М. И. Жадан, Е. М. Березовская
г. Гомель, УО «ГГУ им. Ф. Скорины»

О ВОПРОСАХ ЧИТАЕМОСТИ АЛГОРИТМОВ В КУРСЕ «ИНФОРМАТИКА»

Вопросы читаемости программы и ее результатов являются важными в квалификационной характеристике программиста. Известно, что программа значительно реже пишется, чем читается. Тот, кто пробовал погрузиться даже в свою программу, написанную несколько месяцев (лет) назад, знает, что повторное ее чтение превращается в некоторую пытку, если проблеме чтения не было уделено должного внимания. Такую программу возможно легче написать с нуля учтя соответствующие изменения [1].

Легко читаемая программа позволяет читателю ее понять и, причем достаточно быстро. Ее наполняют нужной полезной избыточностью. Избыточность не является простым критерием читаемости и надо ее использовать умело. Следует отметить, что ключом к читаемости программы являются не нагромождения объяснений, а сама программа. Надо найти как можно простые и естественные формы выражения, позволяющие установить читателю непосредственное соответствие между текстом программы и тем, что происходит при ее выполнении. Всякий программист должен сосредоточить свое внимание именно на этом основном свойстве программы. Оценка стиля программы является субъективной, однако приведем некоторые принципы, применяемые в программировании [2]:

– самым трудным элементом при чтении программы, это статическое понимание динамической последовательности операторов. Необходимо

как можно реже (без особой надобности) использовать такие операторы как Goto, Break, Continue, Exit;

– использование осмысленных имен. Следует запомнить принцип – надо рассматривать имя как инвариант программы. Например, если в программе используется переменная с именем «Минимум_массива», то это должно означать, что значение переменной «Минимум_массива» является самым малым в рассматриваемом массиве. Следует также писать «Хкоордината» и «Укоордината», а не «координатаХ» и «координатаУ». Использование таких имен как «В», «ВВ», «ВВВ», «ВВВВ» является мазохистской практикой, поскольку возможна либо опечатка, либо ошибка;

– используя незначащие пробелы особенно удобно показывать динамическую структуру программы, т. е. необходимо использовать смещения для выделения управляющих структур и их размещения;

– использовать декомпозицию сложных программных модулей (например, содержащих более 10 имен или 20 строк выполняемой программы);

– использовать «полезные» комментарии:

а) указывающие на сложный фрагмент программы, использующей нетривиальную технику программирования;

б) для «входных» и «выходных» утверждений: аргументов, результатов, модифицированных данных;

в) при современной концепции программирования, основанной на поэтапном составлении программы, комментарии являются идеальным средством выражения, позволяющим в окончательной программе отразить эти этапы.

Общей чертой всех комментариев является то, что они не должны служить украшением программы, добавленными в нее искусственно, а быть составными частями программ, задуманными и построенными одновременно с этими программами. Только так можно избежать от бесполезных и противоречивых комментариев.

Посмотрим, как используются описанные выше методики составления программ в школьном курсе «Информатика» на примере учебника «Информатика-8» и соответствующей рабочей тетради.

Приведем некоторые примеры, которые говорят о практически полной не читаемости результатов приведенных в книгах программ. Приведенные протоколы работы программ не дают никакой возможности определить, какая все же задача была запрограммирована.

Пример 1. Результаты выполнения программы при вводе чисел 16 и 13 (с. 11):

```
Введите целое число:  16
8                    – целочисленное деление четного числа на 2
Введите целое число:  13
13
```

Пример 2. Результаты выполнения программы для чисел 5 и 7, а также для чисел 0 и 10 (с. 12):

Введите два целых числа: 5 7
6 – среднее арифметическое двух чисел,
не равных нулю
Введите два целых числа: 0 10
– нет никакого сообщения,
возможно машина повисла.

Пример 3. Результаты выполнения программы для чисел 27 и 347 (с. 16):

Введите целое число: 27 – только для двухзначных чисел
2 – количество десятков
7 – количество единиц
Введите целое число: 345
Не могу найти числа

Пример 4. Результат выполнения программы (с. 29):

Sum = 55 – сумма первых 10 натуральных чисел

Пример 5. Результат выполнения программы при $n = 30$ (с. 37):

n=30
6 слагаемых – количество слагаемых из четных чисел 2,4,…,
сумма которых меньше 30.

Пример 6. Результат выполнения программы при $a = 2$ (с. 47):

основание a=2 – введите целое число a < 5
наименьшее число n=6 – наименьшее n, для которого $a^n \geq 50$

Теперь приведем несколько примеров на неудачную читаемость протокола работы программы (результата), по которому практически невозможно узнать, о чем идет речь в программе, т. е. невозможно восстановить условие задачи. В тетради предлагаются задания по которым необходимо составить программы со следующими протоколами ее работы.

Пример 1. Пример ввода:
454
Пример вывода:
Является – число является перевертышем

Пример 2. Пример ввода:
3 10 2
Пример вывода:
Не могут – быть сторонами треугольника

Пример 3. Пример ввода:
-5 7
Пример вывода:
2 – точка лежит во 2 четверти

Пример 4. Пример ввода:
6 0
Пример вывода:
1 – точка лежит на оси координат

Пример 5. Пример ввода:
12
Пример вывода:
6 – количество делителей числа 12

Пример 6. Пример ввода:
7 5
Пример вывода:
2 – номер меньшего из 7 и 5.

Справа от примеров кратко указан смысл решаемых задач.

В книгах имеются хорошо оформленные программы и результаты с точки зрения читаемости, однако мы остановились на имеющихся, на наш взгляд, неприятных моментах. Хотелось бы верить, что в новых изданиях книг и пособий по информатике будут учитываться, изложенные выше методики составления читаемых программ и результатов.

Литература

1. Мейер, Б. Методы программирования : в 2-х томах. Т. 2 / Б. Мейер, К. Бодуэн. – М. : Мир, 1988. – 368 с.

2. Kerninghan, W. The Elements of Programming Style / W. Kerninghan, P. J. Plander. – McGraw-Hill, New York, 1974. – 420 p.