УВАЖАЕМЫЕ СТУДЕНТЫ!

ВАМ НЕОБХОДИМО ВЫПОЛНИТЬ СЛЕДУЮЩЕЕ:

1. Ознакомиться с теорией и законспектировать лекцию не меньше трех листов, составить и ответить на вопросы.

2. Предоставит отчет конспекта лекции прислать в виде скриншото в течении трех дней. тел. 072-17-44-9-22.

3. Отправить преподавателю на почту **v.vika2014@mail.ru** и указать свою Ф.И.О, группу, и название дисциплины

Тема: **Этапы решения задач на ЭВМ. Данные и величины**

Этапы решения задач на ЭВМ

Условно ***программированием*** можно назвать научную и практическую деятельность по созданию программ. Основной частью программирования является процесс решения задачи на ЭВМ, который можно разбить на следующие этапы:

1) математическая или информационная формулировка задачи;

2) выбор численного или иного метода решения поставленной задачи;

3) построение алгоритма решения поставленной задачи;

4) выбор языка программирования и запись построенного алгоритма по его правилам, т.е. написание текста программы;

5) отладка программы – это процесс обнаружения, локализации и устранения возможных ошибок;

6) выполнение программы, т.е. получение требуемого результата.

Рассмотрим более подробно некоторые наиболее важные из приведенных этапов.

Понятие алгоритма

Понятие алгоритма занимает центральное место в современной математике и программировании.

Алгоритмизация – сведение задачи к последовательным этапам действий так, что результаты предыдущих действий используются при выполнении последующих.

Рассмотрим вначале некоторые наиболее важные (фундаментальные) понятия программирования.

1. ***Действие*** – это некоторая операция, имеющая конкретную продолжительность и приводящая к совершенно конкретному результату.

2. Каждое действие предполагает наличие некоторых данных, над которыми это действие совершается и по изменению состояния которых определяют результат этого действия.

3. Каждое действие должно быть таким, чтобы его можно было описать при помощи какого-либо языка (или набора формул); такое описание называют ***инструкция***.

4. Если действие можно разложить на составные части, то его называют ***процессом*** (или вычислением).

5. Описание характера проведения процесса, т.е. последовательности выполняемых действий без привязки к какому-то конкретному процессору, называют ***алгоритмом.***

Числовой алгоритм – детально описанный способ преобразования числовых входных данных в выходные при помощи математических операций. Существуют нечисловые алгоритмы, которые используются в экономике, технике и научных исследованиях.

В общем, ***алгоритм*** – строгий и четкий набор правил, определяющий последовательность действий, приводящих к достижению поставленной цели.

Свойства алгоритмов

Дискретность – значения новых величин (данных) вычисляются по определенным правилам из других величин с уже известными значениями.

Определенность(детерминированность)– каждое правило из набора однозначно, а сами данные однозначно связаны между собой, т.е. последовательность действий алгоритма строго и точно определена.

Результативность (конечность) – алгоритм решает поставленную задачу за конечное число шагов.

Массовость – алгоритм разрабатывается так, чтобы его можно было применить для целого класса задач, например, алгоритм вычисления определенных интегралов с заданной точностью.

Сложность алгоритма

Выполнение любого алгоритма требует определенного объема памяти компьютера для размещения данных и программы, а также времени по обработке этих данных – эти ресурсы ограничены и, следовательно, правомочен вопрос об эффективности их использования. Таким образом, в самом широком смысле понятие эффективности связано со всеми вычислительными ресурсами, необходимыми для работы алгоритма.

Однако обычно под *«самым эффективным»* понимается алгоритм, обеспечивающий наиболее быстрое получение результата, поэтому рассмотрим именно временну́ю сложность алгоритмов.

Время работы алгоритма удобно выражать в виде функции от одной переменной, характеризующей *«размер»* конкретной задачи, т.е. объем входных данных, необходимых для ее решения. Тогда сравнительная сложность задач и может оцениваться через ее размер.

Поскольку описание задачи, предназначенной для решения посредством вычислительного устройства, можно рассматривать в виде слова конечной длины, представленной символами конечного алфавита, в качестве формальной характеристики размера задачи можно принять длину входного слова. Например, если стоит задача определения максимального числа в некоторой последовательности из n элементов, то и размер задачи будет n, поскольку любой вариант входной последовательности можно задать словом из n символов.

Временная сложность алгоритма – это функция, которая каждой входной длине слова n ставит в соответствие *максимальное* (для всех конкретных задач длиной n) время, затрачиваемое алгоритмом на ее решение.

Различные алгоритмы имеют различную временную сложность и выяснение того, какие из них окажутся *достаточно эффективны*, а какие нет, определяется многими факторами. Однако для сравнения эффективности алгоритмов был предложен простой подход, позволяющий прояснить ситуацию. Речь идет о различии между *полиномиальными* и *экспоненциальными* алгоритмами.

Полиномиальным называется алгоритм, временнàя сложность которого выражается некоторой полиномиальной функцией размера задачи n. Алгоритмы, временнáя сложность которых не поддается подобной оценке, называются *экспоненциальными*.

Задача считается *труднорешаемой*, если для нее не удается построить полиномиального алгоритма. Это утверждение не является категорическим, поскольку известны задачи, в которых достаточно эффективно работают и экспоненциальные алгоритмы. Примером может служить симплекс-метод, который успешно используется при решении задач линейного программирования, имея функцию сложности $f(n) = 2^n$. Однако подобных примеров не очень много, и общей следует признать ситуацию, когда эффективно исполняемыми можно считать полиномиальные алгоритмы с функциями сложности n, n^2 или n^3.

Например, при решении задачи поиска нужного элемента из n имеющихся в худшем варианте сложность равна n; если же оценить среднюю трудоемкость (продолжительность поиска), то она составит $(n+1)/2$ – в обоих случаях функция сложности оказывается линейной n.

Сложность задачи вычисления определителя системы n линейных уравнений с n неизвестными характеризуется полиномом 3-й степени.

Повышение быстродействия элементов компьютера уменьшает время исполнения алгоритма, но не уменьшает степень полинома сложности.