Федеральное государственное автономное образовательное учреждение высшего образования **«Национальный исследовательский университет ИТМО»**

Факультет Программной Инженерии и Компьютерной Техники

Лабораторная работа **№6**

**«Численное решение обыкновенных дифференциальных уравнений»**

по дисциплине «Вычислительная математика**»**

Вариант: **11**

**Преподаватель:**   
Малышева Татьяна Алексеевна

**Выполнил:**

ХХХХХХХХХХХ

**Группа:** ХХХХ

Санкт-Петербург, 2025 г.

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

# 2. Необходимые формулы

## Усовершенствованный метод Эйлера

## Метод Рунге-Кутты IIII порядка

## Метод Адамса

### Предиктор

### Корректор

## Метод Рунге

### Определение погрешности

# 2. Программная реализация задачи

## Листинг методов

from typing import Callable, List

def improved\_euler\_method(f: Callable, xs: List[float], y0: float) -> List[float]:

ys = [y0]

h = xs[1] - xs[0]

for i in range(len(xs) - 1):

y\_pred = ys[i] + h \* f(xs[i], ys[i])

ys.append(ys[i] + 0.5 \* h \* (f(xs[i], ys[i]) + f(xs[i + 1], y\_pred)))

return ys

def fourth\_order\_runge\_kutta\_method(f: Callable, xs: List[float], y0: float) -> List[float]:

ys = [y0]

h = xs[1] - xs[0]

for i in range(len(xs) - 1):

k1 = h \* f(xs[i], ys[i])

k2 = h \* f(xs[i] + h / 2, ys[i] + k1 / 2)

k3 = h \* f(xs[i] + h / 2, ys[i] + k2 / 2)

k4 = h \* f(xs[i] + h, ys[i] + k3)

ys.append(ys[i] + (k1 + 2 \* k2 + 2 \* k3 + k4) / 6)

return ys

def adams(f: Callable, xs: List[float], y0: float) -> List[float]:

if len(xs) < 4:

raise ValueError("Для метода Адамса требуется минимум 4 точки")

ys = fourth\_order\_runge\_kutta\_method(f, xs[:4], y0)[:4]

h = xs[1] - xs[0]

for i in range(3, len(xs) - 1):

f\_prev = [f(xs[j], ys[j]) for j in range(i - 3, i + 1)]

y\_pred = ys[i] + h \* (55 \* f\_prev[3] - 59 \* f\_prev[2] + 37 \* f\_prev[1] - 9 \* f\_prev[0]) / 24

f\_pred = f(xs[i + 1], y\_pred)

ys.append(ys[i] + h \* (9 \* f\_pred + 19 \* f\_prev[3] - 5 \* f\_prev[2] + f\_prev[1]) / 24)

return ys

## Примеры работы

### Пример №1

Выберите функцию

1: y + (1 + x)\*y^2

2: f' = sin(x) - y

3: f' = y / x

Введите номер функции: 1

Задайте начальные условия:

Введите интервал: 0.1 0.9

Введите шаг: 0.01

Введите y0: 0.01

Введите точность: 0.001

Выберите метод:

1. Усовершенствованный метод Эйлера

2. Метод Рунге-Кутта 4-го порядка

3. Метод Адамса

Введите номер метода: 1

Алгоритм выполнил 2 итерацию

i x\_i y\_i y\_точн

0: 0.10000000 0.01000000 0.01000000

1: 0.10500000 0.01005068 0.01005068

2: 0.11000000 0.01010162 0.01010162

3: 0.11500000 0.01015283 0.01015283

4: 0.12000000 0.01020430 0.01020430

5: 0.12500000 0.01025604 0.01025604

6: 0.13000000 0.01030804 0.01030805

7: 0.13500000 0.01036032 0.01036032

8: 0.14000000 0.01041287 0.01041287

9: 0.14500000 0.01046568 0.01046569

10: 0.15000000 0.01051878 0.01051878

11: 0.15500000 0.01057214 0.01057215

12: 0.16000000 0.01062579 0.01062579

13: 0.16500000 0.01067971 0.01067972

14: 0.17000000 0.01073392 0.01073392

15: 0.17500000 0.01078840 0.01078840

16: 0.18000000 0.01084317 0.01084317

17: 0.18500000 0.01089822 0.01089822

18: 0.19000000 0.01095356 0.01095356

19: 0.19500000 0.01100918 0.01100919

20: 0.20000000 0.01106510 0.01106510

21: 0.20500000 0.01112130 0.01112131

22: 0.21000000 0.01117780 0.01117781

23: 0.21500000 0.01123459 0.01123460

24: 0.22000000 0.01129168 0.01129169

25: 0.22500000 0.01134906 0.01134907

26: 0.23000000 0.01140675 0.01140675

27: 0.23500000 0.01146473 0.01146474

28: 0.24000000 0.01152302 0.01152303

29: 0.24500000 0.01158161 0.01158162

30: 0.25000000 0.01164051 0.01164051

31: 0.25500000 0.01169971 0.01169972

32: 0.26000000 0.01175922 0.01175923

33: 0.26500000 0.01181904 0.01181905

34: 0.27000000 0.01187918 0.01187919

35: 0.27500000 0.01193963 0.01193964

36: 0.28000000 0.01200039 0.01200040

37: 0.28500000 0.01206147 0.01206148

38: 0.29000000 0.01212288 0.01212289

39: 0.29500000 0.01218460 0.01218461

40: 0.30000000 0.01224664 0.01224666

41: 0.30500000 0.01230902 0.01230903

42: 0.31000000 0.01237171 0.01237172

43: 0.31500000 0.01243474 0.01243475

44: 0.32000000 0.01249809 0.01249810

45: 0.32500000 0.01256178 0.01256179

46: 0.33000000 0.01262580 0.01262581

47: 0.33500000 0.01269016 0.01269017

48: 0.34000000 0.01275485 0.01275487

49: 0.34500000 0.01281989 0.01281990

50: 0.35000000 0.01288526 0.01288528

51: 0.35500000 0.01295098 0.01295100

52: 0.36000000 0.01301705 0.01301706

53: 0.36500000 0.01308346 0.01308347

54: 0.37000000 0.01315022 0.01315023

55: 0.37500000 0.01321733 0.01321734

56: 0.38000000 0.01328479 0.01328481

57: 0.38500000 0.01335261 0.01335263

58: 0.39000000 0.01342079 0.01342080

59: 0.39500000 0.01348932 0.01348934

60: 0.40000000 0.01355822 0.01355824

61: 0.40500000 0.01362748 0.01362750

62: 0.41000000 0.01369710 0.01369712

63: 0.41500000 0.01376709 0.01376711

64: 0.42000000 0.01383746 0.01383748

65: 0.42500000 0.01390819 0.01390821

66: 0.43000000 0.01397929 0.01397931

67: 0.43500000 0.01405078 0.01405080

68: 0.44000000 0.01412264 0.01412266

69: 0.44500000 0.01419487 0.01419490

70: 0.45000000 0.01426750 0.01426752

71: 0.45500000 0.01434050 0.01434052

72: 0.46000000 0.01441389 0.01441392

73: 0.46500000 0.01448767 0.01448770

74: 0.47000000 0.01456184 0.01456187

75: 0.47500000 0.01463641 0.01463643

76: 0.48000000 0.01471137 0.01471139

77: 0.48500000 0.01478673 0.01478675

78: 0.49000000 0.01486248 0.01486251

79: 0.49500000 0.01493864 0.01493867

80: 0.50000000 0.01501521 0.01501523

81: 0.50500000 0.01509218 0.01509220

82: 0.51000000 0.01516956 0.01516958

83: 0.51500000 0.01524735 0.01524737

84: 0.52000000 0.01532555 0.01532558

85: 0.52500000 0.01540417 0.01540420

86: 0.53000000 0.01548321 0.01548324

87: 0.53500000 0.01556267 0.01556270

88: 0.54000000 0.01564256 0.01564259

89: 0.54500000 0.01572287 0.01572290

90: 0.55000000 0.01580360 0.01580364

91: 0.55500000 0.01588477 0.01588481

92: 0.56000000 0.01596638 0.01596641

93: 0.56500000 0.01604841 0.01604845

94: 0.57000000 0.01613089 0.01613092

95: 0.57500000 0.01621381 0.01621384

96: 0.58000000 0.01629717 0.01629720

97: 0.58500000 0.01638098 0.01638101

98: 0.59000000 0.01646523 0.01646527

99: 0.59500000 0.01654994 0.01654998

100: 0.60000000 0.01663510 0.01663514

101: 0.60500000 0.01672072 0.01672076

102: 0.61000000 0.01680680 0.01680683

103: 0.61500000 0.01689333 0.01689337

104: 0.62000000 0.01698034 0.01698038

105: 0.62500000 0.01706781 0.01706785

106: 0.63000000 0.01715575 0.01715579

107: 0.63500000 0.01724416 0.01724420

108: 0.64000000 0.01733305 0.01733310

109: 0.64500000 0.01742242 0.01742246

110: 0.65000000 0.01751227 0.01751231

111: 0.65500000 0.01760260 0.01760265

112: 0.66000000 0.01769343 0.01769347

113: 0.66500000 0.01778474 0.01778478

114: 0.67000000 0.01787654 0.01787658

115: 0.67500000 0.01796884 0.01796888

116: 0.68000000 0.01806164 0.01806168

117: 0.68500000 0.01815494 0.01815498

118: 0.69000000 0.01824874 0.01824879

119: 0.69500000 0.01834305 0.01834310

120: 0.70000000 0.01843787 0.01843792

121: 0.70500000 0.01853321 0.01853326

122: 0.71000000 0.01862906 0.01862911

123: 0.71500000 0.01872543 0.01872549

124: 0.72000000 0.01882233 0.01882238

125: 0.72500000 0.01891975 0.01891980

126: 0.73000000 0.01901770 0.01901776

127: 0.73500000 0.01911619 0.01911624

128: 0.74000000 0.01921520 0.01921526

129: 0.74500000 0.01931476 0.01931482

130: 0.75000000 0.01941486 0.01941492

131: 0.75500000 0.01951551 0.01951556

132: 0.76000000 0.01961670 0.01961676

133: 0.76500000 0.01971845 0.01971851

134: 0.77000000 0.01982075 0.01982081

135: 0.77500000 0.01992361 0.01992367

136: 0.78000000 0.02002703 0.02002709

137: 0.78500000 0.02013102 0.02013108

138: 0.79000000 0.02023558 0.02023564

139: 0.79500000 0.02034070 0.02034077

140: 0.80000000 0.02044641 0.02044647

141: 0.80500000 0.02055269 0.02055276

142: 0.81000000 0.02065956 0.02065963

143: 0.81500000 0.02076701 0.02076708

144: 0.82000000 0.02087506 0.02087513

145: 0.82500000 0.02098370 0.02098377

146: 0.83000000 0.02109293 0.02109300

147: 0.83500000 0.02120277 0.02120284

148: 0.84000000 0.02131321 0.02131328

149: 0.84500000 0.02142426 0.02142433

150: 0.85000000 0.02153592 0.02153599

151: 0.85500000 0.02164820 0.02164827

152: 0.86000000 0.02176110 0.02176117

153: 0.86500000 0.02187462 0.02187469

154: 0.87000000 0.02198877 0.02198884

155: 0.87500000 0.02210355 0.02210362

156: 0.88000000 0.02221896 0.02221904

157: 0.88500000 0.02233502 0.02233510

158: 0.89000000 0.02245172 0.02245180

159: 0.89500000 0.02256906 0.02256914

### Пример №2

Выберите функцию

1: y + (1 + x)\*y^2

2: f' = sin(x) - y

3: f' = y / x

Введите номер функции: 2

Задайте начальные условия:

Введите интервал: 1 10

Введите шаг: 1

Введите y0: 1

Введите точность: 0.1

Выберите метод:

1. Усовершенствованный метод Эйлера

2. Метод Рунге-Кутта 4-го порядка

3. Метод Адамса

Введите номер метода: 2

Алгоритм выполнил 2 итерацию

i x\_i y\_i y\_точн

0: 1.00000000 1.00000000 0.33452406

1: 1.50000000 0.97866211 0.57494397

2: 2.00000000 0.97519997 0.73038977

3: 2.50000000 0.88921008 0.74085038

4: 3.00000000 0.68030953 0.59044979

5: 3.50000000 0.36236673 0.30793542

6: 4.00000000 -0.00939387 -0.04242162

7: 4.50000000 -0.35767687 -0.37781266

8: 5.00000000 -0.60553812 -0.61792426

9: 5.50000000 -0.69734553 -0.70506166

10: 6.00000000 -0.61368744 -0.61855352

11: 6.50000000 -0.37690664 -0.37998210

12: 7.00000000 -0.04610414 -0.04800189

13: 7.50000000 0.29704315 0.29595887

14: 8.00000000 0.56810519 0.56759687

15: 8.50000000 0.70046439 0.70035124

16: 9.00000000 0.66156157 0.66168608

17: 9.50000000 0.46082867 0.46104794

### Пример №3

Выберите функцию

1: y + (1 + x)\*y^2

2: f' = sin(x) - y

3: f' = y / x

Введите номер функции: 3

Задайте начальные условия:

Введите интервал: 10 20

Введите шаг: 1

Введите y0: 3

Введите точность: 0.01

Выберите метод:

1. Усовершенствованный метод Эйлера

2. Метод Рунге-Кутта 4-го порядка

3. Метод Адамса

Введите номер метода: 3

Алгоритм выполнил 1 итерацию

i x\_i y\_i y\_точн

0: 10.00000000 3.00000000 3.00000000

1: 11.00000000 3.30000000 3.30000000

2: 12.00000000 3.60000000 3.60000000

3: 13.00000000 3.90000000 3.90000000

4: 14.00000000 4.20000000 4.20000000

5: 15.00000000 4.50000000 4.50000000

6: 16.00000000 4.80000000 4.80000000

7: 17.00000000 5.10000000 5.10000000

8: 18.00000000 5.40000000 5.40000000

9: 19.00000000 5.70000000 5.70000000

# Вывод

В ходе выполнения данной лабораторной работы я рассмотрел и реализовал численные методы решения обыкновенных дифференциальных уравнений: усовершенствованный метод Эйлера, метод Рунге-Кутта 4-го порядка и метод Адамса. Реализация этих методов была написана на языке Python.  
Я также реализовал правило Рунге для оценки точности одношаговых методов. Визуализация результатов позволила продемонстрировать эффективность каждого из методов. Во время работы я поработал с численными методами в решении обыкновенных дифференциальных уравнений.