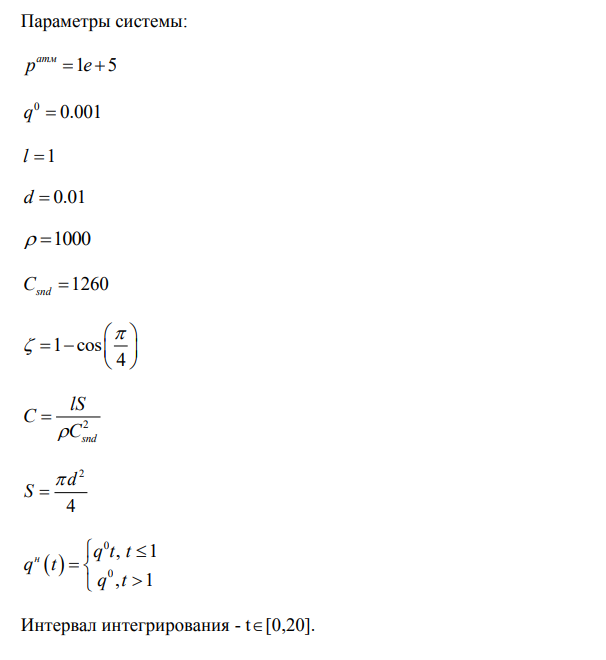
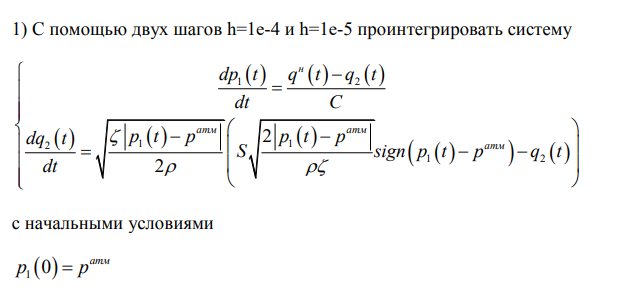
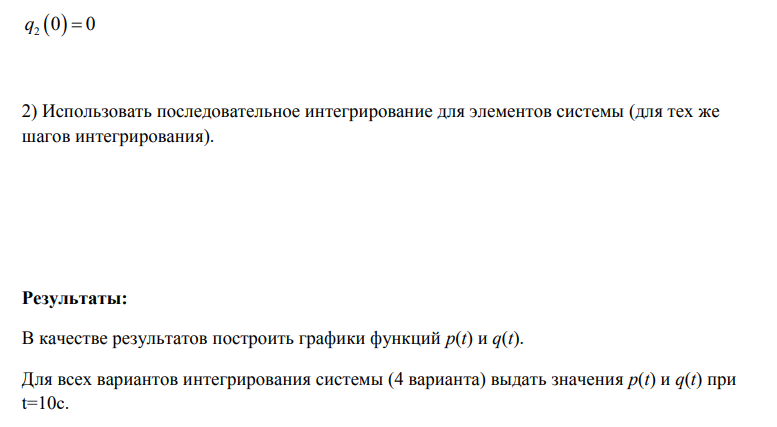
|  |  |  |
| --- | --- | --- |
|  | | |
| Федеральное государственное бюджетное  образовательное учреждение высшего образования «Новосибирский государственный технический университет» | | |
|  | | |
| Кафедра прикладной математики | | |
| Практическое задание № 5 | | |
| по дисциплине «Численное моделирование динамических систем, описываемых обыкновенными дифференциальными уравнениями» | | |
| **Системы ОДУ** | | |
|  | | |
|  |  |  |
| Группа ПМ-01 | будник светлана |
|  | самсонов семён |
|  |  |
|  |  |
| Преподаватель | ВАгин денис владимирович |
|  | Сивенкова Анастасия Павловна |
|  |  |
| Новосибирск, 2022 | | |

# Условия задачи







## Выполнение работы

1. **Прямое интегрирование**
   1. **Прямое интегрирование h = 10^-4**

|  |  |  |
| --- | --- | --- |
| t | p1(t) | q2(t) |
| 1,000000000000000E+01 | 1,237426235401580E+05 | 1,000000165196020E-03 |

* 1. **Прямое интегрирование h = 10^-5**

|  |  |  |
| --- | --- | --- |
| t | p1(t) | q2(t) |
| 1,000000000000000E+01 | 1,237425714517380E+05 | 1,000000166177060E-03 |

1. **Последовательно интегрирование**
   1. **Последовательное интегрирование h = 10^-4**

|  |  |  |
| --- | --- | --- |
| t | p1(t) | q2(t) |
| 1,000000000000000E+01 | 1,237371644013580E+05 | 9,999501714521610E-04 |

* 1. **Последовательное интегрирование h = 10^-5**

|  |  |  |
| --- | --- | --- |
| t | p1(t) | q2(t) |
| 1,000000000000000E+01 | 1,237419264740360E+05 | 9,999951724982960E-04 |

# Текст программы

class Program

{

static double y = 1;

static double t = 0;

const double h = 0.0001;

public static void Main(string[] args)

{

double a = 0.0, b = 20.0;

int n = (int)Math.Round((b - a) / h) + 1;

double[] grid\_ = new double[n];

double[] res\_p = new double[n];

double[] res\_q = new double[n];

Create\_grid(grid\_, a, b);

int part = 1;

while (part != 0)

{

Console.WriteLine("Menu:");

Console.WriteLine("Direct (1)");

Console.WriteLine("Sequential (2)");

Console.WriteLine("Exit (0)");

part = int.Parse(Console.ReadLine());

StreamWriter? file = null;

switch (part)

{

case (1):

{

Direct(grid\_, res\_p, res\_q, n);

file = new StreamWriter("Прямой.txt");

break;

}

case (2):

{

Sequential(grid\_, res\_p, res\_q, n);

file = new StreamWriter("Последовательный.txt");

break;

}

default:

break;

}

if (file == null)

{

continue;

}

int k = 0;

for (int i = 0; i < n; i += 200)

{

k++;

var str = $"{grid\_[i], 20:e15} {res\_p[i], 20:e15} {res\_q[i],20:e15}";

file.WriteLine(str);

//Console.WriteLine(str);

}

Console.WriteLine($"Количество выведенных элементов {k}");

//std::cout << std::setw(7) << grid\_[(n) / 2] << std::setw(25) << res\_p[(n) / 2]

//<< std::setw(25) << res\_q[(n) / 2] << std::endl;

file.Close();

file = null;

}

}

static void Create\_grid(double[] grid, double t\_start, double t\_end)

{

int n = grid.Length;

grid[0] = t\_start;

for (int i = 1; i < n - 1; i++)

{

grid[i] = i \* h;

}

grid[n - 1] = t\_end;

}

static void Direct(double[] grid, double[] p, double[] q, int n)

{

p[0] = 1e5;

q[0] = 0.0;

for (int i = 1; i < n; i++)

{

double k1\_p = Func\_p(grid[i - 1], q[i - 1]);

double k1\_q = Func\_q(grid[i - 1], p[i - 1], q[i - 1]);

double k2\_p = Func\_p(grid[i - 1] + h / 2, q[i - 1] + h / 2 \* k1\_q);

double k2\_q = Func\_q(grid[i - 1] + h / 2, p[i - 1] + h / 2 \* k1\_p, q[i - 1] + h / 2 \* k1\_q);

double k3\_p = Func\_p(grid[i - 1] + h / 2, q[i - 1] + h / 2 \* k2\_q);

double k3\_q = Func\_q(grid[i - 1] + h / 2, p[i - 1] + h / 2 \* k2\_p, q[i - 1] + h / 2 \* k2\_q);

double k4\_p = Func\_p(grid[i - 1] + h, q[i - 1] + h \* k3\_q);

double k4\_q = Func\_q(grid[i - 1] + h, p[i - 1] + h \* k3\_p, q[i - 1] + h \* k3\_q);

p[i] = p[i - 1] + h / 6 \* (k1\_p + 2 \* k2\_p + 2 \* k3\_p + k4\_p);

q[i] = q[i - 1] + h / 6 \* (k1\_q + 2 \* k2\_q + 2 \* k3\_q + k4\_q);

}

}

static void Sequential(double[] grid, double[] p, double[] q, int n)

{

p[0] = 1e5;

q[0] = 0;

for (int i = 1; i < n; i++)

{

double k1\_p = Func\_p(grid[i - 1], q[i - 1]);

double k2\_p = Func\_p(grid[i - 1] + h / 2, q[i - 1] + h / 2 \* q[i - 1]);

double k3\_p = Func\_p(grid[i - 1] + h / 2, q[i - 1] + h / 2 \* q[i - 1]);

double k4\_p = Func\_p(grid[i - 1] + h, q[i - 1] + h \* q[i - 1]);

p[i] = p[i - 1] + h / 6 \* (k1\_p + 2 \* k2\_p + 2 \* k3\_p + k4\_p);

double k1\_q = Func\_q(grid[i - 1], p[i], q[i - 1]);

double k2\_q = Func\_q(grid[i - 1] + h / 2, p[i], q[i - 1] + h / 2 \* k1\_q);

double k3\_q = Func\_q(grid[i - 1] + h / 2, p[i], q[i - 1] + h / 2 \* k2\_q);

double k4\_q = Func\_q(grid[i - 1] + h, p[i], q[i - 1] + h \* k3\_q);

q[i] = q[i - 1] + h / 6 \* (k1\_q + 2 \* k2\_q + 2 \* k3\_q + k4\_q);

}

}

static double Func\_p(double t, double q)

{

double l = 1;

double d = 0.01;

double r = 1000;

double Csnd = 1260;

double s = Math.PI \* d \* d / 4;

double C = l \* s / r / Csnd / Csnd;

return (q\_n(t) - q) / C;

}

static double Func\_q(double t, double p, double q)

{

double Patm = 1e5;

double q0 = 0.001;

double ksi = 1 - Math.Sqrt(2) / 2;

double l = 1;

double d = 0.01;

double r = 1000;

double Csnd = 1260;

double s = Math.PI \* d \* d / 4;

double C = l \* s / r / Csnd / Csnd;

return Math.Sqrt(ksi / 2 / r \* Math.Abs(p - Patm)) \* (s \* Math.Sqrt(2 / ksi / r \* Math.Abs(p - Patm)) \* (p - Patm > 0 ? 1 : -1) - q);

}

static double q\_n(double t)

{

double q0 = 0.001;

if (t < 1)

return q0 \* t;

else

return q0;

}

}

# Сравнение методов:

|  |  |
| --- | --- |
| Погрешность на крупном шаге прямого метода | |
| delta p1(t) | delta q2(t) |
| 0,05208842 | 9,8104E-13 |

|  |  |
| --- | --- |
| Погрешность на крупном шаге последовательного метода | |
| delta p1(t) | delta q2(t) |
| 4,762072678 | 4,5001E-08 |

# Вывод:

При сравнении погрешностей одинаковых методов между крупным и мелким шагами, получили, что погрешность на прямом методе меньше, чем на последовательном методе.