Шипулин Павел Андреевич

Балл: 107

Задача №1

Условие

Учитель информатики Игорь Петрович составил программу для тестирования младшеклассников по арифметике. Она выдаёт примеры на умножение и деление в пределах 100, в которых пропущено одно из чисел. Вам требуется написать другую программу, которая для заданных примеров будет определять правильный ответ.

Входные данные: запись примера на умножение или деление в форме A*B=C или A:B=C, где A, B, C - целые неотрицательные числа. Одно из чисел заменено знаком вопроса.

Выходные данные: число, которое требуется подставить вместо знака вопроса, чтобы пример превратился в равенство.

Известно, что результат, сомножители, делимое и делитель являются натуральными числами.

Исходные данные	Результат
12*?=48	4
18/9=?	2

Исходный код

```
 var
  s,sa,sb,sc:string;
  a,b,c,err:longint;
  sign:char;

begin
  readln(s);

  if pos('*',s)>0 then
  sign:='*'
  else
  sign:='/';

  sa:=copy(s,1,pos(sign,s)-1);
  sb:=copy(s,pos(sign,s)+1,pos('=',s)-(pos(sign,s)+1));
  sc:=copy(s,pos('=',s)+1,length(s)-pos('=',s));

  val(sa,a,err);
  val(sb,b,err);
  val(sc,c,err);
  if sa='?' then
  begin
    if sign='*' then
      writeln(c div b)
    else
      writeln(c*b);
  end
  else if sb='?' then
  begin
    if sign='*' then
      writeln(c div a)
    else
      writeln(a div c);
  end
  else
  begin
    if sign='*' then
      writeln(a*b)
    else
      writeln(a div b);
  end;
end.
```

Задача №2

Условие

Строительной организации поступил заказ на проведение отделочных работ в здании строящейся школы. Некоторое количество помещений требуется облицевать плиткой. По желанию заказчика облицовка должна быть выполнена в двух цветах: периметр каждой стены - тёмной плиткой в один ряд, остальную площадь - светлой плиткой.

Необходимо рассчитать общее количество коробок плитки, которая потребуется для выполнения заказа.

Входные данные: в первой строке - через пробел натуральные числа K (количество стен, которые требуется облицевать), N (количество плиток в одной коробке), X (ширина одной плитки в сантиметрах), Y (высота одной плитки в сантиметрах). Далее K строк, в каждой строке по 2 натуральных числа через пробел: ширина и высота очередной стены в сантиметрах.

K < 1000, N < 200, X и Y < 50; ширина и высота стен не превышает 1000.

Выходные данные: требуемое количество коробок.

При решении задачи считать, что:

- - плитка укладывается вплотную, без зазоров;
- - если высота или ширина стены не кратна высоте или ширине плитки, то излишек плиток крайнего ряда отрезается и далее нигде не используется;
- - плитка поставляется только полными коробками.

Исходные данные	Результат
1 10 20 30 100 200	4
3 20 20 20 300 270 400 250 200 270	31

Исходный код

```
 var
  s,i,k,n,x,y,w,h:longint;
begin
  readln(k,n,x,y);
  s:=0;
  for i:=1 to k do
  begin
    readln(w,h);
    s:=s+(w*h) div (x*y);
  end;
  k:=0;
  if s mod n>0 then k:=1;
  s:=s div n+k;
  writeln(s);
end.
```

Задача №3

Условие

В строке записано истинное логическое выражение с тремя переменными a, b и c. Над переменными применяется операция "штрих Шеффера".

Штрих Шеффера — бинарная логическая операция, являющаяся эквивалентной операции И-НЕ и обозначающаяся вертикальной чертой "|".

Входные данные: логическое выражение, состоящее из имён переменных a, b, c, вертикальной черты и круглых скобок. Никаких операций, помимо штриха Шеффера, не предусмотрено. Скобки задают приоритет выполнения операций. Длина входной строки не превышает 50 символов.

Выходные данные: 8 цифр 0 и 1, записанных неразрывно и означающих значения функции для каждой из комбинаций значений переменных:

0, 0, 0;

0, 0, 1;

0, 1, 0;

0, 1, 1;

1, 0, 0;

1, 0, 1;

1, 1, 0;

1, 1, 1.

Исходные данные	Результат
a|b|c	10101011
a|(b|c)	11110001

Исходный код

```
 type
  abcstack =
  record
    len:integer;
    data:array[1..1000] of boolean;
  end;

var
  res:abcstack;
  eq,s,opers:string;
  i,j:integer;
  a,b,c,p1,p2:boolean;

function pop(var st:abcstack):boolean;
begin
  pop:=st.data[st.len];
  st.len:=st.len-1;
end;

procedure push(var st:abcstack; v:boolean);
begin
  st.len:=st.len+1;
  st.data[st.len]:=v;
end;

begin
  readln(s);

  opers:='[';
  for i:=1 to length(s) do
  begin
    if s[i]='|' then
    begin
      if length(opers)>1 then
      if opers[length(opers)]='|' then
      begin
        if (opers[length(opers)-1]='(') or (opers[length(opers)-1]='[') then
        begin
          opers:=copy(opers,1,length(opers)-1);
          eq:=eq+'|';
        end;
      end;
      opers:=opers+'|';
    end
    else if s[i]='(' then
    opers:=opers+'('
    else if s[i]=')' then
    begin
      eq:=eq+'|';
      opers:=copy(opers,1,length(opers)-2);
    end
    else if s[i]<>'(' then
    eq:=eq+s[i];
  end;
  eq:=eq+opers;

  for i:=0 to 7 do
  begin
    a:=false;
    b:=false;
    c:=false;
    if i shr 2=1 then a:=true;
    if (i shr 1) and 1=1 then b:=true;
    if i and 1=1 then c:=true;

    res.len:=0;
    for j:=1 to length(eq) do
    begin
      if eq[j]='a' then
      push(res,a)
      else if eq[j]='b' then
      push(res,b)
      else if eq[j]='c' then
      push(res,c)
      else if eq[j]='|' then
      begin
        p2:=pop(res);
        p1:=pop(res);
        push(res,not(p1 and p2));
      end;
    end;

    if res.data[1] then
    write(1)
    else
    write(0);
  end;
end.
```

Задача №4

Условие

Строку B можно получить из строки A с помощью следующих преобразований: замены одного символа на другой в той же позиции, вставки символа, удаления символа.

Требуется определить минимальное количество преобразований, которое потребуется для преобразования заданной строки A в строку B.

Входные данные: 2 строки из букв латинского алфавита, длина каждой из которых не превышает 20.

Выходные данные: целое число - искомое количество преобразований.

Исходные данные	Результат
abcd acbe	3
abcde acbe	2

Исходный код

```
 var
  a,b:string;
  s,i,l:integer;
begin
  readln(a);
  readln(b);
  l:=length(b);

  a:=a+'--';
  b:=b+'--';
  s:=0;
  for i:=1 to l do
  if a[i]<>b[i] then
  begin
    if a[i+1]=b[i] then
    begin
      delete(a,i,1);
      s:=s+1;
    end
    else if (a[i]=b[i]) and (a[i+1]=b[i+2]) then
    begin
      insert(b[i+1],a,i+1);
      s:=s+1;
    end
    else
    begin
      a[i]:=b[i];
      s:=s+1;
    end;
  end;
  writeln(s);
end.
```

Задача №5

Условие

Дано множество отрезков на плоскости. Требуется определить, сколько равносторонних треугольников можно построить из этих отрезков. При этом отрезки можно перемещать, но нельзя менять их размер и угол наклона к осям. Один отрезок можно включать только в один треугольник.

Входные данные: в первой строке натуральное N < 100 - количество отрезков. Далее в каждой строке по 4 числа, записанных через пробел: x1, y1, x2, y2 - координаты начала и конца отрезка.

Выходные данные: натуральное число - количество треугольников.

Треугольник считать равносторонним, если разность длин каждой пары его сторон не превышает 1% длины наименьшей стороны в паре.

Исходные данные	Результат
5 0 0 3 0 0 0 1.5 2.598 0 2.598 1.5 0 10 0 10.5 2.6 10 0 13 0	1
4 0 0 0 2 0 0 1 2 0 0 2 2 0 0 2 0	0

Исходный код

```
 type
  vector2 =
  record
    x,y:real;
  end;

var
  x1,y1,x2,y2,cosalpha1,cosalpha2,mn:real;
  i,j,k,n,s:integer;
  used:array[1..100] of boolean;
  vec:array[1..100] of vector2;
  len:array[1..100] of real;

begin
  readln(n);

  for i:=1 to n do
  begin
    readln(x1,y1,x2,y2);
    vec[i].x:=x2-x1;
    vec[i].y:=y2-y1;
    len[i]:=sqrt(sqr(vec[i].x)+sqr(vec[i].y));
    used[i]:=false;
  end;

  s:=0;
  for i:=1 to n do
  for j:=i+1 to n do
  for k:=j+1 to n do
  if (used[i]) or (used[j]) or (used[k]) then continue
  else
  begin
    mn:=len[i];
    if mn>len[j] then mn:=len[j];
```

```
    if mn>len[k] then mn:=len[k];
    cosalpha1:=abs((vec[i].x*vec[j].x+vec[i].y*vec[j].y) / (len[i]*len[j]));
    cosalpha2:=abs((vec[i].x*vec[k].x+vec[i].y*vec[k].y) / (len[i]*len[k]));
    if (0.49<cosalpha1) and (cosalpha1<0.51) and (0.49<cosalpha2) and (cosalpha2<0.51) and (abs(len[i]-len[j])/mn<0.01) and (abs(len[i]-len[k])/mn<0.01) and (abs(len[k]-len[j])/mn<0.0
    begin
      s:=s+1;
      used[i]:=true;
      used[j]:=true;
      used[k]:=true;
    end;
  end;

  writeln(s);
end.
```

Задача №6

Условие

Фирма BugMail запустила свой публичный почтовый сервис. Любой желающий имеет возможность создать там свою учётную запись, привязанную к существующей электронной почте, при этом перед разработчиками стоит задача отследить уникальность регистрируемых e-mail адресов.

Они пробуют реализовать систему проверки с учётом требований RFC (Request for Comments - документы, содержащие технические спецификации и стандарты, широко применяемые в сети Интернет), но из-за запутанности стандарта решено ограничиться следующими правилами:

- - адрес e-mail обязательно должен содержать единственный символ @, который идёт не первым и не последним;
- - в правой части адреса после @ идёт регистронезависимое имя домена 2 уровня и больше, то есть содержащее не менее одной точки не в начале и не в конце;
- - имя домена может содержать буквы, цифры, дефисы "-" и точки;
- - если в имени домена точка стоит первой или последней, то она не несёт смыслового значения: домены ".aaa.bb" и "aaa.bb." тождественны;
- - имя абонента (левая часть адреса до @) является регистронезависимым и может содержать буквы, цифры, дефисы, точки и символ "+";
- - если в имени абонента есть символ +, то часть от символа + до @ считается тегом и не влияет на уникальность адреса.

В рамках данной задачи считать, что буквы в адресах могут быть только латинскими.

Требуется для каждого адреса определить, является ли он корректным и уникальным.

Входные данные: в первой строке количество адресов N - натуральное число <= 1000, далее по одному адресу в строке, каждый не более 100 символов.

Выходные данные: N строк, по одной на каждый адрес входных данных. По каждому адресу из входных данных должно быть выдано:

- - сообщение "incorrect" без кавычек, если адрес не соответствует описанным правилам;
- - сообщение "correct, unique" без кавычек, если адрес корректен и встретился впервые (не имеет аналогов среди предыдущих адресов);
- - сообщение "double of <...>" без кавычек, если адрес корректен, но ранее встретился его аналог. Вместо ... должно быть выведено первое вхождение этого адреса. Например, "aBcD+rt@example.com" является аналогом адреса "abcd@EXAMPLE.COM."

Исходный код

```
 var
  name,domen:set of char;
  adres:array[1..1000] of string;
  i,n:integer;
  dbl:string;

function ps(sub:char; s:string):byte;
var
  i:byte;
begin
  ps:=0;
  for i:=1 to length(s) do
  if s[i]=sub then
  begin
    ps:=i;
    break;
  end;
end;

function ps(sub:char; s:string; p:byte):byte;
var
  i:byte;
begin
  ps:=0;
  for i:=p to length(s) do
  if s[i]=sub then
  begin
    ps:=i;
    break;
  end;
end;

function upcases(s:string):string;
var
  i:integer;
begin
  for i:=1 to length(s) do
  s[i]:=upcase(s[i]);
  upcases:=s;
end;

function morph(s:string):string;
var
  l,r:string;
begin
  if ps('+',s)>0 then
  delete(s,ps('+',s),ps('@',s)-ps('+',s));

  l:=copy(s,1,ps('@',s)-1);
  r:=copy(s,ps('@',s),length(s)-ps('@',s)+1);

  if l[1]='.' then delete(l,1,1);
  if l[length(l)]='.' then delete(l,length(l),1);

  morph:=upcases(l+r);
end;

function check_ad(s:string; m:integer):byte;
var
  right,left:string;
  flag:boolean;
  i:integer;
begin
  check_ad:=0;
  if (ps('@',s)=1) or (ps('@',s)=length(s)) or (ps('@',s,ps('@',s)+1)>0) then
    check_ad:=1
  else
  begin
    right:=copy(s,ps('@',s)+1,length(s)-ps('@',s));
    left:=copy(s,1,ps('@',s)-1);

    if not ((right[1]<>'.') and (right[length(right)]<>'.') and (ps('.',right)>0)) then
      check_ad:=1
    else
    begin
      flag:=true;
      for i:=1 to length(right) do
      if not (upcase(right[i]) in domen) then
      begin
        flag:=false;
        check_ad:=1;
        break;
      end;
```

```
      if flag then
      begin
        for i:=1 to length(left) do
        if not (upcase(left[i]) in name) then
        begin
          flag:=false;
          check_ad:=1;
          break;
        end;

        if flag then
        begin
          for i:=1 to m-1 do
          begin
            dbl:=adres[i];
            if morph(s)=morph(dbl) then
            begin
              check_ad:=2;
              break;
            end;
          end;
        end;
      end;
    end;
  end;
end;

begin
  readln(n);

  name:=['A','B','C','D','E','F','G','H','J','K','L','M','N','O','P','Q','R','S','T','U','V','W','X','Y','Z','.','-','+','0','1','2','3','4','5','6','7','8','9'];
  domen:=['A','B','C','D','E','F','G','H','J','K','L','M','N','O','P','Q','R','S','T','U','V','W','X','Y','Z','.','-','0','1','2','3','4','5','6','7','8','9'];

  dbl:='';
  for i:=1 to n do
  begin
    readln(adres[i]);

    case check_ad(adres[i],i) of
      0:writeln('correct, unique');
      1:writeln('incorrect');
      2:writeln('double of <',dbl,'>');
    end;
  end;

end.
```

Задача №7

Условие

Разреженной называется такая матрица, в которой количество ненулевых элементов существенно меньше количества нулевых. Разреженные матрицы можно хранить в форматах специального вида, например, только индексы ненулевых элементов и их значения.

Требуется написать программу, которая для разреженной матрицы большого размера определит номер строки с наибольшим средним арифметическим положительных элементов.

Входные данные: в первой строке три натуральных числа N, M и K, записанные через пробел. N - количество строк в разреженной целочисленной матрице, M - количество столбцов, K - количество ненулевых элементов. N и M не превышают 10^6 , K меньше N*M и не превышает 10^4. Далее по одному в строке указаны ненулевые элементы этой матрицы. Для представления каждого элемента через пробел записаны индексы его строки и столбца, а также значение. Индексы считаются с единицы, значения элементов по модулю не превышают 1000.

Выходные данные: номер искомой строки.

Исходные данные	Результат
4 3 1 1 1 10	1

Исходный код

```
 var
  a,c:array[1..1000000] of longint;
  n,i,v,x,y,k,mi,m:longint;
  mx:real;

begin
  readln(n,m,k);

  for i:=1 to k do
  begin
    readln(x,y,v);
    if v>0 then
    begin
      a[x]:=a[x]+v;
      c[x]:=c[x]+1;
    end;
  end;

  mx:=0;
  mi:=0;
  for i:=1 to n do
  if c[i]>0 then
  begin
    if mx<a[i]/c[i] then
    begin
      mi:=i;
      mx:=a[i]/c[i];
    end;
  end;

  writeln(mi);
end.
```