Министерство цифрового развития, связи и массовых коммуникаций Российской Федерации
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Сибирский государственный университет телекоммуникаций и информатики»
(СибГУТИ)

Институт информатики и вычислительной техники

09.03.01 "Информатика и вычислительная техника"
профиль "Электронно-вычислительные машины, комплексы, системы и сети"

Курсовая работа

по дисциплине «Вычислительная математика» на тему: Решение системы обыкновенных дифференциальных уравнений математической модели SEIR-D для Новосибирской области методом Эйлера

Выполнил:
Студент гр. ИВ-022 _____/Урянский Я. И./
ФИО студента

«29» апреля 2022 г.

Проверил:
Ассистент кафедры ПМиК _____/Агалаков А. А./
ФИО преподавателя

«__» _____ 2022 г. Оценка _____

Новосибирск 2022

Оглавление

Задание	3
Теория	3
Ход работы	7
Демонстрация работы	9
Вывод	20
Литература	20
Приложение	21

Задание

Решите систему уравнений (5) модель SEIR-D для Новосибирской области с коэффициентами из таблицы 1. Решение найдите с помощью метода Эйлера на участке времени от 0 до 90 дней с точностью до 2 знака после запятой.

Теория

В рамках модели SEIR-D распространение коронавируса COVID-19 описывается системой из 5 нелинейных обыкновенных дифференциальных уравнений на отрезке t ∈ [t0, T].

$$\begin{cases} \dfrac{dS}{dt} = -c(t-\tau)\left(\dfrac{\alpha_I S(t)I(t)}{N} + \dfrac{\alpha_E S(t)E(t)}{N}\right) + \gamma R(t), \\ \dfrac{dE}{dt} = c(t-\tau)\left(\dfrac{\alpha_I S(t)I(t)}{N} + \dfrac{\alpha_E S(t)E(t)}{N}\right) - (\kappa+\rho)E(t), \\ \dfrac{dI}{dt} = \kappa E(t) - \beta I(t) - \mu I(t), \\ \dfrac{dR}{dt} = \beta I(t) + \rho E(t) - \gamma R(t), \\ \dfrac{dD}{dt} = \mu I(t). \end{cases} \tag{5}$$

Здесь N = S + E + I + R + D — вся популяция.

Функция, использующая ограничения на передвижения граждан:

$$c(t) = 1 + c^{\text{isol}}\left(1 - \frac{2}{5}a(t)\right), \quad c(t) \in (0, 2).$$

Начальные данные:

$S(t_0) = S_0$, $E(t_0) = E_0$, $I(t_0) = I_0$, $R(t_0) = R_0$, $D(t_0) = D_0$.

$S_0 = 2\,798\,170 - q_8 - q_9$, $E_0 = q_8$, $I_0 = 0$, $R_0 = q_9$, $D_0 = 0$

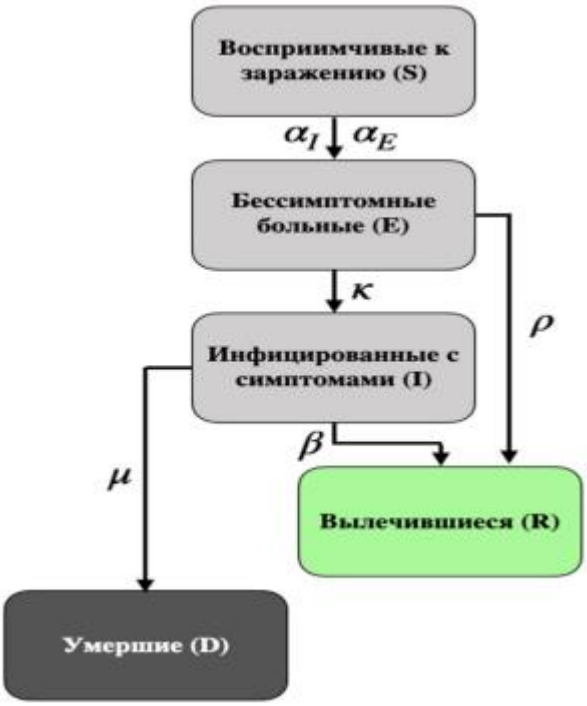

Рисунок 1. – Схема математической модели SEIR-D

Таблица 1. Восстановленные параметры для периода измерений 23.03.2020-31.05.2020, Новосибирская область

Модель	α_E	α_I	κ	ρ	β	ν	ε_{CH}	μ	c^{isol}	E_0	R_0
SIER-D	0.999	0.999	0.042	0.952	0.999	-	-	0.0188	0	99	24

Таблица 2. Описание параметров модели SEIR-D и их пределы изменения

Символ	Описание	Пределы
a(t)	Индекс самоизоляции по данным Яндекс	(0, 5)
τ	Латентный период (характеризует запаздывание выделения вирионов или заразности)	2 дня
α_I	Параметр заражения между инфицированным и восприимчивым населением, который связан с контагиозностью вируса и социальными факторами	(0, 1)
α_E	Параметр заражения между бессимптомной и восприимчивой группами населения	(0, 1)
κ	Частота появления симптомов в открытых случаях, что приводит к переходу от бессимптомной к инфицированной популяции	(0, 1)
ρ	Скорость восстановления выявленных случаев (случаи, которые выявлены, но выздоравливают без каких-либо симптомов)	(0, 1)
β	Скорость выздоровления зараженных случаев	(0, 1)
γ	Скорость повторного заражения. Этот параметр является обратной величиной уровня иммунитета вируса (0 – устойчивый иммунитет, 0.001 – вероятность повторного заражения)	0
c^{isol}	Коэффициент влияния индекса самоизоляции на заражаемость	(0, 1)
μ	Смертность в результате COVID-19	(0, 0.1)
E_0	Начальное количество бессимптомных инфицированных	(0, 600)
R_0	Начальное количество вылеченных индивидуумов	(0, 600)

Описание метода Эйлера

Дана задача Коши для ОДУ 1-го порядка:

$$\begin{cases} y' = f(x, y), \\ y(x_0) = y_0. \end{cases}$$

Нужно найти решение данного уравнения в каждой точке отрезка $[x_0, b]$.

Разобьём отрезок $[x_0, b]$ на n равных промежутков длиной $h = (b - x_0) / n$.

Получим сетку $\{x_0, x_1, \dots, x_n = b\}$ с шагом $x_i = x_0 + hi$, $i = 0, 1, \dots, n$ – узлы сетки.

Проинтегрируем данное уравнение на отрезке $[x_i, x_{i+1}]$:

$$y_{i+1} = y_i + \int_{x_i}^{x_{i+1}} f(x, y(x))dx.$$

Данное уравнение эквивалентно задаче Коши. Вычислим интеграл по квадратурной формуле прямоугольников, взяв за основной узел x_i:

$y_{i+1} = y_i + hf(x_i, y_i) + O(h^2)$, $y_0 = y(x_0)$, $i = 0, 1, \dots, n$. – Алгоритм метода Эйлера численного интегрирования задачи Коши.

Ход работы

В заголовочном файле “coefficients.h” через директиву #define определим коэффициенты и параметры модели SEIR-D. Также, добавив препроцессорную директиву #pragma once, сообщим компилятору, чтобы данный заголовочный файл при компиляции подключался строго один раз.

В заголовочном файле “functions.h” через директиву #include подключим библиотеку cstdio. Данная библиотека нужна будет для работы с файлом, в который будем записывать результат работы программы. Также, через директиву #include, подключаем заголовочный файл “coefficients.h” с параметрами и коэффициентами модели SEIR-D. Аналогично, как и с файлом “coefficients.h”, добавим препроцессорную директиву #pragma once.

В данном заголовочном файле (functions.h) “запишем” уравнения системы нелинейных ОДУ в виде кода на языке программирования C++. Т.е. мы представим данные уравнения в виде функций, которые возвращают значение типа double, при указанных параметрах. Например, для уравнения $\frac{dS}{dt} = -c(t-\tau)\left(\frac{\alpha_I S(t)I(t)}{N} + \frac{\alpha_E S(t)E(t)}{N}\right) + \gamma R(t)$ код на C++ будет иметь вид:

```
double function_s(double S, double E, double I, double R, double
N) {
    return -1 * function_c(c_isol) * ((alpha_I * S * I) / N  +
(alpha_E * S * E) / N) + gamma * R;
}
```

Также, в этом же заголовочном файле определим функцию double euler_method(double t0, double T, double h). Данная функция является методом решения заданной системы ОДУ. Алгоритм, описанный в данной функции, следующий:

Открываем файл “out.txt” на запись. Определяем переменную int n. Формула, по которой определяем данную переменную: (T – t0) / h + 1, где T – конец отрезка, t0 – начало отрезка, h – шаг. Увеличиваем на 1, потому что в дальнейшем мы будем определять массивы, и первым элементом массива

будут начальные значения. Объявляем массивы S, E, I, R, D, N, t типа double и размера n. Заполняем первые элементы массивов начальными значениями. Затем в файл “out.txt” записываем данные значения. Следующим действием определяем цикл for(int i = 1; i < n; i++). В теле цикла записываем в i-е “ячейки” массивов посчитанные по формулам значения. Для массива t справедлива формула: $t_0 + ih$. Для массива S: $S_{i-1} + hf_S(S_{i-1} + h / 2 * f_S(S_{i-1}, E_{i-1}, I_{i-1}, R_{i-1}, N_{i-1}), E_{i-1}, I_{i-1}, R_{i-1}, N_{i-1})$. Для массива E: $E_{i-1} + hf_E(S_{i-1}, E_{i-1} + h / 2 * f_E(S_{i-1}, E_{i-1}, I_{i-1}, N_{i-1}), I_{i-1}, N_{i-1})$. Для массива I: $I_{i-1} + hf_I(E_{i-1}, I_{i-1} + h / 2 * f_I(E_{i-1}, I_{i-1}))$. Для массива R: $R_{i-1} + hf_R(E_{i-1}, I_{i-1}, R_{i-1} + h / 2 * f_R(E_{i-1}, I_{i-1}, R_{i-1}))$. Для массива D: $D_{i-1} + hf_D(I_{i-1})$. Для массива N: $S_i + E_i + I_i + R_i + D_i$. После данных операций, записываем текущее значение i-й “ячейки” массивов в файл “out.txt”. Закрываем файл “out.txt” и высвобождаем память из-под массивов, которую выделили во время их объявления.

В файле “main.cpp” через директиву #include подключаем библиотеку iostream, она же “библиотека потока ввода-вывода”. Также, через ту же самую директиву подключаем файл “functions.h”. Далее, подключаем пространство имен std. В самой же функции int main() определяем переменные t0, T, h, выводим сообщение об этих входных данных и о том, куда будут записаны выходные данные, затем вызываем функцию euler_method(t0, T, h) и завершаем работу функции int main() с кодом 0.

Демонстрация работы

```
yan@DESKTOP-HVFJFTN:~/Univer/VM/coursework$ g++ -o app main.cpp
yan@DESKTOP-HVFJFTN:~/Univer/VM/coursework$ ./app
Входные данные: t0 = 0, T = 90, h = 1
Выходные данные записаны в файл 'out.txt'
```

Рисунок 2. – Результат работы программы

```
 1   0.000000 2798170.00 2798047.00 99.00 0.00 24.00 0.00
 2   1.000000 2798167.89 2797948.11 99.49 2.04 118.25 0.00
 3   2.000000 2798166.83 2797846.68 102.03 3.07 215.00 0.04
 4   3.000000 2798166.25 2797741.70 105.60 3.64 315.20 0.10
 5   4.000000 2798165.89 2797632.58 109.76 4.00 419.38 0.16
 6   5.000000 2798165.63 2797518.96 114.30 4.26 527.87 0.24
 7   6.000000 2798165.41 2797400.54 119.11 4.49 640.94 0.32
 8   7.000000 2798165.20 2797277.10 124.17 4.70 758.82 0.40
 9   8.000000 2798165.00 2797148.39 129.46 4.91 881.74 0.49
10   9.000000 2798164.79 2797014.20 134.98 5.13 1009.89 0.59
11   10.000000 2798164.58 2796874.29 140.74 5.35 1143.52 0.68
12   11.000000 2798164.36 2796728.42 146.73 5.58 1282.85 0.78
13   12.000000 2798164.13 2796576.35 152.97 5.82 1428.11 0.89
14   13.000000 2798163.89 2796417.81 159.47 6.06 1579.55 1.00
15   14.000000 2798163.65 2796252.55 166.24 6.32 1737.42 1.11
16   15.000000 2798163.39 2796080.28 173.28 6.59 1902.00 1.23
17   16.000000 2798163.12 2795900.73 180.62 6.87 2073.55 1.35
18   17.000000 2798162.84 2795713.59 188.25 7.16 2252.36 1.48
19   18.000000 2798162.55 2795518.55 196.19 7.46 2438.73 1.62
20   19.000000 2798162.25 2795315.30 204.45 7.78 2632.96 1.76
21   20.000000 2798161.93 2795103.50 213.05 8.11 2835.37 1.90
22   21.000000 2798161.60 2794882.81 221.99 8.45 3046.30 2.06
23   22.000000 2798161.26 2794652.88 231.29 8.81 3266.07 2.21
24   23.000000 2798160.91 2794413.34 240.96 9.17 3495.06 2.38
25   24.000000 2798160.54 2794163.80 251.01 9.56 3733.62 2.55
26   25.000000 2798160.16 2793903.87 261.47 9.96 3982.13 2.73
27   26.000000 2798159.76 2793633.14 272.33 10.37 4241.00 2.92
28   27.000000 2798159.34 2793351.19 283.62 10.81 4510.62 3.11
29   28.000000 2798158.91 2793057.59 295.34 11.25 4791.41 3.32
30   29.000000 2798158.47 2792751.87 307.52 11.72 5083.82 3.53
```

Рисунок 3.1. – Результат работы программы (файл “out.txt”)

```
31    30.000000 2798158.00 2792433.58 320.17 12.21 5388.29 3.75
32    31.000000 2798157.52 2792102.24 333.31 12.71 5705.29 3.98
33    32.000000 2798157.02 2791757.34 346.94 13.23 6035.29 4.22
34    33.000000 2798156.50 2791398.37 361.09 13.77 6378.80 4.47
35    34.000000 2798155.96 2791024.81 375.77 14.34 6736.31 4.73
36    35.000000 2798155.40 2790636.12 391.00 14.92 7108.37 5.00
37    36.000000 2798154.82 2790231.72 406.79 15.53 7495.50 5.28
38    37.000000 2798154.22 2789811.06 423.16 16.16 7898.28 5.57
39    38.000000 2798153.59 2789373.53 440.12 16.81 8317.26 5.87
40    39.000000 2798152.94 2788918.53 457.70 17.49 8753.05 6.19
41    40.000000 2798152.27 2788445.43 475.90 18.19 9206.24 6.52
42    41.000000 2798151.58 2787953.59 494.75 18.91 9677.47 6.86
43    42.000000 2798150.86 2787442.36 514.26 19.66 10167.37 7.21
44    43.000000 2798150.12 2786911.06 534.45 20.44 10676.59 7.58
45    44.000000 2798149.35 2786359.00 555.33 21.25 11205.80 7.97
46    45.000000 2798148.55 2785785.49 576.91 22.08 11755.70 8.37
47    46.000000 2798147.73 2785189.81 599.22 22.94 12326.98 8.78
48    47.000000 2798146.88 2784571.21 622.26 23.84 12920.35 9.21
49    48.000000 2798146.00 2783928.97 646.05 24.76 13536.56 9.66
50    49.000000 2798145.09 2783262.32 670.61 25.71 14176.33 10.13
51    50.000000 2798144.15 2782570.48 695.94 26.69 14840.43 10.61
52    51.000000 2798143.19 2781852.69 722.06 27.70 15529.63 11.11
53    52.000000 2798142.19 2781108.14 748.97 28.75 16244.70 11.63
54    53.000000 2798141.17 2780336.03 776.69 29.83 16986.45 12.17
55    54.000000 2798140.11 2779535.56 805.23 30.94 17755.66 12.73
56    55.000000 2798139.03 2778705.90 834.58 32.08 18553.14 13.32
57    56.000000 2798137.91 2777846.25 864.77 33.26 19379.72 13.92
58    57.000000 2798136.76 2776955.77 895.78 34.47 20236.20 14.54
59    58.000000 2798135.58 2776033.64 927.62 35.72 21123.42 15.19
```

Рисунок 3.2. – Результат работы программы (файл “out.txt”)

```
60   59.000000 2798134.38 2775079.03 960.29 37.00 22042.19 15.86
61   60.000000 2798133.14 2774091.12 993.79 38.31 22993.35 16.56
62   61.000000 2798131.87 2773069.10 1028.11 39.66 23977.71 17.28
63   62.000000 2798130.57 2772012.15 1063.25 41.04 24996.09 18.03
64   63.000000 2798129.24 2770919.48 1099.19 42.46 26049.31 18.80
65   64.000000 2798127.88 2769790.29 1135.93 43.91 27138.15 19.60
66   65.000000 2798126.49 2768623.82 1173.44 45.39 28263.42 20.42
67   66.000000 2798125.08 2767419.31 1211.70 46.91 29425.88 21.27
68   67.000000 2798123.63 2766176.04 1250.71 48.45 30626.28 22.16
69   68.000000 2798122.17 2764893.29 1290.42 50.03 31865.36 23.07
70   69.000000 2798120.68 2763570.40 1330.81 51.64 33143.82 24.01
71   70.000000 2798119.17 2762206.72 1371.85 53.28 34462.34 24.98
72   71.000000 2798117.64 2760801.65 1413.50 54.94 35821.57 25.98
73   72.000000 2798116.09 2759354.62 1455.71 56.64 37222.11 27.01
74   73.000000 2798114.52 2757865.11 1498.45 58.35 38664.53 28.08
75   74.000000 2798112.94 2756332.66 1541.67 60.09 40149.35 29.17
76   75.000000 2798111.35 2754756.84 1585.30 61.85 41677.05 30.30
77   76.000000 2798109.75 2753137.31 1629.29 63.64 43248.04 31.47
78   77.000000 2798108.14 2751473.77 1673.58 65.43 44862.69 32.66
79   78.000000 2798106.53 2749765.98 1718.10 67.25 46521.31 33.89
80   79.000000 2798104.92 2748013.79 1762.78 69.07 48224.12 35.16
81   80.000000 2798103.32 2746217.12 1807.55 70.91 49971.29 36.46
82   81.000000 2798101.72 2744375.96 1852.32 72.75 51762.91 37.79
83   82.000000 2798100.14 2742490.39 1897.01 74.59 53598.99 39.16
84   83.000000 2798098.58 2740560.59 1941.53 76.44 55479.46 40.56
85   84.000000 2798097.03 2738586.80 1985.80 78.28 57404.16 42.00
86   85.000000 2798095.52 2736569.39 2029.71 80.11 59372.84 43.47
87   86.000000 2798094.03 2734508.80 2073.17 81.93 61385.15 44.97
88   87.000000 2798092.59 2732405.59 2116.08 83.74 63440.66 46.51
```

Рисунок 3.3. – Результат работы программы (файл “out.txt”)

```
89   88.000000 2798091.18 2730260.40 2158.33 85.53 65538.83 48.09
90   89.000000 2798089.82 2728073.99 2199.83 87.30 67679.01 49.70
91   90.000000 2798088.52 2725847.24 2240.45 89.04 69860.45 51.34
```

Рисунок 3.4. – Результат работы программы (файл “out.txt”)

Таблица 3. Результат работы программы

t	N	S	E	I	R	D
0	2798170.00	2798047.00	99.00	0.00	24.00	0.00
1	2798167.89	2797948.11	99.49	2.04	118.25	0.00
2	2798166.83	2797846.68	102.03	3.07	215.00	0.04
3	2798166.25	2797741.70	105.60	3.64	315.20	0.10
4	2798165.89	2797632.58	109.76	4.00	419.38	0.16
5	2798165.63	2797518.96	114.30	4.26	527.87	0.24
6	2798165.41	2797400.54	119.11	4.49	640.94	0.32
7	2798165.20	2797277.10	124.17	4.70	758.82	0.40
8	2798165.00	2797148.39	129.46	4.91	881.74	0.49
9	2798164.79	2797014.20	134.98	5.13	1009.89	0.59
10	2798164.58	2796874.29	140.74	5.35	1143.52	0.68
11	2798164.36	2796728.42	146.73	5.58	1282.85	0.78
12	2798164.13	2796576.35	152.97	5.82	1428.11	0.89
13	2798163.89	2796417.81	159.47	6.06	1579.55	1.00
14	2798163.65	2796252.55	166.24	6.32	1737.42	1.11
15	2798163.39	2796080.28	173.28	6.59	1902.00	1.23
16	2798163.12	2795900.73	180.62	6.87	2073.55	1.35
17	2798162.84	2795713.59	188.25	7.16	2252.36	1.48
18	2798162.55	2795518.55	196.19	7.46	2438.73	1.62
19	2798162.25	2795315.30	204.45	7.78	2632.96	1.76
20	2798161.93	2795103.50	213.05	8.11	2835.37	1.90
21	2798161.60	2794882.81	221.99	8.45	3046.30	2.06
22	2798161.26	2794652.88	231.29	8.81	3266.07	2.21
23	2798160.91	2794413.34	240.96	9.17	3495.06	2.38
24	2798160.54	2794163.80	251.01	9.56	3733.62	2.55
25	2798160.16	2793903.87	261.47	9.96	3982.13	2.73
26	2798159.76	2793633.14	272.33	10.37	4241.00	2.92
27	2798159.34	2793351.19	283.62	10.81	4510.62	3.11
28	2798158.91	2793057.59	295.34	11.25	4791.41	3.32
29	2798158.47	2792751.87	307.52	11.72	5083.82	3.53
30	2798158.00	2792433.58	320.17	12.21	5388.29	3.75
31	2798157.52	2792102.24	333.31	12.71	5705.29	3.98
32	2798157.02	2791757.34	346.94	13.23	6035.29	4.22
33	2798156.50	2791398.37	361.09	13.77	6378.80	4.47
34	2798155.96	2791024.81	375.77	14.34	6736.31	4.73
35	2798155.40	2790636.12	391.00	14.92	7108.37	5.00
36	2798154.82	2790231.72	406.79	15.53	7495.50	5.28
37	2798154.22	2789811.06	423.16	16.16	7898.28	5.57
38	2798153.59	2789373.53	440.12	16.81	8317.26	5.87
39	2798152.94	2788918.53	457.70	17.49	8753.05	6.19
40	2798152.27	2788445.43	475.90	18.19	9206.24	6.52
41	2798151.58	2787953.59	494.75	18.91	9677.47	6.86
42	2798150.86	2787442.36	514.26	19.66	10167.37	7.21
43	2798150.12	2786911.06	534.45	20.44	10676.59	7.58
44	2798149.35	2786359.00	555.33	21.25	11205.80	7.97
45	2798148.55	2785785.49	576.91	22.08	11755.70	8.37
46	2798147.73	2785189.81	599.22	22.94	12326.98	8.78
47	2798146.88	2784571.21	622.26	23.84	12920.35	9.21
48	2798146.00	2783928.97	646.05	24.76	13536.56	9.66
49	2798145.09	2783262.32	670.61	25.71	14176.33	10.13

50	2798144.15	2782570.48	695.94	26.69	14840.43	10.61
51	2798143.19	2781852.69	722.06	27.70	15529.63	11.11
52	2798142.19	2781108.14	748.97	28.75	16244.70	11.63
53	2798141.17	2780336.03	776.69	29.83	16986.45	12.17
54	2798140.11	2779535.56	805.23	30.94	17755.66	12.73
55	2798139.03	2778705.90	834.58	32.08	18553.14	13.32
56	2798137.91	2777846.25	864.77	33.26	19379.72	13.92
57	2798136.76	2776955.77	895.78	34.47	20236.20	14.54
58	2798135.58	2776033.64	927.62	35.72	21123.42	15.19
59	2798134.38	2775079.03	960.29	37.00	22042.19	15.86
60	2798133.14	2774091.12	993.79	38.31	22993.35	16.56
61	2798131.87	2773069.10	1028.11	39.66	23977.71	17.28
62	2798130.57	2772012.15	1063.25	41.04	24996.09	18.03
63	2798129.24	2770919.48	1099.19	42.46	26049.31	18.80
64	2798127.88	2769790.29	1135.93	43.91	27138.15	19.60
65	2798126.49	2768623.82	1173.44	45.39	28263.42	20.42
66	2798125.08	2767419.31	1211.70	46.91	29425.88	21.27
67	2798123.63	2766176.04	1250.71	48.45	30626.28	22.16
68	2798122.17	2764893.29	1290.42	50.03	31865.36	23.07
69	2798120.68	2763570.40	1330.81	51.64	33143.82	24.01
70	2798119.17	2762206.72	1371.85	53.28	34462.34	24.98
71	2798117.64	2760801.65	1413.50	54.94	35821.57	25.98
72	2798116.09	2759354.62	1455.71	56.64	37222.11	27.01
73	2798114.52	2757865.11	1498.45	58.35	38664.53	28.08
74	2798112.94	2756332.66	1541.67	60.09	40149.35	29.17
75	2798111.35	2754756.84	1585.30	61.85	41677.05	30.30
76	2798109.75	2753137.31	1629.29	63.64	43248.04	31.47
77	2798108.14	2751473.77	1673.58	65.43	44862.69	32.66
78	2798106.53	2749765.98	1718.10	67.25	46521.31	33.89
79	2798104.92	2748013.79	1762.78	69.07	48224.12	35.16
80	2798103.32	2746217.12	1807.55	70.91	49971.29	36.46
81	2798101.72	2744375.96	1852.32	72.75	51762.91	37.79
82	2798100.14	2742490.39	1897.01	74.59	53598.99	39.16
83	2798098.58	2740560.59	1941.53	76.44	55479.46	40.56
84	2798097.03	2738586.80	1985.80	78.28	57404.16	42.00
85	2798095.52	2736569.39	2029.71	80.11	59372.84	43.47
86	2798094.03	2734508.80	2073.17	81.93	61385.15	44.97
87	2798092.59	2732405.59	2116.08	83.74	63440.66	46.51
88	2798091.18	2730260.40	2158.33	85.53	65538.83	48.09
89	2798089.82	2728073.99	2199.83	87.30	67679.01	49.70
90	2798088.52	2725847.24	2240.45	89.04	69860.45	51.34

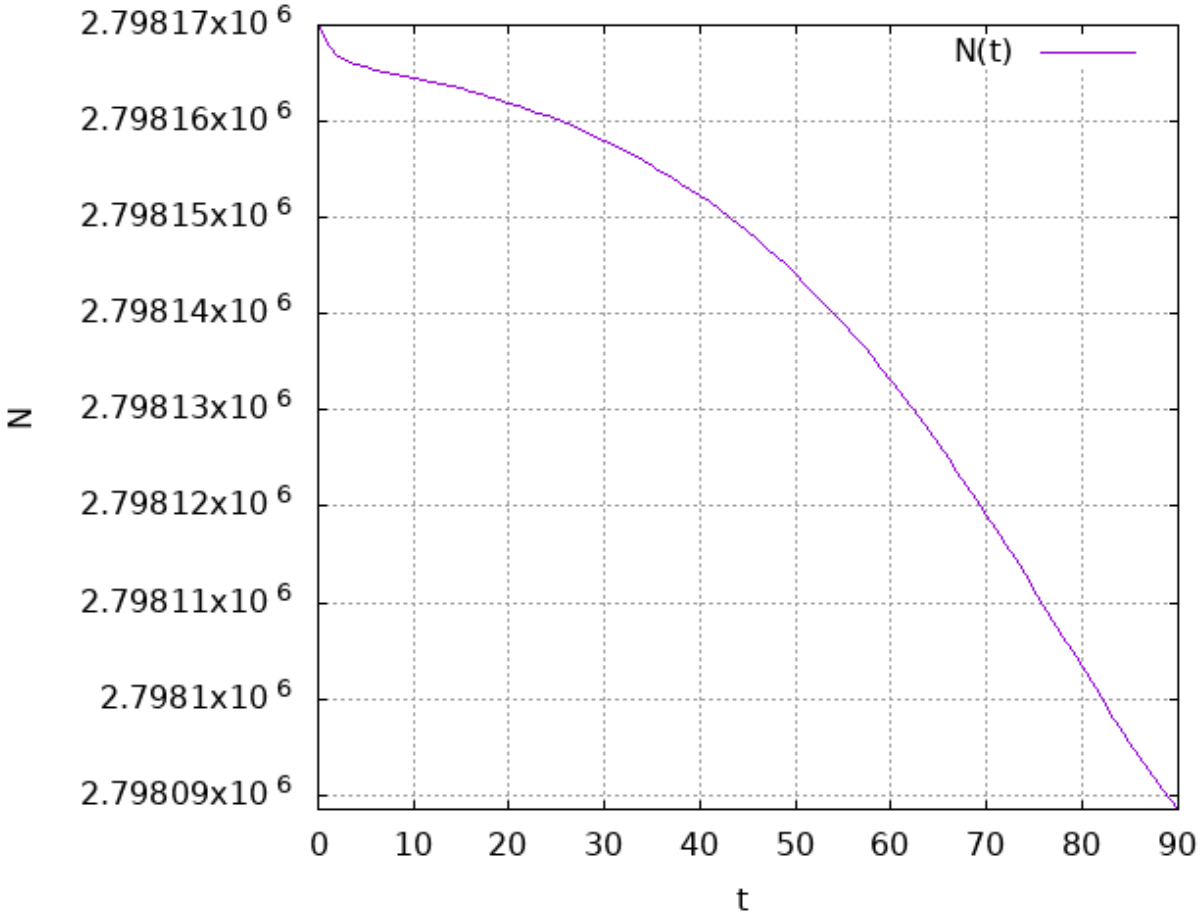

Рисунок 4. – График зависимости популяции от номера дня: N(t)

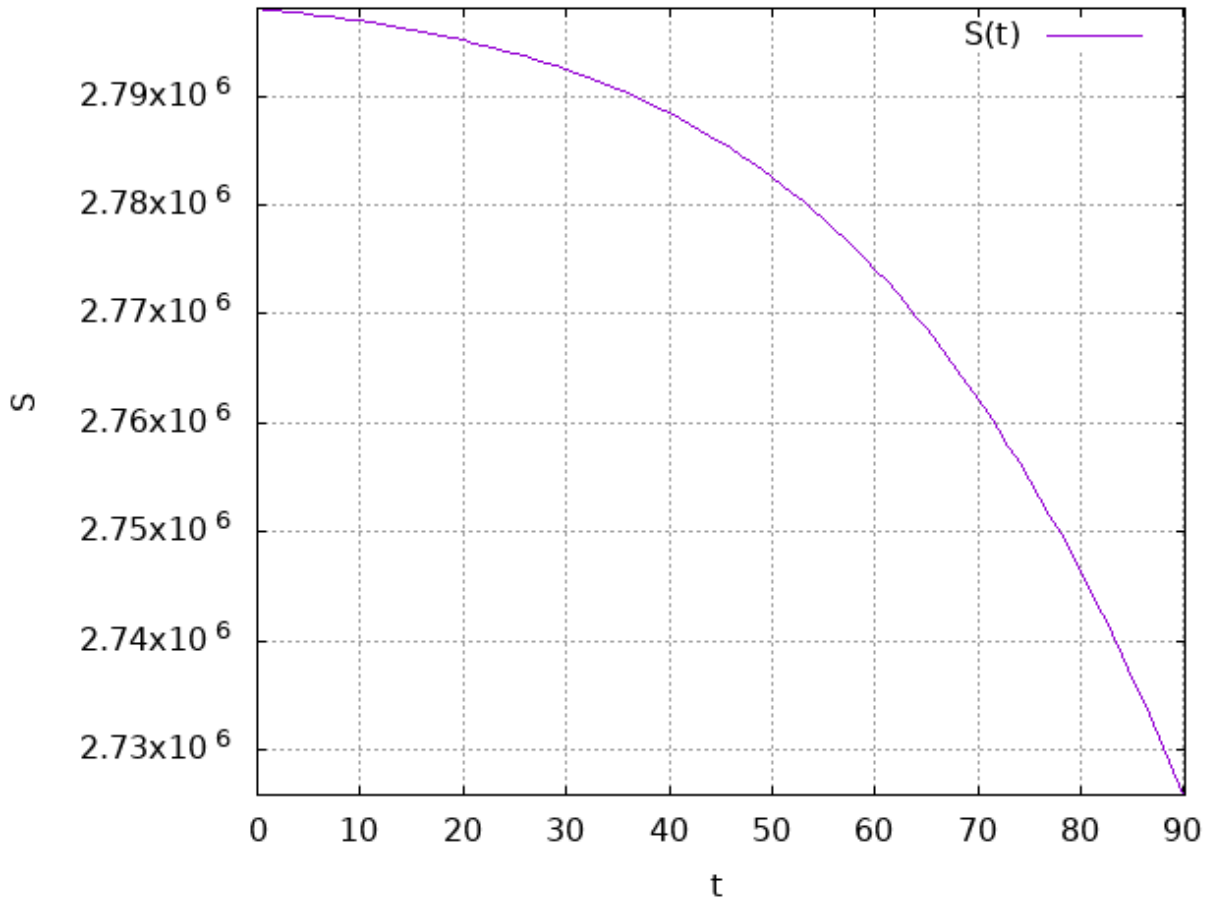

Рисунок 5. – График зависимости количества восприимчивых (незараженных) индивидуумов с 3 лет от номера дня: S(t)

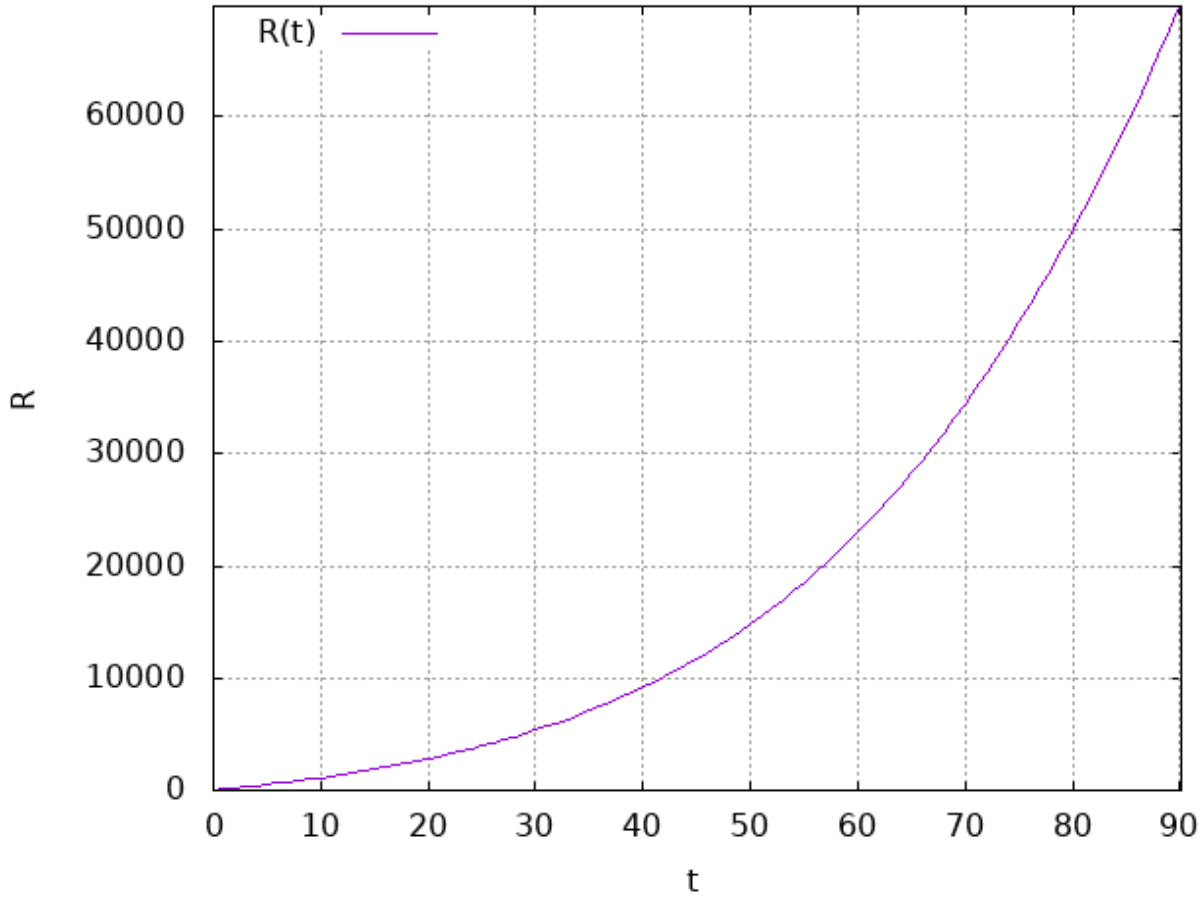

Рисунок 6. – График зависимости количества вылеченных людей от номера дня: R(t)

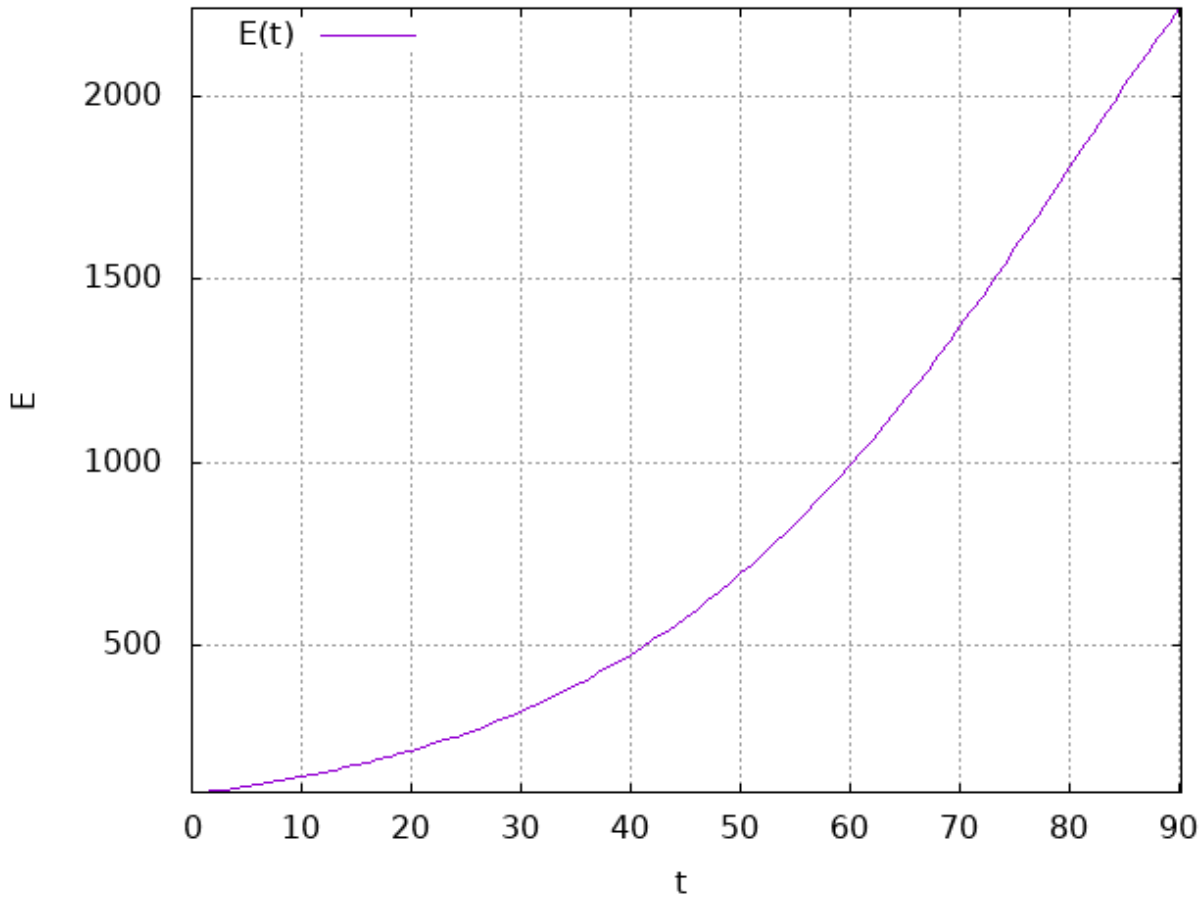

Рисунок 7. – График зависимости количества зараженных или находящихся в инкубационном периоде индивидуумов от номера дня: E(t)

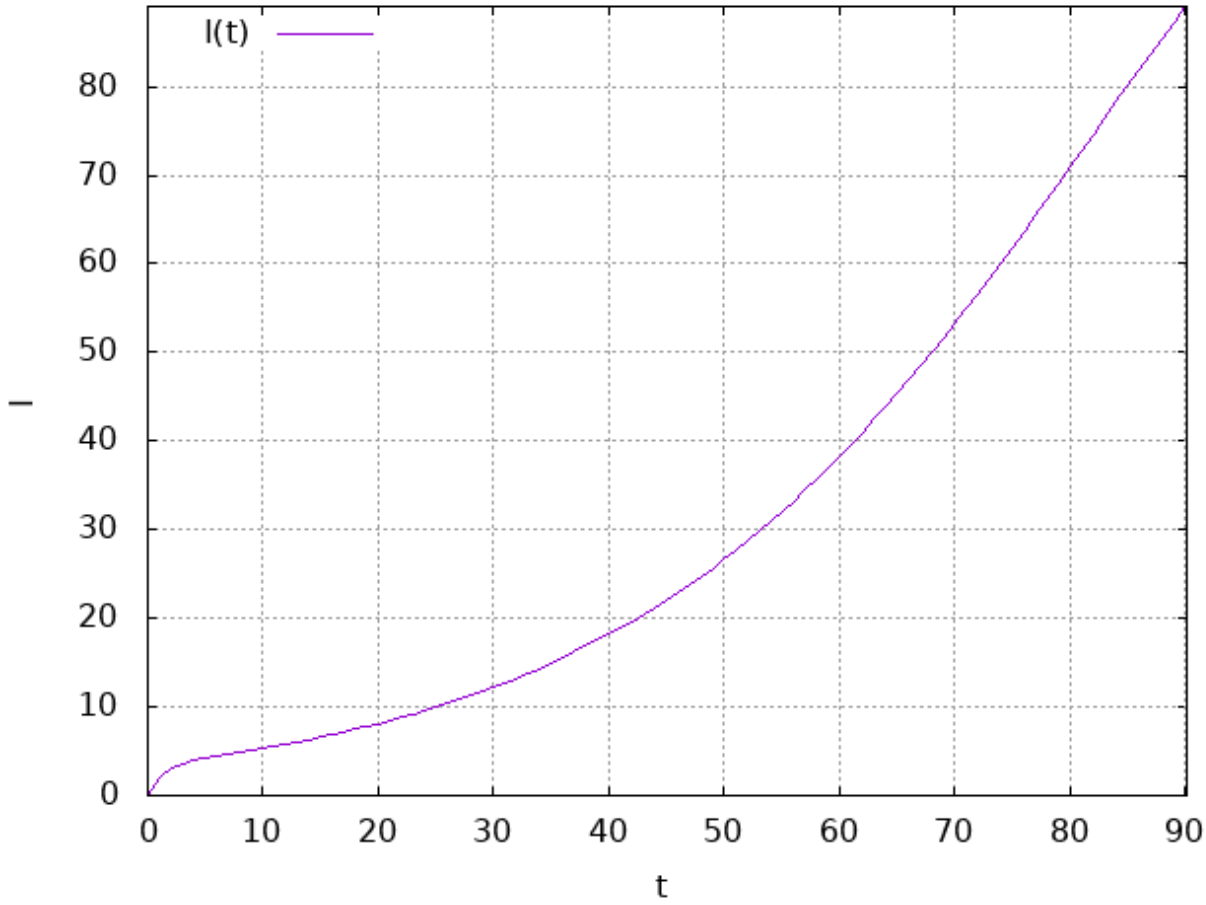

Рисунок 8. – График зависимости количества инфицированных с симптомами от номера дня: I(t)

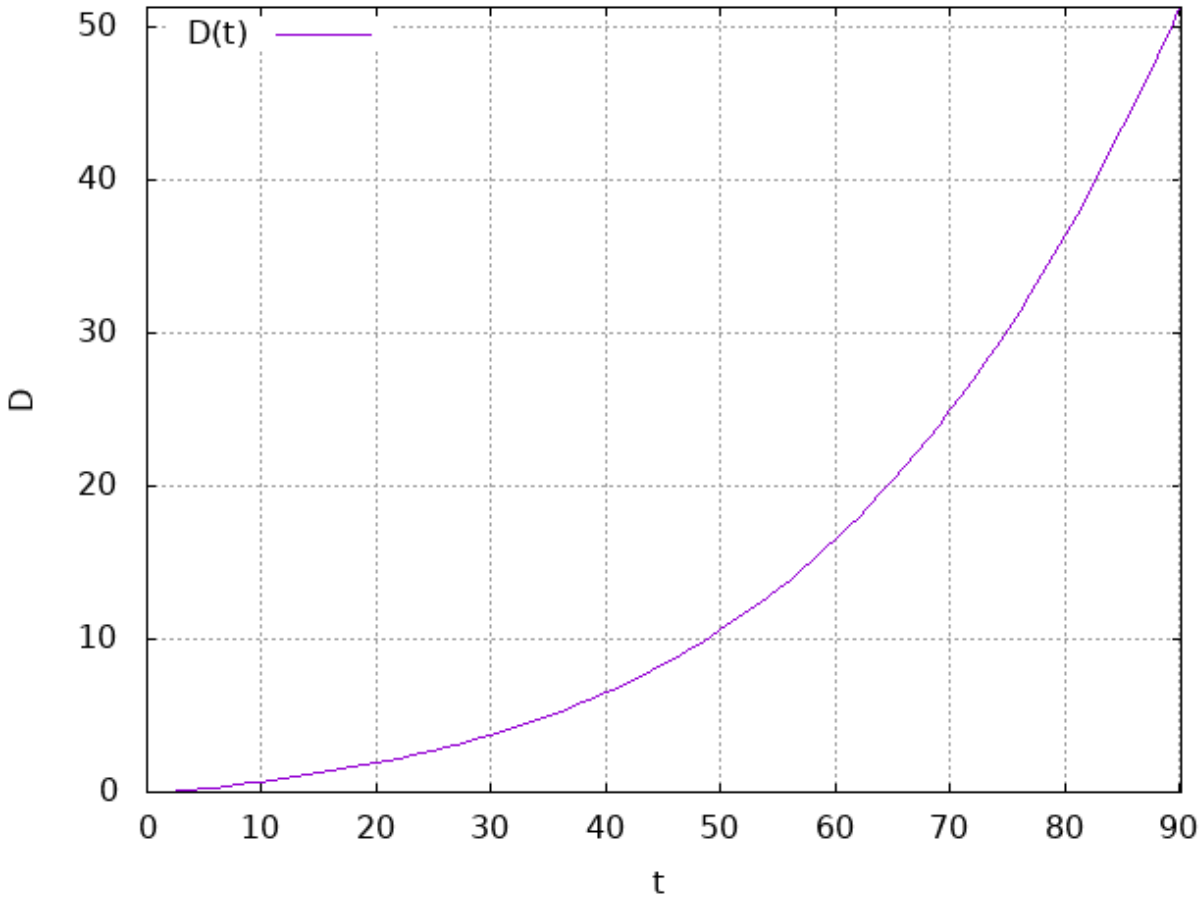

Рисунок 9. – График зависимости количества умерших от номера дня: D(t)

Вывод

В ходе выполнения работы были получены навыки решения обыкновенных дифференциальных уравнений методом Эйлера. Также была изучена математическая модель распространения эпидемий SEIR-D. Написана программа на языке C++, решающая систему обыкновенных дифференциальных уравнений математической модели SEIR-D методом Эйлера.

Результатом работы является таблица с решением системы ОДУ модели SEIR-D на участке времени от 0 до 90 дней с точностью до 2 знака после запятой. Также построены 6 графиков зависимостей переменных S, E, I, R, D, N от переменной t. Построение графиков выполнено в свободной программе “gnuplot”.

Литература

1. Лисейкин В. Д., Паасонен В. И. Высокоточные разностные схемы и адаптивные сетки для решения дифференциальных уравнений. ИВТ СО РАН, Учебное пособие, Новосибирск 2018. 35 - 45 с.
2. Лукинов В. Л. Численные методы решения ОДУ [Электронный ресурс] [сайт]. [2022]. URL: https://eios.sibsutis.ru/pluginfile.php/176360/mod_resource/content/2/%D0%A0%D0%B5%D1%88%D0%B5%D0%BD%D0%B8%D0%B5_%D0%94%D0%A3.pdf (дата обращения: 28.04.2022).
3. Kunal Menda, Lucas Laird, Mykel J. Kochenderfer, Rajmonda S. Caceres Explaining COVID-19 outbreaks with reactive SEIRD models [Электронный ресурс] [сайт]. [2021]. URL: https://www.nature.com/articles/s41598-021-97260-0 (дата обращения: 29.04.2022).

Приложение

Файл “coefficients.h”:

```
#pragma once

#define alpha_E 0.999
#define alpha_I 0.999
#define k 0.042
#define ro 0.952
#define betta 0.999
#define mu 0.0188
#define c_isol 0
#define gamma 0
#define E0 99
#define R0 24
#define I0 0
#define D0 0
```

Файл “functions.h”:

```
#pragma once

#include <cstdio>
#include <iostream>
#include "coefficients.h"

double function_a(double t) {
    return t;
}

double function_c(double t) {
    return 1 + c_isol * (1 - 2 / 5 * function_a(t));
}
```

```
double function_s(double S, double E, double I, double R, double
N) {
    return -1 * function_c(c_isol) * ((alpha_I * S * I) / N  +
(alpha_E * S * E) / N) + gamma * R;
}

double function_e(double S, double E, double I, double N) {
    return function_c(c_isol) * ((alpha_I * S * I) / N + (alpha_E
* S * E) / N) - (k + ro) * E;
}

double function_i(double E, double I) {
    return k * E - betta * I - mu * I;
}

double function_r(double E, double I, double R) {
    return betta * I + ro * E - gamma * R;
}

double function_d(double I) {
    return mu * I;
}

void euler_method(double t0, double T, double h) {
    FILE *out = fopen("out.txt", "w"); // Открываем файл out.txt
на запись
    int n = (T - t0) / h + 1; // Количество промежутков
    double *S = new double[n]; // Восприимчивые (незараженные)
индивидуумы с 3 лет;
    double *E = new double[n]; // Зараженные или находящиеся в
инкубационном периоде индивидуумы;
    double *I = new double[n]; // Инфицированные с симптомами;
    double *R = new double[n]; // Вылеченные;
    double *D = new double[n]; // Умершие;
```

```
    double *N = new double[n]; // Вся популяция;
    double *t = new double[n]; // Номер дня.

    /* Начальные значения для массивов: */
    E[0] = E0;
    R[0] = R0;
    S[0] = 2798170 - E[0] - R[0];
    I[0] = I0;
    D[0] = D0;
    N[0] = S[0] + E[0] + I[0] + R[0] + D[0];
    t[0] = t0;

    fprintf(out, "%lf %.2lf %.2lf %.2lf %.2lf %.2lf %.2lf\n", t[0], N[0], S[0], E[0], I[0], R[0], D[0]); // Запись в файл начальных значений

    /* Высчитываем значения для i-х ячеек массивов */
    for (int i = 1; i < n; i++) {
        t[i] = t0 + i * h;
        S[i] = S[i - 1] + h * function_s(S[i - 1] + h / 2 * function_s(S[i - 1], E[i - 1], I[i - 1], R[i - 1], N[i - 1]), E[i - 1], I[i - 1], R[i - 1], N[i - 1]);
        E[i] = E[i - 1] + h * function_e(S[i - 1], E[i - 1] + h / 2 * function_e(S[i - 1], E[i - 1], I[i - 1], N[i - 1]), I[i - 1], N[i - 1]);
        I[i] = I[i - 1] + h * function_i(E[i - 1], I[i - 1] + h / 2 * function_i(E[i - 1], I[i - 1]));
        R[i] = R[i - 1] + h * function_r(E[i - 1], I[i - 1], R[i - 1] + h / 2 * function_r(E[i - 1], I[i - 1], R[i - 1]));
        D[i] = D[i - 1] + h * function_d(I[i - 1]);
        N[i] = S[i] + E[i] + I[i] + R[i] + D[i];
```

```
        fprintf(out, "%lf %.2lf %.2lf %.2lf %.2lf %.2lf %.2lf\n",
t[i], N[i], S[i], E[i], I[i], R[i], D[i]); // Запись в файл
высчитанных значений
    }

    fclose(out); // Закрываем файл out.txt

    /* Высвобождаем память, которую выделили для массивов */
    delete[] S;
    delete[] E;
    delete[] I;
    delete[] R;
    delete[] D;
    delete[] N;
    delete[] t;
}
```

Файл “main.cpp”:

```
#include <iostream>
#include "functions.h"

using namespace std;

int main() {
    double t0 = 0; // Начало отрезка
    double T = 90; // Конец отрезка
    double h = 1; // Шаг

    cout << "Входные данные: t0 = " << t0 << ", T = " << T << ",
h = " << h << endl;
    cout << "Выходные данные записаны в файл 'out.txt'" << endl;

    euler_method(t0, T, h);
```

```
    return 0;
}
```