

Лабораторная работа №3

Индивидуальное задание №26

26	$\int_{0.7}^{2.1} \frac{\sqrt{1,7x^2+0,5} dx}{1,4+\sqrt{1,2x+1,3}}$	€
	2.2	

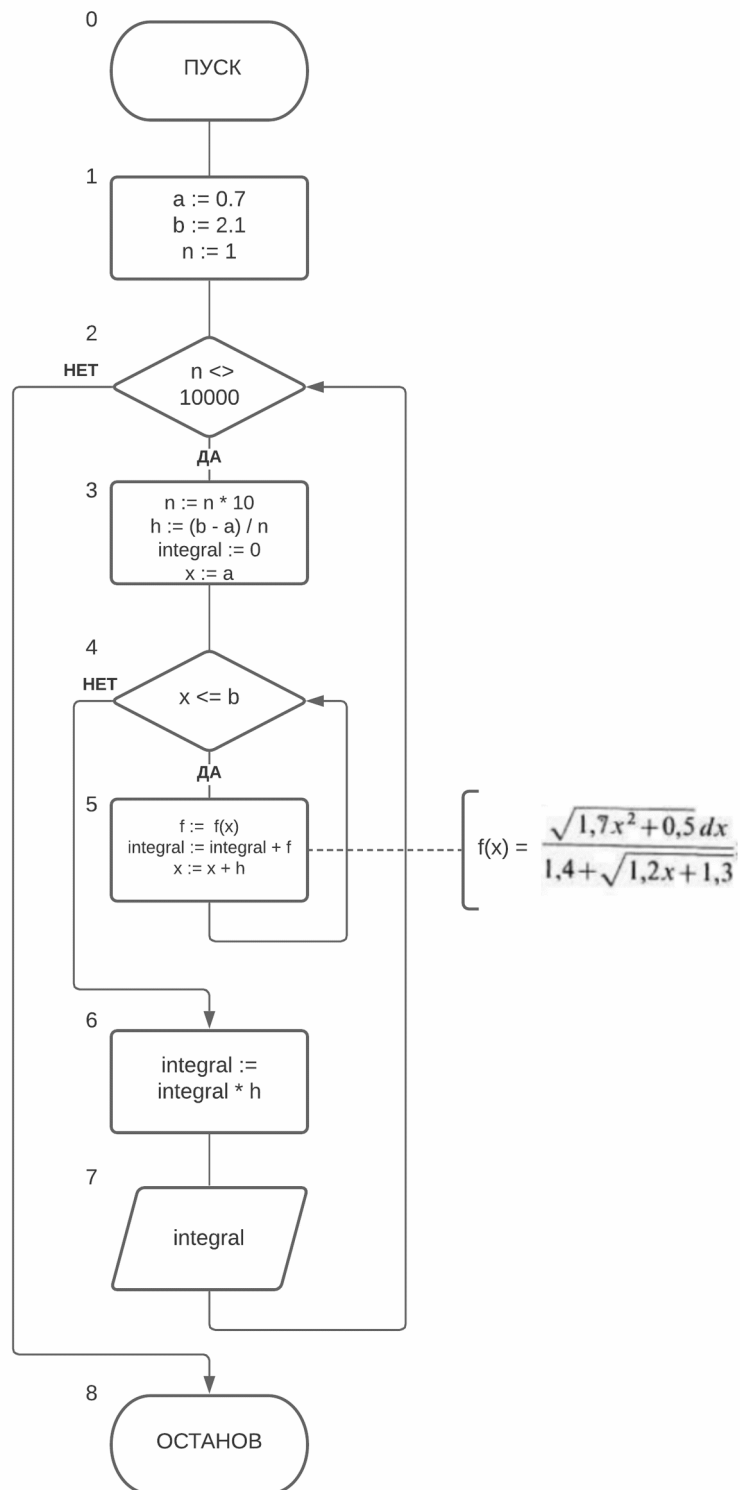
Задание 1.1

Задача: Написать программу для вычисления определенного интеграла из индивидуального задания методом прямоугольника левых частей. Протестировать программу на определенном интеграле, вычисленном в ходе выполнения самостоятельной работы 3

Математическая модель

$$S \approx \int_a^b f(x) dx \approx y_0 \cdot h + y_1 \cdot h + y_2 \cdot h + y_3 \cdot h + \dots + y_{n-1} \cdot h \approx h \cdot (y_0 + y_1 + y_2 + y_3 + \dots + y_{n-1})$$

Блок-схема



Описание переменных

Переменная	Тип	Суть
a	real	Нижний предел интегрирования
b	real	Верхний предел интегрирования
x	real	Аргумент функции
f	real	Значение подынтегральной функции
integral	real	Искомый интеграл
h	real	Шаг
n	integer	Количество разбиений

Код программы

```
main.pas
1  program LR3_1;
2  var
3      a, b, x, f, h, integral: real;
4      n: integer;
5  begin
6      a := 0.7;
7      b := 2.1;
8      n := 1;
9
10     while n <> 10000 do
11     begin
12         n := n * 10;
13         h := (b - a) / n;
14         integral := 0;
15
16         x := a;
17         while x <= b do begin
18
19             f := (sqrt(1.7 * sqr(x) + 0.5)) / (1.4 + sqrt(1.2 * x + 1.3));
20
21             integral := integral + f;
22             x := x + h;
23         end;
24
25         integral := integral * h;
26         writeln('Интеграл при ', n, ' разбиений = ', integral:0:7);
27     end;
28 end.
29
```

input

Compiled Successfully. memory: 1620 time: 0 exit code: 0

```
Интеграл при 10 разбиений = 0.8440186
Интеграл при 100 разбиений = 0.8835294
Интеграл при 1000 разбиений = 0.8744975
Интеграл при 10000 разбиений = 0.8747745
```

Задача: Протестировать программу на определенном интеграле, вычисленным в ходе выполнения самостоятельной работы 3

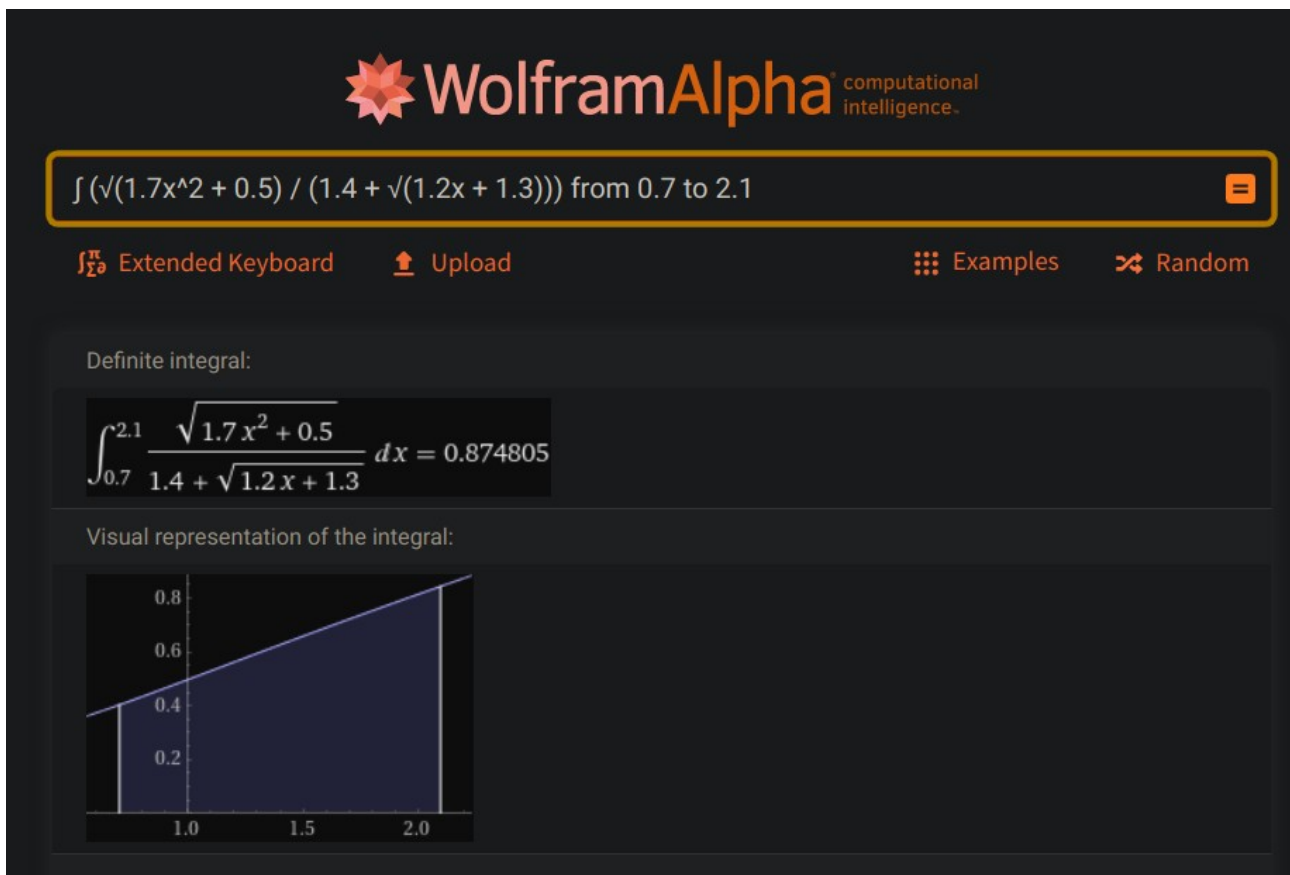
```
main.pas
1  program SR3_1;
2  var
3      a, b, x, f, h, integral: real;
4      n: integer;
5  begin
6      a := 1;
7      b := 10;
8      n := 1;
9
10     while n <> 10000 do
11     begin
12         n := n * 10;
13         h := (b - a) / n;
14         integral := 0;
15
16         x := a;
17         while x <= b do begin
18             f := sqr(x);
19             x := x + h;
20             integral := integral + f;
21         end;
22
23         integral := integral * h;
24         writeln('Интеграл при ', n, ' разбиений = ', integral:0:7);
25     end;
26 end.
```

input

Compiled Successfully. memory: 1556 time: 0 exit code: 0

```
Интеграл при 10 разбиений = 289.6650000
Интеграл при 100 разбиений = 337.5571500
Интеграл при 1000 разбиений = 332.5546215
Интеграл при 10000 разбиений = 333.0454512
```

Проверка правильности



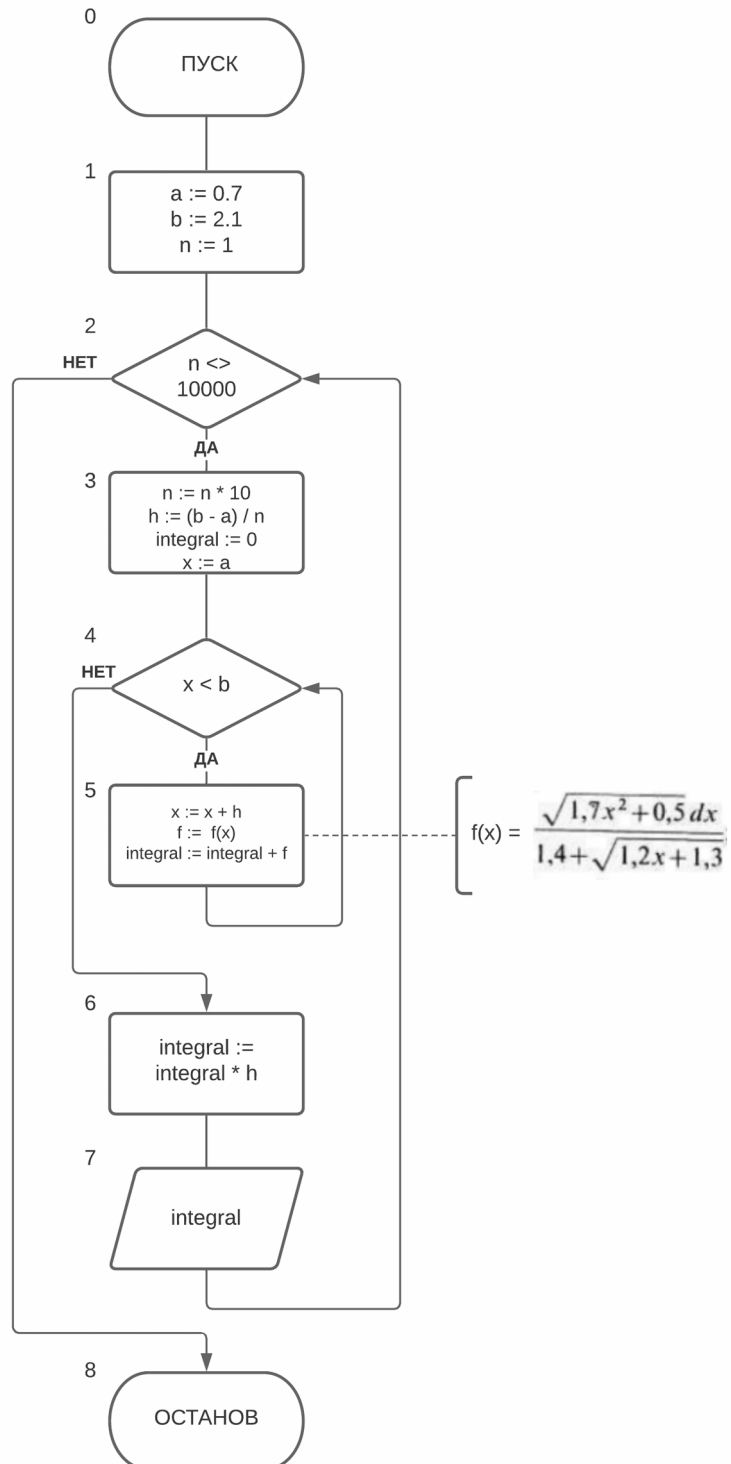
Задание 1.2

Задача: Написать программу для вычисления определенного интеграла из индивидуального задания методом прямоугольника правых частей. Протестировать программу на определенном интеграле, вычисленном в ходе выполнения самостоятельной работы 3

Математическая модель

$$S \approx \int_a^b f(x) dx \approx y_1 \cdot h + y_2 \cdot h + y_3 \cdot h + y_4 \cdot h + \dots + y_n \cdot h \approx h \cdot (y_1 + y_2 + y_3 + y_4 + \dots + y_n)$$

Блок-схема



Описание переменных

Переменная	Тип	Суть
a	real	Нижний предел интегрирования
b	real	Верхний предел интегрирования
x	real	Аргумент функции
f	real	Значение подынтегральной функции
integral	real	Искомый интеграл
h	real	Шаг
n	integer	Количество разбиений

Код программы


```
main.pas
1 program LR3_2;
2 var
3     a, b, x, f, h, integral: real;
4     n: integer;
5 begin
6     a := 0.7;
7     b := 2.1;
8     n := 1;
9
10    while n <> 10000 do
11    begin
12        n := n * 10;
13        h := (b - a) / n;
14        integral := 0;
15
16        x := a;
17        while x < b do begin
18            x := x + h;
19            f := (sqrt(1.7 * sqr(x) + 0.5)) / (1.4 + sqrt(1.2 * x + 1.3));
20
21            integral := integral + f;
22        end;
23
24        integral := integral * h;
25        writeln('Интеграл при ', n, ' разбиений = ', integral:0:7);
26    end;
27 end.
28
```

input

Compiled Successfully. memory: 1628 time: 0 exit code: 0

```
Интеграл при 10 разбиений = 0.9055814
Интеграл при 100 разбиений = 0.8897443
Интеграл при 1000 разбиений = 0.8751131
Интеграл при 10000 разбиений = 0.8748361
```

Задача: Протестировать программу на определенном интеграле, вычисленным в ходе выполнения самостоятельной работы 3

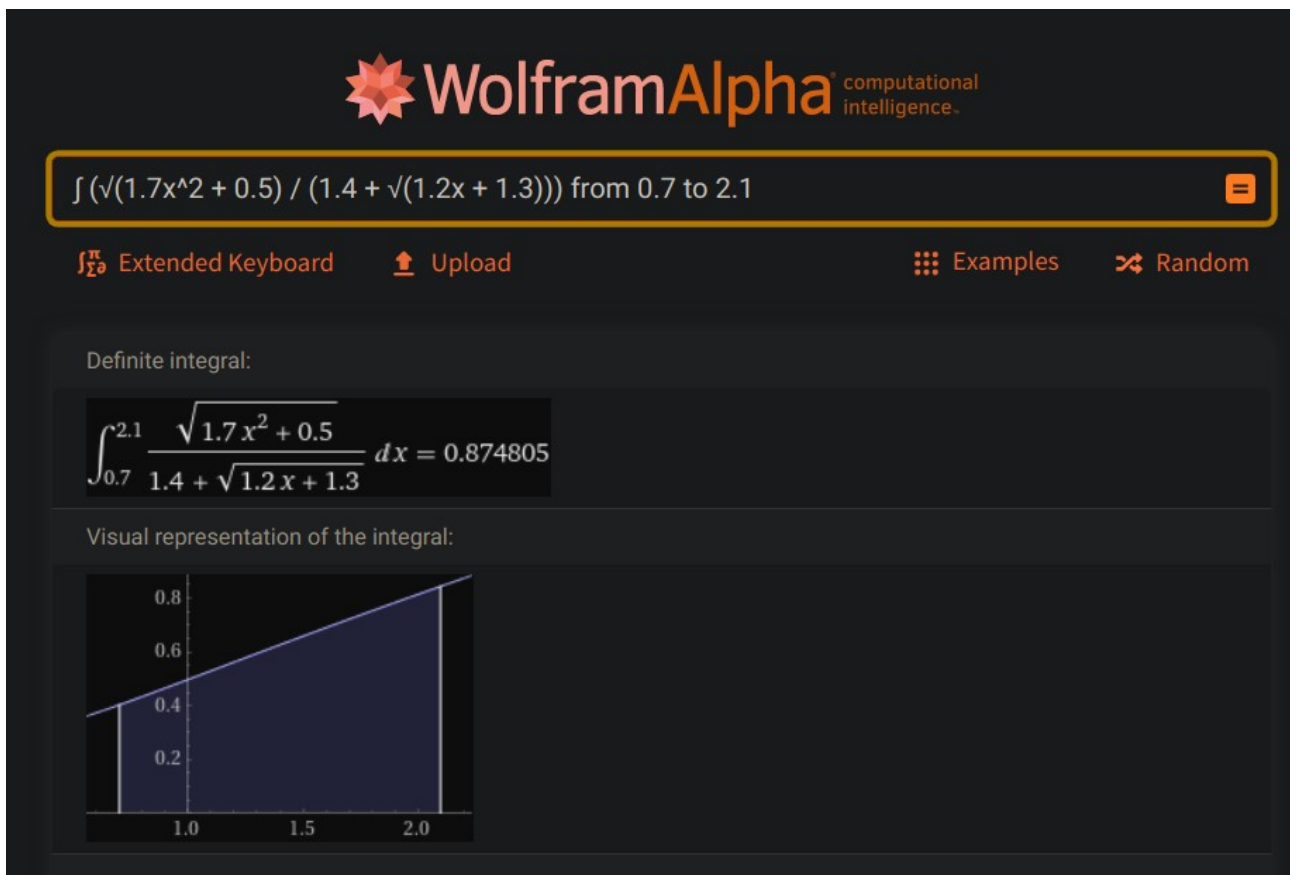
```
main.pas
1  program SR3_2;
2  var
3      a, b, x, f, h, integral: real;
4      n: integer;
5  begin
6      a := 1;
7      b := 10;
8      n := 1;
9
10     while n <> 10000 do
11     begin
12         n := n * 10;
13         h := (b - a) / n;
14         integral := 0;
15
16         x := a;
17         while x < b do begin
18             x := x + h;
19             f := sqr(x);
20
21             integral := integral + f;
22         end;
23
24         integral := integral * h;
25         writeln('Интеграл при ', n, ' разбиений = ', integral:0:7);
26     end;
27 end.
```

input

Compiled Successfully. memory: 1580 time: 0 exit code: 0

```
Интеграл при 10 разбиений = 378.7650000
Интеграл при 100 разбиений = 346.6298790
Интеграл при 1000 разбиений = 333.4456215
Интеграл при 10000 разбиений = 333.1345674
```

Проверка правильности



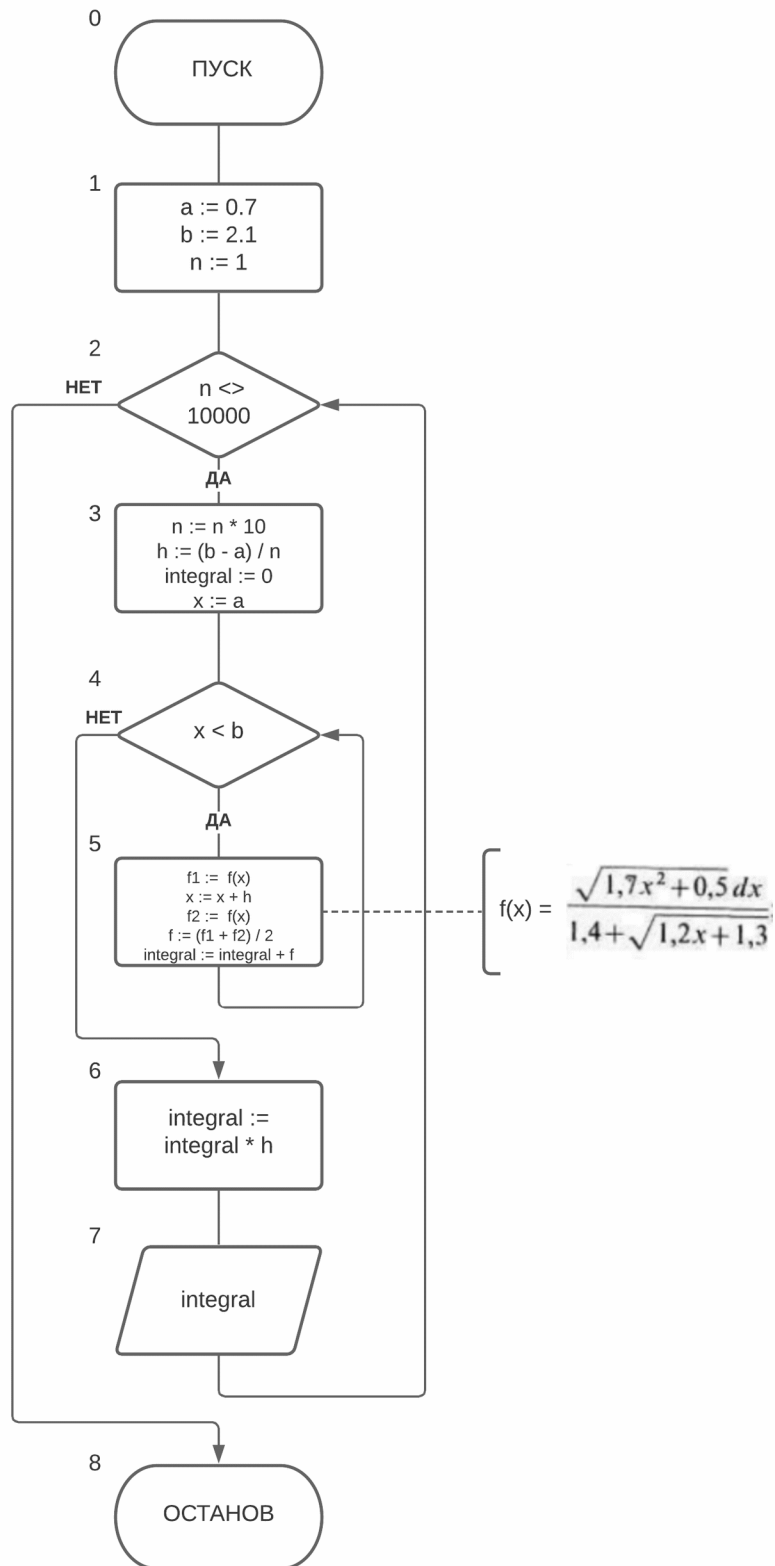
Задание 2.1

Задача: Написать программу для вычисления определенного интеграла из индивидуального задания методом трапеций. Протестировать программу на определенном интеграле, вычисленным в ходе выполнения самостоятельной работы 3

Математическая модель

$$S \approx \int_a^b f(x) dx \approx h \cdot \left(\frac{y_0 + y_n}{2} + y_1 + y_2 + \dots + y_{n-1} \right)$$

Блок-схема



Описание переменных

Переменная	Тип	Суть
a	real	Нижний предел интегрирования
b	real	Верхний предел интегрирования
x	real	Аргумент функции
f	real	Среднее значение подынтегральной функции
f1	real	Значение подынтегральной функции в точке X
f2	real	Значение подынтегральной функции в точке X + h
integral	real	Искомый интеграл
h	real	Шаг
n	integer	Количество разбиений

Код программы

main.pas

```
1 program LR3_3;
2 var
3     a, b, x, f, f1, f2, h, integral: real;
4     n: integer;
5 begin
6     a := 0.7;
7     b := 2.1;
8     n := 1;
9
10    while n <> 10000 do
11        begin
12            n := n * 10;
13            h := (b - a) / n;
14            integral := 0;
15
16            x := a;
17            while x < b do begin
18                f1 := (sqrt(1.7 * sqr(x) + 0.5)) / (1.4 + sqrt(1.2 * x + 1.3));
19                x := x + h;
20                f2 := (sqrt(1.7 * sqr(x) + 0.5)) / (1.4 + sqrt(1.2 * x + 1.3));
21                f := (f1 + f2) / 2;
22                integral := integral + f;
23            end;
24
25            integral := integral * h;
26            writeln('Интеграл при ', n, ' разбиений = ', integral:0:7);
27        end;
28    end.
```

input

Compiled Successfully. memory: 1508 time: 0 exit code: 0

```
Интеграл при 10 разбиений = 0.8748000
Интеграл при 100 разбиений = 0.8866368
Интеграл при 1000 разбиений = 0.8748053
Интеграл при 10000 разбиений = 0.8748053
```

Задача: Протестировать программу на определенном интеграле, вычисленным в ходе выполнения самостоятельной работы 3

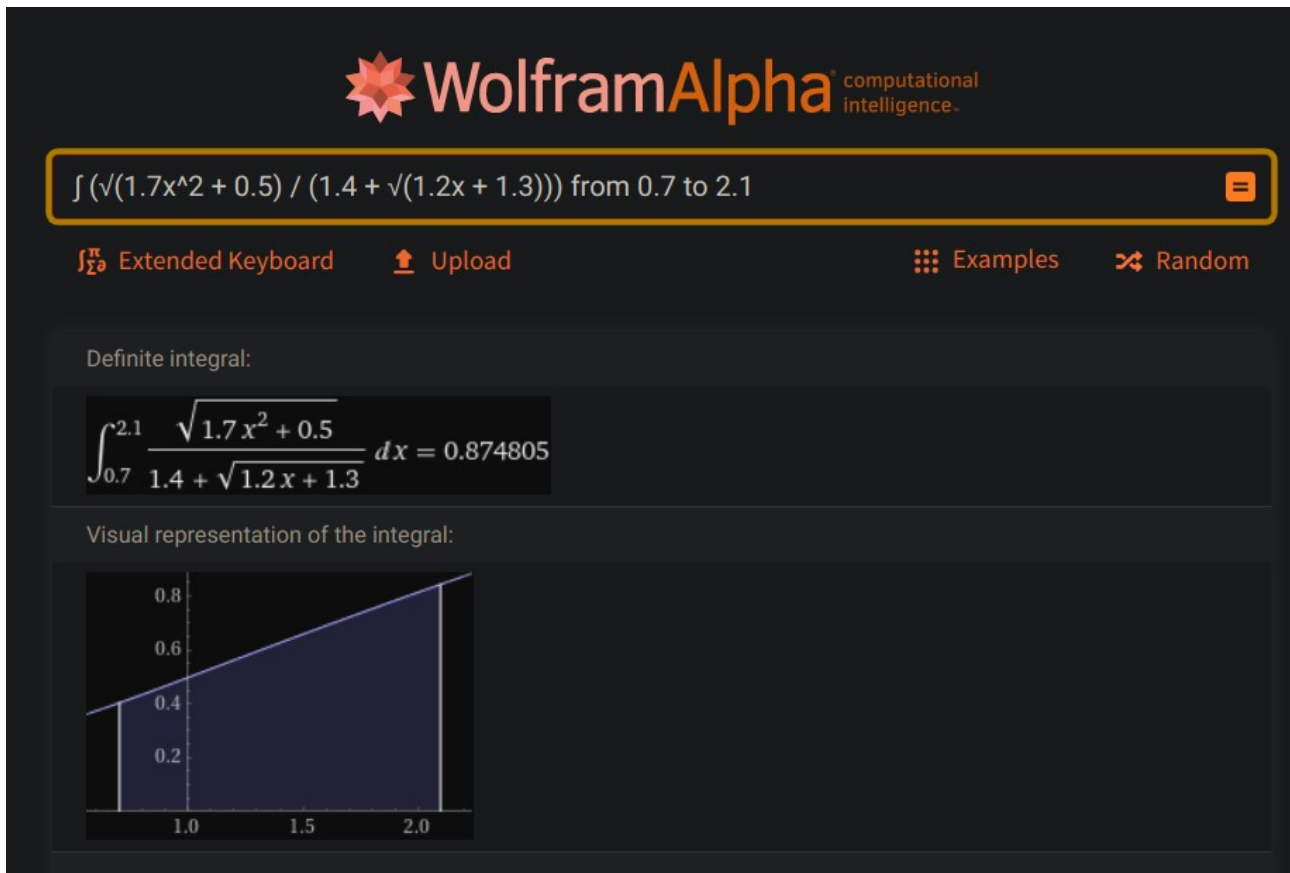
```
main.pas
1  program SR3_3;
2  var
3      a, b, x, f, f1, f2, h, integral: real;
4      n: integer;
5  begin
6      a := 1;
7      b := 10;
8      n := 1;
9
10     while n <> 10000 do
11     begin
12         n := n * 10;
13         h := (b - a) / n;
14         integral := 0;
15
16         x := a;
17         while x < b do begin
18             f1 := sqr(x);
19             x := x + h;
20             f2 := sqr(x);
21             f := (f1 + f2) / 2;
22             integral := integral + f;
23         end;
24
25         integral := integral * h;
26         writeln('Интеграл при ', n, ' разбиений = ', integral:0:7);
27     end;
28 end.
```

input

Compiled Successfully. memory: 1504 time: 0 exit code: 0

```
Интеграл при 10 разбиений = 334.2150000
Интеграл при 100 разбиений = 342.0935145
Интеграл при 1000 разбиений = 333.0001215
Интеграл при 10000 разбиений = 333.0900093
```

Проверка правильности



Задание 2.2

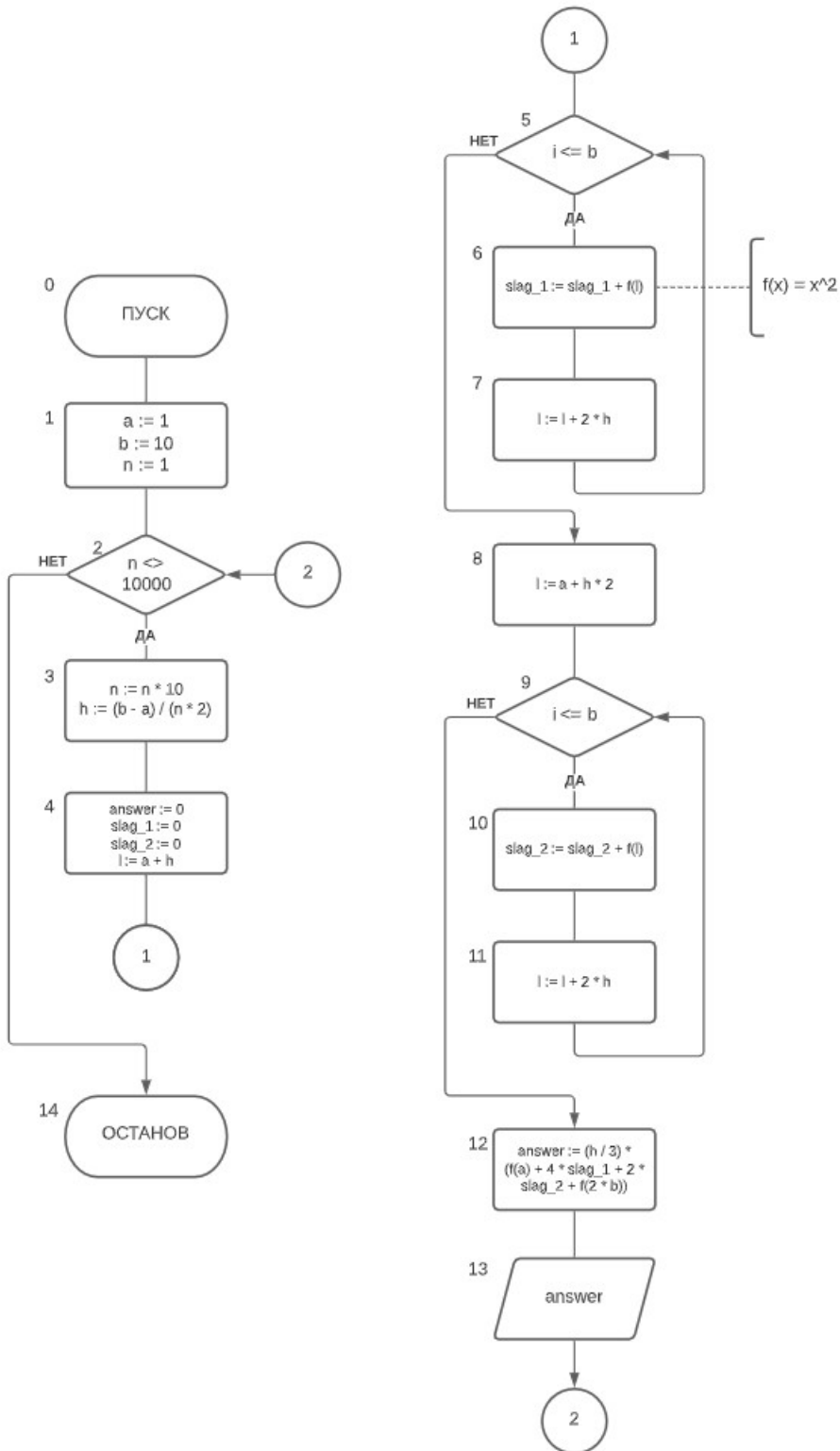
Задача: Написать программу для вычисления определенного интеграла из индивидуального задания методом парабол. Протестировать программу на определенном интеграле, вычисленным в ходе выполнения самостоятельной работы 3

Математическая модель

Формула метода Симпсона имеет вид

$$\int_a^b f(x) dx \approx \frac{h}{3} \left(f(x_0) + 4 \sum_{i=1}^n f(x_{2i-1}) + 2 \sum_{i=1}^{n-1} f(x_{2i}) + f(x_{2n}) \right).$$

Блок-схема



Описание переменных

Переменная	Тип	Суть
a	real	Нижняя граница интеграла
b	real	Верхняя граница интеграла
n	integer	Количество разбиений
h	real	Шаг итерации
i	real	Счетчик цикла
answer	real	Значение интеграла
slag_1	real	Слагаемое 1 в скобках
slag_2	real	Слагаемое 2 в скобках
nz	real	Значение ф-ии в нижней границе интегрирования
kz	real	Значение ф-ии в верхней границе интегрирования
f	real	Значение подынтегральной функции

Код программы

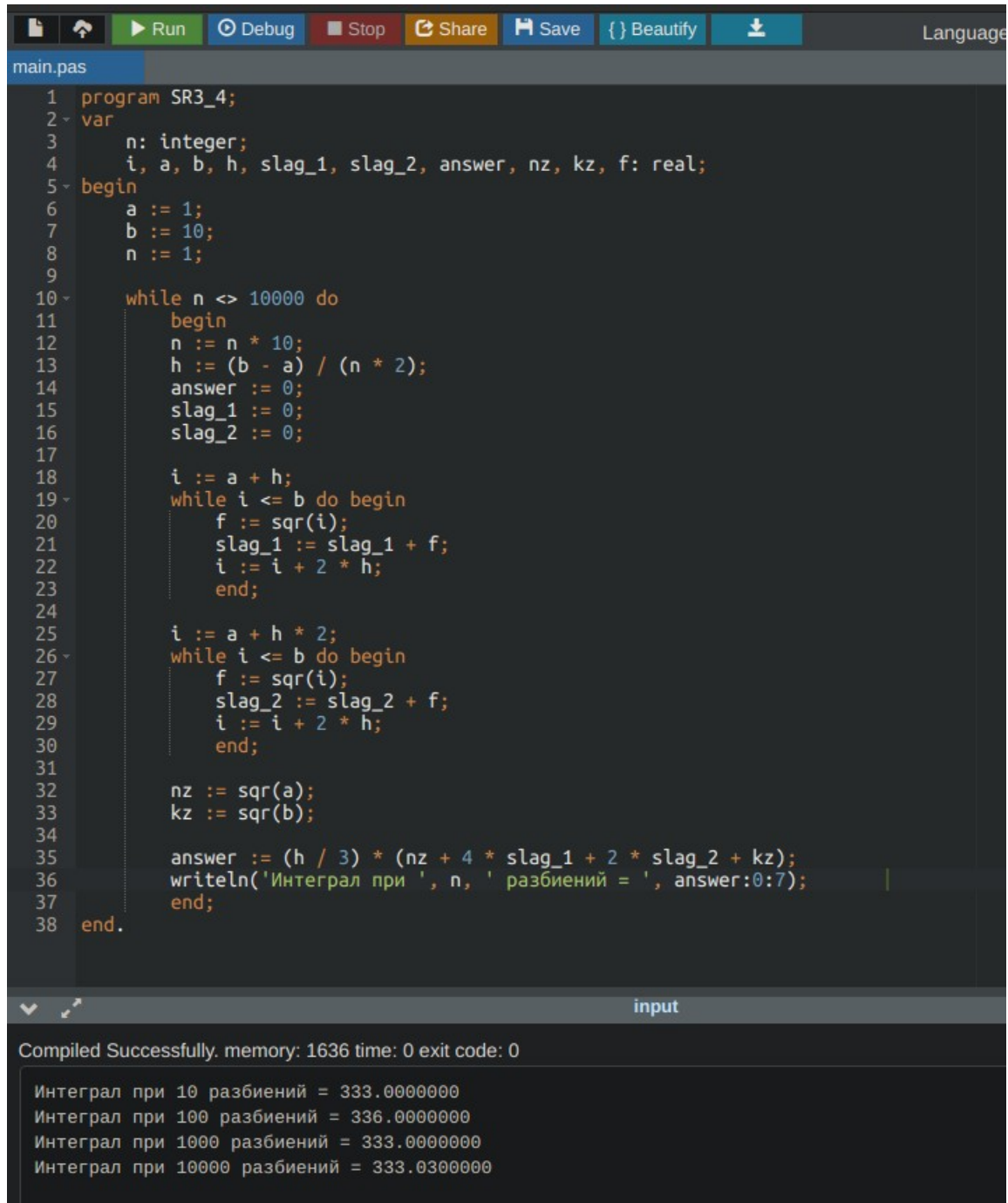
```
main.pas
1 program LR3_Z4;
2 var
3     n: integer;
4     i, a, b, h, slag_1, slag_2, answer, nz, kz, f: real;
5 begin
6     a := 0.7;
7     b := 2.1;
8     n := 1;
9
10    while n <> 10000 do
11        begin
12            n := n * 10;
13            h := (b - a) / (n * 2);
14            answer := 0;
15            slag_1 := 0;
16            slag_2 := 0;
17
18            i := a + h;
19            while i <= b do begin
20                f := (sqrt(1.7 * sqr(i) + 0.5)) / (1.4 + sqrt(1.2 * i + 1.3));
21                slag_1 := slag_1 + f;
22                i := i + 2 * h;
23            end;
24
25            i := a + h * 2;
26            while i <= b do begin
27                f := (sqrt(1.7 * sqr(i) + 0.5)) / (1.4 + sqrt(1.2 * i + 1.3));
28                slag_2 := slag_2 + f;
29                i := i + 2 * h;
30            end;
31
32            nz := (sqrt(1.7 * sqr(a) + 0.5)) / (1.4 + sqrt(1.2 * a + 1.3));
33            kz := (sqrt(1.7 * sqr(b) + 0.5)) / (1.4 + sqrt(1.2 * b + 1.3));
34
35            answer := (h / 3) * (nz + 4 * slag_1 + 2 * slag_2 + kz);
36            writeln('Интеграл при ', n, ' разбиений = ', answer:0:7);
37        end;
38    end.
```

input

Compiled Successfully. memory: 1508 time: 0 exit code: 0

```
Интеграл при 10 разбиений = 0.8748054
Интеграл при 100 разбиений = 0.8787394
Интеграл при 1000 разбиений = 0.8748053
Интеграл при 10000 разбиений = 0.8748053
```

Задача: Протестировать программу на определенном интеграле, вычисленным в ходе выполнения самостоятельной работы 3



```
1 program SR3_4;
2 var
3     n: integer;
4     i, a, b, h, slag_1, slag_2, answer, nz, kz, f: real;
5 begin
6     a := 1;
7     b := 10;
8     n := 1;
9
10    while n <> 10000 do
11    begin
12        n := n * 10;
13        h := (b - a) / (n * 2);
14        answer := 0;
15        slag_1 := 0;
16        slag_2 := 0;
17
18        i := a + h;
19        while i <= b do begin
20            f := sqr(i);
21            slag_1 := slag_1 + f;
22            i := i + 2 * h;
23        end;
24
25        i := a + h * 2;
26        while i <= b do begin
27            f := sqr(i);
28            slag_2 := slag_2 + f;
29            i := i + 2 * h;
30        end;
31
32        nz := sqr(a);
33        kz := sqr(b);
34
35        answer := (h / 3) * (nz + 4 * slag_1 + 2 * slag_2 + kz);
36        writeln('Интеграл при ', n, ' разбиений = ', answer:0:7);
37    end;
38 end.
```

input

Compiled Successfully. memory: 1636 time: 0 exit code: 0

Интеграл при 10 разбиений = 333.0000000
Интеграл при 100 разбиений = 336.0000000
Интеграл при 1000 разбиений = 333.0000000
Интеграл при 10000 разбиений = 333.0300000

Проверка правильности

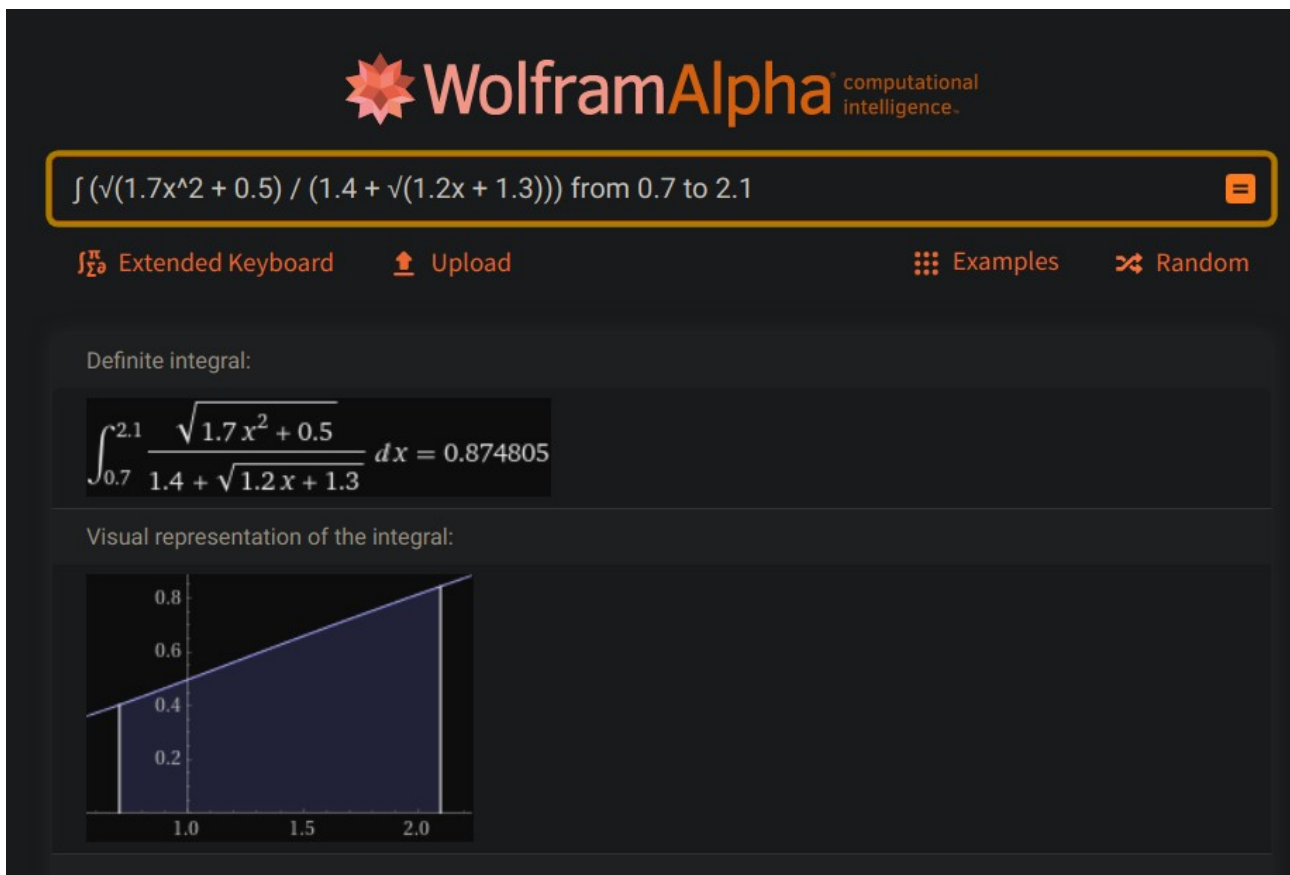


Таблица с результатами вычислений

Сравнение результатов вычислений

N Количество разбиений	Н шаг	I Метод левых частей прямоугольн иков	I Метод правых частей прямоугольн иков	I Метод трапеций	I Метод парабол
10	0,14	0,8440186	0,9055814	0,8748	0,8748054
100	0,014	0,8835294	0,8897443	0,8866368	0,8787394
1000	0,0014	0,8744975	0,8751131	0,8748053	0,8748053
10000	0,00014	0,8747745	0,8748361	0,8748053	0,8748053

Задание 3.1: Вывод

Все рассмотренные методы, кроме метода парабол оказались достаточно точными только при большом количестве разбиений. Метод Симпсона же имеет достаточно низкую погрешность ($<0,01$) даже при 10 разбиениях. Это позволяет назвать его наиболее точным.

Задание 3.2: Вывод

Точность всех представленных методов напрямую зависит от количества разбиений, следовательно, точность каждого из них можно понизить повышением количества разбиений.