***Приложение В***

Задание 1

по Модифицированному методу Ньютона

|  |  |  |  |
| --- | --- | --- | --- |
| Типовая задача (условие) | Математическая модель | Выводимые  данные: тип данных | Входные данные: тип данных |
| Для примера:  Вычислить уравнение 12+cos(1)-3 |  | 2.259957821505073 | X = 1  A = 3 |
| Для примера:  Вычислить уравнение 12+cos(1)-3,5 |  | 2.6915395889389178 | X = 1  A = 3,5 |
| Для примера:  Вычислить уравнение 12+cos(1)-4 |  | 3.123121356372763 | X = 1  A = 4 |
| Для примера :  Вычислить уравнение 12+cos(1)-4,5 |  | 3.554703123806608 | X = 1  A = 4,5 |
| Для примера:  Вычислить уравнение 12+cos(1)-5 |  | 3.9862848912404534 | X = 1  A = 5 |
| Для примера:  Вычислить уравнение 12+cos(1)-5,5 |  | 4.417866658674298 | X = 1  A = 5,5 |
| Для примера:  Вычислить уравнение 12+cos(1)-6 |  | 4.849448426108143 | X = 1  A = 6 |
| Для примера:  Вычислить уравнение 12+cos(1)-6,5 |  | 5.281030193541988 | X = 1  A = 6,5 |
| Для примера:  Вычислить уравнение 12+cos(1)-7 |  | 5.712611960975833 | X = 1  A = 7 |

по Методу простых итераций

|  |  |  |  |
| --- | --- | --- | --- |
| Типовая задача (условие) | Математическая модель | Выводимые  данные: тип данных | Входные данные: тип данных |
| Для примера:  Вычислить уравнение 12+cos(1)-3 |  | -55.588799919401325 | X = 1  A = 3 |
| Для примера:  Вычислить уравнение 12+cos(1)-3,5 |  | -70.0078322768962 | X = 1  A = 3,5 |
| Для примера:  Вычислить уравнение 12+cos(1)-4 |  | -83.56802495307927 | X = 1  A = 4 |
| Для примера :  Вычислить уравнение 12+cos(1)-4,5 |  | -95.27530117665097 | X = 1  A = 4,5 |
| Для примера:  Вычислить уравнение 12+cos(1)-5 |  | -104.58924274663137 | X = 1  A = 5 |
| Для примера:  Вычислить уравнение 12+cos(1)-5,5 |  | -111.55540325570391 | X = 1  A = 5,5 |
| Для примера:  Вычислить уравнение 12+cos(1)-6 |  | -116.79415498198925 | X = 1  A = 6 |
| Для примера:  Вычислить уравнение 12+cos(1)-6,5 |  | -121.34880011912183 | X = 1  A = 6,5 |
| Для примера:  Вычислить уравнение 12+cos(1)-7 |  | -126.43013401281212 | X = 1  A = 7 |

|  |  |  |  |
| --- | --- | --- | --- |
| Класс | Назначение | Поля класса | Методы класса |
| Main | Спрашивает и передает параметры для вычисления | double x — начальное число  double а0 — минимальное значение для цинка  double аn — максимальное значение для цикла  double h - шаг | main  Запускает программу |
| ModNewtonMethod | Класс хранящий методы для решения | - | Function - метод в котором храниться функция F(x)  derivative - метод в котором продифференциированое функция F(x)  modifiedNewton - метод который высчитывает результат по Модифицированному методу Ньютона  simpleIteration - метод который высчитывает результат по Методу простых итераций |

|  |  |
| --- | --- |
| Созданные классы (название) | Листинг (код, скрин) |
| Main | import java.util.Scanner;  public class Main {  public static void main(String[] args) {  Scanner sc = new Scanner(System.*in*);  System.*out*.print("Введи начальное число: ");  double x = sc.nextDouble();  System.*out*.print("Введи минимальное а: ");  double a0 = sc.nextDouble();  System.*out*.print("Введи максимальное а: ");  double an = sc.nextDouble();  System.*out*.print("Введи шаг: ");  double h = sc.nextDouble();   for (double a = a0; a <= an; a = a + h) {  System.*out*.println("Ответ: " + ModNewtonMethod.*simpleIteration*(x, a));  }  } } |
| ModNewtonMethod | public class ModNewtonMethod {  public static double function(double x, double a) {  return Math.*pow*(x, 2) + Math.*cos*(x) - a;  }  public static double derivative(double x) {  return 2 \* x - Math.*sin*(x);  }  public static double modifiedNewton(double x, double a) {  double fx0 = *function*(x, a);  double dfx0 = *derivative*(x);  x = x - fx0 / dfx0;  return x;  }  public static double simpleIteration(double x, double a) {  double prev=a;  double next=prev;  for (int i = 0; i < x; i++) {  double M=0;  for (int j = 0; j < x; j++) {  if (*derivative*(j)>M){  M=*derivative*(j);  }  }  next=prev-*derivative*(prev)/(M+0.1);  prev=next;  }  return next;  } } |
| ModNewtonMethodTest | import static org.junit.jupiter.api.Assertions.\*;  class ModNewtonMethodTest {   @org.junit.jupiter.api.Test  void function() {  double actual = ModNewtonMethod.*function*(1,1);  *assertEquals*(ModNewtonMethod.*function*(1,1),actual);  }   @org.junit.jupiter.api.Test  void derivative() {  double actual = ModNewtonMethod.*derivative*(1);  *assertEquals*(ModNewtonMethod.*derivative*(1),actual);  }   @org.junit.jupiter.api.Test  void modifiedNewton() {  double actual = ModNewtonMethod.*modifiedNewton*(1,3);  *assertEquals*(ModNewtonMethod.*modifiedNewton*(1,3),actual);  }   @org.junit.jupiter.api.Test  void simpleIteration() {  double actual = ModNewtonMethod.*simpleIteration*(1,3);  *assertEquals*(ModNewtonMethod.*simpleIteration*(1,3),actual);  } } |

Задание 2

по формуле левых прямоугольников

|  |  |  |  |
| --- | --- | --- | --- |
| Типовая задача (условие) | Математическая модель | Выводимые  данные: тип данных | Входные данные: тип данных |
| Для примера:  Вычислить уравнение  ln x |  | 0.26616923207259047 | A = 1  B = 2  E = 3 |

по формуле правых прямоугольников

|  |  |  |  |
| --- | --- | --- | --- |
| Типовая задача (условие) | Математическая модель | Выводимые  данные: тип данных | Входные данные: тип данных |
| Для примера:  Вычислить уравнение  ln x |  | 0.49721829225923886 | A = 1  B = 2  E = 3 |

|  |  |  |  |
| --- | --- | --- | --- |
| Класс | Назначение | Поля класса | Методы класса |
| Main | Спрашивает и передает параметры для вычисления | Double a — Число с точкой (нужно для обозначения начала интеграла)  Double b — Число с точкой (нужно для обозначения конца интеграла)  Double e — Число с точкой (нужно для обозначения количества интегралов) | main  Запускает программу |
| MA | Класс хранящий методы для решения | - | Function — метод в котором храниться функция F(x)  leftRectangleMethod — метод который высчитывает результат по формуле левых прямоугольников  rightRectangleMethod - метод который высчитывает результат по формуле правых прямоугольников |

|  |  |
| --- | --- |
| Созданные классы (название) | Листинг (код, скрин) |
| Main |  |
| MA |  |
| MATest | import org.junit.jupiter.api.Test;  import static org.junit.jupiter.api.Assertions.\*;  class MATest {   @Test  void function() {  double actual = MA.*function*(1);  *assertEquals*(MA.*function*(1),actual);  }   @Test  void leftRectangleMethod() {  double actual = MA.*leftRectangleMethod*(1,2,3);  *assertEquals*(MA.*leftRectangleMethod*(1,2,3), actual);  }   @Test  void rightRectangleMethod() {  double actual = MA.*rightRectangleMethod*(1,2,3);  *assertEquals*(MA.*rightRectangleMethod*(1,2,3), actual);  }} |