

**МЕЖГОСУДАРСТВЕННЫЙ СТАНДАРТ**  
**ГОСУДАРСТВЕННАЯ СИСТЕМА ОБЕСПЕЧЕНИЯ ЕДИНСТВА ИЗМЕРЕНИЙ**

**Измерение расхода и количества жидкостей  
и газов методом переменного перепада давления**

**ПРОЦЕДУРА И МОДУЛЬ РАСЧЕТОВ. ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ**

**State system for ensuring the uniformity of measurements.  
Measurement of liquids and gases flow rate and quantity by differential  
pressure method. Procedure and module of calculations. Software**

МКС 17.020  
ОКСТУ 0008

Дата введения 1999-01-01

**Предисловие**

1 РАЗРАБОТАН фирмой "Газприборавтоматика" РАО "Газпром", Всероссийским научно-исследовательским центром стандартизации, информации и сертификации сырья, материалов и веществ (ВНИЦСМВ) Госстандарта России

ВНЕСЕН Госстандартом России

2 ПРИНЯТ Межгосударственным Советом по стандартизации, метрологии и сертификации (протокол № 11-97 от 25 апреля 1997 г.)

За принятие проголосовали:

Наименование государства	Наименование национального органа по стандартизации
Азербайджанская республика	Азгосстандарт
Республика Армения	Армгосстандарт
Республика Белоруссия	Госстандарт Белоруссии
Грузия	Грузстандарт
Республика Казахстан	Госстандарт Республики Казахстан
Киргизская Республика	Киргизстандарт
Республика Молдова	Молдовастандарт
Российская Федерация	Госстандарт России
Республика Таджикистан	Таджикгосстандарт
Туркменистан	Главная государственная инспекция Туркменистана

Республика Узбекистан	Узгосстандарт
-----------------------	---------------

3 ВВЕДЕН В ДЕЙСТВИЕ непосредственно в качестве государственного стандарта Российской Федерации с 1 января 1999 г. постановлением Государственного комитета Российской Федерации по стандартизации, метрологии и сертификации от 11 декабря 1997 г. № 410 межгосударственный стандарт ГОСТ 8.563.3-97

4 ВЗАМЕН ГОСТ 23720-79, ГОСТ 26969-86, РД 50-213-80, МИ 2204-92, МИ 2346-95

## Содержание

### [1. Область применения](#)

### [2. Нормативные ссылки](#)

### [3. Процедура расчета расхода и количества жидкостей и газов](#)

3.1 Исходные данные

3.2 Выходные данные

3.3 Листинг процедуры расчета расхода и количества жидкостей и газов

### [4. Модуль расчета погрешности определения расхода и количества жидкостей и газов](#)

4.1 Исходные данные

4.2 Выходные данные

4.3 Листинг модуля расчета погрешности определения расхода и количества жидкостей и

газов

## 1 ОБЛАСТЬ ПРИМЕНЕНИЯ

Настоящий стандарт устанавливает программное обеспечение расчета расхода и количества жидкостей и газов (далее - среда), а также расчета погрешностей определения расхода и количества сред.

Листинги программ, приведенные в настоящем стандарте, разработаны согласно требованиям, изложенным в ГОСТ 8.563.1 и ГОСТ 8.563.2.

## 2 НОРМАТИВНЫЕ ССЫЛКИ

В настоящем стандарте использованы ссылки на следующие стандарты:

ГОСТ 8.563.1-97 ГСИ. Измерение расхода и количества жидкостей и газов методом переменного перепада давления. Диафрагмы, сопла ИСА 1932 и трубы Вентури, установленные в заполненных трубопроводах круглого сечения. Технические условия

ГОСТ 8.563.2-97 ГСИ. Измерение расхода и количества жидкостей и газов методом переменного перепада давления. Методика выполнения измерений с помощью сужающих устройств

ГОСТ 30319.1-96 Газ природный. Методы расчета физических свойств. Определение физических свойств природного газа, его компонентов и продуктов его переработки

ГОСТ 30319.2-96 Газ природный. Методы расчета физических свойств. Определение коэффициента сжимаемости

ГОСТ 30319.3-96 Газ природный. Методы расчета физических свойств. Определение физических свойств по уравнению состояния

## 3 ПРОЦЕДУРА РАСЧЕТА РАСХОДА И КОЛИЧЕСТВА ЖИДКОСТЕЙ И ГАЗОВ

Процедура расчета расхода и количества жидкостей и газов (далее - процедура) написана на алгоритмическом языке ТУРБО ПАСКАЛЬ 7.0. Обращение к процедуре - QCALC.

### 3.1 Исходные данные

Исходные данные передаются в процедуру QCALC в виде глобальных параметров; для работы процедуры необходимо использовать модули Dos и Crt.

3.1.1 NNit - порядковый номер измерительного трубопровода.

3.1.2 NSubA[NNit] - номер среды (таблица 1).

Таблица 1

#### Наименование и номер среды

Среда	NSubA[NNit]
Природный газ	0
Метан	1
Этан	2
Пропан	3
Изобутан	4
н-Бутан	5
Изопентан	6
н-Пентан	7
Гексан	8
Гептан	9
Октан	10
Ацетилен	11
Этилен	12
Пропилен	13
Бензол	14
Толуол	15
Метанол	16
Метилмеркаптан	17
Моноксид углерода	18
Диоксид углерода	19

Сероводород	20
Диоксид серы	21
Водяной пар	22
Вода	23
Кислород	24
Воздух	25
Гелий	26
Неон	27
Аргон	28
Водород	29
Аммиак	30
Азот	31

3.1.3 NMethKA[NNit] - номер метода расчета коэффициента сжимаемости природного газа по ГОСТ 30319.2 (таблица 2).

Таблица 2

#### Наименование и номер метода расчета

Метод расчета	NMethKA[NNit]
NX19 мод.	0
GERG-91 мод.	1
AGA8-92DC	2
ВНИЦСМВ	3

3.1.4 NSuzA[NNit] - номер типа сужающего устройства (таблица 3).

Таблица 3

#### Наименование и номер типа сужающего устройства

Тип сужающего устройства	NSuzA[NNit]
Диафрагма	0
Сопло ИСА 1932	1
Сопло Вентури	2
Труба Вентури (литой входной конус)	3
Труба Вентури (обработанный входной конус)	4

Труба Вентури (сварной входной конус)	5
---------------------------------------	---

3.1.5 YR - массив концентраций 16 компонентов природного газа, молярные доли (таблица 4).

Таблица 4

**Наименование компонента и его порядковый номер в массиве**

Наименование компонента	Порядковый номер
Метан	1
Этан	2
Пропан	3
н-Бутан	4
Изобутан	5
н-Пентан	6
Изопентан	7
Гексан	8
Гептан	9
Октан	10
Азот	11
Диоксид углерода	12
Сероводород	13
Гелий	14
Моноксид углерода	15
Водород	16

3.1.6 Параметры среды

$\rho_{oc}$  - плотность природного газа при стандартных условиях, кг/м<sup>3</sup>.

(стандартные условия:  $P_c = 1,01325$  бар,  $t_c = 20$  °C)\*.

\* Условные обозначения параметров при стандартных условиях - по ГОСТ 8.563.1 и ГОСТ 8.563.2.

Y<sub>a</sub> - концентрация азота в природном газе, молярные доли.

Y<sub>y</sub> - концентрация диоксида углерода в природном газе, молярные доли.

P - давление среды, бар.

T - температура среды, °C.

### 3.1.7 Характеристики сужающего устройства

AlfaSU - температурный коэффициент линейного расширения материала сужающего устройства,  $1/^\circ\text{C}$ .

Dd20 - диаметр отверстия сужающего устройства при  $20^\circ\text{C}$ , мм.

Rn - начальный радиус закругления входной кромки диафрагмы, мм.

TaiP - период поверки диафрагмы, год.

Dr - перепад давления на сужающем устройстве, бар.

### 3.1.8 SodSuA[NNit] - номер способа отбора давления на диафрагме:

1) 0 - угловой;

2) 1 - фланцевый;

3) 2 - трехрадиусный.

### 3.1.9 Характеристики измерительного трубопровода

AlfaT - температурный коэффициент линейного расширения материала измерительного трубопровода,  $1/^\circ\text{C}$ .

Dt20 - внутренний диаметр измерительного трубопровода при  $20^\circ\text{C}$ , мм.

Rsh - эквивалентная шероховатость материала измерительного трубопровода, мм.

### 3.1.10 TauAv - время, за которое определяют количество среды, ч.

### 3.1.11 VarRoA[NNit] - номер способа определения плотности природного газа при рабочих условиях:

1) 0 - плотность рассчитывают по ГОСТ 30319.1;

2) 1 - плотность измеряют. Rot - измеренное значение плотности.

## 3.2 Выходные данные

Ro - плотность среды при рабочих условиях,  $\text{кг}/\text{м}^3$ .

KZ - коэффициент сжимаемости среды.

Карра - показатель адиабаты среды.

Mu - динамическая вязкость среды,  $\text{мкПа}\cdot\text{с}$ .

Qc - объемный расход среды, приведенный к стандартным условиям,  $\text{м}^3/\text{ч}$ .

Vc - объем среды (количество среды, выраженное в кубических метрах), приведенный к стандартным условиям,  $\text{м}^3$ .

Vm - масса среды (количество среды, выраженное в тоннах), т.

Hs[1] - высшая удельная теплота сгорания среды,  $\text{МДж}/\text{м}^3$ .

Hs[2] - низшая удельная теплота сгорания среды,  $\text{МДж}/\text{м}^3$ .

### 3.3 Листинг процедуры расчета расхода и количества жидкостей и газов

В нижеприведенной процедуре вызываются две внешние программы:

1) TpNg.exe - расчет теплофизических свойств природного газа в соответствии с требованиями ГОСТ 30319.2 и ГОСТ 30319.3;

2) TpSubs.exe - расчет теплофизических свойств компонентов природного газа и продуктов его переработки в соответствии с требованиями ГОСТ 30319.1.

Типы используемых переменных: Fl: text; NNit: byte; Dd, Dt, Dd20, Dt20, RSh, Rn, TauP, AlfaT, AlfaSU, Roc, Ya, Yy, Dp, P, T, Ro, Rot, Mu, Kappa, KZ, Eps, KSh, Kk, Cb, KRe, Re, Vc, Vm, TauAv: real; NSubA, NSuzA, SodSuA, NMethKA, VarRoA: array[1..30] of byte; YR: array[1..16] of real; Hs: array[1..2] of real;

```
Procedure Qcalc;

var
  I, IBeg, IFin: byte; Code: integer;
  Bet, Bet4, Ec, Rd, Psi, Rk, L1, L2, Alfa, Qcb, ARe, R0, KCb, Qc, Vcv, Log: real;
  HsS: string [10];

label
  1, 3;

const
  RocSubs: array[1..31] of real = (0.6682, 1.2601, 1.8641, 2.488,
    2.4956, 3.147, 3.174, 3.898, 4.755,
    5.812, 1.09, 1.1733, 1.776, 3.469,
    4.294, 1.587, 2.045, 1.1649, 1.8393,
    1.4311, 2.718, 0.787, 998.23,
    1.33116, 1.20445, 0.16631, 0.8385,
    1.6618, 0.08375, 0.716, 1.1649);
  HsSubs1: array[1..31] of real = (37.12, 65.43, 93.85, 122.8, 123.6, 0.0,
    0.0, 0.0, 0.0, 0.0, 54.47, 59.04, 86.88,
    0.0, 0.0, 0.0, 52.70, 11.77, 0.0, 23.61,
    0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
    11.88, 16.11, 0.0);
  HsSubs2: array[1..31] of real = (33.43, 59.87, 86.37, 113.4, 114.1, 0.0,
    0.0, 0.0, 0.0, 0.0, 52.62, 55.34, 81.29,
    0.0, 0.0, 0.0, 48.94, 11.77, 0.0, 21.75,
    0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0,
    10.04, 13.32, 0.0);

  CalcTpNg = 'TpNg.exe'; CalcTpSubs = 'TpSubs.exe';

begin { QCalc }

{ Расчет физических свойств среды }
assign(Fl, 'IRD'); rewrite(Fl);
if NSubA[NNit]=0 then begin
  writeln(Fl, NMethKA[NNit]);

  if NMethKA[NNit] >= 2 then begin
    IBeg:= 1;
    repeat
      IFin:= IBeg+3;
      for I:= IBeg to IFin do write(Fl, YR[I]:14, BL);
```

```

        writeln(Fl);
        IBeg:=IFin+1
        until IBeg > 16;
    end
        else
            writeln(Fl,Roc:14,Bl,Ya:14,Bl,Yy:14);
    end
        else
            writeln(Fl,NSubA[NNit]);
            writeln(Fl,P:14,Bl,T:14);
            close(Fl);
            TextColor(7);
            gotoxy(19,9);
            writeln(' |-----| ');
            gotoxy(19,10);
            writeln(' | ');
            gotoxy(19,11);
            write(' | ');
            TextColor(135);
            write('Ж Д И Т Е');
            TextColor(7);
            writeln(' | ');
            gotoxy(19,12);
            writeln(' | ');
            gotoxy(19,13);
            writeln(' | В Ы П О Л Н Я Е Т С Я Р А С Ч Е Т | ');
            gotoxy(19,14);
            writeln(' | ');
            gotoxy(19,15);
            writeln(' |_____| ');
            if NSubA[NNit]=0 then begin
                gotoxy(21,12);
                swapvectors; exec(CalcTpNg,CalcTpNg); swapvectors;
                TextColor(7);
                gotoxy(19,9);
                writeln('-----');
                gotoxy(19,10);
                writeln(' | ');
                gotoxy(19,11);
                write(' | ');
                TextColor(135);
                write('Ж Д И Т Е');
                TextColor(7);
                writeln(' | ');
                gotoxy(19,12);
                writeln(' | ');
                gotoxy(19,13);
                writeln(' | В Ы П О Л Н Я Е Т С Я Р А С Ч Е Т | ');
                gotoxy(19,14);
                writeln(' | ');
                gotoxy(19,15);
                writeln(' |_____| ');
            end
            else begin
                swapvectors; exec(CalcTpSubs,CalcTpSubs); swapvectors;
                Roc:= RocSubs[NSubA[NNit]]
            end;
            assign(Fl,'IRD'); reset(Fl);
            if(NSubA[NNit]=0) and (NMethKA[NNit] >= 2) then

```



```

        readln(Fl,Roc);
if NSubA[NNit]=0 then begin
    readln(Fl,Hs[1],Hs[2]);
    for I:= 1 to 2 do begin
        Str(Hs[I]:10,HsS); Val(HsS,Hs[I],Code)
        end;
end
                                else begin
Hs[1]:= HsSubs1[NSubA[NNit]]; Hs[2]:= HsSubs2[NSubA[NNit]]
end;
readln(Fl,Ro,Kappa,Mu);
close(Fl); erase(Fl);
if (NSubA[NNit]=0) and (VarRoA[NNit]=1) then Ro:= Rot;
KZ:= P*Roc*293.15/Ro/(T+273.15)/1.01325;
if NSubA[NNit]=0 then str(Roc:6:4,RocStr);

{ Расчет: 1)диаметров сужающего устройства и измерительного трубопровода при
рабочей температуре; 2)относительного диаметра; 3)коэффициента скорости входа }
Dd:= (1.0+AlfaSU*(T-20.0))*Dd20;
Dt:= (1.0+AlfaT*(T-20.0))*Dt20;
Bet:= Dd/Dt; Bet4:= sqr(Bet)*sqr(Bet);
Ec:= 1.0/sqrt(1.0-Bet4);

{ Расчет коэффициента расширения }
Eps:= 1.0;
if NSubA[NNit] <> 23 then begin
    if NSuzA[NNit]=0 then
        Eps:= 1.0-(0.41+0.35*Bet4)*Dp/P/Kappa;
if NSuzA[NNit] <> 0 then begin
    Psi:= 1.0-Dp/P;
    Eps:= Kappa*r_(Psi,2.0/Kappa)/(Kappa-1.0)*
        (1.0-Bet4)/(1.0-Bet4*r_(Psi,2.0/Kappa))*
        (1.0-r_(Psi,(Kappa-1.0)/Kappa))/
        (1.0-Psi);
    Eps:= sqrt(Eps)
    end;
end;

{ Расчет поправочного коэффициента на шероховатость внутренней поверхности
измерительного трубопровода без учета числа Рейнольдса }
KSh:= 1.0;
if (NSuzA[NNit] <= 2) and (RSh <> 0.0) then begin
    ARe:= 0.5; Rd:= RSh/Dt; Log:= Ln(Rd*1.e4)/2.3026;
    if NSuzA[NNit]=0 then begin
        if Log <= (1.0/10.0/Bet4+8.0)/14.0 then begin
            R0:=0.0; goto 1;
        end;
        R0:= 0.07*Log-0.04
    end
    else begin
        if Log <= (1.0/10.0/Bet4+5.0)/9.0 then begin
            R0:= 0.0; goto 1;
        end;
        R0:= 0.045*Log-0.025
    end;
KSh:= 1.0+Bet4*R0*ARE
end;
1:;

```

```

{ Расчет поправочного коэффициента на притупление входной кромки отверстия
диафрагмы }
Kk:= 1.0;
if (NSuzA[NNit]=0) and (Dd20<125.0) then begin
  if TauP=0.0 then Rk:= Rn;
  if TauP=1.0 then Rk:= 0.0292+0.85*Rn;
  if (TauP <> 0.0) and (TauP <> 1.0) then
    Rk:= 0.195-(0.195-Rn)*(1.0-Exp(-TauP/3.0))*3.0/TauP;
  Kk:= 1.0547-0.0575*Exp(-149.0*Rk/Dd)
end;

{ Расчет коэффициента истечения при числе Рейнольдса, стремящемся к
бесконечности }
if NSuzA[NNit]=0 then begin
  L1:= 0.0; L2:= 0.0;
  if SodSuA[NNit]=1 then begin
    L1:= 25.4/Dt; L2:= L1;
    if L1 >= 0.4333 then L1:= 0.039 else L1:= 0.09*L1
  end;
  if SodSuA[NNit]=2 then begin L1:= 0.039; L2:= 0.47 end;
  Cb:= 0.5959+0.0312*r_(Bet,2.1)-0.184*sqr(Bet4)+
    L1*Bet4/(1.0-Bet4)-0.0337*L2*r_(Bet,3)
end;
if NSuzA[NNit]=1 then Cb:= 0.99-0.2262*r_(Bet,4.1);

{ Для сопла Вентури Cb = C, так как KRe = 1 }
if NSuzA[NNit]=2 then Cb:= 0.9858-0.196*r_(Bet,4.5);
{ Для труб Вентури Cb = C = const, так как KRe = 1 и Re > 2.e5 }
case NSuzA[NNit] of
  3: Cb:= 0.984;
  4: Cb:= 0.995;
  5: Cb:= 0.985
end;

{ Расчет коэффициента расхода сужающего устройства и расхода при числе
Рейнольдса, стремящемся к бесконечности }

Alfa:= Cb*Ec;
Qcb:= 0.039986*Alfa*KSh*Kk*Eps*sqr(Dd)*
  sqrt(1.e3*Dp*Ro)/Roc;
Re:= 4.e6*Qcb*Roc/3.6/3.141592653/Mu/Dt;

{ Расчет поправочного коэффициента на число Рейнольдса }
case NSuzA[NNit] of
  0: KRe:= 1.0+1.426/(1.0+Cb*r_(Re,0.75)/64.28/r_(Bet,2.5));
  1: KRe:= 1.0+0.86/(1.0+Cb*r_(Re,1.15)/923.9/sqr(Bet)/(33.0*
    r_(Bet,2.15)-17.5));
  2: KRe:= 1.0;
  3: KRe:= 1.0;
  4: KRe:= 1.0;
  5: KRe:= 1.0;
end;

{ Определение действительного значения числа Рейнольдса }
Re:= Re*KRe;

{ Расчет поправочного коэффициента на шероховатость внутренней поверхности
измерительного трубопровода с учетом числа Рейнольдса для всех сужающих
устройств, кроме труб Вентури }

```

```

if (NSuzA[NNit] <= 2) and (RSh <> 0.0) then begin
  Qcb:= Qcb/KSh;
  if Re > 1.e4 then begin
    if Re < 1.e6 then ARe:= 1.0-sqr(Ln(Re)/2.3026-6.0)/4.0
      else ARe:= 1.0;
    KSh:= 1.0+Bet4*R0*ARe
  end;
  if (Re <= 1.e4) or (KSh <= 1.0005) then KSh:= 1.0;
  Qcb:= Qcb*KSh
end;

{ Определение коэффициента истечения для труб Вентури в зависимости от
числа Рейнольдса }
if (Re < 2.e5) and (NSuzA[NNit] > 2) then
  case NSuzA[NNit] of
    3: begin
      if Re <= 6.e4 then Cb:= 0.957;
      if (Re > 6.e4) and (Re <= 1.e5) then Cb:= 0.966;
      if (Re > 1.e5) and (Re <= 1.5e5) then Cb:= 0.976;
      if Re > 1.5e5 then Cb:= 0.982
    end;
    4: begin
      if Re <= 4.e4 then Cb:= 0.970;
      if (Re > 4.e4) and (Re <= 8.e4) then Cb:= 0.977;
      if (Re > 8.e4) and (Re <= 1.2e5) then Cb:= 0.992;
      if Re > 1.2e5 then Cb:= 0.998
    end;
    5: begin
      if Re <= 6.e4 then Cb:= 0.960;
      if (Re > 6.e4) and (Re <= 1.e5) then Cb:= 0.970;
      if Re > 1.e5 then Cb:= 0.980
    end;
  end;
KCb:= 1.0;

{ Определение поправки на коэффициент истечения для труб Вентури в
зависимости от числа Рейнольдса }
if (Re < 2.e5) and (NSuzA[NNit] > 2) then
  case NSuzA[NNit] of
    3: KCb:= Cb/0.984;
    4: KCb:= Cb/0.995;
    5: KCb:= Cb/0.985;
  end;

{ Расчет расхода и количества среды при действительном значении числа
Рейнольдса }
Qc:= Qcb*KRe*KCb; Vcv:= Qc*TauAv; Vc:= Vcv;
if (NSubA[NNit]=22) or (NSubA[NNi]=23) then
  Vm:= Vcv*Roc/1000.0;
end; { QCalc }
{-----}
function r_(A,R: real): real;
function r_/ begin r_:= exp(R*ln(A)) end;

```

#### 4 МОДУЛЬ РАСЧЕТА ПОГРЕШНОСТИ ОПРЕДЕЛЕНИЯ РАСХОДА И КОЛИЧЕСТВА ЖИДКОСТЕЙ И ГАЗОВ

Все процедуры и функции модуля расчета погрешности определения расхода и количества жидкостей и газов (далее - модуль ERRQSNX) написаны на алгоритмическом языке ТУРБО ПАСКАЛЬ 7.0. Обращение к модулю ERRQSNX осуществляется с помощью оператора вызова основной процедуры QS.

#### 4.1 Исходные данные

Исходные данные передаются в процедуры модуля ERRQSNX в виде глобальных параметров; для работы процедур модуля необходимо использовать модули Dos и Crt.

4.1.1 NSubA[NNit], NSuzA[NNit], NMethKA[NNit], SodSuA[NNit], VarRoA[NNit], Dd20, Dt20, AlfaT, AlfaSU, RSh, Rn, TauP, TauAv - см. 3.1.

4.1.2 VarPA[NNit] - номер варианта измерения давления:

- 1) 0 - измеряют избыточное давление;
- 2) 1 - измеряют абсолютное давление.

4.1.3 NVarYA[NNit] - номер варианта задания концентраций компонентов природного газа:

- 1) 0 - задание полного компонентного состава (см. 3.1.5);
- 2) 1 - задание концентраций азота и диоксида углерода.

4.1.4 Характеристики гидравлических сопротивлений

AmountR - количество гидравлических сопротивлений до сужающего устройства (от 2 до 9).

NRA[J] - массив номеров, характеризующих тип гидравлических сопротивлений, которые расположены до сужающего устройства ( $1 \leq J \leq 9$ ) (таблица 5).

Таблица 5

#### Наименование (тип) гидравлических сопротивлений и их номер

Наименование (тип) гидравлического сопротивления	Номер
Пробковый кран	1
Запорный клапан (вентиль)	2
Затвор (заслонка)	3
Конфузор, сужение	4
Диффузор, расширение	5
Отвод (колено), тройник	6
Струевыпрямитель	7
Симметричное резкое сужение	8
Симметричное резкое расширение	9
Задвижка, равнопроходный шаровой кран	10
Группа колен в одной плоскости или разветвляющийся поток	11

Группа колен в разных плоскостях или смешивающиеся потоки	12
Гильза термометра, плотномера или карман диаметром $\leq 0,03 D$	13
Гильза термометра, плотномера или карман диаметром $\leq 0,13 D$	14
Сопротивление неопределенного типа	15

LRA[J] - отношение расстояния от сужающего устройства до гидравлического сопротивления (включая гидравлическое сопротивление за сужающим устройством) к диаметру измерительного трубопровода ( $1 \leq J \leq 10$ ).

#### 4.1.5 Характеристики сужающего устройства

DAC[1] - толщина материала диафрагмы, мм.

DAC[2] - предел текучести материала диафрагмы при 20 °С, МПа.

DAC[3] - модуль Юнга материала диафрагмы при 20 °С, МПа.

#### 4.1.6 Характеристики измерительного трубопровода

TAC[1] - эксцентриситет между осями измерительного трубопровода и сужающего устройства, мм.

TAC[2] - расстояние от уступа до отверстия для отбора давления, мм.

TAC[3] - высота уступа между двумя секциями измерительного трубопровода, мм.

#### 4.1.7 Характеристики измерительного комплекса (измерение перепада давления)

SIZMDPD - номер варианта состава измерительного комплекса:

- 1) 0 - средство измерений с линейной функцией преобразования;
- 2) 1 - средство измерений с квадратичной функцией преобразования;
- 3) 2 - измерительный преобразователь и средство измерений с линейной функцией преобразования;
- 4) 3 - измерительный преобразователь и средство измерений с квадратичной функцией преобразования;
- 5) 4 - измерительный преобразователь, средство измерений с линейной функцией преобразования и планиметр (или интегратор) с линейной функцией преобразования;
- 6) 5 - измерительный преобразователь, средство измерений с линейной функцией преобразования и планиметр (или интегратор) с квадратичной функцией преобразования;
- 7) 6 - измерительный преобразователь, средство измерений с квадратичной функцией преобразования и планиметр (или интегратор) с линейной функцией преобразования;
- 8) 7 - измерительный преобразователь, корнеизвлекающий преобразователь, средство измерений с линейной функцией преобразования и планиметр (или интегратор) с линейной функцией преобразования.

EDPA[J] - массив погрешностей измерительного комплекса (таблицы 6, 7, 8).

Таблица 6

### Наименование погрешности средства измерений или измерительного преобразователя и ее номер (J) в массиве EDPA[J]

Наименование погрешности	J
--------------------------	---

Систематическая составляющая, %	1
Случайная составляющая, %	2
Класс точности, %	3
Линейность, %	4
Гистерезис, %	5
Повторяемость, %	6
Стабильность, %	7
Удельная температурная погрешность, %/°C	8
Удельная погрешность, обусловленная изменением напряжения, %/В	9
Дополнительная погрешность, обусловленная вибрацией, %	10
Дополнительная погрешность, обусловленная изменением сопротивления, %	11
Удельная погрешность, обусловленная изменением статического давления, %/бар	12

Таблица 7

**Наименование погрешности средства измерений или корнеизвлекающего преобразователя и ее номер (J) в массиве EDPA[J]**

Наименование погрешности	J
Систематическая составляющая, %	13
Случайная составляющая, %	14
Класс точности, %	15
Линейность, %	16
Гистерезис, %	17
Повторяемость, %	18
Стабильность, %	19
Удельная температурная погрешность, %/°C	20
Дополнительная погрешность, обусловленная изменением напряжения, %	21
Дополнительная погрешность, обусловленная вибрацией, %	22

Таблица 8

**Наименование погрешности средства измерений и ее номер (J) в массиве EDPA[J]**

Наименование погрешности	J

Систематическая составляющая, %	23
Случайная составляющая, %	24
Класс точности, %	25
Линейность, %	26
Гистерезис, %	27
Повторяемость, %	28
Стабильность, %	29
Удельная температурная погрешность, %/°C	30
Дополнительная погрешность, обусловленная изменением напряжения, %	31
Дополнительная погрешность, обусловленная вибрацией, %	32

EDPA[33] - систематическая составляющая погрешности планиметра (или интегратора), %.

EDPA[34] - случайная составляющая погрешности планиметра (или интегратора), %.

EDPA[35] - верхний предел измерения перепада давления, бар.

#### 4.1.8 Характеристики измерительного комплекса (измерение давления)

SIZMPD - номер варианта состава измерительного комплекса:

- 1) 0 - средство измерений с линейной функцией преобразования;
- 2) 1 - средство измерений с квадратичной функцией преобразования;
- 3) 2 - измерительный преобразователь и средство измерений с линейной функцией преобразования;
- 4) 3 - измерительный преобразователь и средство измерений с квадратичной функцией преобразования;
- 5) 4 - измерительный преобразователь, средство измерений с линейной функцией преобразования и планиметр (или интегратор) с линейной функцией преобразования;
- 6) 5 - измерительный преобразователь, средство измерений с линейной функцией преобразования и планиметр (или интегратор) с квадратичной функцией преобразования;
- 7) 6 - измерительный преобразователь, средство измерений с квадратичной функцией преобразования и планиметр (или интегратор) с линейной функцией преобразования.

EPA[J] - массив погрешностей измерительного комплекса (таблицы 9, 10).

Таблица 9

#### **Наименование погрешности средства измерений или измерительного преобразователя и ее номер (J) в массиве EPA[J]**

Наименование погрешности	J
Систематическая составляющая, %	1
Случайная составляющая, %	2
Класс точности, %	3
Линейность, %	4

Гистерезис, %	5
Повторяемость, %	6
Стабильность, %	7
Удельная температурная погрешность, %/°C	8
Удельная погрешность, обусловленная изменением напряжения, %/В	9
Дополнительная погрешность, обусловленная вибрацией, %	10
Дополнительная погрешность, обусловленная изменением сопротивления, %	11

Таблица 10

**Наименование погрешности средства измерений и ее номер (J) в массиве ЕРА[J]**

Наименование погрешности	J
Систематическая составляющая, %	12
Случайная составляющая, %	13
Класс точности, %	14
Линейность, %	15
Гистерезис, %	16
Повторяемость, %	17
Стабильность, %	18
Удельная температурная погрешность, %/°C	19
Дополнительная погрешность, обусловленная изменением напряжения, %	20
Дополнительная погрешность, обусловленная вибрацией, %	21

ЕРА[22] - систематическая составляющая погрешности планиметра (или интегратора), %.

ЕРА[23] - случайная составляющая погрешности планиметра (или интегратора), %.

ЕРА[24] - основная погрешность барометра, %.

ЕРА[25] - верхний предел измерения атмосферного давления, бар.

ЕРА[26] - верхний предел измерения давления, бар.

4.1.9 Характеристики измерительного комплекса (измерение температуры)

SIZMTD - номер варианта состава измерительного комплекса:

- 1) 0 - термометр;
- 2) 1 - измерительный преобразователь и средство измерений с линейной функцией преобразования;
- 3) 2 - измерительный преобразователь и средство измерений с квадратичной функцией преобразования;
- 4) 3 - измерительный преобразователь, средство измерений с линейной функцией преобразования и



планиметр (или интегратор) с линейной функцией преобразования;

5) 4 - измерительный преобразователь, средство измерений с линейной функцией преобразования и планиметр (или интегратор) с квадратичной функцией преобразования;

6) 5 - измерительный преобразователь, средство измерений с квадратичной функцией преобразования и планиметр (или интегратор) с линейной функцией преобразования;

7) 6 - измерительный преобразователь, вторичный преобразователь, средство измерений с линейной функцией преобразования и планиметр (или интегратор) с линейной функцией преобразования.

ETA[J] - массив погрешностей измерительного комплекса (таблицы 11, 12, 13).

ETA[0] - абсолютная погрешность термометра, °C.

Таблица 11

**Наименование погрешности измерительного преобразователя и ее номер (J) в массиве ETA[J]**

Наименование погрешности	J
Систематическая составляющая, %	1
Случайная составляющая, %	2
Класс точности, %	3
Линейность, %	4
Гистерезис, %	5
Повторяемость, %	6
Стабильность, %	7
Удельная температурная погрешность, %/°C	8
Удельная погрешность, обусловленная изменением напряжения, %/В	9
Дополнительная погрешность, обусловленная вибрацией, %	10
Дополнительная погрешность, обусловленная изменением сопротивления, %	11

Таблица 12

**Наименование погрешности средства измерений или вторичного преобразователя и ее номер (J) в массиве ETA(J)**

Наименование погрешности	J
Систематическая составляющая, %	12
Случайная составляющая, %	13
Класс точности, %	14
Линейность, %	15
Гистерезис, %	16

Повторяемость, %	17
Стабильность, %	18
Удельная температурная погрешность, %/°С	19
Дополнительная погрешность, обусловленная изменением напряжения, %	20
Дополнительная погрешность, обусловленная вибрацией, %	21

Таблица 13

**Наименование погрешности средства измерений и ее номер (J) в массиве ЕТА[J]**

Наименование погрешности	J
Систематическая составляющая, %	22
Случайная составляющая, %	23
Класс точности, %	24
Линейность, %	25
Гистерезис, %	26
Повторяемость, %	27
Стабильность, %	28
Удельная температурная погрешность, %/°С	29
Дополнительная погрешность, обусловленная изменением напряжения, %	30
Дополнительная погрешность, обусловленная вибрацией, %	31

ЕТА[32] - систематическая составляющая погрешности планиметра (или интегратора), %.

ЕТА[33] - случайная составляющая погрешности планиметра (или интегратора), %.

ЕТА[34] - верхний предел измерения температуры, °С.

ЕТА[35] - нижний предел измерения температуры, °С.

4.1.10 Характеристики измерительного комплекса (измерение плотности природного газа при стандартных условиях)

ЕРОСА[J] - массив погрешностей измерительного комплекса (таблица 14).

Таблица 14

**Наименование погрешности измерительного преобразователя и ее номер (J) в массиве ЕРОСА[J]**

Наименование погрешности	J
Систематическая составляющая, %	1
Случайная составляющая, %	2

Класс точности, %	3
Повторяемость, %	4
Удельная температурная погрешность, %/°C	5

EROSA[6] - относительная погрешность средства измерений, %.

EROSA[7] - EROSA[22] - относительные погрешности измерения концентраций компонентов природного газа (см. 3.1.5), %.

4.1.11 ErrTau - погрешность определения интервала времени TauAV, %.

4.1.12 DHD - абсолютная погрешность хода приводного механизма диаграммы, мин.

4.1.13 Параметры эксплуатации (измеряемые за определенный промежуток времени - месяц, год и т.д.)

T1LO, T1HO - нижнее и верхнее значения температуры окружающей среды, определяемые измерительным преобразователем, °C.

T2LO, T2HO - нижнее и верхнее значения температуры окружающей среды, определяемые средством измерений, °C.

UL, UH - нижнее и верхнее значения напряжения питания, В.

TLG, THG - нижнее и верхнее значения температуры среды, °C.

PLG, PHG - нижнее и верхнее значения давления среды, бар.

DPL, DPH - нижнее и верхнее значения перепада давления на сужающем устройстве, бар.

ROSL, ROSH - нижнее и верхнее значения плотности природного газа при стандартных условиях, кг/м<sup>3</sup>.

YIM[J,1] - верхние значения концентраций компонентов природного газа ( $1 \leq J \leq 16$ , см. 3.1.5).

YIM[J,2] - нижние значения концентраций компонентов природного газа ( $1 \leq J \leq 16$ , см. 3.1.5).

RoL, RoH - нижнее и верхнее значения плотности природного газа при рабочих условиях, кг/м<sup>3</sup>.

4.1.14 Характеристика измеряемых параметров:

IfConstRo = 0, если плотность природного газа при стандартных условиях - непрерывно измеряемая величина; IfConstRo = 1, если плотность природного газа при стандартных условиях - условно-постоянная величина;

IfConstY = 0, если концентрации компонентов природного газа - непрерывно измеряемые величины; IfConstY = 1, если концентрации компонентов природного газа - условно-постоянные величины;

IfConstP = 0, если давление среды - непрерывно измеряемая величина; IfConstP = 1, если давление среды - условно-постоянная величина;

IfConstT = 0, если температура среды - непрерывно измеряемая величина; IfConstT = 1, если температура среды - условно-постоянная величина;

IfConstDp = 0, если перепад давления - непрерывно измеряемая величина; IfConstDp = 1, если перепад давления - условно-постоянная величина.

4.1.15 Характеристика измерительного комплекса (измерение плотности природного газа при рабочих условиях)

ERowA[J] - массив погрешностей измерительного комплекса (таблицы 15, 16).

Таблица 15

**Наименование погрешности измерительного преобразователя и ее номер (J) в массиве ERowA[J]**

Наименование погрешности	J
Систематическая составляющая, %	1
Случайная составляющая, %	2
Класс точности, %	3
Линейность, %	4
Гистерезис, %	5
Повторяемость, %	6
Стабильность, %	7
Удельная температурная погрешность, %/°C	8
Удельная погрешность, обусловленная изменением напряжения, %/В	9
Дополнительная погрешность, обусловленная вибрацией, %	10

Таблица 16

**Наименование погрешности корнеизвлекающего средства измерений и ее номер (J) в массиве ERowA[J]**

Наименование погрешности	J
Систематическая составляющая, %	11
Случайная составляющая, %	12
Класс точности, %	13
Линейность, %	14
Гистерезис, %	15
Повторяемость, %	16
Стабильность, %	17
Удельная температурная погрешность, %/°C	18
Дополнительная погрешность, обусловленная изменением напряжения, %	19
Дополнительная погрешность, обусловленная вибрацией, %	20

ERowA[21] - верхний предел измерения плотности газа при рабочих условиях, кг/м<sup>3</sup>.

ERowA[22] - нижний предел измерения плотности газа при рабочих условиях, кг/м<sup>3</sup>.

## 4.2 Выходные данные

RO, KAPPA, MU, QC, VC, VM, HS[1], HS[2] - см. 3.2.

HSV[1] - высшая теплота сгорания (энергосодержание) среды, МДж.

HSV[2] - низшая теплота сгорания (энергосодержание) среды, МДж.

EQR - случайная составляющая относительной погрешности расчета расхода среды для измерительного трубопровода с порядковым номером NNit.

EQS - систематическая составляющая относительной погрешности расчета расхода среды для измерительного трубопровода с порядковым номером NNit.

EQ1 - полная относительная погрешность расчета расхода среды для измерительного трубопровода с порядковым номером NNit.

EVC - полная относительная погрешность расчета количества среды для измерительного трубопровода с порядковым номером NNit за время TauAV.

EH[1] - полная относительная погрешность расчета высшей теплоты сгорания среды для измерительного трубопровода с порядковым номером NNit.

EH[2] - полная относительная погрешность расчета низшей теплоты сгорания среды для измерительного трубопровода с порядковым номером NNit.

## 4.3 Листинг модуля расчета погрешности определения расхода и количества жидкостей и газов

### 4.3.1 Типы используемых переменных:

FL: text; NNIT, IERR, SIZMDPD, SIZMPD, SIZMTD: byte; T1HO, T1LO, UH, UL, PHG, PLG, THG, TLG, DPH, DPL, T2HO, T2LO, ROSH, ROSL, EDPS, EDPR, EDP, EPS, EPR, EP, ETS, ETR, ET, EROSS, EROSR, EROS, EROWS, EROWR, EROW, EQS, EQR, EQ1, EVC, DD20, DT20, DD, DT, ALFAT, ALFASU, RSH, RN, TAUP, TAUAV, ERRTAU, DHD, RO, KAPPA, MU, VM, RO SC, QC, VCV, VC, RoH, RoL, RoM: real; RocStr: string[6]; LRA: array [1..10] of real; DAC, TAC: array [1..3] of real; EPA: err1; YIM: err2; HS, HSV, EH: err3; NMETHKA, NSUBA, NSUZA, NVARYA, VARPA, VarRoA, AMOUNTR, SODSUA: err7; NRA: err8; EDPA: err9; ETA: err10; EROSA, ERowA: err11;

type err1 = array [1..26] of real; err2 = array [1..16, 1..2] of real; err3 = array [1..2] of real; err7 = array [1..30] of byte; err8 = array [1..10] of byte; err9 = array [1..35] of real; err10 = array [0..35] of real; err11 = array [1..22] of real;

### 4.3.2 Листинг модуля ERRQSNX

```
Unit ERRQSNX;
Interface
{-----}
Procedure QS;

Implementation

uses DOS, CRT;
{-----}
var
  Zc: real;

const
```

```

Bi:array[1..16] of real = (0.0436,0.0894,0.1288,0.1783,0.1703,0.2345,
                          0.2168,0.2846,0.3521,0.4278,0.0173,0.0728,
                          0.1,0.0,0.02,-0.0051);
Procedure DP(E:err9; DPC:real; var EDPSY,EDPRA,EDP:real);
forward;
Procedure P(E:err1; PM:real; var EPSY,EPRA,EP:real);
forward;
Procedure T(E:err10; TA:real; var ETSY,ETRA,ET:real);
forward;
Procedure ROS(E:err11; RIM:err1; var EROSS,EROSR,EROS:real);
forward;
Procedure C(BET,RE,DP,KSH,KK:real; var ERC:real; var IERR:byte);
forward;
Procedure EPSI(BET,DP,P:real; var EEPS:real);
forward;
Procedure EPH(YR:err1; T,P,ROS:real; var ERO,EKAP,EVIS:real);
forward;
Procedure EPHP(N:byte; T,P:real; var ERO,EKAP,EVIS:real);
forward;
Procedure ROW(E:err11; var EROWS,EROWR,EROW:real);
forward;
Procedure QCalc(NRQ:byte; T,P,DP,YA,YY,DD0,DT0:real; YR:err1;
var BET,RE,KSH,KK,ROC,QC,VCV:real; var VCC:real; var VMC,ROX,
KAPPAX,MUX:real; var HS:err3);
forward;
Procedure HSP(RIM:err1; YA,YY:real; var ERRH1,ERRH2:real);
forward;
function r_(A,R:real):real; forward;
{-----}
{ Основная процедура: выполняет диспетчерские функции, а также расчет
коэффициентов влияния измеряемых параметров и погрешностей определения расхода
и количества среды }
  Procedure QS;

const
  EDD = 0.07; EDT = 0.40;

label
  1,3;

var
  I,J,NRQ: byte;
  RIM,RIMX,DVY,TETYI:err1; DVR,YA,YY,DVDP,DVPG,DVTG,T1,T2:real;
  DVAY,HSX :err3;
  BET,RE,KSH,KK,QC1,VCV1,VM1,QC2,VCV2,VM2:real;
  VC1,VC2:real;
  ERC,EEPS,ERO,EKAP,EVIS,ERRH1,ERRH2:real;
  DPMD,DPM1,DPM2,TETDP,DDD,DD01,DD02,TETDD,DDT,DT01,DT02,TETDT,
  DKAP,KAP1,KAP2,TETKAP,DMU,MU1,MU2,TETMU,TETC,TETEPS,TETRO,
  TETYA,TETYY,
  DROS,ROS1,ROS2,TETROS,DYA,DYY,YA1,YY1,YA2,YY2,R01,R02,DTMG,
  TETT,DPMG,
  TETP,EYR:real;
  DRIM:real;

begin { QS }
  ROSC:= 0.0; DVR:= 0.0; for I:= 1 to 2 do DVAY[I]:= 0.0;
  for I:=1 to 16 do DVY[I]:=0.0;
  if NSUBA[NNIT]=0 then begin

```

```

if NVARYA[NNIT]=1 then begin
ROSC:= 2.0*ROSH*ROSL/(ROSH+ROSL);
if IfConstRo <> 0 then DVR:= 100.0*(ROSH-ROSL)/(ROSH+ROSL);
YA:= 0.5*(YIM[11,1]+YIM[11,2]);
YY:= 0.5*(YIM[12,1]+YIM[12,2]);
for J:=11 to 12 do begin
I:= J-10;
if ((YIM[J,1] <> 0) and (IfConstY <> 0)) or
((YIM[J,2] <> 0) and (IfConstY <> 0)) then
DVAY[I]:= 100.0*(YIM[J,1]-YIM[J,2])/(YIM[J,1]+YIM[J,2]) end;
end else
for I:= 1 to 16 do begin RIM[I]:= 0.5*(YIM[I,1]+YIM[I,2]);
if ((YIM[I,1] <> 0) and (IfConstY <> 0)) or
((YIM[I,2] <> 0) and (IfConstY <> 0)) then
DVY[I]:= 100.0*(YIM[I,1]-YIM[I,2])/(YIM[I,1]+YIM[I,2]);
RIMX[I]:=RIM[I]
end;
end; {endif}
PMG:= sqr(0.5*(sqr(PHG)+sqr(PLG)));
DVPG:= 0.0; if IfConstP <> 0 then DVPG:= 100.0*(PHG-PLG)/(PHG+PLG);
T1:=THG+273.15; T2:=TLG+273.15;
TMG:= 4.0*T1*T2/sqr(sqr(T1)+sqr(T2))-273.15;
DVTG:= 0.0; if IfConstT <> 0 then DVTG:= 100.0*(T1-T2)/(T1+T2);
DPM:= sqr(0.5*(sqr(DPH)+sqr(DPL)));
DVDP:= 0.0; if IfConstDp <> 0 then DVDP:= 100.0*(DPH-DPL)/(DPH+DPL);
if(NSubA[NNit]=0) and (VarRoA[NNit]=1) then
RoM:= sqr(0.5*(sqr(RoH)+sqr(RoL)));
NRQ:= 1;
QCalc(NRQ, TMG, PMG, DPM, YA, YY, DD20, DT20, RIM, BET, RE, KSH, KK, ROSC,
QC, VCV, VC, VM, RO, KAPPA, MU, HS);
C(BET, RE, DPH, KSH, KK, ERC, IERR);
if IERR=0 then begin
DP(EDPA, DPM, EDPS, EDPR, EDP);
P(EPA, PMG, EPS, EPR, EP);
T(ETA, TMG, ETS, ETR, ET);
ROS(EROSA, RIM, EROSS, EROSR, EROS);
EPSI(BET, DPM, PMG, EEPS);
EPH(RIM, TMG, PMG, ROSC, ERO, EKAP, EVIS);
if (NSubA[NNit]=0) and (VarRoA[NNit]=1) then begin
ROW(EROWA, EROWS, EROWR, EROW); ERO:= EROWS
end
else EROWR:= 0.0;
NRQ:= 0;
TETDP:= 0.0;
if EDP <> 0.0 then begin
DPMD:= 0.001*EDP*DPM; DPM1:= DPM+DPMD; DPM2:=DPM-DPMD;
QCalc(NRQ, TMG, PMG, DPM1, YA, YY, DD20, DT20, RIM, BET, RE, KSH, KK, ROSC,
QC1, VCV1, VC1, VM1, RO, KAPPA, MU, HS);
QCalc(NRQ, TMG, PMG, DPM2, YA, YY, DD20, DT20, RIM, BET, RE, KSH, KK, ROSC,
QC2, VCV2, VC2, VM2, RO, KAPPA, MU, HS);
TETDP:= DPM*(QC1-QC2)/(2.0*DPMD)/QC;
end;
DDD:= 0.001*EDD*DD20; DD01:= DD20+DDD; DD02:= DD20-DDD;
QCalc(NRQ, TMG, PMG, DPM, YA, YY, DD01, DT20, RIM, BET, RE, KSH, KK, ROSC,
QC1, VCV1, VC1, VM1, RO, KAPPA, MU, HS);
QCalc(NRQ, TMG, PMG, DPM, YA, YY, DD02, DT20, RIM, BET, RE, KSH, KK, ROSC,
QC2, VCV2, VC2, VM2, RO, KAPPA, MU, HS);
TETDD:= DD20*(QC1-QC2)/(2.0*DDD)/QC;
DDT:= 0.001*EDT*DT20; DT01:= DT20+DDT; DT02:= DT20-DDT;

```

```

QCalc(NRQ, TMG, PMG, DPM, YA, YY, DD20, DT01, RIM, BET, RE, KSH, KK, ROSC,
      QC1, VCV1, VC1, VM1, RO, KAPPA, MU, HS);
QCalc(NRQ, TMG, PMG, DPM, YA, YY, DD20, DT02, RIM, BET, RE, KSH, KK, ROSC,
      QC2, VCV2, VC2, VM2, RO, KAPPA, MU, HS);
TETDT:= DT20*(QC1-QC2)/(2.0*DDT)/QC;
DKAP:= 0.001*EKAP*KAPPA; KAP1:= KAPPA+DKAP; KAP2:= KAPPA-DKAP;
QCalc(NRQ, TMG, PMG, DPM, YA, YY, DD20, DT20, RIM, BET, RE, KSH, KK, ROSC,
      QC1, VCV1, VC1, VM1, RO, KAP1, MU, HS);
QCalc(NRQ, TMG, PMG, DPM, YA, YY, DD20, DT20, RIM, BET, RE, KSH, KK, ROSC,
      QC2, VCV2, VC2, VM2, RO, KAP2, MU, HS);
TETKAP:= KAPPA*(QC1-QC2)/(2.0*DKAP)/QC;
DMU:= 0.001*EVIS*MU; MU1:= MU+DMU; MU2:= MU-DMU;
QCalc(NRQ, TMG, PMG, DPM, YA, YY, DD20, DT20, RIM, BET, RE, KSH, KK, ROSC,
      QC1, VCV1, VC1, VM1, RO, KAPPA, MU1, HS);
QCalc(NRQ, TMG, PMG, DPM, YA, YY, DD20, DT20, RIM, BET, RE, KSH, KK, ROSC,
      QC2, VCV2, VC2, VM2, RO, KAPPA, MU2, HS);
TETMU := MU*(QC1-QC2)/(2.0*DMU)/QC;
TETC:= 1.0; TETEPS:= 1.0; TETRO:= 0.5;
if(NSubA[NNit]=0) and (VarRoA[NNit]=1) then begin
  TETRO:= 0.0;
  if EROW <> 0.0 then begin
    DROS:= 0.001*EROW*RoM; RO1:= RoM+DROS; RO2:=RoM-DROS;
    QCalc(NRQ, TMG, PMG, DPM, YA, YY, DD20, DT20, RIM, BET, RE, KSH, KK, ROSC,
          QC1, VCV1, VC1, VM1, RO1, KAPPA, MU, HS);
    QCalc(NRQ, TMG, PMG, DPM, YA, YY, DD20, DT20, RIM, BET, RE, KSH, KK, ROSC,
          QC2, VCV2, VC2, VM2, RO2, KAPPA, MU, HS);
    TETRO:= RoM*(QC1-QC2)/(2.0*DROS)/QC
  end;
end;
NRQ:= 1;
TETYA:= 0.0; TETYY:= 0.0; for I:= 1 to 16 do TETYS[I]:= 0.0;
if (NSUBA[NNIT] <> 0) or (NVARYA[NNIT]=0) then TETROS:= 1.0
else begin
  TETROS:= 0.0;
  if EROS <> 0.0 then begin
    DROS:= 0.001*EROS*ROSC; ROS1:= ROSC+DROS; ROS2:= ROSC-DROS;
    QCalc(NRQ, TMG, PMG, DPM, YA, YY, DD20, DT20, RIM, BET, RE, KSH, KK, ROS1,
          QC1, VCV1, VC1, VM1, RO1, KAP1, MU1, HSX);
    QCalc(NRQ, TMG, PMG, DPM, YA, YY, DD20, DT20, RIM, BET, RE, KSH, KK, ROS2,
          QC2, VCV2, VC2, VM2, RO2, KAP2, MU2, HSX);
    TETROS:= ROSC*(QC1-QC2)/(2.0*DROS)/QC;
  end;
  if (YA <> 0.0) and (EROSA[17] <> 0.0) then begin
    DYA:= 0.001*EROSA[17]*YA; YA1:= YA+DYA; YA2:= YA-DYA;
    QCalc(NRQ, TMG, PMG, DPM, YA1, YY, DD20, DT20, RIM, BET, RE, KSH, KK, ROSC,
          QC1, VCV1, VC1, VM1, RO1, KAP1, MU1, HSX);
    QCalc(NRQ, TMG, PMG, DPM, YA2, YY, DD20, DT20, RIM, BET, RE, KSH, KK, ROSC,
          QC2, VCV2, VC2, VM2, RO2, KAP2, MU2, HSX);
    TETYA:= YA*(QC1-QC2)/(2.0*DYA)/QC
  end;
  if (YY <> 0.0) and (EROSA[18] <> 0.0) then begin
    DYY:= 0.001*EROSA[18]*YY; YY1:= YY+DYY; YY2:= YY-DYY;
    QCalc(NRQ, TMG, PMG, DPM, YA, YY1, DD20, DT20, RIM, BET, RE, KSH, KK, ROSC,
          QC1, VCV1, VC1, VM1, RO1, KAP1, MU1, HSX);
    QCalc(NRQ, TMG, PMG, DPM, YA, YY2, DD20, DT20, RIM, BET, RE, KSH, KK, ROSC,
          QC2, VCV2, VC2, VM2, RO2, KAP2, MU2, HSX);
    TETYY:= YY*(QC1-QC2)/(2.0*DYY)/QC
  end;
end; {endif}

```



```

TETT:= 0.0;
if ET <> 0.0 then begin
  DTMG:= 0.001*ET*(TMG+273.15);
QCalc(NRQ, TMG+DTMG, PMG, DPM, YA, YY, DD20, DT20, RIM, BET, RE, KSH, KK,
  ROSC, QC1, VCV1, VC1, VM1, RO1, KAP1, MU1, HSX);
QCalc(NRQ, TMG-DTMG, PMG, DPM, YA, YY, DD20, DT20, RIM, BET, RE, KSH, KK,
  ROSC, QC2, VCV2, VC2, VM2, RO2, KAP2, MU2, HSX);
  TETT:= (TMG+273.15)*(QC1-QC2)/(2.0*DTMG)/QC
end;
TETP:= 0.0;
if EP <> 0.0 then begin
  DPMG:= 0.001*EP*PMG;
QCalc(NRQ, TMG, PMG+DPMG, DPM, YA, YY, DD20, DT20, RIM, BET, RE, KSH, KK,
  ROSC, QC1, VCV1, VC1, VM1, RO1, KAP1, MU1, HSX);
QCalc(NRQ, TMG, PMG-DPMG, DPM, YA, YY, DD20, DT20, RIM, BET, RE, KSH, KK,
  ROSC, QC2, VCV2, VC2, VM2, RO2, KAP2, MU2, HSX);
  TETP:= PMG*(QC1-QC2)/(2.0*DPMG)/QC
end;
EYR:= 0.0;
if(NSUBA[NNIT]=0) and (NVARYA[NNIT]=0) then
  for I:= 1 to 16 do
    if (RIM[I] <> 0.0) and (EROSA[I+6] <> 0.0) then begin
      DRIM:= 0.001*EROSA[I+6]*RIM[I];
      RIMX[I]:= RIM[I]+DRIM;
      for J:=1 to 16 do
        if (J <> I) and (RIM[J] <> 0.0) then
          RIMX[J]:= RIM[J]*(1.0-DRIM/(1.0-RIM[I]));
QCalc(NRQ, TMG, PMG, DPM, YA, YY, DD20, DT20, RIMX, BET, RE, KSH, KK, ROS1,
  QC1, VCV1, VC1, VM1, RO1, KAP1, MU1, HSX);
      RIMX[I]:= RIM[I]-DRIM;
      for J:= 1 to 16 do
        if (J <> I) and (RIM[J] <> 0.0) then
          RIMX[J]:= RIM[J]*(1.0+DRIM/(1.0-RIM[I]));
QCalc(NRQ, TMG, PMG, DPM, YA, YY, DD20, DT20, RIMX, BET, RE, KSH, KK, ROS2,
  QC2, VCV2, VC2, VM2, RO2, KAP2, MU2, HSX);
      TETYI[I]:= 500.0*(QC1-QC2)/QC/EROSA[I+6];
      EYR:=EYR+sqr(TETYI[I])*(sqr(EROSA[6+I])+sqr(DVY[I]))
    end; {endif}
EQR:= sqrt(sqr(TETDP)*(sqr(EDPR)+sqr(DVDP))+sqr(TETP)*
  (sqr(EPR)+sqr(DVPG))+sqr(TETT)*(sqr(ETR)+sqr(DVTG))+
  sqr(TETROS)*(sqr(EROSR)+sqr(DVR))+sqr(TETYA)*
  (sqr(EROSA[17])+sqr(DVAY[1]))+sqr(TETYI)*
  (sqr(EROSA[18])+sqr(DVAY[2]))+EYR+sqr(TETRO*EROWR));
EQS:= sqrt(sqr(TETDP*EDPS)+sqr(TETP*EPS)+
  sqr(TETT*ETS)+sqr(TETROS*EROSS)+sqr(TETC*ERC)+
  sqr(TETEPS*EEPS)+sqr(TETDD*EDD)+sqr(TETDT*EDT)+
  sqr(TETRO*ERO)+sqr(TETKAP*EKAP)+sqr(TETMU*EVIS));

EQ1:= sqrt(sqr(EQS)+sqr(EQR));
if ERRTAU=0.0 then ERRTAU:= 100.0*DHD/(60*TAUAV);
EVC:= sqrt(sqr(EQS)+sqr(EQR)+sqr(ERRTAU));
HSP(RIM, YA, YY, ERRH1, ERRH2);
if ERRH1=0.0 then begin
  EH[1]:= 0.0; EH[2]:= 0.0
end else begin
  EH[1]:= sqrt(sqr(ERRH1)+sqr(EVC));
  EH[2]:= sqrt(sqr(ERRH2)+sqr(EVC))
end; {endif}
HSV[1]:= Hs[1]*VCV; HSV[2]:= Hs[2]*VCV

```

```

        end else begin
case IERR of
    1: begin
        writeln(Fl,
'расстояние между первым перед сужающим устройством гидравлическим
сопротивлением');
        writeln(Fl,
'и сужающим устройством меньше допустимой величины
(7.2.4, 7.3.1 ГОСТ 8.563.1).');
        end;
    2: begin
        writeln(Fl,
'расстояние между сужающим устройством и гидравлическим сопротивлением за
сужающим устройством ');
        writeln(Fl,
'меньше допустимой величины (7.2.4 ГОСТ 8.563.1).');
        end;
    3: begin
        writeln(Fl,
'дополнительная погрешность коэффициента истечения, обусловленная
сокращением');
        writeln(Fl,
'длин прямых участков между сужающим устройством и гидравлическими сопро-
тивлениями, ');
        writeln(Fl,
'превышает 1 % (7.2.4 ГОСТ 8.563.1).');
        end;
    4: begin
        writeln(Fl,
'эксцентриситет между осями измерительного трубопровода и сужающего
устройства превышает допустимую');
        writeln(Fl,
'величину (7.5.2.3 ГОСТ 8.563.1).');
        end;
    5: begin
        writeln(Fl,
'высота уступа между двумя секциями измерительного трубопровода превышает
допусти-');
        writeln(Fl,
'мую величину (7.5.1.4 ГОСТ 8.563.1).');
        end;
    end; { endcase }
end; {endif}
end; { QS }
{-----}
{ Расчет погрешностей измерения перепада давления на сужающем устройстве }

```

Procedure DP;

```

var
    E1DPSC, E1DPRC, EDP1SC, EDP1RC, E2DPS, E3DPS, E6DPS, EDDPS,
    EDP2S, EVDPS, EDP5S, EDPRSC, EDPRRC, EDPRS, ERDPS: real;
    YDP: array [1..2] of real;

begin { DP }
if (E[1]<>0) or (E[2]<>0) then
begin E1DPSC:=E[1];
      E1DPRC:=E[2]
end
end

```

```

else begin
  if E[3]<>0 then
    begin
      E1DPSC:= E[3];
      E1DPRC:= 0.0
    end
  else begin
      E1DPSC:= sqrt(sqr(E[4])+sqr(E[5]));
      E1DPRC:= sqrt(sqr(E[6])+sqr(E[7]))
    end;
end; {endif}
E2DPS:= E[8]*(T1HO-T1LO);
E3DPS:= E[9]*(UH-UL);
E6DPS:= E[12]*(PHG-PLG);
EDDPS:=sqrt(sqr(E1DPSC)+sqr(E2DPS)+sqr(E3DPS)+sqr(E[10])+
            sqr(E[11])+sqr(E6DPS));

if SIZMDPD > 1 then begin
if(E[13]<>0) or (E[14]<>0) then
  begin
    EDP1SC:= E[13];
    EDP1RC:= E[14];
  end
else begin
  if E[15]<>0 then
    begin
      EDP1SC:= E[15];
      EDP1RC:= 0.0
    end
  else begin
      EDP1SC := sqrt(sqr(E[16]) + sqr(E[17]));
      EDP1RC := sqrt(sqr(E[18]) + sqr(E[19]));
    end;
end; {endif}
EDP2S:=E[20]*(T2HO-T2LO);
EVDPS:=sqrt(sqr(EDP1SC)+sqr(EDP2S)+sqr(E[21])+sqr(E[22]));
if SIZMDPD=7 then begin
if (E[23]<>0) or (E[24]<>0) then
  begin
    EDPRSC:=E[23];
    EDPRRC:=E[24]
  end
else begin
  if E[25]<>0 then
    begin
      EDPRSC:= E[15];
      EDPRRC:= 0.0
    end
  else begin
      EDPRSC:= sqrt(sqr(E[26]) + sqr(E[27]));
      EDPRRC:= sqrt(sqr(E[28]) + sqr(E[29]))
    end;
end; {endif}
EDPRS:= E[30]*(T2HO-T2LO);
ERDPS:= sqrt(sqr(EDPRSC)+sqr(EDPRS)+sqr(E[31])+sqr(E[32]));
end;
end; {endif}
YDP[1]:= E[35]/DPC; YDP[2]:=2.0*sqrt(E[35]/DPC);
  case SIZMDPD of

```

```

0: begin
  EDPSY:= YDP[1]*EDDPS;
  EDPRA:= YDP[1]*E1DPRC
end;
1: begin
  EDPSY:= YDP[2]*EDDPS;
  EDPRA:= YDP[2]*E1DPRC
end;
2: begin
  EDPSY:=sqrt(sqr(YDP[1]*EDDPS)+sqr(YDP[1]*EVDPS));
  EDPRA:=sqrt(sqr(YDP[1]*E1DPRC)+sqr(YDP[1]*EDP1RC))
end;
3: begin
  EDPSY:=sqrt(sqr(YDP[1]*EDDPS)+sqr(YDP[2]*EVDPS));
  EDPRA:=sqrt(sqr(YDP[1]*E1DPRC)+sqr(YDP[2]*EDP1RC))
end;
4: begin
  EDPSY:=sqrt(sqr(YDP[1]*EDDPS)+sqr(YDP[1]*EVDPS)+sqr(YDP[1]*E[33]));
  EDPRA:=sqrt(sqr(YDP[1]*E1DPRC)+sqr(YDP[1]*EDP1RC)+sqr(YDP[1]*E[34]))
end;
5: begin
  EDPSY:=sqrt(sqr(YDP[1]*EDDPS)+sqr(YDP[1]*EVDPS)+sqr(YDP[2]*E[33]));
  EDPRA:=sqrt(sqr(YDP[1]*E1DPRC)+sqr(YDP[1]*EDP1RC)+
    sqr(YDP[2]*E[34]))
end;
6: begin
  EDPSY:=sqrt(sqr(YDP[1]*EDDPS)+sqr(YDP[2]*EVDPS)+sqr(YDP[2]*
    E[33]));
  EDPRA:=sqrt(sqr(YDP[1]*E1DPRC)+sqr(YDP[2]*EDP1RC)+
    sqr(YDP[2]*E[34]))
end;
7: begin
  EDPSY:=sqrt(sqr(YDP[1]*EDDPS)+sqr(YDP[2]*ERDPS)+
    sqr(YDP[2]*EVDPS)+sqr(YDP[2]*E[33]));
  EDPRA:=sqrt(sqr(YDP[1]*E1DPRC)+sqr(YDP[2]*EDPRRC)+
    sqr(YDP[2]*EDP1RC)+sqr(YDP[2]*E[34]))
end;
end; { endcase }
EDP:=sqrt(sqr(EDPSY)+sqr(EDPRA));
end; { DP }
{-----}
{ Расчет погрешностей определения давления среды }
Procedure P;

```

```

var
  E1PSC,E1PRC,E1SC,E1RC,E2PS,E3PS,EDPS,EP2S,EVPS,EBP: real;
  YP: array [1..2] of real;

```

```

begin { P }
if (E[1]<>0) or(E[2]<>0) then
begin
  E1PSC:= E[1];
  E1PRC:= E[2]
end
else begin
if E[3]<>0 then
begin
  E1PSC:= E[3];
  E1PRC:= 0

```

```

end
else begin
  E1PSC:=sqrt(sqr(E[4])+sqr(E[5]));
  E1PRC:=sqrt(sqr(E[6])+sqr(E[7]))
end;
end; {endif}
E2PS:=E[8]*(T1HO-T1LO);
E3PS:=E[9]*(UH-UL);
EDPS:=sqrt(sqr(E1PSC)+sqr(E2PS)+sqr(E3PS)+sqr(E[10])+
sqr(E[11]));

if SIZMPD>1 then begin
if (E[12]<>0) or(E[13]<>0) then
begin
  E1PSC:= E[12];
  E1RC:= E[13]
end
else begin
if E[14]<>0 then
begin
  E1PSC:= E[14];
  E1RC:= 0
end
else begin
  E1PSC:= sqrt(sqr(E[15])+sqr(E[16]));
  E1RC:= sqrt(sqr(E[17])+sqr(E[18]))
end;
end; {endif}
EP2S:=E[19]*(T2HO-T2LO);
EVPS:=sqrt(sqr(E1PSC)+sqr(EP2S)+sqr(E[20])+sqr(E[21]));
end;
  YP[1]:= E[26]/PM; YP[2]:= 2.0*sqrt(E[26]/PM); EBP:= 0.0;
if VARPA[NNIT]=0 then EBP:= sqrt(E[24]*E[25]/PM);
  case SIZMPD of
0: begin
  EPSY:=sqrt(sqr(YP[1]*EDPS)+EBP);
  EPRA:=YP[1]*E1PRC
end;
1: begin
  EPSY:=sqrt(sqr(YP[2]*EDPS)+EBP);
  EPRA:=YP[2]*E1PRC
end;
2: begin
  EPSY:=sqrt(sqr(YP[1]*EDPS)+sqr(YP[1]*EVPS)+EBP);
  EPRA:=sqrt(sqr(YP[1]*E1PRC)+sqr(YP[1]*EP1RC))
end;
3: begin
  EPSY:=sqrt(sqr(YP[1]*EDPS)+sqr(YP[2]*EVPS)+EBP);
  EPRA:=sqrt(sqr(YP[1]*E1PRC)+sqr(YP[2]*EP1RC))
end;
4: begin
  EPSY := sqrt(sqr(YP[1]*EDPS)+sqr(YP[1]*EVPS)+sqr(YP[1]*E[23])+
EBP);
  EPRA := sqrt(sqr(YP[1]*E1PRC)+sqr(YP[1]*EP1RC)+sqr(YP[1]*
E[24]))
end;
5: begin
  EPSY:=sqrt(sqr(YP[1]*EDPS)+sqr(YP[1]*EVPS)+sqr(YP[2]*E[23])+
EBP);

```

```

        EPRA:=sqrt(sqr(Y[1]*E1PRC)+sqr(YP[1]*EP1RC)+sqr(YP[2]*E(24)))
    end;
    6: begin
        EPSY:=sqrt(sqr(YP[1]*EDPS)+sqr(YP[2]*EVPS)+sqr(YP[2]*E[23])+
            EBP);
        EPRA:=sqrt(sqr(YP[1]*E1PRC)+sqr(YP[2]*EP1RC+sqr(YP[2]*
            E[24])))
    end;

    end; { endcase }
    EP := sqrt(sqr(EPSY)+sqr(EPRA));
end; { P }
{-----}
{ Расчет погрешностей определения температуры среды }
Procedure T;

```

```

var
    E1TSC,E1TRC,ET1SC,ET1RC,ET2S,EVTS,DET,E2TS,E3TS,EDTS,
    ET2SC,ET2RC,ETXS,EXTS: real;
    YT: array [1..2] of real;

```

```

begin { T }
    if SIZMTD=0 then begin
        E1TSC:= 100.0*E[0]/(TA+273.15);
        E1TRC:= 0.0
    end else begin
        if (E[1]<>0) or (E[2]<>0) then
            begin
                E1TSC:= E[1];
                E1TRC:= E[2]
            end
        else begin
            if E[3]<>0 then
                begin
                    E1TSC:= E[3];
                    E1TRC:= 0.0
                end
            else begin
                E1TSC:= sqrt(sqr(E[4])+sqr(E[5]));
                E1TRC:= sqrt(sqr(E[6])+sqr(E[7]))
            end;
        end; {endif}
        E2TS:= E[8]*(T1HO-T1LO);
        E3TS:= E[9]*(UH-UL);
        EDTS:= sqrt(sqr(E1TSC)+sqr(E2TS)+sqr(E3TS)+sqr(E[10])+
            sqr(E[11]));
        if (E[12]<>0) or (E[13]<>0) then
            begin
                ET1SC:=E[12];
                ET1RC:=E[13]
            end
        else begin
            if E[14]<>0 then
                begin
                    ET1SC:= E[14];
                    ET1RC:= 0
                end
            else begin
                ET1SC:=sqrt(sqr(E[15])+sqr(E[16]));
            end
        end
    end
end;

```

```

        ET1RC:=sqrt(sqr(E[17])+sqr(E[18]))
    end;
end; {endif}
ET2S:= E[19]*(T2HO-T2LO);
EVTS:=sqrt(sqr(ET1SC)+sqr(ET2S)+sqr(E[20])+sqr(E[21]));

if SIZMTD=6 then begin
if(E[22]<>0) or (E[23]<>0) then
    begin
        ET2SC:=E[22];
        ET2RC:=E[23]
    end
else begin
    if E[24]<>0 then
        begin
            ET2SC:= E[24];
            ET2RC:= 0.0
        end
    else begin
        ET2SC:= sqrt(sqr(E[25])+sqr(E[26]));
        ET2RC:= sqrt(sqr(E[27])+sqr(E[28]))
    end;
end; {endif}
ETXS:=E[29]*(T2HO-T2LO);
EXTS:=sqrt(sqr(ET2SC)+sqr(ETXS)+sqr(E[30])+sqr(E[31]));
end; {endif}

end; {endif}
DET:=(E[34]-E[35])/(TA+273.15);
YT[1]:= DET;  YT[2]:= 2.0*sqrt(DET);
    case SIZMTD of
0: begin
    ETSY:= YT[1]*E1TSC;
    ETRA:= YT[1]*E1TRC
end;
1: begin
    ETSY:=sqrt(sqr(YT[1]*EDTS)+sqr(YT[1]*EVTS));
    ETRA:=sqrt(sqr(YT[1]*E1TRC)+sqr(YT[1]*ET1RC))
end;
2: begin
    ETSY:=sqrt(sqr(YT[1]*EDTS)+sqr(YT[2]*EVTS));
    ETRA:=sqrt(sqr(YT[1]*E1TRC)+sqr(YT[2]*ET1RC))
end;
3: begin
    ETSY:= sqrt(sqr(YT[1]*EDTS)+sqr(YT[1]*EVTS)+sqr(YT[1]*
        E[32]));
    ETRA:= sqrt(sqr(YT[1]*E1TRC)+sqr(YT[1]*ET1RC)+sqr(YT[1]*
        E[33]))
end;
4: begin
    ETSY:= sqrt(sqr(YT[1]*EDTS)+sqr(YT[1]*EVTS)+sqr(YT[2]*
        E[32]));
    ETRA:= sqrt(sqr(YT[1]*E1TRC)+sqr(YT[1]*ET1RC)+sqr(YT[2]*
        E[33]))
end;
5: begin
    ETSY:= sqrt(sqr(YT[1]*EDTS)+sqr(YT[2]*EVTS)+sqr(YT[2]*
        E[32]));
    ETRA:= sqrt(sqr(YT[1]*E1TRC)+sqr(YT[2]*ET1RC)+sqr(YT[2]*

```

```

        E[33]))
end;
6: begin
    ETSY:= sqrt(sqr(YT[1]*EDTS)+sqr(YT[1]*EXTS)+sqr(YT[1]*EVTS)+
        sqr(YT[1]*E[32]));
    ETRA:= sqrt(sqr(YT[1]*E1TRC)+sqr(YT[1]*ET2RC)+sqr(YT[1]*
        ET1RC)+sqr(YT[1]*E[33]))
end;
end { endcase }
ET:= sqrt(sqr(ETSY)+sqr(ETRA));
end; { T }
{-----}
{ Расчет погрешностей определения плотности среды при стандартных условиях }
Procedure ROS;
const
    EROSI:array[1..31] of real=(0.05,0.05,0.2,0.3,0.3,0.0,0.0,0.0,0.0,0.0,
    0.1,0.1,0.2,0.0,0.0,0.0,0.1,0.1,0.05,0.1,
    0.3,0.0,0.1,0.05,0.05,0.05,0.05,0.3,0.05,0.3,0.05);
var
    E1RSC,E1RRC,E2RSC,EDRSC: real;
    I: integer;
begin { ROS }
    if NSUBA[NNIT]<>0 then begin
        EROSS:= EROSI[NSUBA[NNIT]]; EROS:= EROSS; EROSR:= 0.0
    end else begin
        if NVARYA[NNIT]=1 then begin
            if(E[1]<>0) or (E[2]<>0) then
                begin
                    E1RSC:= E[1];
                    E1RRC:= E[2];
                    E2RSC:= E[5]*(T1HO-T1LO)
                end
            else begin
                if(E[3]<>0) or(E[4]<>0) then
                    begin
                        E1RSC:= E[3];
                        E1RRC:= E[4];
                        E2RSC:= E[5]*(T1HO-T1LO)
                    end
                else begin
                    E1RSC:= 0.0;
                    E1RRC:= 0.0;
                    E2RSC:= 0.0
                end
            end
        end
        else begin
            E1RRC:= 0.0;
            E2RSC:= 0.0; E[6]:= 0.0;
            case NMETHKA[NNIT] of
                0: E1RSC:= 0.1;
                1: E1RSC:= 0.1;
                2: E1RSC:= 0.1;
                3: if RIM[13]<>0 then E1RSC:= 0.15
                    else E1RSC:= 0.1;
            end; {endcase}
        end; {endif}
        EDRSC:=sqrt(sqr(E1RSC)+sqr(E2RSC));
        EROSS:=sqrt(sqr(EDRSC)+sqr(E[6])); EROSR:=E1RRC;

```



```

    EROS:=sqrt(sqr(EROSS)+sqr(EROSR))
end; {endif}
end; { ROS }
{-----}
{ Расчет погрешности определения коэффициента истечения }
Procedure C;

var
    EC0,EL,EE,EH,EX,ESH,EK,EDM,A,B,EXMIN,EXMAX,HD,HDC: real;
    L1,L2,L21,DL: real;
    Lvent: array[1..10] of real;
    NR,I,MR: byte;

const
    AK: array [1..16] of real = (14.5,17.5,21.0,5.0,16.0,10.0,
    22.0,12.5,47.5,11.5,13.5,33.5,5.0,12.0,54.5,0.0);
    BK: array [1..16] of real = (30.5,64.5,38.5,114.0,185.0,
    113.0,0.0,26.5,54.5,82.0,82.5,115.0,0.0,9.5,65.0,8.55);
    CK: array [1..16] of real = (2.0,4.1,1.4,6.8,7.2,5.2,0.0,
    1.9,1.8,6.7,3.7,4.0,0.0,1.0,1.6,0.55);
    BETI: array[1..10] of real = (
    0.3,0.35,0.4,0.45,0.5,0.55,0.6,0.65,0.7,0.75);

label
    1,3,5,7;

Procedure LinVent1;

var
    I: byte;

const
    Lvent11: array[1..10] of real =(
    0.5,0.5,1.5,1.5,1.5,1.5,2.5,2.5,3.5,3.5);
    Lvent21: array[1..10] of real = (
    0.0,0.5,0.5,0.5,0.5,0.5,0.5,1.5,2.5,3.5);
    Lvent31: array[1..10] of real = (
    0.5,0.5,0.5,1.0,1.5,1.5,1.5,2.5,3.5,4.5);
    Lvent41: array[1..10] of real = (
    0.0,0.0,0.0,0.5,0.5,0.5,1.0,1.5,2.0,3.0);
    Lvent51: array[1..10] of real =(
    0.5,0.5,0.5,0.5,1.5,1.5,2.5,2.5,2.5,3.5);
    Lvent61: array[1..10] of real =(
    0.5,0.5,0.5,0.5,8.5,12.5,17.5,23.5,27.5,29.9);

label
    1;

begin { LinVent1 }
    case NRA[1] of
        10: for I:=1 to 10 do Lvent[I]:= Lvent11[I];
        4: for I:=1 to 10 do Lvent[I]:= Lvent21[I];
        5: for I:=1 to 10 do Lvent[I]:= Lvent31[I];
        6: for I:=1 to 10 do Lvent[I]:= Lvent41[I];
        11: for I:=1 to 10 do Lvent[I]:= Lvent51[I];
        12: for I:=1 to 10 do Lvent[I]:= Lvent61[I];
    end;
    L1:= 0.0;
    if BET<BETI[1] then begin L1:= Lvent[1]; goto 1 end;
    if BET>BETI[10] then begin L1:= Lvent[10]; goto 1 end;

```

```

    for I:= 1 to 9 do
        if (BET >= BETI[1]) and (BET <= BETI[I+1]) then begin
            if(Lvent[I]=0.0) or (Lvent[I+1]=0.0) then goto 1;
            L1:= Lvent[I]+(Lvent[I+1]-Lvent[I])*(BET-BETI[I])/0.05; goto 1
        end;
    1: if L1=0.0 then exit; if LRA[1] < L1 then IERR:= 1
end; { LinVent1 }

```

Procedure LinVent2;

```

var
    J: byte;

```

```

const
    Lvent12: array[1..10] of real = (
        1.5,2.5,2.5,3.5,3.5,4.5,4.5,4.5,5.5,5.5);
    Lvent22: array[1..10] of real = (
        0.5,1.5,2.5,4.5,5.5,6.5,8.5,9.5,10.5,11.5);
    Lvent32: array[1..10] of real = (
        1.5,1.5,1.5,2.5,2.5,3.5,3.5,4.5,5.5,6.5);
    Lvent42: array[1..10] of real = (
        0.5,0.5,0.5,1.0,1.5,2.5,3.0,4.0,4.0,4.5);
    Lvent52: array[1..10] of real = (
        1.5,1.5,1.5,1.5,2.5,2.5,3.5,4.5,4.5,4.5);

```

```

label
    1;

```

```

begin { LinVent2 }
    DL:= 0.0;
    case NRA[I] of
        10: for J:= 1 to 10 do Lvent[J]:= Lvent12[J];
        4: for J:= 1 to 10 do Lvent[J]:= Lvent22[J];
        5: for J:= 1 to 10 do Lvent[J]:= Lvent32[J];
        6: for J:= 1 to 10 do Lvent[J]:= Lvent42[J];
        11: for J:= 1 to 10 do Lvent[J]:= Lvent52[J];
    end;
    if BET < BETI[1] then begin L1:= Lvent[1]; goto 1 end;
    if BET > BETI[10] then begin L1:= Lvent[10]; goto 1 end;
    for J:= 1 to 9 do
        if (BET >=BETI [J]) and (BET <= BETI[J+1]) then begin
            L1:= Lvent[J]+(Lvent[J+1]-Lvent[J])*(BET-BETI[J])/0.05; goto 1
        end;
    1: if LRA[I] < L1 then DL:= 0.5
end; { LinVent2 }

```

```

begin { C }
    IERR:= 0;
    case NSUZA[NNIT] of
        0: if BET <= 0.6 then EC0:= 0.6 else EC0:= BET;
        1: if BET <= 0.6 then EC0:= 0.8 else EC0:= 2*BET-0.4;
        2: EC0:= 1.2+1.5*r_(BET,4.0);
        3: begin
            EC0:= 0.7;
            if Re<=6.e4 then EC0:= 2.5;
            if (Re>6.e4) and (Re <= 1.e5) then EC0:= 2.0;
            if (Re>1.e5) and (Re <= 1.5e5) then EC0:= 1.5;
            if (Re>1.5e5) and (Re <= 2.0e5) then EC0:= 1.0
        end;

```

```

4: begin
    EC0:= 1.0;
    if Re <= 4.e4 then EC0:= 3.0;
    if (Re>4.e4) and (Re <= 1.2e5) then EC0:= 2.5;
    if (Re>1.2e5) and (Re <= 2.0e5) then EC0:= 1.5
end;
5: begin
    EC0:= 1.5;
    if Re <= 6.e4 then EC0:=3.0;
    if (Re>6.e4) and (Re <= 2.e5) then EC0:= 2.5
end;
end;
if NSUZA[NNit] <= 2 then begin
    if (NRA[1]=7) and (LRA[1]<22.0) then begin IERR:= 1; goto 1 end;
    if (NRA[1]=13) and (LRA[1]<3.0) then begin IERR:= 1; goto 1 end;
    if LRA[1]<5.0 then begin IERR:= 1; goto 1 end;
end
else begin
    if (NRA[1]=10) or (NRA[1]=4) or (NRA[1]=5) or
        (NRA[1]=6) or (NRA[1]=11) or (NRA[1]=12) then begin
        LinVent1; if IERR<>0 then goto 1; if L1 <> 0.0 then goto 3
        end;
    if (NRA[1]=7) and (LRA[1]<22.0) then begin IERR:= 1; goto 1 end;
    if (NRA[1]=13) and (LRA[1]<3.0) then begin IERR:= 1; goto 1 end;
    if LRA[1]<5.0 then begin IERR:=1; goto 1 end;
end;
3: NR:=AMOUNTR;
    L1:= AK[16]+BK[16]*r_(BET,CK[16]);
if (LRA[NR+1]/L1<0.5) or (LRA[NR+1]<0.5) then begin
    IERR:= 2; goto 1
end;
MR:= NRA[1];
L1:= AK[MR]+BK[MR]*r_(BET,CK[MR]);
MR:= NRA[2];
L2:= 0.5*(AK[MR]+BK[MR]*r_(0.7,CK[MR]));
if NSUZA[NNit]>2 then
    case NRA[2] of
        10: L2:= 2.75;
        4: L2:= 5.25;
        5: L2:= 2.75;
        6: L2:= 2.0;
        11: L2:= 2.25;
    end;
L21:= LRA[2]-LRA[1];
if (LRA[1] < L1) and (L21 < L2) then begin
    LRA[1]:= LRA[2]-L2; if LRA[1]<0.0 then LRA[1]:= abs(LRA[1])
end;
EL:=0.0;
if NSUZA[NNit] <= 2 then
    for I:= 1 to NR do begin
        if (NRA[I]=13) and (LRA[I]<5.0) then begin
            DL:= 0.5; goto 5
        end;
        MR:= NRA[I]; DL:=0.0; L1:=AK[MR]+BK[MR]*r_(BET,CK[MR]);
        if LRA[I]<L1 then DL:=1.0/(0.8+19.2/r_(L1/LRA[I],4.0));
5: if DL>EL then EL:= DL
    end
else
    for I:= 1 to NR do begin

```

```

        if (NRA[I]=10) or (NRA[I]=4) or (NRA[I]=5) or
            (NRA[I]=6) or (NRA[I]=11) then begin
            LinVent2; goto 7
        end;
        if (NRA[I]=13) and (LRA[I]<5.0) then begin
            DL:= 0.5; goto 7
        end;
        MR:= NRA[I]; DL:= 0.0; L1:= AK[MR]+BK[MR]*r_(BET,CK[MR]);
        if LRA[I]<L1 then DL:=1.0/(0.8+19.2/r_(L1/LRA[I],4.0));
7: if DL>EL then EL:= DL
end;
L1:= AK[16]+BK[16]*r_(BET,CK[16]);
if LRA[NR+1]<L1 then EL:= EL+0.5;
    if EL>1.0 then begin IERR:= 3; goto 1 end;
EE:= 0.0;
if NSUZA[NNit]=0 then begin
    EDM:=DT*sqrt(0.1*DP*(0.681-0.651*BET)/DAC[2]);
    if DAC[1]<EDM then begin
        A:= BET*(13.5-15.5*BET); B:= 117.0-106.0*r_(BET,1.9);
        EE:= 0.1*DP*sqrt(DT)*(A*DT/DAC[1]-B)/DAC[3]/sqrt(DAC[1])
    end; {endif}
    end; {endif}
EX:= 0.0;
EXMIN:= 0.0025*DT/(0.1+2.3*sqrt(BET)*sqrt(BET));
EXMAX:= 2.0*EXMIN;
if TAC[1]>EXMAX then begin IERR:= 4; goto 1 end;
if TAC[1]>EXMIN then EX:= 0.3;
EH:=0.0; HD:=TAC[3]/DT;
if HD>0.003 then begin
    HDC:=0.002*(TAC[2]/DT+0.4)/(0.1+2.3*sqrt(BET)*sqrt(BET));
    if (HD <= HDC) and (HD <= 0.05) then EH:= 0.2
    else begin
        IERR:= 5; goto 1
    end; {endif}
end; {endif}
ESH:=100.0*(KSH-1.0); EK:=100.0*(KK-1.0);
ERC:=sqrt(sqrt(ECO+EL+EE+EH+EX)+sqrt(ESH)+sqrt(EK));
1:
end; { C }

{-----}
{ Расчет методической составляющей погрешности определения коэффициента
расширения }
Procedure EPSI;
begin { EPSI }
    EEPS:= 0.0;
    if NSUBA[NNIT]<>23 then begin
        case NSUZA[NNIT] of
            0: if BET <= 0.75 then EEPS:= 4.0*DP/P
                else EEPS:= 8.0*DP/P;
            1: EEPS:=2.0*DP/P;
        else
            EEPS:= DP*(4.0+100.0*r_(BET, 8.0))/P;
        end;
    end; {endif}
end; { EPSI }

{-----}
{ Расчет методических погрешностей определения плотности, показателя
адиабаты и динамической вязкости }

```

Procedure EPH;

var  
N: byte;

const  
EPH3: array [1..2,1..3,1..3] of real=((0.3,0.4,0.2),  
(0.9,1.0,0.6),(2.0,3.0,2.0)),((0.6,1.3,0.4),(0.6,1.1,0.6),  
(2.0,3.0,2.0));

begin { EPH }

if NSUBA[NNIT] = 0 then begin  
  case NMETHKA[NNIT] of  
    0: begin  
      if ROS <= 0.70 then ERO:= 0.2  
      else begin  
        if ROS <= 0.76 then ERO:= 0.5  
        else ERO:= 1.7;  
        end; {endif}  
      end;  
    1: ERO:= 0.2;  
    2: ERO:= 0.2;  
    3: begin  
      if YR[13]=0.0 then begin  
        if T >-3.0 then begin  
          ERO:= EPH3[1,1,3]; EKAP:= EPH3[1,2,3];  
          EVIS:= EPH3[1,3,3]  
        end else begin  
          if P>60.0 then begin  
            ERO:= EPH3[1,1,2]; EKAP:= EPH3[1,2,2];  
            EVIS:= EPH3[1,3,2]  
          end else begin  
            ERO:= EPH3[1,1,1]; EKAP:= EPH3[1,2,1];  
            EVIS:= EPH3[1,3,1]  
          end; {endif}  
      end; {endif}  
    end else begin  
      if T >-3.0 then begin  
        ERO:= EPH3[2,1,3]; EKAP:= EPH3[2,2,3];  
        EVIS:= EPH3[2,3,3]  
      end else begin  
        if P>60.0 then begin  
          ERO:= EPH3[2,1,2]; EKAP:= EPH3[2,2,2];  
          EVIS:= EPH3[2,3,2]  
        end else begin  
          ERO:= EPH3[2,1,1]; EKAP:= EPH3[2,2,1];  
          EVIS:= EPH3[2,3,1]  
        end; {endif}  
      end; {endif}  
    end; {endif}  
  end;  
end; { endcase }  
if NMETHKA[NNIT]<>3 then begin  
  if T >-3 0 then begin  
    EKAP:= EPH3[1,2,3]; EVIS:= EPH3[1,3,3]  
  end else begin  
    if P>60.0 then begin  
      EKAP:= EPH3[1,2,2]; EVIS:= EPH3[1,3,2]

```

        end else begin
            EKAP:= EPH3[1,2,1]; EVIS:= EPH3[1,3,1]
        end; {endif}
    end; {endif}
    EKAP:= EKAP+2.0;
    if P <= 5.0 then EVIS:= EVIS+3.0
    else EVIS:= EVIS+6.0
    end; {endif}
end else begin
    N:= NSUBA[NNIT];
    EPHP(N,T,P,ERO,EKAP,EVIS);
end; {endif}
end; { EPH }
{-----}
{ Методические погрешности определения плотности, показателя адиабаты и
вязкости чистых веществ }
Procedure EPHP;

const
    TC:array [1..31] of real=(-82.0,32.0,96.0,134.0,152.0,187.0,
    196.0,234.0,300.0,326.0,77.0,9.0,92.0,327.0,357.0,307.0,227.0,
    -140.0,31.0,100.0,187.0,374.0,374.0,-119.0,-132.0,-268.0,
    -173.0,-73.0,-240.0,132.0,-147.0);
    PC:array [1..31,1..2]of real=((45.0,45.0),(48.0,48.0),
    (41.0,41.0),(36.0,36.0),(37.5,37.5),(33.0,33.0),(33.0,33.0),
    (29.0,29.0),(26.0,40.0),(25.0,25.0),(50.0,50.0),(50.0,50.0),
    (46.0,46.0),(50.0,40.0),(50.0,40.0),(50.0,30.0),(50.0,50.0),
    (35.0,30.0),(73.0,73.0),(90.0,90.0),(50.0,30.0),(217.0,217.0),
    (217.0,217.0),(50.0,50.0),(35.0,35.0),(2.3,2.3),(27.0,27.0),
    (49.0,49.0),(13.0,13.0),(113.0,113.0),(34.0,33.0));
    ER:array [1..31,1..3] of real=((0.06,0.09,0.09),
    (0.04,0.29,0.29),(0.03,0.21,0.21),(0.07,0.1,0.1),
    (0.4,0.6,0.6),(0.2,0.3,0.3),(0.2,0.3,0.3),(0.4,0.5,0.5),
    (0.3,0.5,0.2),(0.2,0.4,0.4),(0.5,1.3,1.0),(0.08,0.48,0.48),
    (0.1,1.0,1.0),(0.1,0.5,0.1),(0.1,0.5,0.1),(0.4,0.8,0.5),
    (1.0,3.0,1.5),(0.3,0.2,0.2),(0.1,0.4,0.4),(0.25,0.25,0.25),
    (0.4,1.0,0.5),(0.1,0.2,0.2),(0.03,0.03,0.03),(0.12,0.4,0.4),
    (0.02,0.05,0.05),(0.2,0.2,0.2),(0.25,0.3,0.15),
    (0.25,0.3,0.15),(0.25,0.2,0.2),(0.05,0.1,0.1),
    (0.03,0.04,0.04));
    EK:array [1..31,1..3] of real=((0.7,1.7,1.7),(0.2,2.4,2.4),
    (0.14,0.69,0.69),(4.0,4.0,4.0),(4.0,4.0,4.0),(1.5,1.5,1.5),
    (1.2,1.2,1.2),(0.8,0.8,0.8),(3.0,4.0,2.0),(3.0,2.0,2.0),
    (3.0,6.0,5.0),(0.4,1.4,1.4),(0.15,1.2,1.2),(2.0,4.0,1.5),
    (2.0,4.0,1.5),(5.0,10.0,6.0),(6.0,10.0,8.0),(0.5,2.0,2.0),
    (0.6,0.6,0.6),(1.5,1.5,1.5),(2.0,6.0,3.0),(2.5,2.5,2.5),
    (2.0,2.0,2.0),(3.0,1.8,1.8),(0.5,0.7,0.7),(3.4,3.4,3.4),
    (1.5,2.0,1.0),(1.5,2.0,1.0),(2.0,2.0,2.0),(4.0,4.0,4.0),
    (1.6,1.6,1.6));
    EV:array [1..31,1..2] of real=((4.7,1.5),(2.0,2.0),(3.0,2.0),
    (2.0,2.0),(3.0,2.0),(4.0,4.0),(4.0,3.0),(5.0,4.0),(1.0,4.5),
    (2.0,2.0),(2.0,5.0),(2.0,2.0),(5.0,3.0),(1.0,4.5),(1.0,4.5),
    (1.5,3.0),(6.0,6.0),(1.0,3.0),(0.7,1.4),(2.3,2.3),(1.0,5.0),
    (0.47,1.1),(2.0,2.0),(1.8,1.3),(4.7,4.0),(2.6,2.6),(2.0,2.0),
    (2.0,2.0),(3.0,5.0),(2.0,2.0),(1.1,4.0));
begin { EPHP }
    if((9<=N) and (N<=11)) or((14<=N) and (N<=17)) or(N=21) or
    ((27<=N) and (N<=28)) then begin
        if P <= PC[N,1] then begin

```

```

        ERO:= ER[N,1]; EKAP:= EK[N,1]
    end else begin
        if T<=TC[N] then begin
            ERO:= ER[N,2]; EKAP:= EK[N,2]
        end else begin
            ERO:= ER[N,3]; EKAP:= EK[N,3]
        end; {endif}
    end; {endif}
end else begin
    if T<=TC[N] then begin
        ERO:= ER[N,1]; EKAP:= EK[N,1]
    end else begin
        ERO:= ER[N,2]; EKAP:= EK[N,2]
    end; {endif}
end; {endif}
if P<=PC[N,2] then EVIS:= EV[N,1]
else EVIS:= EV[N,2];
end { EPHP }
{-----}
Procedure ROW;

var
    E1ROSC,E1RORC,
    E2ROS,E3ROS,EDROS,
    ERO1SC,ERO1RC,ERO2S,EVROS: real;
    YRO: array [1..2] of real;

begin { ROW }
    if (E[1]<>0) or (E[2]<>0) then
    begin
        E1ROSC:= E[1];
        E1RORC:= E[2]
    end
    else begin
        if E[3]<>0 then
        begin
            E1ROSC:= E[3];
            E1RORC:= 0.0
        end
        else begin
            E1ROSC:= sqrt(sqr(E[4])+sqr(E[5]));
            E1RORC:= sqrt(sqr(E[6])+sqr(E[7]))
        end;
    end; {endif}
    E2ROS:= E[8]*(T1HO-T1LO);
    E3ROS:= E[9]*(UH-UL);
    EDROS:= sqrt(sqr(E1ROSC)+sqr(E2ROS)+sqr(E3ROS)+sqr(E[10]));
    if (E[11]<>0) or (E[12]<>0) then
    begin
        ERO1SC:=E[11];
        ERO1RC:=E[12];
    end
    else begin
        if E[13]<>0 then
        begin
            ERO1SC:= E[13];
            ERO1RC:= 0.0
        end
    end
    else begin

```

```

        ERO1SC:= sqrt(sqr(E[14])+sqr(E[15]));
        ERO1RC:= sqrt(sqr(E[16])+sqr(E[17]));
    end;
end; {endif}
ERO2S:= E[18]*(T2HO-T2LO);
EVROS:= sqrt(sqr(ERO1SC)+sqr(ERO2S)+sqr(E[19])+sqr(E[20]));
YRO[1]:= (E[21]-E[22])/RoM; YRO[2]:= (sqrt(E[21])-sqrt(E[22]))/sqrt(RoM);
EROWS:= sqrt(sqr(YRO[1]*EDROS)+sqr(YRO[2]*EVROS));
EROWR:= sqrt(sqr(YRO[1]*ElRORC)+sqr(YRO[2]*ERO1RC));
EROW:= sqrt(sqr(EROWS)+sqr(EROWR))
end; { ROW }

{-----}
{ Расчет расхода и количества среды }
Procedure QCalc;

var
    I,IBeg,IFin: byte;
    Bet4,Ec,Eps,Rd,Psi,Rk,Cb,L1,L2,Alfa,
    Qcb,ARe,R0,KRe,KCb,Log: real;
    HsS: string[10]; Code: integer;

label
    1,3;

const
    HsSubs1: array[1..31] of real= (37.12,65.43,93.85,122.8,123.6,0.0,
        0.0,0.0,0.0,0.0,54.47,59.04,86.88,
        0.0,0.0,0.0,52.70,11.77,0.0,23.61,
        0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
        11.88,16.11,0.0);
    HsSubs2: array[1..31] of real= (33.43,59.87,86.37,113.4,114.1,0.0,
        0.0,0.0,0.0,0.0,52.62,55.34,81.29,
        0.0,0.0,0.0,48.94,11.77,0.0,21.75,
        0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0,
        10.04,13.32,0.0);
    RocSubs: array[1..31] of real= (0.6682,1.2601,1.8641,2.488,
        2.4956,3.147,3.174,3.898,4.755,
        5.812,1.09,1.1733,1.776,3.469,
        4.294,1.587,2.045,1.1649,1.8393,
        1.4311,2.718,0.787,998.23,
        1.33116,1.20445,0.16631,0.8385,
        1.6618,0.08375,0.716,1.1649);
    Rocii: array[1..16] of real= (0.66692,1.25004,1.83315,2.41623,
        2.41623,2.99934,2.99934,3.58246,
        4.16558,4.74869,1.16455,1.82954,
        1.41682,0.16639,1.1644,0.0838);

    CalcTpNg ='TpNg.exe'; CalcTpSubs ='TpSubs.exe';
begin { QCalc }
    if NRQ<>0 then begin
{ Расчет физических свойств среды }
        assign(Fl,'IRD'); rewrite(Fl);
        if NSubA[NNIT]=0 then begin
            writeln(Fl,NMethKA[NNIT]);
            if NMETHKA[NNIT] >= 2 then begin
                IBeg:= 1;
                repeat
                    IFin:= IBeg+3;
                    for I:= IBeg to IFin do write(Fl,YR[I]:14,BL);

```



```

        writeln(Fl),IBeg:= IFin+1
    until IBeg>16;
end
    else begin
        if NVARYA[NNIT]=0 then begin
            Roc:= 0.0; for I:= 1 to 16 do Roc:= Roc+YR[I]*Rocii[I];
            Zc:= 0.0; for I:= 1 to 16 do Zc:= Zc+YR[I]*Bi[I];
            Roc:= Roc/(1.0-sqr(Zc));
            Ya:= YR[11]; Yy:= YR[12];
            end; {endif}
            writeln(Fl,Roc:14,BL,Ya:14,BL,Yy:14)
        end
    end
else
writeln(Fl,NSubA[NNIT]);
writeln(Fl,P:14,BL,T:14);
close(Fl);
TextColor(7);
gotoxy(19,9);
writeln(' |-----| ');
gotoxy(19,10);
writeln(' | ');
gotoxy(19,11);
write(' | ');
TextColor(135);
write('Ж Д И Т Е');
TextColor(7);
writeln(' | ');
gotoxy(19,12);
writeln(' | ');
gotoxy(19,13);
writeln(' | В Ы П О Л Н Я Е Т С Я Р А С Ч Е Т | ');
gotoxy(19,14);
writeln(' | ');
gotoxy(19,15);
writeln(' |_____| ');
if NSubA[NNIT]=0 then begin
    gotoxy(21,12);
    swapvectors; exec(CalcTpNg, CalcTpNg); swapvectors;
    TextColor(7);
    gotoxy(19,9);
    writeln(' |-----| ');
    gotoxy(19,10);
    writeln(' | ');
    gotoxy(19,11);
    write(' | ');
    TextColor(135);
    write('Ж Д И Т Е');
    TextColor(7);
    writeln(' | ');
    gotoxy(19,12);
    writeln(' | ');
    gotoxy(19,13);
    writeln(' | В Ы П О Л Н Я Е Т С Я Р А С Ч Е Т | ');
    gotoxy(19,14);
    writeln(' | ');
    gotoxy(19,15);
    writeln(' |_____| ');
end
end

```

```

        else begin
            swapvectors; exec(CalcTpSubs, CalcTpSubs); swapvectors;
            Roc:= RocSubs[NSubA[NNIT]]
        end;
assign(Fl,'IRD'); reset(Fl);
if (NSubA[NNIT]=0) and (NMethKA[NNIT] >= 2) then
    readln(Fl,Roc);
if NSubA[NNIT]=0 then begin
    readln(Fl,Hs[1],Hs[2]);
    for I=1 to 2 do begin
        Str(Hs[I]:10,HsS); Val(HsS,Hs[I],Code)
    end;
end
else begin
    Hs[1]:= HsSubs1[NSubA[NNIT]];
    Hs[2]:= HsSubs2[NSubA[NNIT]]
end;

    readln(Fl,ROX,KAPPAX,MUX);
    close(Fl); erase(Fl);
    if NSubA[NNIT]=0 then str(Roc:6:4, RocStr);
end; {endif}
if (NSubA[NNit]=0) and (VarRoA[NNit]=1) and
(NRQ<>0) then ROX:= RoM;
{ Расчет: 1) диаметров сужающего устройства и измерительного трубопровода
при рабочей температуре; 2) коэффициента скорости входа }
Dd:= (1.0+AlfaSU*(T-20.0))*Dd0;
Dt:= (1.0+AlfaT*(T-20.0))*Dt0;
Bet:= Dd/Dt; Bet4:= sqr(Bet)*sqr(Bet);
Ec:= 1.0/sqrt(1.0 - Bet4);
{ Расчет коэффициента расширения }
Eps:= 1.0;
if NSubA[NNIT]<>23 then begin
    if NSuzA[NNIT]=0 then
        Eps:= 1.0-(0.41+0.35*Bet4)*Dp/P/KAPPAX;
    if NSuzA[NNIT]<>0 then begin
        Psi:= 1.0-Dp/P;
        Eps:=KAPPAX*r_(Psi,2.0/KAPPAX)/(KAPPAX-1.0)*
            (1.0-Bet4)/(1.0-Bet4*r_(Psi,2.0/KAPPAX))*
            (1.0-r_(Psi,(KAPPAX-1.0)/KAPPAX))/
            (1.0-Psi);
        Eps:= sqrt(Eps)
    end;
end;
{ Расчет поправочного коэффициента на шероховатость внутренней поверхности
измерительного трубопровода без учета числа Рейнольдса }
KSh:= 1.0;
if (NSuzA[NNit] <= 2) and (RSh <> 0.0) then begin
    ARe:= 0.5; Rd:= RSh/Dt; Log:= Ln(Rd*1.e4)/2.3026;
    if NSuzA[NNit]=0 then begin
        if Log <= (1.0/10.0/Bet4+8.0)/14.0 then begin
            R0:= 0.0; goto 1;
        end;
        R0:=0.07*Log-0.04
    end
    else begin
        if Log <= (1.0/10.0/Bet4+5.0)/9.0 then begin
            R0:= 0.0; goto 1;
        end;
    end;
end;

```

```

        R0:= 0.045*Log-0.025
    end;
    KSh:= 1.0+Bet4*R0*ARE
end;
1:;
{ Расчет поправочного коэффициента на притупление входной кромки отверстия
диафрагмы }
    Kk:= 1.0;
    if(NSuzA[NNIT]=0) and (Dd0<125.0) then begin
        if TauP= 0.0 then Rk:= Rn;
        if TauP= 1.0 then Rk:= 0.0292+0.85*Rn;
        if (TauP <> 0.0) and (TauP <> 1.0) then
            Rk:= 0.195-(0.195-Rn)*(1.0-Exp(-TauP/3.0))*
                3.0/TauP;
        Kk:=1.0547-0.0575*Exp(-149.0*Rk/Dd)
    end;
{ Расчет коэффициента истечения при числе Рейнольдса, стремящемся к
бесконечности }
    if NSuzA[NNit]=0 then begin
        L1:=0 .0; L2:= 0.0;
        if SodSuA[NNit]=1 then begin
            L1:= 25.4/Dt; L2:= L1;
            if L1 >= 0.4333 then L1:= 0.039 else L1:= 0.09*L1
        end;
        if SodSuA[NNit]=2 then begin L1:= 0.039; L2:= 0.47 end;
        Cb:= 0.5959+0.0312*r_(Bet, 2.1)-0.184*sqr(Bet4)+
            L1*Bet4/(1.0 - Bet4)-0.0337*L2*r_(Bet,3)
        end;
        if NSuzA[NNIT] = 1 then Cb:= 0.99-0.2262* r_(Bet,4.1);
        { Для сопла Вентури Cb = C, так как KRe = 1 }
        if NSuzA[NNIT]=2 then Cb:= 0.9858-0.196*r_(Bet,4.5);
        { Для труб Вентури Cb = C = const, так как KRe = 1 и Re > 2.e5 }
        case NSuzA[NNIT] of
            3: Cb:= 0.984;
            4: Cb:= 0.995;
            5: Cb:= 0.985
        end;
{ Расчет коэффициента расхода сужающего устройства и расхода при числе
Рейнольдса, стремящемся к бесконечности }
        Alfa:= Cb*Es;
        Qcb:= 0.039986*Alfa*KSh*Kk*Eps*sqr(Dd)*
            sqrt(1.e3*Dp*ROX)/Roc;
        Re:= 4.e6*Qcb*Roc/3.6/3.141592653/MUX/Dt;
{ Расчет поправочного коэффициента на число Рейнольдса }
        case NSuzA[NNIT] of
            0: KRe:= 1.0+1.426/(1.0+Cb*r_(Re,0.75)/64.28/
                r_(Bet, 2.5));
            1: KRe:= 1.0+0.86/(1.0+Cb*r_(Re, 1.15)/923.9/
                sqr(Bet)/(33.0*r_(Bet,2.15)-17.5));
            2: KRe:= 1.0;
            3: KRe:= 1.0;
            4: KRe:= 1.0;
            5: KRe:= 1.0;
        end;
{ Определение действительного значения числа Рейнольдса }
        Re:= Re*KRe;
{ Расчет поправочного коэффициента на шероховатость внутренней поверхности
измерительного трубопровода с учетом числа Рейнольдса для всех сужающих
устройств, кроме труб Вентури }

```

```

if (NSuzA[NNit] <= 2) and (RSh <> 0.0) then begin
  Qcb:= Qcb/KSh;
  if Re>1.e4 then begin
    if Re<1.e6 then ARe:= 1.0-sqr(Ln(Re)/2.3026-6.0)/4.0
      else ARe:= 1.0;
    KSh:= 1.0+Bet4*RO*ARe
  end;
  if (Re <= 1.e4) or (KSh <= 1.0005) then KSh:= 1.0;
  Qcb:= Qcb*KSh
end;
{ Определение коэффициента истечения для труб Вентури в зависимости
от числа Рейнольдса }
if (Re<2.e5) and (NSuzA[NNIT]>2) then
  case NSuzA[NNIT] of
    3: begin
      if Re <= 6.e4 then Cb:= 0.957;
      if (Re > 6.e4) and (Re <= 1.e5) then Cb:= 0.966;
      if (Re > 1.e5) and (Re <= 1.5e5) then Cb:= 0.976;
      if Re > 1.5e5 then Cb:= 0.982
      end;
    4: begin
      if Re <= 4.e4 then Cb:= 0.970;
      if (Re > 4.e4) and (Re <= 8.e4) then Cb:= 0.977;
      if (Re > 8.e4) and (Re <= 1.2e5) then Cb:= 0.992;
      if Re > 1.2e5 then Cb:= 0.998
      end;
    5: begin
      if Re <= 6.e4 then Cb:= 0.960;
      if (Re > 6.e4) and (Re <= 1.e5) then Cb:= 0.970;
      if Re > 1.e5 then Cb:= 0.980
      end;
  end;
  KCb:= 1.0;
{ Определение поправки на коэффициент истечения для труб Вентури,
в зависимости от числа Рейнольдса }
if (Re<2.e5) and (NSuzA[NNIT]>2) then
  case NSuzA[NNIT] of
    3: KCb:= Cb/0.984;
    4: KCb:= Cb/0.995;
    5: KCb:= Cb/0.985;
  end;
{ Расчет расхода и количества среды при действительном значении
числа Рейнольдса }
Qc:= Qcb*KRe*KCb; Vcv:= Qc*TauAv; Vc:= Vcv;
if (NSubA[NNit]=22) or (NSubA[NNit]=23) then begin
  Vm:= Vcv*Roc/1000.0; Qc:= Qc*Roc/1000.0 end;
end; { QCalc }
{ ----- }
{ Расчет погрешностей определения теплоты сгорания }
Procedure HSP;

const
  DHS: array[1..31] of real =(0.11,0.11,0.28,0.42,0.42,0.0,0.0,0.0,
    0.0,0.0,0.14,0.22,0.28,0.0,0.0,0.0,
    0.41,0.14,0.0,0.41,0.0,0.0,0.0,0.0,
    0.0,0.0,0.0,0.0,0.11,0.5,0.0);
  HsNg1: array[1..16] of real= (37.04,64.91,92.29,119.7,119.3,
    147.0,146.8,174.5,201.8,229.2,0.0,
    0.0,23.37,0.0,11.76,11.89);

```

```

HsNg2 array[1..16] of real= (33.37,59.39,84.94,110.5,110.1,
                             136.0,135.7,161.6,187.1,212.7,0.0,
                             0.0,21.53,0.0,11.76,10.05);
var
  H1,H2,SH1,SH2: real; I: byte;

begin { HSP }
  if NSUBA[NNIT]=0 then begin
    if(NMETHKA[NNIT]=0) or (NMETHKA[NNIT]=1) then begin
      H1:= (0.51447*ROSC+0.05603-0.65689*YA-YY);
      H2:= (0.52190*ROSC+0.04242-0.65197*YA-YY);
      ERRH1:= sqrt(sqr(0.51447*ROSC*EROS)+sqr(YY*EROSA[18])+
                   sqr(0.65689*YA*EROSA[17]))/H1;
      ERRH2:= sqrt(sqr(0.52190*ROSC*EROS)+sqr(YY*EROSA[18])+
                   sqr(0.65197*YA*EROSA[17]))/H2
    end else begin
      SH1:= 0.0; SH2:= 0.0; ERRH1:= 0.0; ERRH2:= 0.0; Zc:= 0.0;
      for I:= 1 to 16 do begin
        Zc:= Zc+RIM[I]*Bi[I]; SH1:= SH1+RIM[I]*HSNG1[I];
        SH2:= SH2+RIM[I]*HSNG2[I];
        ERRH1:= ERRH1+sqr(RIM[I]*HSNG1[I]*EROSA[I+6]);
        ERRH2:= ERRH2+sqr(RIM[I]*HSNG2[I]*EROSA[I+6])
      end; {endfor}
      Zc:= 1.0-sqr(Zc); ERRH1:= Zc*sqrt(ERRH1)/SH1;
                          ERRH2:= Zc*sqrt(ERRH2)/SH2

      end; {endif}
    end else begin
      ERRH1:= DHS[NSUBA[NNIT]]; ERRH2:= ERRH1
    end; {endif}
  end { HSP }
{ ----- }
  function r_; begin r_:= exp(R*ln(A)) end;
END. { ERRQSNX }

```