Лабораторная работа № 5 «Пользовательские процедуры и функции.»

Цель: разработать и научиться использовать алгоритмы, в вычислениях которых было бы рационально использовать пользовательские процедуры и функции

Оборудование: ПК, среда разработки «PascalABC»

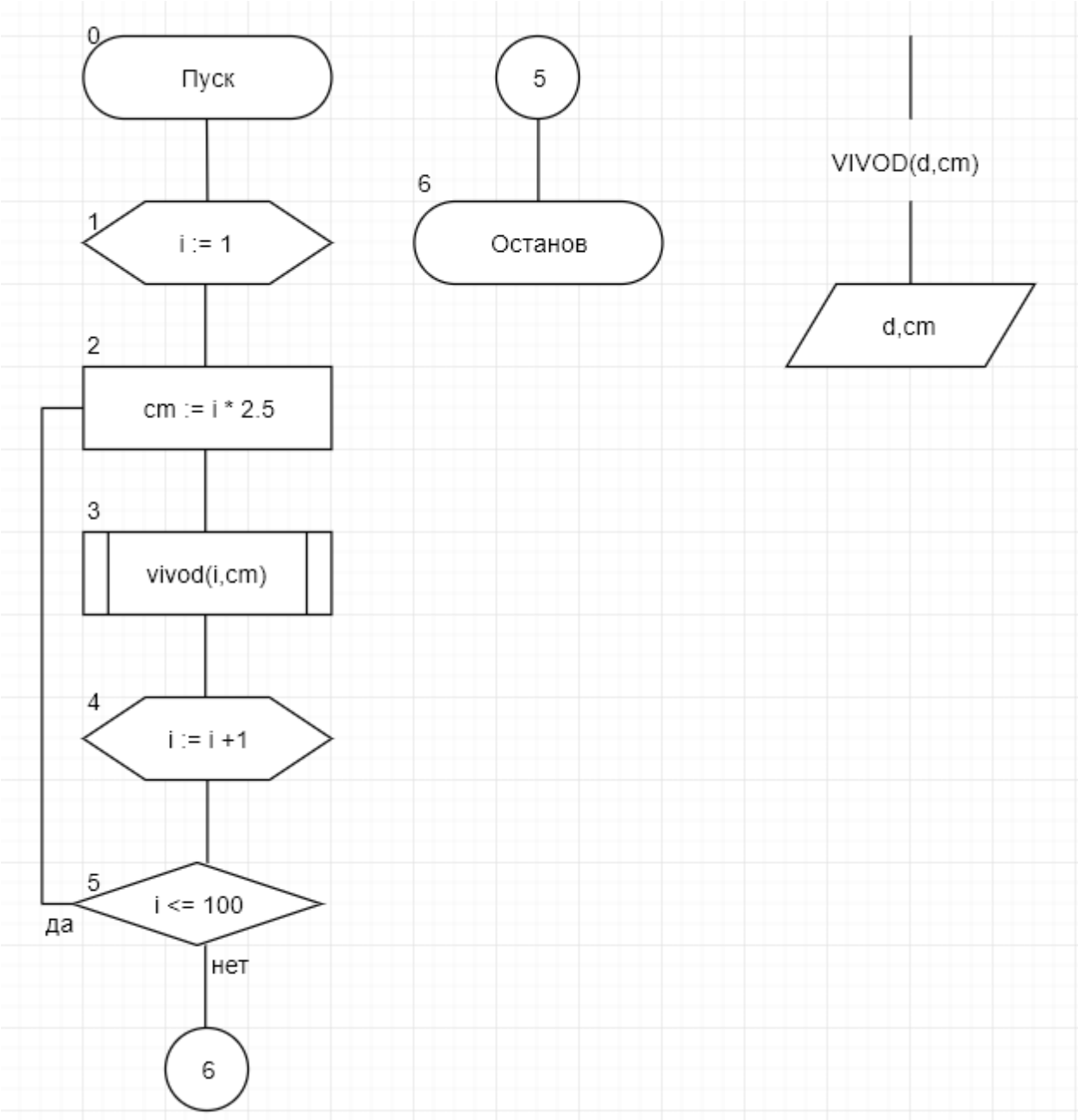
Задание 1

Постановка задачи: Перевести дюймы в сантиметры от 0 до 100 дюймов. Результаты вывести в виде таблицы. Операторы для формирования вывода таблицы оформить в виде пользовательской процедуры.

Математическая модель:

1 дюйм=2.5см

Блок-схема:



Список идентификаторов:

Название	Тип	Функция
i	integer	Управление циклом
cm	real	Хранение итогового значения
d	integer	Работа с процедурой, присваивается значение i

Код программы:

```
Program zadaniel;
Var
i : integer;
  d,cm : real;
Procedure vivod(d,cm :real);
begin
writeln('   ',d,'    ', cm);
end;
begin
 writeln('Дюймы' ,' | ','Сантиметры');
 for i := 0 to 100 do
 Begin
  cm := i * 2.5;
  vivod(i, cm);
 End;
end.
```

Результаты вычислений:

Окно вывода

```
Дюймы | Сантиметры
   0     0
   1     2.5
   2     5
   3     7.5
   4     10
   5     12.5
   6     15
   7     17.5
   8     20
   9     22.5
```

Анализ результатов вычисления: Алгоритм, основанный на детерминированном циклическом процессе, позволяет максимально быстро и просто решить данную задачу. Все вычисления происходят в цикле, управляемым переменной «i» типа «integer», которая в свою очередь изменяется по рекуррентной формуле (i = i + 1), а еще является шкалой дюймов. Переменная «cm» типа «real» содержит в себе значение в сантиметрах. Для более читабельного вида кода, разумнее использовать процедуру для вывода на экран. На процедуру передаются два значения в дюймах и сантиметрах (переменные «d» и «cm» типа «real») и выводятся на экран.

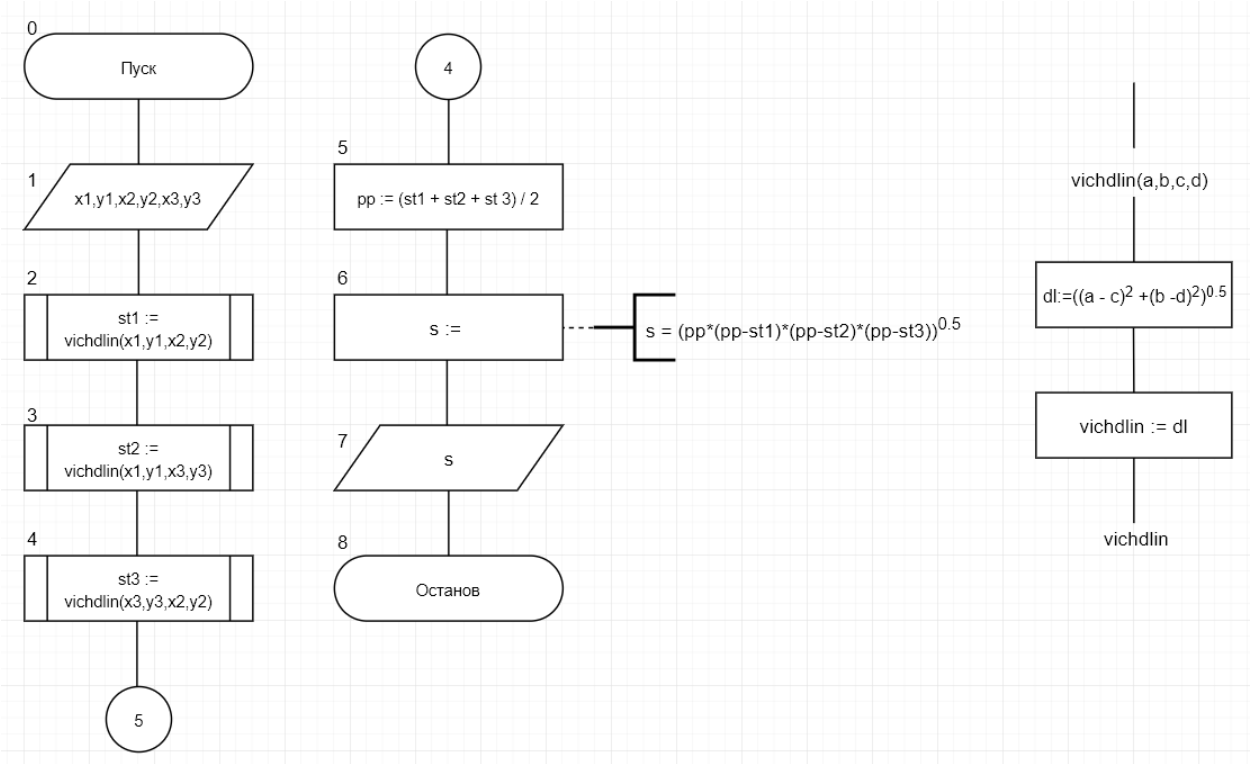
Задание 2

Постановка задачи: Найти площадь треугольника, заданного координатами своих вершин, определив функцию для расчета длины отрезка по координатам его вершин

Математическая модель:

$$\boldsymbol{d = \sqrt{(x1 - x2)^2 + (y1 - y2)^2}}$$

Блок-схема:



Список идентификаторов:

Название	Тип	Функция
x1	real	Хранение значения x координаты 1 вершины
x2	real	Хранение значения x координаты 2 вершины
x3	real	Хранение значения x координаты 3 вершины
y1	real	Хранение значения y координаты 1 вершины
y2	real	Хранение значения y координаты 2 вершины
y3	real	Хранение значения y координаты 3 вершины
st1	real	Хранение значения длины 1 стороны
st2	real	Хранение значения длины 2 стороны
st3	real	Хранение значения длины 3 стороны
pp	real	Хранение значения полупериметра
s	real	Хранение значения площади треугольника

a	real	Переменная передающая функции значения x координаты 1 вершины
b	real	Переменная передающая функции значения x координаты 2 вершины
c	real	Переменная передающая функции значения y координаты 1 вершины
d	real	Переменная передающая функции значения y координаты 2 вершины
dl	real	Хранение значения длины стороны

Код программы:

```
Program zadanie2;
Var
s,pp, x1,x2,x3,y1,y2,y3,st1,st2,st3 : real;
Function vichdlin(a,b,c,d :real) : real;
Var
  dl: real;
begin
  dl := sqrt((a - c) * (a - c) + (b - d) *(b - d));
  vichdlin := dl;
end;
begin
  Writeln('Введите координаты первой точки');
  Readln(x1,y1);
  Writeln('Введите координаты второй точки');
  Readln(x2,y2);
  Writeln('Введите координаты третьей точки');
  Readln(x3,y3);
  st1 := vichdlin(x1,y1,x2,y2);
  st2 := vichdlin(x1,y1,x3,y3);
  st3 := vichdlin(x3,y3,x2,y2);
  pp := (st1 + st2 + st3) /2;
  s := sqrt(pp * (pp - st1) * (pp - st2) * (pp - st3));
  writeln('s = ', s);
end.
```

Результаты вычислений:

Окно вывода

```
Введите координаты первой точки
0
0
Введите координаты второй точки
3
0
Введите координаты третьей точки
0
2
s = 3
```

Анализ результатов вычисления: Данная задача требует однотипного, но при этом достаточно громоздкого вычисления. Потому, в целях рационализации мы вводим пользовательскую функцию «vichdlin» которая вычисляет значения длины отрезка и возвращает значения типа «real». Самое первое действие выполняемое основной программой, это считывание с клавиатуры координат вершин, которые заносятся в переменные «x1», «x2», «x3», «y1», «y2», «y3» типа «real». Их ввод происходит по всем правилам математики (в начале x, а потом y). Сама функция работает с переменными «a», «b», «c», «d» типа «real» которым присваиваются соответствующие значения координат концов отрезка. Также в целях оптимизации вычислений, понадобилось ввести промежуточную переменную «pp» типа «real», которая используется 4 раза в вычислении итогового значения. Само же итоговое значение хранится в переменной «s» типа «real»

Задание 3

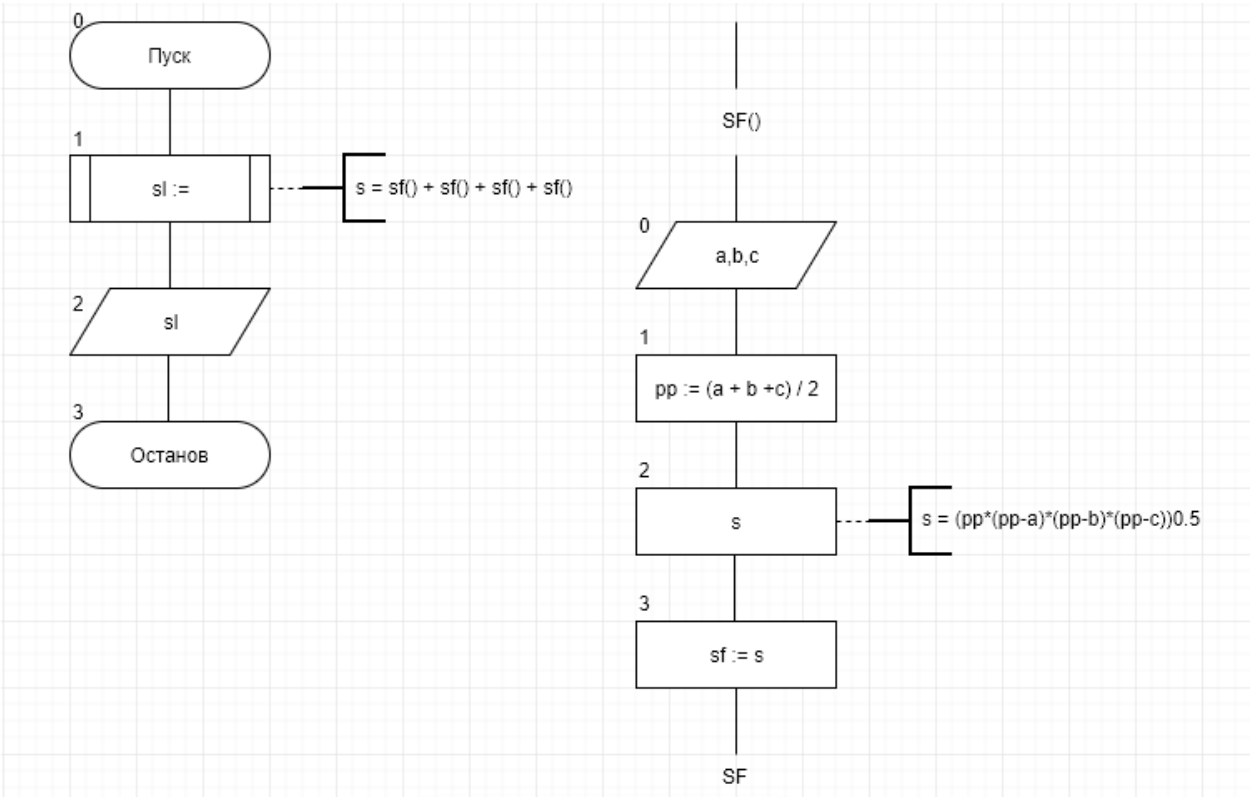
Постановка задачи: Вычислить площадь фигуры, заданной сторонами. Фигура не является прямоугольником, а треугольники, которые ее составляют, не являются прямоугольными.

Математическая модель:

$$pp = \frac{a + b + c}{2}$$

$$s = \sqrt{pp \cdot (pp - a) \cdot (pp - b) \cdot (pp - c)}$$

Блок-схема:



Список идентификаторов:

Название	Тип	Функция
sl	real	Хранение значения площади итоговой фигуры
a	real	Хранение значения длины первой стороны
b	real	Хранение значения длины второй стороны
c	real	Хранение значения длины третьей стороны
pp	real	Хранения значения полупериметра
s	real	Хранения значения площади треугольника

Код программы:

```
Program zadanie3;
  Var
    sl : real;
  Function sf() : real;
    Var s,pp,a,b,c: real;
    begin
    Writeln('Введите длину первой стороны');
    readln(a);
    Writeln('Введите длину второй стороны');
    readln(b);
    Writeln('Введите длину третьей стороны');
    readln(c);
    pp := (a + b + c) /2;
    s := sqrt(pp * (pp - a) * (pp - b) * (pp - c));
    sf := s;
    end;
  begin
   sl := sf() + sf() + sf() +sf();
   writeln(sl);
  end.
```

Результаты вычислений:

Окно вывода

```
Введите длину первой стороны
2
Введите длину второй стороны
3
Введите длину третьей стороны
2
Введите длину первой стороны
3
Введите длину второй стороны
4
Введите длину третьей стороны
5
Введите длину первой стороны
0.5
Введите длину второй стороны
1
Введите длину третьей стороны
0.75
Введите длину первой стороны
2.5
Введите длину второй стороны
3
Введите длину третьей стороны
4
Площадь итоговой фигуры: 11.9111691442966
```

Анализ результатов вычисления: Решение данной задачи требует повторного, достаточно сложного вычисления. В целях оптимизации, разумнее будет ввести пользовательскую функцию «sf», которая будет считать площадь каждого из четырех треугольников сразу после ввода сторон и возвращать значение «s» типа «real». Функция считывает длину сторон треугольника и заносит их в переменные «a», «b» и «c» типа «real». Дальше, в целях рационализации мы вводим промежуточную переменную «pp» типа «real». Итоговое значение площади заносится в переменную «» типа «». Общая площадь фигуры вычисляется путем последовательного сложения площадей треугольников в переменной «sl» типа «real». Т.е. происходит вызов функции, которая сразу же после выполнения вычислений передает вычисленное значение в переменную «sl». Так как точность итогового значения не была указана, я не стал использовать округление, тк возможно при разных введённых значениях, я бы получал одно и тоже выходное.

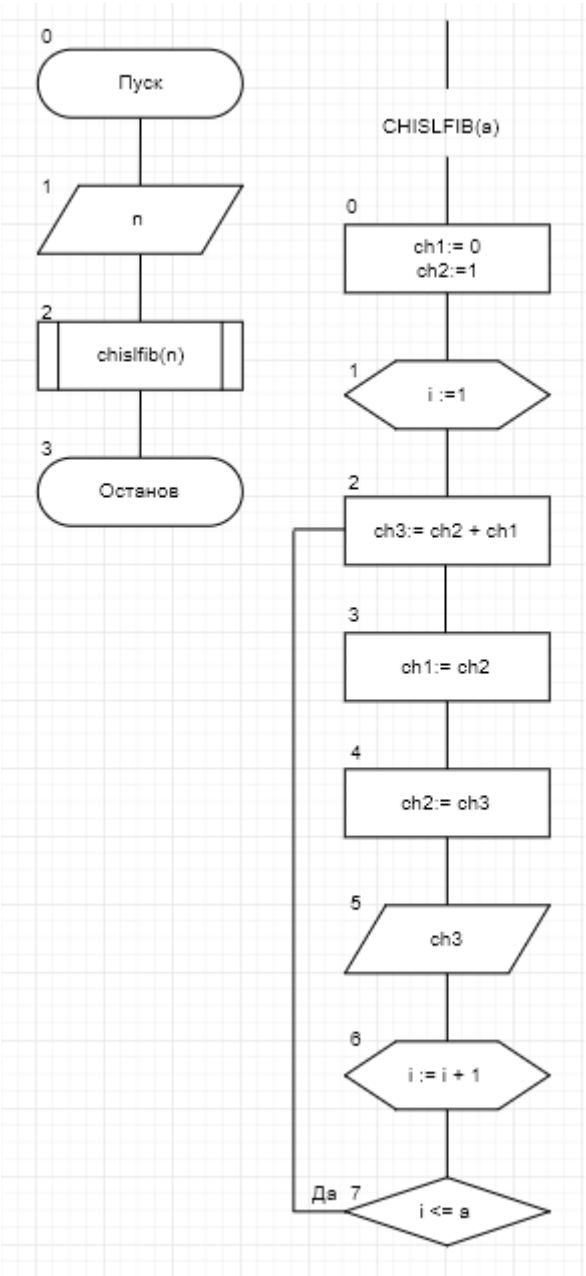
Задание 4

Постановка задачи: С клавиатуры вводится число. Вывести на экран столько элементов ряда Фибоначчи, сколько указал пользователь. Вычисление ряда организовать в функцию.

Математическая модель:

Ряд Фибоначчи = 0, 1, 1, 2, 3, 5, 8, 13…

Блок-схема:



Список идентификаторов:

Название	Тип	Функция

n	integer	Хранение значения кол-ва чисел Фибоначчи
a	integer	Переменная работающая в процедуре равная n
i	integer	Управление циклом подсчета чисел
ch1	integer	Хранение значения 1 числа Фибоначчи необходимого для подсчета
ch2	integer	Хранение значения 2 числа Фибоначчи необходимого для подсчета
ch3	integer	Считаемое число Фибоначчи

Код программы:

```
Program zadanie4;
  Var
    n : integer;
  Procedure chislfib(a :integer);
    Var
     ch1,ch2, ch3, i : integer;
    begin
      ch1 := 0;
      ch2 := 1;
      i := 1;
      while i <= a do
        Begin
          ch3 := ch2 + ch1;
          ch1 := ch2;
          ch2 := ch3;
          write(ch3,' , ');
          i := i +1;
        end;
    end;
  begin
   writeln('Введите кол-во чисел которые вы хотите увидеть:');
   readln(n);
   writeln('Числа Фибоначчи:');
   chislfib(n);
  end.
```

Результаты вычислений:

Окно вывода

```
Введите кол-во чисел которые вы хотите увидеть:
6
Числа Фибоначчи:
1 , 2 , 3 , 5 , 8 , 13 ,
```

Анализ результатов вычисления: Ряд чисел выводится, благодаря алгоритму, основанному на циклическом вычислении. В целях рационализации вычислений, введем пользовательскую процедуру «chislfib». Процедуре, а точнее переменной «a» типа «integer», передается значение переменной «n» типа «integer», которое хранит в себе число, введённое с

клавиатуры. Управление циклом осуществляется переменной «i» типа «integer», которая меняется по рекуррентной формуле (i = i + 1). Перед началом цикла, переменным «ch1» и «ch2» типа «integer» присваиваются значения первых чисел ряда Фибоначчи. Далее в цикле находится последующее число ряда, хранящиеся в переменной «ch3» типа «integer», и происходит пере присваивание переменных «ch1» и «ch2» по формулам: ch1 = ch2, ch2 = ch3. Цикл остановиться тогда, когда значение переменной «» станет больше значения введённого с клавиатуры.

Вывод: Вычислительные процессы для решения данных задач были разного типа, но в каждой из задач, вполне рационально было использовать пользовательские процедуры или функции. Также некоторые задачи имели бы менее оптимизированный вычислительный процесс без процедуры или функции. Последнюю же задачу, я бы предпочел решать немного другим способом, которые является более эффективным по времени, но более затратным по памяти. Но, к сожалению, инструментарий для решения тем способом, нам пока недоступен.