Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий

Кафедра вычислительной техники

УТВЕРЖДАЮ
Заведующий кафедрой
_____ А.И.Легалов
подпись инициалы, фамилия
«____» _____ 2017 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.01 Информатика и вычислительная техника

Библиотека стандартных функций для функционально-потокового языка параллельного программирования

Руководитель	_____	_____	Матковский И.В
	подпись, дата	должность, ученая степень	инициалы, фамилия
Консультант	_____	_____	Доррер Г.А.
	подпись, дата	должность, ученая степень	инициалы, фамилия
Выпускник	_____		Полтавец К.В.
	подпись, дата		инициалы, фамилия
Нормоконтроль	_____		_____
	подпись, дата		инициалы, фамилия

Красноярск 2017

СОДЕРЖАНИЕ

ВВЕДЕНИЕ

Существует несколько подходов к программированию, один из них - параллельный, при котором существует возможность исполнения несколько частей программы одновременно. Изначально параллельное программирование было делом только специалистов, которых волновали задачи для больших суперкомпьютеров. Теперь же, когда на многоядерных процессорах начали работать обычные приложения, параллельное программирование быстро становится технологией, которую должен освоить и уметь применять любой профессиональный разработчик ПО.

Для создания параллельных программ используются языки функционального программирования, в которых выполнение каждого оператора осуществляется по готовности его данных, что является очень удобным для решения различных вычислительных задач и задач, связанных с обработкой данных. Так же эти языки позволяют писать программы в независимости от архитектуры, под которую они разрабатываются, и лишь после отладки программ адаптировать их под конкретные устройства или архитектуры. В число таких языков входят Haskell и Пифагор.

Пифагор - функционально-потоковый язык программирования, предназначенный для разработки архитектурно-независимых параллельных программ. Название является сокращением фразы «**П**араллельный **И**нформационно-**Ф**ункциональный **А**л**ГОР**итмический» или «**P**arallel **I**nformational and **F**unctional **A**l**GOR**ithmic» [4]. Использование этого языка позволяет при разработке программ не учитывать ресурсные ограничения. Ресурсные ограничения учитываются на этапе выполнения, осуществляется сжатие параллелизма. Эта позволяет обеспечивает архитектурную независимость разрабатываемых программ. Однако в данном языке существует проблема с недостаточно полной библиотекой стандартных функций, что является минусом при разработке достаточно сложных программ.

Библиотека стандартных функций подразумевает собой файлы с набором уже готового программного кода, решаемого определенные задачи. Для использования этих библиотек необходимо указать эти файлы в заголовке программы и использовать вызов необходимых функций.

Из этого следует что необходимо изучить стандартные библиотеки в других языках программирования, доработать имеющую библиотеку языка Пифагор, проанализировав его, а также полностью описать созданные функции и привести примеры их использования.

1 Анализ программного обеспечения и постановка задач

1.1 Язык программирования Пифагор

Для решения задач путем параллельных вычислений в 1995 в Красноярском Государственном Техническом Университете был разработан функциональный язык параллельного программирования Пифагор. Его разработка велась при учете других разработок в этой предметной области, а также ряда функциональных и потоковых моделей параллельных вычислений и языков программирования, построенных на их основе.

Язык Пифагор обладает следующими особенностями:

- архитектурная независимость, достигаемая за счёт описания в программе только информационных связей;
- асинхронный параллелизм, поддерживаемый выполнением операций по готовности данных;
- параллелизм на уровне операций;
- отсутствие переменных, что позволяет избежать конфликтов, связанных с совместным использованием памяти параллельными процессами;

Несмотря на эффективность при разработке параллельных программ языку Пифагор присуще некоторые недостатки - отсутствуют средства контроля типов и определения пользовательских типов и сложных структур данных.

Также общим минусом языков как последовательного, так и параллельного программирования, является то что параллельная программа трудно поддается линейному представлению в виде текста, однако двумерное описание параллельных программ не распространено, несмотря на всеобщее использование визуального программирования.

Пример описания функции вычитания чисел списка:

```
Sub << funcdef Parametrs
{
    Par1 << Parametrs : 1;
    Par2 << Parametrs : 2;
    (Par1, Par2) : - >> return
}
```

Из списка Parametrs выделяется первый и второй элементы. Следующим этапом возвращается из функции вычитание элементов списка. Вызов функции выглядит так:

```
(14,5): Sub
```

Результатом будет: 4

Так как язык Пифагор функционально-потоковый в нём отсутствует стандартное описания цикла. Но цикл можно организовать с помощью рекурсии, то есть обращение функции к самой себе. Для того чтобы такое обращение не было бесконечным, в тексте функции должно быть условие, по достижению которого дальнейшее обращение к подпрограмме не происходит.

IDE Pifagor представляет собой интегрированную среду разработки, предназначенный для разработки и отладки программ на языке программирования Пифагор (рисунок 1).

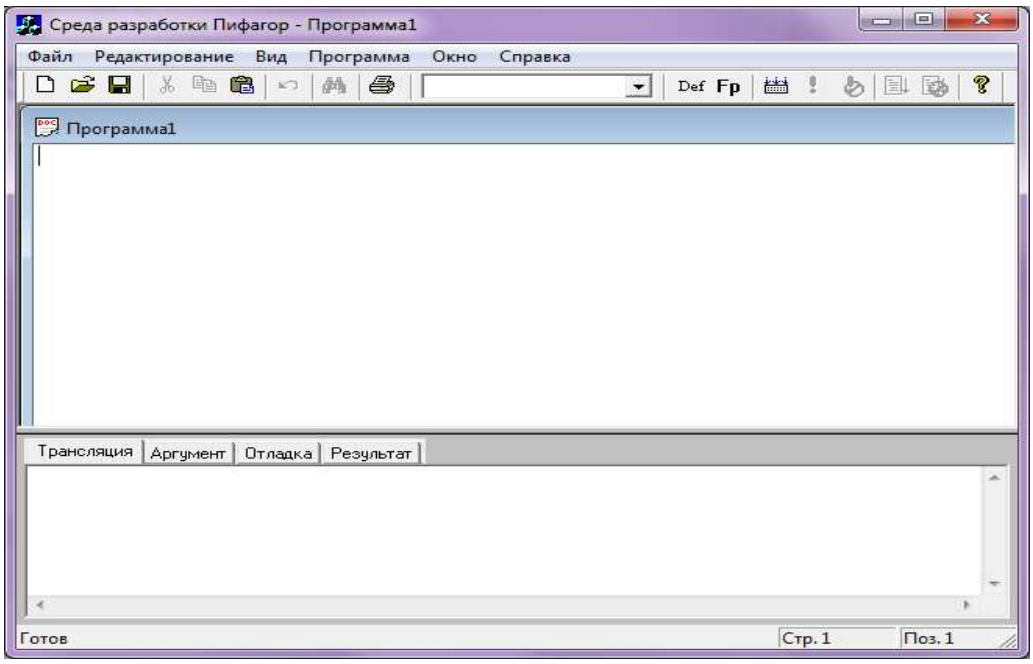

Рисунок 1 – IDE Pifagor

1.2 Библиотеки стандартных функций

Библиотеки стандартных функций языка программирования - набор модулей, классов, объектов, констант, глобальных переменных, шаблонов, макросов, функций и процедур, доступных для вызова из любой программы, написанной на этом языке и присутствующих во всех реализациях языка. Библиотеки могут содержать в себе объявления констант и типов данных, различные процедуры, обладающие максимальной универсальностью, для работы на различных архитектурах и для использования в большом кругу решаемых задач, а также необходимое количество алгоритмов для обеспечения работы с различными объектами, с которыми может взаимодействовать программа. Во многих языках программирования библиотеки могут совпадать по своему назначению, как совпадает и класс решаемых задач. Примерный список решаемых функций, которые содержат стандартные библиотеки:

- операции ввода-вывода данных на терминал и файловыми операциями ввода-вывода;

- работа с динамической памятью;
- конвертация данных между типами;
- функции для работы со структурами данных: строками, массивами:
- математические операции;
- функции для обеспечения обработки исключений и ошибок в программе;
- функции для работы с сетью;
- функции для поддержки многопоточности.

1.2.1 Стандартные библиотеки в языке C++

Стандартные библиотеки языка C++ представлены в виде текстовых файлов заголовков (header-файлов), которые вставляются в текст программы с помощью директивы include препроцессора Си [5]. Список файлов, содержащих стандартные библиотеки представлен в таблице 1.

Таблица 1 - стандартные библиотеки языка C++

Реализуемые функции	**Список файлов**
Функции, для работы с массивами, списками и контейнерами	<bitset>, <algorithm> <deque>, <list>, <map>, <queue>, <set>, <stack>, <vector>, <iterator>
Функции, для работы со строками	<string>
Математические функции	<numeric>, <complex>, <valarray>, <limits>
Системные, сетевые и другие функции	<new> <typeinfo> <locale>, <memory>, <functional>

1.2.2 Библиотеки классов Java

В языке Java все классы происходят от класса Object, и, соответственно, наследуют методы этого класса. Некоторые библиотеки классов подключаются автоматически и называются встроенными. Другие библиотеки классов вы должны подключать в исходном тексте приложения Java явным образом при помощи оператора import [6]. Список подключаемых классов языка Java представлен в таблице 2.

Таблица 2 - библиотеки классов Java

Реализуемые функции	**Список библиотек**
Функции, для работы с массивами, списками и контейнерами	java.util.Map, java.util.List, java.util.Set, java.util.SortedSet, java.util.SortedMap, java.util.Collection
Функции, для работы со строками	java.text
Математические функции	java.math
Системные, сетевые и другие функции	java.io, java.net, java.awt, java.awt.image, java.awt.peer, java.applet

1.2.3 Библиотеки классов Python

Python - высокоуровневый язык программирования общего назначения, ориентированный на повышение производительности разработчика и читаемости кода [10]. В языке Python стандартные модули называются модулями. Каждая программа может импортировать модуль и получить доступ к его классам, функциям и объектам. Список подключаемых модулей языка Python представлен в таблице 3.

Таблица 3 - модули Python

Реализуемые функции	**Список модулей**
Функции, для работы с массивами, списками и контейнерами	collections, bisect, array
Функции, для работы со строками	string, re, struct
Математические функции	fractions, math
Системные, сетевые и другие функции	shutil, sys, smtplib, time

1.3 Анализ рассмотренных библиотек

Язык программирования помимо реализации собственно языка - компилятора, обычно содержит библиотеки стандартных функций. Изучив состав стандартных библиотек нескольких языков можно увидеть, что большая часть функций в этих языках совпадает.

Стандартные функции в языках программирования:

- операции с массивами, контейнерами, множествами - объединение, пересечение, вычитание, симметричная разность
- работа с строками - сравнение, поиск элемента \ подстроки, замена элемента, конкатенация;
- математические функции - тригонометрические, гиперболические, экспоненциальные, логарифмические и прочие;
- операции работы с сетью;
- функции ввода-вывода;

- функции при работе с графикой.
- системные функции.

Использование этих функций позволяет упростить и ускорить разработку собственных программ. В языке Пифагор же некоторые из этих функций отсутствуют, что замедляет и затрудняет разработку собственных более сложных алгоритмов и программ.

Анализируя категории функций применительно к реализации в языке Пифагор можно прийти к следующим выводам:

- операции с массивами, множествами, строками, математические операции - являются важными с точки зрения их наибольшей распространённости, являются базовыми для построения собственных алгоритмов, а также органично сочетаются с возможностями языка Пифагор;
- графические функции в данный момент не имеют возможности реализации из-за отсутствия манипулирования графическими объектами в языке Пифагор;
- системные и сетевые функции являются более архитектурно-зависимыми и не решают определенные практические задачи и лишь обеспечивают работу прикладных программ.

1.4 Вывод

Был произведен анализ библиотек стандартных функций языков C++, Java. В ходе анализа выделен ряд наиболее популярных категорий функций. Из этого ряда выделены категории, представляющих наибольший интерес с точки зрения развития языка Пифагор.

2 Стандартные библиотеки функций

В главе 1.3 мы выделили категории стандартных функций, теперь подробнее рассмотрим в соответствующие категории и объясним их необходимость, а также рассмотрим возможность их реализации.

2.1 Функции для работы со множествами

Множество - это составной тип данных для представления набора некоторых элементов как единого целого. Область значений множества - набор всевозможных подмножеств, составленных из его элементов.

Операции при работе со множествами:

- Пересечение;
- Объединение;
- Вычитание;
- Добавление элемента;
- Удаление элемента.

В языке Пифагор объектом данных является список, это позволит реализовать данные операции.

2.2 Функции для работы со строками

Строки в языках программирования обычно представлены массивом символов. Строка может содержать символы, цифры и специальные знаки, заключенные в кавычки. В языке Пифагор строка является массивом элементов, что позволяет применять к ней некоторые операции, применяемые к массивам.

Список операция при работе со строками:

- Конкатенация - сцепления нескольких строк в одну;
- Сравнение строк;
- Поиск элемента / подстроки в строке;

- Замена элемента / подстроки;
- Удаление элемента.

2.3 Математические функции

Математические функции это выполняющие некоторые часто используемые математические задачи.

Математические функции подразделяют на следующие категории:

- Тригонометрические функции;
- Гиперболические функции;
- Экспоненциальные и логарифмические функции.

Функциональный язык Пифагор вполне позволяет выполнять все эти операции с целыми и действительными числами.

2.5 Описание функций стандартной библиотеки

Для реализации некоторых функций из выделенных категорий в качестве источников были изучены известные алгоритмы решения данных задач в других языках программирования, например, в C++ [6], а также материалы по теории множеств и математическим функциям [8, 9], другие были разработаны самостоятельно основываясь на знании языка Пифагор и описании работы и результатов алгоритмов.

2.5.1 Поиск индекса элемента

Функция mult.search реализует поиск индекса элемента массива. Входные данные - необходимый элемент и массив для поиска. Выходные данные индексы искомого элемента в массиве. Алгоритм состоит в переборе массива и сравнение каждого элемента с искомым.

Функция реализована на основе стандартного алгоритма линейного поиска [XXX]. Последовательный обход элементов и запоминание индексов элементов, совпавших с искомым реализован рекурсивно; для экономии памяти было решено не создавать уменьшенных копий просматриваемого

массива, но хранить в памяти индекс, с которого начинается поиск на текущей итерации.

2.5.2 Замена элементов во множестве или строке

Функция mult.replace реализует замену элементов во множестве или строке. Входные данные - искомый элемент для замены, заменяющий элемент и массив для поиска. Выходные данные - измененный массив. Алгоритм состоит в переборе массива и замена искомого элемента с замещающим.

2.5.3 Пересечение двух множеств

Функция mult.cross реализует пересечение двух множеств. Пересечение - это все элементы, которые входят в оба множества. Входные данные - два массива. Выходные данные - массив пересечения. Алгоритм состоит в переборе первого множества с каждым элементом второго и вывод одинаковых.

2.5.4 Конкатенация строк и массивов

Функция mult.concat реализует конкатенацию, или по-другому слияние, неограниченного количества строк или массивов. Входные данные - несколько строк или массива. Выходные данные - новый массив со всеми элементами нескольких строк и массивов. Алгоритм состоит в последовательном выводе каждого элемента каждого из подаваемых множеств в новый массив.

2.5.5 Поиск максимального и минимального элемента

Функция mult.maxminsearch реализует поиск максимального и минимального элемента. Входные данные - массив. Выходные данные - максимальный элемент, минимальный. Суть алгоритма заключается в обходе всех элементов массива и сравнения каждого из них со значением, которое считается максимумом или минимум на данном шаге.

2.5.6 Поиск индекса элемента

Функция mult.rev реализует инвертирование массива. Входные данные - массив. Выходные данные - инвертированный массив. Алгоритм состоит выводе изначального массива путем его обхода с конечного до первого.

2.5.7 Вывод n символов слева

Функция mult.left реализует вывод n элементов с начала массива. Входные данные - длина и массив. Выходные данные - массив n-ой длины.

2.5.8 Вывод n символов справа

Функция mult.right реализует вывод n конечных элементов массива. Входные данные - длина и массив. Выходные данные - массив n-ой длины.

2.5.9 Декартово произведение двух множеств

Функция mult.dekart реализует декартово произведение двух множеств - множество, элементами которого являются все возможные упорядоченные пары элементов исходных множеств. Входные данные - два множества. Выходные данные - массив из упорядоченных пар двух исходных множеств.

2.5.10 Арифметическая прогрессия

Функция math.prog реализует вычисление арифметической прогрессии и ее суммы. Арифметическая прогрессия - последовательность вида:

a1, a1+d… a 1+d(n-1),

где a1 – первый член прогрессии, d - разность прогрессии, n -длина последовательности. Входные данные функции - a1, d и n. Выходные данные - последовательность и её сумма.

2.5.11 Геометрическая прогрессия

Функция math.geoprog реализует вычисление геометрической прогрессии. Геометрической прогрессии - последовательность вида:

$b_1=b_1, b_2=b_1*q… b_n=b_{n-1}*q$

где b_1 - первый член прогрессии, d - разность прогрессии, n -длина последовательности. Входные данные функции - b_1, d и n. Выходные данные - последовательность и её сумма.

2.5.12 Растояние между двумя точками

Функция math.segment вычисляет расстояния между двумя точками. Расстояние между двумя точками равно квадратному корню из суммы квадратов разностей координат по каждой оси. Входные данные – координаты двух точек. Выходные – расстояние между ними.

2.5.13 Деление отрезка в данном отношении

Функция math.divline вычисляет координаты (x, y) точки делящим отрезок в заданном отношении. Входные данные координаты двух точек – (x1, x2), (y1, y2) и отношение (m1, m2). Выходные - координаты точки. Вычисление происходит по формулам:

$x = (m2*x1 + m1*x2) / (m1+m2)$,

$y = (m2*y1 + m1*y2) / (m1+m2)$.

2.5.14 Последовательность Фибоначчи

Функция math.fib выводит последовательность Фибоначчи n-ой длины. Последовательность Фибоначчи - в которой первые два числа равны 1 и 1, а каждое последующее число равно сумме двух предыдущих чисел. Входные данные функции n. Выходные данные - последовательность n-ой длины.

2.5.15 Площадь поверхности сферы

Функция math.sorb вычисляет площадь поверхности сферы. Входные данные – радиус сферы r. Выходные данные – площадь поверхности сферы S. Вычисление происходит по формуле:

$S = 4*pi*r^2$.

2.5.16 Площадь цилинда

Функция math.sbarrel вычисляет площадь цилиндра. Входные данные – радиус цилиндра r и его высота h. Выходные данные – площадь цилиндра S. Вычисление происходит по формуле:

*S = 2*pi*r*h.*

2.5.17 Объем пирамиды

Функция math.vpyr вычисляет объем пирамиды. Входные данные – площадь основания S пирамиды и её высота h. Выходные данные – объем пирамиды V. Вычисление происходит по формуле:

*V = (S*h) / 3.*

2.5.18 Площадь круга

Функция math.circle вычисляет площадь круга. Входные данные – радиус круга r. Выходные данные – площадь цилиндра S. Вычисление происходит по формуле:

$S = 2*pi* r^2$.

2.5.19 Площадь треугольника

Функция math.triangle вычисляет площадь треугольника. Входные данные – длины трех сторон треугольника. Выходные данные – площадь треугольника S. Вычисление происходит по формулам:

P = (a+b +c) / 2,

S = sqrt(p(p-a)*(p-b)*(p-c)).*

2.5.20 Объем цилиндра

Функция math.vbarrel вычисляет объем цилиндра. Входные данные – радиус цилиндра r и его высота h. Выходные данные – объем цилиндра V. Вычисление происходит по формуле:

$V = h*pi*r^2$.

2.5.21 Площадь трапеции

Функция math.strap вычисляет площадь трапеции. Входные данные – две длины её оснований a, b и высота h. Выходные данные – площадь трапеции S. Вычисление происходит по формуле:

$S = ((a+b) / 2)* h.$

2.5.22 Объем сферы

Функция math.vorb вычисляет объем сферы. Входные данные – радиус сферы. Выходные данные – объем сферы V. Вычисление происходит по формуле:

$V = (4*pi*r^3)/3.$

2.6 Вывод

В результате анализа различных групп стандартных функций в разных языках программирования были выявлены основные операции, которые в дальнейшем будут реализованы на синтаксисе языке Пифагор и рассмотрена базовая алгоритмическая структура соответствующих функций.

3 Разработка библиотеки стандартных функций

В данном разделе будут подробно рассмотрены некоторые алгоритмы функций, реализованных для библиотеки стандартных функций языка Пифагор, и приведены результаты их исполнения в качестве примеров. В приложении А перечислен список функций для языка Пифагор с описанием их назначения. В приложении Б идет полный листинг функций языка Пифагор.

3.1 Функция поиска элемента массива

Функция mult.search реализует поиск элемента массива. Использует дополнительную функцию mult.isearch, для циклического прохода по массиву. Входными данные функции mult.search являются необходимый элемент и массив элементов, в котором происходит поиск, выходными данными являются индексы заданного элемента в массиве. Входными данные функции mult.isearch являются элемент, массив и счетчик цикла, выходными данными является индекс найденного элемента или возвращает «end» в конце выполнения либо если элемент не найдет в массиве. Так как в языке нет циклов, для их реализации используются рекурсии. Полный алгоритм функции search предоставлен на рисунке 2, функции mult.isearch на рисунке 3.

Функция поиска элемента search принимает составной параметр (x,A), где x – искомый элемент, A – массив, котором осуществляется поиск. Для работы search использует вспомогательную функцию isearch, принимающую тройку параметров (x,A,startIndex), где x – искомый элемент, A – массив, котором осуществляется поиск, startIndex – индекс, на котором осуществляется поиск в текущей итерации рекурсии. Если startIndex превышает длину массива A, поиск завершается. В обратном случае результатом данной итерации становится результат поиска x в массиве A с

позиции startIndex+1, к которому, в зависимости от равенства A[startIndex] и x, добавляется или не добавляется startIndex.

Код функции search:

```
mult.search << funcdef param  //определение функции search
{
e<<param:1; mass<<param:2; //определение входных параметров
return<<(mass,e,1): mult.isearch; //вызов функции isearch и завершение
}
```

Код функции mult.isearch:

```
mult.isearch << funcdef par //определение функции isearch
{
mass<<par:1; e<<par:2; i<<par:3; ; //определение входных параметров
.^[((i,mass:|):[>,=<]):?]^//; // условие цикла
(
"end", //завершение по окончании цикла
{
.^[((e,mass:i):[=,!=]):?]^//сравнение текущего элемента и искомого
(
{i //вывод индекса
{
block{
ii<<(i,1):+; // увеличение счетчика
(mass,e,ii): mult.isearch >>break;// переход к следующему элементу
}
},
{
block{
ii<<(i,1):+; // увеличение счетчика
```

(mass,e,ii): isearch >>break; ;// переход к следующему элементу

}

}

)

}

)>>return

}

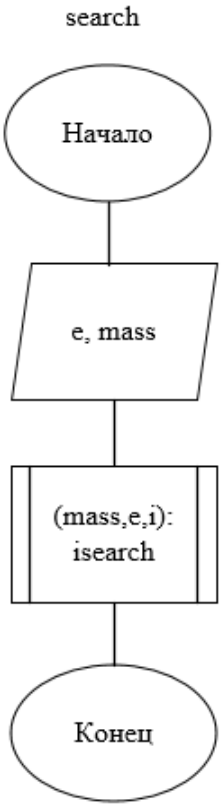

Рисунок 2 – алгоритм работы функции mult.search

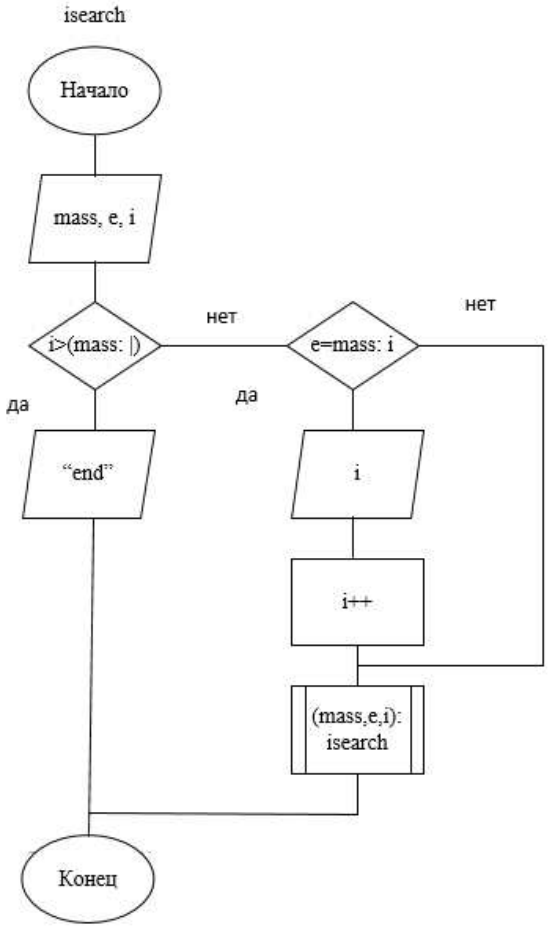

Рисунок 3 – алгоритм работы функции mult.isearch

Таблица 4 – пример использования программы

Входные данные	**Выходные данные**
(22,(1,2,3,22,4,5,22))	[4,7,("end")]
('z',('a','k','b','l','r','b'))	[("end")]
('b',(1,2,'b',22,4,'b',22))	[3,6,("end")]

3.2 Функция вычисления n чисел Фибоначчи

Функция math.fib используется для нахождения n чисел последовательности Фибоначчи – последовательность в которой каждое следующее число равно сумме предыдущих двух. Вызывает дополнительную функцию math.fib в качестве цикла, который проходит n шагов для вычисления n чисел Фи. Входные данные функции math.fib - количество искомых чисел, выходные данные - последовательность. Входные данные math.ifib - два предыдущие числа последовательности, счетчик шагов и

количество шагов. Алгоритм функции fib на рисунке 4, алгоритм math.ifib на рисунке 5.

Код функции fib:

```
math fib << funcdef x //определение функции fib
{
i<<1; n<<(x:1,1):-; //определение входных параметров
return<<(i,(0,i,i,n):ifib); //вызов функции ifib и завершение
}
```

Код функции math.ifib:

```
math ifib<< funcdef x //определение функции ifib
{
a<<x:1; b<<x:2; i<<x:3; n<<x:4; //определение входных параметров
c<<(a,b):+; //вычисление текущего элемента последовательности
return<< .^[((i,n):[<,=>]):?]^ // условие цикла
(
{c, //вывод элемента последовательности
block{
ii<<(i,1):+; // увеличение счетчика
(b,c,ii,n):math.ifib>>break; // переход к следующему элементу
}
},c//вывод элемента последовательности и завершение
))}
```

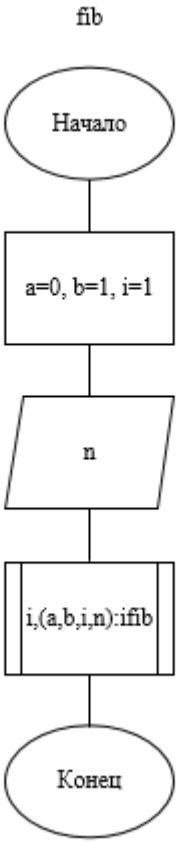

Рисунок 4 - алгоритм работы функции math.fib

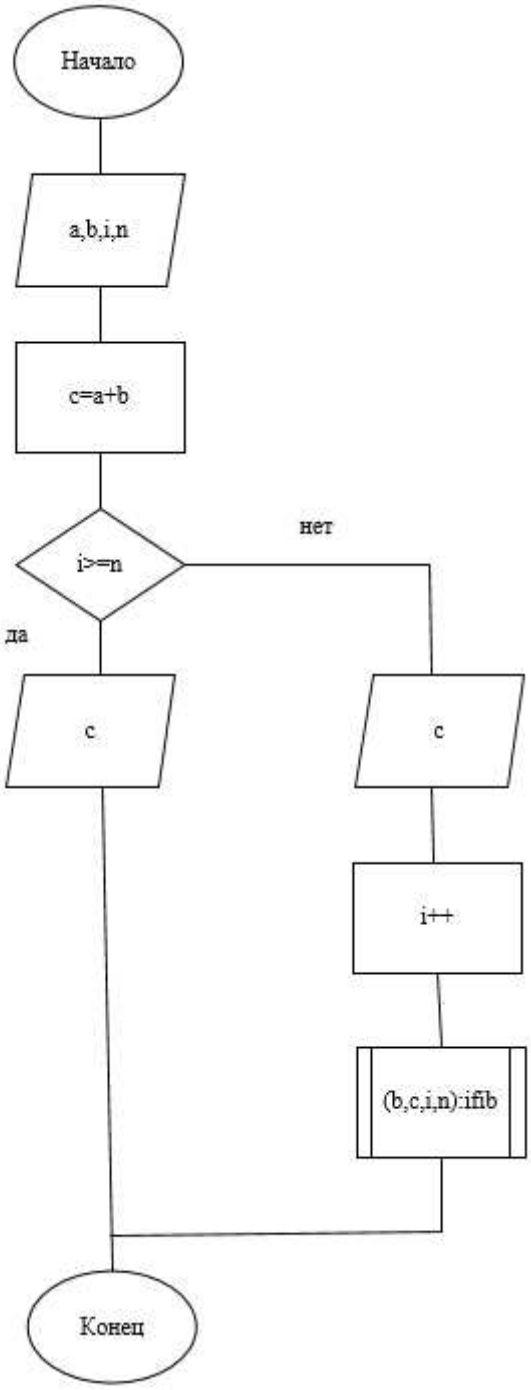

Рисунок 5 - алгоритм работы функции math.ifib

Таблица 5 - пример использования программы

Входные данные	**Выходные данные**
5	(1,1,2,3,5)
13	(1,1,2,3,5,8,13,21,34,55,89,144,233)
22	(1,1,2,3,5,8,13,21,34,55,89,144,233,377,610,987, 1597,2584,4181,6765,10946,17711)

3.3 Функция конкатенации строк или массивов

Функция mult.con выполняет конкатенацию - операция склеивания объектов линейной структуры. Входными данными con служат счетчик и строки, представленные массивом символьных элементов, или массивы с произвольными элементами, выходными данными является одна строка или массив. Вызывается рекурсивно в зависимости от количества дополнительную функцию mult.iconcat для склейки одной строки. Входные данные mult.iconcat счетчик шагов и строка. Полный алгоритм con предоставлен на рисунке 6, функции mult.iconcat на рисунке 7.

Код функции con:

mult.*con<< funcdef param* //определение функции *con*

{

i<<param:1; m<<param:2; //определение входных параметров

*m3<<.^[((i,m:|):[<,=>]):?]^* // условие цикла

(

{(1,m:i): mult.iconcat, // вызов функции iconcat

block{

ii<<(i,1):+; // увеличение счетчика

(ii,m): mult.con>>break; , // вызов функции con

}

},

{(1,m:i): mult.iconcat} //вывод последнего элемента

);return<<m3;} // завершение

Код функции mult.iconcat:

mult.iconcat<< funcdef par //определение функции iconcat

{

mass<<par:2; i<<par:1; //определение входных параметров

*return<<.^[((i,mass:|):[>=,<]):?]^* // условие цикла

(

mass:i, //вывод последнего элемента и завершение

{mass:i, , //вывод элемента

block{ii<<(i,1):+; // увеличение счетчика

(ii,mass): mult.*iconcat>>break;}})}* // переход к следующему элементу

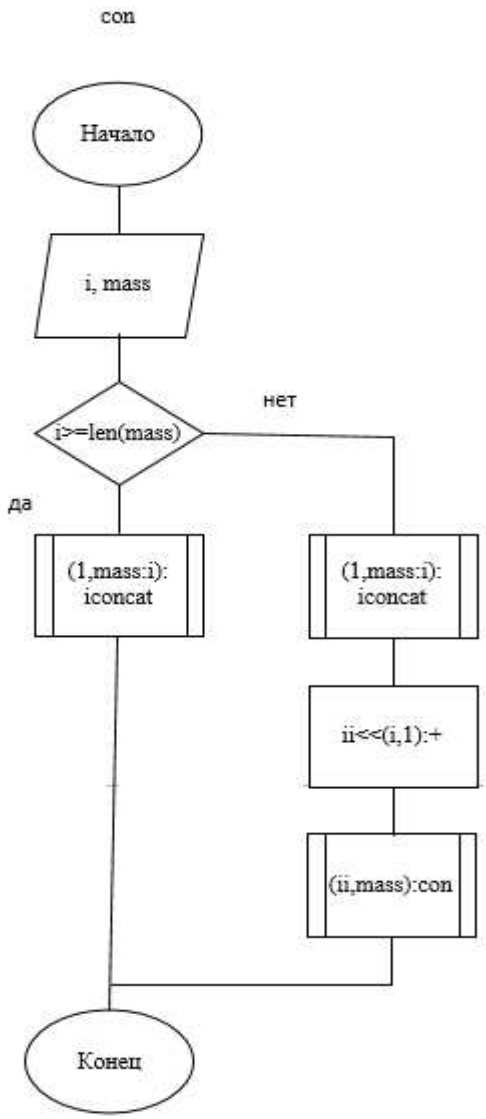

Рисунок 6 - алгоритм работы функции mult.con

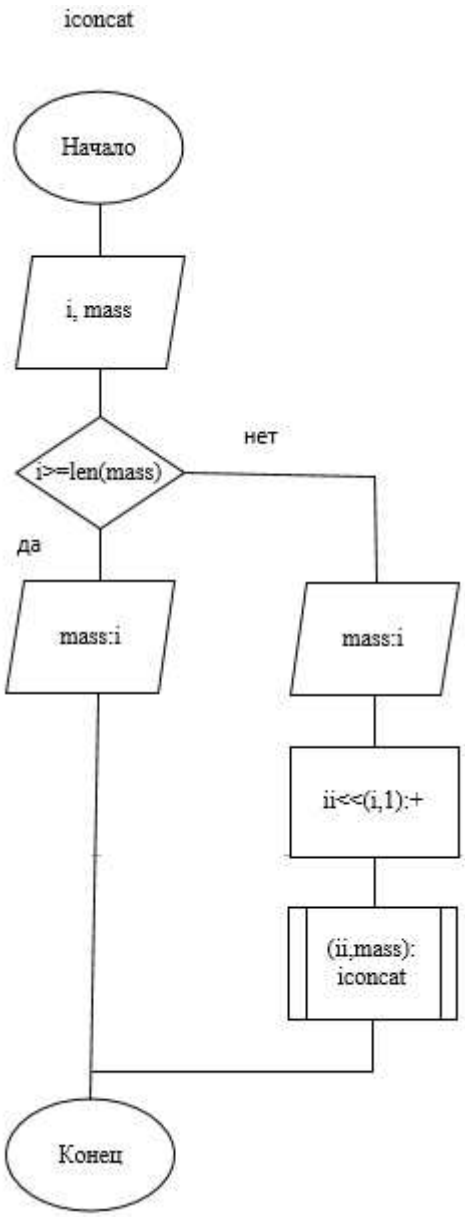

Рисунок 7 - алгоритм работы функции mult.iconcat

Таблица 6 - пример использования программы

Входные данные	**Выходные данные**
((1,2),(5,6))	[1,2,5,6]
(('a','b'),('c','d'),('f','e'))	["abcdfe"]
(('1','a'),('2','b'),('3','c'),('4','d'))	["1a2b3c4d"]

3.4 Вывод

В результате разработаны стандартные функции в различных категориях для языка программирования Пифагор.

ЗАКЛЮЧЕНИЕ

Был рассмотрен и изучен функционально-потоковый язык программирования Пифагор, цель которого разработка архитектурно независимых параллельных программ. Также при изучении библиотек стандартных функций в различных языках программирования были выявлены основные категории, которые необходимо реализовать на языке Пифагор. Разработан комплект функций на языке Пифагор для пользования ими в дальнейшем разработчиками.

СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1 Стандарт организации «Общие требования к построению, изложению и оформлению документов учебной деятельности» [Электронный ресурс] - Режим доступа: http://about.sfu-kras.ru/node/8127;

2 Функциональная модель параллельных вычислений и язык программирования "Пифагор" © 2002-2003 А.И. Легалов, Ф.А. Казаков, Д.А. Кузьмин, Д.В. Привалихин [Электронный ресурс] - Режим доступа: http://www.softcraft.ru/parallel/fpp/;

3 Функциональный язык для создания архитектурно-независимых параллельных программ, А. И. Легалов [Электронный ресурс] - Режим доступа: www.ict.nsc.ru/jct/getfile.php?id=670;

4 Материал из Википедии, свободной энциклопедии, про Пифагор (язык программирования) [Электронный ресурс] - Режим доступа: https://ru.wikipedia.org/wiki/Пифагор_(язык_программирования);

5 Материал из Википедии, свободной энциклопедии, стандартные библиотеки [Электронный ресурс] - Режим доступа: https://ru.wikipedia.org/wiki/Стандартная_библиотека;

6 Материал из Википедии, свободной энциклопедии, стандартные библиотеки языка C++ [Электронный ресурс] - Режим доступа: https://ru.wikipedia.org/wiki/Библиотека_языка_C%2B%2B;

7 Библиотеки классов Java [Электронный ресурс] - Режим доступа: www.helloworld.ru/texts/comp/lang/java/java5/vol1/ch5.html

8 Материал из Википедии, свободной энциклопедии, операции со множествами [Электронный ресурс] - Режим доступа: https://ru.wikipedia.org/wiki/Множество

9 Справочник по высшей математике / М. Я. Выгодский. - Москва: Астрель, 2006. - 991, [1] с.: ил. ISBN 5-17-012238-1, 5-271-03651-0;

10 Основные модули в Python 3. Описание модулей и перевод документации к ним. [Электронный ресурс] – Режим доступа: https://pythonworld.ru/moduli

ПРИЛОЖЕНИЕ А

Список функций и их описание

Таблица А.1 - список функций и их описание

Категория	Имя функции	Описание
Операции со множествами, строками, массивами - файл mult.pfg	search	Поиск индекса элемента списка
	mult.isearch	Доп. функция для search
	mult.replace	Замена элементов во множестве или строке
	mult.ireplace	Доп. функция для replace
	mult.cross	Пересечение множеств
	mult.scross	Доп. функция для cross
	mult.concat	Конкатенация неограниченного количества строк или множеств
	mult.con	Доп. функция для concat
	iconcat	Доп. функция для con
	mult.maxminsearch	Поиск максимального и минимального элемента
	mult.imaxsearch	Доп. функция для maxminsearch
	mult.iminsearch	Доп. функция для maxminsearch
	mult.rev	Инвертирование массива
	mult.irev	Доп. функция для irev

Продолжение Таблицы А.1

Категория	**Имя функции**	**Описание**
Операции со множествами - файл mult.pfg	mult.left	Вывод n элементов с начала
	mult.ileft	Доп. функция для left
	mult.right	Вывод n элементов с конца
	mult.iright	Доп. функция для right
	mult.dekart	Декартово произведение двух множеств
	mult.idekart	Доп. функция для dekart
Математические операции - файл math.pfg	math.prog	Вычисление арифметической прогрессии и ее суммы
	math.iprog	Доп. функция для prog
	math.geoprog	Вычисление геометрической прогрессии
	math.igeoprog	Доп. функция для geoprog
	math.segment	Вычисление расстояния между двумя точками
	math.divline	Вычисления точки делящим отрезок в заданном соотношении
	math.pi	Возвращает значение Pi до 6 знаков после запятой
	math.fib	Поиск N чисел Фиббоначи
	math.ifib	Доп. функция для fib

Окончание Таблицы А.1

Категория	**Имя функции**	**Описание**
Математические операции - файл math.pfg	math.sorb	Вычисление площади сферы
	math.sbarrel	Вычисление площади цилиндра
	math.vpyr	Вычисление объем пирамиды
	math.circle	Вычисления площади круга по радиусу
	math.triangle	Вычисление площади треугольника, по трем сторонам
	math.vbarrel	Вычисление объема цилиндра
	math.strap	Вычисление площадь трапеции
	math.vorb	Вычисление объема сферы

ПРИЛОЖЕНИЕ Б

Листинг программ на языке Пифагор

```
mult.concat<< funcdef param
{
	return<<(1,param):mult.con;
}
mult.con<< funcdef param
{
	i<<param:1;
	m<<param:2;
	m3<<.^[((i,m:|):[<,=>]):?]^
	(
		{
			(1,m:i):mult.iconcat,
			block{
				ii<<(i,1):+;
				(ii,m):mult.con>>break;
			}
		},
		{(1,m:i):mult.iconcat}
	);
	return<<m3;
}
mult.iconcat<< funcdef par
{
	mass<<par:2;  i<<par:1;
	return<<.^[((i,mass:|):[>=,<]):?]^
	(
	mass:i,
	{
```

```
            mass:i,
            block{
                ii<<(i,1):+;
                (ii,mass):mult.iconcat>>break;
            }
        }
        )
}
mult.cross<< funcdef param
{
        mass1<<param:2; mass2<<param:3; i<<param:1;
        return<<.^[((i,mass2:|):[<,=>]):?]^
        (
            {
            (mass1,mass2:i,1):mult.scross,
                block{
                    ii<<(i,1):+;
                    (ii,mass1,mass2):mult.cross>>break;
                }
            },
            (mass1,mass2:i,1):mult.scross
        )
}
mult.scross << funcdef par
{
        mass<<par:1; x<<par:2; i<<par:3;
        return<<.^[((i,mass:|):[>,=<]):?]^
        (
        "not found",
        {
```

```
        .^[((x,mass:i):[=,!=]):?]^
        (
                {mass:i},
                {
                        block{
                                ii<<(i,1):+;
                                (mass,x,ii):mult.scross>>break;
                        }
                }
        )
        }
        )
}
mult.dekart<< funcdef param
{
        i<<param:1;
        mass1<<param:2;
        mass2<<param:3;
        return<<.^[((i,(mass1:|,1):-):[=<,>]):?]^
        (
                {
                (mass1,mass2,i,1):mult.idekart,
                        block{
                                ii<<(i,1):+;
                                (ii,mass1,mass2):mult.dekart>>break;
                        }
                },
                {(mass1,mass2,i,1):mult.idekart}
        )
}
```

```
mult.idekart<< funcdef par
{
	mass1<<par:1;
	mass2<<par:2;
	l<<par:3;
	x<<mass1:l;
	i<<par:4;
	.^[((i,(mass2:|,1):-):[>,=<]):?]^
	(
	(x,mass2:i),
	{
		(x,mass2:i),
		block{
			ii<<(i,1):+;
			(mass1,mass2,par:3,ii):mult.idekart>>break;
		}
	}
	)>>return
}
mult.left<< funcdef param
{
	i<<param:1;
	m<<param:2;
	return<<.^[((i,m:|):[>=,<]):?]^
	(
		{(1,m:|,m):mult.ileft},
		{(1,i,m):mult.ileft}
	)
}
mult.ileft<< funcdef par
```

```
{
	mass<<par:3;  i<<par:1;
	return<<.^[((i,par:2):[>=,<]):?]^
	(
	mass:i,
	{
		mass:i,
		block{
			ii<<(i,1):+;
			(ii,par:2,mass):mult.ileft>>break;
		}
	}
	)
}
mult.right<< funcdef param
{
	i<<param:1;
	m<<param:2;
	z<<(m:|,i):-;
	n<<(z,1):+;
	return<<.^[((i,m:|):[>=,<]):?]^
	(
		{(1,m):mult.iright},
		{(n,m):mult.iright}
	)
}
mult.iright<< funcdef par
{
	mass<<par:2;  i<<par:1;
	return<<.^[((i,mass:|):[>=,<]):?]^
```

```
	(
	mass:i,
	{
		mass:i,
		block{
			ii<<(i,1):+;
			(ii,par:2,mass):mult.iright>>break;
		}
	}
	)
}
mult.maxminsearch << funcdef param
{
	mass<<param;
	return<<((mass,mass:1,1):mult.imaxsearch,(mass,mass:1,1):mult.iminsearc
h);
}
mult.imaxsearch << funcdef par
{
	mass<<par:1; max<<par:2; i<<par:3;
	.^[((i,mass:|):[>,=<]):?]^
	(
		max,
		{
			.^[((max,mass:i):[=>,<]):?]^
			(
			{
				block{
					ii<<(i,1):+;
					(mass,max,ii):mult.imaxsearch>>break;
```

```
                    }
                },
                {
                    block{
                        ii<<(i,1):+;
                        (mass,mass:i,ii):mult.imaxsearch>>break;
                    }
                }
                )
            }

        )>>return
}
mult.iminsearch << funcdef par
{
        mass<<par:1; min<<par:2; i<<par:3;
        .^[((i,mass:|):[>,=<]):?]^
        (
            min,
            {
                .^[((min,mass:i):[<,=>]):?]^
                (
                {
                    block{
                        ii<<(i,1):+;
                        (mass,min,ii):mult.iminsearch>>break;
                    }
                },
                {
                    block{
```

```
                    ii<<(i,1):+;
                    (mass,mass:i,ii):mult.iminsearch>>break;
                }
            }
            )
        }

    )>>return
}
mult.rev<< funcdef par
{
    return<<(par,par:|):mult.irev;
}
mult.irev<< funcdef par
{
    mass<<par:1;  n<<par:2;
    return<<.^[((n,1):[<=,>]):?]^
    (
    mass:n,
    {
        mass:n,
        block{
            ii<<(n,1):-;
            (mass,ii):mult.irev>>break;
        }
    }
    )
}
mult.search << funcdef param
{
```

```
        e<<param:1; mass<<param:2;
        return<<(mass,e,1):mult.isearch;
}
mult.isearch << funcdef par
{
        mass<<par:1; e<<par:2; i<<par:3;
        .^[((i,mass:|):[>,=<]):?]^
        (
        "end",
        {
        .^[((e,mass:i):[=,!=]):?]^
        (
                {
                        {i},
                        block{
                                ii<<(i,1):+;
                                (mass,e,ii): mult.isearch >>break;
                        }
                },
                {
                        block{
                                ii<<(i,1):+;
                                (mass,e,ii): mult.isearch >>break;
                        }
                }

        )
        })>>return
}
```

```
math.sqrt << funcdef Param
{
        ({(Param,Param):math.sqrt.sqrt_x:2},{("Param<0 in Sqrt",(2):+)}):
[(Param,0):(>=,<):?]:.>>return
}
math.sqrt.sqrt_x << funcdef Param
{
        A << Param:1;  X << Param:2;
         Xn << (X,(((X,X):*,A):-,(2,X):*):/):-;
         ({(A,Xn)}, {(A,Xn):sqrt.sqrt_x}):[((Xn,X):-
:abs,0.00001):(<,>=):?]:.>>return
}
math.abs << funcdef x
{
        return<<({x:-},{x}):[(x,0):(<,=>):?]:.
}
math.asmd<< funcdef Param
{
        a << Param : 1;
        b << Param : 2;
        f<<(a,b):[+,-,/,*];
        return<<f;
}
math.pi<< funcdef Param
{
        z<<3.141592;
        return<<z;
}
math.divline << funcdef Param
{
```

```
        x1<<Param:1:1;
        y1<<Param:1:2;
        x2<<Param:2:1;
        y2<<Param:2:2;
        m1<<Param:3:1;
        m2<<Param:3:2;
        x<<(((m2,x1):*,(m1,x2):*):+,(m1,m2):+):/;
        y<<(((m2,y1):*,(m1,y2):*):+,(m1,m2):+):/;
        return<<(x,y);
}
math.fib << funcdef x
{
        i<<1;
        n<<(x:1,1):-;
        return<<(1,(0,i,i,n):ifib);
}
math.ifib<< funcdef x
{
        a<<x:1; b<<x:2; i<<x:3; n<<x:4;
        c<<(a,b):+;
        return<< .^[((i,n):[<,=>]):?]^
        (
                {c,
                block{
                        ii<<(i,1):+;
                        (b,c,ii,n):math.ifib>>break;
                        }
                },c
        )
}
```

```
math.geoprog<< funcdef par
{
	return<<(par:1,(1,par:1,par:2,par:3):math.igeoprog);
}
math.igeoprog<< funcdef par
{
	i<<par:1; a<<par:2; q<<par:3; n<<par:4;
	c1<<(a,q):*;
	return<< .^[((i,(n,1):-):[<,=>]):?]^
	(
		{
			c1,
			block{
				ii<<(i,1):+;
				(ii,c1,q,n):math.igeoprog>>break;
			}
		},
		c1
	)
}
math.prog<< funcdef par
{
	return<<(par:1,(1,par:1,par:2,par:3,par:1):math.iprog);
}
math.iprog<< funcdef par
{
	i<<par:1; a<<par:2; d<<par:3; n<<par:4; s<<par:5;
	c<<(d,i):*;
	c1<<(c,a):+;
	ss<<(s,c1):+;
```

```
        return<< .^[((i,(n,1):-):[<,=>]):?]^
        (
                {
                        c1,
                        block{
                                ii<<(i,1):+;
                                (ii,a,d,n,ss):math.iprog>>break;
                        }
                },
                {c1,(ss)}
        )
}
math.sbarrel << funcdef Param
{
        r<<Param:1;
        h<<Param:2;
        r2<<(r,2):*;
        pi<<():math.pi;
        z<<(r2,pi):*;
        q<<(z,h):+;
        return<<(q,z):*;
}
math.segment << funcdef Param
{
        a<<(Param:2:1,Param:1:1):-;
        a2<<(a,a):*;
        b<<(Param:2:2,Param:1:2):-;
        b2<<(b,b):*;
        d<<(a2,b2):+;
        d:math.sqrt>>return;
```

```
}
math.sorb << funcdef Param
{
	r<<Param:1;
	r2<<(r,r):*;
	pi<<():math.pi;
	q<<(r2,pi):*;
	s<<(q,4):*;
	return<<s;
}
math.strap << funcdef Param
{
	a<<Param:1;
	b<<Param:2;
	h<<Param:3;
	z<<(a,b):+;
	q<<(2,h):*;
	return<<(z,q):/;
}
math.triangle << funcdef Param
{
	a<<Param:1;
	b<<Param:2;
	c<<Param:3;
	p<<(((a,b):+,c):+,2):/;
	p1<<(p,a):-;
	p2<<(p,b):-;
	p3<<(p,c):-;
	pz<<(((p1,p2):*,p3):*,p):*;
	s<<pz:math.sqrt;
```

```
        return<<s;
}

math.vbarrel << funcdef Param
{
        r<<Param:1;
        h<<Param:2;
        r2<<(r,r):*;
        pi<<():math.pi;
        z<<(r2,math.pi):*;
        q<<(z,h):*;
        return<<z;
}
math.vorb << funcdef Param
{
        r<<Param:1;
        r2<<(r,r):*;
        r3<<(r2,r):*;
        pi<<():math.pi;
        q<<(r3,pi):*;
        z<<(q,4):*;
        return<<(z,3):/;
}
math.vpyr << funcdef Param
{
        s<<Param:1;
        h<<Param:2;
        v<<(s,h):*;
        return<<(v,3):/;
}
```

ОТЗЫВ
руководителя о квалификационной работе бакалавра
Полтавец Кирилла Васильевича
на тему: «Библиотека стандартных функций для функционально-потокового языка параллельного программирования».

Выпускная квалификационная работа посвящена разработке библиотеки стандартных функций для функционально-потокового языка параллельного программирования Пифагор. Практическая польза данной темы основывается на предоставлении программистам уже готовых и проверенных процедур, которые сэкономят их время и убережет от технических ошибок в написании собственных алгоритмов.

В данной работе произведен анализ библиотек стандартных функций в нескольких языках программирования, произведена категоризация для библиотек стандартных функций, выбрано множество функций для реализации в библиотеке стандартных функций для языка Пифагор, разработана библиотека стандартных функций на языке Пифагор.

В результате разработана библиотека стандартных функций для языка Пифагор включающая в себя ряд функций, реализующих различные алгоритмы для работы с массивами, множествами, строками, математическими и геометрическими операциями.

Полтавец К.В. ориентируется в современных информационных технологиях, имеет навыки работы с программным обеспечением ЭВМ. При выполнении выпускной квалификационной работы бакалавра Полтавец К.В. самостоятельно изучил язык программирования Пифагор, а также стандартные библиотеки в языках C++, Java и Python.

Выполненная работа характеризует Полтавец Кирилла Васильевича как специалиста, способного самостоятельно изучать и осваивать современные информационные технологии и применять полученные знания на практике.

Пояснительная записка проанализирована системой “Антиплагиат”; По результатам проверки оригинальный текст составляет 89,67%. Отчет системы прилагается.

Недостатком работы можно назвать нехватку в разработанной библиотеке действительно сложных функций и недостаточно большое количество функций в целом.

Рекомендую продолжение научной деятельности в магистратуре.

Считаю, что работа заслуживает оценки «хорошо», а Полтавец К.В. присвоения степени бакалавра по направлению 09.03.01 «Информатика и вычислительная техника».

ст. преподаватель кафедры ВТ ИКИТ СФУ Матковский И.В.

Министерство образования и науки Российской Федерации
Федеральное государственное автономное образовательное
учреждение высшего образования
«Сибирский федеральный университет»

НАУЧНАЯ БИБЛИОТЕКА

660049, Красноярск, пр. Свободный ,79/10, тел.(3912) 2-912-820, факс (3912) 2-912-773
E-mail: bik@sfu-kras.ru

ОТЧЕТ
о результатах проверки в системе «АНТИПЛАГИАТ»

Автор: Полтавец К.В.
Заглавие: Библиотека стандартных функций для функционально-потокового языка параллельного программирования
Вид документа: Выпускная квалификационная работа бакалавра

По результатам проверки оригинальный текст составляет 89,67%

Источник	Коллекция / модуль поиска	Ссылка на источник	Доля в отчете	Доля в тексте
Турбо Паскаль 7.0	bibliorossica	http://www.bibliorossica.com/book.html?&currBookId=11011	0	0,28
	citations		0,31	0,31
229578	directmedia	http://biblioclub.ru/index.php?page=book_red&id=229578	0,4	0,72
240907	directmedia	http://biblioclub.ru/index.php?page=book_red&id=240907	0,23	0,23
Легалов, Александр Иванович диссертация ... доктора технических наук : 05.13.11 Красноярск 2005	disser.rsl	http://dlib.rsl.ru/rsl01002000000/rsl01002883000/rsl01002883426/rsl01002883426.pdf	0,02	0,8
Привалихин, Денис Викторович диссертация ... кандидата технических наук : 05.13.11 Красноярск 2004	disser.rsl	http://dlib.rsl.ru/rsl01002000000/rsl01002743000/rsl01002743978/rsl01002743978.pdf	0,27	0,58
Кузьмин, Дмитрий Александрович диссертация ... кандидата технических наук : 05.13.11 Красноярск 2004	disser.rsl	http://dlib.rsl.ru/rsl01002000000/rsl01002622000/rsl01002622876/rsl01002622876.pdf	0	0,57
Горский, Сергей Алексеевич диссертация ... кандидата технических наук : 05.13.11 Иркутск 2008	disser.rsl	http://dlib.rsl.ru/rsl01004000000/rsl01004177000/rsl01004177878/rsl01004177878.pdf	0	0,51
Капустин, Дмитрий Сергеевич диссертация ... кандидата технических наук : 05.13.11 Вологда 2013	disser.rsl	http://dlib.rsl.ru/rsl01006000000/rsl01006758000/rsl01006758463/rsl01006758463.pdf	0,06	0,5
Казаков, Фёдор Александрович диссертация ... кандидата технических наук : 05.13.11 Красноярск 2003	disser.rsl	http://dlib.rsl.ru/rsl01002000000/rsl01002616000/rsl01002616569/rsl01002616569.pdf	0	0,47
Дьяченко, Роман Александрович диссертация ... доктора технических наук : 05.13.01 Краснодар 2014	disser.rsl	http://dlib.rsl.ru/rsl01007000000/rsl01007916000/rsl01007916838/rsl01007916838.pdf	0,36	0,36

Источник	Коллекция / модуль поиска	Ссылка на источник	Доля в отчете	Доля в тексте
Швец, Дмитрий Александрович диссертация ... кандидата технических наук : 05.13.11 Красноярск 2004	disser.rsl	http://dlib.rsl.ru/rsl01002000000/rsl01002744000/rsl01002744155/rsl01002744155.pdf	0	0,35
Тан Шейн диссертация ... кандидата технических наук : 05.13.01 Москва 2014	disser.rsl	http://dlib.rsl.ru/rsl01007000000/rsl01007577000/rsl01007577178/rsl01007577178.pdf	0	0,25
Редькин, Андрей Владимирович диссертация ... кандидата технических наук : 05.13.11 Красноярск 2009	disser.rsl	http://dlib.rsl.ru/rsl01004000000/rsl01004361000/rsl01004361997/rsl01004361997.pdf	0	0,19
Фадеев, Роман Викторович диссертация ... кандидата технических наук : 05.13.17, 05.13.11 Таганрог 2005	disser.rsl	http://dlib.rsl.ru/rsl01002000000/rsl01002748000/rsl01002748876/rsl01002748876.pdf	0,06	0,16
Лежебоков, Валерий Валерьевич диссертация ... кандидата технических наук : 05.13.01 Волгоград 2009	disser.rsl	http://dlib.rsl.ru/rsl01004000000/rsl01004408000/rsl01004408889/rsl01004408889.pdf	0	0,15
Манахов, Павел Алексеевич диссертация ... кандидата технических наук : 05.13.17 Москва 2012	disser.rsl	http://dlib.rsl.ru/rsl01006000000/rsl01006504000/rsl01006504094/rsl01006504094.pdf	0	0,13
Сухорослов, Олег Викторович диссертация ... кандидата технических наук : 05.13.18 Москва 2005	disser.rsl	http://dlib.rsl.ru/rsl01002000000/rsl01002800000/rsl01002800700/rsl01002800700.pdf	0	0,11
Стандартная библиотека	internet	http://ru.wikipedia.org/wiki/Стандартная библиотека	1,62	1,62
1. Алгоритм-организованная последовательность действий приводящая к решению поставленных задач. Язык записи алгоритма,близкий к естественному- это псевдокод	internet	http://userdocs.ru/informatika/1897/index.html	0	1,62
	internet	http://www.softcraft.ru/parallel/fpp/intro.shtml	1,49	1,49
Пифагор (язык программирования)	internet	http://ru.wikipedia.org/wiki/Пифагор (язык программирования)	1,23	1,23
Download	internet	http://mais.uniyar.ac.ru/sites/default/files/journal/private/19_5__81-99.pdf	0,8	1,03
полный текст	internet	http://ict.nsc.ru/jct/getfile.php?id=670	0	0,77
	internet	http://p6.ru/referats/files41446432/14/pda-0522.zip	0,68	0,68
Введение в технологии параллельного программирования	internet	http://www.masters.donntu.edu.ua/2011/fknt/lyamina/library/parallel_program.htm	0,57	0,57
Объектно-ориентированное программирование (5/18)	internet	http://www.bsuir.by/m/12_100229_1_90135.pdf#5	0,52	0,52
Общая схема структуры программы в ЯП. (1/2)	internet	http://studopedia.org/14-92640.html#1	0,42	0,42

Источник	Коллекция / модуль поиска	Ссылка на источник	Доля в отчете	Доля в тексте
Языковая и инструментальная поддержка функционально-потокового параллельного программирования диссертация по информатике, вычислительной технике и управлению, скачайте бесплатно автореферат диссертации на тему 'Математическое и программное обеспечение в...	internet	http://tekhnosfera.com/yazykovaya-i-instrumentalnaya-podderzhka-funktsionalno-potokovogo-parallelnogo-programmirovaniya	0	0,35
Методическая система обучения основам параллельного программирования будущих учителей информатики (1/2)	internet	http://netess.ru/3pedagogika/661279-1-metodicheskaya-sistema-obucheniya-osnovam-parallelnogo-programmirovaniya-buduschih-uchiteley-informatiki.php#1	0,19	0,31
Программирование на Java	internet	http://lib.rus.ec/b/364392	0,29	0,29
	internet	http://www.citforum.ru/programming/theory/parall_prog/	0	0,28
	internet	http://sysbin.com/files/os/unix_os.zip	0,23	0,23
Диссертация_Лылов_Е.В..pdf	internet	http://www.science.vsu.ru/dissertations/1270/%D0%94%D0%B8%D1%81%D1%81%D0%B5%D1%80%D1%82%D0%B0%D1%86%D0%B8%D1%8F_%D0%9B%D1%8B%D0%BB%D0%BE%D0%B2_%D0%95.%D0%92..pdf	0	0,18
C++ Павловская C++ Программирование на языке высокого уровня	internet	http://www.docme.ru/doc/118856/c---pavlovskaya-c---programmirovanie-na-yazyke-vysokogo-urovnya	0,16	0,16
224625	lan	http://e.lanbook.com/journal/issue.php?p_f_journal=2544&p_f_year=2015&p_f_issue=2	0	0,26
9350	lan	http://e.lanbook.com/books/element.php?pl1_id=9350	0,23	0,23
62752	lan	http://e.lanbook.com/books/element.php?pl1_id=62752	0,21	0,21
Бурба_2 глава	sfukras		0	0,35
Федеральное государственное автономное образовател.txt	sfukras		0	0,33

Частично оригинальные блоки: 10,33%
Оригинальные блоки: 89,67%
Заимствование из белых источников: 0,31%
Итоговая оценка оригинальности: 89,98%

Подготовлено автоматически с помощью системы «Антиплагиат»
дата: 16.06.2017

Федеральное государственное автономное
образовательное учреждение
высшего образования
«СИБИРСКИЙ ФЕДЕРАЛЬНЫЙ УНИВЕРСИТЕТ»

Институт космических и информационных технологий

Кафедра вычислительной техники

УТВЕРЖДАЮ
Заведующий кафедрой
_____ А.И.Легалов
подпись инициалы, фамилия
«19» 06 2017 г.

БАКАЛАВРСКАЯ РАБОТА

09.03.01 Информатика и вычислительная техника

Библиотека стандартных функций для функционально-потокового языка параллельного программирования

Руководитель	19.06.17		Матковский И.В
	подпись, дата	должность, ученая степень	инициалы, фамилия
Консультант			Доррер Г.А.
	подпись, дата	должность, ученая степень	инициалы, фамилия
Выпускник	19.06.2017		Полтавец К.В.
	подпись, дата		инициалы, фамилия
Нормоконтроль	19.06.17		Иванов В.И.
	подпись, дата		инициалы, фамилия

Красноярск 2017