Рекомендации
по решению задач по информатике (раздел программирование)
районных и городских олимпиад 2005-2006 учебный год.

Приведены краткие алгоритмы и решения на языке программирования Паскаль.

Сокращения:

- r - рекомендовано для районных олимпиад,
- r-g -рекомендовано для районных и городских олимпиад,
- g - рекомендовано для городских олимпиад.

УСЛОВИЯ ЗАДАЧ

Задача 1.(r1)

В трехмерном пространстве задан куб с ребром длиной 1. Одна из вершин в точке с координатами, например,(0,0,0), а противоположная с координатами (1,1,1). . По двум введенным вершинам определить, находятся ли они на одном ребре куба или на одной его грани.

Задача 2. (g1).По ребрам куба ползут тараканы. Все они ползут из вершины (000) в вершину (111) так, что каждый из них проползает при этом три ребра. Найти самое «истоптанное» ребро.

Задача 3 (r3,g3) Нужно расшифровать длинное секретное донесение. Таблица кодировки неизвестна, зато известно, что каждый символ закодирован восемью битами.

Задача 4 (r2) Вводится последовательность целых чисел (в диапазоне от -32000 до 32000). Требуется разбить их на пары так, чтобы произведения чисел в парах были одинаковы.

Входные данные: Вводится N (количество чисел), 1<=N<=100. Затем вводятся N целых чисел.

Выходные данные: Если разбить на пары можно, то выдать произведение чисел в паре (все варианты), иначе - сообщение "Нет решения".

Задача 5 (g4) Есть N лампочек и M переключателей, каждый из которых какие-то лампочки переключает, а какие-то нет. Сначала часть ■ лампочек включена. Договоримся горящие лампочки обозначать 1, а выключенные - 0. Для переключателей будем писать 1, если переключатель меняет состояние данной лампочки, и 0, если не меняет. Тогда любой переключатель можно представить строкой из N нолей и единиц. Начальное и конечное состояние лампочек
также закодируем строкой из 0 и 1. «Применение» переключателя к лампочкам приводит к тому, что включенные лампочки становятся выключенными, а выключенные - включенными (естественно, это справедливо только для лампочек, к которым есть доступ от этого переключателя). Напишите программу, которая определяет, какие переключатели нужно применить, чтобы лампочки перешли в конечное состояние.

Входные данные:

Вводятся числа N и M (1<=N<=50, 1<=M<=50).

Вводится строка, характеризующая начальное состояние
лампочек.

Вводится строка, характеризующая конечное состояние
лампочек.

Вводится M строк, характеризующих переключатели. Выходные данные: Номера переключателей, которые нужно применить, причем каждый переключатель можно применить не более 1 раза, либо сообщение "Нет решения".

Задача 6. (r4,g2) Написать программу, которая объединяет два упорядочен
ных по возрастанию массива в один, также упорядоченный по возрастанию массив. Рекомендуемый вид экрана во время рабо ты программы приведен ниже (данные, введенные пользовате лем, выделены полужирным шрифтом).

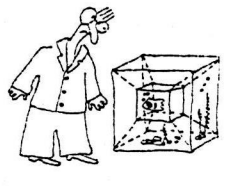
Объединение двух упорядоченных по возрастанию массивов.
Введите в одной строке элементы первого массива,
(5 целых чисел) —> 1 3 5 7 9
Введите в одной строке элементы второго массива,
(5 целых чисел) —> 2 4 6 8 **10**
Массив — результат
1 2 3 4 5 6 7 8 9 10
Для завершения работы нажмите <Enter>.

АЛГОРИТМЫ И РЕШЕНИЯ ЗАДАЧ

*Задача 1.(r1)*Две вершины принадлежат одному ребру, если у них совпадают две координаты. Например, у вершин (ПО) и (100) совпадают первая и последняя координаты.

Две вершины принадлежат одной грани, если у них совпадает хотя бы одна координата.

Чтобы получить ответ, достаточно сосчитать количество совпадений:

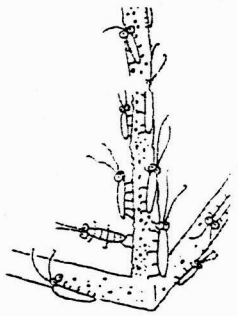


```
var si, s2: string;
  i, c: integer; begin
 readln(si);
 readln(s2);
 c:=0;

{считаем количество
совпадений}
 for i:=1 to 3 do if s1[i] = s2[i] then inc(c);
{выводим результат}
```

```
 if c >= 2 then writeln("Принадлежат одному
ребру")          else writeln("Не принадлежат
одному
                        ребру"); if c >= 1 then
 writeln("Принадлежат одной грани")
        else writeln("Не принадлежат одной
грани"); end.
```

Задача 2.(g1).



Нам нужно для каждого ребра запомнить количество тараканов, которые по нему проползли. Для этого нужно так организовать хранение данных, чтобы было удобно с ними работать.

Давайте для этого сначала пронумеруем вершины. Вершина задается своими тремя двоичными координатами. Поэтому ей можно приписать номер, соответствующий образованному двоичному числу из трех знаков. Получим номера от 0 до 7. Например, вершина ОН будет **иметь** номер 3.

Теперь составим таблицу 7 х 7, в которой в ячейке (т, п) будет храниться ребро (то есть количество протопавших по нему тараканов), идущее из m-той вершины в п-тую.

Много клеток в этой таблице нам не понадобится. Во-первых, потому что тараканы ползут по ребрам только в одном направлении, а в таблице присутствуют оба. Во-вторых, потому что не из каждой вершины в каждую проходит ребро. У нас в таблице будет 49 клеток, а понадобится только 12. Очень неэкономно, зато удобно.

```
var
  a: arraytO..7,0..7] of integer; {массив ребер}
  m, i, j, imax, jmax: integer;
  a: string;
  vl, v2: integer;

{возвращает номер ребра, записанного в строке s} function
getnum(s:string):integer; var n: integer; begin
    n:=0;
    if s[l] = "l" then n:=n+4;
    if s[2] = "1" then n:=n+2;
    if s[3] = "1" then n:=n+l;
    getnum:=n; end;

{возвращает двоичную запись п - того ребра} function getstr(n:
integer): string; var s:string; begin
    s:='000';
    if n mod 2 = 1  then s[3]:='l';
    n:=n div 2;
    if n mod 2 = 1  then s[2]:='l';
    n:=n div 2;
    if n mod 2 = 1  then s[2]:='l';
    getstr:=s; end;

begin

  for i:=0 to 7 do
    for j:=0 to 7 do a[i,j]:=O; {сначала тараканов не было
нигде}

  readln(m); {количество тараканов} for i:=1 to m do begin
      readln(s) ;
      vl:=getnum(copy(s,1,3)); {вычисляем номера вершин}
      v2:=getnum(copy(s,5,3));
inc(a[0,vl]); {прибавляем тараканов к нужным реб
рам}
inc(a[vl,v2]);
inc(a[v2,7]);
end;

  imax:=0; {находим самое истоптанное ребро} Д\"5» э j max:=0;
  for i:=0 to 7 do for j:=0 to 7 do
      if a[i,j]>a[imax,jmax] then begin
            imax:=i;                          jmax:=j; end;
                          {вывод}
writeln(getstr(imax)); writeln(getstr(jmax));
end.
```

Задача 3(r3,g3)

Я завела два массива длиной 256 - comf и codf. В первый я записала статистику файла common.tct, а во второй - файла secret.cod (в г-тую ячейку записывается количество повторений /-го байта в файле).

Далее 256 раз была выполнена следующая операция: находим наиболее часто встречающийся символ в файле secret.cod и ставим ему в соответствие наиболее часто встречающийся символ в файле common.trt. Это соответствие записывается в массив key (Номер ячейки — это байт кода, а ее содержимое - байт текста). После этого в массивах comf и codf в найденные ячейки записывается «-1», чтобы не влиять на поиски следующего максимального элемента.

После этого просматривается файл secret.cod, и каждому байту ставится в соответствие (то есть выводится в файл secret.trt) символ из массива *key*.

```
type
  TFrequency = array[0..255] of longint;
{массив для статистики}
  TFile = file of byte;
```

```
{процедура заполняет массив fr статистикой файла f}
procedure LoadFrequency(var f: TFile;var fr:
  TFrequency) ; var b:byte;

  begin
    reset(f);
    for b:=0 to 255 do fr[b]:=O; while not eof(f) do begin
      read(f,b); inc(fr[b]); end;
    close(f); end;

{функция возвращает номер наибольшей ячейки массива fr и
записывает в нее -1}
function MaxChar(var fr: TFrequency): byte; var m, i: byte; begin m:
  =0;for i:=l to 255 do
      if fr[i] > fr[m] then m:=i;

    MaxChar:= m;
    fr[m] : = -1; end;

var
  com, cod, txt : TFile; comf, codf :TFrequency; key: array[0..255]
  of byte; i: integer; b: byte;

begin
  assign(com, "common.txt") ;
  assign(cod, "secret.cod");
  assign(txt, "secret.txt");

  LoadFrequency(com,comf); {считаем статистику}
  LoadFrequency(cod,codf) ;

  {заполняем массив key} for i:=0 to 255 do
    key[MaxChar(codf)]:=MaxChar(comf);
  rewrite(txt); reset(cod); {расшифровка} while not eof(cod) do
  begin
    read(cod,b) ;  <s write(txt, key[b]); end;

close(txt); close(cod); end.
```

Я взяла два отрывка из произведения Булгакова «Мастер и Маргарита» длиной 6000 строк. Один из них был записан в файл common.Ш., а другой закодирован и записан в файл secret.cod. Эта программа расшифровала текст в файл secret.txt. Вот отрывок расшифрованного текста:

'.етрудно доядатьсгп что толстгк с баяровой Пиыиономи-ейп которояо ,оместили в клинике в комнате l))7 бзл !иканор :ванович Мосой.

?о,ал он однакоп к ,ро1Тессору Ттравинскому не сраыуп а ,редварительно ,обзвав в друяом месте.

Кт друяояо этояо места у 'икаиора :вановича осталось в вос,оминании мало чеяо. ?омнилсг только ,исьменн:# столп шкаП и диван.

Оам с !иканором :вановичемп у которояо ,еред ялаыами как-то мутилось от ,риливов крови и душевнояо воыбужденигп всту,или в раыяоворп но раыяовор взшел какой-то страннзйп ,утанзйп а вернее скаыать совсем не взшел.

Видно, что это расшифровка только в первом приближении. Но теперь расшифровать текст до конца очень просто. Например, сразу видно, что нужно заменить следующие символы:

С и м в о л	і	я	п		П	З	ы
З а м е н а	Н	г		п	ф	ы	3

И так далее. Интересно, что символы в одной строчке таблицы встречаются примерно с одинаковой частотой (иначе не произошло бы ошибки при расшифровке).

```
Задача 4. (r2);{*Пары*} const MaxN=100;
type Mas=array[O..MaxN] of longint; var QNum,PNum:Mas;
  N,Q,P,O:integer;(*Q - <0, P - >0,0 - =0*)
  Okrboolean;

procedure PrintYes(l:longint);
 begin
 writeln('OTBeT: ',1);
 Ok:=True;
 end;

procedure but;
 var j,tp:integer;
 begin
 Q:=0;P:=0;O:=0;
 writeln('BBOfl >');read(N);
 for i:=l to N do begin read(tp);
  if tp<0 then begin Inc(Q);QNum[Q]:=tp; end else if tp>0 then begin
  Inc(P);PNum[P]:=tp; end else Inc(O);
 end;
 if N mod 2<X) then Ok:=False;
 end;

 procedure Sort(var Q:Mas;num:integer); var ij:integer;
   l:longint; begin
  for i:=l to num do for j:=l to num-i do if Q[j]"QEJ+l] then begin
   1:-Q[fl;
   QD]:=QD+4;
   Q0+1JH;
  end; end;

procedure Solve;
var tp: longint;
i:integer;
```

```
begin
if 0>=(N div 2) then PrintYes(O) else begin if P=Q then begin
 tp:=PNum[1]*QNum[1];
  i:=2;
  while (i<=N) and (PNum[i]*QNum[i]=tp) do Inc(i);
   if i>P then PrintYes(tp); end;
  if (P mod 2=0) and (Q mod 2=0) then begin if PoQ then
  tp:=PNum[l]*PNum[P] else tp:=QNum[l]*QNum[Q];
  i:=l;
  while (i<=P div 2) and (PNum[P-i+l]*PNum[i]=tp) do Inc(i);
  if i>P div 2 then i:=l else i:=(Q div 2)+2;
  while (i<=Q div 2) and (QNum[Q-i+l]*QNum[i]=tp) do Inc(i); if
    i=(Q div 2)+l then PrintYes(rp);
 end; end; end;

BEGIN
Ok:=False;
Init;
if N mod 2=0 then begin
 Sort(QNum,Q);
 Sort(PNum,P);
 Solve;
end;
if Not Ok then writeln('HeT решения');
END.
```

Задача 5.(g4). Пример: Входные данные:

```
53
10010
11000
11100
10001
10110 Выходные данные: 1  3
```

Решение g4.

```
{ Лампочки }
Const MaxN=50; Const MaxM=50;
Type Perekluk= Array [L.MaxN] Of Boolean;
   Sset=Set Of Byte;
Var St, Fin: Perekluk;
  P: Arrayf 1 ..MaxM] Of Perekluk;
  N, M: Integer;
  Who: Array[l..MaxN] Of Sset;
  Res, CanUse: Sset;

Procedure GetPr(Var T: Perekluk); Var S: String; i: Integer;
 Begin
 ReadLn(S);
 For i:=l To Length(S) Do T[i]:=(S[i]=T); End;

Procedure Init; Var i: Integer; Begin
 WriteLnCN, M,...'); ReadLn(N, M);
 GetPr(St);
 GetPr(Fm); For i:=l To M Do Begin
 GetPr(P[i]);
 Who[i]:=[i]; End;
CanUse:=[l..M]; Res:=[]; End;

Procedure SetUp(A, B: Perekluk;Var C: Perekluk); Var i: Integer;
Begin
 C:=A;
 For i:=l To N Do If B[i] Then C[i]:=(A[i] Xor B[i]);
End;

Procedure MakeStep(k: Integer); Vari,j: Integer; Begin i:=l;
 While (i<=M) And Not(P[i,k] And (i In CanUse)) Do Inc(i);
 Ifi<=M Then Begin
    If St[k]oFin[k] Then Begin SetUp(St,P[i],St);
      Res:=(Res+Who[i])-Res*Who[i]; End;
  CanUse:=CanUse-[i]; For j:=i+l To M Do If P[j,k] And (j In
  CanUse) Then Begin
      SetUp(PD],P[i],PD]);
  WhoG]:=(Who[j]+Who[i])-(WhoD]*Who[i]); End; End; End;

Procedure Solve; Var i, Min: Integer;
Begin
 i:=l;
 If N<M Then Min:=N Else Min:=M;
 For i:=l To Min Do MakeStep(i);
End;

Procedure Done; Var i: Integer; Begin i:=l;
 While (i<=N) And (St[i]=Fin[i]) Do Inc(i); If i<=N Then
 WriteLn('HeT решения') Else Begin Write('PeiHemie:");
 For i:=l To M Do If i In Res Then Write(i,"); If Res=Q Then
 WriteCHmero не переключайте'); WriteLn; End; End;
```

Задача 6 (r4,g2)

```
{ Объединение (слияние) двух упорядоченных массивов
в один } const
  HB = 5; {  размер исходных массивов } var
  a,b: array[1..HB] of integer; { исходные массивы } c:
  array[1..2*HB] of integer; { массив - результат } k,l,m :
  integer;  ( индексы массивов a,Ь и с }

begin
  writeln('Объединение двух упорядоченных по
          возрастанию ', 'массивов.');
  writeln('Введите в одной строке элементы первого ',
          'массива,');
  write('(',HB,'  целых чисел)  ->  '); for k:=l to
  HB-1 do
     read(a[k]); readln(a[HB]);

  writeln('Введите в одной строке элементы
  второго массива'); write('C,HB,'  целых чисел)  -
  >  ');


for 1:=1 to HB-1 do
    read(b[l]); readln(b[HB]);

к:=1; l:=1; m:=l; repeat
     if a [к] < b[l] then
           begin
       c[m]:=a[k] ; m:=m+l; k:=k+l; end else
       if a[k] > b[l] then
           begin
         c[m]:=b[l]; m:=m+l;1:=1+1; end else
           begin
           c[m]:=a[k]; c[m+l]:=b[l]; k:=k+l; 1:=1+1; m:=m+2;
           end;
```

```
until (k > HB) or (l > HB); { один из двух исходных массивов
полностью переписан в массив C }
while к <= HB do { есть эл-ты A, не переписанные в C } begin
    c[m]:=a[k];
    к:=к-l;
    m:=m-l; end;
while l <= HB do { есть эл-ты B, не переписанные в C } begin
    c[m]:=b[l];
l:=l+1;
m:=m+1;
for I:=1 to 2*HB do
write(c[I],' ');
readln;
end.
```