**Лабораторная работа № 3**

**Детерминированные вычислительные процессы с управлением по аргументу.**

**Численное интегрирование.**

**Общее:**

1. **Тема: детерминированные циклические вычислительные процессы с управлением по аргументу. Численное интегрирование.**
2. **Цель: научиться реализовывать алгоритмы численного интегрирования посредством детерминированных циклических вычислительных процессов с управлением по аргументу с помощью Free Pascal.**
3. **Используемое оборудование: ПК, среда программирования Lazarus.**

Далее к каждому заданию будет начинаться с пункта №4(постановка задачи).

**Задание 1.**

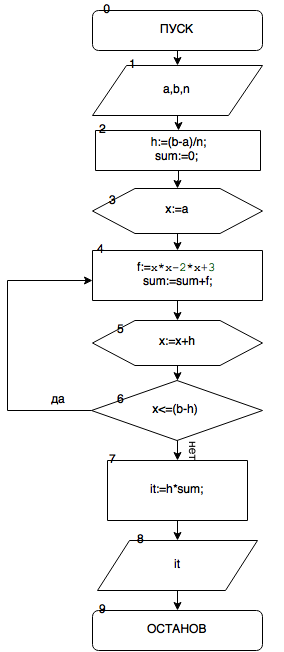
**Постановка задачи:** Написать программу для вычисления определенного интеграла методом прямоугольника левых частей. Протестировать программу на определенном интеграле, вычисленном в ходе выполнения самостоятельной работы 3. Вычислить определенный интеграл из индивидуального задания

**Математическая модель:**

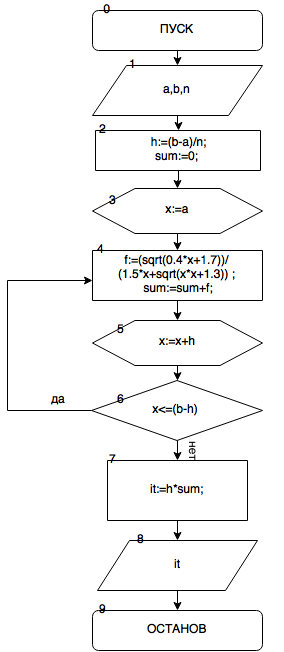
C:\Users\iuytr\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Мат мод.png

Блок-схема на след. странице.

**Блок-схема для подпункта 1(Протестировать программу на определенном интеграле, вычисленном в ходе выполнения самостоятельной работы 3):**



**Блок-схема для подпункта 2(Вычислить определенный интеграл из индивидуального задания):**



**Список идентификаторов:**

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Смысл |
| a | real | Пределы интегрирования |
| b | real | Пределы интегрирования |
| h | real | Шаг |
| it | real | Итог |
| n | real | Кол-во отрезков разбиения |
| sum | real | Сумма |
| x | real | Параметр цикла |
| f | real | Функция |

**Код программы(подпункт 1):**

**program** levye;

**var**

a,b,f,sum,it,x,n,h:real;

**begin**

Writeln('Vvedite a');

Readln(a);

Writeln('Vvedite b');

Readln(b);

Writeln('Vvedite n');

Readln(n);

h:=(b-a)/n;

sum:=0;

x:=a;

**while x** <= (b-h)

**do**

**begin**

f:=x\*x-2\*x+3;

sum:=sum+f;

x:=x+h;

**end**;

it:=h\*sum;

Writeln('Integral raven',' ',it:2:4);

readln();

**end**.

**Код программы(подпункт 2):**

**program** levye;

**var**

a,b,f,sum,it,x,n,h:real;

**begin**

Writeln('Vvedite a');

Readln(a);

Writeln('Vvedite b');

Readln(b);

Writeln('Vvedite n');

Readln(n);

h:=(b-a)/n;

sum:=0;

x:=a;

**while x** <= (b-h)

**do**

**begin**

f:=(sqrt(0.4\*x+1.7))/(1.5\*x+sqrt(x\*x+1.3));

sum:=sum+f;

x:=x+h;

**end**;

it:=h\*sum;

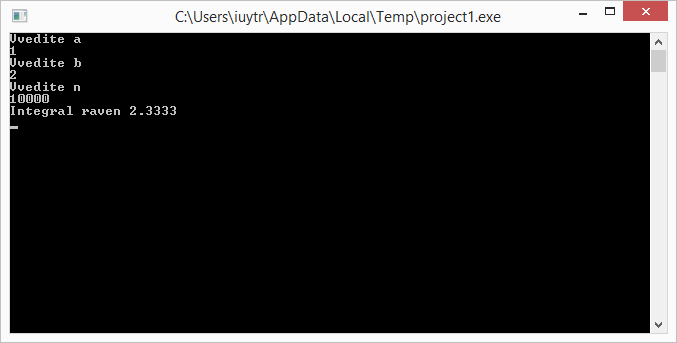
Writeln('Integral raven',' ',it:2:4);

readln();

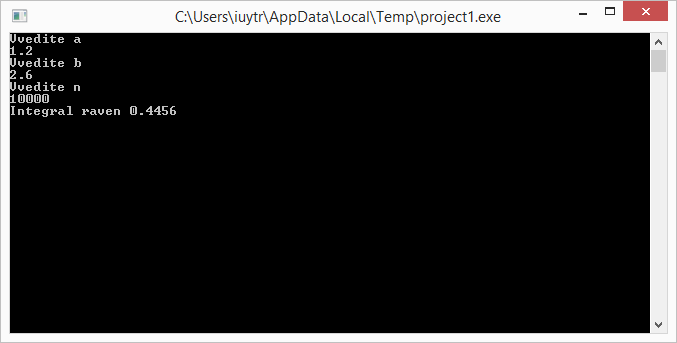
**end**.

**Результат работы программы:**

**Подпункт 1:**

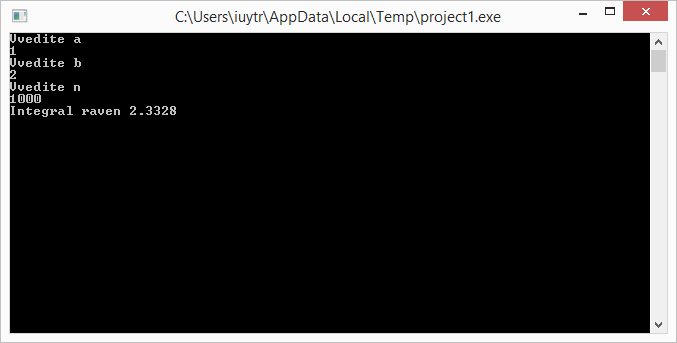


**Подпункт 2:**



**Анализ:** мы написали программу, которая вычисляет определенный интеграл методом прямоугольников левых частей. Сначала мы вводим левую границу(a), потом границу b, потом количество отрезков разбиения, далее программа считает шаг h= (b-a)/n и начинает вычислять интеграл по заданному алгоритму. Примечание: чем больше количество отрезков разбиения, тем выше точность ответа. Например, для подпункта 1 изначально в самостоятельной работе №3 мною был получен ответ , что приблизительно равно 2,3333.

И если мы возьмем количество отрезков разбиения 10 000, то действительно получим этот результат, однако если мы возьмем, например, 1000, то получим вот это:



Это не совсем точный результат. Таким образом, количество отрезков разбиения действительно влияет на ответ.

**Вывод:** мы написали программу, которая вычисляет определенный интеграл методом прямоугольников левых частей. Чем больше количество отрезков разбиения, тем точнее результат.

**Задание 2.**

**Постановка задачи:** 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. Вычислить определенный интеграл из индивидуального задания

**Математическая модель:**

C:\Users\iuytr\AppData\Local\Microsoft\Windows\INetCache\Content.Word\Мат мод 2.png

**Блок-схема:**

**1 подпункт:**

**2 подпункт:**

**Список идентификаторов:**

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Смысл |
| a | real | Пределы интегрирования |
| b | real | Пределы интегрирования |
| h | real | Шаг |
| it | real | Итог |
| n | real | Кол-во отрезков разбиения |
| sum | real | Сумма |
| x | real | Параметр цикла |
| f | real | Функция |

**Код программы:**

**1 подпункт:**

**program** pravye;

**var**

a,b,f,sum,it,x,n,h:real;

**begin**

Writeln('Vvedite a');

Readln(a);

Writeln('Vvedite b');

Readln(b);

Writeln('Vvedite n');

Readln(n);

h:=(b-a)/n;

sum:=0;

x:=a+h;

**while x**<=b

**do**

**begin**

f:=x\*x-2\*x+3;

sum:=sum+f;

x:=x+h;

**end**;

it:=h\*sum;

Writeln('Integral raven',' ',it:2:4);

readln();

**end**.

**2 подпункт:**

**program** pravye;

**var**

a,b,f,sum,it,x,n,h:real;

**begin**

Writeln('Vvedite a');

Readln(a);

Writeln('Vvedite b');

Readln(b);

Writeln('Vvedite n');

Readln(n);

h:=(b-a)/n;

sum:=0;

x:=a+h;

**while x**<=b

**do**

**begin**

f:=(sqrt(0.4\*x+1.7))/(1.5\*x+sqrt(x\*x+1.3));

sum:=sum+f;

x:=x+h;

**end**;

it:=h\*sum;

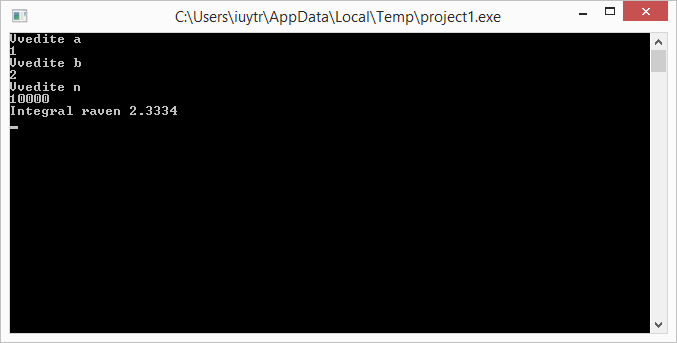
Writeln('Integral raven',' ',it:2:4);

readln();

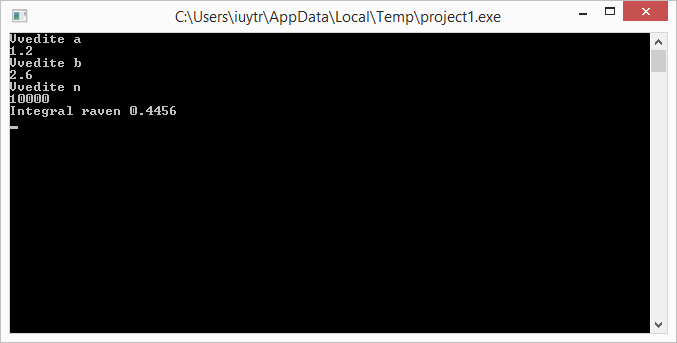
**end**.

**Результат работы программы:**

**1 подпункт:**

****

**2 подпункт:**



**Анализ:** мы написали программу, которая вычисляет определенный интеграл методом прямоугольников правых частей. Сначала мы вводим левую границу(a), потом границу b, потом количество отрезков разбиения, далее программа считает шаг h= (b-a)/n и начинает вычислять интеграл по заданному алгоритму. Примечание: чем больше количество отрезков разбиения, тем выше точность ответа.

**Вывод:** мы написали программу, которая вычисляет определенный интеграл методом прямоугольников правых частей.

**Задание 3.**

**Постановка задачи:**

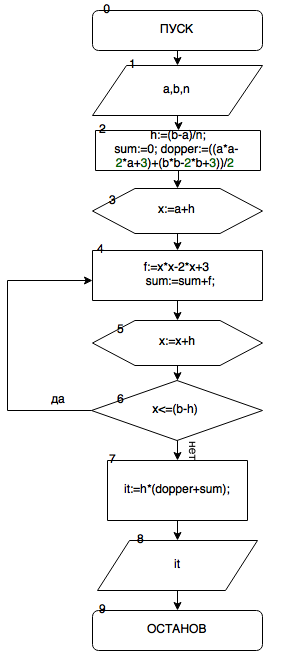
Написать программу для вычисления определенного интеграла из индивидуального задания методом трапеций. Протестировать программу на определенном интеграле, вычисленном в ходе выполнения самостоятельной работы 3. Вычислить определенный интеграл из индивидуального задания

**Математическая модель:**

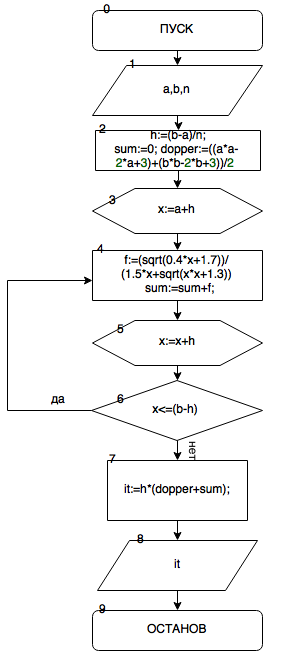
Мат мод 3

**Блок-схема:**

**1 подпункт:**



**2 подпункт:**



**Список идентификаторов:**

|  |  |  |
| --- | --- | --- |
| Имя | Тип | Смысл |
| dopper | real | Доп.переменная для удобства счета |
| a | real | Пределы интегрирования |
| b | real | Пределы интегрирования |
| h | real | Шаг |
| it | real | Итог |
| n | real | Кол-во отрезков разбиения |
| sum | real | Сумма |
| x | real | Параметр цикла |
| f | real | Функция |

**Код программы:**

**1 подпункт:**

**program** trapeziya;

**var**

a,b,f,sum,it,x,n,h,dopper:real;

**begin**

Writeln('Vvedite a');

Readln(a);

Writeln('Vvedite b');

Readln(b);

Writeln('Vvedite n');

Readln(n);

h:=(b-a)/n;

sum:=0;

dopper:=((a\*a-2\*a+3)+(b\*b-2\*b+3))/2 ;

x:=a+h;

**while x** < (b-h)

**do**

**begin**

f:=x\*x-2\*x+3;

sum:=sum+f;

x:=x+h;

**end**;

it:=h\*(dopper+sum);

Writeln('Integral raven',' ',it:2:4);

readln();

**end**.

**2 подпункт:**

**program** trapeziya;

**var**

a,b,f,sum,it,x,n,h,dopper:real;

**begin**

Writeln('Vvedite a');

Readln(a);

Writeln('Vvedite b');

Readln(b);

Writeln('Vvedite n');

Readln(n);

h:=(b-a)/n;

sum:=0;

dopper:=((a\*a-2\*a+3)+(b\*b-2\*b+3))/2 ;

x:=a+h;

**while x** < (b-h)

**do**

**begin**

f:=(sqrt(0.4\*x+1.7))/(1.5\*x+sqrt(x\*x+1.3));

sum:=sum+f;

x:=x+h;

**end**;

it:=h\*(dopper+sum);

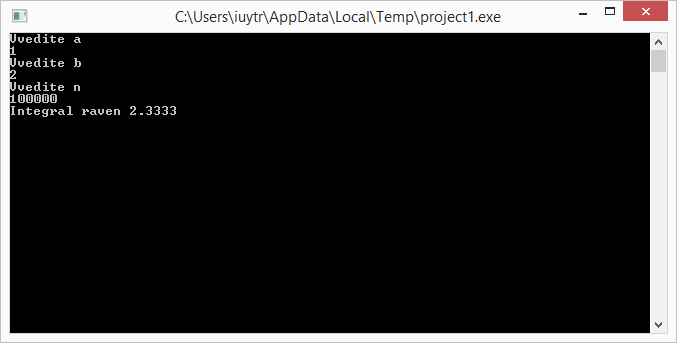
Writeln('Integral raven',' ',it:2:4);

readln();

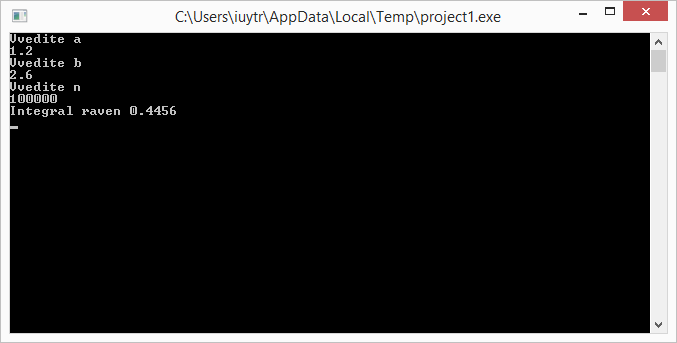
**end**.

**Результат работы программы:**

**1 подпункт:**



**2 подпункт:**



**Анализ:** мы написали программу, которая вычисляет определенный интеграл методом трапеций. Сначала мы вводим левую границу(a), потом границу b, потом количество отрезков разбиения, далее программа считает шаг h= (b-a)/n и начинает вычислять интеграл по заданному алгоритму. Примечание: чем больше количество отрезков разбиения, тем выше точность ответа.

**Вывод:** мы написали программу, которая вычисляет определенный интеграл методом трапеций.

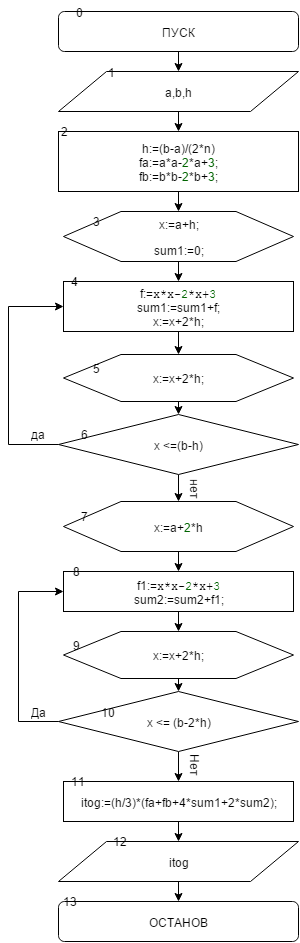
**Задание 4.**

**Постановка задачи:** Написать программу для вычисления определенного интеграла из индивидуального задания методом парабол. Протестировать программу на определенном интеграле, вычисленном в ходе выполнения самостоятельной работы 3. Вычислить определенный интеграл из индивидуального задания

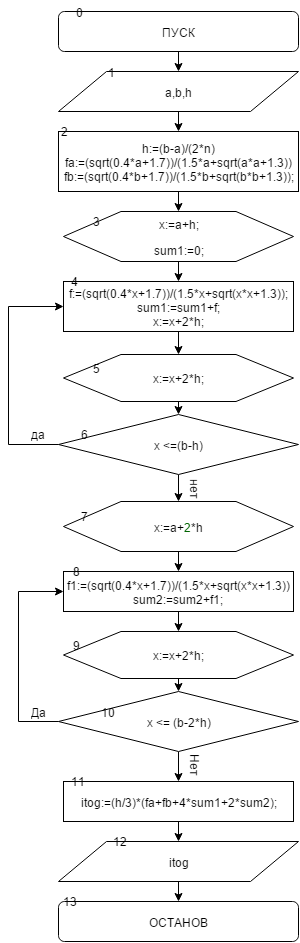
**Математическая модель:**

**Блок-схема:**

**1 подпункт:**



**2 подпункт:**



**Список идентификаторов:**

|  |  |  |
| --- | --- | --- |
| ИМЯ | ТИП | СМЫСЛ |
| a | real | Предел интегрирования |
| b | real | Предел интегрирования |
| n | real | Количество отрезков разбиения |
| x | real | Параметр цикла |
| f | real | Значение первой функции |
| sum1 | real | Первая сумма |
| sum2 | real | Вторая сумма |
| f1 | real | Значение второй функции |
| itog | real | Итог |
| h | real | Шаг |
| fa | real | Значение функции в x=a |
| fb | real | Значение функции в х=b |

**Код программы:**

**1 подпункт:**

**program** parabol1;

**var** a,b,n,x,f,sum1,sum2,f1,itog,h,fa,fb:real;

**begin**

Writeln('Vvedite a');

Readln(a);

Writeln('Vvedite b');

Readln(b);

Writeln('Vvedite n');

Readln(n);

h:=(b-a)/(2\*n);

x:=a+h;

fa:=a\*a-2\*a+3;

fb:=b\*b-2\*b+3;

sum1:=0;

sum2:=0;

**While** x <= (b-h)

**do begin**

f:=x\*x-2\*x+3;

sum1:=sum1+f;

x:=x+2\*h;

**end**;

x:=a+2\*h;

**while** x <= (b-2\*h)

**do begin**

f1:=x\*x-2\*x+3;

sum2:=sum2+f1;

x:=x+2\*h;

**end**;

itog:=(h/3)\*(fa+fb+4\*sum1+2\*sum2);

writeln(itog :2:4);

readln();

**end**.

**2 подпункт:**

**program** parabol2;

**var** a,b,n,x,f,sum1,sum2,f1,itog,h,fa,fb:real;

**begin**

Writeln('Vvedite a');

Readln(a);

Writeln('Vvedite b');

Readln(b);

Writeln('Vvedite n');

Readln(n);

h:=(b-a)/(2\*n);

x:=a+h;

fa:=(sqrt(0.4\*a+1.7))/(1.5\*a+sqrt(a\*a+1.3));

fb:=(sqrt(0.4\*b+1.7))/(1.5\*b+sqrt(b\*b+1.3));

sum1:=0;

sum2:=0;

**While** x <= (b-h)

**do begin**

f:=(sqrt(0.4\*x+1.7))/(1.5\*x+sqrt(x\*x+1.3));

sum1:=sum1+f;

x:=x+2\*h;

**end**;

x:=a+2\*h;

**while** x <= (b-2\*h)

**do begin**

f1:=(sqrt(0.4\*x+1.7))/(1.5\*x+sqrt(x\*x+1.3));

sum2:=sum2+f1;

x:=x+2\*h;

**end**;

itog:=(h/3)\*(fa+fb+4\*sum1+2\*sum2);

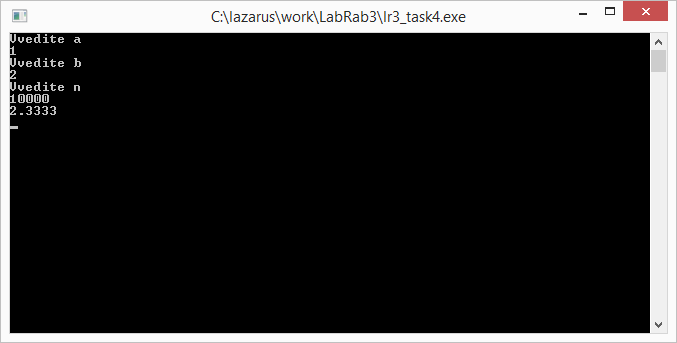
writeln(itog :2:4);

readln();

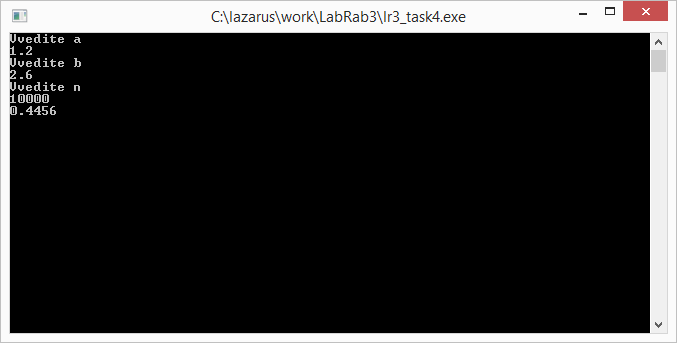
**end**.

**Результат работы программы:**

**1 подпункт:**



**2 подпункт:**



**Анализ:** мы написали программу, которая вычисляет определенный интеграл методом парабол. Сначала мы вводим левую границу(a), потом границу b, потом количество отрезков разбиения, далее программа считает шаг h= (b-a)/(2\*n) и начинает вычислять интеграл по заданному алгоритму. Примечание: чем больше количество отрезков разбиения, тем выше точность ответа.

**Вывод:** мы написали программу, которая вычисляет определенный интеграл методом парабол.

**Анализы результатов в виде таблицы**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| N  Количество разбиений | H  шаг | I  Метод левых частей прямоугольников | I  Метод правых частей прямоугольников | I  Метод трапеций | I  Метод  парабол |
| 10 | 0.1 | 2.0040 | 2.0850 | 2.0540 | 1.9587 |
| 100 | 0.01 | 2.2985 | 2.3084 | 2.3035 | 2.2739 |
| 1000 | 0.001 | 2.3328 | 2.3338 | 2.3333 | 2.3273 |

**1 подпункт:**

**2 подпункт:**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| N  Количество разбиений | H  шаг | I  Метод левых частей прямоугольников | I  Метод правых частей прямоугольников | I  Метод трапеций | I  Метод  парабол |
| 10 | 0.1 | 2.0040 | 2.0850 | 2.0540 | 1.9587 |
| 100 | 0.01 | 2.2985 | 2.3084 | 2.3035 | 2.2739 |
| 1000 | 0.001 | 2.3328 | 2.3338 | 2.3333 | 2.3273 |
| 10000 | 0.0001 | 2.3333 | 2.3334 | 2.3333 | 2.3273 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| N  Количество разбиений | H  шаг | I  Метод левых частей прямоугольников | I  Метод правых частей прямоугольников | I  Метод трапеций | I  Метод  парабол |
| 10 | 0.14 | 0.4228 | 0.3988 | 0.8249 | 0.3729 |
| 100 | 0.014 | 0.4469 | 0.4443 | 0.4871 | 0.4387 |
| 1000 | 0.0014 | 0.4457 | 0.4455 | 0.4497 | 0.4449 |
| 10000 | 0.00014 | 0.4456 | 0.4456 | 0.4460 | 0.4455 |

**Вывод:** рассмотрев данные значения таблицы, можно сделать вывод, что метод левых частей прямоугольников является наиболее точным.

Для того, чтобы повысить точность любого метода, необходимо задать большее количество отрезков разбиения.