|  |  |
| --- | --- |
| Gerb-BMSTU_01 | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

*ФАКУЛЬТЕТ «Информатика и системы управления»*

*КАФЕДРА «Программное обеспечение ЭВМ и информационные технологии»*

**Отчет**

|  |  |
| --- | --- |
| **по лабораторной работе №** | 01 |

**Дисциплина:  *Mоделирование***

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ***ИУ7И-66Б*** |  |  | **Нгуен Ф. С.** |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | **Градов В. М.** |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

*Москва, 2021*

* **Цель работы:** Получение навыков решения задачи Коши для ОДУ методами Пикара и явными методами первого порядка точности (Эйлера) и второго порядка точности (РунгеКутта).
* **Исходные данные.**

|  |  |
| --- | --- |
|  | (1) |

1. **Метод Пикара**

|  |  |
| --- | --- |
|  | (2) |

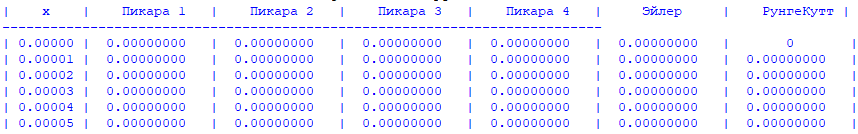
1. **Метод Эйлера:**

|  |  |
| --- | --- |
|  | (3) |

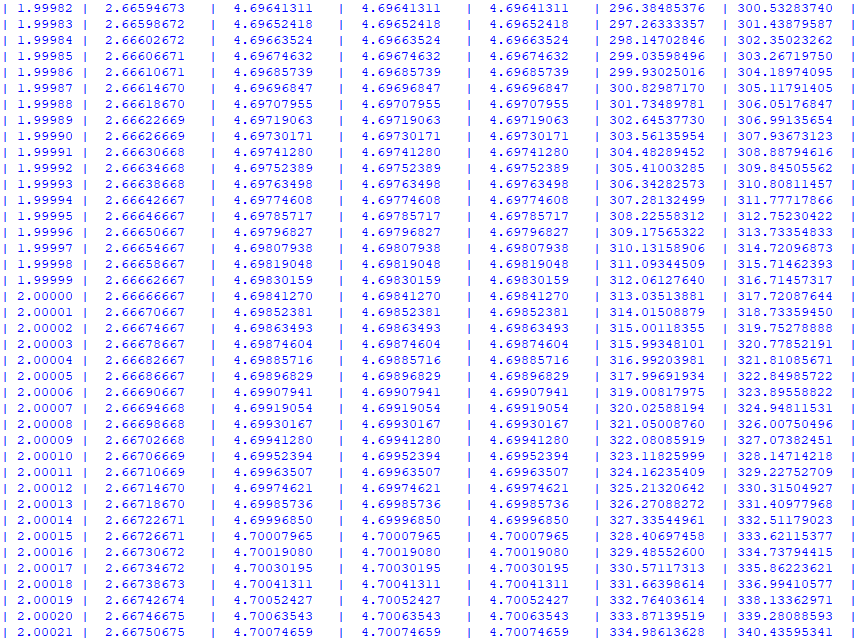
1. **Метод РунгеКутта:**

|  |  |
| --- | --- |
|  | (4) |

1. **Программа**
2. # Эйлера
3. def euler(n, h, x, y):
4. y\_out = []
5. for i in range(n):
6. try:
7. y += h \* f(x, y)
8. y\_out.append(y)
9. x += h
10. except OverflowError:
11. y\_out.append('overflow')
12. for j in range(i, n-1):
13. y\_out.append('-----')
14. break
15. return y\_out
16. # РунгеКутта
17. def rungeKutta(n, h, x, y):
18. y\_out = [y]
19. alpha = 1 / 2
20. h1 = h / (2 \* alpha)
21. for i in range(n):
22. try:
23. k1 = f(x, y)
24. k2 = f(x + h1, y + h1 \* k1)
25. y += h \* ((1 - alpha) \* k1 + alpha \* k2)
26. y\_out.append(y)
27. x += h
28. except:
29. y\_out.append('overflow')
30. for j in range(i, n-1):
31. y\_out.append('-----')
32. break
33. return y\_out
35. # Пикар
36. def picar(n, h, x, y0):
37. def f1(a):
38. return a \*\* 3 / 3
39. def f2(a):
40. return f1(a) + a \*\* 7 / 63
41. def f3(a):
42. return f2(a) + (a \*\* 11) \* (2 / 2079) + (a \*\* 15) / 59535
43. def f4(a):
44. return f3(a) + (a \*\* 15)\*(2 / 93555) + (a \*\* 19)\*(2 / 3393495) + (a \*\* 19)\*(2 / 2488563) + \
45. (a \*\* 23)\*(2 / 86266215) + (a \*\* 23)\*(1 / 99411543) + (a \*\* 27)\*(2 / 3341878155) + (a \*\* 31)\*(1 / 109876902975)
46. y\_out = [[y0, y0, y0, y0]]
47. for i in range(n-1):
48. x += h
49. y\_f1 = f1(x)
50. y\_f2 = f2(x)
51. y\_f3 = f3(x)
52. y\_f4 = f4(x)
53. y\_out.append([y\_f1, y\_f2, y\_f3, y\_f4])
54. return y\_out
56. def main():
57. h = 10 \*\* -5
58. x = 0
59. y0 = 0
60. end = 2.1
61. n = ceil(abs(end - x)/h)+1
62. x\_arr = [x + h\*i for i in range(n)]
63. y\_picar = picar(n, h, x, y0)
64. y\_euler = euler(n, h, x, y0)
65. y\_RungeKutta = rungeKutta(n, h, x, y0)
66. print("| x | Пикара 1 | Пикара 2 | Пикара 3 | Пикара 4 | Эйлер | РунгеКутт |")
67. print("-"\*75)
68. output\_step = int(n/h) # выводим только 100 значений в таблице
69. for i in range(0, n, 10000):
70. print("|{:^9.5f}|{:^15.8f}|{:^15.8f}|{:^15.8f}|{:^15.8f}|{:^15s}|{:^15s}|".format(x\_arr[i], y\_picar[i][0], y\_picar[i][1], y\_picar[i][1], y\_picar[i][1], output(y\_euler[i]), output(y\_RungeKutta[i])))
71. main()
72. **Результат**

****

**…**

****

**…**

1. **Ответы на вопросы:**
   1. ***Укажите интервалы значений аргумента, в которых можно считать решением заданного уравнения каждое из первых 4-х приближений Пикара. Точность результата оценивать до второй цифры после запятой. Объяснить свой ответ.*** 
      * Разница меньше 0.005 в интервал (0, 0.85):

* В интервал (0, 0.85) можно считать решением заданного уравнения каждое из первых 4-х приближений Пикара.
  1. ***Пояснить, каким образом можно доказать правильность полученного результата при фиксированном значении аргумента в численных методах.*** 
     + Численные методы тем точнее, чем меньше шаг
     + Мы сравниваем значения Эйлера при разном шаге (x = 2):
       - При шагe = 10-4 , получаем ≈ 270
       - При шаге = 10-5, получаем ≈ 313
         * *Погрешность при шаге 10-4 была большая.*
* Далее, при шаге = 10-6, получaем ≈ 317.
* При шаге = 10-7, получaем ≈ 317.

*Можно считать, что результат около 317*

* 1. ***Каково значение функции при x=2, т.е. Привести значение u(2).***
     + U(2) ≈ 317