Качество программы и ее разработка

Высокое качество программы достигается, в первую очередь, за счет глубокой проработки схемы алгоритма на этапе проектирования. Это, прежде всего, безошибочность программы, уверенность программиста в том, что она не содержит ошибок, и уверенность пользователя в том, что она правильна.

Уверенность в безошибочности программы определяется ясностью и простотой, читаемостью и легкостью интерпретации ее автором и пользователями, поскольку ошибки в программе могут выявляться в процессе ее создания и эксплуатации. Шансы сделать ошибки уменьшаются, если при разработке создатели программы будут стремиться к тому, чтобы она была понятной другим людям.

Хотя при программировании невозможно избежать ошибок, рекомендуется придерживаться некоторых правил составления программы, которые позволяют быстро обнаружить и устранить ошибки. К числу правил хорошего стиля программирования относят следующие.

1. Необходимо стремиться к наиболее полному изучению поставленной задачи при формулировке задачи, т.е. разработке математической модели. Такое изучение позволяет добиться ясной, предусматривающей множество логических взаимодействий объектов программы. Недостаточно глубокая проработка математической модели приводит к неправильным результатам решения задачи, если модель неприменима к тем классам объектов, для расчета параметров которых она используется в данной программе, или если исходные данные не учитывают особенностей данной модели, а модель, в свою очередь, не учитывает всех взаимодействий между данными.

2. Проработка алгоритма решения задачи связана с возможно более полным учетом общих особенностей процесса вычислений на ЭВМ. Так как в ЭВМ не существует никакого другого внутреннего способа представления комплексного числа, кроме

представления в виде пары вещественных чисел, то попытка вычислить на ЭВМ корень четной степени из вещественного отрицательного числа вызывает прекращение вычислений и сообщение об ошибке. Алгоритм в подобных случаях должен предусматривать анализ знака значения подкоренного выражения и содержать две возможных ветви вычислений – для вещественного и мнимого значений корня. Другой пример связан с приближенным представлением вещественных чисел в ЭВМ. Нецелесообразно сравнивать значения двух вещественных выражений. Сравнение следует заменить проверкой соотношения $|A – B| < e$, где e – малое число; если это неравенство выполняется, то следует считать, что A = B.

Приближенное представление вещественных чисел в ЭВМ может привести и к тому, что вычислениях с числами разного порядка могут получиться неправильные результаты. Например, при вычислении с семью значащими цифрами для A = 199 и B = 0.0001 полученное значение $(A + B)^3 – A^3 = 0.1 * 10^2$ вместо числа $0.1188 * 10^2$; более точный результат получен по другой формуле: $B^3 + 3B^2A + 3A^2B$.

Погрешность результата будет меньшей, если начинать суммировать с меньших по величине чисел, а не с больших, как в первом случае.

3. При разработке алгоритма необходимо стремиться к максимальной простоте и понятности. Это относится как к содержательной стороне, так и к форме записи программы на языке программирования. Применение стандартных приемов структурного программирования делает программу более ясной, хотя в некоторых случаях более громоздкой и менее эффективной. Ясность и простота программы зачастую важнее, чем выигрыш в эффективности.

Хороший набор стандартов поможет сконцентрировать внимание на новых задачах. Рекомендуется программу сначала записать на каком-либо легко воспринимаемом языке, например на языке схем алгоритмов.