**Содержание**

1. Индивидуальное задание

2. Постановка задачи и формализация

3. Выбор, обоснование, краткое описание методов

3.1 Численное интегрирование

3.1.1 Постановка задачи

3.1.2 Выбор и описание метода

3.2 Отыскание корня уравнения

3.2.1 Постановка задачи

3.2.2 Выбор и описание метода (половинное деление)

4. Проверка условий сходимости методов

5. Тестирование программных модулей

5.1 Тестирование модуля численного интегрирования

5.1.1 Схема алгоритма тестирующей программы

5.1.2 Код тестирующей программы

5.1.3 Результат тестирования

5.2 Тестирование модуля отыскания корня уравнения методом половинного деления

5.2.1 Схема алгоритма тестирующей программы

5.2.2 Код тестирующей программы

5.2.3 Результат тестирования

5.3 Прогонка программы

5.3.1 Схема алгоритма программы при прогонке

5.3.2 Код программы при прогонке

5.3.3 Результаты работы программы при прогонке

6. Детализированная схема алгоритма

7. Код программы

8. Полученные результаты

9. Проверка результатов в MathCAD

10. Основные выводы

Список литературы

модуль корень половинный деление

**1. Индивидуальное задание**

Решить уравнение на отрезке x℮[0;2р]



**2. Постановка задачи и формализация**

Задача заключается в поиске корня уравнения f(x)=0 численным методом на отрезке неопределённости [0; 2р], где



Интегрирование проводится численным методом.

Для решения поставленной задачи необходимо разработать следующие модули:

- главный модуль, вводящий исходные данные (требуемую точность и концы отрезка неопределённости) и выводящий конечный результат (решение уравнения)

- модуль, задающий подынтегральное выражение

- модуль, выполняющий численное интегрирование и вычитающий р/2

- модуль, решающий нелинейное уравнение f(x)=0, где f(x) – значение функции, полученное в предыдущем модуле

Укрупнённый алгоритм решения задачи:



**3. Выбор, обоснование, краткое описание методов**

**3.1 Численное интегрирование**

**3.1.1 Постановка задачи**

Если функция f(x) непрерывна на отрезке [a;b] и дифференцируема, то определённый интеграл от этой функции в пределах от a до b существует и может быть вычислен по формуле Ньютона-Лейбница:



Причём



Задача численного интегрирования заключается в нахождении значения определённого интеграла через ряд значений подынтегральной функции yi=f(xi), заданной в точках xi (i=0,1,…,n), причём x0=a, xn=b. Чащё всего интервал разбивают на подынтервалы длиной h=xi+1 – xi

Для получения простых формул интегрирования используют полином нулевой, первой и второй степени и соответственно получаются формулы численного интегрирования: прямоугольников, трапеций, Симпсона.

Замена функции f(x) интерполирующим полиномом приводит к образованию погрешности вычисления значения интеграла



Здесь I1 – точное значение интеграла, I – значение, вычисленное численным методом, R- погрешность расчёта численным методом.

**3.1.2 Выбор и описание метода**

**Выбор метода:**

Находить значение интеграла можно многими способами, среди которых:

1. формула прямоугольников
2. формула трапеций
3. формула Симпсона

Выберем для вычисления интеграла по заданию формулу Симпсона, т.к. подынтегральная функция, имеет нелинейный характер и метод Симпсона обеспечивает

Наибольшую точность, т.к. подынтегральная функция аппроксимируется полиномом 2 порядка.

**Описание метода:**

Если для каждой пары отрезков [xi;xi+2] построить многочлен второй степени, затем проинтегрировать его и воспользоваться свойством аддитивности интеграла, то получим формулу Симпсона:

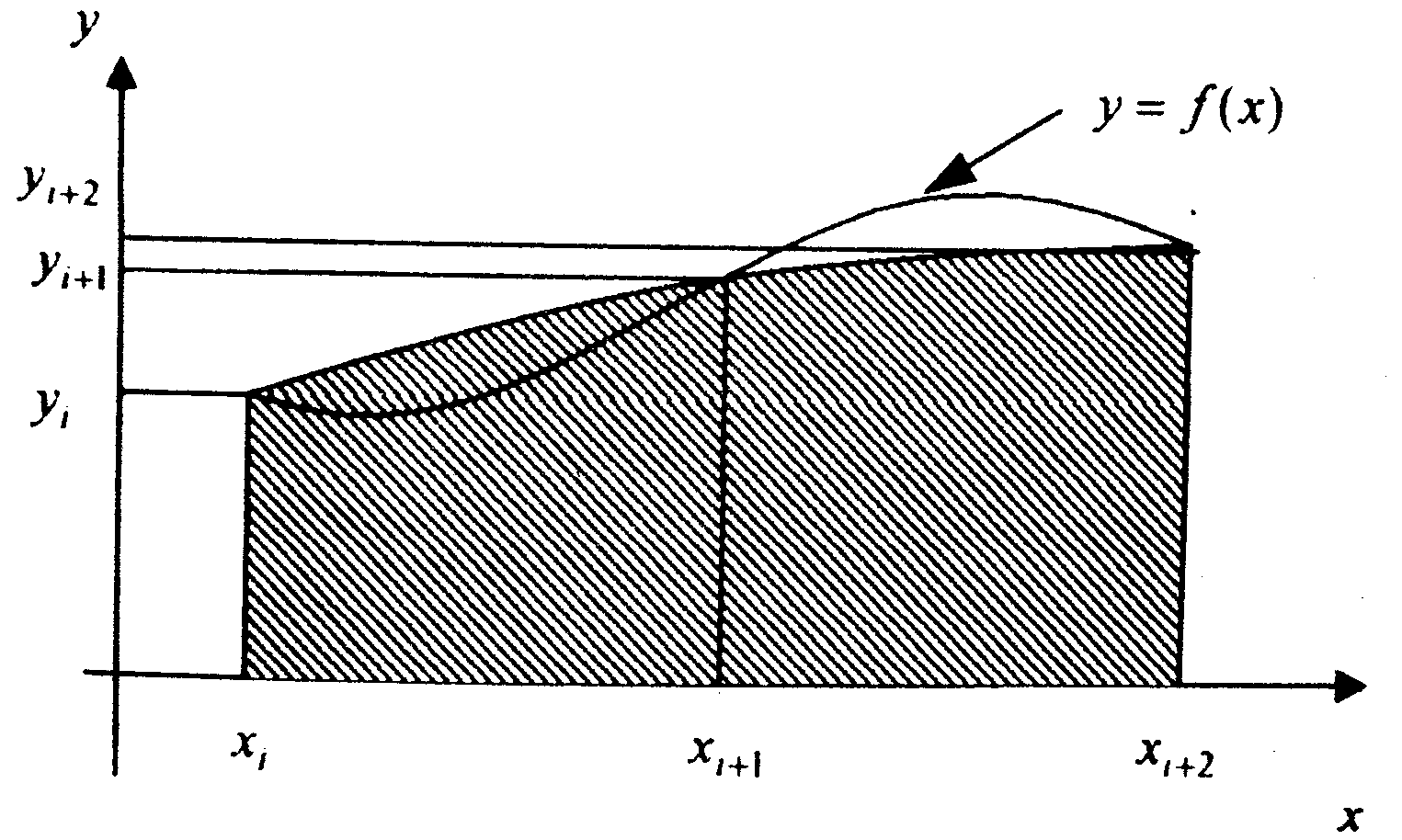


n=2\*m – чётное число



**Геометрическая интерпретация формулы Симпсона:**

На отрезке [xi;xi+2] длиной 2h строится парабола, проходящая через три точки (xi;yi), (xi+1;yi+1), (xi+2;yi+2). Площадь под параболой, заключённой между осью абсцисс и прямыми x=xi, x=xi+2, принимают равной интегралу



**3.2 Поиск корня нелинейного уравнения**

**3.2.1 Постановка задачи**

Пусть требуется найти решение уравнения f(x)=0. f(x) – непрерывная функция в конечном или бесконечном интервале. Если f(x) представляет собой многочлен, то уравнение называют алгебраическим, в противном случае – трансцендентным.

Всякое значение x=x\*, обращающее f(x) в ноль, называется корнем этого уравнения.

Решение задачи отыскания изолированных корней состоит из двух этапов: отделение корней, уточнение корней. При отыскании действительных корней этап отделения производится либо графически, либо аналитически, основываясь на теореме: если f(x) принимает на разных концах отрезка [a;b] разные знаки, то на [a;b] существует по меньшей мере один корень уравнения f(x)=0.

Корень будет единственным на отрезке [a;b], если производная f(x) существует и сохраняет знак внутри [a;b].

**3.2.2 Выбор и описание методов**

**Выбор метода:**

Существует множество методов решения нелинейных уравнений, среди которых:

- метод половинного деления

- метод итераций

- метод Ньютона

- метод хорд

Выберем для решения нелинейного уравнения по заданию метод половинного деления, т.к. он имеет самые простые условия сходимости (не налагает никаких условий на производные f(x)) и прост в алгоритмизации.

**Описание метода:**

Пусть требуется уточнить единственный корень уравнения f(x)=0, принадлежащий отрезку [a;b] (отрезок неопределённости)

Точка c=(a+b)/2 – середина отрезка [a;b].

Если f(c)=0, то корень найден.

В противном случае для дальнейшего рассмотрения оставляют ту половину отрезка неопределённости [a;c] или [c;b], на концах которой знаки функции f(x) различны. При этом получается последовательность вложенных отрезков, содержащая искомый корень.

На каждом шаге длина отрезка неопределённости уменьшается вдвое. Метод сходится всегда.

Условием окончания поиска корня является (b-a)/2n<E или |f(x)|<E, где Е – точность, [a;b] – начальный отрезок неопределённости, n – число итераций

**4. Проверка условий сходимости методов**

**Интегрирование по методу Симпсона**

Для вычисления по методу Симпсона требуется, чтобы функция была непрерывной на отрезке интегрирования.

sin(t)/t=1 при t=0 по первому замечательному пределу, однако при вычислении в QBasic будет выдавать ошибку деления на ноль, поэтом в точке t=0 приравняем 1 искусственно.



Условие для вычисления по методу Симпсона выполняется.

**Отыскание корня нелинейного уравнения методом половинного деления:**



Условие f(a)\*f(b)<0 выполняется при x=[0;5],

Условие единственности корня (sign (f `(x))= const при x=[a;b]) выполняется при x e [0;3]. На отрезке x=[3.3; 2\*р] есть ещё один корень, поэтому сократим отрезок неопределённости до [0; 3] (с учетом условия метода Симпсона).

**5. Тестирование программных модулей**

**5.1 Тестирование модуля численного интегрирования**

Для тестирования модуля, вызовем его для отыскания интеграла



**5.1.1 Схема алгоритма тестирующей программы:**

**Схема алгоритма управляющей программы:**



**Схема алгоритма f(x):**



**Схема алгоритма модуля численного интегрирования при тестировании:**



**5.1.2 Код тестирующей программы:**

DECLARE function integr (afix,x,E)

DECLARE FUNCTION fint (x)

CLS

PRINT "Itog"; E; integr(0,1,0.001)

END

FUNCTION fint (t)

fint = EXP(t)

END FUNCTION

FUNCTION integr (afix, x, E)

aint = afix: bint = x

nint = 2: h = (bint - aint) / 2: s = (fint(aint) + 4 \* fint((aint + bint) / 2) + fint(bint)) \* (h / 3)

DO

nint = 2 \* nint: h = (bint - aint) / nint: s1 = s: cin = 4: x = aint: s = fint(aint) + fint(bint)

FOR i = 1 TO nint - 1

x = x + h: s = s + cin \* fint(x): cin = 6 - cin

NEXT i

s = s \* h / 3

LOOP UNTIL ABS(s - s1) < E

integr = s

END FUNCTION

**5.1.3 Результат тестирования:**



Модуль отработал верно: при точности Е=0.001, I=1.718283, отрезок интегрирования разделился на 4, шаг h=0.25

**5.2 Тестирование модуля поиска корня уравнения методом половинного деления**

Протестируем модуль поиска корня уравнения на примере f(x)=1-x. В качестве отрезка неопределённости возьмём x=[-1;2] . Очевидно, что корень этого уравнения находится в x=1.

**5.2.1 Схема алгоритма тестирующей программы:**

**Схема алгоритма управляющей программы:**



**Схема алгоритма модуля поиска корня уравнения методом половинного деления при тестировании:**



**Схема алгоритма модуля fint(t):**



**5.2.2 Код тестирующей программы:**

DECLARE FUNCTION fint (t)

DECLARE FUNCTION uravn (afix, bfix, E)

CLS

PRINT uravn(-1, 2, .001)

END

FUNCTION fint (t)

fint = 1 - t

END FUNCTION

FUNCTION uravn (afix, bfix, E)

aur = afix: bur = bfix: cur = (aur + bur) / 2: n = 0

PRINT TAB(10); "Promezhutochnie dannie"

PRINT " a b f(a) f(b) b-a"

DO UNTIL (bur - aur) <= E

n = n + 1

IF fint(cur) \* fint(bur) < 0 THEN aur = cur ELSE bur = cur

PRINT USING "##.## ##.## ##.### ##.### ##.###"; aur; bur; fint(aur); fint(bur); bur - aur

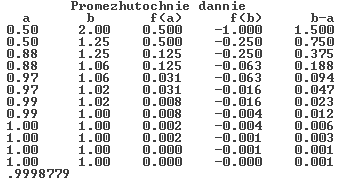
cur = (aur + bur) / 2

LOOP

uravn = cur

END FUNCTION

**5.2.3 Результат тестирования**



Модуль отработал верно.

**5.3 Прогонка программы**

Протестируем главный модуль, задав



Отрезок неопределённости x=[0;3]

Очевидно, что корень находится в x=2. Проверим это.

**5.3.1 Схема алгоритма программы при прогонке:**



**5.3.2 Код программы при прогонке:**

DECLARE FUNCTION fint (t)

DECLARE FUNCTION integr (afix, x, E)

DECLARE FUNCTION uravn (afix, bfix, E)

CLS

LOCATE 1, 15

PRINT "Kursovaya rabota po informatike OTLADKA"

LOCATE 2, 18

PRINT "Gruppa PS0601, Kudlo Alexey"

LOCATE 4, 10

afix = 0: bfix = 3: E = .001

PRINT TAB(14); "Znacheniya f(x) na [a;b]"

PRINT TAB(19); "x f(x)"

FOR i = 0 TO 10

PRINT USING " ##.### ##.####"; i \* .3; integr(0, i \* .3, .0001)

NEXT i

xx = uravn(afix, bfix, E)

PRINT

PRINT TAB(5); "Iskomij koren` x\*="; xx; " bil najden s tochnost`ju E="; E

END

FUNCTION fint (t)

fint = 1

END FUNCTION

FUNCTION integr (afix, x, E)

aint = afix: bint = x

nint = 2: h = (bint - aint) / 2: s = (fint(aint) + 4 \* fint((aint + bint) / 2) + fint(bint)) \* (h / 3)

DO

nint = 2 \* nint: h = (bint - aint) / nint: s1 = s: cin = 4: x = aint: s = fint(aint) + fint(bint)

FOR i = 1 TO nint - 1

x = x + h: s = s + cin \* fint(x): cin = 6 - cin

NEXT i

s = s \* h / 3

LOOP UNTIL ABS(s - s1) < E

x = bint

integr = s - 2

END FUNCTION

FUNCTION uravn (afix, bfix, E)

aur = afix: bur = bfix: cur = (aur + bur) / 2

PRINT

PRINT TAB(15); "Promezhutochnie dannie pri poiske kornya"

PRINT

PRINT TAB(12); "a b f(a) f(b) b-a"

DO UNTIL bur - aur <= E

IF integr(afix, cur, E) \* integr(afix, bur, E) < 0 THEN aur = cur ELSE bur = cur

PRINT USING " ##.## ##.## ##.### ##.### ##.###"; aur; bur; integr(afix, aur, E); integr(afix, bur, E); bur - aur

cur = (aur + bur) / 2

LOOP

uravn = cur

END FUNCTION

**5.3.3 Результат прогонки программы:**



По значениям f(x) можно определить, что f(x)=x-2. Корень f(x) найден правильно.

Проверка результатов тестирования в среде MathCAD не требуется из-за очевидности полученных результатов.

**6. Детализированная схема алгоритма:**



**7. Код программы**

DECLARE FUNCTION fint (t)

DECLARE FUNCTION integr (afix, x, E)

DECLARE FUNCTION uravn (afix, bfix, E)

CLS

LOCATE 1, 15

PRINT "Kursovaya rabota po informatike"

LOCATE 2, 18

PRINT "Gruppa PS0601, Kudlo Alexey"

LOCATE 4, 15

INPUT "Vvedite a, b, E"; afix, bfix, E

PRINT TAB(14); "Znacheniya f(x) na [a;b]"

PRINT TAB(19); "x f(x)"

FOR i = 0 TO 10

PRINT USING " ##.### ##.####"; i \* .3; integr(0, i \* .3, .0001)

NEXT i

xx = uravn(afix, bfix, E)

PRINT

PRINT TAB(5); "Iskomij koren` x\*="; xx; " bil najden s tochnost`ju E="; E

END

FUNCTION fint (t)

IF t = 0 THEN fint = 1 ELSE fint = SIN(t) / t

END FUNCTION

FUNCTION integr (afix, x, E)

aint = afix: bint = x

nint = 2: h = (bint - aint) / 2: s = (fint(aint) + 4 \* fint((aint + bint) / 2) + fint(bint)) \* (h / 3)

DO

nint = 2 \* nint: h = (bint - aint) / nint: s1 = s: cin = 4: x = aint: s = fint(aint) + fint(bint)

FOR i = 1 TO nint - 1

x = x + h: s = s + cin \* fint(x): cin = 6 - cin

NEXT i

s = s \* h / 3

LOOP UNTIL ABS(s - s1) < E

x = bint

integr = s - 1.570796

END FUNCTION

FUNCTION uravn (afix, bfix, E)

aur = afix: bur = bfix: cur = (aur + bur) / 2

PRINT

PRINT TAB(15); "Promezhutochnie dannie pri poiske kornya"

PRINT

PRINT TAB(12); "a b f(a) f(b) b-a"

DO UNTIL bur - aur <= E

IF integr(afix, cur, E) \* integr(afix, bur, E) < 0 THEN aur = cur ELSE bur = cur

PRINT USING " ##.### ##.### ##.### ##.### ##.###"; aur; bur; integr(afix, aur, E); integr(afix, bur, E); bur - aur

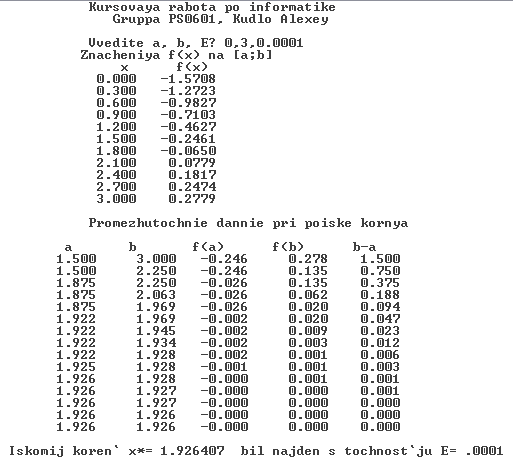
cur = (aur + bur) / 2

LOOP

uravn = cur

END FUNCTION

**8. Полученные результаты**



**9. Проверка результатов в MathCAD**



Полученные в MathCAD и с помощью программы по заданию результаты совпадают

**10. Основные выводы**

1. Обоснованы и выбраны численные методы:

- интегрирования по методу Симпсона

- отыскания корня уравнения (метод половинного деления)

2. Разработаны, протестированы модули, реализующие следующие методы:

- численное интегрирование по методу Симпсона с оценкой погрешности по правилу Рунге

- отыскание корня уравнения по методу половинного деления

3. Программа модульная, содержит следующие модули:

- основной модуль, принимающий исходные данные, передающий их на обработку и выводящий конечный и промежуточный результаты

- модуль численного интегрирования по методу Симпсона,

- модуль отыскания корня уравнения по методу половинного деления, который использует f(x), полученные от модуля численного интегрирования

Во избежание ошибки деления на ноль, модуль, задающий подынтегральную функцию, был модифицирован для выдачи единицы при подаче t=0 (sin(t)/t=1 при t=0 пор первому замечательному пределу)

4. Получены следующие результаты:

Искомый корень x\*=1.926407 был рассчитан с точностью E=0.0001

5. Полученные результаты были проверены в MathCAD

Полученные в ходе работы программы результаты, очень хорошо согласуются с результатами, полученными в MathCAD, требуемая точность E=0.0001 соблюдается.

**Список литературы**

1. Гловацкая А.П., Загвоздкина А.В., Кравченко О.М., Семёнова Т.И., Шакин В.Н: Практикум Численные методы и оптимизация по дисциплине «Информатика»

Москва, МТУСИ, 2004г.

2. А.П.Гловацкая: Конспект лекций «Информатика. Вычислительная математика» Москва, МТУСИ, 2006г.

3. Семёнова Т.И, Шакин В.Н.: Практикум Математический пакет MathCAD в дисциплине «Информатика», Москва, МТУСИ, 2006г.

4. А.В. Загвоздкина: Конспект лекций за 1 семестр 2007-2008 учебного года