**Раздел 2. Курсовая работа**

## Тема 2.1. Методические указания к выполнению курсовой работе

2.1. Цель курсовой работы

2.2. Общие требования к отчету

2.3. Задание

2.4.Постановка задачи

2.5. Описание используемых численных методов

2.6. Тестирование процедур, используемых при решении задачи

2.7. Разработка детализированной схемы алгоритма

2.8. Разработка программы решения задачи

2.9. Результаты

2.10. Проверка результатов с использованием математического пакета

2.11. Основные результаты и выводы по работе

2.12. Оглавление

2.13. Список литературы

2.14. Пример оформления титульного листа

## 1.1. Цель курсовой работы

Целью курсовой работы является:

* получение практических навыков решения физических, математических и технических задач с применением численных методов и методов оптимизации с использованием средств вычислительной техники;
* развитие навыков алгоритмизации, программирования и отладки программ при решении задач вычислительного характера;
* изучение возможностей математических пакетов и развития навыков их практического использования для получения числовых или символьных решений вычислительных задач, а также графических иллюстраций полученных результатов.

В ходе самостоятельной работы требуется провести формализацию задачи, выполнить необходимые предварительные преобразования, составить алгоритм решения задачи, разработать формы ввода исходных данных и вывода результатов, разработать и отладить программу решения задачи, получить и проанализировать полученные результаты.

Результаты работы оформляются в виде отчета.

## 1.2. Общие требования к отчету

Отчет по курсовой работе (пояснительная записка) оформляется средствами **MSWord** на стандартных листах формата А4. Для ввода текста должны быть использованы следующие настройки: шрифт – **Times New Roman**, размер шрифта для ввода основного текста – **12**, для заголовков глав и пунктов следует использовать элемент панели форматирования **Стиль**, позволяющего произвести выбор соответствующего уровня вложения заголовка. Страницы текста пояснительной записки должны быть пронумерованы (на титульном листе номер не указывается).

Все графики, используемые для иллюстраций текста, должны быть построены с использованием средств одного из математических пакетов (например, MathCad или MatLab). Схемы алгоритмов должны быть выполнены в соответствии с принятым ГОСТом (табл.2.1).

Таблица 2.1

|  |  |  |
| --- | --- | --- |
| **Наименование** | **Обозначение** | **Функция** |
| **Процесс** |  | Выполнение операции или группы операций, в результате которых изменяется значение, форма представления или расположения данных |
| **Решение** |  | Выбор направления выполнения алгоритма в зависимости от некоторых переменных условий |
| **Обращение к**  **процедуре** |  | Использование ранее созданных и отдельно описанных алгоритмов |
| **Ввод/вывод** |  | Ввод исходных данных или вывод результатов |
| **Линии потока** |  | Указание последовательности связи между блоками |
| **Соединитель** |  | Указатель связей между прерван­ными линиями потока |
| **Пуск/останов** |  | Начало или конец обработки данных |
| **Комментарий** |  | Пояснения элементов схемы |
| **Модификация** |  | Начало регулярной циклической структуры |

Пояснительная записка должна содержать **следующие обязательные разделы**:

1. титульный лист;
2. индивидуальное задание;
3. описание последовательности решения задачи, проиллюстрированное укрупненной схемой алгоритма решения задачи, снабженной комментариями, поясняющими блоки схемы алгоритма;
4. обоснование выбора методов, используемых в ходе решения задачи, и краткое описание;
5. проверку сходимости выбранных методов и выполнения условий теорем применительно к поставленной задаче;
6. тестирование процедур, реализующих используемые методы, причем проведение тестирования каждой процедуры оформляется в отдельном подразделе и должно содержать:

* описание теста, используемого для проверки программного теста (простейшая функция или уравнение, набор входных данных и известных результатов);
* детализированную схему алгоритма;
* формы для ввода исходных данных и вывода результатов (включая промежуточные);
* код программы тестируемого модуля;
* результаты тестирования и их анализ;
* вывод о проведенном тестировании.

1. описание программы, которое должно содержать:

* детализированные схемы алгоритмов;
* формы ввода исходных данных и вывода результатов, снабженные комментариями;
* код программы, разработанной по принципу процедурного программирования, где обмен данными между процедурами должен осуществляться через список параметров;
* правильность выполнения программы должна быть подтверждена необходимым количеством промежуточных результатов, код программы снабжен соответствующими комментариями, которые могут быть расположены либо на выходной форме, либо присутствовать на дополнительной форме, вызов которой обеспечивается из формы результатов;

1. проверку решения задачи (по заданию преподавателя: ручной расчет или расчет, проведенный с использованием математического пакета);
2. результаты, полученные в ходе выполнения курсовой работы, анализ и выводы;
3. содержание отчета (с указанием страниц);
4. список используемой литературы.

## 2.3. Задание

Вариант задания студенту выдается преподавателем, который ведет в данной группе занятия по курсовой работе. Методы, используемые для решения поставленной задачи, либо также задаются преподавателем, либо, в соответствии с заданием, должны быть выбраны студентом самостоятельно в ходе выполнения работы.

Задание может быть представлено в **формализованном** (математическом) виде, например:

***Функция y(t) задана обыкновенным дифференциальным уравнением***

******

***Найти значение параметра a  при котором  минимально отличается***

***от заданного значения  при .***

Или в **неформализованном** виде (смысловая задача), например:

***Якоб Бернулли доказал, что гибкая нерастяжимая нить (провода, тросы, гирлянды и т.п.), которая подвешена в двух точках, имеет форму цепной линии и описывается уравнением***



***По критерию***

******

***с точностью  определить величину «ошибки Галилея», полагавшего, что провисшая нить имеет форму параболы .***

***Положить, что нить имеет длину м, закреплена на опорах, расстояние между которыми м.***

При получении задания в неформализованном виде перед его выполнением следует провести формализацию. Этот этап предшествует описанию последовательности решения задачи.

## 2.4. Постановка задачи

Начинать выполнение задания рекомендуется с **постановки задачи**, где достаточно кратко в словесной форме описывается последовательность решения задачи. При необходимости в этом разделе должна быть проведена формализация задания с обоснованием применения численного метода. В качестве иллюстрации постановки задачи должна быть приведена **укрупненная схема алгоритма** решения задачи, которая представляет собой ее логическую последовательность решения. В укрупненной схеме алгоритма каждый блок представляет собой этап решения задачи, поэтому в блоке принято приводить словесное описание действий (например, «Вычисление оптимального значения параметра k»). Кроме того, в отдельные блоки целесообразно выделять те фрагменты, которые впоследствии предполагается оформить в виде процедур-подпрограмм или процедур-функций (например, подынтегральную или целевую функцию).

## 2.5. Описание используемых численных методов

Перед описанием используемых численных методов необходимо произвести их выбор и обоснование возможности их использования для решения поставленной задачи.

Если методы, необходимые для решения задачи, заданы преподавателем, то их выбор обосновывается с точки зрения возможности их применения для решения поставленной задачи. В случае, если методы предлагается выбрать самостоятельно, выбор каждого метода должен быть обоснован с точки зрения его необходимости и эффективности применения.

Если в задании предлагается решение отдельных подзадач провести двумя или более методами, то обосновывается возможность их применения, а в выводах проводится сравнительный анализ результатов, а также анализ числовых и качественных характеристик, свойственных использованию заданных методов.

Описание самих методов должно быть кратким и проводиться с учетом поставленной задачи. Например, если описываемый метод используется для решения нелинейного уравнения, то в ходе описания метода в разделе выбора начального приближения требуется провести исследования функции заданного уравнения и произвести выбор начального приближения для решаемого уравнения.

**2.6. Тестирование процедур, используемых при решении задачи**

Каждая отдельная подзадача, выявленная в процессе описания задачи (численный метод, построение таблицы значений функции, построение графика функции и т.п.), должна быть оформлена в виде процедуры. Чтобы удостовериться в правильности программной реализации процедуры, ее следует протестировать.

**Тест** представляет собой набор исходных данных и известных результатов. Так, например, известно, что . Таким образом, если использовать в качестве подынтегральной функции , а в качестве пределов значения 0 и , то результатом работы программы, реализующей метод численного интегрирования, должна быть 1. Следует помнить, что применение численных методов позволяет получить результат с заданной степенью точности, поэтому полученное количество верных знаков зависит от заданной вами точности.

В программе тестируемая часть должна быть оформлена в виде процедуры.

В разделе тестирования **для каждой тестируемой программы** должен быть описан выбранный тест, приведена детализированная схема алгоритма, код программы, результаты выполнения программы (с выводом промежуточных результатов), проверка тестового варианта с использованием математического пакета, а также сделан вывод о сравнении полученных результатов с тестовыми данными.

**2.7. Разработка детализированной схемы алгоритма**

Детализированная схема алгоритма представляет собой набор схем алгоритмов: **главной процедуры** и ряда **вспомогательных процедур**. Схема алгоритма главной процедуры практически повторяет укрупненную схему алгоритма. Она должна содержать только обращения к процедурам или процедурам-функциям, выполняющим конкретные действия (например, ввод данных, вычисление корня уравнения, вывод результатов и т.п.). Однако, в отличие от укрупненной схемы алгоритма, где в каждом блоке использовалось словесное описание действий, здесь содержимым каждого блокаслужит обращение к разработанной процедуре, а, следовательно, в блоке должны быть указаны имя процедуры и (в скобках) список параметров. В соответствии с измененным назначением должна быть изменена и конфигурация блока.

Все схемы алгоритма должны быть снабжены комментариями, поясняющими параметры, назначение процедур и их основные участки.

Схемы алгоритмов рекомендуется выполнять с использованием специальных программных средств (например, **Microsoft Visio**)**.**

**2.8. Разработка программы решения задачи**

После проведения тестирования отдельных процедур приступают к составлению программы решения задачи в целом (проекта). Следует помнить, что структура программы должна полностью соответствовать разработанному алгоритму, и не содержать глобальных переменных.

Проект должен иметь как минимум **две формы**.

**1-я форма должна содержать следующие обязательные элементы:**

* Фамилия И.О. студента,
* номер группы,
* содержание задания,
* исходные данные.

**2-я форма должна содержать все промежуточные и окончательные результаты.**

При необходимости количество форм может быть увеличено.

Код программы должен быть снабжен **комментариями**. Назначение процедур комментируется в 1-й строке процедуры. Переменные и массивы комментируются в строке объявления. Например:

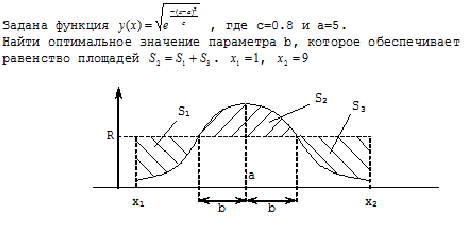
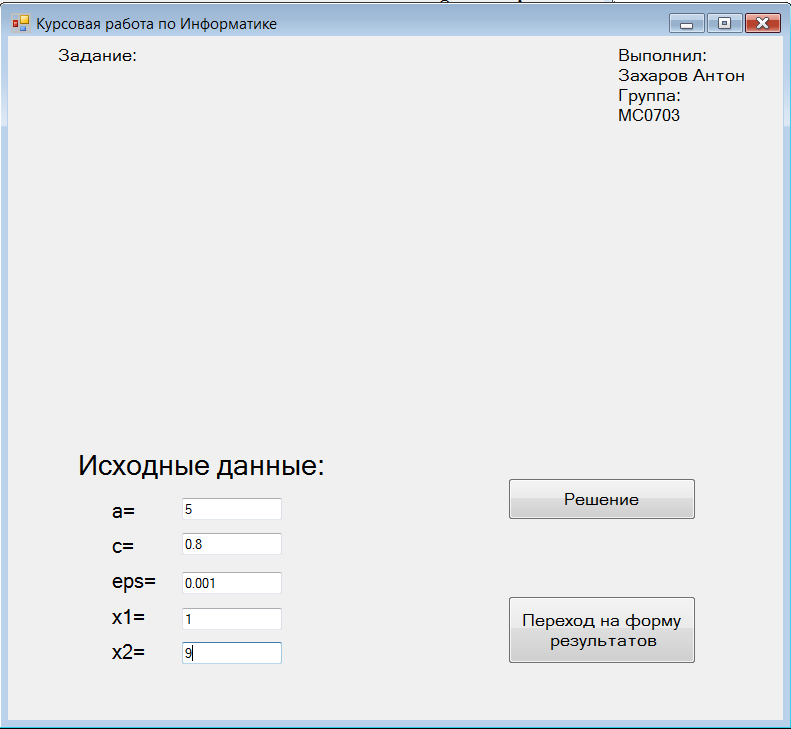
Dim a As Single ‘нижний предел интегрирования

Dim x(n) As Single ‘узлы интерполяции

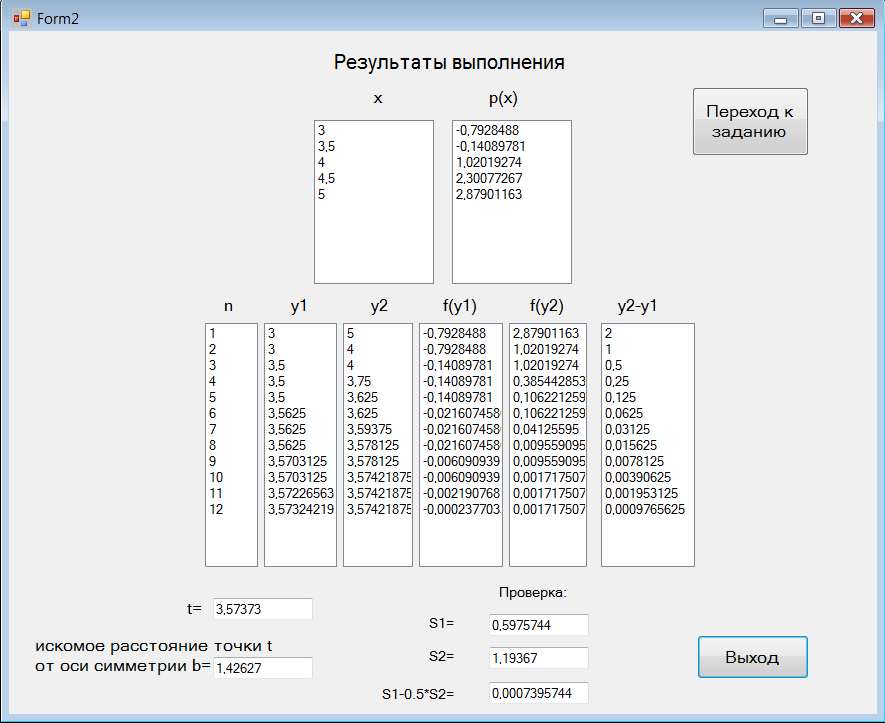
**2.9. Результаты**

В разделе результаты должны быть приведены распечатки двух форм, первая из которых должна содержать все заданные по заданию исходные данные, а вторая – промежуточные и окончательные результаты, полученные в ходе выполнения программы. Например:

1-я форма



2-я форма



**2.10. Проверка результатов с использованием   
математического пакета**

Проверка полученных результатов должна производиться с использованием математического пакета (**Mathcad** или **MathLAB**). Она заключается в решении поставленной в курсовой работе задачи средствами одного из математических пакетов. При этом расчет должен быть проведен с использованием необходимых встроенных функций или символьных операций, проиллюстрирован с помощью таблиц и графиков, а получаемые в ходе расчета результаты–прокомментированы.

Задача, поставленная в курсовой работе, считается решенной, если результаты, полученные при выполнении программы, с заданной степенью точности совпадают с результатами расчета средствами математического пакета.

Ниже приведен пример решения задачи, поставленной в курсовой работе с использованием пакета MathСad:

|  |
| --- |
|  |

**2.11. Основные результаты и выводы по работе**

Этот раздел пояснительной записки является итоговым. Поэтому после его прочтения должно быть полностью понятно, что и как сделано студентом в курсовой работе, какие результаты получены и в какой степени они точны.

Раздел целесообразно оформить в виде ряда пунктов, каждый из которых должен отражать отдельный этап работы. Например:

1. *Разработана последовательность решения задачи, которая заключается в отыскании минимального расстоянии между заданной точкой M(2;2.5) и точкой пересечения двух кривых y1(x, q) и y2(x, q).*
2. *Для решения задачи разработаны и оттестированы процедуры, реализующие выбранные методы (метод половинного деления и метод дихотомии). Результаты тестирования показали работоспособность алгоритмов и программ. Указанные процедуры были использованы при разработке программы решения задачи, поставленной в курсовой работе.*
3. *Абсцисса точки пересечения функции была найдена путем решения нелинейного уравнения y1(x, q) - y2(x, q)=0. В результате решения уравнения получено значение корня х=2.5003, обеспечивающее значение функции y(x)=-0.00005.*
4. *Проведена проверка результатов работы программы с использованием пакета Mathcad. Результаты проверки полностью совпадают с результатами, полученными при выполнении программы.*

*и т.п. …*

**2.12. Оглавление**

Оглавление может быть сформировано автоматически с использованием средств текстового редактора Word. Для этого необходимо, установив курсор в место вставки оглавления, в меню **Вставка** выбрать команду **Ссылка**, а затем **Оглавление и указатели**. В появившемся окне (во вкладке **Оглавление**) установить количество уровней вложений (главы, пункты и т.п.), выбрать по появившемуся образцу стиль оглавления и место указания страниц. После нажатия на клавишу **ОК** оглавление появится на указанном месте. Однако следует помнить, что автоматическое формирование оглавления возможно только в том случае, если заголовки разделов и пунктов текста были введены в соответствующем стиле (элемент **Стиль** на панели **Форматирование**). При этом наименование раздела целесообразно вводить стилем Заголовок1, а пункта – Заголовок2. Если возникает необходимость деления пункта на подпункты, то используется стиль – Заголовок3.

Перед вставкой оглавления рекомендуется произвести нумерацию страниц. В этом случае при вставке оглавления номера страниц, соответствующие разделам и пунктам, расставляются автоматически. Если предполагается, что часть листов будет вставлена (например, отчет необходимо дополнить какими-либо иллюстрациями), то перед расстановкой страниц в нужные места вставляют пустые страницы.

***2.13. Список литературы***

Список литературы должен содержать нумерованный перечень книг, учебников и пособий, используемых при написании отчета. Каждый элемент списка должен содержать: Фамилии И.О. авторов, наименование, издание, год выпуска и количество страниц. Например:

1. Кравченко О.М., Семенова Т.И., Шакин В.Н. Модели решения вычислительных задач (Численные методы и оптимизация). Уч. Пособие/МТУСИ.– М., 2003.-72с.

2. Семенова Т.И., Шакин В.Н. Практикум. Математический пакет Mathcad в дисциплине ИНФОРМАТИКА/МТУСИ.–М., 2006.-121с.

***2.15. Пример оформления титульного листа***

|  |
| --- |
| **Федеральное государственное образовательное бюджетное учреждение высшего профессионального образования**  **Московский технический университет связи и информатики**  **КУРСОВАЯ РАБОТА**  по информатике  Выполнил: студент гр. МС0701  **Иванов М.И.**  Проверила: **Семёнова Т.И.**  Москва 2011 г. |

***Тема 2. Пример выполнения курсовой работы***

# 1. Задание

Вычислить с точностью  , где (a; b) – координаты точки минимума функции  .

Исходные данные:

|  |  |  |
| --- | --- | --- |
| c | d |  |
| 0 | 5 |  |

# 2. Постановка задачи

В курсовой работе требуется вычислить с заданной точностью  определённый интеграл , для которого неизвестны пределы интегрирования **a** и **b**. Эти пределы представляют собой координаты точки минимума функции , для которой заданы границы отрезка поиска минимума. Прежде чем приступить к поиску минимума требуется исследовать заданную функцию с точки зрения существования минимума на заданном отрезке и убедиться, что на этом отрезке минимум единственный.

Значения найденных координат точки минимума (Xmin и Fmin) требуется сравнить между собой, для того чтобы меньшее значение принять за нижний предел интегрирования (**a**), а большее – за верхний предел интегрирования (**b**).

Решение данной задачи требует использование двух численных методов:

- метода одномерной оптимизации;

- метода вычисления определённого интеграла.

Последовательность решения задачи показано ниже в виде укрупнённой схемы алгоритма.



Рис.1 Укрупненная схема алгоритма

# 3. Выбор и обоснование используемых методов

Для решения задачи одномерной оптимизации будет использован ***метод золотого сечения***. Этот метод эффективен с точки зрения количества вычислений функций на отдельной итерации, хотя проигрывает методу дихотомии по количеству требуемых итераций для достижения заданной точности.

Для вычисления определённого интеграла будем использовать **формулу Симпсона**, так как в состав нашей подынтегральной функции входят тригонометрические функции, а следовательно подынтегральная функции нелинейная. Для обеспечения требуемой точности интегрирования используем **метод двойного просчета**, в котором достижения заданной точности проверяются **правилом Рунге**.

## 3.1. Метод золотого сечения

В основу метода положено разбиение отрезка неопределенности **[a;b]** в соотношении золотого сечения, такого, что отношение длины его большей части ко всей длине отрезка равно отношению длины его меньшей части к длине его большей части:

***l***

*l2* *l1*



где **k1, k2** - коэффициенты золотого сечения.

В методе **золотого сечения** каждая точка **(х1 и х2**)осуществляет золотое сечение отрезка.



или



Точка **х1** осуществляет золотое сечение не только отрезка **[a;b]**, но и отрезка **[a;х2]**. Точно так же точка **х2**осуществляет золотое сечение не только отрезка **[a;b]**, но и отрезка **[х1;b].** Это приводит к тому, что значение целевой функции на каждой итерации (кроме первой) вычисляется один раз.

Сокращение отрезка неопределенности следующим образом:

1) если **f(x1) ≤ f(x2),** то **x\*∈[a0;x2]**

2) если **f(x1) > f(x2),** то **x\*∈[x1;b0]**

После каждой итерации длина отрезка неопределенности сокращается в **1.618** раза. Длина конечного отрезка неопределенности **Δn = 0.618nΔ0,** где **Δ0= (b-a)** – начальная длина отрезка.

Условие окончания процесса итераций **Δn  ε.** Отсюда можно найти количество итераций, необходимое для достижения точки минимума:

отсюда или



Методом золотого сечения можно найти минимум только при условии, если отрезок содержит единственный минимум, то есть целевая функция на данном отрезке – унимодальной.

Проверим условие унимодальности для заданного в курсовой отрезка [c, d].



Из графика видно что функция f(t) на заданном отрезке [0,5] не унимодальна, следовательно необходимо сократить отрезок [c,d] .



## 

**Вывод**: Из значений таблицы видно, что отрезке [c,d]=[4,5] функция f(t) унимодальна, так как f`(t) - не убывающая, а f``(t) >0 . Таким образом, для нахождения минимума заданной функции можно использовать **метод золотого сечения**.

## 3.2. Метод Симпсона

Формула Симпсона используется для вычисления определенного интеграла , у которых подынтегральная функция нелинейная или содержит элементарные функции. Для формулы Симпсона используется квадратичный интерполирующий полином (первая формула Ньютона), следовательно, за элементарный интервал интегрирования принимается отрезок [xi;xi+2]. При этом отрезок интегрирования [a;b] разбивается наn отрезков, где n=2m – четное число.

Для элементарного интеграла формула Симпсона имеет вид:

В пределах отрезка [xi;xi+2], на котором подынтегральная функция аппроксимирована многочленом второго порядка, получим приближенную формулу Симпсона:



Тогда для всего интервала интегрирования [a;b] формула Симпсона выглядит следующим образом:

или



при



При вычислении определённого интеграла с заданной точностью используется прием, основанный на двукратном вычислении значения интеграла вначале с шагом h(где h=(b-a)/n), а затем с шагом h/2. Полученные значения интегралов Ih и Ih/2 могут быть применены для оценки погрешности интегрирования по формуле:



где k=4 – для формулы Симпсона.

Это правило называется **правилом Рунге** (или правилом двойного просчета).

# 4. Тестирование процедур, реализующих выбранные методы

## 4.1. Метод золотого сечения

Для проверки правильности процедуры, реализующей метод золотого сечения, в качестве теста используем функцию , имеющую на отрезке [-0.5;0.5] единственный минимум Xmin= 0 , а значение функции в этой точке f(Xmin)=2.

Проведем проверку правильности тестовых данных с использованием средств пакета MathCad:



Ниже приведены схема тестируемой процедуры, код программы и результаты её выполнения.



Option Explicit On

Option Strict On

Imports System.Math

Public Class Form1

Private Sub Button2\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click

End

End Sub

Private Sub Button1\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

Dim a, b, eps, Xmin, Fmin As Single

Vvod(a, b, eps)

zol\_seh(a, b, eps, Xmin, Fmin)

TextBox4.Text = CStr(Xmin)

TextBox5.Text = CStr(Fmin)

End Sub

Sub Vvod(ByRef a As Single, ByRef b As Single, ByRef eps As Single)

a = CSng(Val(TextBox1.Text))

b = CSng(Val(TextBox2.Text))

eps = CSng(Val(TextBox3.Text))

End Sub

Sub zol\_seh(ByVal a As Single, ByVal b As Single, ByVal eps As Single, ByRef Xmin As Single, ByRef Fmin As Single)

Dim k1, k2, f1, f2, x1, x2 As Single

Dim n As Integer

k1 = CSng((3 - Sqrt(5)) / 2)

k2 = CSng((Sqrt(5) - 1) / 2)

x1 = a + k1 \* (b - a)

x2 = a + k2 \* (b - a)

f1 = f(x1) : f2 = f(x2)

Do Until b - a < eps

n = n + 1

If f1 < f2 Then

b = x2 : x2 = x1

x1 = a + k1 \* (b - a) : f2 = f1 : f1 = f(x1)

Else

a = x1 : x1 = x2

x2 = a + k2 \* (b - a) : f1 = f2 : f2 = f(x2)

End If

ListBox1.Items.Add(n)

ListBox2.Items.Add(a)

ListBox3.Items.Add(b)

ListBox4.Items.Add(x1)

ListBox5.Items.Add(x2)

ListBox6.Items.Add(f(x1))

ListBox7.Items.Add(f(x2))

ListBox8.Items.Add(b - a)

Loop

Xmin = (a + b) / 2

Fmin = f(Xmin)

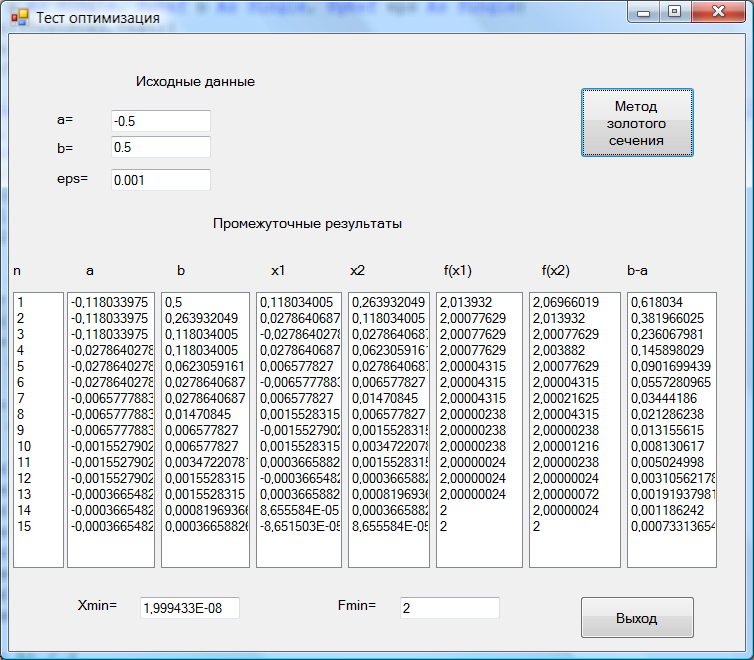
End Sub

Function f(ByVal x As Single) As Single

f = CSng(x ^ 2 + 2)

End Function

End Class



Данные теста совпали с результатами выполнения с точностью = 0.001.

## 4.2. Метод Симпсона

В качестве теста для проверки процедуры, реализующей метод Симпсона, используем известный интеграл:



Значения интеграла получено с использованием средств пакета MathCad.

Ниже приведены схема тестируемой процедуры, код программы и результаты её выполнения.



Option Explicit On

Option Strict On

Imports System.Math

Public Class Form1

Private Sub Button2\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click

End

End Sub

Private Sub Button1\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

Dim a, b, Eps, S As Single

vvod(a, b, Eps)

Runge(a, b, Eps, S)

TextBox4.Text = CStr(Mid(CStr(S), 1, 7))

End Sub

Sub vvod(ByRef a As Single, ByRef b As Single, ByRef Eps As Single)

a = CSng(Val(TextBox1.Text))

b = CSng(Val(TextBox2.Text))

Eps = CSng(Val(TextBox3.Text))

End Sub

Function f(ByVal x As Single) As Single

f = CSng(Cos(x))

End Function

Sub Runge(ByVal a As Single, ByVal b As Single, ByVal Eps As Single, ByRef S As Single)

Dim h, S1 As Single

Dim n As Integer

n = 2

h = (b - a) / n

S = Simp(a, b, n, h)

ListBox1.Items.Add(n)

ListBox2.Items.Add(Mid(CStr(h), 1, 7))

ListBox3.Items.Add(Mid(CStr(S), 1, 7))

Do

n = 2 \* n : h = h / 2 : S1 = S

S = Simp(a, b, n, h)

ListBox1.Items.Add(n)

ListBox2.Items.Add(Mid(CStr(h), 1, 7))

ListBox3.Items.Add(Mid(CStr(S), 1, 7))

Loop Until Abs(S1 - S) / 15 <= Eps

End Sub

Function Simp(ByVal a As Single, ByVal b As Single, ByVal n As Integer, ByVal h As Single) As Single

Dim c, S As Single

Dim i As Integer

S = f(a) + f(b)

c = 4

For i = 1 To n - 1

S = S + c \* f(a + i \* h)

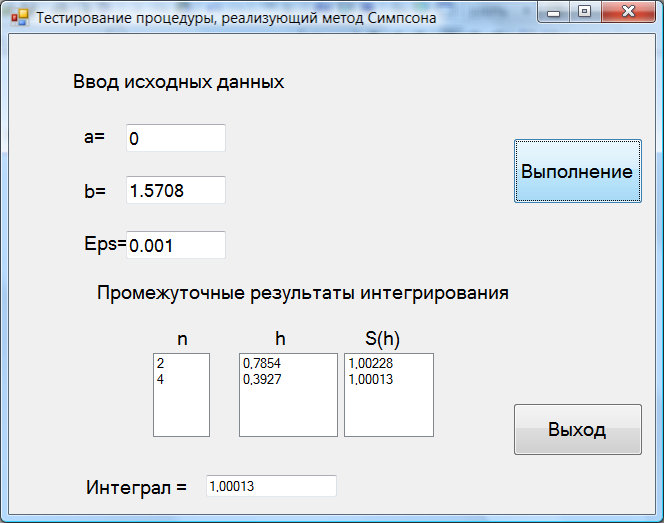
c = 6 - c

Next

Simp = S \* h / 3

End Function

End Class



Данные теста совпали с результатами выполнения с точностью = 0.001.

# 5. Детализированная схема алгоритма решения задачи в целом

1. Главная процедура



2. Процедура ввода исходных данных



3. Процедура – функция (целевая функция)



4. Процедура печати строки таблицы в методе золотого сечения



5. Процедура – функция (подынтегральная функция)



6. Процедура печати строки в методе двойного просчета



Схемы алгоритмов процедур: zol\_seh (метод золотого сечения), Runge ( метод двойного просчета), Simp (формула Симпсона), расположены в разделе 4 – тестирование.

# 6. Код программы

Option Explicit On

Option Strict On

Imports System.Math

Public Class Form1

Private Sub Button3\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button3.Click

End

End Sub

Private Sub Button2\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button2.Click

Form2.Show()

Form2.Focus()

End Sub

Private Sub Button1\_Click(ByVal sender As System.Object, ByVal e As System.EventArgs) Handles Button1.Click

Dim c, d As Single 'Границы отрезка унимодальности

Dim tmin, Fmin As Single 'координаты точки минимума

Dim Eps As Single 'Заданная точность

Dim a, b As Single 'пределы интегрирования

Dim S As Single 'значение определённого интеграла

vvod(c, d, Eps)

zol\_seh(c, d, Eps, tmin, Fmin)

Form2.TextBox4.Text = CStr(tmin)

Form2.TextBox5.Text = CStr(Fmin)

If tmin > Fmin Then

a = Fmin : b = tmin

Else

a = tmin : b = Fmin

End If

Form2.TextBox1.Text = CStr(a)

Form2.TextBox2.Text = CStr(b)

Runge(a, b, Eps, S)

Form2.TextBox3.Text = CStr(S)

Form2.Show()

Form2.Focus()

End Sub

Sub Vvod(ByRef c As Single, ByRef d As Single, ByRef Eps As Single)

c = CSng(Val(TextBox1.Text))

d = CSng(Val(TextBox2.Text))

Eps = CSng(Val(TextBox3.Text))

End Sub

Sub zol\_seh(ByVal a As Single, ByVal b As Single, ByVal eps As Single, ByRef Xmin As Single, ByRef Fmin As Single)

Dim k1, k2, f1, f2, x1, x2 As Single

Dim n As Integer

k1 = CSng((3 - Sqrt(5)) / 2)

k2 = CSng((Sqrt(5) - 1) / 2)

x1 = a + k1 \* (b - a)

x2 = a + k2 \* (b - a)

f1 = f(x1) : f2 = f(x2)

Do Until b - a < Eps

n = n + 1

If f1 < f2 Then

b = x2 : x2 = x1

x1 = a + k1 \* (b - a) : f2 = f1 : f1 = f(x1)

Else

a = x1 : x1 = x2

x2 = a + k2 \* (b - a) : f1 = f2 : f2 = f(x2)

End If

proc1(n, a, b, x1, x2, b - a)

Loop

Xmin = (a + b) / 2

Fmin = f(Xmin)

End Sub

Function f(ByVal t As Single) As Single

f = CSng(4 \* Sin(t) - Sqrt(t))

End Function

Sub proc1(ByVal n As Integer, ByVal a As Single, ByVal b As Single, ByVal x1 As Single, ByVal x2 As Single, ByVal d As Single)

Form2.ListBox1.Items.Add(n)

Form2.ListBox2.Items.Add(Mid(CStr(a), 1, 7))

Form2.ListBox3.Items.Add(Mid(CStr(b), 1, 7))

Form2.ListBox4.Items.Add(Mid(CStr(x1), 1, 7))

Form2.ListBox5.Items.Add(Mid(CStr(x2), 1, 7))

Form2.ListBox6.Items.Add(Mid(CStr(f(x1)), 1, 7))

Form2.ListBox7.Items.Add(Mid(CStr(f(x2)), 1, 7))

Form2.ListBox8.Items.Add(Mid(CStr(d), 1, 7))

End Sub

Function y(ByVal x As Single) As Single

y = CSng(x \* Sin(x) + 2 \* Cos(x))

End Function

Sub Runge(ByVal a As Single, ByVal b As Single, ByVal Eps As Single, ByRef S As Single)

Dim h, S1 As Single

Dim n As Integer

n = 2

h = (b - a) / n

S = Simp(a, b, n, h)

proc2(n, h, S)

Do

n = 2 \* n : h = h / 2 : S1 = S

S = Simp(a, b, n, h)

proc2(n, h, S)

Loop Until Abs(S1 - S) / 15 <= Eps

End Sub

Function Simp(ByVal a As Single, ByVal b As Single, ByVal n As Integer, ByVal h As Single) As Single

Dim c, S As Single

Dim i As Integer

S = y(a) + y(b)

c = 4

For i = 1 To n - 1

S = S + c \* y(a + i \* h)

c = 6 - c

Next

Simp = S \* h / 3

End Function

Sub proc2(ByVal n As Integer, ByVal h As Single, ByVal S As Single)

Form2.ListBox12.Items.Add(n)

Form2.ListBox11.Items.Add(Mid(CStr(h), 1, 7))

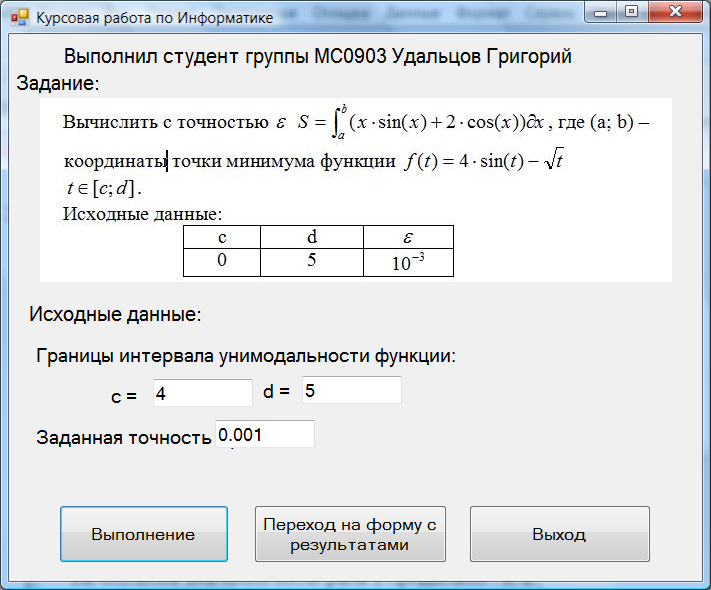
Form2.ListBox10.Items.Add(Mid(CStr(S), 1, 7))

End Sub

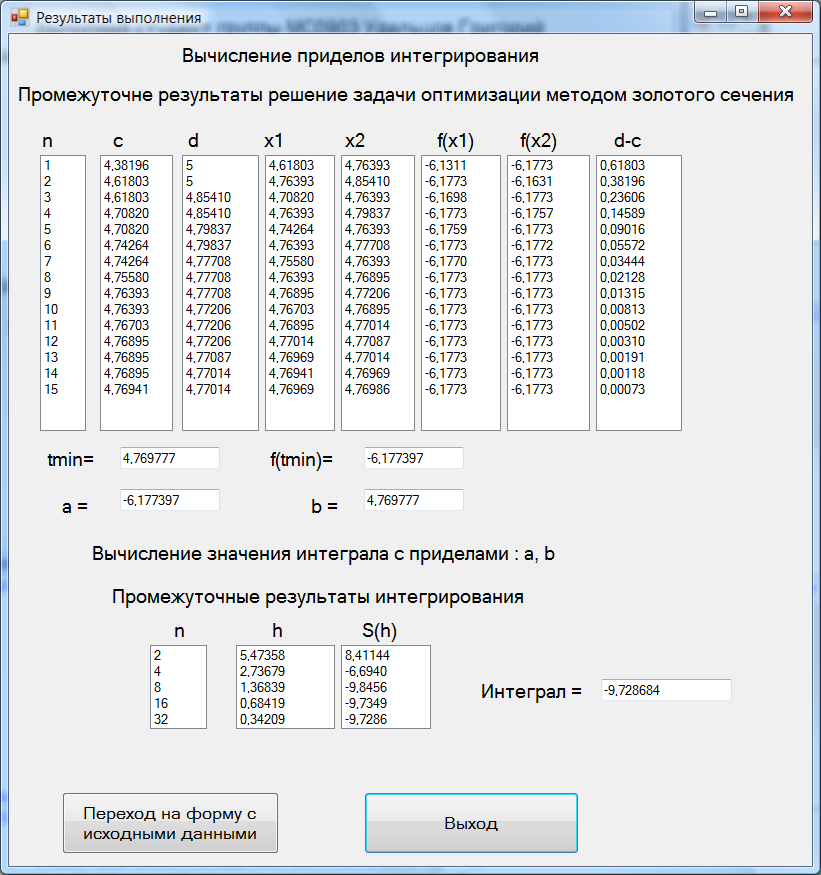
End Class

# 7. Результаты выполнения программы

## 7.1. Форма с заданием и исходными данными



## 7.2. Форма с результатами



# 8. Проверка полученных результатов с использованием пакета MathCad



Из значений таблицы видно , что отрезке [c,d]=[4,5] функция f(t) унимодальна, так как f`(t) - не убывающая, а f``(t) >0 .

Решение задачи одномерной оптимизации



Выбор пределов интегрирования f(tmin)<tmin поэтому



Вычисление определённого интеграла:



# 9. Основные результаты и выводы по работе

1. Разработана последовательность решения задачи, поставленной в курсовой работе, которая состоит в решении задачи одномерной оптимизации ( получение координат точки минимума, являющейся пределами интеграла) и вычислении определённого интеграла от заданной функции с требуемой точностью.
2. Выбраны численные методы, требуемые для решения поставленной задачи:
   * Метод золотого сечения
   * Метод Симпсона.
3. Разработаны и оттестированы процедуры, реализующие численные методы. Результаты тестов показали совпадение полученных результатов с тестовыми данными с заданной точностью 0.001.
4. Разработана программа для решения поставленной в курсовой работе задачи. Выполнение данной программы позволило получить следующие результаты:
   * координаты точки минимума tmin = 4.769777; f(tmin) = -6.177397
   * поскольку f(tmin) < tmin , то

- нижний предел интегрирования a = f(tmin) = -6.177397

- верхний предел интегрирования b = tmin = 4.769777;

* + задача оптимизации решена за 15 итераций



теоретическое количество итераций рассчитано ниже:



* + вычисленное значение определенного интеграла S = -9.728684. Значение получено при h = 0.34209 при количестве разбиений n = 32.

1. Проведена проверка полученных результатов с использованием средств пакета MathCad. Проверка показала, что полученные результаты с заданной степенью точностью совпадают с результатами проверки.

# Список литературы

1. Кравченко О.М., Семенова Т.И., Шакин В.Н. Модели решения вычислительных задач (Численные методы и оптимизация): Уч. Пособие/ МTУСИ.- М., 2003г.
2. Шакин В.Н, Семенова Т.И., Кравченко О.М. Лабораторный практикум. Информатика. Модели и алгоритмы решения задач численными методами с использованием математических пакетов: МTУСИ.- М., 2009 г.
3. Семенова Т.И., Шакин В.Н. Практикум. Математический пакет Mathcad в дисциплине «ИНФОРМАТИКА» »: МTУСИ.- М., 2006 г.
4. Кравченко О.М. Конспект лекций по дисциплине «Информатика», 2010 г.

Оглавление

[1. Задание 108](#_Toc289874226)

[2. Постановка задачи 108](#_Toc289874227)

[3. Выбор и обоснование используемых методов 109](#_Toc289874228)

[3.1. Метод золотого сечения 109](#_Toc289874229)

[3.2. Метод Симпсона 111](#_Toc289874230)

[4. Тестирование процедур, реализующих выбранные методы 112](#_Toc289874231)

[4.1. Метод золотого сечения 112](#_Toc289874232)

[4.2. Метод Симпсона 115](#_Toc289874233)

[5. Детализированная схема алгоритма решения задачи в целом 118](#_Toc289874234)

[6. Код программы 121](#_Toc289874235)

[7. Результаты выполнения программы 123](#_Toc289874236)

[7.1. Форма с заданием и исходными данными 123](#_Toc289874237)

[7.2. Форма с результатами 124](#_Toc289874238)

[8. Проверка полученных результатов с использованием пакета MathCad 125](#_Toc289874239)

[9. Основные результаты и выводы по работе 126](#_Toc289874240)

[Список литературы 127](#_Toc289874241)