**Министерство науки и высшего образования Российской Федерации**

федеральное государственное автономное образовательное учреждение

высшего образования

**«НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ**

**ТОМСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»**

Инженерная школа природных ресурсов

Направление подготовки Химическая технология

Отделение химической инженерии

**ИТЕРАЦИОННЫЕ МЕТОДЫ РЕШЕНИЯ НЕЛИНЕЙНЫХ УРАВНЕНИЙ**

**Лабораторная работа по дисциплине «Углубленный курс информатики»**

Выполнил студент гр. 2Д91 Г.Н. Омельченко

(Подпись)

\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_ 2020 г.

Отчет принят:

Преподаватель

доцент ОХИ ИШПР, к.т.н. В.А. Чузлов

(Подпись)

\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_ 2020 г.

Томск 2020 г.

**Цель работы:** научиться составлять программы итерационных методов решения нелинейных уравнений.

**Теоретическая часть**

**Итерационные методы** – это методы последовательных приближений. В них необходимо задать некоторое приближённое решение – начальное приближение. После этого с помощью некоторого алгоритма проводится один цикл вычислений, называемый итерацией. В результате итерации находят новое приближение. Итерации проводятся до получения решения с требуемой точностью.

Преимущества итерационных методов:

- требуют хранения в памяти машины не всей матрицы системы, а лишь нескольких векторов с n компонентами;

- Погрешности окончательных результатов при использовании итерационных методов не накапливаются, поскольку точность вычислений в каждой итерации определяется результатами предыдущей итерации и практически не зависти от ранее выполненных вычислений.

Суть итерационных методов

1. Вводятся исходные данные:

- коэффициенты уравнений,

- допустимое значение погрешности ε,

- начальные приближения значений неизвестных (вектор – столбец X(0) ).

2. Организуется циклический вычислительный процесс, каждый цикл которого представляет собой одну итерацию – переход от предыдущего приближения X(k-1) к последующему X(k) .

**Метод Ньютона (метод касательных).** Для предотвращения расходимости иногда целесообразно использовать смешанный алгоритм. Он состоит в том, что сначала применяется всегда сходящийся метод (например, метод деления отрезка пополам), а после некоторого числа итераций – быстро сходящийся метод Ньютона.

**Практическая часть**

**Задание 1**

**Исходные данные**: x4+3\*x – 20 = 0

Интервал [1; 2], допустимая точность 10-2

**Задание**

Составьте программу для решения нелинейных уравнений методом половинного деления, простых итераций и методом Ньютона.

**Программная реализация**

**1 способ:**

**Program** ddf;

**const**

zzz = 1e-2;

**function** z(x: real);

**begin**

result:= exp(ln(x) \* 4) + 3 \* x - 20

**end**;

**function** d(a, b: real; zzz: real): real;

**var**

x: real;

**begin**

**repeat**

x:= (a + b) / 2;

**if** z(a) \* z(x) > 0 **then**

a:= x

**else**

b:= x

**until** (abs(a - b) <= zzz) **or** (z(x) = 0);

result:= x

**end**;

**begin**

writeln(d(1, 2, zzz))

**end**.

**Ответ:**

1.9453125

**2 способ:**

**Program** lab9;

**const**

eee = 1e-2;

**function** g(x: real): real;

**begin**

result := exp(ln(20 - 3 \* x) \* 1 / 4)

**end**;

**function** iterations(a, b: real; eee: real): real;

**var**

x: real;

**begin**

result := a;

**repeat**

x := g(result);

result := g(x)

**until** abs(result - x) <= eee;

**end**;

**begin**

writeln(iterations(1, 2, eee))

**end**.

**Ответ:**

1.94037733840934

**3 способ:**

**Program** reger;

**const**

abc = 1e-2;

**function** f(x: real): real;

**begin**

result := exp(ln(x) \* 4) + 3 \* x - 20

**end**;

**function** f1(x: real): real;

**begin**

result := 4 \* exp(ln(x) \* 3) + 3

**end**;

**function** f2(x: real): real;

**begin**

result := 12 \* exp(ln(x) \* 2)

**end**;

**function** together(a, b: real; abc: real): real;

**var**

x: real;

**begin**

**if** f(a) \* f2(a) > 0 **then**

result := a

**else**

**if** f(b) \* f2(b) > 0 **then**

result := b

**else**

**begin**

writeln('Pешений нет!');

**exit**

**end**;

**repeat**

x := result;

result := x - f(x) / f1(x)

**until** abs(result - x) <= abc;

**end**;

**begin**

writeln(together(1, 2, abc))

**end**.

**Ответ:**

1.94047935224908

**Задание 2**

**Исходные данные**:ex + x – 2 = 0

Интервал [0; 1], допустимая точность 10-3

**Задание**

Составьте программу для решения нелинейных уравнений методом половинного деления, простых итераций и методом Ньютона.

**Программная реализация**

**1 способ:**

**Program** asadas;

**const**

ddd = 1e-3;

**function** f(x: real);

**begin**

result:= exp(x) + x - 2

**end**;

**function** dhmy(a, b: real; ddd: real): real;

**var**

x: real;

**begin**

**repeat**

x:= (a + b) / 2;

**if** f(a) \* f(x) > 0 **then**

a:= x

**else**

b:= x

**until** (abs(a - b) <= ddd) **or** (f(x) = 0);

result:= x

**end**;

**begin**

writeln(dhmy(0, 1, ddd))

**end**.

**Ответ:**

0.4423828125

**2 способ:**

**Program** uab;

**const**

eee = 1e-3;

**function** f(x: real): real;

**begin**

result:= ln(2 - x)

**end**;

**function** iterations(a, b: real; eee: real): real;

**var**

x: real;

**begin**

result:= a;

**repeat**

x:= f(result);

result:= f(x)

**until** abs(result - x) <= eee;

**end**;

**begin**

writeln(iterations(0, 1, eee))

**end**.

**Ответ:**

0.442509950010955

**3 способ:**

**Program** dscsd;

**const**

eee = 0.2 \* 1e-4;

**function** f(x: real): real;

**begin**

result := exp(x) + x - 2

**end**;

**function** f1(x: real): real;

**begin**

result := exp(x) + 1

**end**;

**function** f2(x: real): real;

**begin**

result := exp(x)

**end**;

**function** ponewtonu(a, b: real; eee: real): real;

**var**

x: real;

**begin**

**if** f(a) \* f2(a) > 0 **then**

result := a

**else**

**if** f(b) \* f2(b) > 0 **then**

result := b

**else**

**begin**

writeln('Pешений нет!');

**exit**

**end**;

**repeat**

x := result;

result := x - f(x) / f1(x)

**until** abs(result - x) <= eee;

**end**;

**begin**

writeln(ponewtonu(0, 1, eee))

**end**.

**Ответ:**

0.442854401004033

**Задание 3**

**Исходные данные**:ln(x) + x = 0

Интервал [0.5; 1.5], допустимая точность 0.2\*10-4

**Задание**

Составьте программу для решения нелинейных уравнений методом половинного деления, простых итераций и методом Ньютона

**Программная реализация**

**1 способ:**

**Program** erhb;

**const**

eee = 0.2 \* 1e-4;

**function** f(x: real);

**begin**

result:= ln(x) + x

**end**;

**function** dhmy(a, b: real; eee: real): real;

**var**

x: real;

**begin**

**repeat**

x:= (a + b) / 2;

**if** f(a) \* f(x) > 0 **then**

a:= x

**else**

b:= x

**until** (abs(a - b) <= eee) **or** (f(x) = 0);

result:= x

**end**;

**begin**

writeln(dhmy(0.5, 1.5, eee))

**end**.

**Ответ:**

0.567153930664063

**2 способ:**

**Program** sdfvdv;

**const**

eee = 0.2 \* 1e-4;

**function** f(x: real): real;

**begin**

result:= exp(- x)

**end**;

**function** iterations(a, b: real; eee: real): real;

**var**

x: real;

**begin**

result:= a;

**repeat**

x:= f(result);

result:= f(x)

**until** abs(result - x) <= eee;

**end**;

**begin**

writeln(iterations(0.5, 1.5, eee))

**end**.

**Ответ:**

0.567153930664063

**3 способ:**

**Program** fdbf;

**const**

eee = 0.2 \* 1e-4;

**function** f(x: real): real;

**begin**

result := ln(x) + x

**end**;

**function** f1(x: real): real;

**begin**

result := 1 / x + 1

**end**;

**function** f2(x: real): real;

**begin**

result := -1 / (x \* x)

**end**;

**function** ponewtonu(a, b: real; eee: real): real;

**var**

x: real;

**begin**

**if** f(a) \* f2(a) > 0 **then**

result := a

**else**

**if** f(b) \* f2(b) > 0 **then**

result := b

**else**

**begin**

writeln('Pешений нет!');

**exit**

**end**;

**repeat**

x := result;

result := x - f(x) / f1(x)

**until** abs(result - x) <= eee;

**end**;

**begin**

writeln(ponewtonu(0.5, 1.5, eee))

**end**.

**Ответ:**

0.567143290399369

**Задание 4**

**Исходные данные**: 2\*x – e-0.1\*x =0

Интервал [0.2; 1.5], допустимая точность 0.5\*10-4

**Задание**

Составьте программу для решения нелинейных уравнений методом половинного деления, простых итераций и методом Ньютона.

**Программная реализация**

**1 способ:**

**Program** sdfsdc;

**const**

eee = 0.5 \* 1e-4;

**function** f(x: real);

**begin**

result:= 2 \* x - exp(-0.1 \* x)

**end**;

**function** dhmy(a, b: real; eee: real): real;

**var**

x: real;

**begin**

**repeat**

x:= (a + b) / 2;

**if** f(a) \* f(x) > 0 **then**

a:= x

**else**

b:= x

**until** (abs(a - b) <= eee) **or** (f(x) = 0);

result:= x

**end**;

**begin**

writeln(dhmy(0.2, 1.5, eee))

**end**.

**Ответ:**

0.476718139648438

**2 способ:**

**Program** lab9;

**const**

eps = 0.5 \* 1e-4;

**function** g(x: real): real;

**begin**

result:= exp(-0.1 \* x) / 2

**end**;

**function** iterations(a, b: real; eps: real): real;

**var**

x: real;

**begin**

result:= a;

**repeat**

x:= g(result);

result:= g(x)

**until** abs(result - x) <= eps;

**end**;

**begin**

writeln(iterations(0.2, 1.5, eps))

**end**.

**Ответ:**

0.476721637710739

**3 способ:**

**Program** coffindancer;

**const**

eee = 0.5 \* 1e-4;

**function** f(x: real): real;

**begin**

result := 2 \* x - exp(-0.1 \* x)

**end**;

**function** f1(x: real): real;

**begin**

result := 2 + 0.1 \* exp(-0.1 \* x)

**end**;

**function** f2(x: real): real;

**begin**

result := 0.1 \* exp(-0.1 \* x)

**end**;

**function** ponewtonu(a, b: real; eee: real): real;

**var**

x: real;

**begin**

**if** f(a) \* f2(a) > 0 **then**

result := a

**else**

**if** f(b) \* f2(b) > 0 **then**

result := b

**else**

**begin**

writeln('Pешений нет!');

**exit**

**end**;

**repeat**

x := result;

result := x - f(x) / f1(x)

**until** abs(result - x) <= eee;

**end**;

**begin**

writeln(ponewtonu(0.2, 1.5, eee))

**end**.

**Ответ:**

0.476723086001294

**Выводы**

В ходе работы были успешно составлены программы итерационных методов решения нелинейных уравнений.