**24. Численные методы решения дифференциальных уравнений. Метод Эйлера-Коши, Метод Рунге-Кутта.**

Метод Эйлера-Коши является модификацией метода Эйлера. Он основан на том, что половину шага совершается с тангенсом угла наклона касательной в предыдущей точке, а вторую - с тангенсом угла наклона в последующей точке (Рис. 1).

Формула для решения методом Эйлера-Коши будет следующей: - формула Эйлера-Коши. В левой и в правой части формулы Эйлера-Коши стоит неизвестная искомая величина - метод является неявным. Для его реализации находится приближенное значение методом Эйлера, подставляется в правую часть формулы Эйлера-Коши и находится уточненное значение .

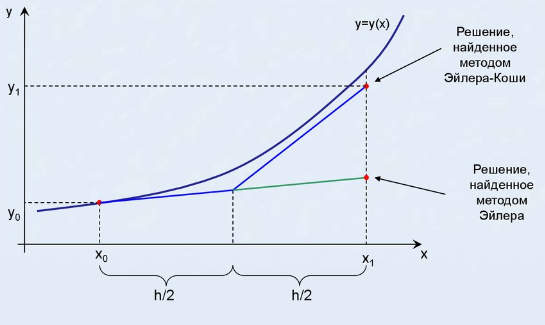
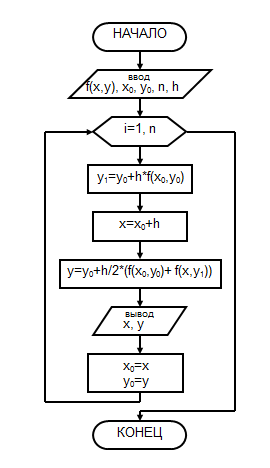


Рисунок 1

Блок схема вышеуказанного метода:



Метод Рунге — Кутты четвёртого порядка при вычислениях с постоянным шагом интегрирования столь широко распространён, что его часто называют просто методом Рунге — Кутты. Он является одношаговым, имеет явную схему, но не всегда устойчив. Для реализации этого метода используются достаточно громоздкие формулы Рунге-Кутта:

Исходный код ниже:

|  |
| --- |
| **package** lagrang;  **public** **class** differential {  **double** dydx(**double** x, **double** y)  {  **return** ((x - y) / 2);  }  // Находит значение y для заданного x, используя размер шага h  // и начальное значение y0 в x0.  **double** rungeKutta(**double** x0, **double** y0, **double** x, **double** h)  {  differential d1 = **new** differential();  // Подсчитать количество итераций, используя размер шага или  // высота шага h  **int** n = (**int**)((x - x0) / h);  **double** k1, k2, k3, k4, k5;  // Итерация по количеству итераций  **double** y = y0;  **for** (**int** i = 1; i <= n; i++)  {  // Применить формулы Рунге Кутты, чтобы найти  // следующее значение у  k1 = h \* (d1.dydx(x0, y));  k2 = h \* (d1.dydx(x0 + 0.5 \* h, y + 0.5 \* k1));  k3 = h \* (d1.dydx(x0 + 0.5 \* h, y + 0.5 \* k2));  k4 = h \* (d1.dydx(x0 + h, y + k3));  // Обновить следующее значение y  y = y + (1.0 / 6.0) \* (k1 + 2 \* k2 + 2 \* k3 + k4);  // Обновляем следующее значение x  x0 = x0 + h;  }  **return** y;  }  **public** **static** **void** main(String args[])  {  differential d2 = **new** differential();  **double** x0 = 0, y = 1, x = 2, h = 0.2;  System.***out***.println("\nЗначение y при x равно : "  + d2.rungeKutta(x0, y, x, h));  }  } |

Блок схема вышеуказанного метода:

