Министерство Образования и Науки Республики Казахстан

Алматинский Колледж Экономики и Права

ОТЧЕТ

по учебной практике по

«Алгоритмизации и программированию»

Выполнил обучающийся

Группы 207–П Громыко Н.

Проверил преподаватель

специальных дисциплин

Боржим С.Л.

Защищен с оценкой \_\_\_\_\_\_

«31» Мая 2019г.

Алматы 2019г.

СОДЕРЖАНИЕ

№ документа

Подпись

Изм.

Лист

Утв.

Н. Контр.

Рецензент

Руковод.

Студент

Боржим С.Л.

Громыко Н.

*У*

Литера

2

Лист

29

Листов

АКЭП 207-П

Практика по алгоритмизации и программированию

Стр.

**ВВЕДЕНИЕ4**

1. Общая часть5
   1. Описание программного обеспечения5
   2. Описание языка программирования Delphi5
      1. Типы данных6
      2. Агрегаты данных 7
      3. Операторы8
         1. Оператор присваивания8
         2. Составной оператор8
         3. Условный оператор8
         4. Оператор выбора8
         5. Операторы цикла8
         6. Операторы ввода-вывода9
2. Специальная часть11
   1. Задача 111
      1. Постановка задачи11
      2. Исходные данные11
      3. Результат11
      4. Математический метод11
      5. Описание подпрограмм11
      6. Макро блок-схема12
      7. Блок-схемы подпрограмм13
      8. Таблица идентификаторов14
      9. Контрольные примеры15
      10. Организация ввода-вывода данных15
      11. Листинг проблемной части16

2.2. Задача 217

2.2.1. Постановка задачи17

* + 1. Исходные данные17
    2. Результат17
    3. Математический метод17
    4. Описание подпрограмм17
    5. Макро блок-схема18
    6. Блок-схемы подпрограмм19
    7. Таблица идентификаторов21
    8. Контрольные примеры21
    9. Организация ввода-вывода данных21
    10. Листинг проблемной части22

2.3. Задача 322

2.3.1. Постановка задачи22

* + 1. Исходные данные23
    2. Результат23
    3. Математический метод23
    4. Описание подпрограмм23
    5. Макро блок-схема24
    6. Блок-схема подпрограммы25

3

Лист

## № Документа

## Подпись

Изм.

## Лист

* + 1. Таблица идентификаторов25
    2. Контрольные примеры26
    3. Организация ввода-вывода данных26
    4. Листинг проблемной части27

1. Заключение28
2. Список литературы29

**ВВЕДЕНИЕ**

4

Лист

## № Документа

## Подпись

Изм.

## Лист

Программное обеспечение (далее ПО) - это программа или комплекс программ, для взаимодействия пользователя с функциями и возможностями компьютера. ПО различают по трем классификациям

1. Системное;
2. Прикладное;
3. Инструментальное.

Системное ПО это совокупность программ, которые входят в ОС, они служат для управления и взаимодействия с компонентами ПО. Например, ОЗУ, процессор устройствами ввода-вывода. Так же к системному ПО относятся драйверы.

Прикладное ПО – это те программы, которые предназначены для взаимодействия с пользователем, это все программы пользователя. Прикладное ПО взаимодействует с компьютером посредством обращения к системному ПО. К прикладному ПО относятся видео редакторы, браузеры, графические редакторы и т.д.

К инструментальному ПО относятся программы для разработки, проектирования, поддержки программ.

1. Общая часть

5

Лист

## № Документа

## Подпись

Изм.

## Лист

* 1. Описание программного обеспечения

Программное обеспечение, как было сказано выше, это комплекс программ, используемых для управления компьютером. Разница между программой и программным обеспечением: Программа — это ограниченный одиночный продукт, работающий в информационно-вычислительной сфере, а программное обеспечение — это совокупность отдельных программ, связанных в одну группу для достижения единого результата. Как и было сказано выше, программное обеспечение делится на системное, прикладное и инструментальное.

Системное программное обеспечение в отличие от прикладного программного обеспечения, системное не решает конкретные практические задачи, а лишь обеспечивает работу других программ. Пример системного программного обеспечения – операционная система Windows 7, которая использовалась на практике.

Прикладное программное обеспечение общего назначения используется для решения наиболее общих задач информационного характера в любой сфере человеческой деятельности. Оно объединяет в себе широко используемые программы большинством пользователей персональных компьютеров, например, текстовые редакторы, электронные таблицы, графические системы, игры, развлечения. Такие, как блокнот (текстовый редактор), калькулятор и прочее. На практике были пользованы NotPad++ (расширенный блокнот), MS Word 2010, калькулятор, Google Chrome.

Программное обеспечение, предназначенное для использования в ходе проектирования, разработки и сопровождения программ, в отличие от прикладного и системного программного обеспечения. Строго говоря, определение последнего включает в себя определение инструментального, поэтому инструментальное можно считать обособленным подклассом прикладного ПО. На практике задачи были выполнены в IDE от Embarcadero – Rad Studio 10.2 и Rad Studio 10.3.

* 1. Описание языка программирования Delphi

Среда разработки Delphi. Использовались две версии IDE, это Embarcadero Dephi 10.2 на домашнем компьютере и Embarcadero Dephi 10.3 на ноутбуке. Среда Delphi используется для разработки и поддержки программ для Windows(в основном), MacOS, iOS, Android, так же в версию 10.2 добавлен компилятор под Linux. Delphi – высокоуровневый объектно-ориентированный язык программирования, являющийся диалектом языка Object Pascal. Основная область его применения – написание прикладных программ.

Ниже перечислены основные составные части Delphi:

6

Лист

## № Документа

## Подпись

Изм.

## Лист

1. Дизайнер Форм
2. Редактор кода
3. Палитра Компонентов
4. Инспектор Объектов
5. Справочник

С помощью этого на форме можно создать удобный пользовательский интерфейс для взаимодействия с пользователем. Например, добавить кнопку, по нажатию которой будет происходить сложения чисел и Edit’ов (компонент) и выводить значение в Label (компонент).

Список компонентов, которые использовались на практике:

* Button – кнопки по которым происходят некоторые события;
* Edit – для ввода пользователем информации (иногда вывода с флагом ReadOnly) ;
* Label – вывод информации пользователю;
* Image – на данной практике для рисования на Canvas’е;
* ComboBox – выбор пользователем месяцев\функций из списка ;
* ListBox – вывод сообщений, последовательностей и т.д;
* OpenDialog – загрузка файлов с помощью диалога Windows;
* MainMenu – навигация с пунктами о том кто сделал, и что делает это программное средство;
* GroupBox – панель, объединяющая компоненты в группы;
* Memo – вывод информации, например, условие задачи;
  + 1. Типы данных

Integer — целые числа.

Например: n: integer;

Этот тип является самым распространённым, который использовался для решения задач, т.к. большинство задач требовали целые числа.

Boolean — логический тип.

Например: res: boolean;

Второй по распространённости тип. На основе того, что возвращает переменная этого типа, зависит, выполнится ли функция, станет ли доступна кнопка и т.д.

Real — действительные числа.

7

Лист

## № Документа

## Подпись

Изм.

## Лист

Используется во всех задачах, где необходимы действительные числа. При выводе использовалась функция Format.

Char — символьный тип данных.

Этот тип использовался в 14.35 по учебнику Пильщикова. Брался один символ и сверялся на вхождение\не вхождение к множествам по условиям 14.35.

String — строковой тип данных.

Использовался во всех задачах, т.к. требуется вывести текстуальное меню и вывода сообщений пользователю.

Word — целочисленный тип.

Этот тип был использован мной в задачах на дату, потому что его диапазон от 0 до 9999, что удобно для описания года.

* + 1. Агрегаты данных

Текстовые файлы. Текстовые файлы использовались практически в каждой задаче, кроме тех, что требовали сгенерировать данные или ввести данные с клавиатуры. Чтобы взять данные из файла, необходима файловая переменная (f: TextFile;).

Массив — это структура данных, представляющая собой набор переменных одинакового типа, имеющих общее имя. Массивы используются для хранения упорядоченной однородной информации в них, к элементу массива следует обращаться по его индексу.

Объявление массива в секции var:

<имя\_массива>: array[<начальный индекс>..<конечный индекс>] of <тип>

Так же массивы могут быть одномерными, двухмерными, n-мерными.

На практике использовались одномерные и двумерные массивы для ввода, преобразования и вывода последовательностей и матриц.

Множество - это набор элементов одного и того же типа. Множество может включать в себя до 256 элементов, а так же может быть пустым. Все элементы множества должны быть различными, т.е. добавить такой же элемент не получится.

Объявление множества:

<имя\_множества>: set of <тип\_элементов>;

На практике множества использовались в задачах Абрамяна Matrix 37-40, Пильщиков 14.35 и других.

* + 1. Операторы

8

Лист

## № Документа

## Подпись

Изм.

## Лист

* + - 1. Оператор присваивания

Оператор присваивания предназначен для присваивания переменной значения.

Объявление:

<имя\_переменной> := <значение>;

Например: res := true, т.е. переменной логического типа присваивается значение «истина».

* + - 1. Составной оператор

В Delphi условный оператор if, циклы while и for требуют в качестве ветвей и тела один оператор, поэтому при необходимости разместить в ветвях условного оператора или теле цикла несколько команд используются составные операторы:

If <условие> then

Begin {начало составного оператора}

… {несколько операторов}

End {конец составного оператора}

* + - 1. Условный оператор

Условный оператор используется в том случае, когда необходимо выполнять те или иные действия, в зависимости от истинности или ложности некоторого логического выражения.

* + - 1. Оператор выбора

Оператор выбора (варианта) используется в тех случаях, когда в зависимости от значения какого-либо выражения необходимо выполнить один из нескольких операторов.

Оператор case состоит из селектора, списка констант, операторов.

При выполнении значение селектора сравнивается со спиcком констант, выполняется та ветка, где значение селектора и списка совпали, если таковых нет, то выполняется ветка else.

* + - 1. Операторы цикла

В Delphi есть три оператора цикла: оператор с постусловием, с предусловием и оператор цикла с параметром. В решении задач я использовал все три.

Оператор цикла с параметром повторяется, пока управляющая переменная не достигла конечного значения. Шаг цикла равен начальному значению и изменятся в теле самого цикла не может. Если необходимо в цикле использовать более одной команды, то необходимо использовать составной оператор.

Пример цикла for:

For i := 1 to 20 do

Begin

R := R+i\*i;

End;

После выполнения цикла переменная R получит сумму квадратов чисел от 1 до 20.

Оператор цикла с предусловием выполняется, пока логическое выражение возвращает значение истина. Он может не выполниться ни разу. Управляющая переменная может изменяться внутри цикла. Если необходимо в цикле использовать более одной команды, то необходимо использовать составной оператор.

9

Лист

## № Документа

## Подпись

Изм.

## Лист

Пример цикла while:

while res and i<Length(arr) do

begin

res := res and arr[i]>=0;

inc(i);

end;

Цикл продолжается, пока не встретится отрицательный элемент или не закончится массив.

Оператор цикла с постусловием выполняется, пока логическое выражение возвращает false. Выполняется минимум один раз. Управляющая переменная может изменяться внутри цикла.

Пример цикла repeat-until:

repeat

res := res and arr[i]>=0;

inc(i);

until not res or not(i<Length(arr));

* + - 1. Операторы ввода-вывода

Оператор ввода – read и readln. Он используется в Delphi чтения данных из файла. При readln осуществляется переход на следующую строку.

Оператор вывода – write и writeln. Он используется в Delphi для ввода данных в файл. При writeln осуществляется перенос строки на следующую строку.

Пример:

If OpenDialog1.Execute then

10

Лист

## № Документа

## Подпись

Изм.

## Лист

Begin

AssignFile(f, OpenDialog1.FileName);

Reset(f);

I := 1;

While not eof(f) do

Begin

Read(f, arr[i]);

Inc(i);

End;

CloseFile(f);

End;

Данная конструкция (и ее вариации) использовалась практически во всех выполненных мной задачах. Она записывает данные из файла f в массив arr.

1. Специальная часть
   1. Задача 1
      1. Постановка задачи

Даны пять различных целых чисел. Найти среди них два числа, модуль разности которых имеет:

а) наибольшее значение;

б) наименьшее значение.

* + 1. Исходные данные

Исходными данными этой программы является файл с последовательностью из пяти различных чисел. За связь с файлом отвечает переменная f (TextFile). Данные вносятся в массив arr[1..5] of integer.

* + 1. Результат

min – разность некоторых двух чисел по модулю, образующих наименьшее значение;

max – разность некоторых двух чисел по модулю, образующих наибольшее значение;

id1, id2 – номера элементов в массиве arr, разность которых по модулю, образует наименьшее значение

id3, id4 – номера элементов в массиве arr, разность которых по модулю, образует наибольшее значение.

* + 1. Математический метод

Пусть arr = <-4, 0, 12, 2, -6>, необходимо проверять каждый i-й с каждым кроме i-го. =>

11

Лист

## № Документа

## Подпись

Изм.

## Лист

id1 = 1; id2 = 5;

id3 = 4; id4 = 5;

max = |(12) – (-6)| = 18;

min = |(-4) – (-6)| = 2;

* + 1. Описание подпрограмм

Button1.Click. В этой процедуре работает OpenDialog и пользователю предлагается выбрать файл для загрузки. После выбора последовательность записывается в массив arr.

Button2.Click. В данной процедуре происходят все сравнения и вычисления. А так же ответ выводится в Label1 и Label2.

* + 1. Макро блок-схема

Конец

Действие

Выполнить

Открыть файл

Об авторе

Решение

Условие

Пока программа не закрыта

Начало

12

Лист

## № Документа

## Подпись

Изм.

## Лист

* + 1. Блок-схемы подпрограмм

13

Лист

## № Документа

## Подпись

Изм.

## Лист

Button2.Click

Button1.Click

OD1.Execute

-

max := -1; min := 1000;

+

AssignFile(f,OpenDialog1.FileName); reset(f);

i := 1, 5

J := 1, 5

i := 1, 5

Конец

|(arr[i]-arr[j])|>max

AssignFile(f,OpenDialog1.FileName); reset(f);

|(arr[i]-arr[j])|<min

id3, id4, max, id1, id2, min

I<>j

min := abs(arr[i]-arr[j]);

id1 := i; id2 := j;

Конец

CloseFile(f);

Button2.Enabled := true;

read(f, arr[i]); CloseFile(f);

Button2.Enabled := true;

14

Лист

## № Документа

## Подпись

Изм.

## Лист

* + 1. Таблица идентификаторов

Таблица 2.1.8.1. Таблица идентификаторов задачи 1

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Имя переменной | Семантика | Тип | Разрядность |
| 1 | f | Файловая переменная | TextFile |  |
| 2 | i | Индекс сверяемого числа | Integer | -32 768.. 32 767 |
| 3 | j | Индекс сверяемого числа | Integer | -32 768.. 32 767 |
| 4 | max | Модуль разности некоторых двух чисел (наибольшее число) | Integer | -32 768.. 32 767 |
| 5 | min | Модуль разности некоторых двух чисел (наименьшее число) | Integer | -32 768.. 32 767 |
| 6 | Id1 | Индекс одного из чисел разности | Integer | -32 768.. 32 767 |
| 7 | Id2 | Индекс одного из чисел разности | Integer | -32 768.. 32 767 |
| 8 | Id3 | Индекс одного из чисел разности | Integer | -32 768.. 32 767 |
| 9 | Id4 | Индекс одного из чисел разности | Integer | -32 768.. 32 767 |

13

Лист

## № Документа

## Подпись

Изм.

## Лист

2.1.9. Контрольные примеры

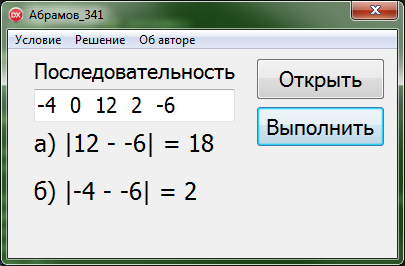
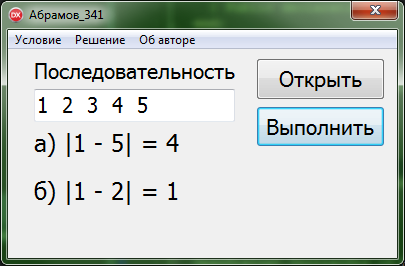


Рисунок 2.1.9.1. Контрольный пример работы программы №1



15

Лист

## № Документа

## Подпись

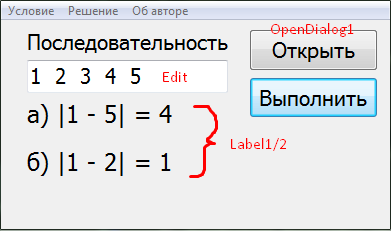
Изм.

## Лист

Рисунок 2.1.9.2. Контрольный пример работы программы №2

* + 1. Организация ввода-вывода данных

Ввод-вывод данных в этой программе производится с помощью OpenDialog1 и файла, а также Edit1, Label1 и Label2. Все это обозначено на рисунке 2.1.10.1.



16

Лист

## № Документа

## Подпись

Изм.

## Лист

1

Рисунок 2.1.10.1. Компоненты ввода-вывода в программе

* + 1. Листинг проблемной части

procedure TForm1.Button1Click(Sender: TObject);

var

f: textFile;

i: Integer;

begin

OpenDialog1.InitialDir := ExtractFilePath(Application.ExeName);

if OpenDialog1.Execute then

begin

AssignFile(f, OpenDialog1.FileName);

reset(f);

for i := 1 to 5 do

begin

read(f, arr[i]);

Edit1.Text := Edit1.Text + IntToStr(arr[i]) + ' ';

end;

CloseFile(f);

Button2.Enabled := true;

end;

end;

procedure TForm1.Button2Click(Sender: TObject);

var

i, j, min, max, id1, id2, id3, id4: Integer;

begin

max := -1;

min := 1000;

for i := 1 to 5 do

for j := 1 to 5 do

if i<>j then

begin

if abs(arr[i]-arr[j])<min then

begin

min := abs(arr[i]-arr[j]);

id1 := i;

id2 := j;

end;

if abs(arr[i]-arr[j])>max then

begin

max := abs(arr[i]-arr[j]);

id3 := i;

id4 := j;

end;

end;

Label2.Caption := 'а) |' + IntToStr(arr[id3]) + ' - ' + IntToStr(arr[id4]) + '| = ' + IntToStr(max);

Label1.Caption := 'б) |' + IntToStr(arr[id1]) + ' - ' + IntToStr(arr[id2]) + '| = ' + IntToStr(min);

end;

* 1. Задача 2

2.2.1. Постановка задачи

Дана непустая последовательность слов из строчных букв; между соседними словами - запятая, за последним словом - точка. Напечатать в алфавитном порядке все согласные буквы, которые входят только в одно слово.

2.2.2. Исходные данные

17

Лист

## № Документа

## Подпись

Изм.

## Лист

Исходными данными этой программы является файл с последовательностью слов. За связь с файлом отвечает переменная f (TextFile). Данные вносятся в строку s: string.

2.2.3. Результат

m – множество содержащее согласные буквы, которые входят только в одно слово.

2.2.4. Математический метод

Пусть s = ‘specialniy,text,dlya,otcheta.’, Необходимо занести все символы, которые входят в последовательность (m), а затем внести повторяющиеся в другое множество (m1). В конце остается лишь отнять от первого множества второе (m := m-m1).

Из этого следует, что m = [d, h, n, p, s, x].

2.2.5. Описание подпрограмм

Button1.Click. В этой процедуре работает OpenDialog и пользователю предлагается выбрать файл для загрузки. После выбора последовательность слов записывается в строку s.

Button2.Click. В данной процедуре происходят преобразования, а результат выводится в ListBox1.

2.2.6.Макро блок-схема

Открыть файл

Выполнить

Решение

Условие

Об авторе

Действие

Пока программа не закрыта

Начало

18

Лист

## № Документа

## Подпись

Изм.

## Лист

Конец

2.2.7. Блок-схемы подпрограмм

19

Лист

## № Документа

## Подпись

Изм.

## Лист

Button1.Click

OD1.Execute

AssignFile(f,OpenDialog1.FileName); reset(f);

read(f, s);CloseFile(f);

Button2.Enabled := true;

Memo2.Text := s

Конец

sm := ''; sg := ['a'..'z'] - ['a', 'e', 'i', 'o', 'u'];

Button2.Click

J := 1, length(s)

ListBox1.Clear;s[length(s)] := ','; m := [];

sm := sm+s[i]

s[i]<>','

20

Лист

## № Документа

## Подпись

Изм.

## Лист

m := m+[sm[i]

not(sm[i] in m) and (sm[i] in sg)

m1 := m1+[sm[i]];

m := m-m1;

sm[i] in sg

m=[]

с

Конец

J := 1, length(s)

‘Таких согласных нет’

2.2.8. Таблица идентификаторов

20

Лист

## № Документа

## Подпись

Изм.

## Лист

Таблица 2.2.8.1.Таблица идентификаторов

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| № | Имя переменной | Семантика | Тип | Разрядность |
| 1 | f | Файл, из которого вводятся слова в s | TextFile |  |
| 2 | s | Последовательность слов | string | 0..255 |
| 3 | sm | Последовательность слов | string | 0..255 |
| 4 | m | Множество букв в строке s | Set of char | 0..255 |
| 5 | M1 | Множество букв в строке | Set of char | 0..255 |
| 6 | sg | Множество согласных букв | Set of char | 0..255 |
| 7 | i | Текущий символ | byte | 0..255 |
| 8 | j | Текущий символ | byte | 0..255 |
| 9 | c | Выводимая буква | char | 0..255 |

2.2.9. Контрольные примеры

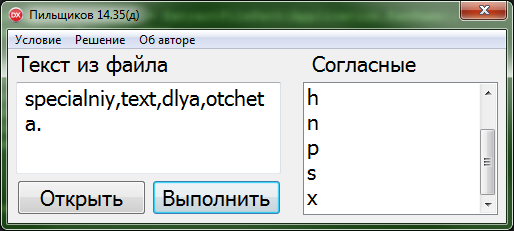


Рисунок 2.2.9.1. Контрольный пример №1

2.2.10. Организация ввода-вывода

21

Лист

## № Документа

## Подпись

Изм.

## Лист

Для ввода-вывода данных в программе использовались такие компоненты как Label и Memo. Они оба показаны ниже, на рисунке 2.2.10.1.

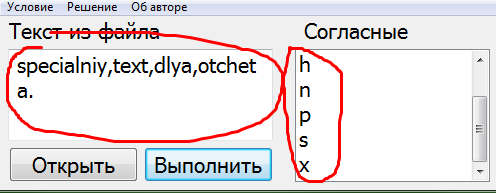


Рисунок 2.2.10.1. Демонстрация ввода-вывода данных

* + 1. Листинг проблемной части

procedure TForm1.Button2Click(Sender: TObject);

var

i, j: byte;

c: char;

begin

ListBox1.Clear;

s[length(s)] := ',';

m := [];

sm := '';

sg := ['a'..'z'] - ['a', 'e', 'i', 'o', 'u'];

for i := 1 to length(s) do

if s[i]<>',' then

sm := sm+s[i];

for i := 1 to length(sm) do

if not(sm[i] in m) and (sm[i] in sg) then

m := m+[sm[i]]

else if sm[i] in sg then

m1 := m1+[sm[i]];

m := m-m1;

if m=[] then

Listbox1.Items.Add('Нет букв по условию')

else

begin

Listbox1.Items.Add('Буквы по условию:');

for c := 'b' to 'z' do

if c in m then

Listbox1.Items.Add(c);

end;

end;

2.3. Задача 3

22

Лист

## № Документа

## Подпись

Изм.

## Лист

2.3.1. Постановка задачи

Необходимо графически изобразить функцию на декартовых координатах. Пользователь выбирает из выпадающего списка функцию, задает нижний и верхний пределы интегрирования. По нажатию на кнопку "Построить" строится интеграл от нижнего до верхнего предела с заданным масштабом (поменять его можно в поле "Масштаб"). Интегрирование производится методом средних прямоугольников.

2.3.2. Исходные данные

x1, x2 – верхний и нижний пределы интегрирования (real);

Функция заданная в ComboBox1.

Scale – заданный масштаб([10..120]).

2.3.3. Результат

Acc1 – значение интеграла;

N – кол-во разбиений диапазона.

2.3.4. Математический метод

Пусть f(x)=2\*x\*x-4\*x, x1 = -5, x2 = 5, тогда на рисунке 2.3.4.1 показан результат работы калькулятора в интернете. N =1280.

Необходимо узнать шаг (h), по формуле h = abs(x1-x2)/n (н.з. n =10)

Начинаем с x= x1+h/2; Получаем значение функции от x и кладем его в acc1(в acc2, хранится предыдущее значение). Продолжаем пока abs(acc1-acc2)<=eps.

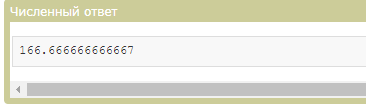


Рисунок 2.3.4.1. Решение онлайн-калькулятора

23

Лист

## № Документа

## Подпись

Изм.

## Лист

2.3.5. Описание подпрограмм

Custom\_func – эта подпрограмма задает функцию для интегрирования.

decBtn, incBtn – уменьшают и увеличивают масштаб.

drawCoord – рисует координатную плоскость на Image1.Canvas.

drawFunc – закрашивает область интеграла.

drawSqFunc – вычисляет интеграл методом средних прямоугольников.

24

Лист

## № Документа

## Подпись

Изм.

## Лист

2.3.6. Макро блок-схема

Начало

Пока программа не закрыта

Действие

Об авторе

Условие

Решение

Увеличить\Уменьшить масштаб

Построить

Конец

2.3.7. Блок-схема подпрограмм

drawSqFunc

J := 1, n

eps := 0.001;

n := 10;

acc1 := acc1 +custom\_func(x); x := x+h;

acc2 := acc1;

acc1 := 0;

acc1 := acc1\*h;

n := n\*2

h := abs(x2-x1)/n;

x := x1+h/2;

abs(acc1-acc2)<=eps

Acc1, n div2

Конец

25

Лист

## № Документа

## Подпись

Изм.

## Лист

2.3.8. Таблица идентификаторов

Все идентификаторы приведены в таблице 2.3.8.1. ниже:

2.3.8.1. Таблица идентификаторов задачи 3

|  |  |  |  |
| --- | --- | --- | --- |
| Имя | Семантика | Тип | Диапазон значений |
| N | Количество делений необходимых, что бы x достигло x2 c шагом h | Integer | -32 768.. 32 767 |
| eps | Точность вычисления | real | 2,9e-39.. 1,7e38 |
| Acc1, acc2 | Текущее и предыдущее значение интеграла | real | 2,9e-39.. 1,7e38 |
| x | Значение для функции custom\_func | real | 2,9e-39.. 1,7e38 |
| h | шаг | real | 2,9e-39.. 1,7e38 |

Продолжение таблицы 2.3.8.1

|  |  |  |  |
| --- | --- | --- | --- |
| i | Текущий прямоугольник | integer | -32 768.. 32 767 |
| X1, x2 | Левая и правая граница интеграла | real | 2,9e-39.. 1,7e38 |
| Func\_x | Х для закрашивания интеграла | real | 2,9e-39.. 1,7e38 |
| scale | Масштаб | 0..120 | 0..255 |

2.3.9. Контрольные примеры

26

Лист

## № Документа

## Подпись

Изм.

## Лист

На рисунке 2.3.9.1 показан интеграл 2\*x\*x-4\*x

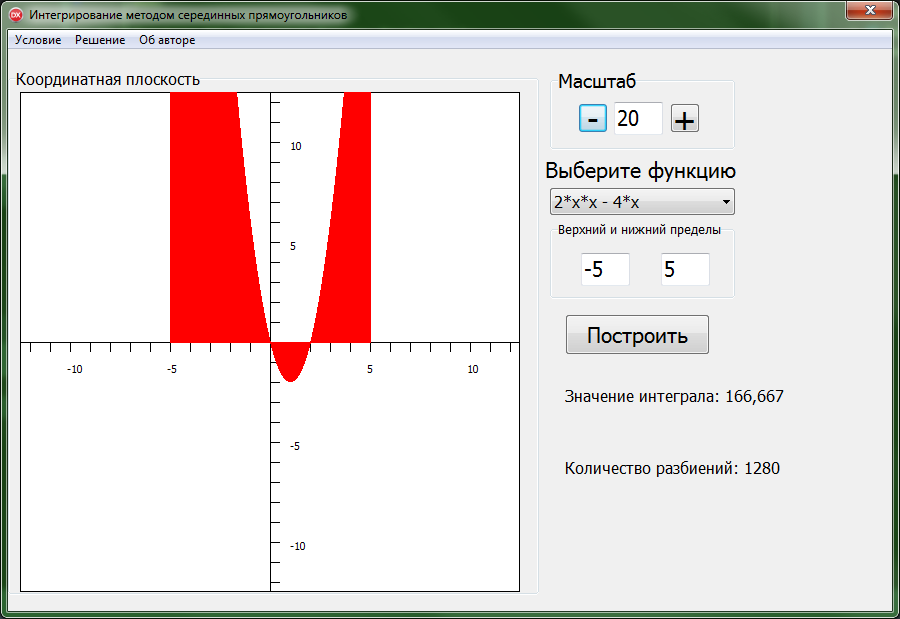
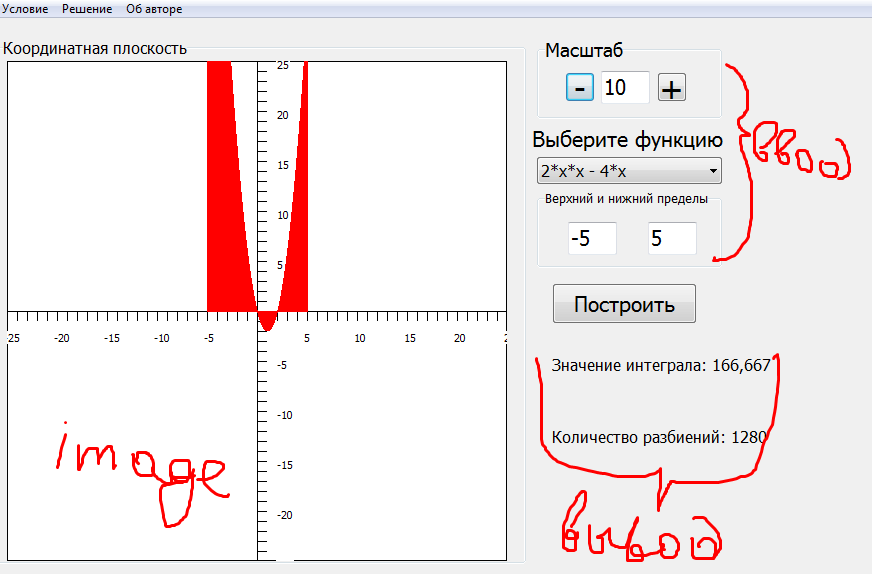


Рисунок 2.3.9.1.

Как можно увидеть программа строит интеграл от -5 до 5 по функции 2\*х\*х-4\*х.

2.3.10. Организация ввода-вывода

На вводе пользователь может выбрать масштаб и функцию, а так же задать пределы интегрирования. Вывод значения интеграла и кол-ва разбиений происходит в Label, интеграл рисуется на Image1 с помощью Canvas.



27

Лист

## № Документа

## Подпись

Изм.

## Лист

Рисунок 2.3.10.1. Демонстрация ввода-вывода

2.3.11. Листинг проблемной части

procedure TForm1.drawSqFunc;

var

eps, acc1, acc2, h, x: real;

n, i: integer;

begin

eps := 0.001;

n := 10;

repeat

acc2 := acc1;

acc1 := 0;

h := abs(x2-x1)/n;

x := x1+h/2;

for i := 1 to n do

begin

acc1 := acc1 + abs(custom\_func(x));

x := x+h;

end;

acc1 := acc1\*h;

n := n\*2

until abs(acc1-acc2)<=eps;

Label1.Caption := 'Значение интеграла: ' + FormatFloat('0.000', acc1);

Label2.Caption := 'Количество разбиений: ' + IntToStr(n div 2);

end;

1. Заключение

28

Лист

## № Документа

## Подпись

Изм.

## Лист

На этой практике было проделано много работы, решено более 37 задач, был получен опыт в отладке программ.

Необходимо было оформить задачи в отчет, с чем возникли некоторые трудности с оформлением блок-схем и их «привязкой» к странице.

Был получен большой опыт при решении задач на дату и время, особенно Абрамов\_829, где необходимо было подсчитать количество пятниц 13-го в веке.

Так же большой задачей это Интегрирование, в ней возникали некоторые трудности: либо правильно рисовался интеграл, либо правильно считался.

На этой практике был получен опит при работе с множествами, массивами, матрицами и выводом графики. Эти знания и умения, несомненно, пригодятся на следующей практике, когда будет разрабатываться desktop-приложение.

Подводя итог, хотелось бы резюмировать, что все задачи необходимо делать все в срок, чтобы не доделывать ночами, так же, по возможности, решать как можно больше задач, а так же делать разминку для глаз, разминаться.

На следующей практике все предыдущие ошибки будут учтены и исключены.

29

Лист

## № Документа

## Подпись

Изм.

## Лист

4. Список литературы

1. Тетрадь по численным методам
2. Wikipedia
3. Bourabai.kz
4. Labs.ord.ru
5. Pascal.helpov.net