**3 тура дистанционной олимпиады по программированию**

**для 10-11 классов**

1. Галиева Гузель Рамильевна, МАОУ СОШ №1, г. Агидель.

[guzel-galieva-2013@mail.ru](mailto:guzel-galieva-2013@mail.ru)

2. Pascal.ABC.NET 3.5.5.

2.1 **Задание 1**

Разработать программу, которая заполняет и выводит на экран таблицу истинности функции **(A \/ B) /\¬C**. Таблица состоит из значений логических переменных A, B, C только для тех строк, в которых значение функции равно введенному с клавиатуры значению **F**.

3.1

**program** logica;

**var** F : integer;

**begin**

write('Значение функции F = '); // ввод данных

read(F);

**if** F=1 **then begin** // вывод ответа при F=1

writeln('Значения переменных A, B, C: 0 1 0');

writeln('Значения переменных A, B, C: 1 0 0');

writeln('Значения переменных A, B, C: 1 1 0'); **end else**

**if** f=0 **then begin** // вывод ответа при F=0

writeln('Значения переменных A, B, C: 0 0 0');

writeln('Значения переменных A, B, C: 0 0 1');

writeln('Значения переменных A, B, C: 0 1 1');

writeln('Значения переменных A, B, C: 1 0 1');

writeln('Значения переменных A, B, C: 1 1 1');

**end else** // условия для неккоректных данных

writeln('Вы ввели некорректные данные!')

**end**.

4.1 **Пример ввода:**

Значение функции F = 1

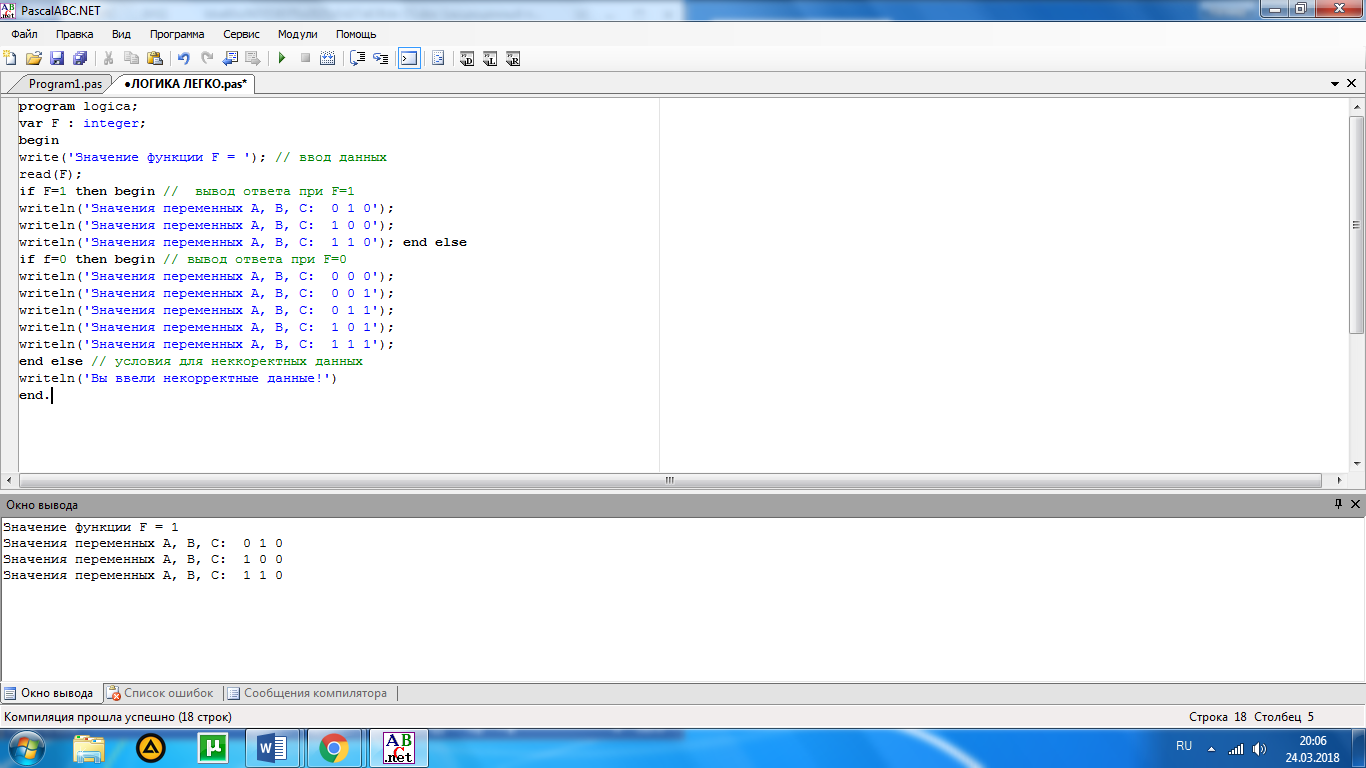
5.1 **Пример вывода:**

Значения переменных A, B, C: 0 1 0

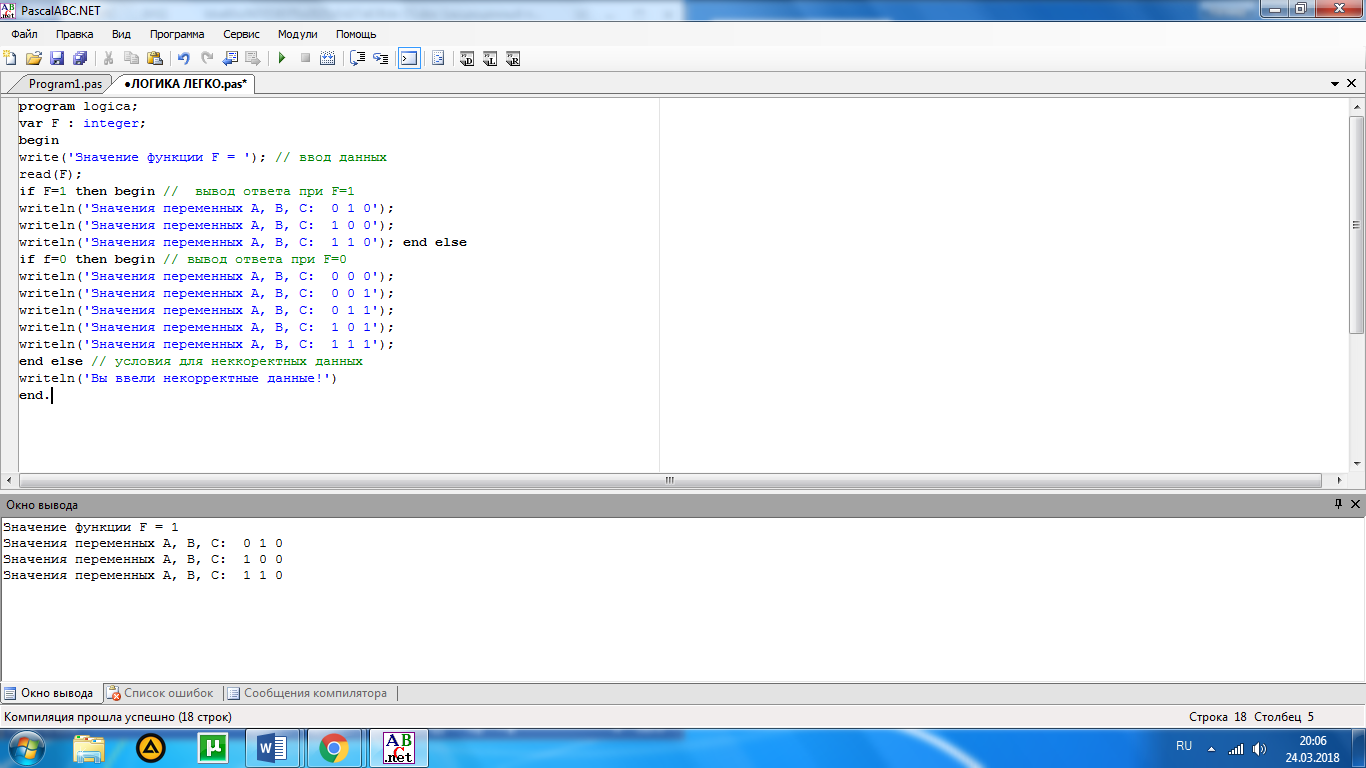
Значения переменных A, B, C: 1 0 0

Значения переменных A, B, C: 1 1 0 // решено правильно

6.1



7.1



2.4

**Задание 4**

Разработать программу, которая по заданным IP-адресу узла и маске подсети определяет десятичные значения байтов в адресе сети (сетевом адресе).

3.4

**PROGRAM** ip;

**VAR** b,u,q1,q2,q3,q4,g1,g2,g3,g4,k,m, n,i, n1,n2, c : integer;

x,y: real;

s1,s2,s, f1,f2,f3,f4,r1,r2,r3,r4,g,q, fr1,u1,u2,u3,u4,z,z1,z2,z3: string;

**begin** // ввод данных

write('Маска подсети: ');

readln(s2);

write('IP-адрес узла: ');

read(s1);

n:= pos('.', s1); // разделение чисел

q:=copy(s1,1,n-1);

val(q,q1,m);

delete(s1,1,n);

n:= pos('.', s1);

q:=copy(s1,1,n-1);

val(q,q2,m);

delete(s1,1,n);

n:= pos('.', s1);

q:=copy(s1,1,n-1);

val(q,q3,m);

delete(s1,1,n);

q:=s1;

val(q,q4,m);

n:= pos('.', s2);

g:=copy(s2,1,n-1);

val(g,g1,m);

delete(s2,1,n);

n:= pos('.', s2);

g:=copy(s2,1,n-1);

val(g,g2,m);

delete(s2,1,n);

n:= pos('.', s2);

g:=copy(s2,1,n-1);

val(g,g3,m);

delete(s2,1,n);

g:=s2;

val(g,g4,m);

**while** q1>0 **do begin** // перевод всех чисел в 2 с.с.

b:= q1 **mod** 2;

q1:= q1 **div** 2;

x:= x+b\*exp(ln(10)\*k);

k:= k+1;

**end**;

str(x,f1);

x:=0;

k:=0;

**while** q2>0 **do begin**

b:= q2 **mod** 2;

q2:= q2 **div** 2;

x:= x+b\*exp(ln(10)\*k);

k:= k+1;

**end**;

str(x,f2);

x:=0;

k:=0;

**while** q3>0 **do begin**

b:= q3 **mod** 2;

q3:= q3 **div** 2;

x:= x+b\*exp(ln(10)\*k);

k:= k+1;

**end**;

str(x,f3);

x:=0;

k:=0;

**while** q4>0 **do begin**

b:= q4 **mod** 2;

q4:= q4 **div** 2;

x:= x+b\*exp(ln(10)\*k);

k:= k+1;

**end**;

str(x,f4);

x:=0;

k:=0;

**while** g1>0 **do begin**

b:= g1 **mod** 2;

g1:= g1 **div** 2;

x:= x+b\*exp(ln(10)\*k);

k:= k+1;

**end**;

str(x,r1);

x:=0;

k:=0;

**while** g2>0 **do begin**

b:= g2 **mod** 2;

g2:= g2 **div** 2;

x:= x+b\*exp(ln(10)\*k);

k:= k+1;

**end**;

str(x,r2);

x:=0;

k:=0;

**while** g3>0 **do begin**

b:= g3 **mod** 2;

g3:= g3 **div** 2;

x:= x+b\*exp(ln(10)\*k);

k:= k+1;

**end**;

str(x,r3);

x:=0;

k:=0;

**while** q4>0 **do begin**

b:= g4 **mod** 2;

g4:= g4 **div** 2;

x:= x+b\*exp(ln(10)\*k);

k:= k+1;

**end**;

str(x,r4);

x:=0;

k:=0;

n:=0;

**if** f1='0' **then** // поразрядная конъюнкция для первх двух чисел

u1:='0' **else**

**if** r1='0' **then**

u1:='0'

**else begin**

**for** i:=1 **to** length(f1) **do**

n:=n+1;

**for** i:= 1 **to** length (r1) **do**

n1:=n1+1;

**if** n>n1 **then begin**

n2:=n-n1;

c:=n1;

Delete(f1,1,n2);

**end**;

**if** n1>n **then begin**

n2:=n1-n;

c:=n;

Delete(r1,1,n2);

**end**;

**if** n1=n **then**

c:=n;

fr1:=f1+r1;

**for** i:= 1 **to** c **do begin**

val(fr1[i],q1,m);

val(fr1[i+c],g1,m);

g1:= q1\*g1;

str(g1,f1);

u1:= u1+f1;

**end**;

val(u1,u,m); // перевод получившегося числа в 10 с.с.

**while** u>0 **do begin**

q1:= u **mod** 10;

u:= u **div** 10;

x:= x+q1\*exp(ln(2)\*k);

k:= k+1;

**end**;

str(x,u1);

**end**;

n:=0;

n1:=0;

c:=0;

x:=0;

k:=0;

**if** f2='0' **then** // аналогично для других пар

u2:='0' **else**

**if** r2='0' **then**

u2:='0'

**else begin**

**for** i:=1 **to** length(f2) **do**

n:=n+1;

**for** i:= 1 **to** length (r2) **do**

n1:=n1+1;

**if** n>n1 **then begin**

n2:=n-n1;

c:=n1;

Delete(f2,1,n2);

**end**;

**if** n1>n **then begin**

n2:=n1-n;

c:=n;

Delete(r2,1,n2);

**end**;

**if** n1=n **then**

c:=n;

fr1:=f2+r2;

**for** i:= 1 **to** c **do begin**

val(fr1[i],q1,m);

val(fr1[i+c],g1,m);

g1:= q1\*g1;

str(g1,f1);

u2:= u2+f1;

**end**;

val(u2,u,m);

**while** u>0 **do begin**

q1:= u **mod** 10;

u:= u **div** 10;

x:= x+q1\*exp(ln(2)\*k);

k:= k+1;

**end**;

str(x,u2);

**end**;

n:=0;

n1:=0;

c:=0;

x:=0;

k:=0;

**if** f3='0' **then**

u3:='0' **else**

**if** r3='0' **then**

u3:='0'

**else begin**

**for** i:=1 **to** length(f3) **do**

n:=n+1;

**for** i:= 1 **to** length (r3) **do**

n1:=n1+1;

**if** n>n1 **then begin**

n2:=n-n1;

c:=n1;

Delete(f3,1,n2);

**end**;

**if** n1>n **then begin**

n2:=n1-n;

c:=n;

Delete(r3,1,n2);

**end**;

**if** n1=n **then**

c:=n;

fr1:=f3+r3;

**for** i:= 1 **to** c **do begin**

val(fr1[i],q1,m);

val(fr1[i+c],g1,m);

g1:= q1\*g1;

str(g1,f1);

u3:= u3+f1;

**end**;

val(u3,u,m);

**while** u>0 **do begin**

q1:= u **mod** 10;

u:= u **div** 10;

x:= x+q1\*exp(ln(2)\*k);

k:= k+1;

**end**;

str(x,u3);

**end**;

n:=0;

n1:=0;

c:=0;

x:=0;

k:=0;

**if** f4='0' **then**

u4:='0' **else**

**if** r4='0' **then**

u4:='0'

**else begin**

**for** i:=1 **to** length(f4) **do**

n:=n+1;

**for** i:= 1 **to** length (r4) **do**

n1:=n1+1;

**if** n>n1 **then begin**

n2:=n-n1;

c:=n1;

Delete(f4,1,n2);

**end**;

**if** n1>n **then begin**

n2:=n1-n;

c:=n;

Delete(r4,1,n2);

**end**;

**if** n1=n **then**

c:=n;

fr1:=f4+r4;

**for** i:= 1 **to** c **do begin**

val(fr1[i],q1,m);

val(fr1[i+c],g1,m);

g1:= q1\*g1;

str(g1,f1);

u4:= u4+f1;

**end**;

val(u4,u,m);

**while** u>0 **do begin**

q1:= u **mod** 10;

u:= u **div** 10;

x:= x+q1\*exp(ln(2)\*k);

k:= k+1;

**end**;

str(x,u4);

**end**;

s:=u1+'.'+u2+'.'+u3+'.'+u4; // сбор всех получившегося чисел в сетевой адрес

writeln(s); // вывод ответа

**end**.

4.4

**Пример ввода:**

Маска подсети: 255.255.120.0

IP-адрес узла: 209.135.197.15 // хочу обратить внимание, что сначала вводиться

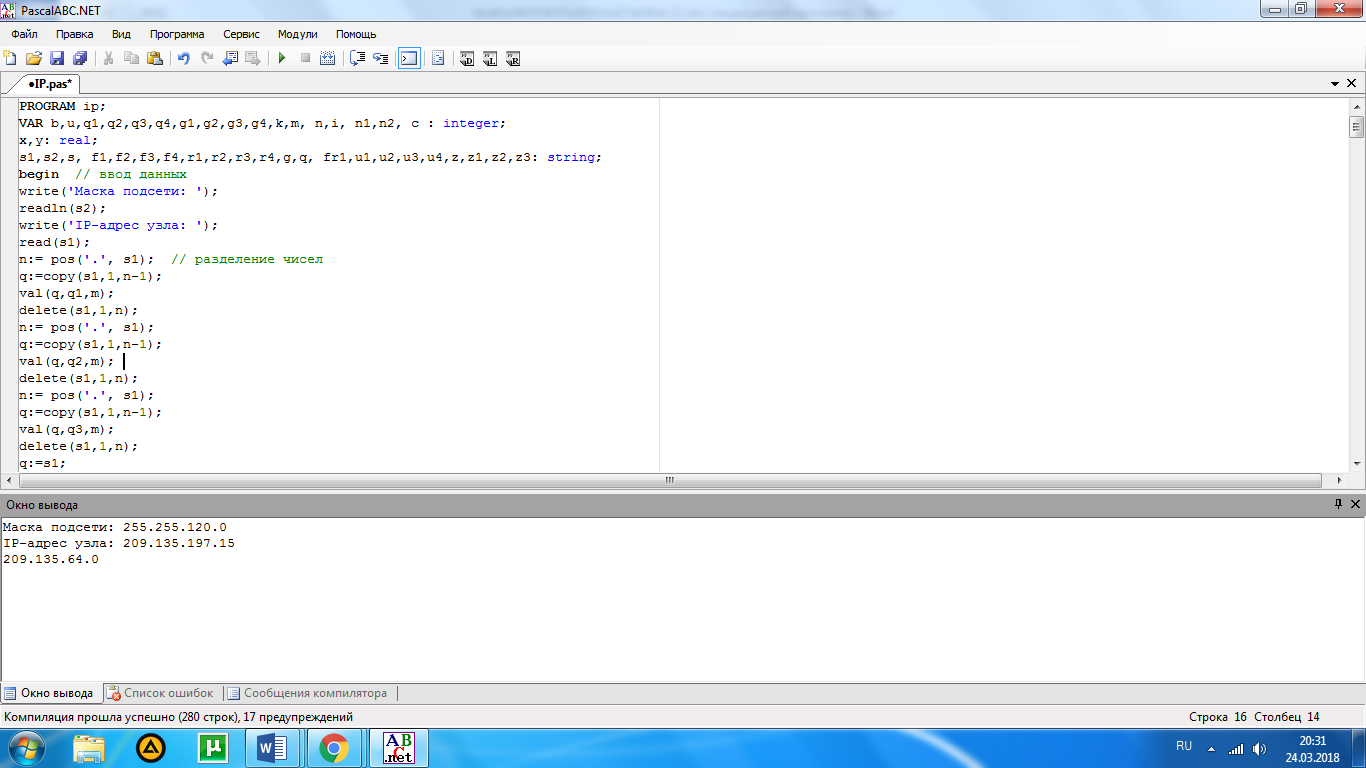
маска подсети, а после IP-адрес узла

5.4

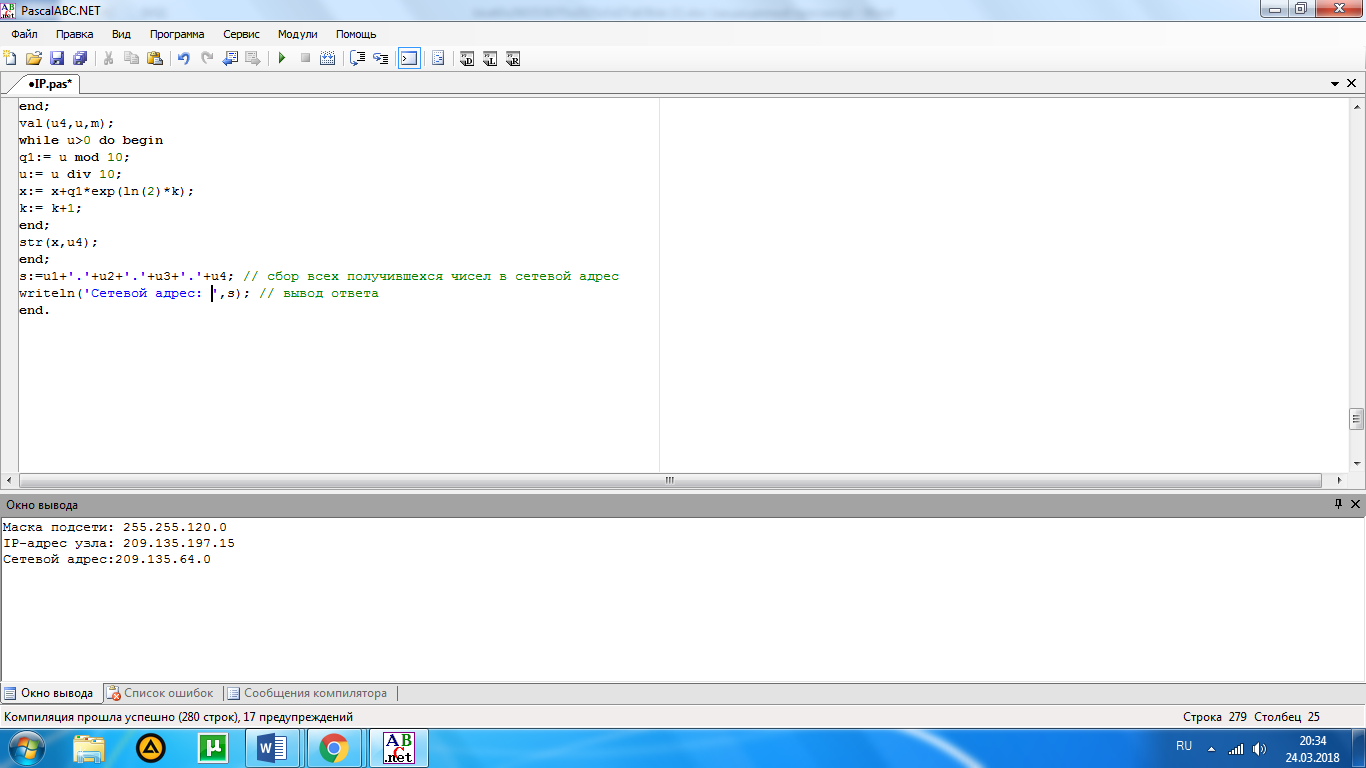
**Пример вывода:**

Сетевой адрес: 209.135.64.0 // решено верно

6.4



7.4



2.5

**Задание 5**

Разработать программу, которая определяет информационный объем всех результатов наблюдений влажности воздуха на метеорологической станции в байтах для некоторого фиксированного количества измерений. Результатом одного измерения влажности воздуха является целое число от 0 до 100 процентов, которое записывается при помощи минимально возможного количества бит.

3.5

**program** qqq;

**var** i: integer;

b: real;

**begin**

writeln('Количество измерений: '); // ввод данных

read(i);

b:=i\*7/8; //вычисление и вывод ответа

writeln('Информационный объем результатов наблюдений в байтах: ',b)

**end**.

4.5

**Пример ввода:**

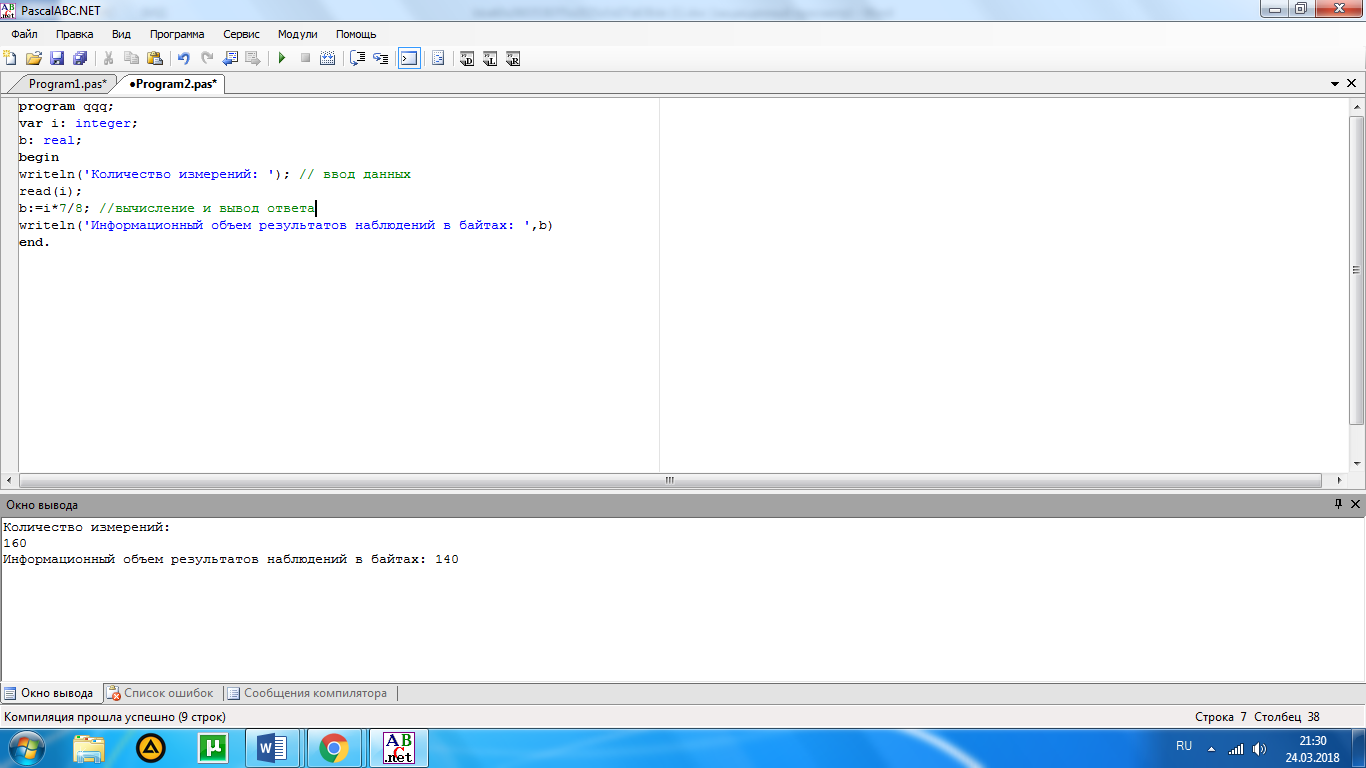
Количество измерений: 160

5.5

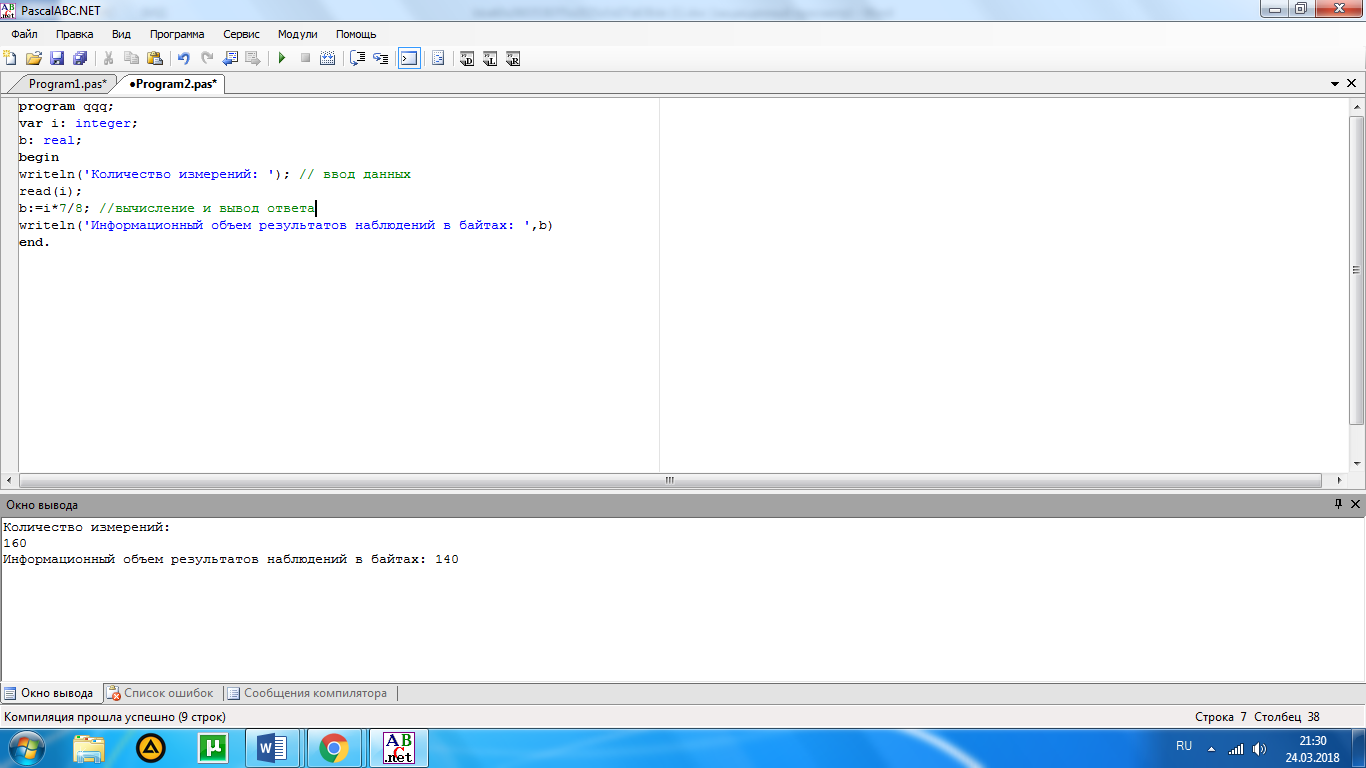
**Пример вывода:**

Информационный объем результатов наблюдений в байтах: 140

6.5



7.5



2.6

**Задание 6**

Разработать программу, которая подсчитывает количество нуклеотидов для заданной в программе строки. Подсчитать количество нуклеотидов A, C, G, T, разделив пробелами получаемые значения.

3.6

**program** chetnukly;

**var** s: string;

i, a,c,g,t: integer;

**begin** // ввод данных

writeln('Дана половинка ДНК, в строковой переменной в программе присвоено значение последовательности нуклеотидов:');

read(s);

**for** i:= 1 **to** length(s) **do begin** // вычисления

**if** s[i]='A' **then**

a:=a+1;

**if** s[i]='C' **then**

c:=c+1;

**if** s[i]='G' **then**

g:=g+1;

**if** s[i]='T' **then**

t:=t+1;

**end**;

writeln(a,' ',c,' ',g,' ',t) // вывод ответа

**end**.

4.6

**Пример**  **ввода:**

Дана половинка ДНК, в строковой переменной в программе присвоено значение последовательности нуклеотидов:

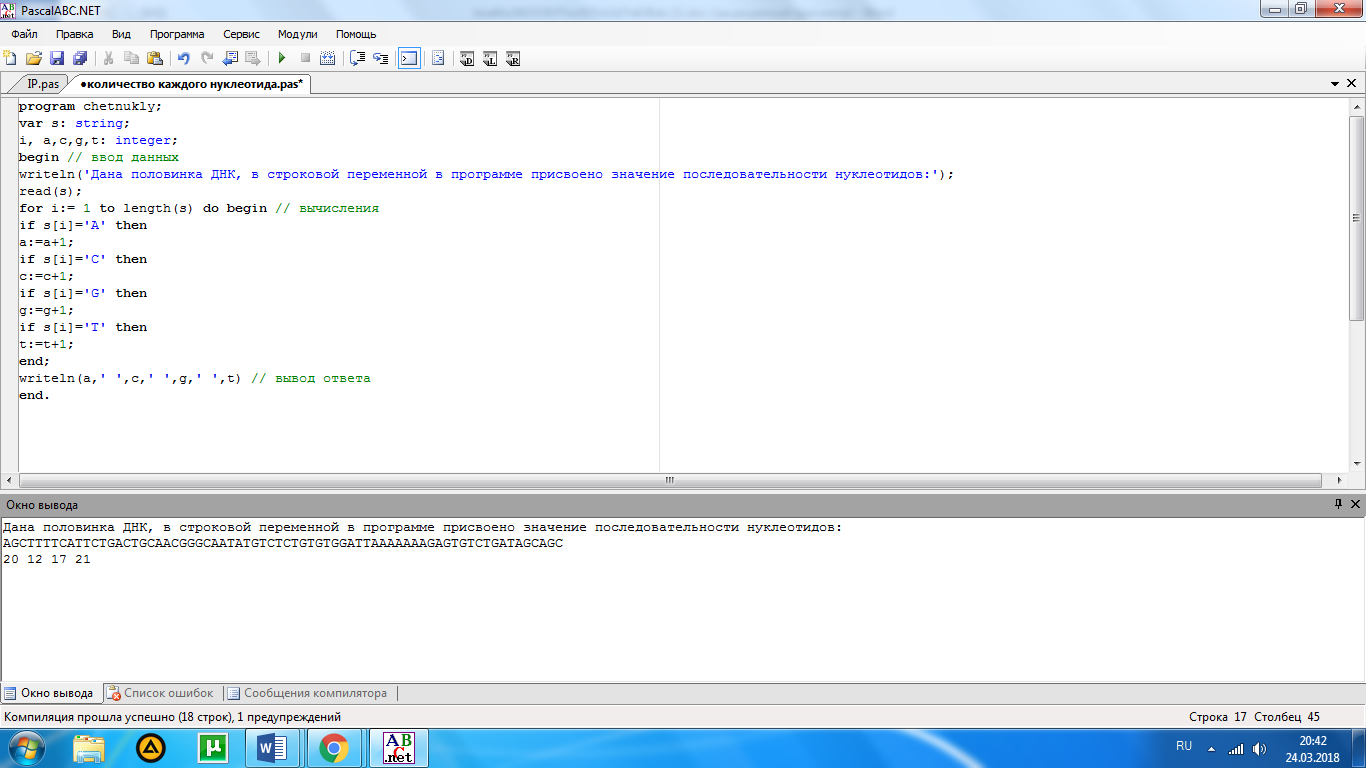
AGCTTTTCATTCTGACTGCAACGGGCAATATGTCTCTGTGTGGATTAAAAAAAGAGTGTCTGATAGCAGC

5.6

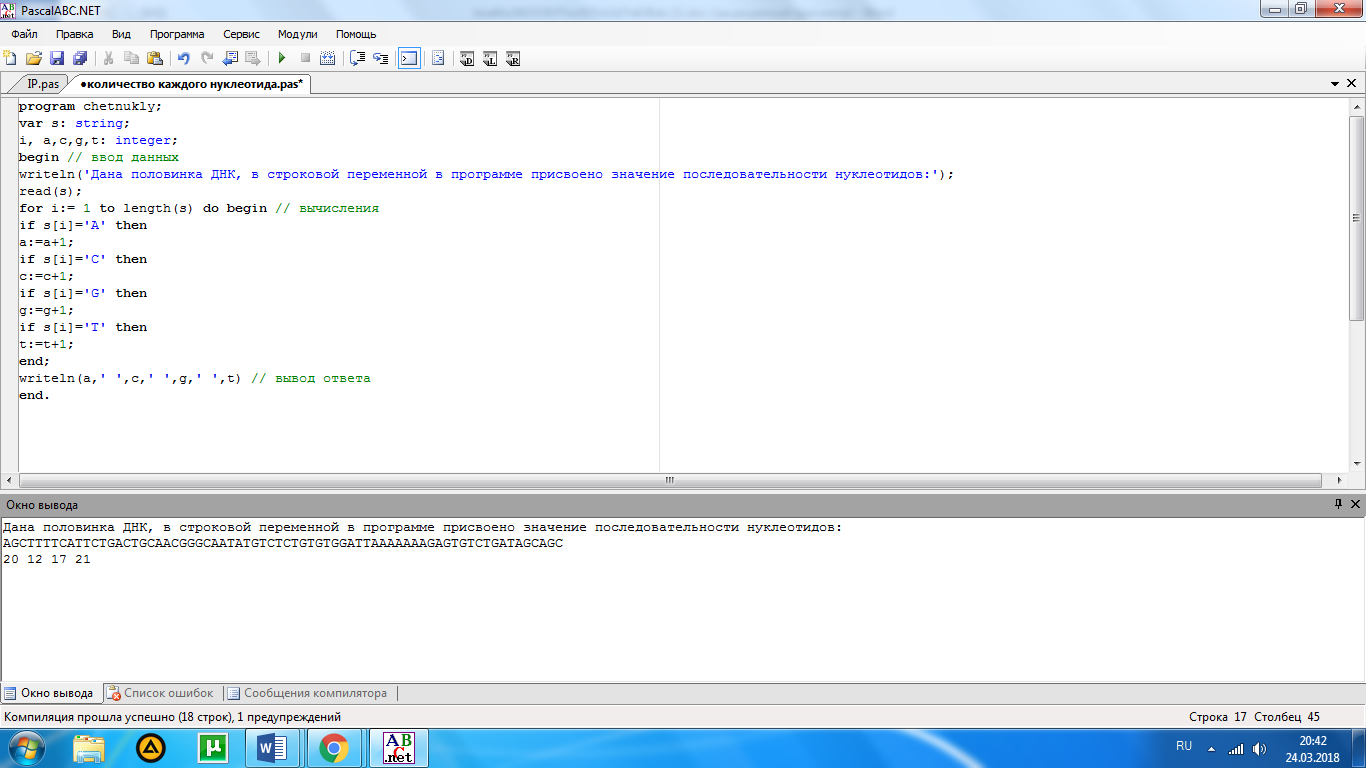
**Пример вывода:**

20 12 17 21 // решено абсолютно верно

6.6



7.6



2.7

**Задание 7**

Разработать программу, которая выполняет построение комплементарной цепочки ДНК для заданной строки ДНК: AAAACCCGGT. Для построение комплементарной цепочки ДНК нужно прочитать цепочку задом наперед, заменив аденин на тимин, а цитозин на гуанин.

3.7

**program** naoborot;

**var** s: string;

i: integer;

**begin** // ввод данных

writeln('Строковой переменной в программе присвоено значение: ');

read(s);

**for** i:= 1 **to** length(s) **do begin** // замена нуклеотидов

**if** s[i]='A' **then**

s[i]:='T' **else**

**if** s[i]='T' **then**

s[i]:='A' **else**

**if** s[i]='C' **then**

s[i]:='G'**else**

**if** s[i]='G' **then**

s[i]:='C';

**end**;

**for** i:= length(s) **downto** 1 **do** // печатание ответа наоборот

write(s[i]);

**end**.

4.7

**Пример ввода:**

Строковой переменной в программе присвоено значение:

AAAACCCGGT

5.7

**Пример вывода:**

ACCGGGTTTT

6.7



7.7



2.8

**Задание 8**

Разработать программу, которая выполняет подсчет количества точечных мутаций.

Даны две строки:

GAGCCTACTAACGGGAT

CATCGTAATGACGGCCT

Требуется найти расстояние Хэмминга, то есть количество различий в символах – несовпадающих элементов в строках.

В примере нахождения расстояние Хэмминга несовпадающие элементы выделены красным цветом:



3.8

**program** haming;

**var** str1, str2,s: string;

k,n,i: integer;

**begin** // ввод данных

writeln('Переменной str1 в программе присвоено значение: ');

readln(str1);

writeln('Переменной str2 в программе присвоено значение:');

readln(str2);

**for** i:= 1 **to** length(str1) **do** // вычисление длины строки

k:=k+1;

s:=str1+str2; // соединение двух строк

**for** i:= 1 **to** k **do begin** // нахождение расстояния

**if** s[i]<>s[i+k] **then begin**

n:=n+1;

**end**;

**end**;

writeln('Расстояние Хэмминга: ',n) // вывод ответа

**end**.

4.8

**Пример ввода:**

Переменной Str1 в программе присвоено значение:

GAGCCTACTAACGGGAT

Переменной Str1 в программе присвоено значение:

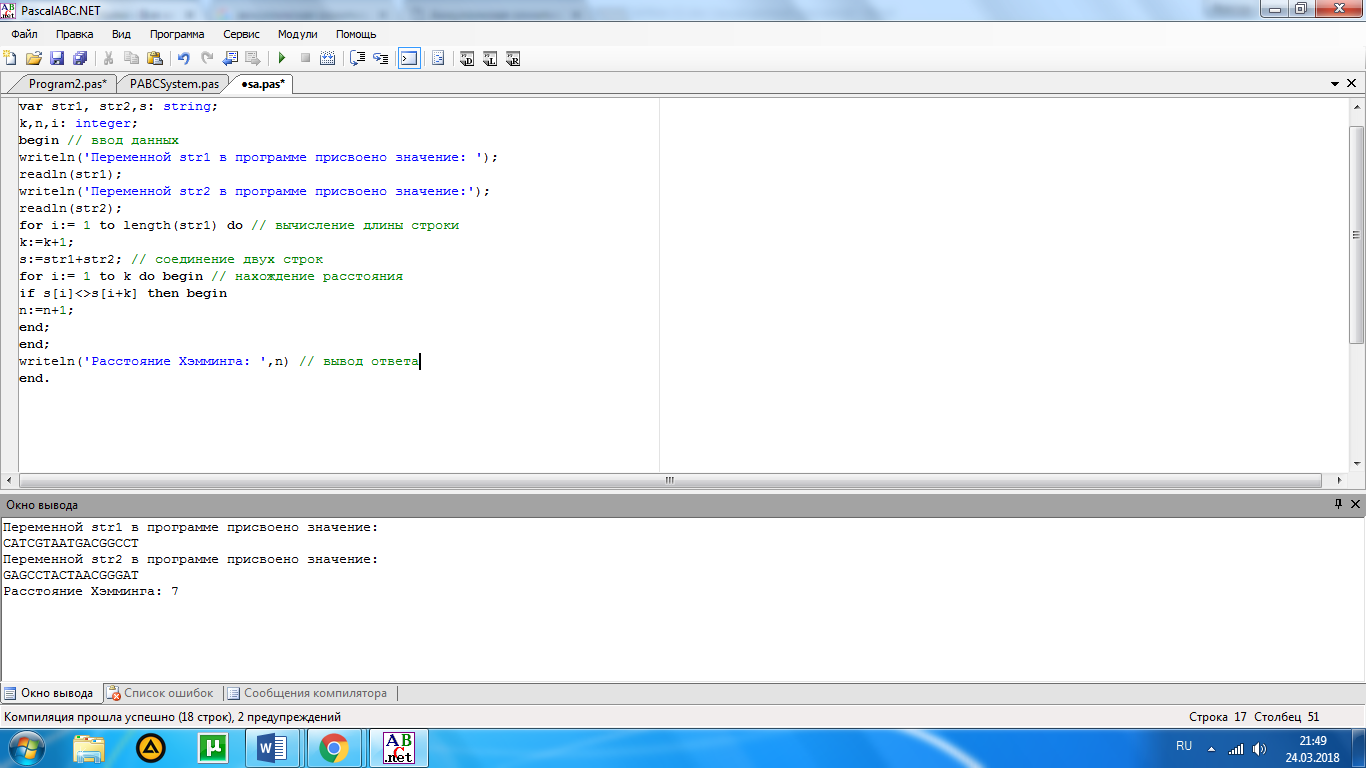
CATCGTAATGACGGCCT

5.8

**Пример вывода:**

Расстояние Хэмминга: **7 //** ответ верный

6.8



7.8

