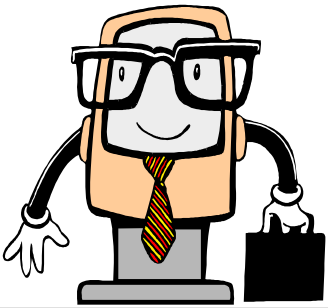

***М.С. Долинский**, к.т.н., доцент кафедры математических проблем управления Гомельского государственного университета им. Ф. Скорины*

Использование форума при обучении программированию первокурсников

Введение

Автор много лет занимается обучением программированию школьников разных возрастов и первокурсников математического факультета (специальности: «Программное обеспечение информационных технологий» и «Прикладная математика») [1-18]. В связи с исключением из основной школьной учебной программы основ алгоритмизации и программирования, большинство первокурсников знакомится с данным материалом только в университете. Это требует поиска новых более интенсивных способов обучения первокурсников. С осени 2009 года одним из способов обучения стало обсуждение решений индивидуальных задач на форуме в специальной теме «Учим друг друга, учимся друг у друга» [19]. Студент, решивший индивидуальную задачу, выкладывает условие, описание решения и исходный текст программы. Далее преподаватель и желающие студенты обсуждают увиденное, предлагают альтернативные и более эффективные варианты решения. В данной статье приводится пример такой работы над одной из задач. Такой материал может быть ин-

тересен для преподавателей как в качестве иллюстрации методики обучения, так и по содержанию. В частности, рассматриваются следующие вопросы: назначение и техника использования функций при кодировании алгоритмов, вложенные и сложные условия, способы повышения производительности программ, жизненный цикл программ, «читабельность» программ (форматирование и комментирование), компромиссы между производительностью и понятностью программ. В то же время, автору представляется, что этот материал может оказаться весьма полезным и интересным и для школьников и студентов, занимающихся самообучением. Всем заинтересованным предлагается следующий порядок работы: откладывать статью в сторону и пытаться самостоятельно выполнить предлагаемое задание после прочтения условия задачи, а также после каждого ответа преподавателя.

Первая попытка студента

Ссылка на задачу: COCI\coci_2007-08\4_January\1 – «VAUVAU»

Условие:

В одной деревне почтальон, молочник и мусорщик сталкивались с одной и той же проблемой каждое утро – дом 18. Дом 18 охранялся двумя собаками. Когда день начинается, первая собака агрессивна A минут, затем спокойна B минут. Аналогично, вторая собака агрессивна C минут, затем спокойна D минут. Обе собаки повторяют свое поведение бесконечно, начиная новый агрессивный период после спокойного периода и т.д.

Вам даны времена прибытия почтальона, молочника и мусорщика, определите как в этот момент собаки (ни одна, одна, обе) атакуют каждого из них.

Формат ввода:

Первая строка содержит четыре целых числа A B C D. Вторая строка содержит целые числа P M G – минута в дне, в которую к дому 18 прибывают соответственно почтальон, молочник и мусорщик. Все числа на вводе от 1 до 999.

Формат вывода:

Три строки, каждая из которых содержит одно из слов ‘both’,’one’, ‘none’, в зависимости от того, сколько собак

атакуют в момент прибытия почтальона, молочника и мусорщика.

Пример ввода:

2 2 3 3
1 3 4

Пример вывода:

both
one
none

Описание решения:

«Читаем все, что от нас требуется. Потом находим, кто придет последний. Потом начинаем циклы проверки для каждой собаки по отдельности. Проверяем в тех местах, где собака агрессивна, а там, где она спокойна, просто пропускаем. Если собака агрессивна, то к счетчику +1 и обрываем. Так же делаем для второй собаки. Потом выводим относительно условия, сколько собак и кого покусали.»

Исходный текст программы на Паскале:

```
var
   a,b,c,d,s1,s2,max,p,i : longint;
   l,k : array [1..3] of longint;
begin
   readln(a,b,c,d); //читаем переменные
   for i:= 1 to 3 do read(l[i]); //почт.,мус., мол., задаю массивом
   max:=l[1]; //нахожу время, когда придет последний
   if l[2]>max then max:=l[2];
   if l[3]>max then max:=l[3];
   for i:= 1 to 3 do k[i]:=0; //обнуляю счетчики
   for i:= 1 to 3 do //начинаем цикл проверки для первой собаки
      begin //промежуток между p и s1
         p:=0; //явл. промежутком агрессии
         s1:=0;
         while (s1<=max+a+b) do //выполняем цикл,пока не
                                // пройдет время
            begin //прихода последнее и +a+b(время агрес-
                  // сии и отдыха)
               s1:=p+a; //время окончания агресии
               if (s1>=l[i]) and (p<l[i]) then //если человек
                  //приходит во время агрессии первой собаки
                  begin
                     inc(k[i]); //к счетчику +1 и обрываем
                                //(уже покусали)
                     break;
```

```
                end;
            s1:=s1+b; //потом я пропускаю время отдыха собаки
            p:=s1; //оно нам не важно и не нужно
        end;
    end;
for i:= 1 to 3 do //аналогично, как и с прошлой собакой
    begin
        p:=0;
        s1:=0;
        while (s1<=max+c+d) do
            begin
                s1:=p+c;
                if (s1>=l[i]) and (p<=l[i]) then
                    begin
                        inc(k[i]);
                        break;
                    end;
                s1:=s1+d;
                p:=s1;
            end;
    end;
    for i:= 1 to 3 do //потом смотрим, сколько собак покусало
        if k[i]=0 then writeln('none') else // молочника, почталь-
                                            //она и мусорщика
            if k[i]=1 then writeln('one') else writeln('both');
end.
```

Первый ответ преподавателя

По-моему, все ответы НЕЗАВИСИМЫ, то есть, ответ для почтальона не зависит от ответов для молочника и мусорщика и т.д. Поэтому решение может быть написано существенно проще.

Конкурс на 10 бонусов – написать более простое решение для этой задачи.

Подсказка: главная программа может выглядеть так:

```
readln(a,b,c,d);
readln(P,M,G);
writeln(Answer(P));
writeln(Answer(M));
writeln(Answer(G));
```

И фактически Ваша задача – написать функцию Answer, которую главная программа вызывает три раза (для времен прибытия почтальона, молочника, мусорщика).

Вторая попытка студента

Решение: Читаем все переменные, пользуемся функцией. В функции я проверяю, во время того, как пришел один из служащих, активны были собаки или нет.

Код:

```
var
   a,b,c,d,p,m,g : longint;
function Answer(k : longint) : string;
   var
      r,ac1,ac2,p1,p2 : longint;
   begin
      ac1:=0;
      ac2:=0;
      p1:=0;
      p2:=0;
      r:=0;
      while (ac1<=k) or (ac2<=k) do
         begin
            ac1:=p1+a;
            ac2:=p2+c;
            if ((ac1>=k) and (p1<k)) then inc(r);
            if ((ac2>=k) and (p2<k)) then inc(r);
            ac1:=ac1+b;
            ac2:=ac2+d;
            p1:=ac1;
            p2:=ac2;
         end;
      if r=0 then Answer:='none' else
      if r=1 then Answer:='one' else Answer:='both';
   end;
begin
   readln(a,b,c,d);
   readln(p,m,g);
   writeln(Answer(p));
   writeln(Answer(m));
   writeln(Answer(g));
end.
```

Так лучше?

Второй ответ преподавателя

Так намного лучше, по-моему. Осталось только объяснить ПОДРОБНЕЕ, как именно работает функция, и каков смысл введенных в ней переменных. А сам код я бы предпочел написать так:

```
var
    a,b,c,d,p,m,g : longint;
function Answer(k : longint) : string;
    var
        r,ac1,ac2,p1,p2 : longint;
    begin
        ac1:=0; ac2:=0; p1:=0; p2:=0; r:=0;
        while (ac1<=k) or (ac2<=k) do
            begin
                ac1:=p1+a; ac2:=p2+c;
                if ((ac1>=k) and (p1<k)) then inc(r);
                if ((ac2>=k) and (p2<k)) then inc(r);
                ac1:=ac1+b; ac2:=ac2+d;
                p1:=ac1; p2:=ac2;
            end;
        if r=0 then Answer:='none';
        if r=1 then Answer:='one';
        if r=2 then Answer:='both';
    end;
begin
    readln(a,b,c,d);
    readln(p,m,g);
    writeln(Answer(p));
    writeln(Answer(m));
    writeln(Answer(g));
end.
```

Третья попытка студента

Откомментированное решение:

```
var
a,b,c,d,p,m,g : longint;
    function Answer(k : longint) : string; //в функцию вводим время
            //прибытия служащего, выводим строку(one,none,both)
        var
            r,ac1,ac2,p1,p2 : longint;
        begin
            ac1:=0; ac2:=0; p1:=0; p2:=0; //переменные времени
                                    //активности, пассивности собак
            r:=0; //и счетчик
            while (ac1<=k) or (ac2<=k) do //делаем, пока актив-
                                        //ности собак будут ниже, чем
                begin                   // время прибытия
                    ac1:=p1+a; //время завершения активности
                               //первой собаки (p1 – начало)
                    ac2:=p2+c; //аналогично со второй
                    if ((ac1>=k) and (p1<k)) then inc(r);
```

```
                    //если время прибытия служащего в
                    //то время, когда собака активна,
                    //к счетчику +1(аналогично с другой собакой)
                if ((ac2>=k) and (p2<k)) then inc(r);
                ac1:=ac1+b;    //время окончания пассивности
                               //первой собаки
                ac2:=ac2+d;    //(ан. со второй)
                p1:=ac1; //начало периода агрессии первой собаки
                p2:=ac2; // (ан. со второй)
              end;
          if r=0 then Answer:='none'; //проверяем, сколько собак
                                      //были активны в момент
          if r=1 then Answer:='one'; //прихода служащего и
                                      //выводим соответственно
          if r=2 then Answer:='both';
       end;
begin
   readln(a,b,c,d); //читаем переменные
   readln(p,m,g);
   writeln(Answer(p)); //выводим значения функции в данной
                       //переменной
   writeln(Answer(m));
   writeln(Answer(g));
end.
```

Вопрос к вам, Михаил Семенович, вы посоветовали написать условия вот так:

```
if r=0 then Answer:='none';
if r=1 then Answer:='one';
if r=2 then Answer:='both';
```

а я написал вот так:

```
if r=0 then Answer:='none' else
if r=1 then Answer:='one' else Answer:='both';
```

Не могу спорить, но в моем случае, если найдется верное условие раньше, то условия в дальнейшем проверять перестанет, это же выгоднее тем, что количество выполняемых процедур сократится, разве не так?

Реплика другого студента

Выскажу свое мнение... Операторов тут не так уж и много, и если выполнятся три, а не один из них (в лучшем случае), то сомневаюсь, что возможно будет почувствовать, хоть какую-то разницу во времени выполнения или памяти. А вот такой код:

```
if r=0 then Answer:='none';
if r=1 then Answer:='one';
if r=2 then Answer:='both';
```

визуально гораздо лучше воспринимается. Я, например, с первого взгляда понял, что в нем происходит. А к твоему варианту:

```
if r=0 then Answer:='none' else
  if r=1 then Answer:='one' else Answer:='both';
```

надо дольше присматриваться

Реплика преподавателя

Спорить – в смысле выяснять истину – можно и даже нужно. Сегодня, с моей точки зрения, основное требование к программам – «читабельность» и понятность. Поскольку программы имеют долгий жизненный цикл, и приходится их читать и понимать для того, чтобы исправлять ошибки и добавлять новую функциональность.

Скорость работы самой программы, по большей части, менее критична. Твой вариант выбора ответа работает действительно чуть быстрее. Мой вариант – понятнее.

Но так как этот оператор выполняется НЕ В ЦИКЛЕ, а всего один раз, то и быстрота там будет НЕОТЛИЧИМАЯ.

Реплика первого студента

Спасибо вам большое обоим. Просто сталкиваясь с задачами USACO, я понял, что надо все упрощать и делать быстрее. А насчет «читабельности» я не спорю. Скажем так, я готовлюсь к преодолению нового барьера, который заключается в достижении быстроты работы программы.

Третий ответ преподавателя

Не надо ЖЕРТВОВАТЬ одним ради другого. Возможно, и нужно писать программы, которые будут обладать ОБОИМИ достоинствами. Возвращаясь к сути задачи (и борясь за ее реальное ускорение). Мне кажется, что в данном случае ее можно решить БЕЗ цикла, прямым вычислением.

Конкурс на 10 бонусов – написать такую программу.

Четвертая попытка студента

```
var
    a,b,c,d,p,m,g : longint;
    function Answer(k : longint) : string;
        var
            r : longint;
        begin
            r:=0;
            if k<=(a+b) then
                begin
                    if (a>=k) then inc(r);
                end
            else
                if k mod (a+b) <= a then inc(r);
            if k<=(c+d) then
                begin
                    if (c>=k) then inc(r);
                end
            else
                if k mod (c+d) <= c then inc(r);
            if r=0 then Answer:='none';
            if r=1 then Answer:='one';
            if r=2 then Answer:='both';
        end;
begin
    readln(a,b,c,d);
    readln(p,m,g);
    writeln(Answer(p));
    writeln(Answer(m));
    writeln(Answer(g));
end.
```

Четвертый ответ преподавателя

Женя, а давай вместо вложенных IF-ов ты напишешь СЛОЖНОЕ условие

```
if (...) and (...) then
```

заодно повыкидывав лишние begin, end, а по возможности и else.

Пятая попытка студента

```
var
    a,b,c,d,p,m,g : longint;
    function Answer(k : longint) : string;
        var
```

```
        r : longint;
    begin
        r:=0;
        if ((k<=(a+b)) and (a>=k)) or (k mod (a+b) <= a) then inc(r);
        if ((k<=(c+d)) and (c>=k)) or (k mod (c+d) <= c) then inc(r);
        if r=0 then Answer:='none';
        if r=1 then Answer:='one';
        if r=2 then Answer:='both';
    end;
begin
    readln(a,b,c,d);
    readln(p,m,g);
    writeln(Answer(p));
    writeln(Answer(m));
    writeln(Answer(g));
end.
```

Действительно, приятнее смотрится.

Пятый ответ преподавателя

Вот это уже похоже на код профессионала.

+20 бонусов.

Просматривал протокол, кто чем занимается, и обнаружил, что Саша тоже сдал эту задачу.

Заглянул в решение и вот что увидел

```
var
    a,b,c,d,p,m,g:longint;
    function Answer(k:longint):string;
        var
            n,s1,s2:longint;
        begin
            n:=0;
            s1:=k mod (a+b);
            s2:=k mod (c+d);
            if (s1<>0) and (s1<=a) then inc(n);
            if (s2<>0) and (s2<=c) then inc(n);
            if n=0 then Answer:='none';
            if n=1 then Answer:='one';
            if n=2 then Answer:='both';
        end;
begin
    readln(a,b,c,d);
    readln(p,m,g);
    writeln(Answer(p));
    writeln(Answer(m));
```

```
    writeln(Answer(g));
end.
```

По-моему, проще и понятнее, чем у Жени. +20 бонусов Саше, ПО-12.

Заключение

В данной статье проведена иллюстрация обучения программированию первокурсников с помощью предметного форума. В частности, при обсуждении решения данной задачи затронуты такие важные понятия, как жизненный цикл и производительность программы, «читабельность» исходных текстов и их грамотное комментирование, разделение задачи на подзадачи, использование функций, сложные условия. Технической основой методики является разработанная инструментальная система дистанционного обучения (Distance Learning Belarus – http://dl.gsu.by). Внедрение данного способа обучения первокурсников с осени 2009 года обеспечило значительные сдвиги качества обучения. Положительным моментом такого подхода является также накопление ссылок на подобные обсуждения [19], что позволяет использовать материалы при обучении следующих поколений студентов.

Литература

1. Долинский, М.С. Об опыте подготовки школьников Гомельской области к республиканским и международным олимпиадам по информатике / М.С. Долинский // Информатизация образования. – 2009. – № 1(54). – С. 29-40.

2. Долинский, М.С. Система интернет-курсов дифференцированного обучения программированию школьников и студентов / М.С. Долинский, М.А. Кугейко // Информатизация образования. – 2010. – № 1(58). – С. 58-68.

3. Долинский, М.С. Как учить думать школьников и студентов? / М.С. Долинский, М.А. Кугейко // Информатизация образования. – 2010. – № 2(59). – С. 62-72.

4. Долинский, М.С. Технология развивающего дифференцированного обучения программированию младших школьников «с чистого листа» / М.С. Долинский, М.А. Ку-

гейко // Информатизация образования. – 2010. – № 3(60). – С. 12-20.

5. Долинский, М.С. Интернет-курс «Базовое программирование» как средство подготовки к областным олимпиадам по информатике / М.С. Долинский, М.А. Кугейко // Информатизация образования. – 2010. – № 4(61). – С. 3-15.

6. Долинский, М.С. Развитие мышления младших школьников на основе флеш-заданий на рисование, раскраску и конструирование в системе DL.GSU.BY / М.С. Долинский, Ю.В. Решетько, М.А. Кугейко // Информатизация образования. – 2011. – № 1(62). – С. 24-35.

7. Долинский, М.С. Какими должны быть задачи на олимпиадах по информатике / М.С. Долинский, М.А. Кугейко // Информатизация образования. – 2011. – № 1(62). – С. 68-76.

8. Долинский, М.С. Флеш-шаблоны для создания заданий развивающего обучения / М.С. Долинский, Ю.В. Решетько, М.А. Долинская // Информатизация образования. – 2011. – № 2(63). – С. 14-28.

9. Долинский, М.С. Конструирование интерактивных флеш-заданий на развитие мышления / М.С. Долинский, Ю.В. Решетько, М.А. Долинская // Информатизация образования. – 2011. – № 3(64). – С. 21-33.

10. Долинский, М.С. Конструирование интерактивных флеш-заданий на развитие мышления на базе произвольных картинок / М.С. Долинский, Ю.В. Решетько, М.А. Долинская // Информатизация образования. – 2011. – № 4(65). – С. 3-14.

11. Долинский, М.С. Конструирование интерактивных флеш-заданий на базе собственных танов / М.С. Долинский, Ю.В. Решетько, Н.С. Лебедько // Информатизация образования. – 2012. – № 1(66). – С. 24-34.

12. Долинский, М.С. Конструктор интерактивных флеш-заданий как открытая система для создания электронных учебных пособий / М.С. Долинский, Ю.В. Решетько, М.А. Долинская, Н.С. Лебедько // Информатизация образования. – 2012. – № 2(67). – С. 35-45.

13. Долинский, М.С. Электронное учебное пособие «Математика. Начальная школа» / М.С. Долинский, Ю.В. Ре-

шетько, Н.С.Лебедько // Информатизация образования. – 2012. – № 3(68). – С. 30-42.

14. Долинский, М.С. Создание электронных учебных пособий для вузовских дисциплин с помощью конструктора флеш-заданий / М.С. Долинский, Ю.В. Решетько // Информатизация образования. – 2012. – № 4(69). – С. 34-45.

15. Долинский, М.С. Интерактивная анимация в электронных учебных пособиях, создаваемых с помощью конструктора флеш-заданий / М.С. Долинский, Ю. В. Решетько, М.А. Долинская // Информатизация образования. – 2013. – № 1(70). – С. 30-38.

16. Долинский, М.С. Учебный интернет-курс и перманентный интернет-конкурс «Математика 1-8 кл.» / М.С. Долинский, Ю.В. Решетько, М.А. Долинская // Информатизация образования. – 2013. – № 2(71). – С. 38-47.

17. Долинский, М.С. Концептуальные основы и практика сквозного развивающего обучения информатике и программированию от детского сада до вуза / М.С. Долинский, Ю. В. Решетько, М.А. Долинская // Информатизация образования. – 2013. – № 3(72). – С. 16-25.

18. Долинский, М.С. Об одном подходе к обучению программированию на первом курсе / М.С. Долинский, М.А. Долинская // Информатизация образования. – 2014. – № 1(73). – С. 32-41.

19. Учим друг друга, учимся друг у друга – 2012 // Сайт дистанционного обучения ГГУ им. Ф.Скорины / Режим доступа: http://dl.gsu.by/NForum/ posts/topicshow/ 1647.dl?postid=35564#35564.

Статья поступила 30.06.2014