Федеральное государственное автономное

образовательное учреждение высшего образования

“Национальный исследовательский университет ИТМО”

Факультет программной инженерии и компьютерной техники

Лабораторная работа №6

по вычислительной математике

Вариант №3

Выполнил:

Голиков Андрей

Группа: P32092

Принял: Рыбаков

Степан Дмитриевич

???

?



Санкт-Петербург, 2023

Цель работы: решить задачу Коши для обыкновенных дифференциальных уравнений численными методами.

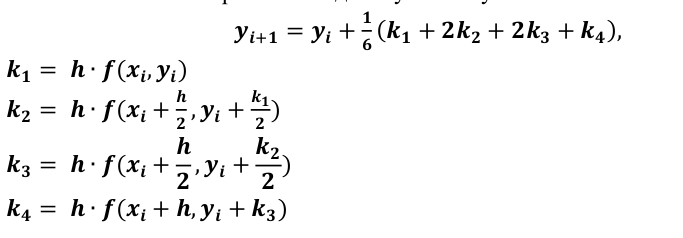
Рабочие формулы:

Метод Эйлера

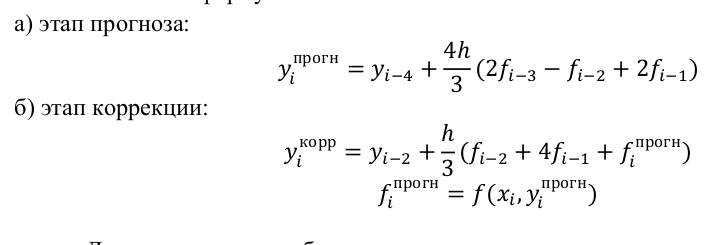


Модифицированный метод Эйлера 

Метод Рунге-Кутта



Метод Милна



Листинг программы:

module Evaluator where

runge :: Points -> Points -> Order -> Eps

runge p1 p2 p = ((snd $ p1 !! 1) - (snd $ p1 !! 2)) / (2^p - 1)

template :: Form -> Order -> String -> Method

template form ord s fnum h (x0, xn) y0 acc = template' h where

f = equations !! fnum

template' d = if eps < acc then (p1, eps, s) else template' (d/2) where

eps = runge p1 p2 ord

p1 = take n1 $ iterate (form f h) (x0, y0)

n1 = (+) 1 $ floor $ (xn - x0) / d

p2 = take n2 $ iterate (form f h) (x0, y0)

n2 = (+) 1 $ floor $ (xn - x0) / (d / 2)

eiler :: Method

eiler = template eiler\_form 1 "eiler"

eiler\_form :: Form

eiler\_form f h (x, y) = (next\_x, next\_y) where

next\_x = x + h

next\_y = y + h \* (f x y)

mod\_eiler :: Method

mod\_eiler = template mod\_eiler\_form 2 "modified eiler"

mod\_eiler\_form :: Form

mod\_eiler\_form f h (x, y) = (next\_x, next\_y) where

next\_x = x + h

next\_y = y + h / 2 \* (f x y + f next\_x e\_y)

e\_y = y + h \* (f x y)

runge\_kutt :: Method

runge\_kutt = template runge\_kutt\_form 4 "runge-kutt"

runge\_kutt\_form :: Form

runge\_kutt\_form f h (x, y) = (next\_x, next\_y) where

next\_x = x + h

next\_y = y + (k1 + 2 \* k2 + 2 \* k3 + k4) / 6

k1 = h \* (f x y)

k2 = h \* (f (x + h / 2) (y + k1 / 2))

k3 = h \* (f (x + h / 2) (y + k2 / 2))

k4 = h \* (f (x + h) (y + k3))

miln :: Method

miln fnum h (x0, xn) y0 acc = (take n seq, acc, "miln") where

n = (+) 1 $ floor $ (xn - x0) / h

f = equations !! fnum

beg = take 4 $ iterate (runge\_kutt\_form f h) (x0, y0)

seq = beg ++ (zipWith4 form seq (tail seq) (drop 2 seq) (drop 3 seq))

form p4 p3 p2 p1 = form' pred where

y4 = snd p4

y2 = snd p2

x0 = h + (fst p1)

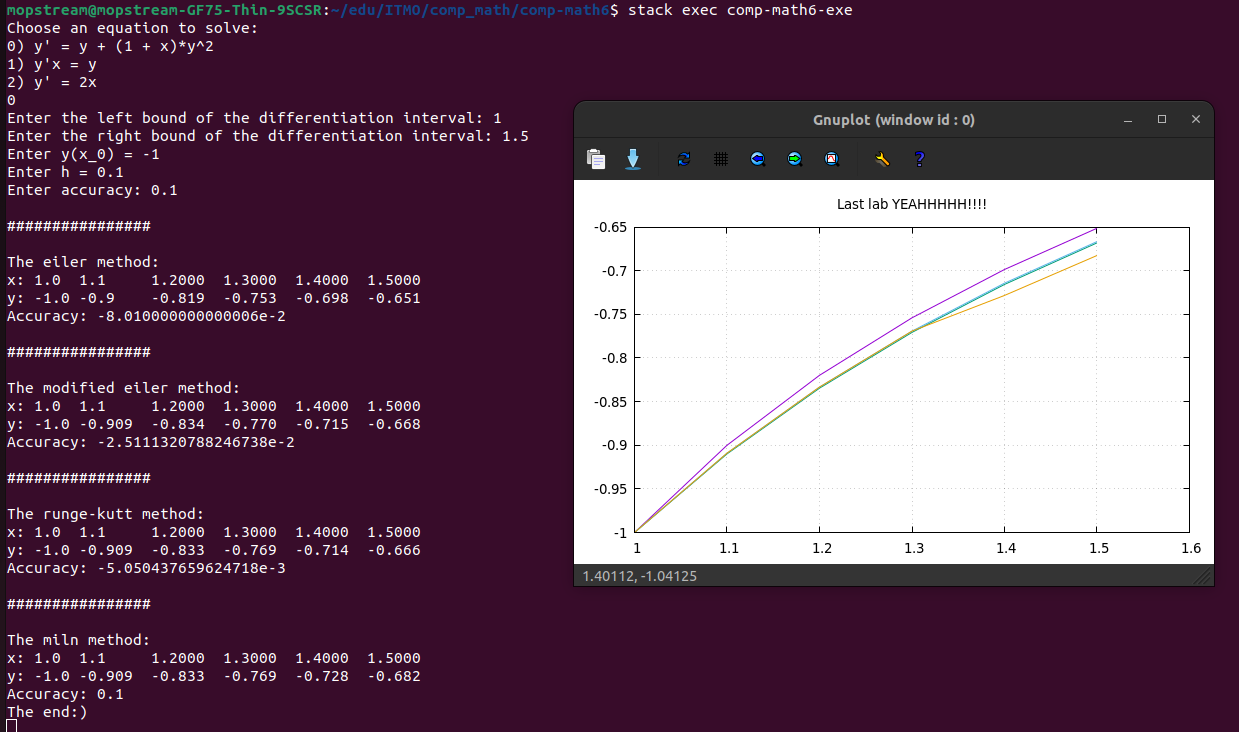
fp = uncurry f

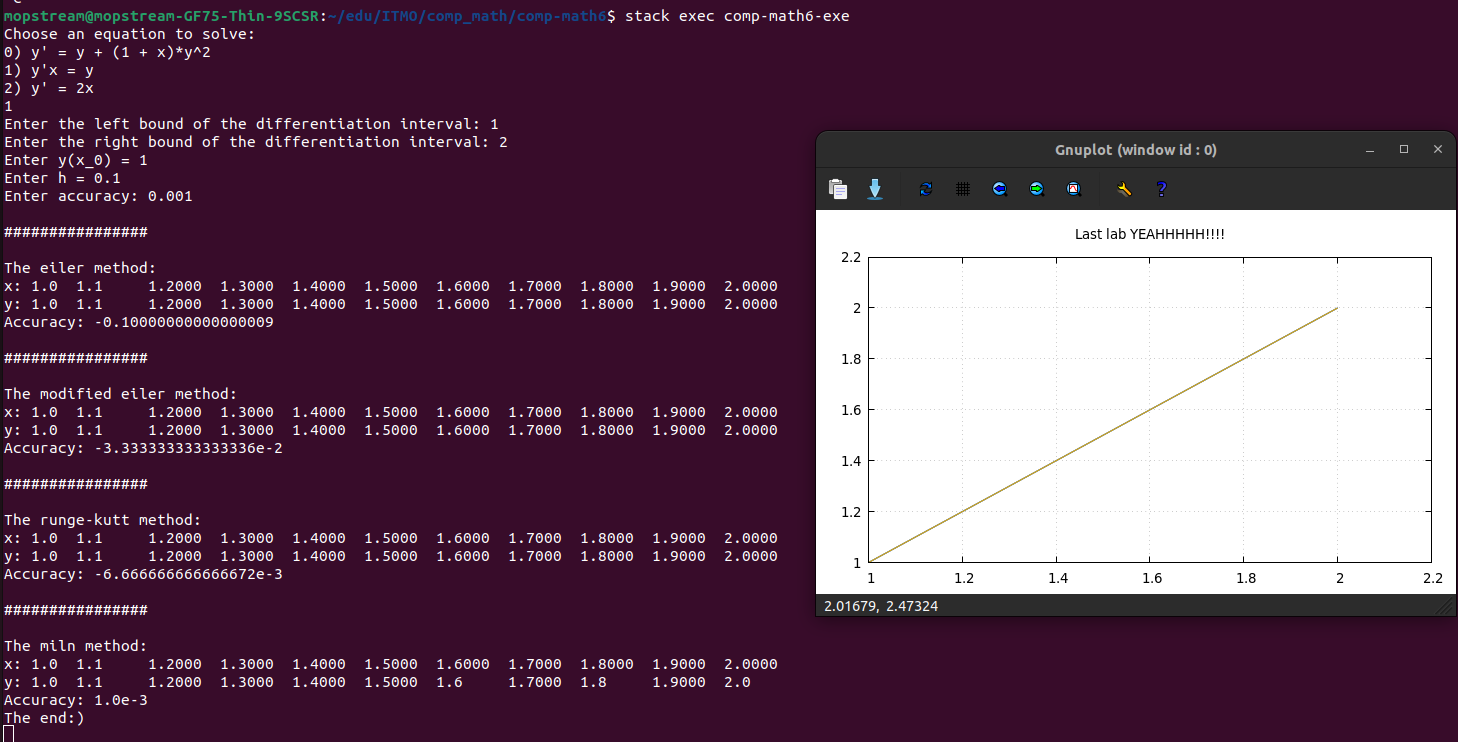
pred = y4 + 4\*h/3 \*(2\*(fp p3) - (fp p2) + 2 \* (fp p1))

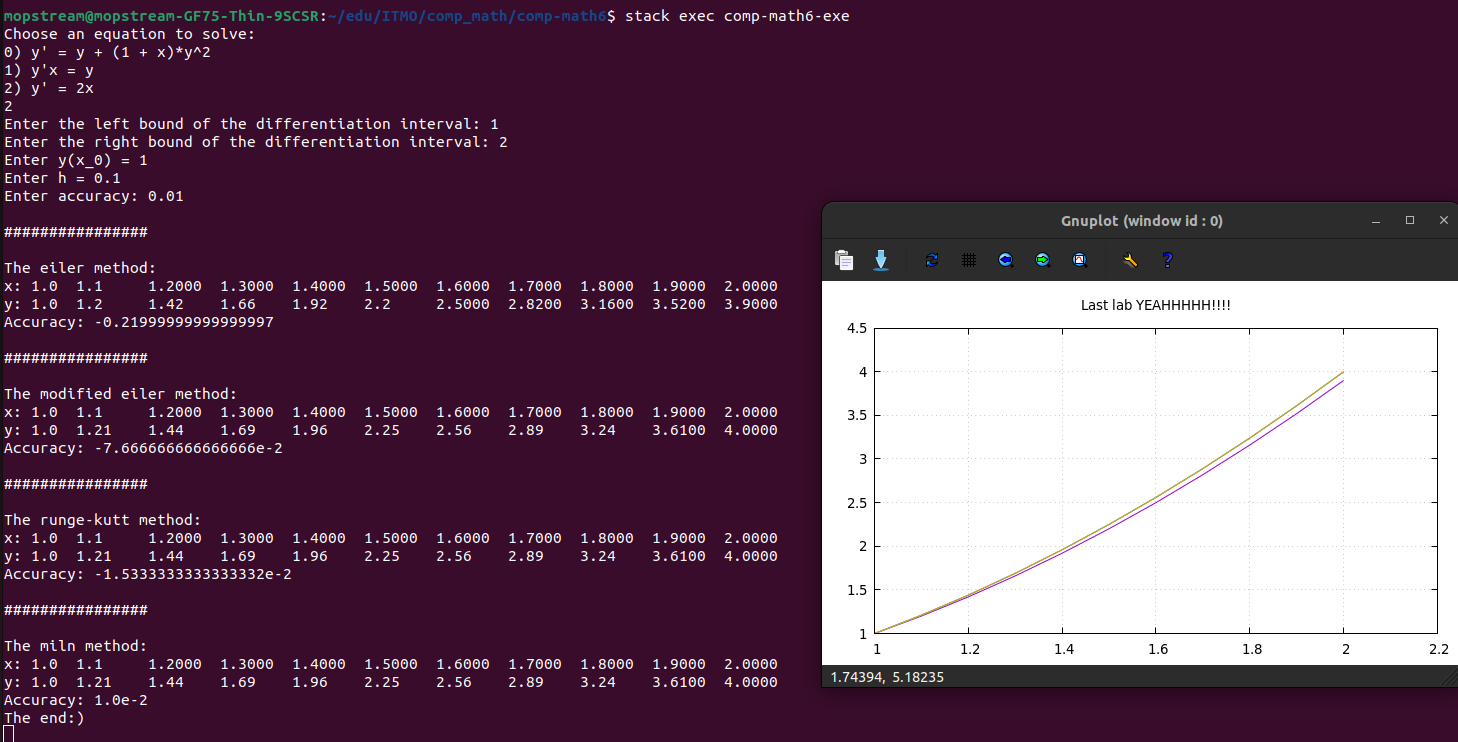
form' pred = if abs (pred - new) < acc then (x0, new) else form' new where

new = y2 + h/3\*((fp p2) + 4\*(fp p1) + (f x0 pred))

Результаты работы программы:







Вывод: реализовал на языке Haskell решение обыкновенных дифференциальных уравнений первого порядка методами Эйлера, модифицированным Эйлера, Рунге-Кутта и Милна. Больше всего отличаются от истинного значения значения, полученные методами Эйлера, так как сам метод дает только точность первого порядка.