Информатика 8-классы

20.04.20

Задание:

Прочитать и ответить на вопросы:

1. Что такое программа?
2. Что такое языки программирования?
3. Перечислите основные этапы развития языков программирования?
4. Что такое транслятор?
5. Запишите два основных способа трансляции

Срок сдачи: до 23.04.20

Краткая история языков программирования

Программа. Язык программирования

*Программу* можно представить как набор последовательных команд (алгоритм) для объекта (исполнителя), который должен их выполнить для достижения определённой цели. Так можно "запрограммировать" человека, если составить для него инструкцию "как приготовить оладьи", а он начнёт чётко ее исполнять. При этом инструкция (программа) для человека будет написана на так называемом естественном языке, например, русском или английском.

Обычно принято программировать не людей, а вычислительные машины, используя при этом специальные языки. Использование особых языков вызвано тем, что машины не в состоянии "понимать" наши, т. е. человеческие, языки. Инструкции для машин пишут на *языках программирования*, которые характеризуются синтаксической однозначностью (например, в них нельзя менять местами определенные слова) и ограниченностью (имеют строго определенный набор слов и символов).

Программа – это набор инструкций для определённого исполнителя.

Язык программирования - это формальный язык, предназначенный для записи программ(обычно для ЭВМ)

*Основные этапы исторического развития языков программирования*

Первые программы писались на **машинном языке**, т.к. для ЭВМ того времени еще не существовало развитого программного обеспечения, а машинный язык — это единственный способ взаимодействия с аппаратным обеспечением компьютера, так называемым "железом". Каждую команду машинного языка напрямую выполняет то или иное электронное устройство. Данные и команды записывали в цифровом виде (например, в шестнадцатеричной или двоичной системах счисления). Понять программу на таком языке очень сложно;

кроме того, даже небольшая программа получалась состоящей из множества строк кода. Ситуация осложнялась еще и тем, что каждая вычислительная машина понимает лишь свой машинный язык.

Людям, в отличие от машин, более понятны слова, чем наборы цифр. Стремление человека оперировать словами, а не цифрами привело к появлению **ассемблеров**. Это языки, в которых вместо численного обозначения команд и областей памяти используются словесно-буквенные.

При этом появляется проблема: машина не в состоянии понимать слова. Необходим какой-нибудь переводчик на ее родной машинный язык. Поэтому, начиная со времен ассемблеров, под каждый язык программирования создаются *трансляторы* — специальные программы, преобразующие программный код с языка программирования в машинный код. Ассемблеры на сегодняшний день продолжают использовать. В системном программировании с их помощью создаются низкоуровневые интерфейсы операционных систем, компоненты драйверов.

После ассемблеров наступил рассвет языков так называемого **высокого уровня**. Для этих языков потребовалось разрабатывать более сложные трансляторы, т. к. языки высокого уровня куда больше удобны для человека, чем для вычислительной машины. В отличие от ассемблеров, которые остаются привязанными к своим типам машин, языки высоко уровня обладают переносимостью. Это значит, что, написав один раз программу на языке программирования высокого уровня, программист может выполнить ее на любом компьютере, если на нем установлен соответствующий ему транслятор.

Следующим значимым шагом было появление **объектно-ориентированных языков** программирования, что, в первую очередь, было связано с усложнением разрабатываемых программ. С помощью таких языков программист как бы управляет виртуальными объектами, что в определенном смысле сближает программу с реальностью. На сегодняшний день в большинстве случаев реализация больших и сложных проектов осуществляется с помощью объектно-ориентированных возможностей языков. Хотя существуют и другие современные парадигмы программирования, поддерживаемые другими или теми же языками.



Разнообразие языков программирования

В настоящее время существует множество различающихся и похожих между собой языков программирования. Причина такого явления становится понятна, если представить то количество и разнообразие задач, которые на сегодняшний день решается с помощью вычислительной техники. Для решения разных задач требуются разные инструменты, т. е. разные языки и парадигмы программирования.

Многие программисты старались в прошлом и возможно пытаются сейчас придумать свой язык программирования, обладающий теми или иными преимуществами. Хотя подавляющее большинство в настоящее время тратят огромное количество времени на изучение уже существующего арсенала инструментов, поддержку и развитие имеющихся языков.

Все существующее многообразие языков можно классифицировать по разным критериям. Например, по типу решаемых задач (языки системного или прикладного назначения, языки для web-разработки, организации баз данных, разработки мобильных приложений и др.). Среди наиболее популярных на сегодняшний день можно отметить Java, C++, PHP, в том числе Python, изучению базовых основ которого посвящен данный курс.

Трансляция

Ранее было сказано, что для перевода кода с языка программирования высокого уровня на машинный язык требуется специальная программа – транслятор (англ. translator — переводчик).

Заложенный в транслятор алгоритм такого перевода сложен. Нам же достаточно знать, что выделяют два основных способа трансляции — **компиляция** программы или ее **интерпретация**.

При компиляции весь *исходный программный код* (тот, который пишет программист) сразу переводится в машинный. Создается так называемый отдельный *исполняемый файл*, который никак не связан с исходным кодом.

Выполнение исполняемого файла обеспечивается операционной системой (ОС). После того как получен исполняемый файл, для его чтения транслятор уже не нужен.

При интерпретации выполнение кода происходит последовательно (условно можно сказать, строка за строкой). Грубо говоря, операционная система взаимодействует с интерпретатором, а не с файлом, содержащим программный код. Интерпретатор же, прочитав очередной кусок исходного кода, переводит его в машинный (или не совсем машинный, но "понятный" для ОС) и "отдает" его ОС. ОС исполняет этот код и ждет следующей "подачки" от интерпретатора. Питон именно такой язык. Он интерпретируемый язык программирования.

Выполнение откомпилированной программы происходит быстрее, т.к. она представляет собой готовый машинный код. Однако на современных компьютерах

снижение скорости выполнения при интерпретации обычно не заметно. Кроме того, интерпретируемые языки обладают рядом преимуществ, среди которых отсутствие подготовительных действий для исполнения программы, что может быть важным для начинающих программировать в первый раз.

