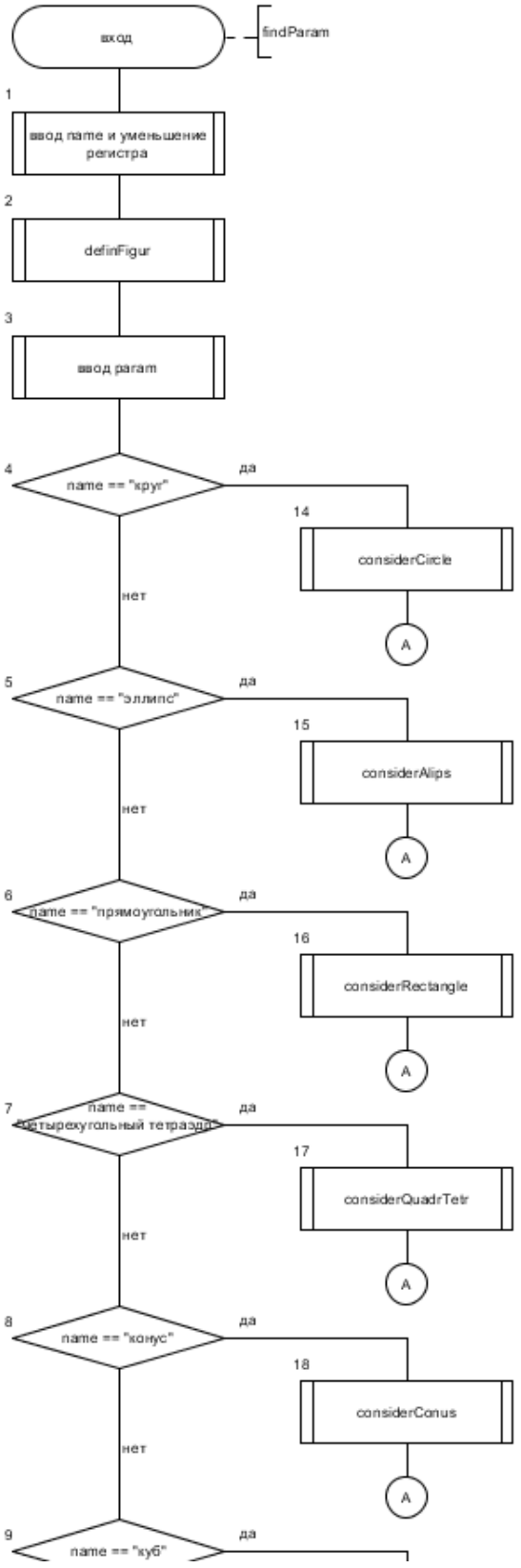
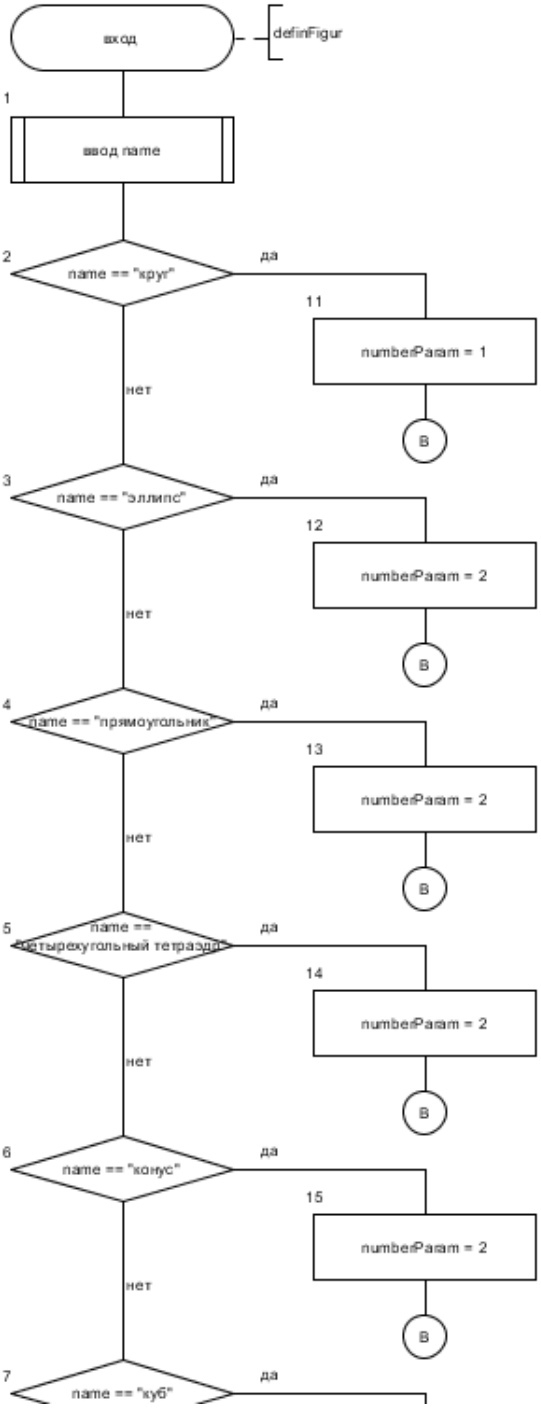
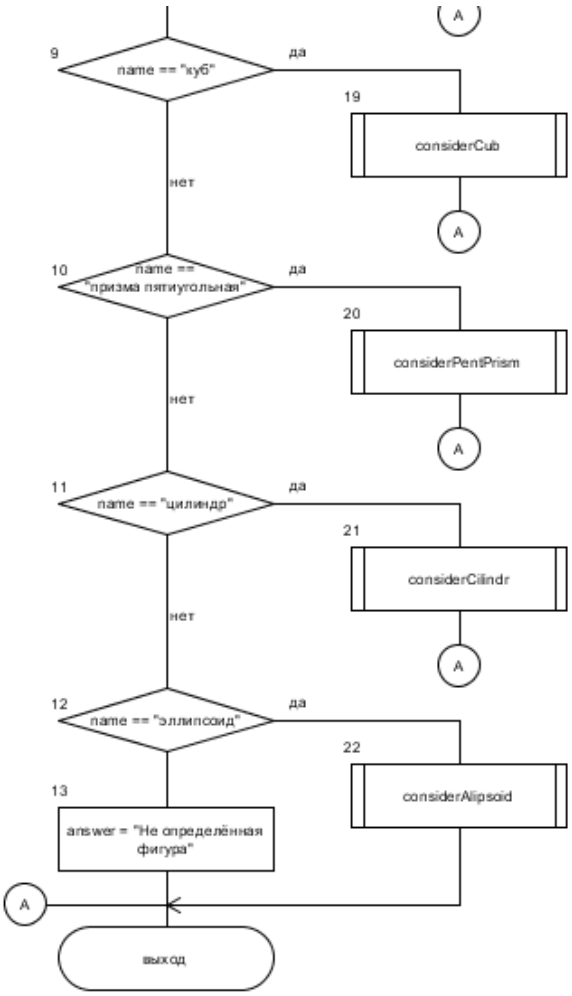
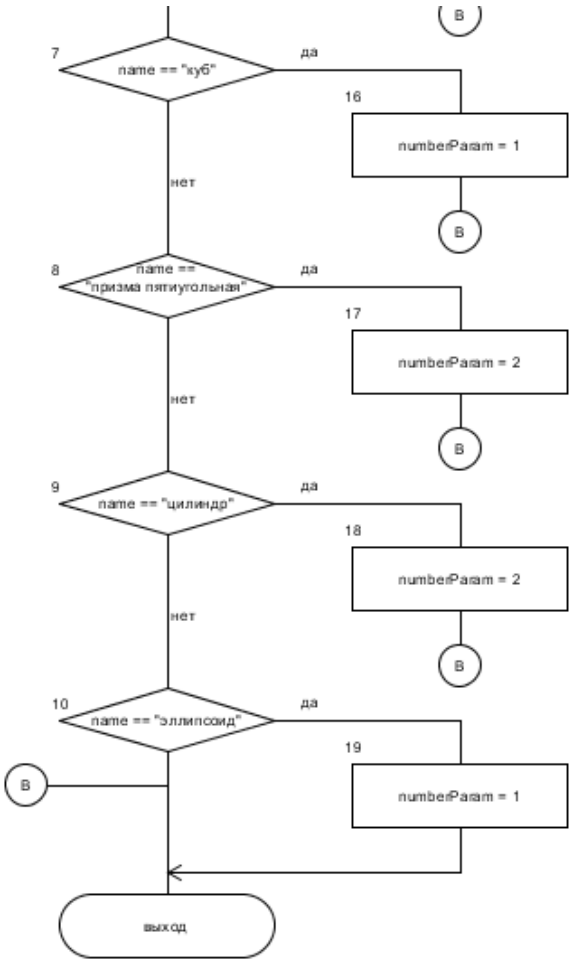
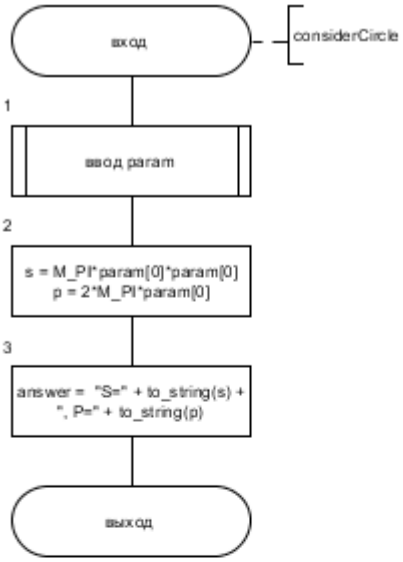
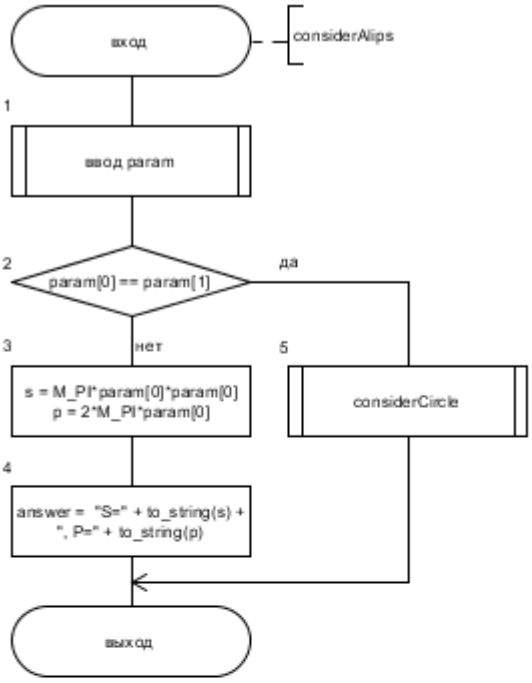

вход
findParam
1
ввод name и уменьшение регистра
2
definFigur
3
ввод param
4
name == "круг"
да
14
considerCircle
A
нет
5
name == "эллипс"
да
15
considerAlips
A
нет
6
name == "прямоугольник"
да
16
considerRectangle
A
нет
7
name == "четырехугольный тетраэдр"
да
17
considerQuadrTetr
A
нет
8
name == "конус"
да
18
considerConus
A
нет
9
name == "куб"
да

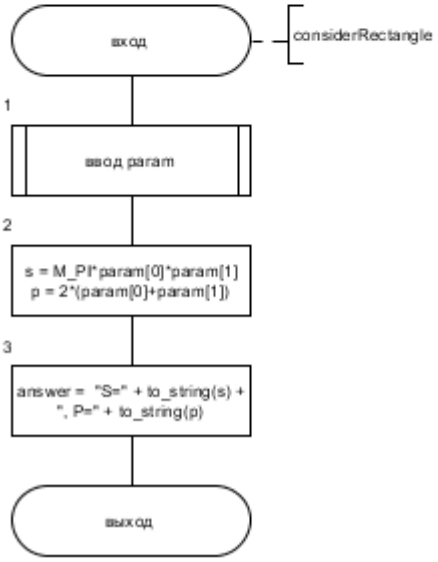

вход
definFigur
1
ввод name
2
name == "круг"
да
11
numberParam = 1
B
нет
3
name == "эллипс"
да
12
numberParam = 2
B
нет
4
name == "прямоугольник"
да
13
numberParam = 2
B
нет
5
name == "четырехугольный тетраэдр"
да
14
numberParam = 2
B
нет
6
name == "конус"
да
15
numberParam = 2
B
нет
7
name == "куб"
да

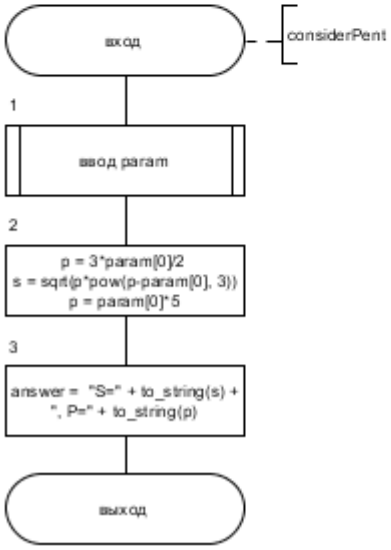

A
9
name == "куб"
да
19
considerCub
нет
A
10
name ==
"призма пятиугольная"
да
20
considerPentPrism
нет
A
11
name == "цилиндр"
да
21
considerCilindr
нет
A
12
name == "эллипсоид"
да
22
considerAlipsoid
13
answer = "Не определённая
фигура"
A
выход

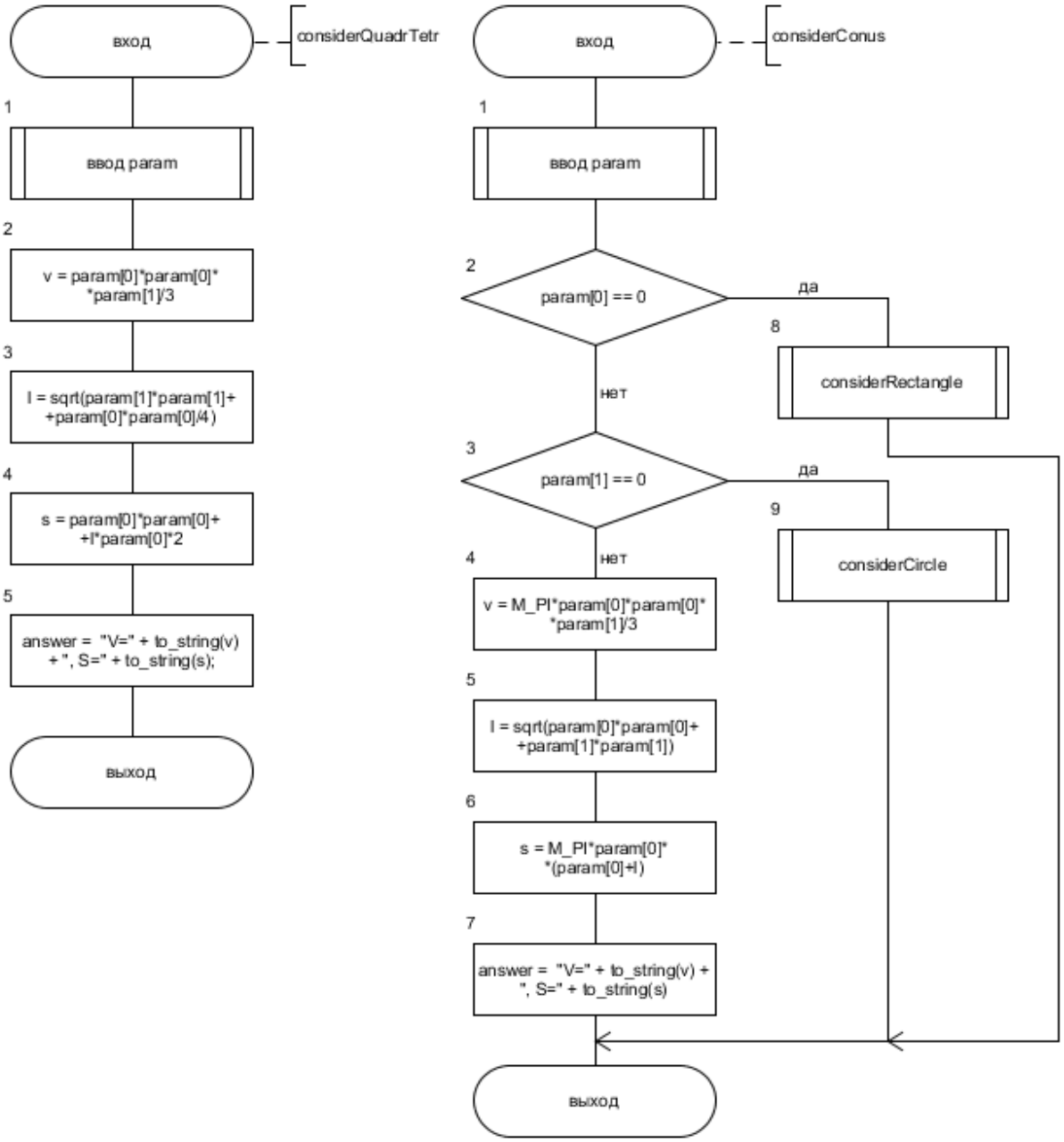

B
7
name == "куб"
да
16
numberParam = 1
нет
B
8
name ==
"призма пятиугольная"
да
17
numberParam = 2
нет
B
9
name == "цилиндр"
да
18
numberParam = 2
нет
B
10
name == "эллипсоид"
да
19
numberParam = 1
B
выход

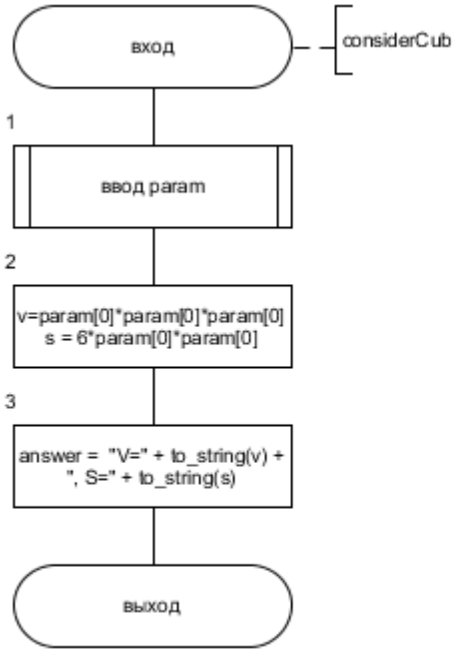

вход
considerCircle
1
ввод param
2
s = M_PI*param[0]*param[0]
p = 2*M_PI*param[0]
3
answer = "S=" + to_string(s) +
", P=" + to_string(p)
выход

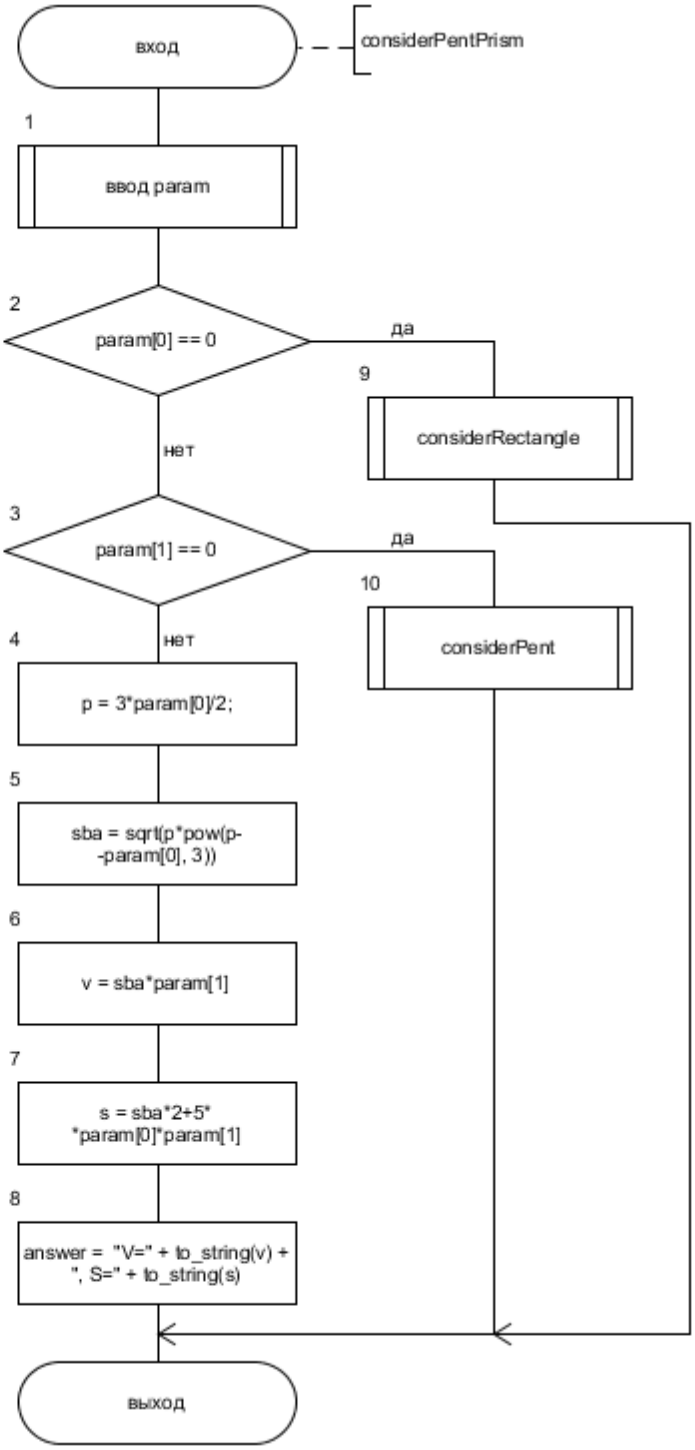

вход
considerAlips
1
ввод param
2
param[0] == param[1]
да
нет
3
s = M_PI*param[0]*param[0]
p = 2*M_PI*param[0]
4
answer = "S=" + to_string(s) +
", P=" + to_string(p)
5
considerCircle
выход

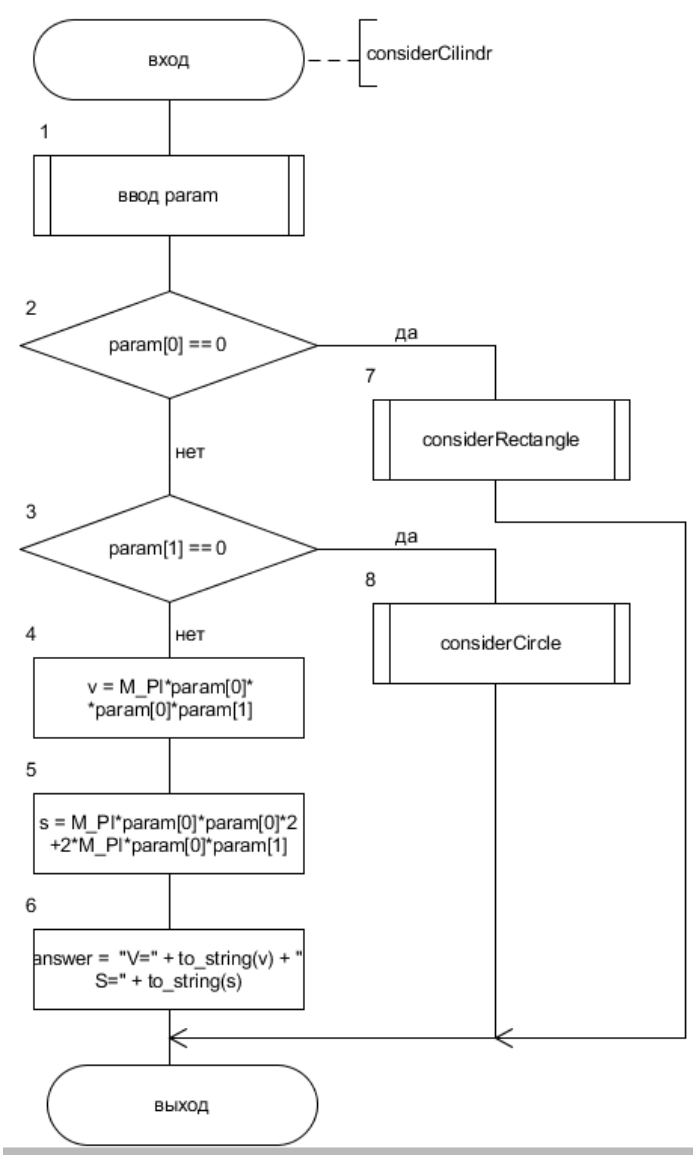

вход
considerRectangle
1
ввод param
2
s = M_PI*param[0]*param[1]
p = 2*(param[0]+param[1])
3
answer = "S=" + to_string(s) +
", P=" + to_string(p)
выход

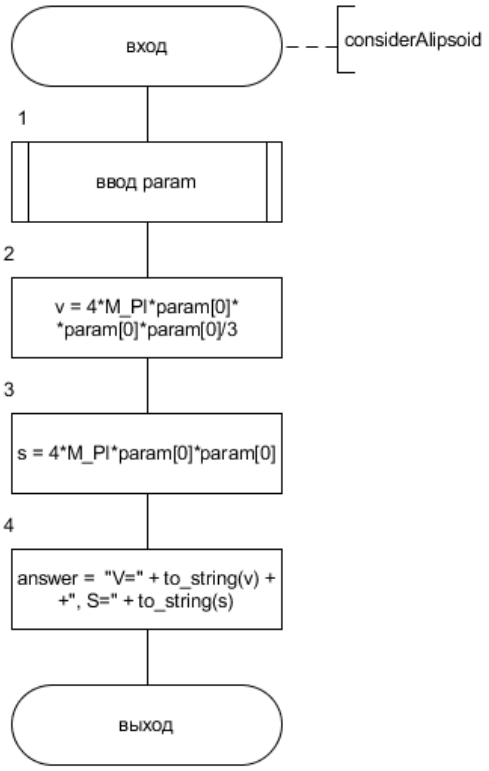

вход
considerPent
1
ввод param
2
p = 3*param[0]/2
s = sqrt(p*pow(p-param[0], 3))
p = param[0]*5
3
answer = "S=" + to_string(s) +
", P=" + to_string(p)
выход


вход
considerQuadrTetr
1
ввод param
2
v = param[0]*param[0]*
*param[1]/3
3
l = sqrt(param[1]*param[1]+
+param[0]*param[0]/4)
4
s = param[0]*param[0]+
+l*param[0]*2
5
answer = "V=" + to_string(v)
+ ", S=" + to_string(s);
выход
вход
considerConus
1
ввод param
2
param[0] == 0
да
8
considerRectangle
нет
3
param[1] == 0
да
9
considerCircle
нет
4
v = M_PI*param[0]*param[0]*
*param[1]/3
5
l = sqrt(param[0]*param[0]+
+param[1]*param[1])
6
s = M_PI*param[0]*
*(param[0]+l)
7
answer = "V=" + to_string(v) +
", S=" + to_string(s)
выход


вход
considerCub
1
ввод param
2
v=param[0]*param[0]*param[0]
s = 6*param[0]*param[0]
3
answer = "V=" + to_string(v) +
", S=" + to_string(s)
выход


вход
considerPentPrism
1
ввод param
2
param[0] == 0
да
9
considerRectangle
нет
3
param[1] == 0
да
10
considerPent
нет
4
p = 3*param[0]/2;
5
sba = sqrt(p*pow(p-
-param[0], 3))
6
v = sba*param[1]
7
s = sba*2+5*
*param[0]*param[1]
8
answer = "V=" + to_string(v) +
", S=" + to_string(s)
выход






```
#define _USE_MATH_DEFINES

#include <iostream>
#include <cmath>

using namespace std;

int definFigur(string name)
{
  int numberParam = 0;
  // двухмерные фигуры
  if(name == "круг")
  {
```

```
  numberParam = 1;
  return numberParam;
}
if(name == "эллипс")
{
  numberParam = 2;
  return numberParam;
}
if(name == "прямоугольник")
{
  numberParam = 2;
  return numberParam;
}
// трёхмерные фигуры
if(name == "четырехугольный тетраэдр")
{
  numberParam = 2;
  return numberParam;
}
if(name == "конус")
{
  numberParam = 2;
  return numberParam;
}
if(name == "куб")
{
  numberParam = 1;
  return numberParam;
}
if(name == "призма пятиугольная")
{
  numberParam = 2;
```

```
    return numberParam;
  }
  if(name == "цилиндр")
  {
    numberParam = 2;
    return numberParam;
  }
  if(name == "эллипсоид")
  {
    numberParam = 1;
    return numberParam;
  }

  return numberParam;
}

string considerCircle(double param[3])
{
  double s = M_PI*param[0]*param[0];
  double p = 2*M_PI*param[0];
  string answer =  "S=" + to_string(s) + ", P=" + to_string(p);
  return answer;
}

string considerAlips(double param[3])
{
  if(param[0] == param[1]) return considerCircle(param);
  double s = M_PI*param[0]*param[1];
  double p = 4*(M_PI*param[0]*param[1]+param[0]-param[1])/(param[0]+param[1]);
  string answer =  "S=" + to_string(s) + ", P=" + to_string(p);
  return answer;
}
```

```
string considerRectangle(double param[3])
{
  double s = param[0]*param[1];
  double p = 2*(param[0]+param[1]);
  string answer =  "S=" + to_string(s) + ", P=" + to_string(p);
  return answer;
}

string considerPent(double param[3])
{
  double p = 3*param[0]/2;
  double s = sqrt(p*pow(p-param[0], 3));
  p = param[0]*5;
  string answer =  "S=" + to_string(s) + ", P=" + to_string(p);
  return answer;
}

string considerQuadrTetr(double param[3])
{
  if(param[1] == 0) return considerRectangle(param);
  double v = param[0]*param[0]*param[1]/3;
  double l = sqrt(param[1]*param[1]+param[0]*param[0]/4);
  double s = param[0]*param[0]+l*param[0]*2;
  string answer =  "V=" + to_string(v) + ", S=" + to_string(s);
  return answer;
}

string considerConus(double param[3])
{
  if(param[0] == 0) return considerRectangle(param);
  if(param[1] == 0) return considerCircle(param);
```

```
  double v = M_PI*param[0]*param[0]*param[1]/3;
  double l = sqrt(param[0]*param[0]+param[1]*param[1]);
  double s = M_PI*param[0]*(param[0]+l);
  string answer =  "V=" + to_string(v) + ", S=" + to_string(s);
  return answer;
}

string considerCub(double param[3])
{
  double v = param[0]*param[0]*param[0];
  double s = 6*param[0]*param[0];
  string answer =  "V=" + to_string(v) + ", S=" + to_string(s);
  return answer;
}

string considerPentPrism(double param[3])
{
  if(param[0] == 0) return considerRectangle(param);
  if(param[1] == 0) return considerPent(param);
  double p = 3*param[0]/2;
  double sba = sqrt(p*pow(p-param[0], 3));
  double v = sba*param[1];
  double s = sba*2+5*param[0]*param[1];
  string answer =  "V=" + to_string(v) + ", S=" + to_string(s);
  return answer;
}

string considerCilindr(double param[3])
{
  if(param[0] == 0) return considerRectangle(param);
  if(param[1] == 0) return considerCircle(param);
  double v = M_PI*param[0]*param[0]*param[1];
```

```
  double s = M_PI*param[0]*param[0]*2+2*M_PI*param[0]*param[1];
  string answer =  "V=" + to_string(v) + ", S=" + to_string(s);
  return answer;
}

string considerAlipsoid(double param[3])
{
  double v = 4*M_PI*param[0]*param[0]*param[0]/3;
  double s = 4*M_PI*param[0]*param[0];
  string answer =  "V=" + to_string(v) + ", S=" + to_string(s);
  return answer;
}

string findParam(string name)
{
  int numberParam = 0;
  // уменьшаем регистр слова
  for(int i = 0; i < name.size(); i++) name[i] = tolower(name[i]);
  numberParam = definFigur(name);
  if(numberParam == 0) return "Не определённая фигура";
  double param[3] = {0};
  for(int i = 0; i < numberParam; i++)
  {
    cin >> param[i];
    if(param[i] < 0)
    {
      cout << "Передан отрицательный параметр, он взят по модулю" << "\n";
      param[i] = -param[i];
    }
  }

  if(name == "круг") return considerCircle(param);
```

```
  if(name == "эллипс") return considerAlips(param);
  if(name == "прямоугольник") return considerRectangle(param);
  if(name == "четырехугольный тетраэдр") return considerQuadrTetr(param);
  if(name == "конус") return considerConus(param);
  if(name == "куб") return considerCub(param);
  if(name == "призма пятиугольная") return considerPentPrism(param);
  if(name == "цилиндр") return considerCilindr(param);
  if(name == "эллипсоид") return considerAlipsoid(param);

  return "Не определённая фигура";
}

int main()
{
  string name, answer, a;
  getline(cin, name);
  answer = findParam(name);
  cout << answer;
  return 0;
}
```