, благодаря чему достигают высокой надежности работы. Интерпретаторы удобно использовать при изучении программирования, так как они дают возможность понять механизм работы каждого оператора языка в отдельности.

Контрольные вопросы:

1. Состав систем программирования

2. Языки программирования

3. Машинно-ориентированные системы программирования

4. Машинно-независимые системы программирования

5. Интерпретаторы и компиляторы