Кафедра №

ОТЧЁТ ПО ПРАКТИКЕ
ЗАЩИЩЁН С ОЦЕНКОЙ

РУКОВОДИТЕЛЬ

должность, уч. степень, звание	подпись, дата	инициалы, фамилия

ОТЧЁТ ПО ПРАКТИКЕ

вид практики Учебная

тип практики Ознакомительная

на тему индивидуального задания Реализация численных методов в среде MATLAB

выполнен

фамилия, имя, отчество обучающегося в творительном падеже

	код	наименование направления
по направлению подготовки	09.03.04	Программная инженерия
направленности	01	Разработка программно-информационных систем
	код	наименование направленности

	номер	подпись, дата	инициалы, фамилия
Обучающийся группы №			

Санкт–Петербург 2022

Содержание:

1.1 Раздел 1. Постановка задачи.

Требуется решить уравнение $at^3 + bt^2 + ct - d = 0$ методом простых итераций. Для отделения корней использовать аналитический метод.

№ варианта	a	b	c	d
17	6	1	1	1.8

Согласно варианту a = 6, b = 1, c = 1, d = 1,8.

$$6 * t3 + t2 + t - 1.8 = 0$$

Постановка задачи:

Пусть функция y=f(x) непрерывна на промежутке [a; b]. Требуется найти корни, такие, что f(x)=0. Для решения этой задачи требуется использовать метод простых итераций, при котором

График:

входными данными будут являться:

a, b, c, d – коэффициенты

A, B – начало и конец отрезка

ε - точность вычислений;

Выходные данные:

x – корень уравнения

1.2Суть метода численной реализации:

Метод простых итераций заключается в том, что функцию преобразуют к виду: **x = φ(x)**. Задается интервал, в котором и будут производиться расчеты и начальное приближение. Процесс строится по схеме $\mathbf{x_n = f(x_{n-1})}$. Вычисления оканчивается, как только выполнится условие: $|x_n - x_{n-1}| \leq \varepsilon$, то есть, как только будет достигнута заданная точность. [1]

1.3Описание ручного счета тестового примера и аналитический метод отделения корней:

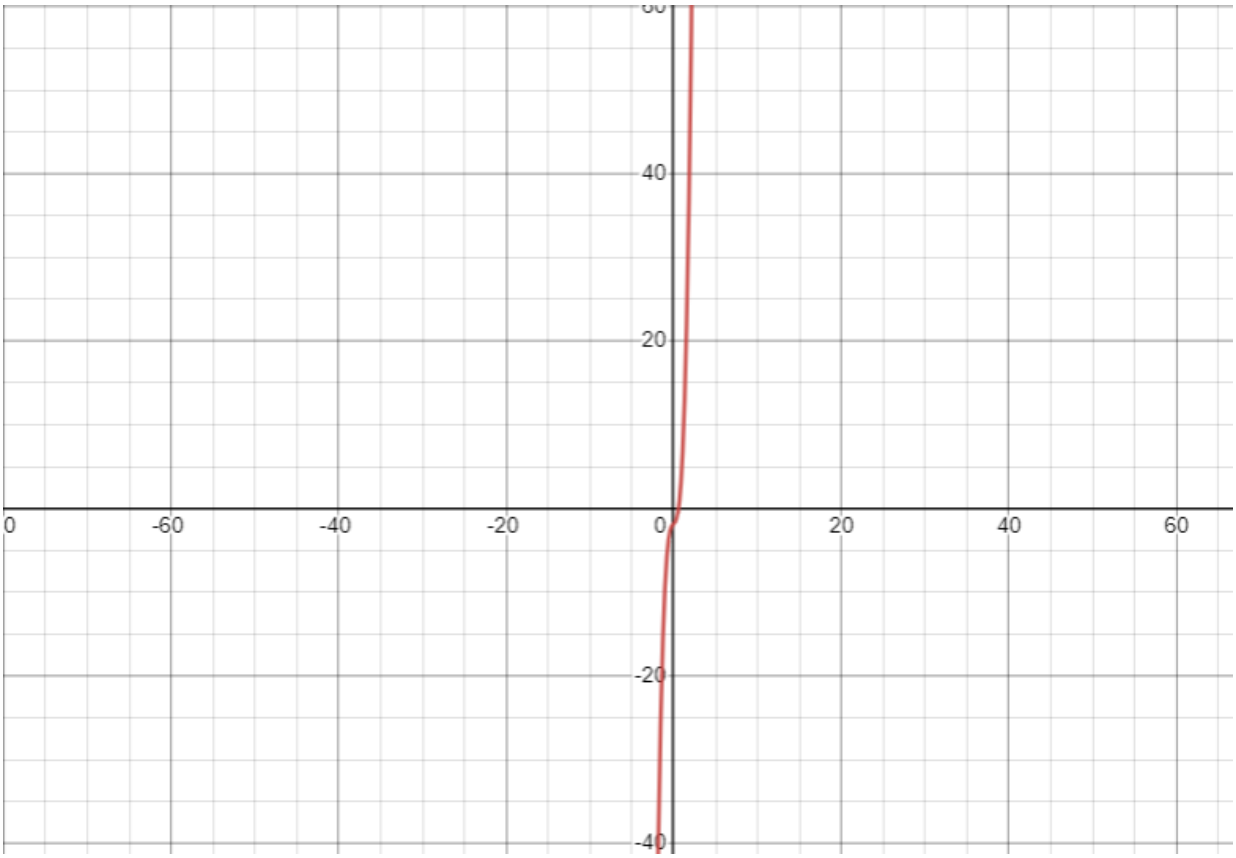
Найдем корни уравнения: $6 * t3 + t2 + t - 1.8 \ = 0$. Для этого выполним следующие шаги:

$$\frac{dF}{dt} = 18 * t^2 + 2 * t + 1$$

$$18 * t^2 + 2 * t + 1 = 0$$

$$D_1 = \ 2^2 - 4 * 18 * 1; \ -68 < 0.$$

График функции:



Тогда функция возрастает на всей области определения, значит, уравнение имеет лишь один корень. Если $t = 0$, тогда $f(0) = -1.8 < 0.$; если $t = 1$, тогда $f(1) = 4.2 > 0$.

Возьмем промежуток [0;1]. Первым элементом последовательности в промежутке [a; b] будет являться середина промежутка, то есть $t_0 = \frac{a+b}{2}$. Следующие элементы будут считаться по формуле $t_n = t_{n-1} - c * f(t_{n-1})$, где $c = \frac{1}{\min(f'(t))+\max(f'(t))}$.

Итак, пусть $\varepsilon = 0{,}01$, промежуток [0;1], тогда:

$t_0 = \frac{0+1}{2} = 0{,}5;$

$$t1 = 0.5 - \frac{1}{11} * (6 * 0.5^3 + 0.5^2 + 0.5 - 1.8) = 0.527$$

$|0.5 - 0.527| \leq 0.01$; Получим, $0{,}027 \leq 0{,}01$ - неверно. Повторяем вычисления:

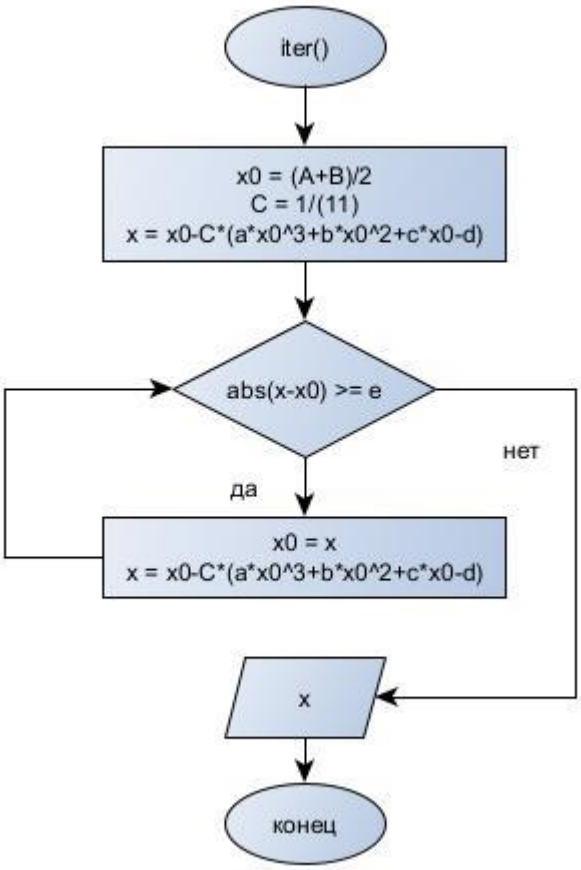
$t2 = 0.527 - \frac{6*0.527^3+0.527^2+0.527-1.8}{11} = 0.538.$

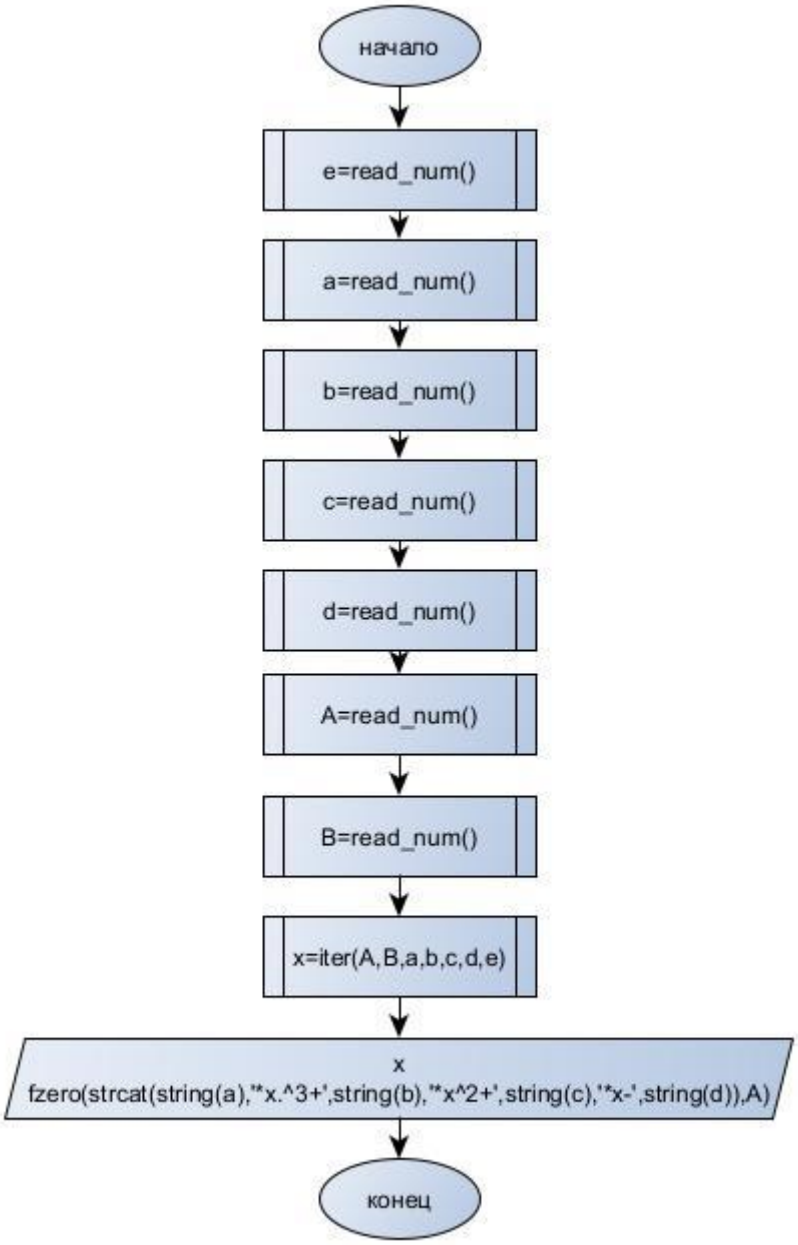
$|0.527 - 0.538| \leq 0.01$ Получим, $0{,}011 \leq 0{,}01$ - неверно. Повторяем вычисления:

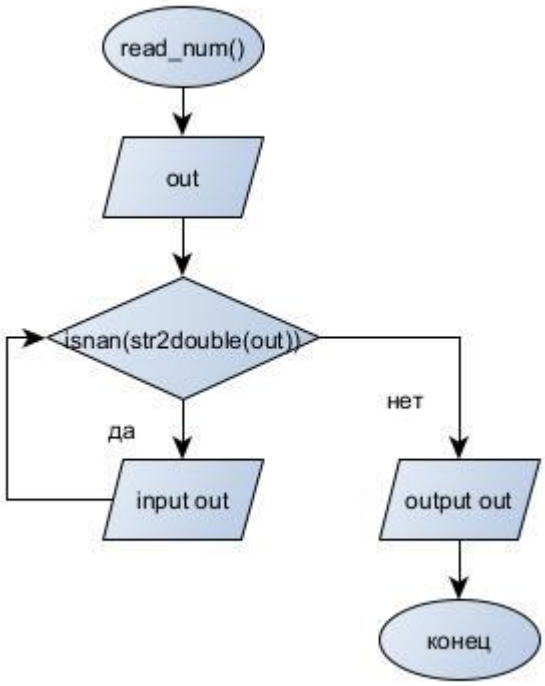
$t3 = 0.538 - \frac{6*0.538^3+0.538^2+0.538-1.8}{11} = 0.542.$

$|0.538 - 0.542| \leq 0.01$ верно. Приближенный корень: $t = 0.542$.

1.4. Разработка алгоритмов решения задачи.




начало
e=read_num()
a=read_num()
b=read_num()
c=read_num()
d=read_num()
A=read_num()
B=read_num()
x=iter(A,B,a,b,c,d,e)
x
fzero(strcat(string(a),'*x.^3+',string(b),'*x^2+',string(c),'*x-',string(d)),A)
конец



1.5. Листинг программы.

Untitled.m:

```
clc; clear;
e = read_num("Введите погрешность(0.001): ");
disp("Введите коэффициенты a, b, c, d")
a = read_num("a(6): ");
b = read_num("b(1): ");
c = read_num("c(1): ");
d = read_num("d(1.8): ");
disp("Введите начало и конец отрезка")
A = read_num("Начало отрезка(0): ");
B = read_num("Конец отрезка(1): ");
x=iter(A,B,a,b,c,d,e);%высчитывание методом простых итераций
fprintf("Метод простых итераций: %f\n",(x))
fprintf("Встроенная функция Matlab: %f\n",(fzero(strcat(string(a),'*x.^3+',string(b),'*x^2+',string(c),'*x-
',string(d)),A)))
```

iter.m:

```
function [x] = iter(A,B,a,b,c,d,e)
x0 = (A+B)/2;
C = 1/(11);
x = x0-C*(a*x0^3+b*x0^2+c*x0-d);
while abs(x-x0) >= e%если текущие минус предыдущее больше чем погрешность, то...
   x0 = x;%предыдущему присваивается текущее
   x = x0-C*(a*x0^3+b*x0^2+c*x0-d);%расчет нового текущего значения
end
```

read_num.m:

```
function [out] = read_num(text)
out = input (text,'s');
while( isnan(str2double(out)))
disp ('Вы ввели не число');
out = input('Введите число:','s');
end
  out = str2double(out);
end
```

1.6. Результат работы программы.

```
Введите погрешность(0.001): 0.001
Введите коэффициенты a, b, c, d
a(6): 6
b(1): 1
c(1): 1
d(1.8): 1.8
Введите начало и конец отрезка
Начало отрезка(0): 0
Конец отрезка(1): 1
Метод простых итераций: 0.543011
```

1.7. Результат реализации метода с помощью встроенных функций.

```
Введите погрешность(0.001): 0.001
Введите коэффициенты a, b, c, d
a(6): 6
b(1): 1
c(1): 1
d(1.8): 1.8
Введите начало и конец отрезка
Начало отрезка(0): 0
Конец отрезка(1): 1
Метод простых итераций: 0.543011
Встроенная функция Matlab: 0.543207
```

1.8.Заключение.

В данной задаче важно корректно указать погрешность и границы отрезка. Разница между результатами методом итераций и встроенной функцией Matlab напрямую зависит от введённой погрешности и границ.

2.1. Раздел 2. Математическая постановка задачи.

Для заданной таблично функции yn = f (xn) и найти значения функции в точках: $x = 1.25$ и $x = 3.55$ с помощью интерполяционного многочлена Ньютона.

№ вар.	x_n	1,2	1,5	1,7	2,0	2,4	2,6	3,1	3,3	3,5	3,9
17	$f(x_n)$	8,278	8,547	8,811	9,087	9,354	9,625	9,897	10,166	10,436	10,707

2.2. Описание метода численной реализации задания.

Условия, которые накладываются на исходные данные:

Значение X должно быть на промежутке от 1.2 до 3.9.

Входные данные:

1)Значения узлов Xn и F(Xn);

2)Значения узла X0, в котором нужно найти значение F(X0) .

Выходные данные:

Приближённое значение функции в X0

[2]

2.3. Описание метода численной реализации задания

Пример решения:Нам даны следующие значения узлов X и Y:

X	1.2	1.5	1.7	2.0	2.4	2.6	3.1	3.3	3.5	3.9
Y	8.278	8.547	8.815	9.087	9.354	9.625	9.897	10.166	10.436	10.707

Нам нужно найти значение Y в точке $x = 1.25$

В общем виде интерполяционный многочлен Ньютона записывается в следующем виде:

$$\mathrm{Pn}(x) = f(x0) + \sum_{k=1}^{n} \left(f(x0; \dots; xk) \times \prod_{i=0}^{k-1} (x - xi) \right)$$

$$(x + a)^n = \sum_{k=0}^{n} \binom{n}{k} x^k a^{n-k}$$

где *n* - степень полинома,

$f(x_0, \dots, x_k)$ - разделенная разность *k*-го порядка, вычисляемая как:

$$f(xi;\ xi + 1; \dots; xi + k) = \frac{f(xi + 1; xi + 2 \dots; xi + k) - f(xi;\ xi + 1; \dots; xi + k - 1)}{xi + k - xi}$$

Пример:

$$f(x0;\ x1) = \frac{f(x1) - f(x0)}{x1 - x0}$$

$$\mathrm{F(x0; x1)} = \frac{269}{300}x + \frac{3601}{500}$$

F(x0;x1) = 8.32

$$f(x0;\ x1; x2) = \frac{f(x1; x2) - f(x0; x1)}{x2 - x0} = \frac{\frac{f(x2) - f(x1)}{x2 - x1} \_ \frac{f(x1) - f(x0)}{x1 - x0}}{x2 - x0}$$

$$\mathrm{F(x0; x1; x2)} = \frac{133}{150}x^2 - \frac{1123}{750}x + \frac{4399}{500}$$

F(x0;x1;x2) = 8.31

$$\mathrm{F(x0;\,x1x2;\,x3)} = -\frac{263}{120}x^3 - \frac{1053}{100}x^2 - \frac{7441}{480}x + \frac{31009}{2000}$$

F(x0;x1;x2;x3)= 8.30

Разделенную разность *k*-го порядка также можно выразить через значения функции в точках с помощью такой формулы:

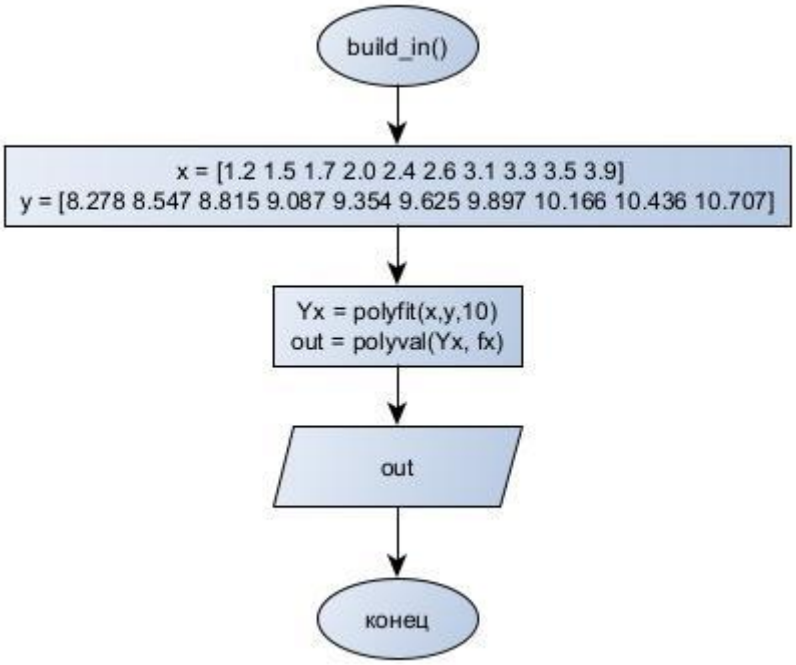
$$f(x0;\ x1;\ldots;xk) = \sum_{i=0}^{k}\left(\frac{f(xi)}{\prod_{j=0,j\neq i}^{k}(xi-xj)}\right)$$

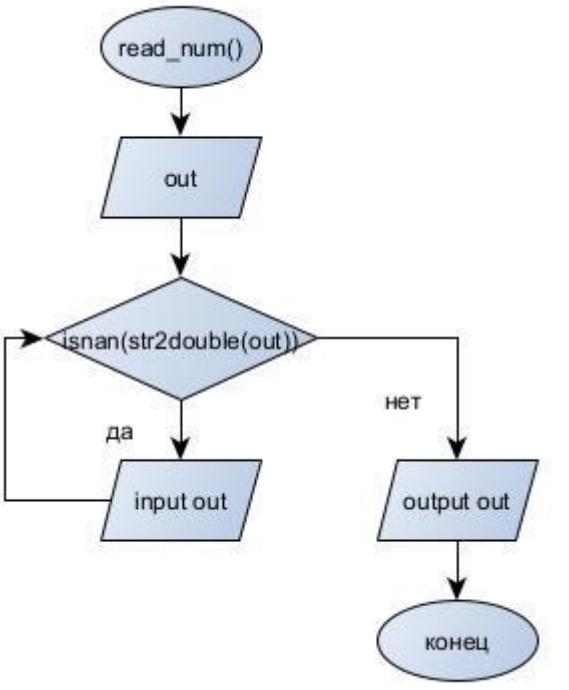
Подставим в эту формулу табличные значения.
Интерполяционный многочлен Ньютона имеет вид:

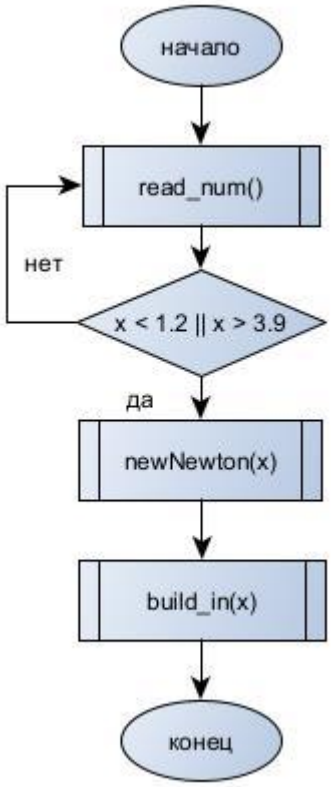
$$\frac{78756131875}{35715856176}x^9 - \frac{1181289078625}{23810570784}x^8 + \frac{17417331949775}{35715856176}X^7 - \frac{301824029665}{10989494208}X^6 + \frac{185906926036533}{190484566272}X^5 - \frac{129581920074296623}{5714536988160}X^4 + \frac{29807341316079641}{865838937600}X^3 - \frac{65308074785162089}{1984214232000}X^2 + \frac{23778782529449341}{1322809488000}X - \frac{52174757989}{12236000}$$

При подстановке $x = 1.25$ получим $Y = 8.50$, что и является ответом.

2.4. Разработка алгоритмов решения задачи.







2.5. Листинг программы.

Untitled.m:

```
clear all
clc
x = 0;
while( x < 1.2 || x > 3.9)
x = read_num();%функция, котороя проверяет корректность введенного числа
end
disp('Рассчёт функцией Ньютона: ')
newNewton(x);%функция, которая рассчитывает значение Y по Ньютону
disp('Встроенная функция Matlab: ')
build_in(x);%функция, которая рассчитывает значение Y по встроенной функции Матлаба
```

read_num.m:

```
function [out] = read_num()
out = input ('Введите число x, которое находится от 1.2 до 3.9: ', 's');
while( isnan(str2double(out)))
disp ('Вы ввели не число, нужно ввести число от 1.2 до 3.9');
out = input('Введите число от 1.2 до 3.9:','s');
end
  out = str2double(out);
end
```

newNewton.m:

```
function out = newNewton(fx)
x = [1.2 1.5 1.7 2.0 2.4 2.6 3.1 3.3 3.5 3.9];
y = [8.278 8.547 8.815 9.087 9.354 9.625 9.897 10.166 10.436 10.707];
n = length(x)-1;%длина вектора x
sum = 0;%для суммы
for i = 1 : n+1
rr = 1;%для умножения
   for j = 1 : n+1
   if j ~= i
   rr = rr*(fx-x(j))/(x(i)-x(j));%вычисление полиномиального коэффициента
   end
   end
sum = sum + y(i) * rr;%получение уравнения
end
out = sum;
disp(out)
```

build_in.m:

```
function [] = build_in(fx)
   x = [1.2 1.5 1.7 2.0 2.4 2.6 3.1 3.3 3.5 3.9];
   y = [8.278 8.547 8.815 9.087 9.354 9.625 9.897 10.166 10.436 10.707];
   warning('off','all');
   Yx = polyfit(x,y,10);
   %вычисление полиномиального коэффициента
   out = polyval(Yx, fx);
   %высчитывает значение Y из полученного уравнения
   disp(out)
end
```

2.6. Результат работы программы.

```
Введите число x, которое находится от 1.2 до 3.9: 1.25
Рассчёт функцией Ньютона:
    8.5024
```

2.7. Результат реализации метода с помощью встроенных функций.

```
Введите число x, которое находится от 1.2 до 3.9: 1.25
Рассчёт функцией Ньютона:
    8.5024

Встроенная функция Matlab:
    8.4028
```

2.8.Заключение.

В данной задаче для нахождения значения функции нужно учитывать условия для интерполирования, результат работы программы совпадет со значением встроенной функции в пределах погрешности.

3.1. Раздел 3. Математическая постановка задачи.

Найдите приближенное значение интеграла Лапласа:

$$\Phi 0(c) = \int_0^c e^{\frac{x^2}{2}} dx$$

с точностью ε = 0,001 методом Симпсона.

17	0,042

3.2. Описание метода реализации задания.

Рассмотрим определенный интеграл $1 = \int_a^b f(x)dx$ функция, непрерывная на отрезке [a;b]. Проведём разбиение отрезка [a;b] на **чётное** количество **равных** отрезков. Чётное количество отрезков обозначают через 2n. [3]

3.3. Описание ручного счета тестового примера предложенным методом.

На практике отрезков может быть:

два: $2n = 2$

четыре: $2n = 4$

восемь: $2n = 8$

десять: $2n = 10$

Разбиение имеет следующий вид:

$[a = x_0; x_1], [x_1; x_2], [x_2; x_3], \ldots, [x_{2n-2}; x_{2n} = b]$

Точки $x_0; x_1; x_2; x_3; x_{2n-2}; x_{2n}$ называют **узлами**.

Формула Симпсона для приближенного вычисления определенного интеграла имеет следующий вид:

$$\int_a^b f(x)dx \approx \frac{h}{3}\left[f(x0) + f(x2n) + 2\left(f(x2) + f(x4) + \cdots + f(x2n-2)\right)\right.$$
$$\left. + 4\left(f(x1) + f(x3) + \cdots + f(x2n-1)\right)\right]$$

, где:

$h = \frac{(b-a)}{2n}$– длина каждого из маленьких отрезков или **шаг**;

$f(xi)$– значения подынтегральной функции в точках x_0; x_1; x_2; x_3; x_{2n-2}; x_{2n} .

Формула подробнее:

$f(x0) + f(x2n)$– сумма первого и последнего значения подынтегральной функции;

$2(f(x2) + f(x4) + \cdots + f(x2n-2)$ – сумма членов с *чётными* индексами умножается на 2;

$4(f(x1) + f(x3) + \cdots + f(x2n-1)$– сумма членов с *нечётными* индексами умножается на 4.

Пример:

$\Phi 0(c) = \int_0^c e^{\frac{x^2}{2}} dx$

c= 0,042

$$2n = 2$$

$$h = \frac{0{,}042 - 0}{2} = 0{,}021$$

i	0	1	2
x_i	0	0.021	0.042

$f(x_i)$	1	0.996478	0.985987

$$\frac{0,021}{3}[1 + 0.985987 + 4 \cdot 0.996478] = 0.08360659$$

$$2n = 4$$

$$h = \frac{0.042 - 0}{4} = 0.0105$$

i	0	1	2	3	4
x_i	0	0.0105	0.021	0.0315	0.042
$f(x_i)$	1	0.999118	0.996478	0.992093	0.985987

$$\frac{0.0105}{3}[1 + 0.985987 + 2 \cdot 0.996478 + 4 \cdot (0.999118 \cdot 0.992093)] = 0.028$$

Погрешность:

$$\frac{[0.042 - 0.028] * 1}{15} = 0.0009$$

Основная идея (для метода Рунге) состоит в вычислении приближения выбранным методом с шагом h, а затем с шагом h/2, и дальнейшем рассмотрении разностей погрешностей для этих двух вычислений.

$k = 4$ для составной формулы Симпсона

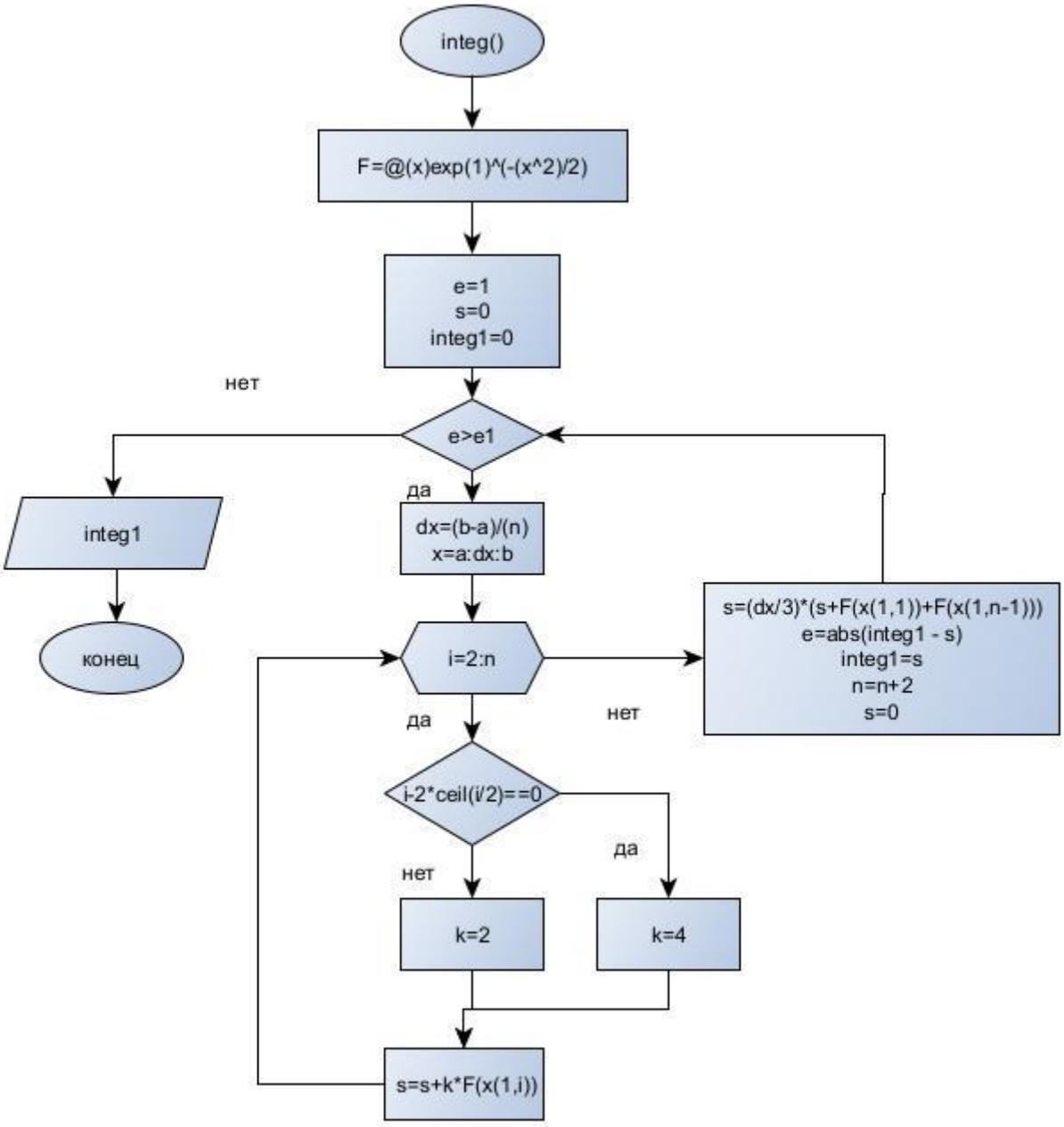
$$I - I_{\frac{h}{2}} \approx \frac{|I_{\frac{h}{2}} - I_h|}{2^k - 1}$$

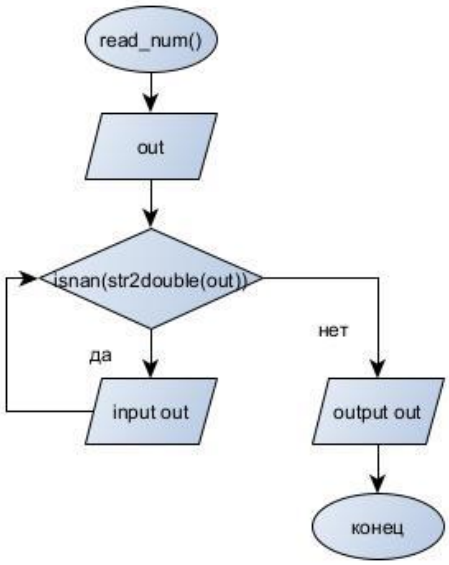
Если неравенство для соответствующего метода не выполняется, то найденное значение интеграла не удовлетворяет заданной точности.

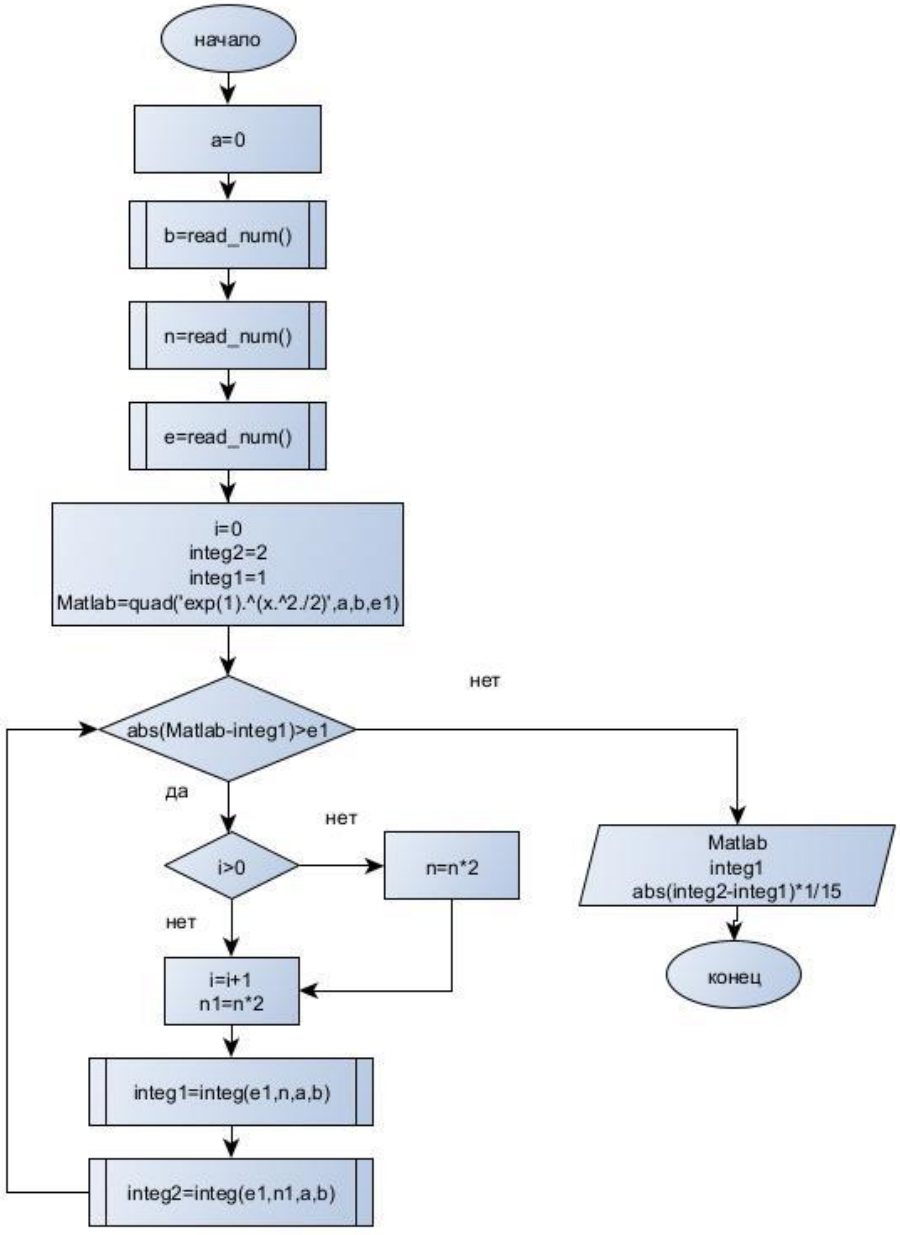
Тогда проводят новые вычисления с шагом h/2 и вновь проверяют выполнение неравенства. Этот прием многократного уменьшения шага применяют до тех пор, пока соответствующее неравенство не станет истинным. [4]

Расписать алгоритм, что если точность не уд

3.4. Разработка алгоритмов решения задачи.




read_num()
out
isnan(str2double(out))
да
нет
input out
output out
конец


начало
a=0
b=read_num()
n=read_num()
e=read_num()
i=0
integ2=2
integ1=1
Matlab=quad('exp(1).^(x.^2./2)',a,b,e1)
нет
abs(Matlab-integ1)>e1
да
нет
i>0
n=n*2
нет
i=i+1
n1=n*2
integ1=integ(e1,n,a,b)
integ2=integ(e1,n1,a,b)
Matlab
integ1
abs(integ2-integ1)*1/15
конец

3.5. Листинг программы.

integ.m:

```
function [integ1] = integ(e1,n,a,b)
F=@(x)exp(1)^(-(x^2)/2);
integ1=0;
e=1;
s=0;
while(e>e1)
   dx=(b-a)/(n);%высчитывание шага
   x=a:dx:b;%определение вектора x исходя из шага
   for i=2:n
      if i-2*ceil(i/2)==0%ceil-округление до ближайшего целого числа к большему
      k=4;%если число оказалось четным, то k=4
      else
      k=2;%если число оказалось не четным, то k=2
      end
   s=s+k*F(x(1,i));
   end
   s=(dx/3)*(s+F(x(1,1))+F(x(1,n-1)));
   e=abs(integ1 - s);
   integ1=s;
   disp("Количество шагов= ")
   n
   n=n+2;
   s=0;
end
```

read_num.m:

```
function [out] = read_num(text)
out = input (text, 's');
   while( isnan(str2double(out)))
   disp ('Вы ввели не число, нужно ввести число');
   out = input('Введите число:','s');
   end
out = str2double(out);
end
```

Untitled.m:

```
clc;
clear;
a=0;%нижняя граница интегрирования
b=read_num('Введите верхнюю границу интегрирования(0.042):\n');
n=read_num('Введите количество равных отрезков(например 2):\n');
e1=read_num('Введите точность(0.001):\n');
i=0;
integ2=2;
integ1=1;
Matlab=quad('exp(1).^(x.^2./2)',a,b,e1);
while(abs(Matlab-integ1)>e1)
```

```
   if(i>0)
      n=n*2;
   end
i=i+1;
n1=n*2;
disp("Подсчет интеграла с шагом n ")
integ1=integ(e1,n,a,b);%рассчет интеграла с шагом n
disp("Подсчет интеграла с шагом 2*n ")
integ2=integ(e1,n1,a,b);%рассчет интеграла с шагом 2n
end
disp('Встроенная функция Matlab: ')
disp(Matlab)%функция матлаба
disp('Функция Симпсона: ')
disp(integ1)%вывод результата с шагом n
disp('Погрешность равна:')
disp(abs(integ2-integ1)*1/15)%рунге
```

3.6. Результат работы программы.

```
Введите верхнюю границу интегрирования(0.042):
0.042
Введите количество равных отрезков(например 2):
2
Введите точность(0.001):
0.001



Функция Симпсона:
    0.0420
```

3.7. Результат реализации метода с помощью встроенных функций.

```
Введите верхнюю границу интегрирования(0.042):
0.042
Введите количество равных отрезков(например 2):
2
Введите точность(0.001):
0.001
Встроенная функция Matlab:
    0.0420

Функция Симпсона:
    0.0420

Погрешность равна:
   7.8092e-08
```

3.8.Заключение.

В данной задаче погрешность подсчета зависит от точности, заданной пользователем, чем больше точность, тем меньше погрешность.

Список использованной литературы:

1. https://studme.org/199277/informatika/metod_prostyh_iteratsiy
2. https://matica.org.ua/metodichki-i-knigi-po-matematike/spravochnik-a-a-gusak-v-m-gusak/31-3-interpoliatcionnyi-mnogochlen-niutona
3. mathprofi.ru/formula_simpsona_metod_trapecij.html
4. https://matica.org.ua/metodichki-i-knigi-po-matematike/kurs-vysshei-matematiki-3/06-metod-runge-kutta