***П.В. Гляков**, к.ф.-м.н., доцент, заведующий кафедрой информационных технологий в культуре Белорусского государственного университета культуры и искусств*

# *Игровая форма обучения алгоритмизации культурологов-менеджеров*

*В статье показывается, как можно использовать игровую форму обучения для проведения занятий по алгоритмизации и программированию с будущими культурологами-менеджерами. Выявляются особенности игровой формы обучения, свидетельствующие о ее высокой результативности.*



## Введение

Учебный план подготовки культурологов-менеджеров Белорусского государственного университета культуры и искусств включает дисциплину «Алгоритмизация и программирование». Наиболее трудной частью этой дисциплины является алгоритмизация, которая содержит элементы теории сложности алгоритмов. Студенты должны научиться не только разрабатывать алгоритмы для решения задач, но и научиться оценивать их эффективность. В качестве вычислительной модели для оценки эффективности алгоритмов используется алгоритмический язык [1].

После того как на лекции рассмотрены такие вопросы, как размер задачи, временная сложность алгоритма, емкостная сложность алгоритма, единицы измерения времени и памяти алгоритма, на практических занятиях студенты приступают к вычислению характеристик времени и памяти алгоритма в худшем случае. При рассмотрении этой темы на практическом занятии оказалось целесообразным использовать игровую форму обучения.

## Разбиение группы на команды

При использовании игровой формы обучения группу делят на 3-4 команды. Это зависит от количества студентов в группе и возможности аудитории, в которой проводится занятие. Студенты одной команды должны иметь возможность общаться между собой, не мешая при этом студентам других команд.

В каждой команде должен быть капитан. Он может назначаться преподавателем или избираться студентами группы. В первом случае разбиение группы на команды осуществляется капитанами. Происходит это так. Каждый капитан поочередно набирает в свою команду по одному студенту. Так происходит до тех пор, пока все студенты не будут распределены. Такой подход позволяет по возможности учесть пожелания капитанов. Составляющими элементами этих пожеланий является следующее: уровень знания алгоритмизации, уровень коммуникабельности, личностная совместимость.

Во втором случае студенты каждой команды сами выбирают себе капитана. При этом они руководствуются в основном теми же критериями, как и в первом случае.

Работа студентов в командах имеет важное значение для будущих культурологов-менеджеров. Поскольку в своей профессиональной деятельности многие из них будут руководителями (менеджерами) разных звеньев. Иногда психологически важно при работе в командах напомнить студентам афоризм: «Никто из Вас не знает так много, как все Вы вместе».

### Задача последовательного поиска

После разбиения группы на команды преподаватель формулирует условие задачи, алгоритм решения которой надо построить и оценить его эффективность. Задача должна быть такая, чтобы допускала достаточную вариативность решения. Удобно для этой цели взять задачу последовательного поиска.

*Условие. Дан массив целых чисел* ***a*** *размерности* ***n****. Требуется определить, есть ли в нем элемент* $a_i$, *равный некоторому целому числу* ***b****. Если такой элемент есть, то строковой переменной* ***c*** *присвоить значение «да», в противном случае присвоить значение «нет».*

Когда команды разработают свои алгоритмы, капитаны команд выходят к доске и записывают алгоритмы решения задачи. Доска для этих целей должна быть стандартная с разворотами, чтобы студенты могли видеть решения всех команд. Иногда в представленных решениях могут быть ошибки, и студенты команды не видят их или не могут устранить. Тогда преподаватель подбирает такие исходные данные для исполнения алгоритма, на которых алгоритм не даст требуемого решения. Капитану и его команде предлагают исполнить алгоритм и устранить ошибку. Если команда не в состоянии устранить ошибку, преподаватель обращается к другим коман-

дам, чтобы они указали, как можно устранить ошибку, или сам предлагает собственный вариант устранения ошибки.

Иногда при решении задачи последовательного поиска капитаны представляют одинаковые алгоритмы. В этом случае больше получит та команда, которая представила свой алгоритм первой. Обычно алгоритмы, которые представляют команды, на алгоритмическом языке выглядят следующим образом.

**Алгоритм Поиск1.**

```
алг Поиск1(цел таб a[1:n], нат n, цел b, лит c)
   арг a, n, b
   рез c
нач цел i
   i:=1, c:= "нет"                2
   пока iJn                       n+1
   нц
      если a[i]=b                 n
         то c:= "да", i:=n+1      0
         иначе i:=i+1             n
      все
   кц
кон
```

Идея алгоритма Поиск1 заключается в том, что последовательно просматриваются элементы массива **a** и сравниваются с величиной **b**. Как только будет обнаружен элемент, равный **b**, цикл завершает свою работу, поскольку переменной **i**, используемой для организации цикла, присваивается значение, которое делает условие цикла ложным.

**Алгоритм Поиск2.**

```
алг Поиск2(цел таб a[1:n], нат n, цел b, лит c)
   арг a, n, b
   рез c
нач цел i
   i:=1, c:="нет"                 2
   пока iJn и c:="нет"            2(n+1)
   нц
      если a[i]=b                 n
         то c:="да"               0
         иначе i:=i+1             n
      все
   кц
кон
```

В алгоритме Поиск2 условие цикла является составным. В этом условии проверяется не только переменная **i**, которая используется для последовательного просмотра элементов

массива **a**, но и переменная **c**, которой предварительно установлено значение «нет». Как только будет найден искомый элемент, этой переменной будет присвоено значение «да» и цикл завершит свою работу.

**Алгоритм Поиск3.**

```
алг Поиск3(цел таб a[1:n], нат n, цел b, лит c)
   арг a, n, b
   рез c
нач цел i
   i:=1, c:="нет"                2
   пока iJn                      n+1
   нц
      если a[i]=b                n
         то c:="да"              0
      все
      i:=i+1                     n
   кц
кон
```

Алгоритм Поиск3 не завершает свою работу, как только найдет искомый элемент. Он будет осуществлять просмотр всех элементов массива **a** и каждый раз, когда будет встречаться элемент $\mathbf{a}_i$, равный **b**, переменной **c** будет присваиваться значение «да». Интуитивно становится понятным, что в среднем случае этот алгоритм будет работать дольше, чем алгоритмы Поиск1 и Поиск4.

**Алгоритм Поиск4.**

```
алг Поиск4(цел таб a[1:n], нат n, цел b, лит c)
   арг a, n, b
   рез c
нач цел i
   i:=1                          1
   пока iJn                      n+1
   нц
      если a[i]=b                n
         то i:=n+1               0
      все
      i:=i+1 n
   кц
   если i=n+2                    1
      то c:="да"                 0
      иначе c:="нет"             1
   все
кон
```

Цикл в алгоритме Поиск4 завершает свою работу сразу же после обнаружения искомого элемента массива **a**. Для

этой цели так же, как и в алгоритме Поиск1, переменной **i**, используемой в условии цикла, присваивается значение, которое делает условие цикла ложным. Отличие от алгоритма Поиск1 состоит в том, что значение переменной **c**, в которой формируется результат выполнения алгоритма, присваивается после завершения работы цикла.

## Характеристики алгоритмов

Вычислим характеристики времени T(n) и памяти M(n) для худшего случая, т. е. такого случая, когда алгоритм будет требовать при выполнении наибольшего количества времени. Такой случай соответствует ситуации, в которой массив **a** не содержит элемента $\mathbf{a}_i$, равного числу **b**.

С правой стороны строк приведенных алгоритмов Поиск1, Поиск2, Поиск3, Поиск4 мы указали, сколько раз выполняется либо проверка условия цикла и разветвления, либо оператор присваивания. Команды это должны сделать сами. Просуммировав эти значения, мы получаем соответственно следующие характеристики времени в худшем случае для приведенных алгоритмов: $T_1(n) = 3n + 3$, $T_2(n) = 4n + 4$, $T_3(n) = 3n + 3$, $T_4(n) = 3n + 4$. Для всех четырех алгоритмов память, требуемая для выполнения алгоритма в худшем случае, будет одинаковой: $M(n) = n + 4$. В эту величину входит память, требуемая для размещения исходных данных, результата и вспомогательной переменной **i**.

Из вычисленных характеристик времени выполнения алгоритма в худшем случае видно, что при достаточно большом размере задачи **n** алгоритм Поиск2 будет выполняться дольше остальных в 4/3 раза:

$$\lim_{n \to \infty} \frac{4n+4}{3n+4} = \frac{4}{3}.$$

Как правило, командам не удается найти более быстрый алгоритм для решения задачи последовательного поиска. Поэтому преподаватель сам знакомит студентов с идеей алгоритма для выполнения быстрого последовательного поиска (БПП) и сам записывает на доске алгоритм БПП, который приведен ниже [2].

**Алгоритм БПП.**

```
алг БПП(цел таб a[1:n+1], нат n, цел b, лит c)
  арг a, n, b
  рез c
```

```
нач цел i
   i:=1, a[1:n+1]:=b          2
   пока a[i]№b                n+1
   нц
      i:=i+1                  n
   кц
   если iJn                   1
      то c:="да"              0
      иначе c:="нет"          1
   все
кон
```

Командам предлагается вычислить для этого алгоритма характеристики времени и памяти в худшем случае. Они будут равны соответственно следующим величинам:

$$T_{БПП}(n) = 2n + 5, \qquad M_{БПП}(n) = n + 5.$$

При сравнении характеристики времени алгоритма БПП с характеристикой времени алгоритма Поиск1 (который является одним из наиболее быстрых предложенных командами) при достаточно большом значении n получаем следующее:

$$\lim_{n \to \infty} \frac{3n+3}{2n+5} = \frac{3}{2}.$$

Это означает, что алгоритм БПП выполняется в 1,5 раза быстрее, чем алгоритм Поиск1, говоря другими словами, его использование вместо алгоритма Поиск1 позволяет экономить электроэнергию в 1,5 раза.

### Начисление баллов

Как и всякая игра, игровая форма проведения занятия имеет дух состязательности. И чтобы этот дух не угасал, необходимо постоянно фиксировать достижения команд. Поэтому на доске ведется учет баллов, заработанных командами. Баллы в нашем случае назначаются следующим образом (предполагается, что группа разбита на 4 команды). На занятии выделяют три этапа.

На первом этапе команда, разработавшая первой правильный алгоритм решения задачи последовательного поиска, получает 4 балла. Команда, которая была второй, получает 3 балла и т.д. Команда, допустившая ошибку при разработке алгоритма, баллов не получает. Команда, которая первой обнаружила ошибку в алгоритме другой команды, получает дополнительно 1 балл.

Второй этап состязания начинается после того, как все четыре алгоритма решения задачи последовательного поиска записаны на доске и в них ликвидированы ошибки. На этом этапе командам предлагается вычислить характеристики времени и памяти разработанных ими алгоритмов. Опять, как и на первом этапе состязания, команда, которая первая вычислила характеристику времени своего алгоритма (и она не содержит ошибок), получает 4 балла. Остальные команды в порядке очередности получают на 1 балл меньше относительно друг друга. Поощрительный балл добавляется той команде, у которой характеристика времени выплнения алгоритма оказалась лучше. За правильное вычисление характеристики памяти разработанных алгоритмов командам начисляется по одному баллу.

Третий этап начинается после того, как преподавтель напишет на доске алгоритм быстрого последовательного поиска. На этом этапе разыгрываются 2 балла. Из них один начисляется команде, первой вычислившей правильно характеристику времени алгоритма, а второй – команде за вычисление соответственно характеристики памяти алгоритма. Заработанные командами баллы учитываются преподавателем при сдаче экзамена по дисциплине «Алгоритмизация и программирование».

## Выводы

Описанная методика проведения практического занятия по теме «Эффективность алгоритмов решения задачи последовательного поиска» апробировалась на протяжении нескольких лет в группах культурологов-менеджеров. Было выявлено, что она обладает следующими особеностями: высокой мотивацией студентов, высоким духом состязательности, высокой интенсивностью учебного процесса, высокой эмоциональной окраской, большим объёмом выполненной учебной работы. Перечисленные особенности свидетельствуют о высокой эффективности учебного процесса с описанной игровой формой обучения алгоритмизации будущих культурологов-менеджеров.

## Литература

1. Ахо, А. Построение и анализ вычислительных алгоритмов / А. Ахо, Дж. Хопкрофт, Дж. Ульман. – М.: Мир, 1979. – 536 с.

2. Кнут, Д. Искусство программирования для ЭВМ: Т. 3. Сортировка и поиск / Д. Кнут. – М.: Мир, 1978. – 848 с.

*Статья поступила 09.06.2010.*