Министерство образования и науки Кыргызской Республики

**Кыргызский Государственный Технический Университет**

**им. И. Раззакова**

Кафедра программного обеспечения компьютерных систем

Направление бакалаврской подготовки 710400

ПРОГРАММНАЯ ИНЖЕНЕРИЯ

Бакалаврский курс: Основы разработки и анализа требований к ПО

**Отчет**

по разработке и анализу требований

для программного обеспечения по поиску экстремума функции методом Successive Parabolic Interpolation Method

к курсу: «Основы разработки и анализа требований к ПО»

Выполнил: студент гр. ПИ(б)-1-17

Маликов Сыймык

Проверил: к.т.н., профессор

Тен Иосиф Григорьевич

Бишкек 2020

Оглавление

[Глава 1: Описание проблемы 3](#_Toc63718624)

[Глава 2: Описание Successive Parabolic Interpolation Method 3](#_Toc63718625)

[Глава 3: Спецификация требований к программному обеспечению 5](#_Toc63718626)

[3.1. Наименование программной разработки 5](#_Toc63718627)

[3.2. Описание бизнес-процесса и анализ расхождения 5](#_Toc63718628)

[3.3. Бизнес-цель разработки программного обеспечения 6](#_Toc63718629)

[3.4. Пользовательская история разработки программного обеспечения 6](#_Toc63718630)

[3.5. Функциональные требования к разработке программного обеспечения 7](#_Toc63718631)

[3.6. Нефункциональные требования на разработку программного обеспечения 8](#_Toc63718632)

[3.7. Ограничения на разработку программного обеспечения 8](#_Toc63718633)

[Глава 4. Стадии проектирования системы. 9](#_Toc63718634)

[4.1. Блок-схема работы программы, реализующей Successive Parabolic Interpolation Method для поиска максимума. 9](#_Toc63718635)

[4.2. Блок-схема работы программы, реализующей Successive Parabolic Interpolation Method для поиска минимума. 10](#_Toc63718636)

[4.2. Проектирование интерфейса пользователя программы, реализующей Successive Parabolic Interpolation Method. 11](#_Toc63718637)

[5. Настройки элементов управления формы, реализующей Successive Parabolic Interpolation Method. 11](#_Toc63718638)

[5.1. Код программы, реализующей Successive Parabolic Interpolation Method. 14](#_Toc63718639)

[5.2. Подключенные библиотеки. 23](#_Toc63718640)

[Глава 6. Верификация ПО. 23](#_Toc63718641)

[Глава 7. Оценка достижения бизнес-цели разработки программного обеспечения 33](#_Toc63718642)

## Глава 1: Описание проблемы

Найти решение задачи оптимизации для произвольной заданной нелинейной целевой функции f(x) с заданной допустимой погрешностью Epsilon не более величины ***1E–28*** методом последовательной параболической интерполяции (Successive Parabolic Interpolation Method). Нелинейная функция f(x) имеет произвольный аналитический вид, составленный из математических функций (полиномов различных степеней, тригонометрических – sin(x), cos(x), log(x) и. т. д.), которая имеет математический смысл, и для которой существует хотя бы одно решение задачи.

## Глава 2: Описание Successive Parabolic Interpolation Method

|  |  |
| --- | --- |
| Mathematical Description of Algorithm of the SPIM | Computational Description of Algorithm of the SPIM |
| ||IF x > b /\* Successive Parabolic Interpolation Method to find a minimum \*/  | |THEN  | | |IF fx > fb  | | | |THEN c=x; fc=fx;  | | | |ELSE a=b; fa=fb; b=x; fb=fx;  | | |endIF  | |ELSE  | | |IF fx > fb  | | | |THEN a=x; fa=fx;  | | | |ELSE c=b; fc=fb; b=x; fb=fx;  | | |endIF  |endIF | **INPUT** {a; b; c; Epsilon; Max; f(X);}  **Body of algorithm**  fa = f(a)  fb = f(b)  fc = f(c)  |FOR (i=0; i<Max; i++)  | x= b - ((fb-fc)\*(b-a)^2-(fb-fa)\*(b-c)^2)/  | (2\*((fb-fc)\*(b-a)-(fb-fa)\*(b-c)))  | fx = f(x)  | |IF x > b /\* Successive Parabolic Interpolation Method to find a minimum \*/  | | |THEN  | | | |IF fx > fb  | | | | |THEN  | | | | | c = x  | | | | | fc = fx  | | | |ELSE  | | | | | a = b  | | | | | fa = fb  | | | | | b = x  | | | | | fb = fx  | | | |END IF  | |ELSE  | | |THEN  | | | |IF fx > fb  | | | | |THEN  | | | | | a = x  | | | | | fa = fx  | | | |ELSE  | | | | | c = b  | | | | | fc = fb  | | | | | b = x  | | | | | fb = fx  | | | |END IF  | |END IF  |END FOR  **OUTPUT**  PRINT ‘The optimum solution x\* equal’ x  PRINT ‘The optimum solution was found with the desired tolerance’ Epsilon  PRINT ‘The minimum of objective function f(x\*) is’ fx  PRINT ‘The accuracy is ±‘ Epsilon |
| a – Initial point of parabola;  b – second point of parabola;  c – third point of parabola;  fa – function of point a  fb – function of point b  fc – function of point c | a – Initial point of parabola;  b – second point of parabola;  c – third point of parabola;  Epsilon – Tolerance;  x – new point of parabola  Max – Limit number of iterations. |

## Глава 3: Спецификация требований к программному обеспечению

## 3.1. Наименование программной разработки

Система поиска экстремума нелинейной функции методом поразрядного приближения

## 3.2. Описание бизнес-процесса и анализ расхождения

При решении различных бизнес-задач отделу исследований (пользователям) требуется находить экстремум произвольной нелинейной функции с высокой точностью. Для решения таких задач в этом отделе имеется 5 ставок специалистов. Два из них занимаются формализацией задачи и определением/нахождением вида нелинейной функции, описывающей как можно точнее модель поведения реальной системы, а остальные занимаются решением задачи – поиском экстремума нелинейного уравнения для найденной нелинейной функции. При больших трудозатратах (заняты два специалиста высокой квалификации), не всегда обеспечивается приемлемая и единообразная/унифицированная точность решения задачи. Бизнесу желательно было бы гарантированно обеспечить любую допустимую погрешность решения задачи при любых видах нелинейной функции f(x) с минимальными трудозатратами.

Бизнес-процессы компании выглядят следующим образом:

1. Есть 5 ставок специалистов занимающихся решением задач:
   1. 2 из них занимаются формализацией задачи и определением/нахождением вида нелинейной функции
   2. 3 из них занимаются поиском экстремума для найденной нелинейной функции.
2. Для поиска экстремума найденной нелинейной функции специалисты используют метод последовательной параболической интерполяции (Successive Parabolic Interpolation Method), однако не всегда удается обеспечить приемлемую и единообразную точность решения задачи.

Анализ расхождения

В отделе исследований, бизнес-процесс которого требуется автоматизировать, штатное расписание включает 5 штатных единиц.

Двое из этих 5-ти сотрудников занимаются формализацией задачи и определением аналитического вида нелинейной функции.

Для решения проблемы формализации надо использовать теорию регрессионного анализа экспериментальных данных и для решения полученных проблем надо привлекать метод наименьших квадратов.

В результате таких действий получают аналитический вид нелинейную функцию f(x).

Автоматизация такого бизнес-процесса (труда таких сотрудников) относится к тактическому уровню управления и требует разработки дорогостоящих программ на основе OLAP- систем, которые сопровождает не технологией баз данных, а технологией хранилищ данных.

Трое из этих 5-ти сотрудников занимаются решением задачи-поиском экстремума для найденной нелинейной функции. Автоматизация такого бизнес-процесса (труда таких сотрудников) относится к оперативному уровню управления.

Каждый случай системы ПО уникален: либо она построена на пустом месте с помощью прикладного ПО, которое впервые разрабатывается (реализуется), либо переделана из имеющегося в наличии коммерческого пакета программ.

Оперативный уровень управления обрабатывает оперативные бизнес-данные и документы, типа заказов и счетов. Это – царство OLTP-система, сопровождаемых обычной технологией баз данных.

Бизнес-процессы на первом уровне управления достаточно легко и с минимальными затратами можно автоматизировать. Разработка программного обеспечения для такого уровня управления является достаточно простой задачей и не требует чрезмерно больших затрат от бизнеса.

## 3.3. Бизнес-цель разработки программного обеспечения

Разработка данного программного обеспечения для поиска решения задачи оптимизации, при любых видах нелинейной целевой функции ***f(x),*** позволит **увеличить** производительность отдела исследований «Оптимизации» до уровня не менее 100 задач в час с допустимой погрешность не более 1E-28, без увеличения штаба сотрудников.

## 3.4. Пользовательская история разработки программного обеспечения

***Пользователь:***

* должен вводить аналитическое выражение для требуемой нелинейной функции ***f(x)***;
* должен вводить значение точки, **а** параболы;
* должен вводить значение точки **b** параболы;
* должен вводить значение точки **с** параболы;
* должен вводить значение допустимой погрешности ***Epsilon*** решения задачи;
* должен вводить допустимое максимальное затраты времени ***t\_max*** на поиск решения задачи;
* должен вводить допустимое максимальное количество ***k\_max*** итераций.
* определяет критерий поиска экстремума (найти минимум или максимум)

***Программа:***

* должна провести parsing – синтаксический анализ выражения введенной нелинейной функции (кроме натурального логарифма (ln) и экспоненты (exp)) и преобразовать аналитическую запись нелинейной функции в виде символов в вычисляемую процедуру в виде подпрограммы-функции;
* должна дать возможность ввести значение точки **a** параболы
* должна дать возможность ввести значение точки **b** параболы
* должна дать возможность ввести значение точки **с** параболы
* должна дать возможность ввести значение допустимой погрешности ***Epsilon***
* должна дать возможность ввести ограничения по количеству итераций ***k\_max***
* должна дать возможность ввести ограничения по количеству времени ***t\_max***
* должна найти экстремум заданной нелинейной функции **f(x)** по алгоритму поразрядного приближения (Successive Parabolic Interpolation Method)
* должна выводить найденное значение ***x\**** экстремума, заданного нелинейной функции;
* должна выводить значение заданной нелинейной функции ***f(x\*)*** в этой точке;
* должна выводить общее затраченное время (в миллисекундах) на поиск решения ***Elapsed Time*** для оценки производительности программы;
* должна выводить значение абсолютной погрешности решения ***Abs(x1–x0)***.

## 3.5. Функциональные требования к разработке программного обеспечения

**SRS\_1.1** - При наличии двух и более неизвестных аргументов, система должна сообщить об этом и дождаться подтверждения от пользователя.

**SRS\_1.2** – Система должна находить минимум или максимум функции по выбору пользователя.

**SRS\_1.3** - Если за заданное количество времени t\_max не достигнуто решение с требуемой погрешностью Epsilon, то система должна должна спросить у пользователя продолжить ли поиск, увеличив количество времени t\_max, или же остановить поиск, затем дождаться ответа от пользователя.

1) Если продолжить поиск, то система должна увеличить в два раза время работы и запустить программу.

2) Если остановить поиск при недостигнутой погрешности, то система должна предупреждать пользователя о том, что полученное решение задачи не удовлетворяет требуемой точности решения по причине недостаточного заданного ограничения на длительность времени работы программы;

**SRS\_1.4** - Система должна обеспечить вывод результата решения x\* и значения функции в этой точке f(x\*);

**SRS\_1.5** - Система должна обеспечить оценку качества решения задачи заданным методом (Successive Parabolic Interpolation Method) путем измерения и вывода количества времени потраченных на поиск решения с заданной погрешностью;

**SRS\_1.6** - Система должна принимать любое символьное выражение для нелинейной функции, которая имеет математический смысл. В случае ошибки во введенном аналитическом выражении нелинейной функции f(x) система должна выводить сообщение: “Функция f(x) не имеет математического смысла”

**SRS\_1.7** - Система должна обеспечить вывод достигнутого значения погрешности оптимального решения произвольного нелинейного уравнения в качестве оценки валидности результата.

**SRS\_1.8** – Система должна проверять значения введённых точек **a, b, c.** В случае если значения точек равны, система должна выводить сообщение: “Введите разные значения для точек a, b, c “

3.5.1. Требования к данным

**SRS\_2.1** - Система должна обеспечивать возможность ввода любых числовых значений допустимой погрешности Epsilon как в формате с фиксированной точностью “0.0000000000000000000000000001”, так и в экспоненциальном формате “1e-28”.

**SRS\_2.2** - Система должна обеспечивать ввод любого числового значения ***a, b, c***;

**SRS\_2.3** - Поскольку функция одномерная, то в качестве аргумента функции может быть задана только одна буква латинского алфавита (x или y), как строчные (lower-case letter), так и прописные (upper-case letter).

## 3.6. Нефункциональные требования на разработку программного обеспечения

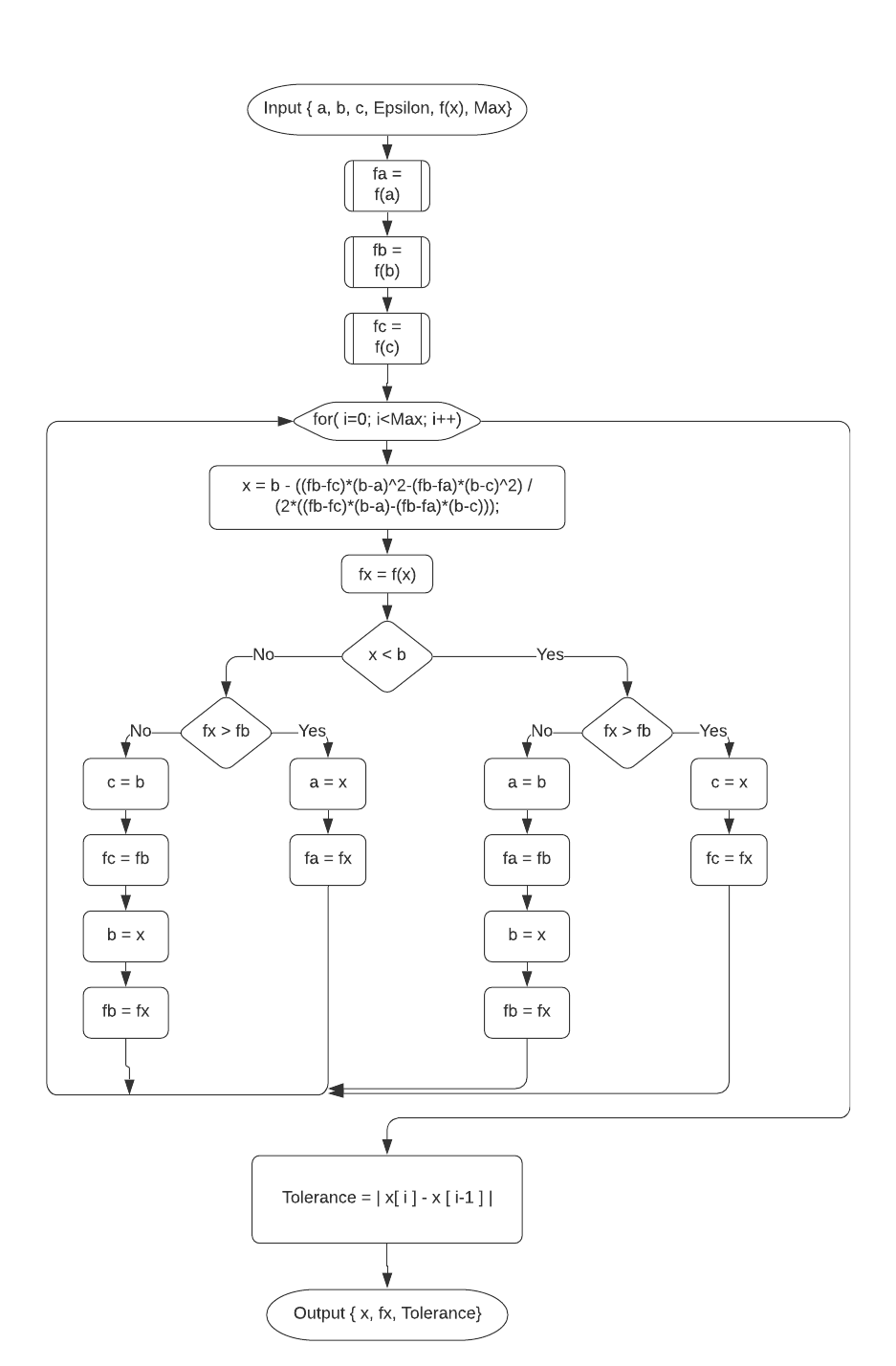
* Программа в процессе поиска решения должна отображать индикатор состояния процесса выполнения операции ***Progress Bar***, чтобы пользователь понимал, что система не зависла, а находится в рабочем состоянии;
* Система должна обеспечить возможность очищения текстовых полей.
* Сбой системы не должен происходить чаще чем один раз на тысячу запусков программы;
* Система не должна позволять пользователю изменять выходные данные;

## 3.7. Ограничения на разработку программного обеспечения

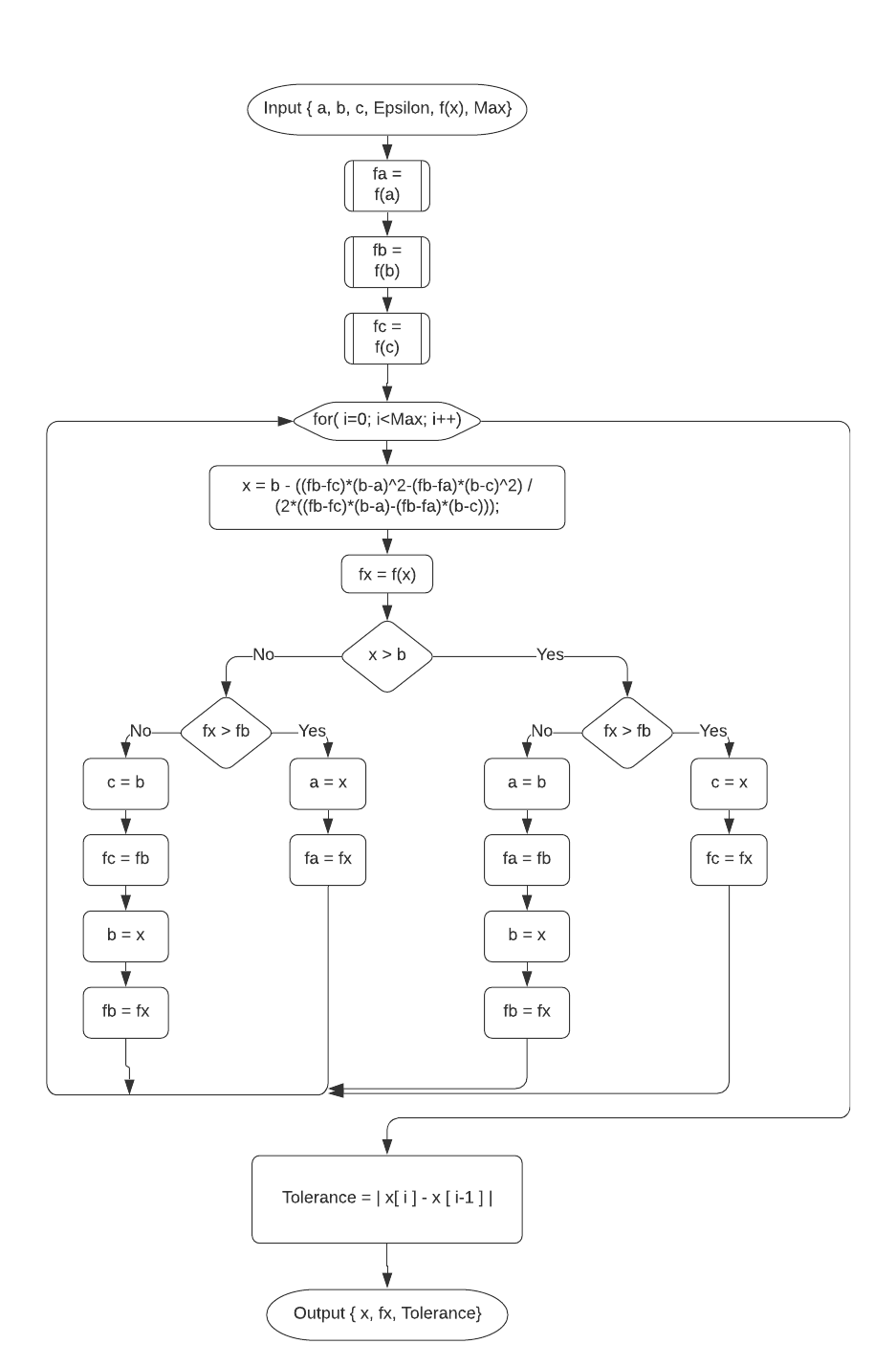
* Разработать ПО на языке JavaScript;
* Система должна находить оптимальное решение произвольного нелинейного уравнения с допустимой погрешностью не более 1e-28;
* Система должна находить оптимальное решение произвольного нелинейного уравнения с момента запуска процедуры вычисления до момента вывода оптимального решения за время не более одной минуты;
* Временные затраты на разработку ПО не должны превышать одного месяца;
* Система должна быть разработана для эксплуатации на компьютерах c операционной системой Windows7/8/8.1/10, с оперативной памятью не более 2 гигабайт;
* Система должна понимать аналитические выражения для нелинейной функции, которые включают следующие математические символы:
  + Стандартных функций – *sin, cos, tan, cot, asin, acos, atan, acot, log*;
  + Основных вычислительных операций: сложение (+), вычитание (-), умножение (\*), деление (/), и возведение в степень (^);
  + круглые скобки любой вложенности;

# Глава 4. Стадии проектирования системы.

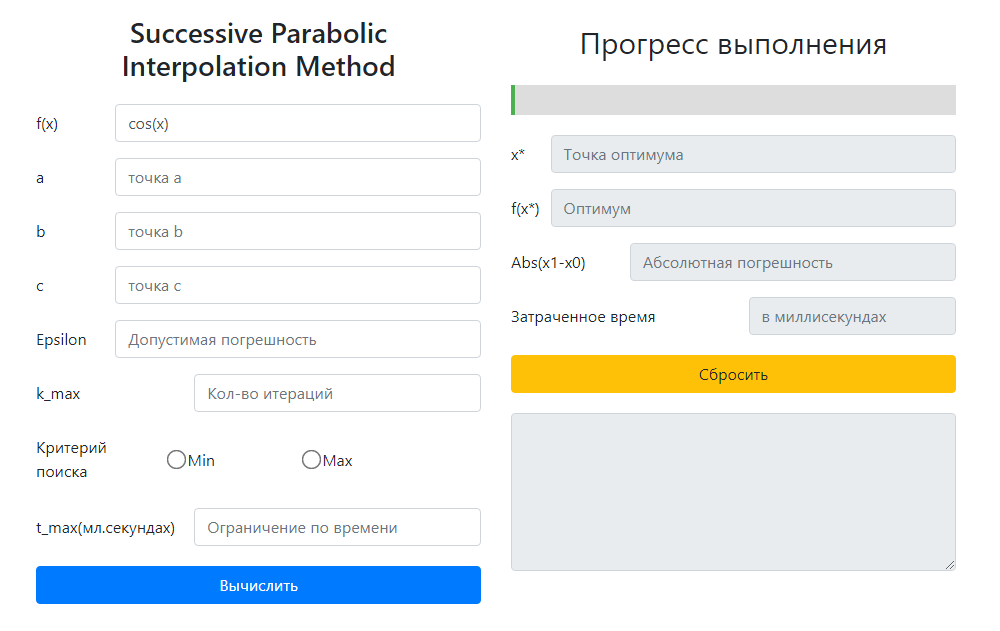
# 4.1. Блок-схема работы программы, реализующей Successive Parabolic Interpolation Method для поиска максимума.



# 4.2. Блок-схема работы программы, реализующей Successive Parabolic Interpolation Method для поиска минимума.



# 4.2. Проектирование интерфейса пользователя программы, реализующей Successive Parabolic Interpolation Method.

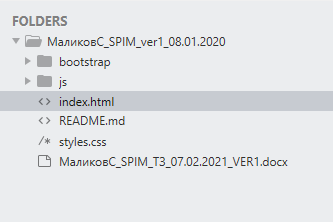


# 5. Настройки элементов управления формы, реализующей Successive Parabolic Interpolation Method.

|  |  |  |  |
| --- | --- | --- | --- |
| **Number of control** | **Control** | **Property** | **Setting** |
| 1 | Label | Appearance (Text) | f(x) |
| Design (Name) | fx |
| 3 | Label | Appearance (Text) | a |
| Design (Name) | a |
| 4 | Label | Appearance (Text) | Epsilon |
| Design (Name) | Epsilon |
| 5 | Label | Appearance (Text) | b |
| Design (Name) | b |
| 6 | Label | Appearance (Text) | c |
| Design (Name) | c |
| 7 | Label | Appearance (Text) | k\_max |
| Design (Name) | Max |
| 8 | Label | Appearance (Text) | Критерий поиска |
| Design (Name) | criterion |
| 9 | Label | Appearance (Text) | t\_max(мл.секундах) |
| Design (Name) | t\_max |
| 10 | TextBox | Appearance (Text) | cos(x) |
| Design (Name) | fx |
| 11 | TextBox | Appearance (Text) | Точка a |
| Design (Name) | a |
| 12 | TextBox | Appearance (Text) | Допустимая погрешность |
| Design (Name) | Epsilon |
| 13 | TextBox | Appearance (Text) | Точка b |
| Design (Name) | b |
| 14 | TextBox | Appearance (Text) | Точка с |
| Design (Name) | с |
| 15 | TextBox | Design (Name) | Кол-во итераций |
| Appearance (Text) | Max |
| 16 | RadioButton | Appearance (Text) | Min |
| Design (Name) | minimum |
| 16 | RadioButton | Appearance (Text) | Max |
| Design (Name) | maximum |
| 15 | TextBox | Design (Name) | Ограничение по времени |
| Appearance (Text) | t\_max |
| 13 | Button | Appearance (Text) | Вычислить |
| Design (Name) | Submit |
| 15 | ProgressBar | Design (Name) | progressBar |
| Appearance (Text) |  |
| 19 | Label | Appearance (Text) | x\* |
| Design (Name) | x |
| 20 | Label | Design (Name) | fx1 |
| Appearance (Text) | f(x\*) |
| 21 | Label | Appearance (Text) | Abs(x1-x0) |
| Design (Name) | absTol |
| 22 | Label | Design (Name) | timeWaste |
| Appearance (Text) | Затраченное время: |
| 23 | TextBox | Appearance (Text) | Точка оптимума |
| Design (Name) | x |
| Behavior (ReadOnly) | True |
| 24 | TextBox | Design (Name) | fx1 |
| Appearance (Text) | Оптимум |
| Behavior (ReadOnly) | True |
| 25 | TextBox | Appearance (Text) | Абсолютная погрешность |
| Design (Name) | absTol |
| Behavior (ReadOnly) | True |
| 26 | TextBox | Design (Name) | timeWaste |
| Appearance (Text) | в миллисекундах |
| Behavior (ReadOnly) | True |
| 27 | Button | Appearance (Text) | Сбросить |
| Design (Name) | Reset |
| 26 | TextArea | Design (Name) | OutputText |
| Appearance (Text) |  |
| Behavior (ReadOnly) | True |

# 5.1. Код программы, реализующей Successive Parabolic Interpolation Method.

Шаг 1: Код программы для index.tml, где хранятся элементы управления.



<!DOCTYPE html>

<html ng-app="main">

<head>

<title>Successive Parabolic Interpolation Method</title>

<meta charset="utf-8">

<meta name="viewport" content="initial-scale=1">

<meta http-equiv="Cache-Control" content="no-cache, no-store, must-revalidate">

<meta name="description" content="">

<meta name="viewport" content="initial-scale=1, maximum-scale=1, user-scalable=no">

<!-- styles -->

<link rel="stylesheet" href="styles.css">

<!--Bootstrap -->

<script src="bootstrap/jquery-3.2.1.slim.min.js"></script>

<script src="bootstrap/popper.min.js"></script>

<script src="bootstrap/bootstrap.min.js"></script>

<link rel="stylesheet" href="bootstrap/bootstrap.min.css">

<!-- mathjs -->

<script src="js/Math.js"></script>

<!-- Plot math -->

<script src="js/Plot.js"></script>

<!-- SPIM-->

<script src="js/SuccessiveParabolicInterpolationMethod.js"></script>

</head>

<body>

<div class="container">

<form name="input">

<div class="row">

<div class="col-1">

</div>

<div class="col-5">

<h3>Successive Parabolic Interpolation Method</h3>

<div class="form-group row">

<label for="fx" class="col-sm-2 col-form-label">f(x)</label>

<div class="col-sm-10">

<div class="dropdown">

<input type="text" name="dropdownMenuButton" class="form-control dropdown-toggle" id="dropdownMenuButton" data-toggle="dropdown" aria-haspopup="true" aria-expanded="false" value="cos(x)">

<div class="dropdown-menu" aria-labelledby="dropdownMenuButton">

<a class="dropdown-item" onclick="fx\_dropdown('0.01\*x^3+x^2-2\*x-2');">0.01\*x^3+x^2-2\*x-2</a>

<a class="dropdown-item" onclick="fx\_dropdown('log(2^(8\*x-4))');">log(2^(8\*x-4))</a>

<a class="dropdown-item" onclick="fx\_dropdown('x^2-4\*sin(x)');">x^2-4\*sin(x)</a>

<a class="dropdown-item" onclick="fx\_dropdown('4\*x^3-2\*x-6');">4\*x^3-2\*x-6</a>

<a class="dropdown-item" onclick="fx\_dropdown('1500\*x^3-0.001\*x^2-0.01\*x');">1500\*x^3-0.001\*x^2-0.01\*x</a>

</div>

</div>

</div>

</div>

<div class="form-group row">

<label for="a" class="col-sm-2 col-form-label">a</label>

<div class="col-sm-10">

<input type="text" name="a" id="a" class="form-control" placeholder="точка a">

</div>

</div>

<div class="form-group row">

<label for="b" class="col-sm-2 col-form-label">b</label>

<div class="col-sm-10">

<input type="text" name="b" id="b" class="form-control" placeholder="точка b">

</div>

</div>

<div class="form-group row">

<label for="c" class="col-sm-2 col-form-label">c</label>

<div class="col-sm-10">

<input type="text" name="c" id="c" class="form-control" placeholder="точка с">

</div>

</div>

<div class="form-group row">

<label for="Epsilon" class="col-sm-2 col-form-label">Epsilon</label>

<div class="col-sm-10">

<input type="text" name="Epsilon" id="Epsilon" class="form-control" placeholder="Допустимая погрешность">

</div>

</div>

<div class="form-group row">

<label for="Max" class="col-sm-4 col-form-label">k\_max</label>

<div class="col-sm-8">

<input type="text" name="Max" id="Max" class="form-control" placeholder="Кол-во итераций">

</div>

</div>

<div class="form-group row">

<label for="criterion" class="col-sm-2 col-form-label">Критерий поиска</label>

<div id="radiobutton">

<input type="radio" name="criterion" value="minimum"> Min

</div>

<div id="radiobutton">

<input type="radio" name="criterion" value="maximum"> Max

</div>

</div>

<div class="form-group row">

<label for="t\_max" class="col-sm-4 col-form-label">t\_max(мл.секундах)</label>

<div class="col-sm-8">

<input type="text" name="t\_max" id="t\_max" class="form-control" placeholder="Ограничение по времени">

</div>

</div>

<input type="button" name="subimt" class="btn btn-primary btn-block margened" value="Вычислить" onclick="SPIM();">

</div>

<div class="col-5">

<!-- Output -->

<p id="progressBar">Прогресс выполнения</p>

<div id="myProgress">

<div id="myBar"></div>

</div>

<div class="form-group row">

<label for="x1" class="col-sm-1 col-form-label">x\*</label>

<div class="col-sm-11">

<input type="text" name="x1" id="x1" class="form-control" placeholder="Точка оптимума" readonly>

</div>

</div>

<div class="form-group row">

<label for="yf1" class="col-sm-1 col-form-label">f(x\*)</label>

<div class="col-sm-11">

<input type="text" name="yf1" id="yf1" class="form-control" placeholder="Оптимум" readonly>

</div>

</div>

<div class="form-group row">

<label for="absTol" class="col-sm-3 col-form-label">Abs(x1-x0)</label>

<div class="col-sm-9">

<input type="text" name="absTol" id="absTol" class="form-control" placeholder="Абсолютная погрешность" readonly>

</div>

</div>

<div class="form-group row">

<label for="timeWaste" class="col-sm-6 col-form-label">Затраченное время</label>

<div class="col-sm-6">

<input type="text" name="timeWaste" id="timeWaste" size="40" class="form-control" placeholder="в миллисекундах" readonly>

</div>

</div>

<input type="reset" value="Сбросить" class="btn btn-warning btn-block margened">

<div class="form-group">

<textarea class="form-control" id="OutputText" rows="6" readonly></textarea>

</div>

</div>

</div>

</form>

</div>

<!-- Output Plot -->

<div id="plot"></div>

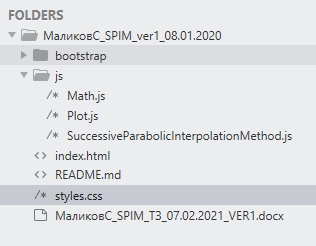
<!-- Output Table -->

<table id="myTable"></table>

</body>

</html>

Шаг 2: Код программы для style.css, где хранятся все стили страницы.



div{

background-color: ;

}

table {

font-family: arial, sans-serif;

border-collapse: collapse;

width: 100%;

}

td,

th {

border: 1px solid #dddddd;

text-align: left;

padding: 5px;

}

tr:nth-child(even) {

background-color: #dddddd;

}

#myProgress {

width: 100%;

background-color: #ddd;

margin: 20px 0px 20px 0px;

}

#myBar {

width: 1%;

height: 30px;

background-color: #4CAF50;

}

.margened{

margin:20px 0px 20px 0px;

}

#radiobutton{

margin: 20px 20px 20px 70px;

}

input[type="radio"] {

-ms-transform: scale(1.5); /\* IE 9 \*/

-webkit-transform: scale(1.5); /\* Chrome, Safari, Opera \*/

transform: scale(1.5);

}

#progressBar{

margin: 20px;

}

p{

text-align: center;

font-size: 30px;

}

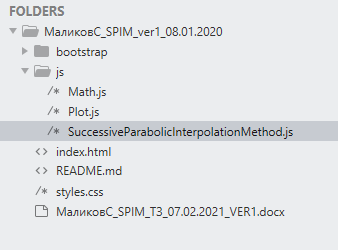
h3{

text-align: center;

padding: 15px;

}

Шаг 3: Код программы для Successive Parabolic Interpolation Method.js, реализующий нахождение оптимального решения произвольного нелинейного уравнения методом Successive Parabolic Interpolation Method.



function SPIM() {

// Переменные считанные с панели ввода

let Epsilon = document.getElementById("Epsilon").value;

Epsilon = parseFloat(Epsilon).toFixed(28); // Максимальная допустимая погрешность Tolerance

let a = parseFloat(document.getElementById("a").value);

let b = parseFloat(document.getElementById("b").value);

let c = parseFloat(document.getElementById("c").value);

let Max = document.getElementById("Max").value;

let fx = document.getElementById("dropdownMenuButton").value;

let t\_max = document.getElementById("t\_max").value;

try { // Проверка на корректность искомой функции

solve(a, fx);

} catch(err) {

alert('Функция f(x) не имеет математического смысла');

document.getElementById("OutputText").value ="Функция f(x) не имеет математического смысла.";

document.getElementById("OutputText").style.color = 'red';

return 0;

}

// Промежуточные переменные и переменные для вывода

let x0;

let fx0;

let fa = solve(a, fx);

let fb = solve(b, fx);

let fc = solve(c, fx);

let iterations = 0;

let arr\_x = [];

let results = []; // Результат выполения поитерационно

var time = 0; // Подсчет времени

if(fx.match(/x/) == "x"){ // Верификация №1

if(fx.match(/y/) == "y"){

alert("Функция должна иметь только одну переменную");

document.getElementById("OutputText").value ="Функция должна иметь только одну переменную.";

document.getElementById("OutputText").style.color = 'red';

return 0;

}

}

if(Max<=0){ //Проверка положительности кол-во итераций

alert("Кол-во итераций должно быть больше нуля");

document.getElementById("OutputText").value ="Кол-во итераций должно быть больше нуля.";

document.getElementById("OutputText").style.color = 'red';

return 0;

}

var radio\_btn = document.getElementsByName('criterion'); // Обработка радио кнопки(максимум, минимум)

for(i = 0; i < radio\_btn.length; i++) {

if(radio\_btn[i].checked){

if(radio\_btn[i].value == "minimum"){ // Если критерий был выбран на поиск минимума

time = performance.now(); // начинается подсчет времени выполнения алгоритма

while(iterations<Max){

iterations++;

x0 = b - ((fb-fc)\*(b-a)^2-(fb-fa)\*(b-c)^2)/(2\*((fb-fc)\*(b-a)-(fb-fa)\*(b-c)));

fx0 = solve(x0, fx);

if(x0>b){

if(fx0>fb){

c = x0;

fc = fx0;

}

else{

a = b;

fa = fb;

b = x0;

fb = fx0;

}

}

else{

if(fx0>fb){

a = x0;

fa = fx0;

}

else{

c = b;

fc = fb;

b = x0;

fb = fx0;

}

}

arr\_x.push(x0);

let result = {

n: iterations,

x0: x0,

Epsilon: Epsilon,

fx0: fx0

}

results.push(result);

}

time = performance.now() - time; // окончание подсчета времени

if(time>t\_max){ // Верификация №2

if(confirm("Решение не найдено за заданное кол-во времени, хотите увеличить время в двое и продолжить поиск корня?")){

t\_max = t\_max\*2;

document.getElementById("t\_max").value = t\_max;

SPIM();

}

else{

document.getElementById("OutputText").value ="Решение не найдено с заданной точностью "+document.getElementById("Epsilon").value+" за заданное кол-во времени.";

document.getElementById("OutputText").style.color = 'red';

return 0;

}

}

else{

if(Math.abs(math.eval((arr\_x[arr\_x.length-1])-(arr\_x[arr\_x.length-2])))<Epsilon){

document.getElementById("OutputText").value ="Решение найдено для минимума функции: "+fx+"\n"+"Данное решения является валидным т.к выполняется условие ABS(x1-x0)<Tolerance:"+"\n"+Math.abs(math.eval((arr\_x[arr\_x.length-1])-(arr\_x[arr\_x.length-2]))).toFixed(28)+"<"+Epsilon+". Затрачено "+time.toFixed(3)+" миллисекунд";

document.getElementById("OutputText").style.color = '#03a30e';

}

else{

document.getElementById("OutputText").value ="Решение не найдено с заданной точностью за заданное кол-во итераций";

document.getElementById("OutputText").style.color = 'red';

}

}

}

else{ // Если критерий был выбран на поиск максимума

time = performance.now(); // начинается подсчет времени выполнения алгоритма

while(iterations<Max){

iterations++;

let result = {

n: iterations,

x1: x1,

Epsilon: Epsilon,

yf1: yf1

}

results.push(result);

}

time = performance.now() - time; // окончание подсчета времени

if(time>t\_max){ // Верификация №2

if(confirm("Решение не найдено за заданное кол-во времени, хотите увеличить время в двое и продолжить поиск корня?")){

t\_max = t\_max\*2;

document.getElementById("t\_max").value = t\_max;

SPIM();

}

else{

document.getElementById("OutputText").value ="Решение не найдено с заданной точностью "+document.getElementById("Epsilon").value+" за заданное кол-во времени.";

document.getElementById("OutputText").style.color = 'red';

return 0;

}

}

/\* else{

if(Math.abs(math.eval(x1-x0))<Epsilon){

document.getElementById("OutputText").value ="Решение найдено для максимума функции: "+fx+"\n"+"Данное решения является валидным т.к выполняется условие ABS(x1-x0)<Tolerance:"+"\n"+Math.abs(math.eval(x1-x0)).toFixed(28)+"<"+Epsilon+". Затрачено "+time.toFixed(3)+" миллисекунд";

document.getElementById("OutputText").style.color = '#03a30e';

}

else{

document.getElementById("OutputText").value ="Решение не найдено с заданной точностью за заданное кол-во итераций";

document.getElementById("OutputText").style.color = 'red';

}

}\*/

}

}

}

var i = 0;

if (i == 0) { //Прогресса выполения(Progress Bar)

i = 1;

var elem = document.getElementById("myBar");

var width = 1;

var id = setInterval(frame, 10);

function frame() {

if (width >= 100) {

clearInterval(id);

i = 0;

} else {

width++;

elem.style.width = width + "%";

}

}

}

document.getElementById("absTol").value = Math.abs(math.eval((arr\_x[arr\_x.length-1])-(arr\_x[arr\_x.length-2])));

document.getElementById("x1").value = x0;

document.getElementById("yf1").value = fx0;

document.getElementById("timeWaste").value = time;

drawTable(results); // Вывод данных на таблицу

drawGraph(fx);

}

function solve(value, equation) {

return math.eval(equation.replace(/[xyXY]+/g, value));

}

function drawTable(results) {

let table = document.getElementById("myTable");

table.innerHTML = "";

let row;

results.reverse();

results.filter(function(result) {

row = table.insertRow(0);

row.insertCell(0).appendChild(document.createTextNode(result.n));

row.insertCell(1).appendChild(document.createTextNode(result.x0));

row.insertCell(2).appendChild(document.createTextNode(result.fx0));

});

row = table.insertRow(0);

row.insertCell(0).appendChild(document.createTextNode("n"));

row.insertCell(1).appendChild(document.createTextNode("x\*"));

row.insertCell(2).appendChild(document.createTextNode("f(x)\*"));

}

function drawGraph(equation) {

try {

const expr = math.compile(equation)

const xValues = math.range(-10, 10, 0.5).toArray()

const yValues = xValues.map(function(x) {

return expr.eval({ x: x })

})

const trace1 = {

x: xValues,

y: yValues,

type: 'scatter'

}

const data = [trace1]

Plotly.newPlot('plot', data)

} catch (err) {

console.error(err)

}

}

function fx\_dropdown(equation){

document.getElementById("dropdownMenuButton").value = equation;

}

# 5.2. Подключенные библиотеки.

Math.js это библиотека на языке JavaScript, для парсинга функций.

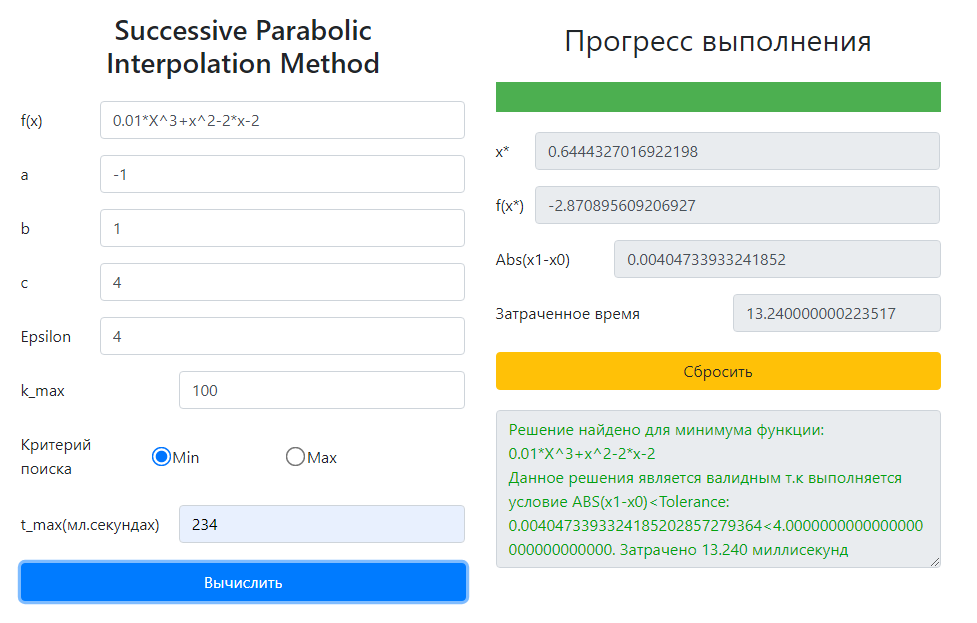
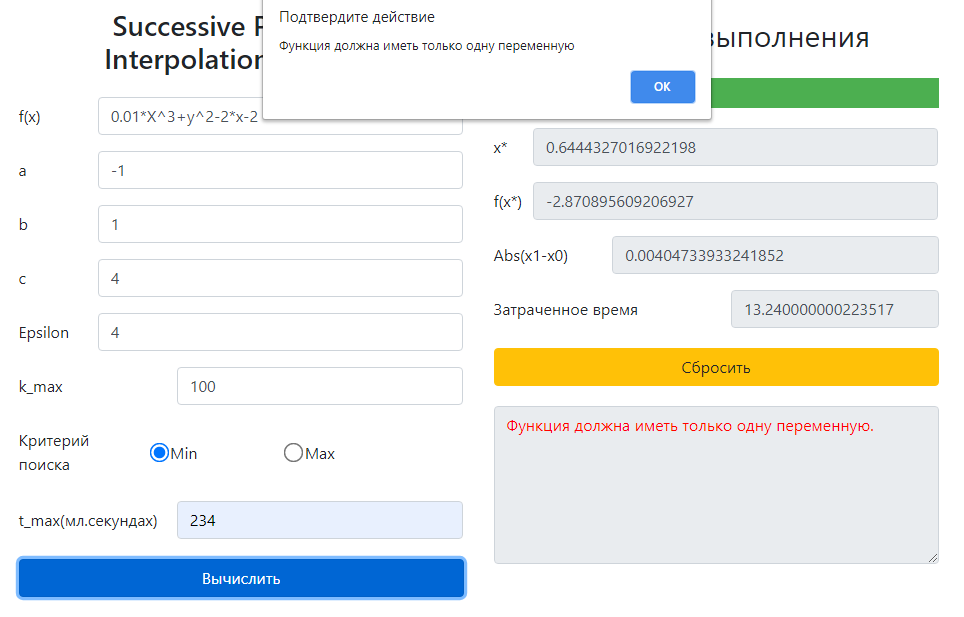
Доступно на https://mathjs.org/download.html

Plot.js это библиотека для построения графика функций. Доступно на https://plotly.com/javascript/

# Глава 6. Верификация ПО.

Верификация №1. Проверка:

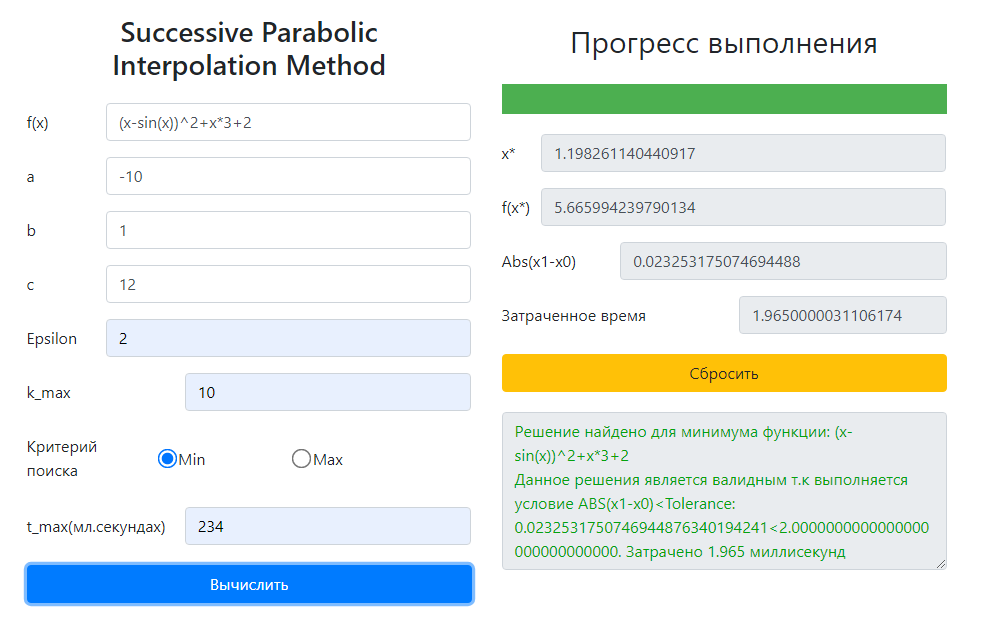
SRS\_1.1 – При наличии двух и более неизвестных аргументов, система должна сообщить об этом и ждать подтверждения от пользователя.

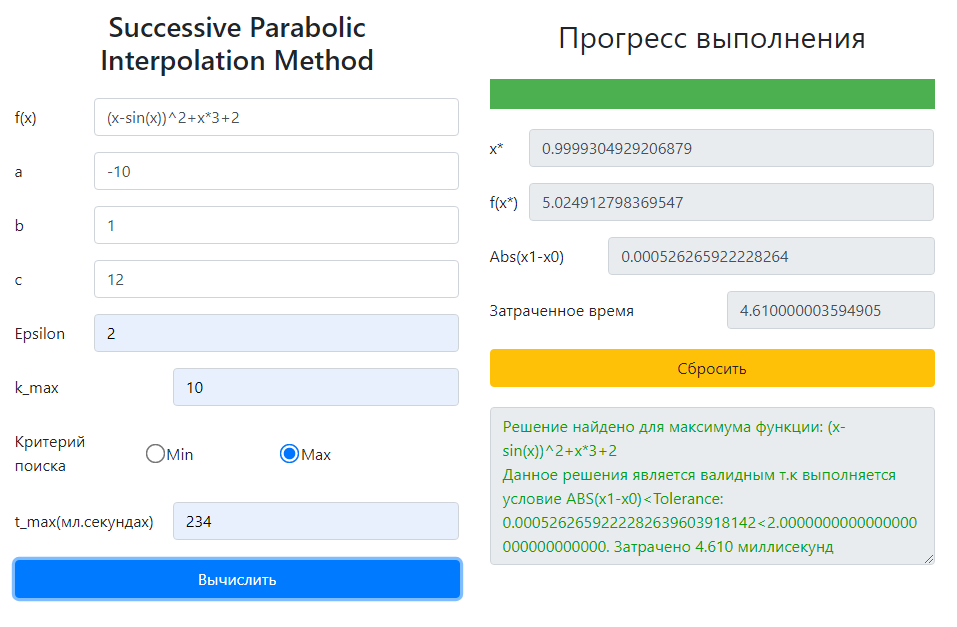
 

Вывод: в качестве аргумента функции система принимает только одну любую букву латинского алфавита, как строчные (lower-case letter), так и прописные (upper-case letter).

Верификация №2. Проверка:

SRS\_1.2 – Система должна находить минимум или максимум функции по выбору пользователя.





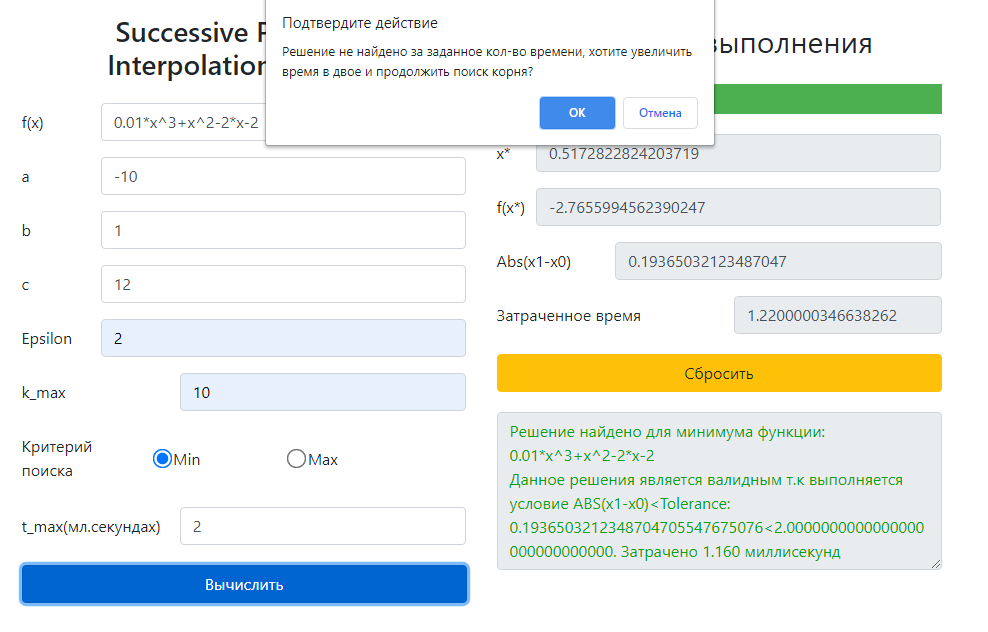
Вывод: система находит минимум или максимум функции по выбору пользователя.

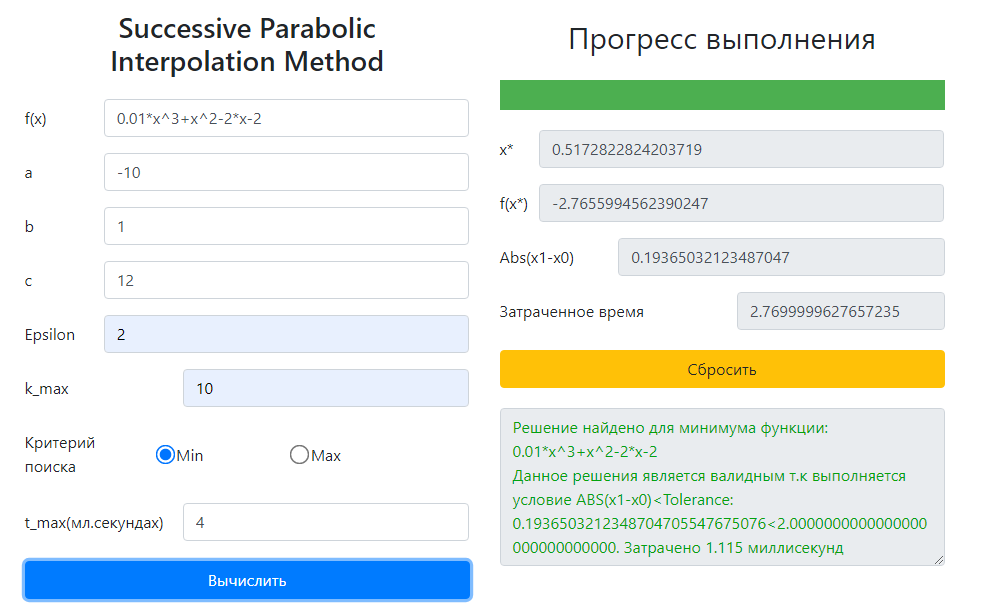
Верификация №3. Проверка:

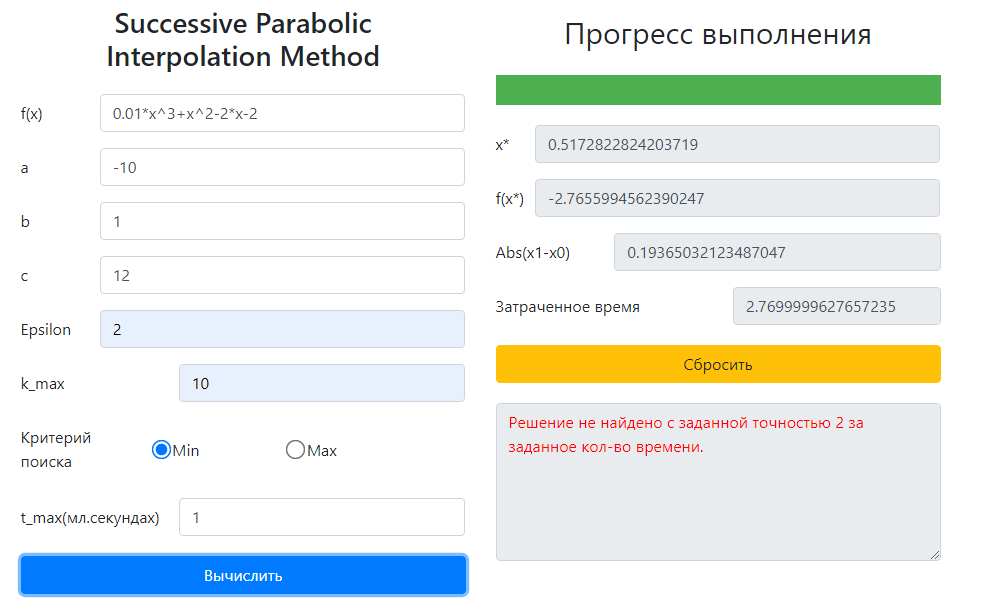
SRS\_1.3 - Если за заданное количество времени t\_max не достигнуто решение с требуемой погрешностью Epsilon, то система должна спросить у пользователя продолжить ли поиск, увеличив количество времени t\_max, или же остановить поиск, затем дождаться ответа от пользователя.

1) Если продолжить поиск, то система должна увеличить в два раза время работы и запустить программу.

2) Если остановить поиск при недостигнутой погрешности, то система должна предупреждать пользователя о том, что полученное решение задачи не удовлетворяет требуемой точности решения по причине недостаточного заданного ограничения на длительность времени работы программы;



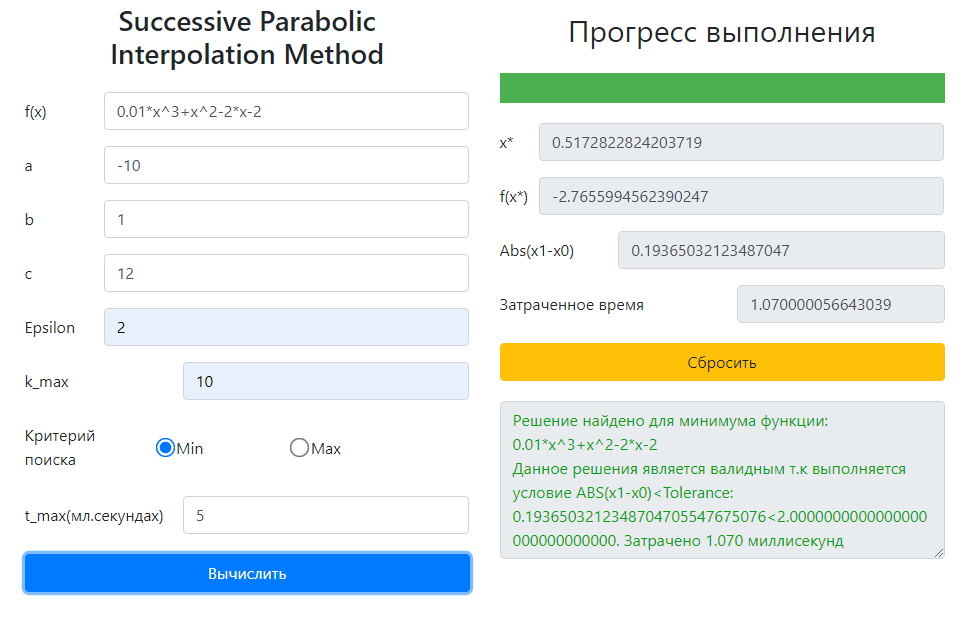




Вывод: если система не находит корень за заданное количество времени, то она предлагает увеличить количество времени или же прекратить.

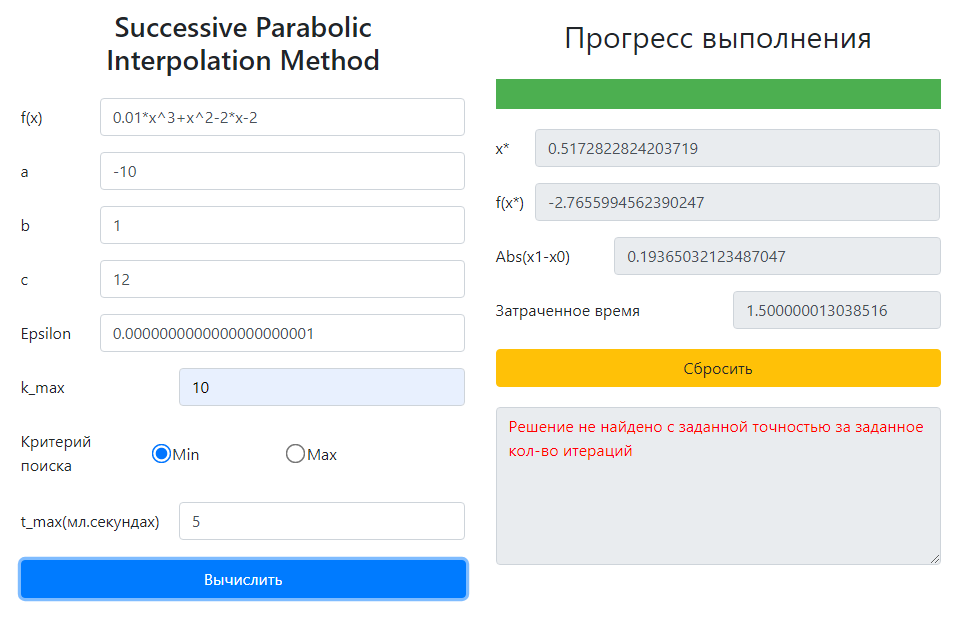
Верификация №6. Проверка:

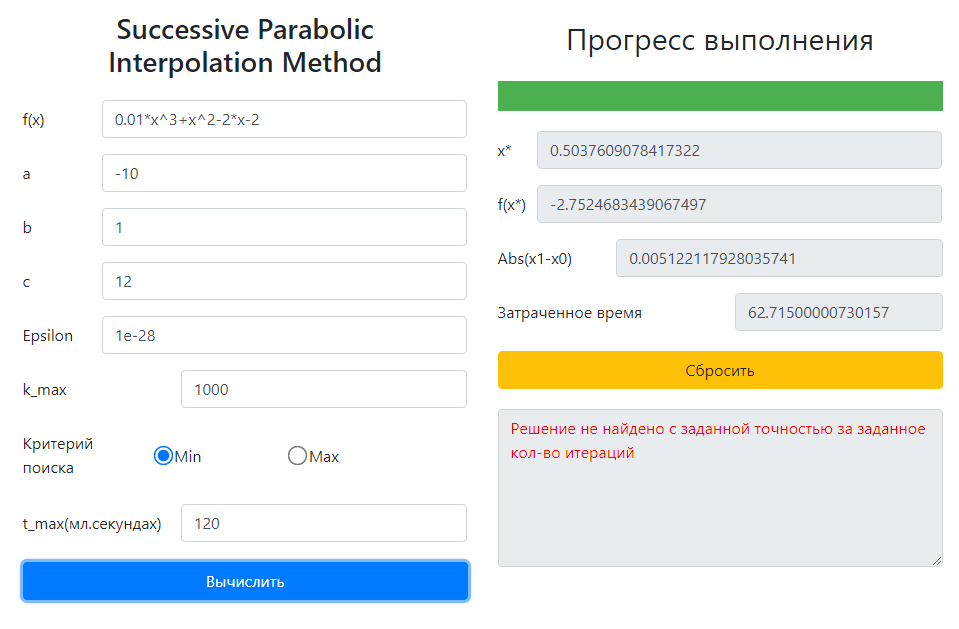
**SRS\_1.5** - Система должна обеспечить оценку качества решения задачи заданным методом (**Successive Parabolic Interpolation Method**) путем измерения и вывода количества времени потраченных на поиск решения с заданной погрешностью;



Верификация №7. Проверка:

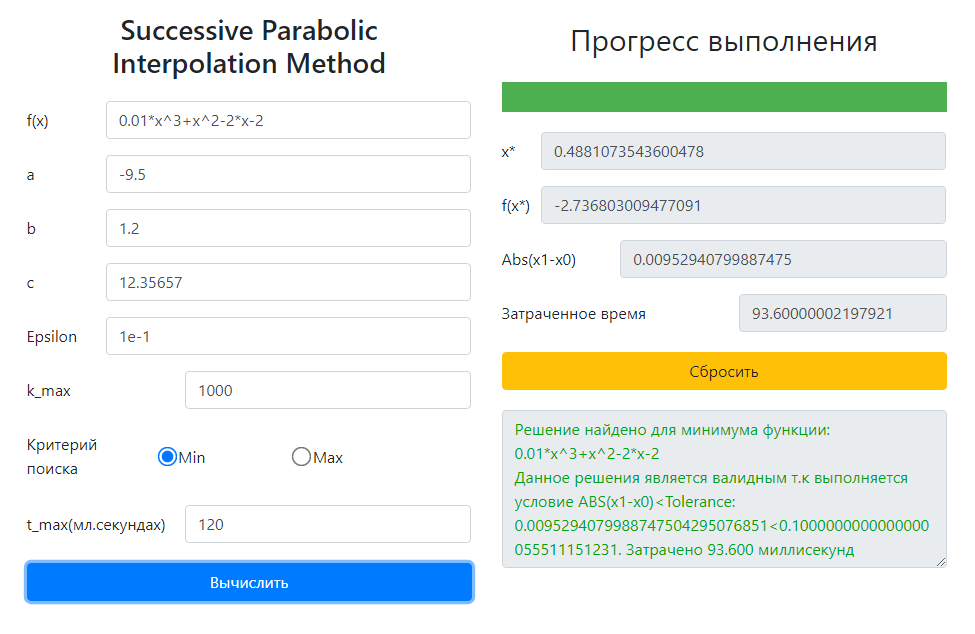
SRS\_2.1 - Система должна обеспечивать возможность ввода любых числовых значений допустимой погрешности Epsilon как в формате с фиксированной точностью “0.0000000000000000000000000001”, так и в экспоненциальном формате “1e-28”.





Верификация №8. Проверка:

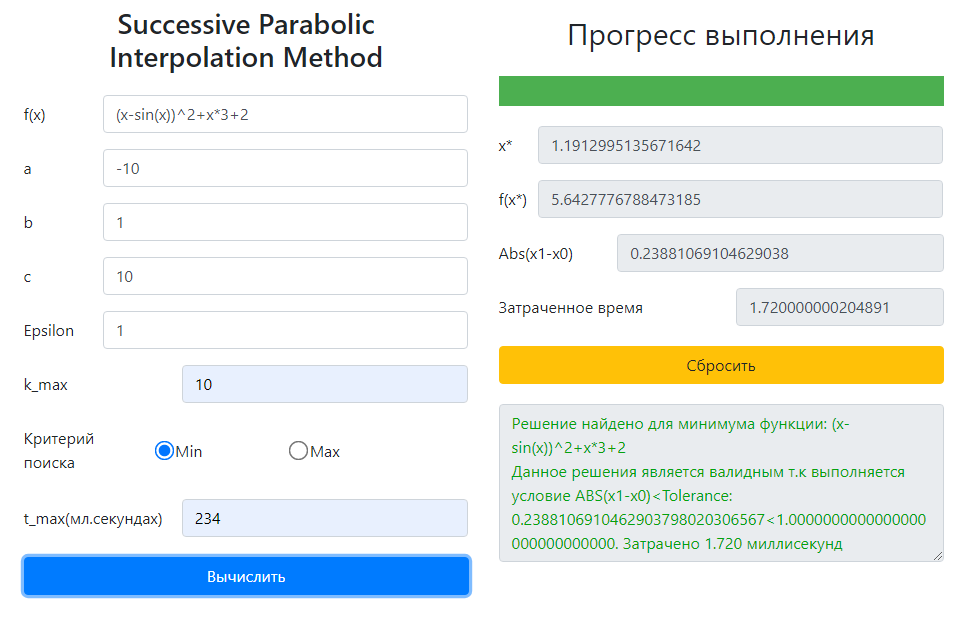
SRS\_2.2 - Система должна обеспечивать ввод любого числового значения ***a, b, c***;

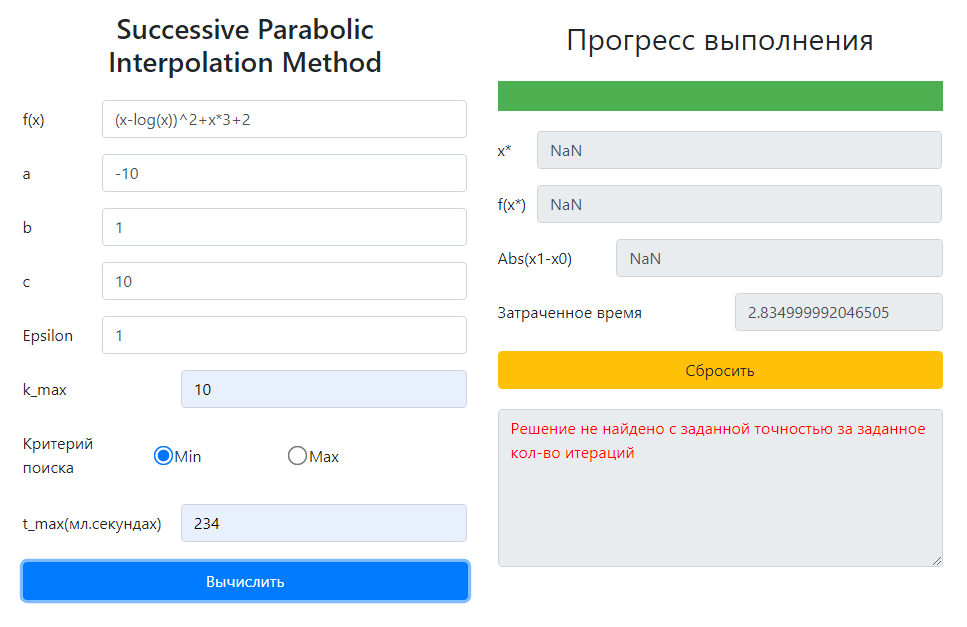


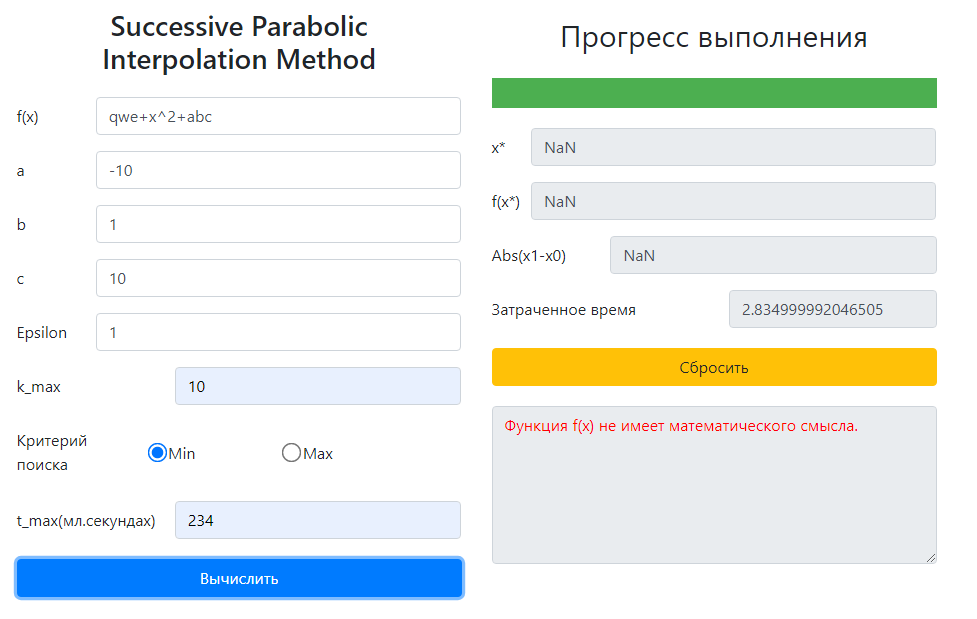
Вывод: система обеспечивает ввод любого начального числового значения ***x.***

Верификация №9. Проверка:

SRS\_2.3 - Система должна принимать любое символьное выражение для нелинейной функции, которая имеет математический смысл;







# Глава 7. Оценка достижения бизнес-цели разработки программного обеспечения

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Функция | Значение точки a | Значение точки b | Значение точки c | Погрешность | Вывод | Потраченное время(мсек) 1сек = 1000мсек |
| x^2-4\*sin(x) | -6 | -2 | 3 | 1e-28 | Решение не найдено | - |
| 0.01\*x^3+x^2-2\*x-2 | -3 | 1 | 3 | 1e-28 | Решение не найдено | - |
| -4\*x^2+2\*x+6 | -10 | 1 | 10 | 1e-28 | Решение не найдено | - |
| (x-sin(x))^2+x\*3+2 | -8 | 3 | 7 | 1e-28 | Решение не найдено | - |
| log(2^(8\*x-4)) | 2 | 5 | 12 | 1e-28 | Решение не найдено | - |

Из таблицы видно, что программа не может решать задачи с заданной точностью 1e-28, т.е продукт не соответствует заданным критериям бизнес цели, это означает не пригодность ПО в использовании заказчиком и не соответствует ожиданиям и требованиям.

Как итог, после тестирования можно сказать, что данное ПО не прошло валидацию.