РОЗВ'ЯЗАННЯ ЗАВДАНЬ ІІ ЕТАПУ ВСЕУКРАЇНСЬКОЇ УЧНІВСЬКОЇ ОЛІМПІАДИ З ІНФОРМАТИКИ 2016-2017 НАВЧАЛЬНОГО РОКУ

9-11 КЛАС

Задача 1: Прямокутник

Якщо одну сторону прямокутника позначити за **x**, то друга буде дорівнювати **x–a**, а їх сума **x+x–a=p/2**. Звідки x=(p/2+a)/2=(p+2*a)/4. Тоді площа прямокутника буде дорівнювати (p+2*a)/4*((p+2*a)/4-a)= (p+2*a)/4*(p–2*a)/4=(p*p-4*a)/16. Враховуючи, що відповідь за умовою є число натуральне, маємо (p*p-4*a) div 16

Задача 2: Відрізки однієї прямої

Будемо міркувати спершу так: через одну точку не можна провести відрізків, тому відповідь буде 0; через дві точки можна провести один відрізок – відповідь 1; для випадку трьох точок: з першої точки, що розташована найлівіше, проведемо до правіших від неї 2 відрізки, а з другої – 1, тому відповідь 2+1=3. Для чотирьох точок: для першої точки – 3 відрізки, для другої – 2, для третьої – 1. Відповідь 3+2+1=6. Узагальнюємо для n точок: для першої точки – n-1 відрізок, для другої точки – n-2, для останньої – 1, тому відповіддю буде сума (n-1)+(n-2)+…+2+1, тобто сума перших n-1 натуральних чисел. Записавши розв'язок через відповідний цикл, отримаємо 8 балів (2 тести не проходять по часу). Якщо не врахували, що сума повинна бути типу int64 для Pascal, то 6 тестів. Для повного розв'язку потрібно записати формулу цієї суми або через суму арифметичної прогресії або міркуваннями, наприклад, такими: з кожної точки до решти можна провести n-1 відрізків, а для n точок – n*(n-1), при цьому кожен відрізок порахувався два рази, тому кінцева формула – n*(n-1) div 2.

Задача 3: Змагання зі стрільби

Для кожного спортсмена потрібно знайти суму найкращих трьох спроб, тобто найбільші три числа. Якщо використати квадратичне сортування елементів для кожного рядка і вибрати три перших, то можна набрати 6 балів. Якщо ж використати швидке сортування, то програма працює на межі часу і можуть не пройти два тести. Якщо ж

використати сортування по індексу або пробігти тричі по елементах кожного рядка для визначення максимального, помічаючи вже вибрані, то такий алгоритм проходить всі тести.

Задача 4: Сума цифр

Зчитуємо даний цифри як один рядок. Рухаючись, наприклад, із кінця рядка виконуємо дії, показані в прикладі. Для знаходження суми цифр отриманого числа використовуємо цикл while, який на кожному кроці знаходить останню цифру і додає до початкової суми, а ця цифра від числа відкидається. Цикл триває, поки число не стає рівним нулю

Приклад програми на Pascal

```
var s:string;
 i,sum,k:longint;
 n:int64;
begin
 readln(s);
 k:=1;n:=0;
 for i:=1 to length(s) do
   begin
     if s[length(s)-i+1]='1' then n:=n+k;
     k:=k*2;
    end;
sum:=0;
while n>0 do
 begin
  sum:=sum+n mod 10;
  n:=n div 10;
 end;
write(sum);
end.
```

Задача 5: Шахова дошка

Це задача з теорії графів. Вершиною графа є шахова клітинка, а ребрами є можливі ходи тури за один хід. Запустивши обхід графа у ширину зі стартової вершини, знаходимо найкоротший шлях і довжину цього шляху до фінішної вершини. У масиві a[][] будемо зберігати інформацію про клітинку (вільна, зайнята, стартова, фінішна), масиві

magx[] будемо зберігати першу координату вершини, куди ми потрапили, у масиві magy[] – другу координату, у масиві p[] – кількість кроків до цієї вершини.

```
var i,j,n,k,t,finx,finy,x,y:longint;
p,magx,magy:array[0..10000] of longint;
a:array[0..101,0..101] of longint;
begin
// зчитування даних
read(n);
for i:=1 to n do
 for j:=1 to n do
  begin
   read(a[i,j]);
   // визначення стартової вершини
  if a[i,j]=2 then
                begin
                 t:=1;          // кількість необроблених вершин
                 a[i,j]:=4;     // перефарбовування вершини, яка оброблена
                 magx[t]:=i; // поміщення до масиву 1 координати цієї вершини
                 magy[t]:=j; // поміщення до масиву 2 координати цієї вершини
                 p[t]:=0        // кількість ходів тури до цієї вершини
                end;
   // визначення фінішної вершини
if a[i,j]=3 then
                begin
                 finx:=i; // запам'ятовування 1 координати фінішної вершини
                 finy:=j; // запам'ятовування 2 координати фінішної вершини
                 a[i,j]:=0; // позначення цієї вершини як вільної
                end;
  end;
// оббивання меж дошки, щоб тура не вийшла за її межі
for i:=0 to n+1 do
 a[i,0]:=1;
for j:=0 to n+1 do
 a[0,j]:=1;
for i:=0 to n+1 do
 a[i,n+1]:=1;
for j:=0 to n+1 do
```

```
 a[n+1,j]:=1;
k:=0; // кількість оброблених вершин
while k<t do // поки кількість оброблених вершин менша за необроблених
 begin
  inc(k); // збільшення номера вершини, що буде оброблятися
  x:=magx[k]; // визначення 1 координати наступної для обробітку вершини
  y:=magy[k]; // визначення 2 координати наступної для обробітку вершини
  if (x=finx) and (y=finy) then   // якщо знайдена фінішна вершина
                          begin
                           writeln(p[k]); // виведення кількості кроків тури до фінішної вершини
                           halt; // вихід з програми
                          end;
  i:=1; // число кроків від координати оброблювальної вершини
  while (a[x+i,y]=0) or (a[x+i,y]=4) do // поки є вільна клітинка або оброблена
    begin
     if a[x+i,y]=0 then // якщо це клітинка ще не оброблена, то
               begin
                inc(t); // збільшення числа вершин, що потребують оброблення
                magx[t]:=x+i; // запис 1 координати до масиву
                magy[t]:=y;    // запис 2 координати до масиву
                a[x+i,y]:=4;  // позначення даної вершини як обробленої
                p[t]:=p[k]+1; // запам'ятовуємо кількість ходів тури до цієї вершини
               end;
     inc(i); // збільшуємо число кроків
    end;
  // аналогічно для інших трьох напрямків
i:=1;
  while (a[x-i,y]=0) or (a[x-i,y]=4) do
   begin
    if a[x-i,y]=0 then
               begin
                inc(t);
                magx[t]:=x-i;
                magy[t]:=y;
                a[x-i,y]:=4;
                p[t]:=p[k]+1;
               end;
    inc(i);
   end;
```

```
 j:=1;
 while (a[x,y+j]=0) or (a[x,y+j]=4) do
  begin
   if a[x,y+j]=0 then
              begin
               inc(t);
               magx[t]:=x;
               magy[t]:=y+j;
               a[x,y+j]:=4;
               p[t]:=p[k]+1;
              end;
   inc(j);
  end;
 j:=1;
 while (a[x,y-j]=0) or (a[x,y-j]=4) do
   begin
    if a[x,y-j]=0 then
              begin
               inc(t);
               magx[t]:=x;
               magy[t]:=y-j;
               a[x,y-j]:=4;
               p[t]:=p[k]+1;
              end;
   inc(j);
  end;
end;
 writeln(-1) // якщо програма не закінчилася, то фінішна клітинка не була знайдена
end.
```