

МЕТОДИКА КОНСТРУИРОВАНИЯ УЧЕБНЫХ ЗАДАЧ С ПОМОЩЬЮ ДИДАКТИЧЕСКОЙ ИНЖЕНЕРИИ

Ринат Рамилович Сулейманов,

кандидат педагогических наук, доцент, член-корреспондент Академии информатизации образования, профессор Российской академии естествознания

• дидактическая инженерия • эвристика • учебная задача

В СТАТЬЕ РАССМАТРИВАЕТСЯ ПРОЕКТИРОВАНИЕ И КОНСТРУИРОВАНИЕ УЧЕБНЫХ ЗАДАЧ С ПОМОЩЬЮ ДИДАКТИЧЕСКОЙ ИНЖЕНЕРИИ.

Под дидактической инженерией мы будем понимать целенаправленное изучение, проектирование и конструирование дидактических объектов, в частности учебных задач. При проектировании и конструировании учебных задач мы используем конструктивное дидактическое мышление и эвристики, использующиеся при решении задач.

Конструктивное дидактическое мышление опирается на следующие принципы: лучше вглубь, чем вширь; процесс важнее, чем результат.

Принцип «лучше вглубь, чем вширь»: лучше решить одну задачу тремя способами, чем три задачи – одним способом.

Принцип «процесс важнее, чем результат»: при решении задач главная цель не просто получить правильный ответ, а стимулировать процессы поиска решения, обмена идеями, аргументации того или иного способа решения.

Под эвристикой мы понимаем область исследования методов решения задач. Ниже приведены эвристики, которые использовались при составлении приведённых ниже учебных задач. При решении задач эвристики могут использоваться как одиночно, так и в комбинации.

Эвристика. Метод подбора

Случайный подбор. Решение находится путём проб и ошибок: сначала проверяется

одна произвольная комбинация, затем – другая, пока случайным образом не будет найдено правильное решение.

Последовательный подбор. Предполагает начало решения задачи не с произвольной комбинации, а с последовательного анализа условия задачи.

Целенаправленный подбор отличается от предыдущих тем, что комбинации подбираются исходя из определённого условия.

Метод подбора эффективен в тех случаях, когда:

- достаточно прозрачна и понятна идея решения;
- есть возможность перебрать все варианты решения;
- в задаче содержится конечное количество вариантов поиска решения.

Эвристика.

Метод введения переменной

Метод введения переменной эффективен в тех случаях, когда:

- в условии задачи содержится фраза «для любого (каждого)»;
- в условии задачи представлено большое количество вариантов или ситуаций;
- задача не может быть решена другим способом, кроме составления уравнения;
- решение задачи требует доказательства и рассмотрения общего случая.

Эвристика. Поиск закономерности

Применение эвристики «поиск закономерности» эффективно в тех случаях, когда:

- задан массив чисел;
- задана какая-нибудь числовая последовательность;
- информация, данная в условии задачи, может быть организована в форме числовой таблицы или последовательности;
- по условию требуется сделать какое-то числовое обобщение.

Ниже приведены примеры учебных задач, составленных с помощью дидактической инженерии.

Задача 1. Вывести на экран все трёхзначные числа, кратные трём.

Решение 1

Число 102 кратно 3, и это – наименьшее из трёхзначных чисел, подходящих нам по условию задачи. Следующее число $102 + 3 = 105$ также кратно 3 и т.д. Тогда организуем цикл и будем выводить на экран все трёхзначные числа, начиная с 102, с шагом 3. Однако поскольку в языке Паскаль шаг цикла может быть равен только 1 или -1, применим небольшую хитрость: организуем цикл перебора чисел от 34 до 333 ($34 = 102/3 \dots 333 = 999/3$) с шагом 1 и будем выводить на экран эти числа, умноженные на 3.

Программа 1

```
program pr1;
var i:integer;
begin
for i:=34 to 333 do
write (3*i, ' ');
end.
```

*Решение 2***Программа 2**

Организация цикла «для»:

```
program pr2;
var i:integer;
begin
for i:=100 to 999 do
if i mod 3=0 then write (i, ' ');
end.
```

Решение 3

Число 102 кратно 3, и это – наименьшее из трёхзначных чисел, подходящих нам по условию задачи. Следующее число $102 + 3 = 105$ также кратно 3 и т.д. Тогда организуем цикл и будем выводить на экран все трёхзначные числа, начиная с 102, с шагом 3.

Программа 3

Организация цикла с предусловием:

```
program pr3;
var i:integer;
begin
i:=99;
while i<999 do
begin
i:=i+3;
write (i, ' ');
end;
end.
```

*Решение 4***Программа 4**

Организация цикла с постусловием:

```
program pr4;
var i:integer;
begin
i:=99;
repeat
i:=i+3;
write (i, ' ');
until i>996;
end.
```

*Решение 5***Программа 5**

Конструирование трёхзначных чисел по перебору единиц, десятков, сотен с последующим выделением тех трёхзначных чисел, которые делятся на три без остатка:

```
program pr5;
var a,b,c:integer;
begin
for a:=1 to 9 do;
for b:=0 to 9 do;
for c:=0 to 9 do;
if (a+b+c) mod 3=0 then write (a*100+b*10+c, ' ');
end.
```

Решение 6

Вспомним ещё одно свойство кратности числу 3: если сумма цифр некоторого числа

кратно 3, то и само это число кратно 3. При этом цифры числа будем извлекать из его соответствующих разрядов. Например, пусть дано число 326. Тогда, чтобы получить цифру сотен (3), нужно выполнить целочисленное деление исходного числа на сто ($326 \text{ div } 100 = 3$).

Для получения цифры десятков нужно из исходного числа вычесть цифру сотен, умноженную на 100, а затем выполнить целочисленное деление на 10 ($326 - 3 \cdot 100 \text{ div } 10 = 2$).

Для получения же цифры единиц нужно из исходного числа выделить остаток от деления на 100 ($326 \text{ mod } 100 = 6$). Именно так в программе в строках:

- $a = i \text{ div } 100$ – определение первой цифры (количества сотен);
- $b = (i - a \cdot 100) \text{ div } 10$ – определение второй цифры (количества десятков);
- $c = i \text{ mod } 100$

мы и получаем цифры сотен, десятков и единиц для числа x . А далее в строке:

```
if (a+b+c) mod 3=0 then write(i, ' ');
```

выполняется проверка кратности суммы цифр числу 3 и при выполнении этого условия производится вывод исходного числа на экран. Строки:

```
for i:=100 to 999 do
begin
...
end;
```

организуют перебор всех трёхзначных чисел (операторные скобки `begin ... end` необходимы, поскольку в теле цикла содержится несколько операторов).

Программа 6

```
program pr6;
var a,b,c,i:integer;
begin
for i:=100 to 999 do
begin
a:=i div 100;
b:=(i-a*100) div 10;
c:=i mod 100;
if (a+b+c) mod 3=0 then write (i, ' ');
end;
end.
```

Задача 2. Вывести на экран все трёхзначные числа, сумма цифр которых равна семи.

Решение 1

Здесь организуется простой перебор в цикле всех возможных трёхзначных чисел (от 100 до 999) с выделением цифр из текущего трёхзначного числа и проверкой выполнения заданного условия:

- $a = i \text{ div } 100$ – определение первой цифры (количества сотен);
- $b = (i - a \cdot 100) \text{ div } 10$ – определение второй цифры (количества десятков);
- $c = i \text{ mod } 100$ – определение третьей цифры (количества единиц);
- проверяется, не равна ли сумма цифр $a + b + c$ искомому числу 7.

Программа 8

```
program pr8;
var a,b,c,i:integer;
begin
for i:=100 to 999 do
begin
a:=i div 100;
b:=(i-a*100) div 10;
c:=i mod 100;
if (a+b+c)=7 then write (i, ' ');
end;
end.
```

Решение 2

Это своего рода «решение от противного»: при помощи двух вложенных циклов выполняется перебор всех возможных значений цифр в разрядах единиц и десятков (каждая цифра – от 0 до 9), затем вычисляется (вычитанием текущих значений этих цифр из исходного числа n) предполагаемое количество сотен и проверяется, лежит ли оно в допустимом диапазоне от 1 до 9. (В отличие от более тривиального перебора в цикле и значений сотен с последующей проверкой равенства суммы текущих значений цифр числу 7, такой способ позволяет значительно уменьшить время работы программы.)

Программа 9

```
program pr9;
var a,b,c:integer;
```

```
begin
for a:=1 to 9 do
for b:=0 to 9 do
for c:=0 to 9 do
if (a+b+c)=7 then write (a*100+b*10+c, ' ');
end.
```

Решение 3

Программа 10

```
program pr10;
var a,b,c:integer;
begin
for a:=1 to 9 do
for b:=0 to 9 do
if (7-a-b)>0 then write (a*100+b*10+7-a-b, ' ');
end. □
```

Литература

1. *Пойа Дж.* Как решать задачу / Дж. Пойа. – М.: Наука, 1966.
2. *Пойа Дж.* Математическое открытие / Дж. Пойа. – М.: Наука, 1976.
3. *Пойа Дж.* Математика и правдоподобные рассуждения / Дж. Пойа. – М.: Наука, 1976.
4. *Чошанов М.А.* Дидактика и инженерия / М.А. Чошанов. М.: БИНОМ, Лаборатория знаний, 2011.