# Введение

Данный сборник задач содержит постановки к 22 инженерным, практическим заданиям, необходимым для развития первичных профессиональных умений и навыков, в том числе, первичных умений и навыков научно-исследовательской деятельности обучающихся.

К числу первичных профессиональных умений и навыков относится свободное использование имеющегося инструментария офисных приложений, таких как:

- *Microsoft Office Word* (подготовка отчётов по выполненным заданиям «Учебной практики»),

- *Microsoft Office Visio* (подготовка блок-схем алгоритмов составленных программ),

*- Microsoft Office Excel* (опыт использования электронных таблицам является базовым, необходимым для последующего проектирования, настройки, администрирования и сопровождения баз данных),

а также пакета прикладных программ *National Instruments LabView* (закрепление на практике навыков программирования и алгоритмизации в сочетании с применением различных математических расчётных методов из разделов:

- арифметика,

- геометрия,

- тригонометрия,

- линейная алгебра,

- теория функции комплексного переменного,

- дифференциальные уравнения,

- и других).

К числу первичных умений и навыков научно-исследовательской деятельности стоит отнести, прежде всего, отработку на практике аналитического вывода формул, построения алгоритмов решения инженерных задач, поиска способов решения инженерных задач, подбора методов соответствующих методов, обоснования принятых решений, формулировки развёрнутого и содержательного вывода о проделанной работе.

Счёт заданий, изложенных в данном сборнике, не случайно начинается с двойки («2»). Подразумевается, что в качестве первого задания «Учебной практики» обучающиеся выполняют отчётную документацию к решённой задаче алгоритмизации и программирования, постановка которой изложена учебно-методическом пособии [1]. Обучающимся необходимо внимательно следить за нумерацией, указанной в постановочных частях задач сборника, поскольку реальные номера задач, согласно программе «Учебной практики», отличаются от номеров пунктов, под которыми эти задачи изложены.

Задачи, входящие в состав сборника, изложены согласно следующей опорной схеме:

- наименование работы;

- постановка части работы, являющейся общей для группы обучающихся (актуально не для всех задач сборника);

- формулировка цели работы;

- пример выполнения общей части работы или её фрагмента (актуально не для всех задач сборника);

- полезные соотношения, необходимые для выполнения работы (актуально не для всех задач сборника);

- варианты индивидуального задания.

Сборник разработан авторами с целью повышения познавательной активности обучающихся, достигаемой за счёт уместных отсылок к справочной литературе, ранее пройденному материалу в курсах:

- «Высшей математики»,

- «Программирования и основ алгоритмизации»,

- «Физики».

Используемый пакет прикладных программ *National Instruments LabView* позволяет подготовить обучающихся к качественному восприятию и лучшему усвоению программной логики, программистской этики, электрических и интегральных схем, а значит и материалов последующих курсов:

- «Теоретических основ электротехники»,

- «Электроники»,

- «Микропроцессорных устройств систем управления». Также упомянутый пакет прикладных программ позволяет обучающимся лучше ориентироваться в материалах курсов:

- «Численные методы в инженерных расчётах»,

- «Математические основы теории систем»,

- «Теория автоматического управления»,

«Цифровая обработка сигналов»

- и других.

Особое внимание обучающимся следует уделить классификации задач, изложенных в сборнике. Отчёты, оформляемые по расчётным задачам, принципиально отличаются по своей структуре от задач программирования и алгоритмизации. Опорные структуры отчётов, оформляемых обучающимися по решённым задачам каждого класса, приведены в Приложениях 1, 2.

# 8 Задание «Разработка простого калькулятора в пакете прикладных программ *National Instruments LabView*».

В пакете прикладных программ *National Instruments* *LabView* разработать виртуальный прибор «Простой Калькулятор», позволяющий оператору по итогам работы этого виртуального прибора наблюдать результаты выполнения следующих арифметических операций:

* сложения двух элементов,
* вычитания двух элементов,
* умножения двух элементов,
* деления двух элементов.

В рамках разработки графического пользовательского интерфейса предусмотреть:

* две ячейки (контроллера) для ввода числовой информации (по количеству элементов, над которыми выполняются арифметические операции);
* четыре ячейки (индикатора) для вывода числовой информации (по количеству используемых арифметических операций);
* лампочку (логический индикатор) для вывода логической информации с целью реагирования на ошибочный ввод исходных данных – ноль в знаменателе.

Все интерфейсные элементы должны содержать значащие имена, отражающие назначение этих элементов, например, ячейки для ввода информации могут быть названы:

* «Операнд 1», «Операнд 2»;
* «Слагаемое 1 / Уменьшаемое / Множитель 1 / Делимое»,

«Слагаемое 2 / Вычитаемое / Множитель 2 / Делитель».

Необходимо продумать индивидуальный авторский дизайн графического пользовательского интерфейса «Простого Калькулятора». Двух одинаковых дизайнерских решений для виртуального прибора «Простой Калькулятор» в одной учебной студенческой группе быть не должно.

Изменения, вносимые в исходные данные, должны сказываться на отображаемых результатах в реальном времени.

При подготовке блок-диаграммы графического кода программы минимизировать количество пересечений линий связи, а также попытаться создать максимально компактную и читаемую схему.

**Выше сформулированы требования для выполнения общей части работы.**

**Цель работы (одна из возможных формулировок)**: закрепление навыков работы в пакете прикладных программ *National Instruments* *LabView*, развитие навыков алгоритмизации и графического программирования.

**Индивидуальная часть работы (переменные):**

В дополнительном файле *\*.vi* создать развитие структуры «Простого Калькулятора» посредством добавления в неё элементов ввода и вывода информации, а также индикации (в случае необходимости), достаточных для демонстрации результата расчёта выражения, заданного по варианту.

Без острой на то необходимости графический код не усложнять! Например, операция возведения в степень не должна вызывать потребности у автора виртуального прибора в использовании специальной функции возведения числа в некоторую степень *N*.

|  |  |  |
| --- | --- | --- |
| **Вар.** | **Выражение** | **Логический индикатор** |
| **1** | (a + b + c + d) \* (– a – b – c – d) | Не требуется |
| **2** | (a + b + c)2 | Не требуется |
| **3** | (a – b + c – d) \* (– a + b – c – d) | Не требуется |
| **4** | (b / a) – (d / c) | Требуется |
| **5** | (a + (b – d) \* c)2 | Не требуется |
| **6** | (d / (b + a)) – (c / (d + a)) | Требуется |
| **7** | 3 \* a + b2 – c + 5 \* d3 | Не требуется |
| **8** | –((b + c) / (a + d)) | Требуется |
| **9** | (a + (b \* c) / d) / (a / (b \* d) – c) | Требуется |
| **10** | ((b + 2) \* (a + 4) / (d + 6) \* (c + 8)) | Требуется |
| **11** | –((b + c) \* (a + d) / c2) | Требуется |
| **12** | 10 \* a + 15 \* b + 20 \* c + 25 \* d | Не требуется |
| **13** | 7 \* (a + b + c + d) \* (– a – 7 \* b – c – d) | Не требуется |
| **14** | (a + b + c + d) / (– a – b – c – d) | Требуется |
| **15** | (a + b + c + d)3 | Не требуется |
| **16** | (a – b + c – d) / (– a + b – c – d) | Требуется |
| **17** | (a / b) + (c / d) | Требуется |
| **18** | (a \* (b – d) + c)2 | Не требуется |
| **19** | (a / (c + a)) + (b / (d \* a)) | Требуется |
| **20** | ((b / a) / (d / c)) – ((a / b) / (c / d)) | Требуется |
| **21** | 5 \* a + b2 + c – 3 \* d3 | Не требуется |
| **22** | –((a + b) / (d – c)) | Требуется |
| **23** | (a – (b \* c) / d) \* (a / (b \* d) + c) | Требуется |
| **24** | 10 \* d – 5 \* c + 20 \* a – 15 \* b | Не требуется |
| **25** | –((b – c) \* (a + d) / (c + 4)) | Требуется |
| **26** | 3 \* (a + c + d) \* (–7 \* b – d) | Не требуется |
| **27** | (a + b + c + d)4 | Не требуется |
| **28** | (a \* (c + a)) / (b \* (d – a)) | Требуется |
| **29** | (a / b)2 + (c / d)2 | Требуется |
| **30** | (a2 + b2 + c2) | Не требуется |

**Индивидуальная часть работы (константы):**

В ещё одном дополнительном файле *\*.vi* создать независимый от двух предыдущих виртуальных приборов иной виртуальный прибор, который смоделировал бы правильный ответ на сложное выражение, заданное по варианту.

Графический пользовательский интерфейс должен содержать в своём составе в удобной для чтения форме исходное сложное выражение, заданное по варианту, а также индикатор, выдающий правильный ответ на упомянутое сложное выражение.

Варианты сложных выражений для данного задания собраны в параграфе 1.4.

# 9 Задание «Моделирование термометров в пакете прикладных программ *National Instruments LabView*».

В пакете прикладных программ *National Instruments* *LabView* создать учебный виртуальный прибор «Термометры», моделирующий конвертацию температуры, заданной в градусах Цельсия, к другим единицам измерения, таким как градусы:

* Кельвина,
* Фаренгейта,
* Ранкина,
* Реомюра,
* Рёмера,
* Ньютона,
* Делиля.

Настроить шкалы деления виртуальных термометров таким образом, чтобы предельные значения шкалы основного термометра, задающего градусы Цельсия, совпадали с предельными значениями дополнительных термометров, показывающих иные единицы измерения **по графическому уровню**.

Величины предельных значений каждого термометра могут и должны отличаться друг от друга. Например, если для градусов Цельсия задан диапазон от -100 до 100, то для градусов Кельвина должен быть задан диапазон от 173,15 до 373,15.

Для повышения индивидуальности выполняемых обучающимися работ по созданию виртуальных приборов диапазоны шкал термометров в градусах Цельсия задаются согласно таблице вариантов.

Таблица 1. Диапазоны шкал виртуальных термометров

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **1** | **2** | **3** | **4** | **5** |
| -30…30 | 2.9…6.1 | -1…99 | 0.12…3.45 | -1000…1000 |
| **6** | **7** | **8** | **9** | **10** |
| 5…55 | 0.1…0.9 | -70.77…15.58 | 0…10 | -50…0 |
| **11** | **12** | **13** | **14** | **15** |
| 0…80 | 12.77…40.92 | -500…300 | -15…15 | -20…-10 |
| **16** | **17** | **18** | **19** | **20** |
| -78.9…-12.3 | -5…55 | -45…10 | -0.9…0.1 | 0…100 |
| **21** | **22** | **23** | **24** | **25** |
| 300…500 | -22…22 | -10000…10 | 0.001…0.1 | -0.001…0 |
| **26** | **27** | **28** | **29** | **30** |
| 1…101 | 0.9…30.9 | 0.0001…1 | 0…4 | -23.4…5.67 |

И, поскольку, прибор является лабораторной установкой – он должен содержать на передней панели заголовок, авторство и год выпуска.

**Внимание!** Для случаев, когда стандартных размеров термометра недостаточно для нормального отображения на шкале заданного диапазона, необходимо принудительно увеличить размеры термометра посредством использования инструмента позиционирования элементов (*Position/Size/Select*).

В процессе проектирования виртуального прибора необходимо решить вопрос рационального размещения функциональных элементов, как на графическом пользовательском интерфейсе – передней панели, так и на блок-диаграмме с кодом. Всё должно быть читаемым, обладать минимальным количеством пересечений линий связи, а также быть понятным не только разработчику виртуального прибора, но и самому-самому неопытному пользователю программного обеспечения.

На блок-диаграмме все функциональные, то есть значащие пересечения линий связи (узлы) обозначить точками. Для этого в меню перейти к настройкам «*Tools > Options…*», далее в выпадающем списке перейти к настройкам блок-диаграммы (*Block Diagram*) и выставить галочку напротив пункта «*Show dots at wire junctions*».

Правила конвертации одних температурных величин в другие представлены в разделе «соотношения, необходимые для выполнения работы».

Чётные варианты как в общей, так и в индивидуальной части работы используют в качестве задающего элемента термометр опорной измерительной системы (в общей части опорной системой измерения являются градусы Цельсия);

нечётные варианты используют в качестве задающего элемента – числовой контроллер опорной системы.

**Цель работы (одна из возможных формулировок)**: ознакомление с интерфейсом пакета прикладных программ *National Instruments* *LabView*, его элементами, настройками и арифметическими операциями. Закрепление навыков составления блок-схем алгоритмов к авторским прикладным программам с разбиением этих блок-схем по процедурам и функциям.

**Соотношения, необходимые для выполнения работы:**

1. Перевод из градусов Цельсия в градусы Кельвина:;

2.1. Перевод из градусов Фаренгейта в градусы Цельсия: ;

2.2. Перевод из градусов Цельсия в градусы Фаренгейта: ;

3. Соответствие градусов Цельсия градусам Ранкина: 

4. Соответствие градусов Цельсия градусам Реомюра: 

5. Соответствие градусов Цельсия градусам Рёмера: ;

6. Соответствие градусов Цельсия градусам Ньютона: ;

7. Перевод из градусов Цельсия в градусы Делиля: 

**Индивидуальная часть задания:**

Выполнить второй виртуальный прибор для исследования конвертации температур, в котором за основу взять единицы измерения, заданные по варианту.

|  |  |
| --- | --- |
| **Вар.** | **Содержание задания** |
| 1. | Выполнить прямой перевод из градусов Кельвина в градусы Делиля.  Цвет входного термометра оранжевый выходного – фиолетовый. |
| 2. | Выполнить прямой перевод из градусов Фаренгейта в градусы Реомюра.  Цвет входного термометра зелёный выходного – красный. |
| 3. | Выполнить прямой перевод из градусов Делиля в градусы Кельвина.  Оба термометра оливкового цвета. |
| 4. | Выполнить прямой перевод из градусов Ньютона в градусы Реомюра.  Оба термометра фиолетового цвета. |
| 5. | Выполнить прямой перевод из градусов Фаренгейта в градусы Рёмера.  Цвет входного термометра серый выходного – розовый. |
| 6. | Выполнить прямой перевод из градусов Реомюра в градусы Рёмера.  Оба термометра зелёного цвета. |
| 7. | Выполнить прямой перевод из градусов Кельвина в градусы Ранкина.  Цвет входного термометра синий выходного – зелёный. |
| 8. | Выполнить прямой перевод из градусов Делиля в градусы Цельсия.  Оба термометра серые. |
| 9. | Выполнить прямой перевод из градусов Фаренгейта в градусы Ранкина.  Цвет входного термометра фиолетовый выходного – синий. |
| 10. | Выполнить прямой перевод из градусов Ньютона в градусы Кельвина.  Цвет входного термометра салатовый выходного – алый. |
| 11. | Выполнить прямой перевод из градусов Делиля в градусы Реомюра.  Цвет входного термометра голубой выходного – серый. |
| 12. | Выполнить прямой перевод из градусов Кельвина в градусы Рёмера.  Цвет входного термометра чёрный выходного – салатовый. |
| 13. | Выполнить прямой перевод из градусов Ньютона в градусы Делиля.  Цвет входного термометра бирюзовый выходного – лайм. |
| 14. | Выполнить прямой перевод из градусов Фаренгейта в градусы Цельсия.  Цвет входного термометра синий выходного – красный. |
| 15. | Выполнить прямой перевод из градусов Делиля в градусы Рёмера.  Цвет входного термометра красный выходного – коричневый. |
| 16. | Выполнить прямой перевод из градусов Кельвина в градусы Реомюра.  Цвет входного термометра розовый выходного – салатовый. |
| 17. | Выполнить прямой перевод из градусов Делиля в градусы Ранкина.  Цвет входного термометра чёрный выходного – красный. |
| 18. | Выполнить прямой перевод из градусов Фаренгейта в градусы Кельвина.  Цвет входного термометра жёлтый выходного – белый. |
| 19. | Выполнить прямой перевод из градусов Ньютона в градусы Цельсия.  Цвет морской волны для входного термометра выходного – красный. |
| 20. | Выполнить прямой перевод из градусов Делиля в градусы Фаренгейта.  Цвет входного термометра голубой выходного – оранжевый. |
| 21. | Выполнить прямой перевод из градусов Кельвина в градусы Цельсия.  Цвет входного термометра белый выходного – чёрный. |
| 22. | Выполнить прямой перевод из градусов Рёмера в градусы Реомюра.  Цвет входного термометра синиё выходного – жёлтый. |
| 23. | Выполнить прямой перевод из градусов Фаренгейта в градусы Делиля.  Оба термометра салатового цвета. |
| 24. | Выполнить прямой перевод из градусов Реомюра в градусы Ньютона.  Оба термометра оранжевого цвета. |
| 25. | Выполнить прямой перевод из градусов Кельвина в градусы Ньютона.  Цвет входного термометра оливковый выходного – оранжевый. |
| 26. | Выполнить прямой перевод из градусов Фаренгейта в градусы Ньютона.  Цвет входного термометра синий выходного – жёлтый. |
| 27. | Выполнить прямой перевод из градусов Рёмера в градусы Цельсия.  Цвет входного термометра серый выходного – розовый. |
| 28. | Выполнить прямой перевод из градусов Делиля в градусы Ньютона.  Оба термометра белого цвета. |
| 29. | Выполнить прямой перевод из градусов Кельвина в градусы Фаренгейта.  Цвет входного термометра красный выходного – синий. |
| 30. | Выполнить прямой перевод из градусов Ньютона в градусы Рёмера.  Цвет входного термометра алый выходного – бледно-зелёный. |

# 10 Задание «Разработка калькулятора с использованием *CASE* структуры в пакете прикладных программ *National Instruments LabView*».

В пакете прикладных программ *National Instruments* *LabView* разработать виртуальный прибор «Умный калькулятор», в состав которого входят операции, заданные согласно таблице вариантов (Таблица 3).

Операции должны быть размещены на графическом пользовательском интерфейсе строго в том порядке, в котором они упоминаются в таблице исходных вариантов. Определение именно той операции, которую пожелал реализовать пользователь, должно выполняться в соответствии с некоторой логикой, задаваемой булевскими элементами управления, например, кнопками или тумблерами.

В процессе разработки графического пользовательского интерфейса предусмотреть:

* числовые контроллеры в необходимом количестве для ввода значений операндов,
* один единственный числовой индикатор для вывода результата/ответа,
* логические контроллеры в необходимом количестве для однозначного определения системой выполняемой операции.

Результат/ответ выводить на индикатор только при единственном включённом логическом контроллере.

Исключить одновременное включение нескольких логических контроллеров и либо не реагировать на подобный входной сигнал – оставлять на выходе ноль, либо при включении нескольких контроллеров выдавать ошибки, формируемые в виде числового кода, например, «9999», «8888», которые следует расшифровать для пользователя на графическом пользовательском интерфейсе дополнительной таблицей соответствия.

Далее приведены примеры расшифровок кодов, которые могут быть размещены на графическом пользовательском интерфейсе:

* «9999» – недопустимо одновременное умножение и деление,
* «8888» – недопустимо одновременное сложение и вычитание,
* «7777» – недопустимо деление на ноль,
* и так далее.

Автору требуется продумать структуру других возможных ошибок самостоятельно.

На блок-диаграмме все функциональные, то есть значащие пересечения линий связи (узлы) обозначить точками. Для этого в меню перейти к настройкам «*Tools > Options…*», далее в выпадающем списке перейти к настройкам блок-диаграммы (*Block Diagram*) и выставить галочку напротив пункта «*Show dots at wire junctions*».

**Цель работы (одна из возможных формулировок)**: закрепление навыков работы и навыков графического программирования в пакете прикладных программ *National Instruments* *LabView*. Приобретение навыков работы c *CASE*-структурой, входящей в состав пакета прикладных программ *National Instruments* *LabView*.

**Индивидуальное задание:**

Таблица 3 – Исходные данные для выполнения индивидуального задания

|  |  |
| --- | --- |
| **Вар.** | **Содержание работы** |
| **1.** | Использовать **три** операнда.  Предусмотреть операции:  - сложения;  - инверсии суммы;  - инверсии суммы объединённой арифметикой;  - сложения объединённой арифметикой. |
| **2.** | Использовать **два** операнда.  Предусмотреть операции:  - умножения;  - деления первого на второй;  - деления второго на первый;  - масштабирования на степень двойки;  - логарифма второго по основанию первого. |
| **3.** | Использовать **один** операнд.  Предусмотреть операции:  - абсолютного значения;  - сигнума;  - синуса;  - экспонирования. |
| **4.** | Использовать **два** операнда.  Предусмотреть операции:  - деления по модулю первого;  - деления по модулю второго;  - инкрементирования первого. |
| **5.** | Использовать **один** операнд.  Предусмотреть операции:  - секанса;  - натурального логарифма;  - декрементирования;  - инверсии;  - получения Пифагоровой константы. |
| **6.** | Использовать **четыре** операнда.  Предусмотреть операции:  - инверсии суммы объединённой арифметикой;  - инверсии разности объединённой арифметикой. |
| **7.** | Использовать **два** операнда.  Предусмотреть операции:  - арктангенса с двумя параметрами;  - арктангенса первого;  - арктангенса второго;  - тангенса первого;  - тангенса второго. |
| **8.** | Использовать **один** операнд.  Предусмотреть операции:  - экспонирования;  - взятия логарифма по основанию 2;  - натурального логарифма;  - десятичного логарифма. |
| **9.** | Предусмотреть вывод констант:  - Пифагоровой;  - основания натурального логарифма;  - гравитационной;  - нечислового значения;  - плюс бесконечности;  - минус бесконечности. |
| **10.** | Использовать **два** операнда.  Предусмотреть операции:  - остатка от деления первого на второй;  - остатка от деления второго на первый. |
| **11.** | Использовать **три** операнда.  Предусмотреть операции:  - квадратного корня из суммы квадратов;  - квадратного корня из модуля разности квадратов;  - квадратного корня из модуля разности;  - квадратного корня из модуля суммы. |
| **12.** | Использовать **один** операнд.  Предусмотреть операции:  - арккосинуса;  - округления до ближайшего целого;  - квадратного корня. |
| **13.** | Использовать **три** операнда.  Предусмотреть операции:  - нахождения периметра треугольника;  - нахождения площади треугольника;  - нахождения высоты треугольника. |
| **14.** | Использовать **два** операнда.  Предусмотреть операции:  - обращение суммы;  - корень квадратный из суммы;  - деление второго на первый;  - масштабирование на степень двойки;  - логарифм второго по основанию первого. |
| **15.** | Использовать **один** операнд.  Предусмотреть операции:  - нахождения диаметра окружности;  - нахождения площади круга;  - нахождения длины окружности. |
| **16.** | Использовать **четыре** операнда.  Предусмотреть операции:  - нахождения периметра прямоугольника;  - нахождения площади прямоугольника;  - нахождения диагонали прямоугольника. |
| **17.** | Использовать **три** операнда.  Предусмотреть операции:  - нахождения объёма фигуры;  - нахождения площади поперечного сечения фигуры;  - нахождения площади продольного сечения фигуры. |
| **18.** | Использовать **один** операнд.  Предусмотреть операции:  - взятия квадратного корня;  - тангенса;  - обращения;  - возведения двойки в степень. |
| **19.** | Использовать **один** операнд.  Предусмотреть операции:  - математического округления;  - округления в большую сторону;  - округления в меньшую сторону. |
| **20.** | Использовать **два** операнда.  Предусмотреть операции:  - вывода численной константы, равной «8»;  - вычитания второго из первого;  - вычитания первого из второго;  - суммы синуса первого и косинуса второго (синус и косинус в одной операции). |
| **21.** | Использовать **один** операнд.  Предусмотреть операции:  - вычисления значения функции отсчётов;  - декрементирования. |
| **22.** | Использовать **три** операнда.  Предусмотреть операции:  - произведения трёх;  - разности суммы первого и третьего из второго;  - суммы трёх;  - частного произведения второго на третье на первое. |
| **23.** | Использовать **один** операнд.  Предусмотреть операции:  - косинуса;  - степени десяти;  - вывода постоянной Планка;  - инкрементирования;  - декрементирования. |
| **24.** | Предусмотреть вывод констант:  - нечисловое значение;  - Авогадро;  - обращённой Пифагорову;  - натурального логарифма десяти. |
| **25.** | Использовать **три** операнда.  Предусмотреть операции:  - вычисления периметра треугольника;  - вычисления высоты треугольника;  - вычисления площади треугольника. |
| **26.** | Использовать **один** операнд.  Предусмотреть операции:  - возведения двойки в степень;  - котангенса;  - вывода обращённого основания натурального логарифма. |
| **27.** | Использовать **два** операнда.  Предусмотреть операции:  - синуса суммы;  - экспонирования суммы;  - обращения суммы;  - возведения двойки в степени суммы. |
| **28.** | Использовать **шесть** операндов.  Предусмотреть операции:  - произведения объединённой арифметикой;  - суммирования объединённой арифметикой. |
| **29.** | Использовать **один** операнд.  Предусмотреть операции:  - косеканса;  - натурального логарифма, проходящего через начало координат. |
| **30.** | Использовать **два** операнда.  Предусмотреть операции:  - косинуса разности;  - вычитания из первого квадрата тангенса второго. |

# 11 Задание «Проекция функций на полярную систему координат в *National Instruments LabView*».

В пакете прикладных программ *National Instruments LabView* создать виртуальный прибор «Полярные координаты», в рамках которого выполнить построение графика заданной по варианту функции (Таблица 1) в декартовой системе координат.

Виртуальный прибор должен предусматривать возможность ввода пользователем диапазона построения графика функциональной зависимости в формате:

* начало диапазона,
* конец диапазона,
* шаг дискретизации.

Исключить ситуацию ошибочного запуска виртуального прибора с нулевым значением шага построения. Любым известным способом.

Учащимся с чётными номерами вариантов следует предусмотреть на графическом пользовательском интерфейсе только одну область для построения графика «*XY Graph*», куда в зависимости от указанного диапазона построения графика функции в декартовой системе координат **центрировано\*** выводить изображение графика функции в полярной системе координат, а также изображение графика функции в Декартовой системе координат.

Учащимся с нечётными номерами вариантов следует предусмотреть на графическом пользовательском интерфейсе две области для построения графика «*XY Graph*», в одну из которых выводить изображение графика функции в Декартовой системе координат, в другую – в полярной системе координат.

\* Под **центрированным** изображением понимается вывод полярного графика функциональной зависимости в окрестностях начала координат, то есть график необходимо «подвинуть» (сместить) на середину (половину) диапазона построения.

Рекомендации по элементной базе для составления кода на блок-диаграмме:

* *CASE*-структура для распознавания нулевого шага построения,
* циклическая структура *For* для перебора значений заданного диапазона с шагом построения,
* смещение точки нуля на половину ширины диапазона для **центрирования** (сложение, вычитание, деление),
* элементы сцепления *Bundle* для объединения координат *x* и *y* в кластер типа «точка графика»,
* элемент *Build Array* для объединения «точек графика» в массив «точек графика», позволяющий выводить изображение нескольких функций в единых координатных осях по различным трассировкам.

На блок-диаграмме все функциональные, то есть значащие пересечения линий связи (узлы) обозначить точками. Для этого в меню перейти к настройкам «*Tools > Options…*», далее в выпадающем списке перейти к настройкам блок-диаграммы (*Block Diagram*) и выставить галочку напротив пункта «*Show dots at wire junctions*».

**Полезные соотношения для выполнения работы:**

Общепринятая функциональная зависимость для построения графика в полярной системе координат: .

Функциональная зависимость для построения графика в полярной системе координат, удобная для понимания и восприятия обучающимися: .

Проецирование на окружность функциональной зависимости: , .

**Цель работы (одна из возможных формулировок)**: закрепление навыков использования графической структуры цикла *For* (или структуры цикла *While* в зависимости от реализации), приобретение навыков конвертации и проецирования значений периодических и не периодических функций на окружность (преобразования в полярную систему координат).

**Варианты индивидуального задания для построения графиков функциональных зависимостей в полярной системе координат:**

Таблица 1 – Варианты индивидуального задания

|  |  |
| --- | --- |
| 1. *sin(3·x)* | 16. *x·sin(x)* |
| 2. *sin(x)·cos(x)* | 17. *sin(cos(sin(a·x)))* |
| 3. *sin(x)·cos(3·x)2* | 18. *1 – cos(x2)2* |
| 4. *x + sin(2·x)* | 19. *cos(sin(4·x))·sin(x)* |
| 5. *1 – cos(3·x)* | 20. *(sin(3·x) ·x)2* |
| 6. *cos(sin(a·x))* | 21. *x2·sin(2·x)* |
| 7. *sin(3·x) ·cos(x)* | 22. *sin(2·x)· cos(sin(3·x))* |
| 8. *sin(cos(x))2* | 23. *cos(x2) + x + sin(2·x)* |
| 9. *sin(a·x) + cos(b·x)* | 24. *sin(2·x)·cos(4·x)* |
| 10. *1 – cos(x)* | 25. *1 – sin(2·x)* |
| 11. *sin2(x)* | 26. *sin(3·x) – cos(2·x)* |
| 12. *x3·cos2(x)* | 27. *sin(x)2·cos(3·x)* |
| 13. *sin(a·x)·cos(b·x)* | 28. *x2·sin(x)* |
| 14. *sin(2·x)· cos(sin(2·x))* | 29. *sin(2·x)· cos(sin(4·x))* |
| 15. *(x + sin(x)) 2* | 30. *x2+2·x+1* |

# 12 Задание «Программа-игра «Угадай число» в *National Instruments LabView*».

В пакете прикладных программ *National Instruments LabView* создать виртуальный прибор, представляющий собой игру «Угадай число». «Угадай число» – однопользовательская игра, игра человека с персональным компьютером.

Суть игры заключается в следующем: пользователь задаёт через графический интерфейс любое целое число, а компьютер, посредством перебора различных реализаций равномерно распределённой случайной величины, должен «угадать» это число.

В качестве выходной информации, демонстрируемой на индикаторе в рамках виртуального прибора «Угадай число», принять количество предпринятых компьютером попыток угадать введённое пользователем значение. Количество попыток совпадает с количеством вхождений в тело (итераций) цикла.

Для построения программного обеспечения использовать:

* структуру *While*,
* генератор псевдослучайных чисел *Random*,
* округление до ближайшего целого,
* умножение на константу, задающую диапазон поиска загаданного числа (в общей части на усмотрение автора, в индивидуальной – согласно варианту),
* смещение диапазона на константу влево/вправо (в общей части на усмотрение автора, в индивидуальной – согласно варианту),
* логику прерывания цикла,
* значения, строго расположенные внутри заданного диапазона.

**Обратите внимание!** Счётчик количества итераций цикла (параметр цикла) начинается с нуля.

На блок-диаграмме все функциональные, то есть значащие пересечения линий связи (узлы) обозначить точками. Для этого в меню перейти к настройкам «*Tools > Options…*», далее в выпадающем списке перейти к настройкам блок-диаграммы (*Block Diagram*) и выставить галочку напротив пункта «*Show dots at wire junctions*».

**Цель работы (одна из возможных формулировок)**: освоение навыков использования циклической структуры *While*, а также навыков расширения диапазона значений для датчика случайных чисел в пакете прикладных программ *National Instruments* *LabView*.

**Варианты индивидуальной части работы:**

Во втором виртуальном приборе выполнить развитие игры «Угадай число», дополнив его надстройкой, содержащей иное представление выходной информацией и изменив масштаб диапазона поиска значений в соответствии с заданным вариантом (в случае необходимости).

|  |  |
| --- | --- |
| **Вар.** | **Задание** |
| **1.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от 1 до 1000. Вывести на графический пользовательский интерфейс по итогам игры оценку интуиции согласно 10 бальной системе. Использовать логику прерывания по совпадению. При подготовке отчёта проиллюстрировать шкалу оценивания (какой оценке соответствует количество попыток). |
| **2.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от 1 до 100. Вывести на графический пользовательский интерфейс по итогам игры среднее арифметическое, посчитанное по значениям, предложенным в качестве ответа. Для проверки вывести значения в строку. Использовать логику продолжения по несовпадению. |
| **3.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от 1 до 10. Вывести на графический пользовательский интерфейс по итогам игры среднее геометрическое, посчитанное по значениям, предложенным в качестве ответа. Для проверки вывести значения в массив. Использовать логику прерывания по совпадению. |
| **4.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от 1 до 100. Вывести на графический пользовательский интерфейс по итогам игры сумму значений, предложенных в качестве ответа. Использовать логику продолжения по несовпадению. |
| **5.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от 1 до 1000. Вывести на графический пользовательский интерфейс по итогам игры сумму значений, предложенных в качестве ответа, вычисленную по модулю 10. Для проверки вывести значения в массив. Использовать логику прерывания по совпадению. |
| **6.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от 1 до 1000. Вывести на графический пользовательский интерфейс по итогам игры график, построенный по значениям, предложенным в качестве ответа, на конкретной итерации. Использовать логику продолжения по несовпадению. |
| **7.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от -300 до 300. Вывести на графический пользовательский интерфейс по итогам игры значения, предложенные в качестве ответа, одной строкой. Использовать логику прерывания по совпадению. |
| **8.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от 1 до 1000. Вывести на графический пользовательский интерфейс по итогам игры максимальное среди значений, предложенных в качестве ответа. Для проверки вывести значения в массив. Использовать логику продолжения по несовпадению. |
| **9.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от 1 до 1000. Вывести на графический пользовательский интерфейс по итогам игры оценку интуиции согласно 100 бальной шкале. Использовать логику прерывания по совпадению. При подготовке отчёта проиллюстрировать шкалу оценивания (какой оценке соответствует количество попыток). |
| **10.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от -100 до 100. Вывести на графический пользовательский интерфейс по итогам игры минимальное среди значений, предложенных в качестве ответа. Для проверки вывести значения в массив. Использовать логику продолжения по несовпадению. |
| **11.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от 1 до 100. Вывести на графический пользовательский интерфейс по итогам игры уникальные значения, предложенные в качестве ответа, одной строкой. Для проверки вывести значения в массив. Использовать логику прерывания по совпадению. |
| **12.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от -50 до 50. Вывести на графический пользовательский интерфейс по итогам игры сумму значений, предложенных в качестве ответа. Использовать логику продолжения по несовпадению. |
| **13.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от -50 до 50. Вывести на графический пользовательский интерфейс по итогам игры разницу, полученную между максимальным и минимальным значением, предложенным в качестве ответа. Использовать логику прерывания по совпадению. |
| **14.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от 1 до 1000. Вывести на графический пользовательский интерфейс по итогам игры среднее арифметическое, посчитанное по значениям, предложенным в качестве ответа. Для проверки вывести значения в массив. Использовать логику продолжения по несовпадению. |
| **15.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от -100 до 100. Вывести на графический пользовательский интерфейс по итогам игры график, построенный по значениям, предложенным в качестве ответа, на конкретной итерации. Использовать логику прерывания по совпадению. |
| **16.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от 1 до 100. Вывести на графический пользовательский интерфейс по итогам игры сумму значений, предложенных в качестве ответа, вычисленную по модулю 4. Для проверки вывести значения в строку. Использовать логику продолжения по несовпадению. |
| **17.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от 1 до 100. Вывести на графический пользовательский интерфейс по итогам игры максимальное среди значений, предложенных в качестве ответа. Для проверки вывести значения в строку. Использовать логику прерывания по совпадению. |
| **18.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от 1 до 10. Вывести на графический пользовательский интерфейс по итогам игры среднее геометрическое, посчитанное по значениям, предложенным в качестве ответа. Для проверки вывести значения в строку. Использовать логику продолжения по несовпадению. |
| **19.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от 1 до 100. Вывести на графический пользовательский интерфейс по итогам игры минимальное среди значений, предложенных в качестве ответа. Для проверки вывести значения в строку. Использовать логику прерывания по совпадению. |
| **20.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от 1 до 100. Вывести на графический пользовательский интерфейс по итогам игры оценку интуиции согласно 5 бальной системе. Использовать логику продолжения по несовпадению. При подготовке отчёта проиллюстрировать шкалу оценивания (какой оценке соответствует количество попыток). |
| **21.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от 1 до 100. Вывести на графический пользовательский интерфейс по итогам игры уникальные значения, предложенные в качестве ответа, в массив. Для проверки вывести значения в строку. Использовать логику прерывания по совпадению. |
| **22.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от -50 до 50. Вывести на графический пользовательский интерфейс по итогам игры среднее арифметическое, посчитанное по модулям значений, предложенных в качестве ответа. Для проверки вывести значения в строку. Использовать логику продолжения по несовпадению. |
| **23.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от 1 до 50. Вывести на графический пользовательский интерфейс по итогам игры разницу, полученную между минимальным и максимальным значением, предложенным в качестве ответа. Использовать логику прерывания по совпадению. |
| **24.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от 1 до 600. Вывести на графический пользовательский интерфейс по итогам игры значения, предложенные в качестве ответа, одной строкой. Использовать логику продолжения по несовпадению. |
| **25.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от -100 до 100. Вывести на графический пользовательский интерфейс по итогам игры сумму модулей значений, предложенных в качестве ответа. Использовать логику прерывания по совпадению. |
| **26.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от 1 до 100. Вывести на графический пользовательский интерфейс по итогам игры график, построенный по значениям, предложенным в качестве ответа, на конкретной итерации. Использовать логику продолжения по несовпадению. |
| **27.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от -100 до 100. Вывести на графический пользовательский интерфейс по итогам игры максимальное среди значений, предложенных в качестве ответа. Для проверки вывести значения в массив. Использовать логику прерывания по совпадению. |
| **28.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от -500 до 500. Вывести на графический пользовательский интерфейс по итогам игры оценку интуиции согласно 10 бальной системе. Использовать логику продолжения по несовпадению. При подготовке отчёта проиллюстрировать шкалу оценивания (какой оценке соответствует количество попыток). |
| **29.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от -500 до 500. Вывести на графический пользовательский интерфейс по итогам игры среднее арифметическое, посчитанное по значениям, предложенным в качестве ответа. Для проверки вывести значения в массив. Использовать логику прерывания по совпадению. |
| **30.** | Создать виртуальный прибор «Угадай число» с поиском по диапазону от -10 до 10. Вывести на графический пользовательский интерфейс по итогам игры сумму модулей значений, предложенных в качестве ответа, вычисленную по модулю 3. Для проверки вывести значения в строку. Использовать логику продолжения по несовпадению. |

# 13 Задание «Построение ломаной линии с использованием структуры *Formula Node* в *National Instruments LabView*».

В пакете прикладных программ *National Instruments* *LabView* выполнить построение ломаной линии, заданной своим изображением по варианту.

Основное требование: использовать структуру *Formula Node* (обёрнутую в циклическую конструкцию *While* или *For*), внутри которой разветвить вычислительный процесс для построения ломаной линии в диапазоне от начального значения (**xn**) до конечного значения (**xk**), с некоторым шагом (**h**), вводимых через графический пользовательский интерфейс.

С синтаксисом Си-образного языка программирования *Formula Node* ознакомиться через раздел *Help* соответствующей структуры.

**Для чётных вариантов:**

Выход влево и вправо за границы диапазона изображения отрисовывать константами (горизонтальными линиями), соответствующими значениям ординат в граничных точках изображения, заданного по варианту.

**Для нечётных вариантов:**

Выход влево и вправо за границы диапазона изображения отрисовывать с прогнозированием звена вида **y = kx + b** или константы в зависимости от того, что подходит на границе диапазона.

На блок-диаграмме все функциональные, то есть значащие пересечения линий связи (узлы) обозначить точками. Для этого в меню перейти к настройкам «*Tools > Options…*», далее в выпадающем списке перейти к настройкам блок-диаграммы (*Block Diagram*) и выставить галочку напротив пункта «*Show dots at wire junctions*».

**Цель работы (одна из возможных формулировок)**: освоение навыков использования условного оператора в пакете прикладных программ *National Instruments* *LabView*.

**Узел формул**

Вычисляет математические формулы и выражения на схожем с языком «Си» языке в рамках блок-диаграммы. Для Узла формул доступны следующие стандартные функции: *abs, acos, acosh, asin, asinh, atan, atanh, ceil, cos, cosh, cot, csc, exp, expm1, floor, getexp, getman, int, intrz, In, Inp1, log, log2, max, min, mod, rand, rem, sec, sign, sin, sinc, sinh, sqrt, tan, tanh*.

Более подробная информация по Узлам формул представлена в следующем разделе («Создание Узлов формул»).

**Создание Узлов формул**

Следующие шаги позволяют создать Узел формул:

1. Перейдите по разделам «Функции > Структуры > Узел формул» (*Functions > Structures > Formula Node*) или Функции > Математика > Формула > Узел формул (*Functions > Mathematics > Formula > Formula Node*) и перенесите компонент на блок-диаграмму. В соответствующих разделах, расположенных далее по тексту, можно посмотреть перечни функций и операторов, которые могут быть использованы в Узлах формул.

2. При выбранном инструменте ввода текста (или инструменте изменения значения) необходимо ввести выражения или расчётные соотношения внутри Узла формул. Каждое присвоение должно быть выполнено для одной единственной переменной, указанной слева от оператора присвоения (=). Каждое присвоение должно завершаться разделителем (;). В соответствующем разделе можно ознакомиться с синтаксисом Узла формул.

**Внимание:** в Узлах формул уместно добавление комментариев посредство заключения их в пару «косая черта – звёздочка» (/\*комментарий\*/), или вслед за двойной косой чертой (//комментарий).

3. Для каждой переменной, входящей в Узел формул извне, необходимо создать вход-клемму, нажав правой кнопкой мыши на границе Узла формул и выбрав «Добавить вход» (*Add Input***)** в контекстном меню. В появившееся окно клеммы с рыжей рамкой необходимо вписать имя переменной (**Внимание:** предварительно необходимо проверить, чтобы клавиатура находилась в английской раскладке). Редактировать имена клемм возможно в любое время, за исключением ситуаций, когда виртуальный прибор запущен на исполнение.

4. Клеммы являются регистрочувствительными элементами. Количество клемм и выражений в Узлах формул не ограничено. Имеется возможность для изменения типа клемм (вход / выход).

5. Выходная клемма для выходной переменной создаётся следующим образом: нажать правой кнопкой мыши на границе Узла формул и в раскрывшемся контекстном меню выбрать «Добавить выход» (*Add Output*). В возникшем окне напечатать имя выходной клеммы (**Внимание:** предварительно необходимо проверить, чтобы клавиатура находилась в английской раскладке). Изменить имя клеммы можно в любое время за исключением случаев, когда виртуальный прибор запущен на исполнение. Выходные клеммы обладают более толстыми рамками по сравнению со входными.

**Внимание:** в Узлах формул не может существовать двух одинаково названных входов и двух одинаково названных выходов, однако, выходы и входы могут иметь одинаковые имена.

(Не обязательная для нормального использования компонента информация) Выходные клеммы по умолчанию выставляются к вещественному типу данных двойной точности. Для изменения типа данных на клеммах, необходимо создать входную клемму с именем, которое в точности совпадает с именем выходной и связать входную клемму с элементом интересующего типа данных. Это действие приводит к изменению типа данных, заданного по умолчанию для всего Узла формул. Также для определения типов данных можно использовать синтаксис Узла формул, выполняя команды внутри самого блока. Например, *int32 y*; изменит тип данных выходной клеммы с именем *y* на 32-битное целое.

Для обеспечения возможности запуска виртуального прибора на исполнение входные и выходные клеммы важно подключить к соответствующим им элементам, расположенным на блок-диаграмме. Все входные клеммы обязательно должны быть соединены проводами. К выходным клеммам жёсткого требования по подсоединению нет.

**Функции Узла формул и Узла выражений**

Нижеследующий список (Таблица 1) содержит наименования функций Узла формул и Узла выражений с их описанием. Эти функции так же могут быть использованы в элементах, расположенных в разделе специальных формул для виртуальных приборов. Все имена функций должны быть строго записаны строчными символами. В Узлах формул и Узлах выражений могут быть использованы операторы, собранные в разделе «Приоритет операторов в Узлах формул и Узлах выражений», а также число Пи.

Таблица 1 – Перечень функций Узла формул

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Функция** | | **Соответствующая графическая функция LabView** | | **Описание** |
| *abs(x)* | | [Абсолютная](glang.chm::/Absolute_Value.html) величина | | Возвращает абсолютную величину *x*. |
| *acos(x)* | | [Значение,](glang.chm::/Inverse_Cosine.html) обратное косинусу | | Вычисляет значение, обратное косинусу (арккосинус) *x* в радианах. |
| *acosh(x)* | | [Значение,](glang.chm::/Inverse_Hyperbolic_Cosine.html) обратное гиперболическому косинусу | | Вычисляет значение, обратное гиперболическому косинусу (гиперболический арккосинус) *x*. |
| *asin(x)* | | [Значение,](glang.chm::/Inverse_Sine.html) обратное синусу | | Вычисляет значение, обратное синусу (арксинус) *x* в радианах. |
| *asinh(x)* | | [Значение,](glang.chm::/Inverse_Hyperbolic_Sine.html) обратное гиперболическому синусу | | Вычисляет значение, обратное гиперболическому синусу (гиперболический арксинус) *x*. |
| *atan(x)* | | [Значение,](glang.chm::/Inverse_Tangent.html) обратное тангенсу | | Вычисляет значение, обратное тангенсу (арктангенс) *x* в радианах. |
| *atanh(x)* | | [Значение,](glang.chm::/Inverse_Hyperbolic_Tangent.html) обратное гиперболическому тангенсу | | Вычисляет значение, обратное гиперболическому тангенсу (гиперболический арктангенс) *x*. |
| *ceil(x)* | | [Округление](glang.chm::/Round_To_posInfinity.html) до целого в большую сторону | | Округляет *x* до следующего, большего целого значения (самое маленькое целое >= *x*). |
| *ci(x)* | | [Интеграл](gmath.chm::/Cosine_Integral.html) косинуса | | Вычисляет интеграл косинуса *x,* где *x* is любое вещественное число. |
| *cos(x)* | | [Косинус](glang.chm::/Cosine.html) | | Вычисляет косинус *x*, где *x* в радианах. |
| *cosh(x)* | | [Гиперболический](glang.chm::/Hyperbolic_Cosine.html) косинус | | Вычисляет гиперболический косинус *x*. |
| *cot(x)* | | [Котангенс](glang.chm::/Cotangent.html) | | Вычисляет котангенс *x* (*1/tan(x)*), где *x* is в радианах. |
| *csc(x)* | | [Косеканс](glang.chm::/Cosecant.html) | | Вычисляет косеканс *x* (*1/sin(x)*), где *x* в радианах. |
| *exp(x)* | | [Экспонента](glang.chm::/Exponential.html) | | Вычисляет значение *e*, возведённое в степень *x*. |
| *expm1(x)* | | [Экспонента,](glang.chm::/Exponential_Arg_minus_1.html) смещённая в начало координат | | Вычисляет значение, на единицу меньшее, чем *e*, возведённое в степень *x* (*(e^x) – 1*). |
| *floor(x)* | | [Округление](glang.chm::/Round_To_negInfinity.html) до ближайшего целого в меньшую сторону | | Отсекает от *x* вещественную часть до целого значения (самое большое целое <= *x*). |
| *gamma(x)* | | [Гамма-функция](gmath.chm::/Gamma_Function.html) | | *Г(n + 1) = n!* для всех натуральных значений *n*. |
| *getexp(x)* | | [Мантисса](glang.chm::/Mantissa_and_Exponent.html) и экспоненциальная форма | | Возвращает *x* в экспоненциальной форме. |
| *getman(x)* | | Мантисса и экспоненциальная форма | | Возвращает мантиссу *x*. |
| *int(x)* | | [Округление](glang.chm::/Round_To_Nearest.html) до ближайшего целого | | Округляет *x* до ближайшего целого. |
| *intrz(x)* | | [без названия] | | Округляет *x* до ближайшего целого между *x* и нулём. |
| *ln(x)* | | [Натуральный](glang.chm::/Natural_Logarithm.html) логарифм | | Вычисляет натуральный логарифм *x* (по основанию *e*). |
| *lnp1(x)* | [Натуральный](glang.chm::/Natural_Logarithm_Arg_1.html) логарифм, смещённый в начало координат | | Вычисляет натуральный логарифм (*x* + 1). | |
| *log(x)* | [Логарифм](glang.chm::/Logarithm_Base_10.html) по основанию 10 | | Вычисляет логарифм *x* (по основанию 10). | |
| *log2(x)* | [Логарифм](glang.chm::/Logarithm_Base_2.html) по основанию 2 | | Вычисляет логарифм *x* (по основанию 2). | |
| *max(x,y)* | [Максимум](glang.chm::/Max_and_Min.html) и минимум | | Сравнивает *x* с *y* и возвращает большее из значений. | |
| *min(x,y)* | [Максимум](glang.chm::/Max_and_Min.html) и минимум | | Сравнивает *x* с *y* и возвращает меньшее из значений. | |
| *mod(x,y)* | [Целое](glang.chm::/Quotient_and_Remainder.html) от деления и остаток | | Вычисляет остаток от деления *x*/*y*, при остатке, округляемом в меньшую сторону. | |
| *pi(x)* | Возвращает точное значение числа Пи 3.14159... | | pi(*x*)=*x*\*Пи pi(1)=Пи  pi(2.4)=2.4\*Пи | |
| *rand( )* | [Случайное](glang.chm::/Random_Number_0_1.html) число из диапазона (0…1) | | Выдаёт значение с плавающей точкой между 0 и 1 исключительно. | |
| *rem(x,y)* | [Целое](glang.chm::/Quotient_and_Remainder.html) от деления и остаток | | Вычисляет целую часть от деления *x*/*y*, при остатке, округляемом до ближайшего целого. | |
| *sec(x)* | [Секанс](glang.chm::/Secant.html) | | Вычисляет секанс *x*, где *x* в радианах (*1/cos(x)*). | |
| *si(x)* | [Интеграл](gmath.chm::/Sine_Integral.html) синуса | | Вычисляет интеграл синуса *x*, где *x* любое вещественное число. | |
| *sign(x)* | [Сигнум](glang.chm::/Sign.html) | | Возвращает  1, если *x* больше 0,  0, если *x* равен 0,  –1, если *x* меньше 0. | |
| *sin(x)* | [Синус](glang.chm::/Sine.html) | | Вычисляет синус *x*, где *x* в радианах. | |
| *sinc(x)* | [Функция](glang.chm::/Sinc.html) отсчётов | | Вычисляет синус *x*, делённый на *x* (*sin(x)/x*), где *x* в радианах. | |
| *sinh(x)* | [Гиперболический](glang.chm::/Hyperbolic_Sine.html) синус | | Вычисляет гиперболический синус *x*. | |
| *spike(x)* | Строб-функция | | *spike(x)* возвращает:  1, если 0 < *x* < 1,  0 для любого другого *x*. | |
| *sqrt(x)* | Квадратный корень | | Вычисляет квадратный корень *x*. | |
| *square(x)* | Квадратичная функция | | *square(x)* возвращает:  1, если 2*n <= x <=* (2*n* + 1),  0, если 2*n* + 1 <= *x <=* (2*n* + 2),  где *x* – любое вещественное и *n* – любое целое. | |
| *step(x)* | Ступенчатая функция | | *step(x)* возвращает:  0, если *x* < 0,  1, если выполняется любое другое условие. | |
| *tan(x)* | Тангенс | | Вычисляет тангенс *x*, где *x* задаётся в радианах. | |
| *tanh(x)* | Гиперболический тангенс | | Вычисляет гиперболический тангенс от *x*. | |

**Приоритет операторов в Узлах формул и Узлах выражений**

Приоритет выполняемых операторов изменяется в соответствии с представленным в Таблице 2 перечнем от низкого до высокого.

Таблица 2 – Приоритет операторов в Узлах формул

|  |  |
| --- | --- |
| = *op*= | переопределение оператора *op* может быть: «+», «-», «\*», «/», «>>», «<<», «&», «^», «|», «%», или «\*\*». = *op*= не применяется в Узлах выражений. |
| ? : | условное вычисление |
| || | логическое ИЛИ |
| && | логическое И |
| | | битовое ИЛИ |
| ^ | битовое исключающее ИЛИ |
| & | битовое И |
| != и == | неравенство и равенство |
| >, <, >=, и <= | больше, меньше, больше или равно, и меньше или равно |
| >> и << | арифметический сдвиг вправо и арифметический сдвиг влево |
| + и - | сложение и вычитание |
| \* и / | умножение и деление |
| +, -, !, ~, ++, и -- | унарный плюс, унарный минус, логическое НЕ, битовое дополнение, пред инкремент, пост инкремент, пред декремент и пост декремент.  **Внимание:** ++ и -- не применимы для Узлов выражений. |
| \*\* | возведение в степень |

**Внимание:** В версиях *LabView* до 6.0 для возведения в степень используется оператор «^». Начиная с версии 6.1 оператор «^» используется в качестве исключающего ИЛИ (*XOR*). Новым оператором для возведения в степень является «*\*\**», например, *x\*\*y*.

Оператор присвоения (=) является право-ассоциативным (группирует справа налево), как и оператор возведения в степень (\*\*). Все остальные бинарные операторы являются лево-ассоциативными.

Численным значением «ИСТИНЫ» является «1», и для «ЛЖИ» численным значением является «0» на выходе. Логическим значением «0» является «ЛОЖЬ», и любое ненулевое численное значение является «ИСТИНОЙ». Логическое значение условного выражения

*<lexpr> ? <texpr>: <fexpr>*

является *<texpr>* (выражением при истине), если логическое значение *<lexpr>* (логического выражения) является «ИСТИНОЙ» и *<fexpr>* (выражением при лжи) в остальных случаях.

**Синтаксис Узла формул**

Синтаксис Узла формул схож с синтаксисом, используемым в текстуальных языках программирования. Важно помнить о постановке разделителя по завершении оператора (;) как и на языке Си. Синтаксис Узла формул использует форму Бакуса-Наура (*BNF*-нотацию). В квадратные скобки размещены необязательные элементы.

*<stmtList> := <stmt> | <stmtList> <stmt>*

*<stmt> : = <assignment> ; | <varDecl> ; | <compoundStmt>*

*| <condStmt> | <iterativeStmt> | <swithcStmt> | <controlStmt> | ;*

*<varDecl> : = <typeSpec> <identifier> [ [<arrIndexList> ] ]*

*| <typeSpec> <identifier> = <condExpr>*

*| <varDecl> <typeSpec> IDENT [ "["<arrIndexList> "]" ]*

*| <varDecl> <typeSpec> IDENT =<condExpr>*

*<arrIndexList> : = NUM | <arrIndexList> ] [ <arrSize>*

*<typeSpec> : = <floatType> |<intType>*

*<assignment>: = <expr> | <leftHandSide> <asgnOP> <assignment>*

*<expr> : = <expr> <binaryoperator> <expr>*

*| <unaryoperator> <expr>*

*| <expr> ? <expr> : <expr>*

*| ( <expr>)*

*| <identifier>*

*| <const>*

*| <function> ( <arglist> )*

*<binaryoperator>:= + | - | \* | / | ^ | != | = = | > | < | >= | <= | && | || | & | | | ^*

*<unaryoperator>: = + | - | !*

*<arglist>: =<aexpr> [ , <arglist> ]*

*<const>: =pi | <number>*

*<condStmt> : = <ifStmt> | <ifElseStmt>*

*<ifElseStmt> : = <ifStmt> else <stmt>*

*<ifStmt> : = if ( <assignment> ) <stmt>*

*<iterativeStmt> : = <doStmt> | <forStmt> | <whileStmt>*

*<doStmt> : = do <stmt> while ( <expression> ) ;*

*<whileStmt> : = while ( <assignment> ) <stmt>*

*<forStmt> : = for ( [<assignment>] ; [<assignment>]*

*; [<assignment>] ) <stmt>*

*<controlStmt> := break; | continue;*

*<switchStmt> := switch (<expr>) { <caseStmtList> }*

*<caseStmtList> := <caseStmt> | <caseStmtList> <caseStmt>*

*<caseStmt> := case NUM : <stmtList> | default : <stmtList>*

**Изменение типа клемм в Узлах формул**

Выполните следующие шаги для изменения типа клеммы Узла формул с выхода на вход или со входа на выход:

1. Щёлкните правой кнопкой мыши по выходной или входной клемме.

2. В контекстном меню выберите «Изменить на вход» (*Change to Input*) или «Изменить на выход» (*Change to Output*).

**Удаление клемм с Узлов формул**

Выполните следующие шаги для удаления клемм с Узлов формул:

1. Щёлкните правой кнопкой мыши на входной или выходной клемме.

2. Выберите пункт контекстного меню удалить (*Remove*) в контекстном меню.

**Варианты ломаных линий, предлагаемых для построения с использованием Узла формул:**

Варианты ломаных линий собраны в параграфе 6.4.

# 14 Задание «Графическое решение систем линейных алгебраических уравнений в *National Instruments LabView*».

В пакете прикладных программ *National Instruments* *LabView* создать виртуальный прибор для построения графиков функциональных зависимостей, объединённых в систему линейных алгебраических уравнений (СЛАУ) в одних координатных осях. Варианты индивидуальных задач (СЛАУ) собраны в Таблице 1.

Программа в составе виртуального прибора должна включать возможность ввода диапазона построения и шага построения графиков функциональных зависимостей. По построенному комбинированному графику необходимо определить решение (или решения) СЛАУ – графический метод решения СЛАУ.

Проверить результат путём решения СЛАУ вручную – аналитический метод решения СЛАУ. Результат аналитического решения СЛАУ, выполненный вручную, записать в раздел «Тестовые примеры» отчёта к решённому индивидуальному заданию.

Основное требование: не использовать на блок-диаграмме структуру *Formula Node* для решения СЛАУ графическим методом. Использование *Formula Node* в работе будет расцениваться как ошибка.

Дополнительное требование к заданию: решение или решения СЛАУ, если таковое имеется (таковые имеются), маркировать жирной точкой (жирными точками).

На блок-диаграмме все функциональные, то есть значащие пересечения линий связи (узлы) обозначить точками. Для этого в меню перейти к настройкам «*Tools > Options…*», далее в выпадающем списке перейти к настройкам блок-диаграммы (*Block Diagram*) и выставить галочку напротив пункта «*Show dots at wire junctions*».

**Цель работы (одна из возможных формулировок)**: закрепление навыков решения СЛАУ, приобретение навыков объединения данных, поступающих на вход графического элемента *XY Graph*.

**Пример решения задачи:**

Рассмотрим систему линейных алгебраических уравнений:



Она представляет собой два объединённых уравнения прямой вида **y = k \* x + b**. Перепишем систему таким образом, чтобы этот факт был представлен в явном виде:



Если построить в одних координатных осях эти две прямые, то можно обнаружить точку их пересечения (Рисунок 1). Эта точка и есть решение СЛАУ.

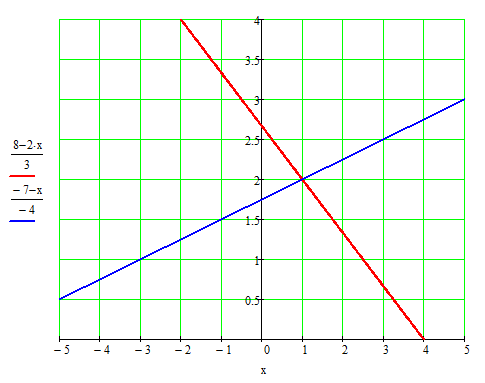


Рисунок 1 – Графический метод решения СЛАУ

**Варианты индивидуального задания по СЛАУ:**

Таблица 1 – Варианты индивидуального задания

|  |  |
| --- | --- |
| 1. | 16. |
| 2. | 17. |
| 3. | 18. |
| 4. | 19. |
| 5. | 20. |
| 6. | 21. |
| 7. | 22. |
| 8. | 23. |
| 9. | 24. |
| 10. | 25. |
| 11. | 26. |
| 12. | 27. |
| 13. | 28. |
| 14. | 29. |
| 15. | 30. |

# 15 Задание «Моделирование построения окружности в *National Instruments LabView*».

В пакете прикладных программ *National Instruments LabView* создать виртуальный прибор, выводящий на интерфейсный элемент *XY Graph* окружность, радиус которой задаётся в реальном времени (либо квази-непрерывно, либо дискретно в зависимости от варианта).

Нечётные варианты реализуют изменение радиуса посредством использования любого непрерывного интерфейсного элемента ввода (*Numeric*, *Deal,* также допускается использование и термометра). В данной постановке изменение радиуса реализуется квази-непрерывно.

Чётные варианты реализуют изменение радиуса окружности посредством использования одного или нескольких дискретных, интерфейсных логических элементов ввода:

**- способ а (однокнопочный), реализуемый при делении номера варианта на четыре без остатка:** тумблер/кнопка типа *Boolean* включён – одно значение радиуса окружности, тумблер/кнопка выключен – другое значение радиуса окружности. Конкретное значение радиуса окружности для конкретного логического состояния задаётся на блок-диаграмме целочисленной константой;

**- способ б (многокнопочный), реализуемый при делении варианта на четыре с остатком:** все тумблеры/кнопки типа *Boolean* выключены – окружность не отображается (нулевой радиус), все тумблеры/кнопки включены – максимальное значение радиуса окружности, тумблеры/кнопки включены и выключены в различных комбинациях – некоторые промежуточные значения между нулём-радиусом и максимальным радиусом. Конкретное значение радиуса окружности для конкретного логического состояния задаётся на блок-диаграмме целочисленной константой.

Основные требования к работе:

- для формирования окружности **в общей части работы** в цикле использовать только тригонометрические функции синуса и косинуса;

**- к защите не принимаются работы, в которых окружность строится при использовании структуры *Formula Node* на блок-диаграмме;**

**-** по итогам выполнения общей части работы сдаётся файл *\*.vi.*

**Цель работы (одна из возможных формулировок)**: освоение навыков использования структуры *For* и умножения массива данных на константу в пакете прикладных программ *National Instruments LabView*.

**Варианты индивидуального задания, связанного с построением окружности:**

Таблица 1 – Варианты индивидуального задания

|  |  |
| --- | --- |
| **Вар.** | **Задание** |
| **1** | Использовать датчик случайных чисел для моделирования броска дротика в мишень. Зажигать световой индикатор, если дротик попал на одну из координатных осей, расположенных внутри окружности. Использовать «Синус и Косинус». |
| **2** | Отобразить на *XY Graph* дополнительные пять окружностей, радиусы которых меньше радиуса основной окружности на 50%, 60%, 70%, 80%, 90%. Использовать уравнение окружности вида: . Ввести на графический пользовательский интерфейс виртуального прибора возможность покоординатного смещения центра окружности относительно начала координат. |
| **3** | Использовать общее уравнение окружности вида: . Параметры данного уравнения задаются посредством использования датчиков случайных чисел. Зажигать световой индикатор, если центр окружности оказался расположен в *delta*-окрестности начала координат. Предусмотреть дополнительно отладочный интерфейсный элемент для принудительного размещения центра окружности в начале координат. |
| **4** | Использовать координаты точки для моделирования броска дротика в мишень. Зажигать световой индикатор, если дротик попал внутрь окружности. Использовать уравнение окружности . |
| **5** | Использовать датчик случайных чисел для моделирования броска дротика в мишень. Зажигать световой индикатор, если дротик попал на окружность. Предусмотреть в качестве отладочного варианта возможность размещения дротика на окружности принудительно и вручную, без использования датчика случайных чисел. Использовать «Синус и Косинус». |
| **6** | Вместе с изменением радиуса окружности выполнять пропорциональное масштабирование параболы (с минимумом в начале координат), не выходящей за пределы окружности. Использовать уравнение окружности вида . Ввести для пользователя виртуального прибора возможность покоординатного смещения параболы относительно начала координат (при смещении параболы от начала координат выход за пределы окружности можно не контролировать). |
| **7** | Использовать общее уравнение окружности вида , генерацию всех параметров этого уравнения реализовать посредством использования датчиков случайных чисел. Предусмотреть вывод побуждающего сообщения для пользователя виртуального прибора в случаях несовместимости значений параметров. Зажигать один из четырёх световых индикаторов в зависимости от координатной четверти, в которую явно попал центр окружности. |
| **8** | Предусмотреть интерфейсные элементы управления, реализующие смещение центра окружности как по горизонтали, так и по вертикали. Для построения окружности использовать «Синус и Косинус». |
| **9** | Отобразить на *XY Graph* треугольник, составленный из прямых линий, за пределами которого изменение радиуса окружности посредством интерфейсного элемента управления индифферентно (к каким-либо изменениям не чувствительно). Параметры треугольника задаются пользователем через графический пользовательский интерфейс. Для построения окружности использовать уравнение вида . |
| **10** | Помимо интерфейсных элементов управления, изменяющих радиус окружности, предусмотреть и другие интерфейсные элементы управления, реализующие смещение центра окружности как по горизонтали, так и по вертикали. Использовать уравнение окружности вида . |
| **11** | Использовать координаты точки для моделирования броска дротика в мишень. Зажигать световой индикатор, если дротик попал на окружность (на границу окружности). Предусмотреть возможность броска нескольких дротиков в указанные точки. Использовать уравнение окружности вида . |
| **12** | Отобразить на *XY Graph* все возможные окружности с шагом 10%, начинающиеся в начале координат и расположенные внутри номинала. Использовать «Синус и Косинус». |
| **13** | Использовать датчик случайных чисел для моделирования броска дротика в мишень. Зажигать один из четырёх световых индикаторов в зависимости от координатной четверти, в которую попал дротик. Использовать «Синус и Косинус». |
| **14** | Отобразить на *XY Graph* дополнительные пять окружностей, радиусы которых меньше радиуса основной окружности на 50%, 60%, 70%, 80%, 90%. Использовать уравнение окружности . |
| **15** | Вместе с изменением радиуса окружности выполнять масштабирование инверсной параболы, максимум которой при нулевом значении радиуса располагается в центре окружности. Предусмотреть возможность сдвига максимума параболы в указанную точку. Использовать уравнение окружности . |
| **16** | Использовать общее уравнение окружности вида , предоставить возможность ввода пользователем всех параметров. Зажигать один из четырёх световых индикаторов в зависимости от координатной четверти, в которую явно попал центр окружности (нахождение центра окружности в начале координат – выключение всех световых индикаторов). |
| **17** | Вместе с изменением радиуса окружности выполнять пропорциональное изменение параболы, минимум которой при нулевом значении радиуса располагается в центре окружности. Парабола может выходить за пределы окружности. Использовать «Синус и Косинус». |
| **18** | Отобразить на *XY Graph* квадрат, составленный из прямых линий, за пределами которого изменение радиуса окружности посредством интерфейсного элемента управления индифферентно (к изменениям не чувствительно). Для построения окружности использовать «Синус и Косинус». Параметры прямых для построения квадрата задаются пользователем через графический пользовательский интерфейс. |
| **19** | Вместе с изменением радиуса окружности выполнять поворот против часовой стрелки прямой фиксированного диапазона, задаваемого пользователем виртуального прибора и проходящей через центр окружности. Использовать уравнение окружности вида . |
| **20** | Использовать датчик случайных чисел для моделирования броска дротика в мишень. Зажигать световой индикатор в том случае, если дротик попал за пределы окружности. Для построения окружности использовать «Синус и Косинус». |
| **21** | Использовать общее уравнение окружности вида , предоставить пользователю виртуального прибора возможность ввода всех параметров этого уравнения. Зажигать световой индикатор в случае, если окружность оказалась в начале координат. |
| **22** | Отобразить на *XY Graph* прямую линию (горизонтальную, вертикальную, наклонную – реакция должна быть адекватной в ответ на любой из этих вариантов), за пределами которой изменение радиуса окружности посредством элемента управления индифферентно (не чувствительно к изменениям). Использовать «Синус и Косинус». |
| **23** | Использовать датчик случайных чисел для моделирования броска дротика в мишень. Зажигать световой индикатор, если дротик попал внутрь окружности. Использовать уравнение окружности . |
| **24** | Отображать на *XY Graph* дополнительные окружности радиусов, отличающихся на единичный шаг от радиусов предшествующих им окружностей. Количество отображаемых окружностей задаётся значением, указанным в соответствующем элементе управления. Использовать «Синус и Косинус». |
| **25** | Вместе с изменением радиуса окружности выполнять поворот по часовой стрелке прямой линии, проходящей через центр окружности. Использовать уравнение окружности . |
| **26** | Отобразить на *XY Graph* три дополнительные окружности, радиусы которых меньше радиуса основной окружности на 40%, 60%, 80%. Использовать «Синус и Косинус». |

# 16 Задание «Разработка конвертеров значений по системам счисления в *National Instruments LabView*».

В пакете прикладных программ *National Instruments LabView* создать виртуальный прибор, предназначенный для перевода значений из одной, заданной системы счисления в другую, указанную систему счисления.

На блок-диаграмме все функциональные, то есть значащие пересечения линий связи (узлы) обозначить точками. Для этого в меню перейти к настройкам «*Tools > Options…*», далее в выпадающем списке перейти к настройкам блок-диаграммы (*Block Diagram*) и выставить галочку напротив пункта «*Show dots at wire junctions*».

**Цель работы (одна из возможных формулировок)**: освоение навыков работы по составлению строк, блоков-функций целочисленного деления, сдвиговых регистров и расширенной структуры *CASE* с более, чем двумя значениями.

**Рекомендации к работе:**

***Алгоритм перевода из десятичной системы счисления в двоичную систему.***

1. Выполнить деление исходного числа на 2. Если результат деления больше или равен 2, продолжать делить его на 2 до тех пор, пока результат деления не станет равен 1.

2. Выписать результат последнего деления и все остатки от деления в обратном порядке в одну строку.

***Алгоритм перевода из двоичной системы счисления в десятичную систему.***

1. Пронумеровать разряды двоичного числа справа налево, начиная с нуля.

2. Умножить каждый ненулевой разряд на 2 в степени его номера и сложить результаты.

***Алгоритм перевода из шестнадцатеричной системы счисления в двоичную систему.***

1. Задать таблицу соответствия четырёхразрядного формата в двоичной системе по отношению к каждому разряду в шестнадцатеричной системе. Например, F это 1111, A – 1010.

2. Поразрядно отщеплять от шестнадцатеричного значения элементы справа налево и записывать четырёх разрядный компонент так же справа налево. Например: 9AF это 1001 1010 1111.

***Алгоритм перевода из двоичной системы счисления в шестнадцатеричную систему.***

1. Задать таблицу соответствия четырёхразрядного формата в двоичной системе по отношению к каждому разряду в шестнадцатеричной системе. Например, 0001 это 1, 1011 – B.

2. Проверить, кратна ли заданная двоичная последовательность 4-м. Если нет, то дополнить её слева недостающим количеством нулей.

3. Идти по последовательности справа налево и отделять четвёрку разрядов. Каждую четвёрку сравнивать с таблицей соответствия и выдавать шестнадцатеричный разряд.

4. Записывать каждый новый разряд так же справа налево.

**Варианты индивидуального задания**

Таблица 2 – Варианты индивидуального задания по системам счисления

|  |  |
| --- | --- |
| **Вар.** | **Задание** |
| 1 | Разработать универсальный конвертор из двоичной системы счисления в 3–10-чные системы счисления (с управляющим числовым элементом). Входной тип данных – массив, выходные типы данных – целые. |
| 2 | Разработать универсальный конвертор из шестнадцатеричной системы счисления в 2–10-чные системы счисления (с выбором основания системы счисления через меню). Входной тип данных – массив, выходные типы данных – строки. |
| 3 | Разработать конвертор из системы счисления, в которой задействован весь кириллический алфавит (А-Я) в десятичную систему счисления. Входной тип данных – строка, выходной тип данных – целое. |
| 4 | Разработать универсальный конвертор из двоичной системы счисления в 11–20-чные системы счисления (с выбором основания системы счисления через меню). Входной тип данных – строка, выходные типы данных – строки. |
| 5 | Разработать конвертор из системы счисления, в которой задействован весь латинский алфавит (A-Z), в двоичную систему счисления. Входной тип данных – массив, выходной тип данных – массив. |
| 6 | Разработать конвертор из двоичной системы счисления в десятичную и шестнадцатеричную системы счисления. Входной тип данных – массив, выходные типы данных – массивы. |
| 7 | Разработать универсальный конвертор из восьмеричной системы счисления в 2–7-чные системы счисления (с выбором основания системы счисления через меню). Входной тип данных – строка, выходные типы данных – массивы. |
| 8 | Разработать универсальный конвертор из двоичной системы счисления в 11–20-чные системы счисления (с выбором основания через управляющий числовой элемент). Входной тип данных – массив, выходные типы данных – массивы. |
| 9 | Разработать универсальный конвертор из двоичной системы счисления в 3–10-чные системы счисления (с выбором основания системы счисления через меню). Входной тип данных – массив, выходные типы данных – целые. |
| 10 | Разработать конвертор из системы счисления, в которой задействован весь кириллический алфавит (А-Я) в двоичную систему счисления. Входной тип данных – массив, выходной тип данных – массив. |
| 11 | Разработать конвертор из шестнадцатеричной системы счисления в двоичную и десятичную системы счисления. Входной тип данных – массив, выходные типы данных – строки. |
| 12 | Разработать конвертор из системы счисления, в которой задействован весь латинский алфавит (A-Z), в десятичную систему счисления. Входной тип данных – строка, выходной тип данных – целое. |
| 13 | Разработать универсальный конвертор из двоичной системы счисления в 3–10-чные системы счисления (с управляющим числовым элементом). Входной тип данных – строка, выходные типы данных – строки. |
| 14 | Разработать конвертор из восьмеричной системы счисления в десятичную и шестнадцатеричную системы счисления. Входной тип данных – массив, выходные типы данных: для шестнадцатеричной – строки, для десятичной – целое. |
| 15 | Разработать конвертор из двоичной системы счисления в десятичную и шестнадцатеричную системы счисления. Входной тип данных – массив, выходные типы данных – строки. |
| 16 | Разработать конвертор из десятичной в системы счисления, в такую систему счисления, в которой задействован весь кириллический алфавит (А-Я). Входной тип данных – целое, выходной тип данных – строка. |
| 17 | Разработать конвертор из шестнадцатеричной системы счисления в двоичную и десятичную системы счисления. Входной тип данных – строка, выходные типы данных: для двоичной – строка, для десятичной – целое. |
| 18 | Разработать конвертор из двоичной системы счисления в десятичную и шестнадцатеричную системы счисления. Входной тип данных – строка, выходные типы данных – массивы. |
| 19 | Разработать универсальный конвертор из шестнадцатеричной системы счисления в 2–10-чные системы счисления (с выбором основания системы счисления через меню). Входной тип данных – строка, выходные типы данных – массивы. |
| 20 | Разработать универсальный конвертор из восьмеричной системы счисления в 2–7-чные системы счисления (с выбором основания системы счисления через меню). Входной тип данных – массив, выходные типы данных – строки. |
| 21 | Разработать универсальный конвертор из двоичной системы счисления в 11–20-чные системы счисления (с выбором основания системы счисления через меню). Входной тип данных – массив, выходные типы данных – массивы. |
| 22 | Разработать универсальный конвертор из восьмеричной системы счисления в 2–7-чные системы счисления (с управляющим числовым элементом). Входной тип данных – целое, выходные типы данных – целые. |
| 23 | Разработать универсальный конвертор из шестнадцатеричной системы счисления в 2–10-чные системы счисления (с управляющим числовым элементом). Входной тип данных – массив, выходные типы данных – целые. |
| 24 | Разработать конвертор из восьмеричной системы счисления в десятичную и шестнадцатеричную системы счисления. Входной тип данных – строка, выходные типы данных – строки. |
| 25 | Разработать конвертор из десятичной в системы счисления, в такую систему счисления, в которой задействован весь латинский алфавит (A-Z). Входной тип данных – целое, выходной тип данных – массив. |
| 26 | Разработать универсальный конвертор из двоичной системы счисления в 11–20-чные системы счисления (с выбором основания через управляющий числовой элемент). Входной тип данных – строка, выходные типы данных – строки. |
| 27 | Разработать универсальный конвертор из двоичной системы счисления в 3–10-чные системы счисления (с выбором основания системы счисления через меню). Входной тип данных – строка, выходные типы данных – строки. |
| 28 | Разработать конвертор из восьмеричной системы счисления в двоичную и шестнадцатеричную системы счисления. Входной тип данных – целое, выходные типы данных – строки. |
| 29 | Разработать конвертор из шестнадцатеричной системы счисления в двоичную и десятичную системы счисления. Входной тип данных – массив, выходные типы данных: для десятичной – целое, для двоичной – массив. |
| 30 | Разработать конвертор из двоичной системы счисления в десятичную и шестнадцатеричную системы счисления. Входной тип данных – строка, выходные типы данных – строки. |

# 17 Задание «…».

В пакете прикладных программ *National Instruments LabView* разработать подпрограмму (виртуальный прибор) для расчёта/исследования функции факториала. Определить границы корректного вычисления факториала и ограничить диапазон пользовательского ввода входного значения. Использовать созданную функцию факториала для расчёта разложенной в ряд Маклорена функции (заданной по варианту). Выполнить поэтапное разложение функции в ряд Маклорена. Сравнить расчётное значение разложенной в ряд Маклорена функции с истинным значением этой же функции. Для случая расчёта с заданным количеством элементов ряда оценить точность выполненных вычислений.

На блок-диаграмме все функциональные, то есть значащие пересечения линий связи (узлы) обозначить точками. Для этого в меню перейти к настройкам «*Tools > Options…*», далее в выпадающем списке перейти к настройкам блок-диаграммы (*Block Diagram*) и выставить галочку напротив пункта «*Show dots at wire junctions*».

**Цель работы (одна из возможных формулировок)**: освоение навыков разработки пользовательских процедур. Закрепление раздела «Ряды» из курса высшей математики. Закрепление раздела «Функции и процедуры» из курса алгоритмизации и технологий программирования.

**Индивидуальная часть:**

Разработать пользовательскую подпрограмму (виртуальный прибор), реализующую вычисление с заданной точностью указанных по варианту функций. В случае образования функции факториала при разложении указанной функции в ряд использовать подпрограмму, составленную в общей части данного задания. По итогам создания и отладки пользовательской подпрограммы создать виртуальный прибор, блок-диаграмма которого содержала бы не более чем:

- элементы ввода данных,

- элементы вывода данных,

- пользовательскую подпрограмму.

**Пример создания и использования ВП-процедур:**

1. Создаётся интерфейс ВП-процедуры, содержащий целочисленный контроллер и целочисленный индикатор.

2. Элементы связываются некоторой функциональной зависимостью, например, входная информация возводится в куб.

3. В режиме блок-диаграммы осуществляется переход к редактированию пиктограммы (нажатие правой кнопкой мыши на пиктограмме).

4. В редакторе пиктограммы исходное изображение очищается.

5. На графическую область наносится пользовательское изображение или растрированный текст.

6. По итогам подтверждения созданной пиктограммы нажатием на кнопку «ОК» её изображение применяется к проекту ВП.

7. В режиме редактирования интерфейса производится переход к настройке внешних связей (нажатие правой кнопкой мыши в области пиктограммы).

8. Элементом соединительный провод выбирается левая секция (входные данные) и после её маркировки чёрным цветом тем же соединительным проводом отмечается интерфейсный элемент, ответственный за входные данные. В данном примере – контроллер *X*.

9. Элементом соединительный провод выбирается правая секция (выходные данные) и после её маркировки чёрным цветом тем же соединительным проводом отмечается интерфейсный элемент, ответственный за выходные данные. В данном примере – индикатор *X^3*.

10. Возврат к привычному режиму реализуется через контекстное меню пиктограммы.

11. Созданный ВП сохраняется под определённым именем в той же директории, куда в дальнейшем планируется сохранение основного виртуального прибора, использующего созданный ВП в качестве процедуры.

12. Создаётся новый ВП, который будет основан на ранее созданном и сохранённом. В нём настраивается аналогичный пользовательский интерфейс.

13. В режиме блок-диаграммы реализуется выбор ВП из файла (*Select a VI...*).

14. Указывается путь к ВП.

15. Пиктограмма ВП становится доступной для размещения на блок-диаграмме.

16. Контроллер и индикатор связываются через ВП-процедуру соединительным проводом.

17. На интерфейс вводятся исходные данные перед запуском ВП.

18. ВП запускается в непрерывном режиме и демонстрирует корректный результат возведения 9 в куб.

**Теоретическое напоминание о рядах Тейлора и рядах Маклорена:**



Рисунок 19 – Фрагмент теории из справочника

**Пример разложения функции в ряд Маклорена:**



Рисунок 20 – Поиск пяти первых элементов ряда Маклорена для функции экспоненты

**Варианты индивидуального задания:**

1. Натуральный логарифм числа **(2 \* x + 1)**, вычисленный с учётом *N* членов ряда (циклическая структура *For*).

2. Натуральный логарифм числа **(x + 1)x**, вычисленный с заданной точностью (циклическая структура *While*).

3. Натуральный логарифм числа **(x + 1)-1**, вычисленный с учётом *N* членов ряда (циклическая структура *For*).

4. Натуральный логарифм числа **(x + 2)**, вычисленный с заданной точностью (циклическая структура *While*).

5. Косинус угла, заданного в градусах, вычисленный с учётом *N* членов ряда (циклическая структура *For*).

6. Натуральный логарифм числа **(x + 1)**, вычисленный с заданной точностью (циклическая структура *While*).

7. Арккосинус, возвращающий угол в градусах, вычисленный с учётом *N* членов ряда (циклическая структура *For*).

8. Гиперболический синус, вычисленный с заданной точностью (циклическая структура *While*).

9. Гиперболический тангенс, вычисленный с учётом *N* членов ряда (циклическая структура *For*).

10. Арктангенс, возвращающий угол в градусах, вычисленный с заданной точностью (циклическая структура *While*).

11. Натуральный логарифм числа **(x + 1)**, вычисленный с учётом *N* членов ряда (циклическая структура *For*).

12. Натуральный логарифм числа **(2 \* x + 1)**, вычисленный с заданной точностью (циклическая структура *While*).

13. Секанс угла, заданного в градусах, вычисленный с учётом *N* членов ряда (циклическая структура *For*).

14. Секанс угла, заданного в градусах, вычисленный с заданной точностью (циклическая структура *While*).

15. Арктангенс, возвращающий угол в градусах, вычисленный с учётом *N* членов ряда (циклическая структура *For*).

16. Натуральный логарифм числа **(x + 1)-1**, вычисленный с заданной точностью (циклическая структура *While*).

17. Натуральный логарифм числа **(x + 1)x**, вычисленный с учётом *N* членов ряда (циклическая структура *For*).

18. Гиперболический косинус, вычисленный с заданной точностью (циклическая структура *While*).

19. Гиперболический синус, вычисленный с учётом *N* членов ряда (циклическая структура *For*).

20. Косеканс угла, заданного в градусах, вычисленный с заданной точностью (циклическая структура *While*).

21. Натуральный логарифм числа **(x + 2)**, вычисленный с учётом *N* членов ряда (циклическая структура *For*).

22. Гиперболический тангенс, вычисленный с заданной точностью (циклическая структура *While*).

23. Косеканс угла, заданного в градусах, вычисленный с учётом *N* членов ряда (циклическая структура *For*).

24. Арксинус, возвращающий угол в градусах, вычисленный с заданной точностью (циклическая структура *While*).

25. Экспонента в степени *x*, вычисленная с учётом *N* членов ряда (циклическая структура *For*).

26. Экспонента в степени x, вычисленная с заданной точностью (циклическая структура *While*).

27. Гиперболический косинус, вычисленный с учётом *N* членов ряда (циклическая структура *For*).

28. Арккосинус, возвращающий угол в градусах, вычисленный с заданной точностью (циклическая структура *While*).

29. Арксинус, возвращающий угол в градусах, вычисленный с учётом *N* членов ряда (циклическая структура *For*).

30. Косинус угла, заданного в градусах, вычисленный с заданной точностью (циклическая структура *While*).

# 18 Задание «…».

В пакете прикладных программ *National Instruments LabView* создать виртуальный прибор для демонстрации возможностей среды при работе с комплексными числами. Расчётные значения комплексных чисел, заданные по вариантам, зафиксировать в графическом пользовательском интерфейсе как настройки, заданные по умолчанию (файл виртуального прибора должен открываться с уже заложенными значениями). Допускается сдача работы как в виде одного виртуального прибора, объединяющего в себе три нижеследующих раздела, так и сдача трёх виртуальных приборов, где каждый раздел оформлен отдельно. **Обратить внимание на то, что индивидуальная часть работы подразумевает создание ВП-процедуры**.

Один из возможных примеров оформления графического пользовательского интерфейса (**внимание:** не реализовывать его один к одному, поскольку в нём, как минимум, отсутствует разделение элементов по разделам):

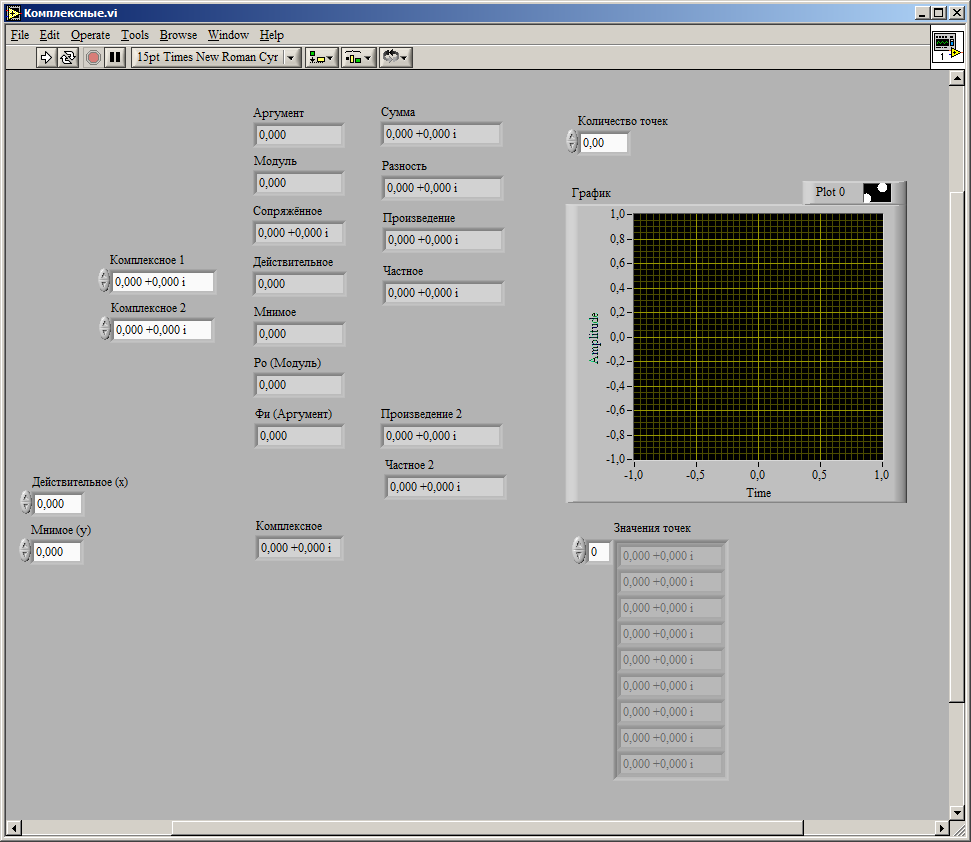


Рисунок 1 – Пример компоновки графического пользовательского интерфейса ВП

**I) Вывести на графический пользовательский интерфейс виртуального прибора посредством стандартных функций обработки комплексных чисел:**

а) комплексно-сопряжённое число к комплексному числу, поданному на вход;

б) комплексное число, полученное через свои параметры: модуль и аргумент, поданные на вход;

в) модуль и аргумент комплексного числа, поданного на вход;

г) комплексное число, полученное через свои действительную и мнимую части, поданные на вход;

д) действительную и мнимую части комплексного числа, поданного на вход.

**Памятка программисту по разделу I:**

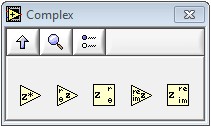
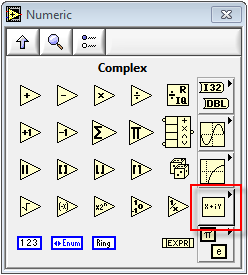


Рисунок 2 – Расположение стандартных функций для работы с комплексными числами

**II) Вывести на графический пользовательский интерфейс виртуального прибора посредством арифметических операций и стандартных функций обработки целых и вещественных чисел:**

а) комплексно-сопряжённое число к комплексному числу, поданному на вход;

б) модуль комплексного числа, поданного на вход, полученный через мнимую и действительную части;

в) аргумент комплексного числа, поданного на вход через мнимую и действительную части;

г) сумму двух комплексных чисел, поданных на вход;

д) разность двух комплексных чисел, поданных на вход;

е) произведение двух комплексных чисел, поданных на вход;

ж) частное двух комплексных чисел, поданных на вход;

з) сумму двух комплексных чисел, поданных на вход, посредством комбинирования их действительных и мнимых частей;

и) разность двух комплексных чисел, поданных на вход, посредством комбинирования их действительных и мнимых частей;

к) произведение двух комплексных чисел, поданных на вход, посредством комбинирования их действительных и мнимых частей;

л) частное двух комплексных чисел, поданных на вход, посредством комбинирования их действительных и мнимых частей.

**Памятка программисту по разделу II:**

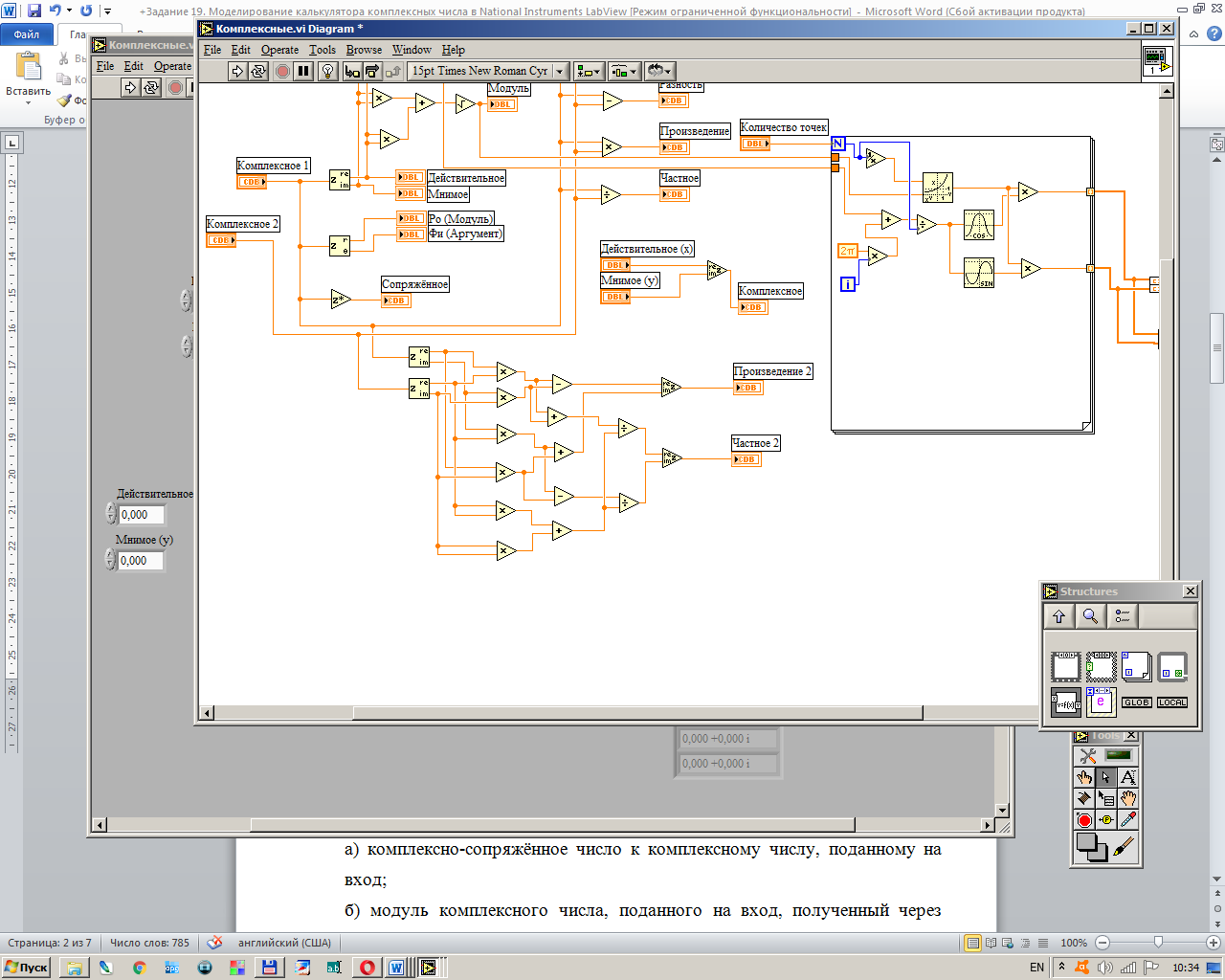


Рисунок 3 – Работа с действительными и мнимыми частями комплексных чисел по-отдельности

**III) Вывести на графический пользовательский интерфейс виртуального прибора результат расчёта корня n-степени из комплексного числа:**

а) в виде точек на комплексной плоскости;

б) в виде массива из n-штук комплексных чисел.

**Памятка программисту по разделу III:**

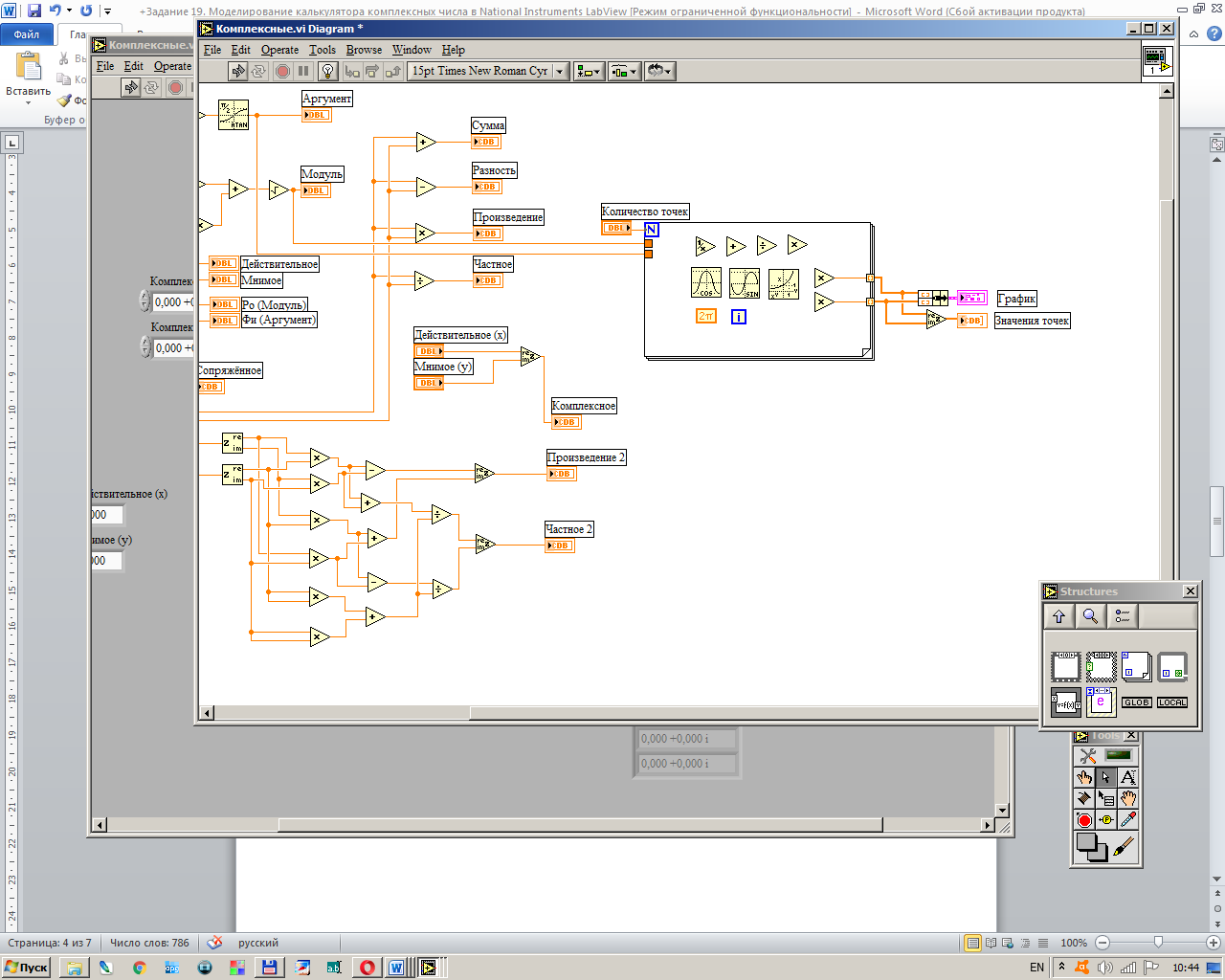


Рисунок 4 – Перечень необходимых и достаточных элементов для реализации формулы Муавра

На блок-диаграмме все функциональные, то есть значащие пересечения линий связи (узлы) обозначить точками. Для этого в меню перейти к настройкам «*Tools > Options…*», далее в выпадающем списке перейти к настройкам блок-диаграммы (*Block Diagram*) и выставить галочку напротив пункта «*Show dots at wire junctions*».

**Цель работы (одна из возможных формулировок)**: освоение навыков работы с комплексными числами. Закрепление раздела «Теория функции комплексной переменной» из высшей математики.

**Памятка программисту по настройке графического пользовательского интерфейса:**

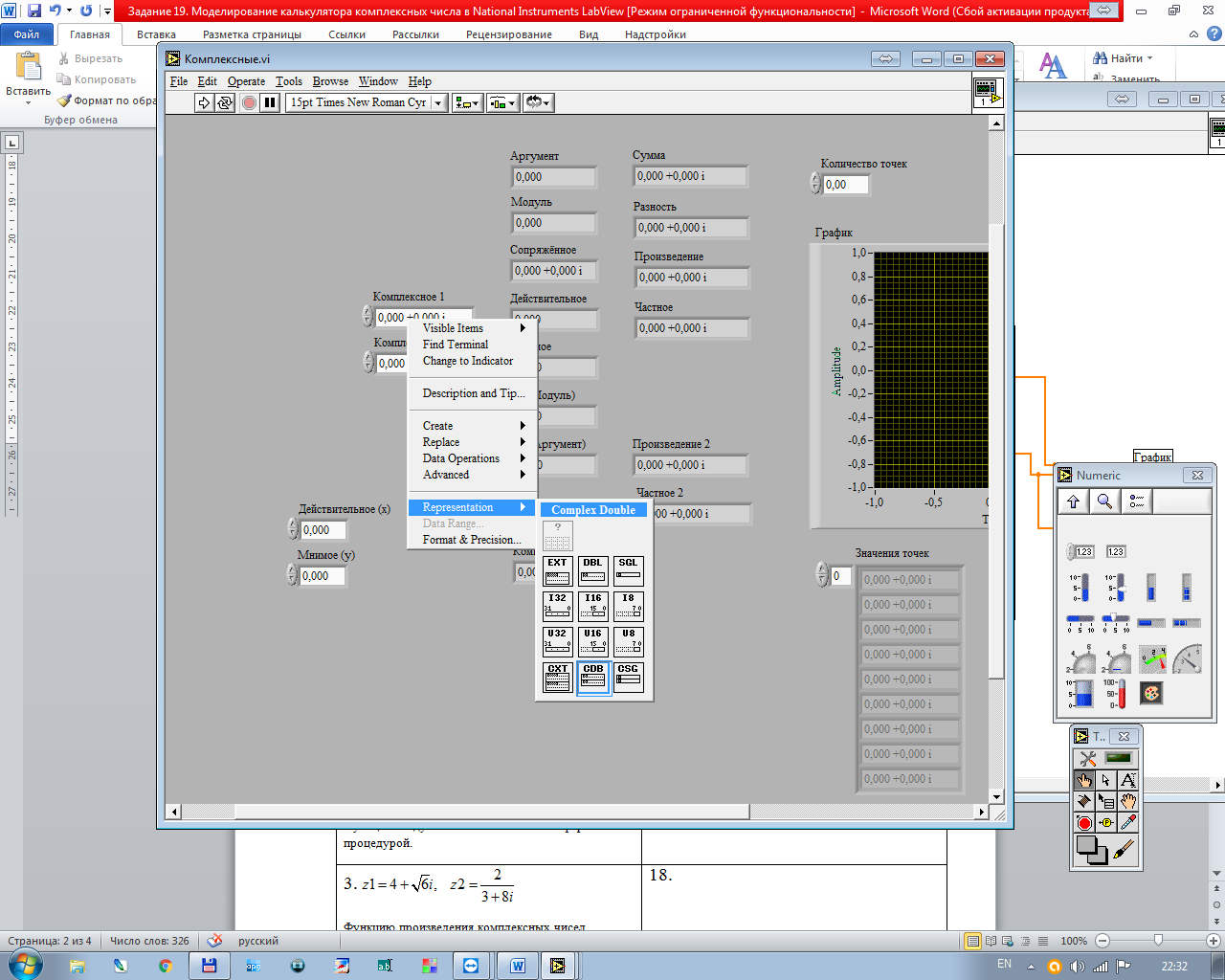


Рисунок 5 – Расположение настройки, отвечающей за выбор в качестве типа данных комплексного числа двойной точности

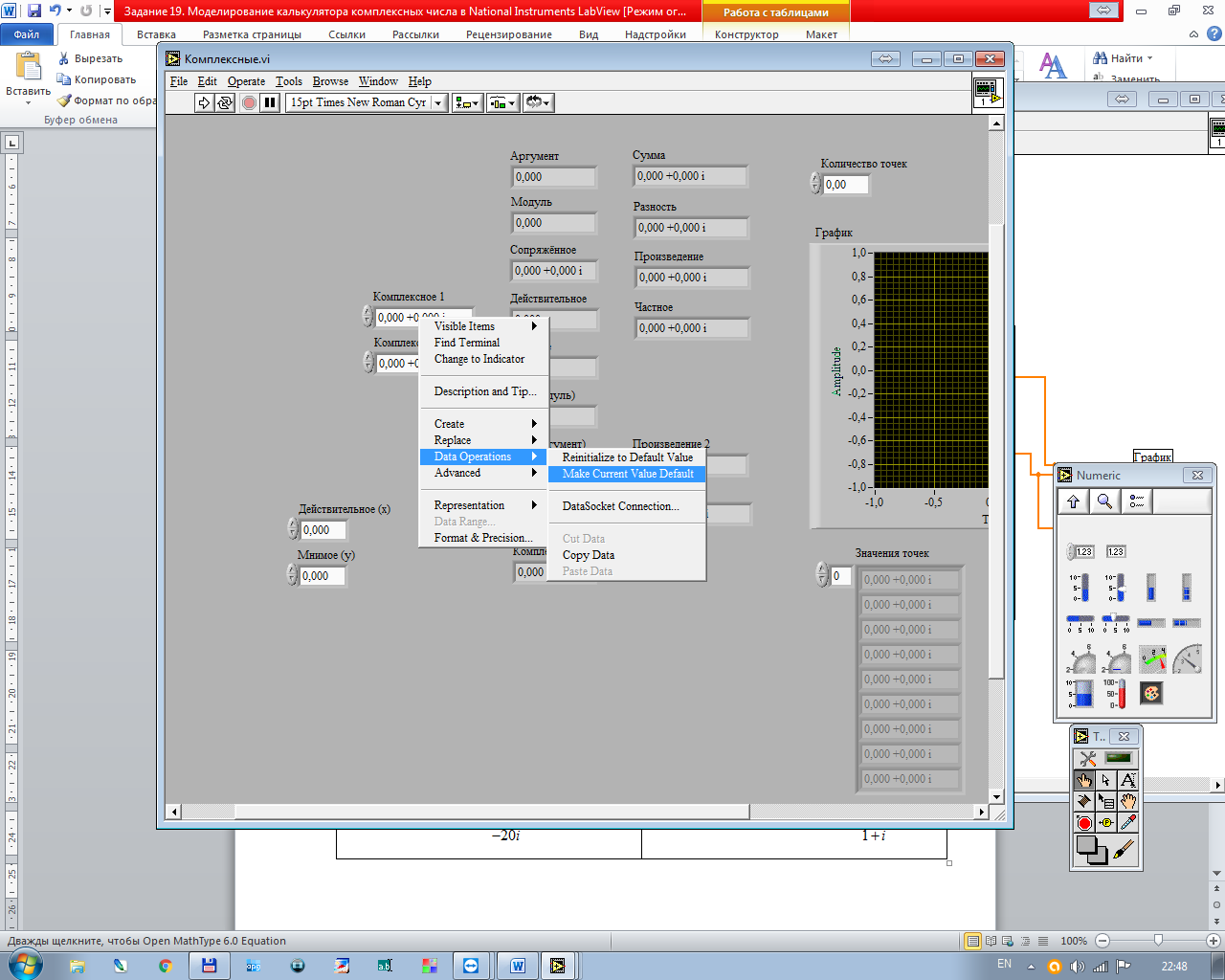


Рисунок 6 – Расположение настройки, отвечающей за установку значения, заданного по умолчанию.

**Исходные комплексные числа по вариантам:**

Таблица 1 – Исходные данные к индивидуальной части работы

|  |  |
| --- | --- |
| 1.  Функцию аргумента комплексного числа оформить процедурой. | 16.  Функцию аргумента, возведённого в квадрат, от комплексного числа оформить процедурой. |
| 2.  Функцию модуля комплексного числа оформить процедурой. | 17.  Функцию модуля, возведённого в квадрат, от комплексного числа оформить процедурой. |
| 3.  Функцию произведения комплексных чисел оформить процедурой. | 18.  Функцию возведения комплексного числа в натуральную степень оформить процедурой. |
| 4.  Функцию частного комплексных чисел оформить процедурой. | 19.  Функции модуля и произведения комплексного числа с самим собой оформить процедурой. |
| 5.  Оформить всю арифметику над парой комплексных чисел в процедуру. | 20.  Ограничиться вычислением корня 3-й степени из комплексного числа. Направить отрезки из начала координат в полученные точки. |
| 6.  Функцию разности частных первого комплексного числа на второе и второго на первое оформить процедурой. | 21.  Функцию суммы частных первого комплексного числа на второе и второго на первое оформить процедурой. |
| 7.  Функцию поиска корня n-й степени из комплексного числа оформить процедурой. | 22.  Функцию аргумента комплексного числа, возведённого в квадрат, оформить процедурой. |
| 8.  Корни трёх различных степеней из одного и того же комплексного числа вывести в одну и ту же координатную плоскость. | 23.  Оформить всю арифметику над парой комплексных чисел в процедуру. |
| 9.  Корни двух различных степеней из одного и того же комплексного числа вывести в одну и ту же координатную плоскость. | 24.  Функции произведения и частного комплексных чисел оформить процедурой. |
| 10.  Функции модуля и аргумента комплексного числа оформить процедурой. | 25.  Корни различных степеней из двух разных комплексных чисел вывести в одну и ту же координатную плоскость. |
| 11.  Корень одной и той же степени из двух разных комплексных чисел вывести в одну и ту же координатную плоскость. | 26.  Функцию поиска корня n-й степени из комплексного числа оформить процедурой. |
| 12.  Функцию модуля комплексного числа, возведённого в квадрат, оформить процедурой. | 27.  Функцию частного комплексных чисел оформить процедурой. |
| 13.  Ограничиться вычислением корня 5-й степени из комплексного числа. Направить отрезки из начала координат в полученные точки. | 28.  Функцию произведения комплексных чисел оформить процедурой. |
| 14.  Функции аргумента и произведения комплексного числа с самим собой оформить процедурой. | 29.  Функцию модуля комплексного числа оформить процедурой. |
| 15.  Добавить процедуру, изображающую окружность, на которую проецируются точки корня n-й степени из комплексного числа. | 30.  Функцию аргумента комплексного числа оформить процедурой. |

# 19 Задание «…».

В пакете прикладных программ *National Instruments LabView* создать виртуальный прибор для решения системы линейных алгебраических уравнений (СЛАУ) матричным методом.

**Общая часть задания:**

1. Выполнить подключение и настройку стандартной функции *Solve Linear Equations* для решения СЛАУ.

2.1. Учащиеся с чётными номерами вариантов реализуют решение СЛАУ методом Крамера при использовании стандартной функции поиска определителя матрицы (*Determinant*).

2.2. Учащиеся с нечётными номерами вариантов реализуют метод решения СЛАУ при использовании стандартной функции поиска обратной матрицы (*Inverse Matrix*).

3.1. Учащиеся с чётными номерами вариантов создают собственную подпрограмму для поиска определителя матрицы, упаковывают её в процедуру и подменяют ею стандартные.

3.2. Учащиеся с нечётными номерами вариантов создают собственную подпрограмму для поиска обратной матрицы, упаковывают её в процедуру и подменяют ею стандартные.

На блок-диаграмме все функциональные, то есть значащие пересечения линий связи (узлы) обозначить точками. Для этого в меню перейти к настройкам «*Tools > Options…*», далее в выпадающем списке перейти к настройкам блок-диаграммы (*Block Diagram*) и выставить галочку напротив пункта «*Show dots at wire junctions*».

**Цель работы (одна из возможных формулировок)**: освоение навыков работы с матрицами. Закрепление раздела «Линейная алгебра» высшей математики.

**Для случаев, когда не удаётся реализовать универсальный алгоритм решения поставленной задачи:**

1. Ввести допущение, что программа не является универсальной и адаптирована под расчёт СЛАУ строго указанной на графическом пользовательском интерфейсе размерности.

2. Прибегнуть к разложению матрицы в частную формулу для поиска её определителя.

**Пример:**

Пусть дана матрица М:



Определитель матрицы М равен:



Для удобства вывода формулы откажемся на время от индексации, а поставим каждому элементу матрицы в соответствие букву латинского алфавита:



Далее для вывода формулы воспользуемся схемой поиска определителя матрицы по минорам:



Определитель рассчитывается строго при фиксированном *j*, то есть для одной любой строки матрицы. Поиск определителя минора для рассматриваемого элемента является рекурсивным обращением к указанной выше схеме. Выходом из рекурсии является достижение размерности 2х2, для которой известно расчётное соотношение вида: «Разность произведений элементов, стоящих на главной диагонали и элементов, стоящих на обратной диагонали».

Тогда для первой строки исходной матрицы рассмотрим её миноры:

1.



Результат был получен следующим образом:



Выполним подстановку переменных:



Упростим:



2.



Результат был получен следующим образом:



Выполним подстановку переменных:



Упростим:



3.



Результат был получен следующим образом:



Выполним подстановку переменных:



Упростим:



4.



Результат был получен следующим образом:



Выполним подстановку переменных:



Упростим:



Вернёмся к исходной матрице М и запишем:



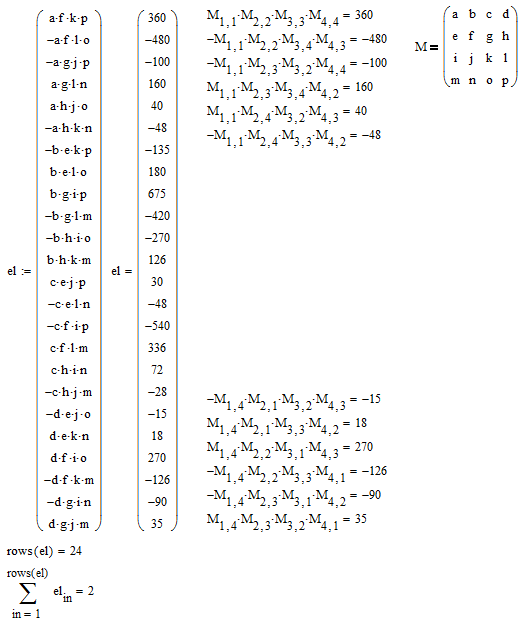
Иначе:



После подстановки выражений и упрощения получим:



Составим таблицу соответствия:



**Варианты индивидуального задания:**

Таблица 1 – Варианты индивидуального задания

|  |  |
| --- | --- |
| 1. | 2. |
| 3. | 4. |
| 5. | 6. |
| 7. | 8. |
| 9. | 10. |
| 11. | 12. |
| 13. | 14. |
| 15. | 16. |
| 17. | 18. |
| 19. | 20. |
| 21. | 22. |
| 23. | 24. |
| 25. | 26. |

# 20 Задание «…».

В пакете прикладных программ *National Instruments LabView* создать виртуальные приборы для загрузки данных из файла и сохранения данных в файл.

Для чётных вариантов необходимо реализовать два виртуальных прибора: один для загрузки данных, другой – для сохранения данных.

Для нечётных вариантов необходимо реализовать один виртуальный прибор, сочетающий в себе как загрузку данных из файла, так и сохранение данных в файл.

Использовать следующие схемы загрузки/сохранения:

а) Расчёт значений функции выполняется в *Microsoft Office Excel* и записывается в файл *fromExcel.txt* (в текстовый файл копируются два столбца: в одном содержатся значения абсцисс, в другом – значения ординат). Между столбцами вручную должны быть удалены элементы табуляции, разделяющие значения, и на их место вставлены разделители в соответствии с таблицей вариантов (Таблица 2). В пакете прикладных программ *National Instruments LabView* выполняется построчное считывание данных из файла *fromExcel.txt* и конвертация их для построения графика функциональной зависимости.

**Внимание:** учесть наличие разрывов в функциональных зависимостях. Наличие соединения точек, расположенных на границах разрыва будет считаться ошибкой.

б) Расчёт значений функции выполняется в пакете прикладных программ *National Instruments LabView* и записывается в файл *fromLabView.txt*. В *Microsoft Office Excel* из файла *fromLabView.txt* копируются рассчитанные значения, и далее по ним проводится построение графика функциональной зависимости.

**Внимание:** учесть наличие разрывов. Все выявленные разрывы отобразить асимптотами.

Для вариантов, делящихся на 5 без остатка, реализовать точность вычислений до одного знака после запятой.

Для вариантов, дающих в остатке единицу при делении на 5, – точность до двух знаков после запятой.

Для вариантов, дающих в остатке двойку при делении на 5, – точность до трёх знаков после запятой.

Для вариантов, дающих в остатке тройку при делении на 5, – точность до четырёх знаков после запятой.

Для вариантов, дающих в остатке четвёрку при делении на 5, – точность до пяти знаков после запятой.

На блок-диаграмме все функциональные, то есть значащие пересечения линий связи (узлы) обозначить точками. Для этого в меню перейти к настройкам «*Tools > Options…*», далее в выпадающем списке перейти к настройкам блок-диаграммы (*Block Diagram*) и выставить галочку напротив пункта «*Show dots at wire junctions*».

**Цель работы (одна из возможных формулировок)**: освоение навыков работы с текстовыми файлами как протоколами передачи информации между средами программирования и пакетами прикладных программ; приобретение знаний, касающихся настройки диалогов и событий; знакомство с механизмом обработки событий в среде с графическим языком программирования.

**Варианты исследуемых функций:**

Таблица 1 – Варианты исследуемых функций

|  |
| --- |
| 1. **а)** x4+2·x2+x+1; **б)** x/(1+1/x); |
| 2. **а)** tg(2·x)·cos(2·x)·lg(2·x); **б)** x·sin(x3); |
| 3. **а)** sinh(x/e)+ex+1 **б)** 1/(x·lg(x)); |
| 4. **а)** sin(1/(a·x))·cos(b·x); **б)** sin(x)·tg(1/x); |
| 5. **а)** sin(x2)· ctg(1/x); **б)** sin(1/(2·x))·cos(1/(3·x)); |
| 6. **а)** x3·cos(x)+x+1; **б)** x/(lg(x) + 1); |
| 7. **а)** x4·sin(x); **б)** tg(x)·sin(3·x); |
| 8. **а)** ctg(5·x); **б)** tg(x)3; |
| 9. **а)** sin(2·x)·cos(4·x); **б)** sin(2·x)· ctg(x); |
| 10. **а)** log2(x); **б)** cos(x)·x2; |
| 11. **а)** x2+2·x+1; **б)** 1/sin(x); |
| 12. **а)** x2·sin(2·x); **б)** tg(2·x)·cos(2·x); |
| 13. **а)** sin(x) ·x; **б)** x·sin(2·x)+2·x·cos(x); |
| 14. **а)** log2(x)·cos(x); **б)** x2/sin(x)+cos(x)/x; |
| 15. **а)** x2·sin(x); **б)** 1/sin(x)+1/cos(x); |
| 16. **а)** tg(2·x)4; **б)** ctg(3·x)·x; |
| 17. **а)** x·lg(x); **б)** sinh(2·x)+ex·sin(x); |
| 18. **а)** lg(x); **б)** x2·cos(x)+ x·sin(x)+x; |
| 19. **а)** sin(2·x)·tg(x); **б)** sin(a·x)·cos(b·x); |
| 20. **а)** sin(a·x)·cos(b·x); **б)** sin(2·x)·tg(x); |
| 21. **а)** x2·cos(x)+ x·sin(x)+x; **б)** lg(x); |
| 22. **а)** sinh(2·x)+ex·sin(x); **б)** x·lg(x); |
| 23. **а)** ctg(3·x)·x; **б)** tg(2·x)4; |
| 24. **а)** 1/sin(x)+1/cos(x); **б)** x2·sin(x); |
| 25. **а)** x2/sin(x)+cos(x)/x; **б)** log2(x)·cos(x); |
| 26. **а)** x·sin(2·x)+2·x·cos(x); **б)** sin(x) ·x; |
| 27. **а)** tg(2·x)·cos(2·x); **б)** x2·sin(2·x); |
| 28. **а)** 1/sin(x); **б)** x2+2·x+1; |
| 29. **а)** cos(x)·x2; **б)** log2(x); |
| 30. **а)** sin(2·x)·ctg(x); **б)** sin(2·x)·cos(4·x); |

Таблица 2 – Варианты разделителей данных

|  |  |  |
| --- | --- | --- |
| 1. **##** | 11. **[** | 21. **>** |
| 2. **Space** | 12. **~** | 22. **|** |
| 3. ++ | 13. **\** | 23. **^** |
| 4. **]** | 14. **<** | 24. **/** |
| 5. \_ | 15. **+** | 25. **&** |
| 6. **//** | 16. **{** | 26. \* |
| 7. **}** | 17. **#** | 27. **@** |
| 8. **№** | 18. **(** | 28. **!** |
| 9. **=** | 19. **)** | 29. **%** |
| 10. **?** | 20. **-** | 30. **;** |

# 21 Задание «…».

В пакете прикладных программ *National Instruments LabView* создать виртуальный прибор, моделирующий поведение типовых сигнализаторов и индикаторов (заданных по вариантам).

На блок-диаграмме все функциональные, то есть значащие пересечения линий связи (узлы) обозначить точками. Для этого в меню перейти к настройкам «*Tools > Options…*», далее в выпадающем списке перейти к настройкам блок-диаграммы (*Block Diagram*) и выставить галочку напротив пункта «*Show dots at wire junctions*».

**Цель работы (одна из возможных формулировок)**: освоение навыков работы с локальными и глобальными переменными, а также структурой последовательного вычислительного процесса внутри системы, адаптированной под потоковые вычисления.

**Варианты сигнализаторов и индикаторов:**

Таблица 1 – Типовые сигнализаторы и индикаторы

|  |
| --- |
| 1. Смоделировать графический индикатор загрузки файла указанного объёма при варьируемой скорости соединения без числовой индикации прогресса. |
| 2. Смоделировать работу термостата на электрическом нагревателе (радиаторе) без числовой индикации. |
| 3. Смоделировать обратный отсчёт на семисегментном индикаторе с девяти до нуля. |
| 4. Смоделировать работу автомобильного светофора с числовой индикацией (обратный отсчёт до смены сигнала). |
| 5. Смоделировать прямой счёт на семисегментном индикаторе от нуля до девяти. |
| 6. На светодиодной матрице размерностью [5x10] выполнить прямой счёт от нуля до девяти. |
| 7. Смоделировать индикатор загрузки файла указанного объёма при варьируемой скорости соединения с числовой индикацией прогресса (показывать, сколько байтов загружено). |
| 8. Смоделировать работу пешеходного светофора без числовой индикации. |
| 9. Смоделировать работу термостата на электрическом нагревателе с числовой индикацией (показывать количество градусов по Цельсию с точностью до двух знаков). |
| 10. Смоделировать работу трамвайного светофора без числовой индикации. |
| 11. Смоделировать обратный отсчёт на двух семисегментных индикаторах от шестнадцати до нуля. |
| 12. Смоделировать работу пешеходного светофора с числовой индикацией (показывать количество секунд, оставшихся до переключения). |
| 13. Смоделировать работу сливного бачка унитаза (нажатие на кнопку инициирует сброс воды, по окончании сброса начинается наполнение до указанного уровня). |
| 14. Смоделировать прямой счёт на двух семисегментных индикаторах от нуля до шестнадцати. |
| 15. Смоделировать работу пешеходного светофора с индикацией обратного отсчёта на семисегментных индикаторах. |
| 16. На светодиодной матрице размерностью [5x10] выполнить обратный отсчёт от девяти до нуля. |
| 17. Смоделировать индикатор загрузки приложения известного объёма при варьируемой скорости соединения. |
| 18. Смоделировать работу трамвайного светофора с числовой индикацией (обратный отсчёт до смены состояний). |
| 19. Выполнить горизонтальное послойное заполнение светодиодной матрицы [5x10] с сокращающийся длительностью задержки на каждом слое. |
| 20. Выполнить вертикальное послойное заполнение светодиодной матрицы [5x10] с увеличивающейся длительностью задержки на каждом слое. |

# 22 Задание «…».

В пакете прикладных программ *National Instruments LabView* создать виртуальный прибор для решения систем обыкновенных дифференциальных уравнений (ОДУ) методами:

- Эйлера (Э);

- Кэша-Карпа (КК) 5-го порядка;

- Рунге-Кутта (РК) 4-го порядка.

Дифференциальные уравнения и разновидности методов, применяемых для их решения заданы по вариантам.

Сопоставить между собой результаты расчётов указанными методами при различных значениях шага.

**Цель работы (одна из возможных формулировок)**: освоение навыков работы со стандартными подпрограммами для решения ОДУ и унификации ввода исходных данных для стандартных подпрограмм этого типа. Закрепление раздела «Дифференциальные уравнения» высшей математики.

**Место расположения стандартных процедур для решения ОДУ:**

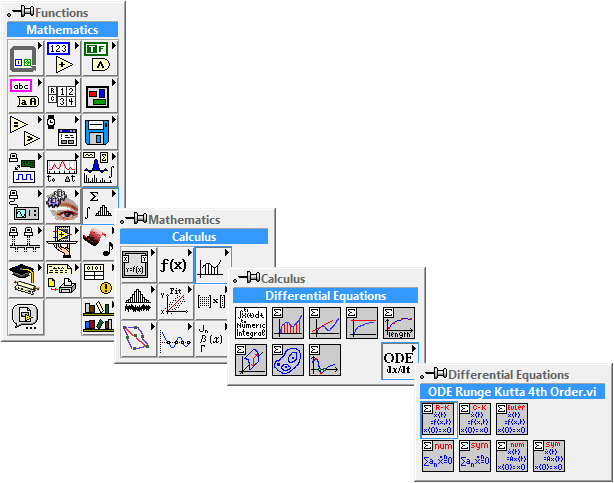


Рисунок 1 – Расположение стандартного блока для решения обыкновенных дифференциальных уравнений методом Рунге-Кутта 4-го порядка

Рассмотрим пример, представленный в справочной системе самого пакета прикладных программ *National Instruments LabView*:

|  |  |
| --- | --- |
| Дана математическая запись системы обыкновенных дифференциальных уравнений с их начальными условиями: | В программу виртуального прибора та же запись должна быть перенесена в следующем векторном формате:  - система:    - расчётный диапазон:    - имена неизвестных функций:    - начальные условия: |
| Интерфейс виртуального прибора может выглядеть приблизительно нижеследующим образом (получено решение для x, чему соответствует индекс 0):    Рисунок 2 – Вывод результата для системы ОДУ по нулевому индексу (для «*x(t)*») | |
| Получено решение для y, чему соответствует индекс 1:    Рисунок 3 – Вывод результата для системы ОДУ по первому индексу (для «*y(t)*») | |
| Получено решение для z, чему соответствует индекс 2:    Рисунок 4 – Вывод результата для системы ОДУ по второму индексу (для «z*(t)*») | |

|  |
| --- |
| Блок-диаграмма виртуального прибора может выглядеть приблизительно нижеследующим образом:    Рисунок 5 – Блок диаграмма корректного подключения стандартного блока для решения ОДУ методом Рунге-Кутта 4-го порядка |

Пример перестроения ОДУ второго порядка под разработанный ранее виртуальный прибор:

Пусть задано ОДУ второго порядка вида: .

С начальными условиями .

Введём замену: .

Тогда: .

При таком раскладе решается следующая система: 

Решением будет та функция, которая соответствует строке: .

Выполним проверку в *MathCAD* с заменой:



Рисунок 6 – Решение дифференциального уравнения второго порядка в пакете прикладных программ *MathCAD* с заменой функций

Выполним прямую проверку в *MathCAD*:



Рисунок 7 – Решение дифференциального уравнения второго порядка в пакете прикладных программ *MathCAD* напрямую

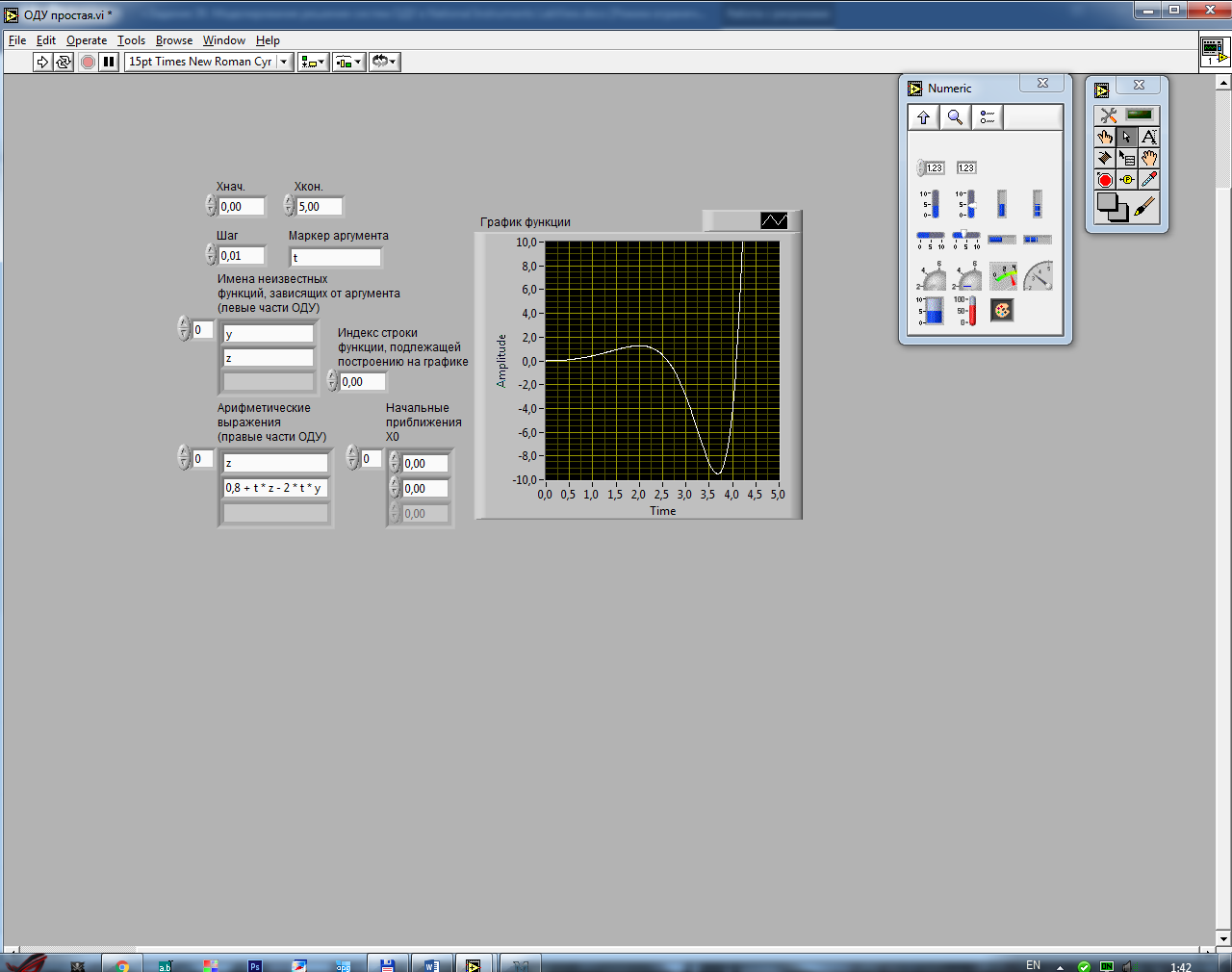


Рисунок 8 – Решение дифференциального уравнения второго порядка в пакете прикладных программ *National Instruments LabView* с заменой функций для самой функции (индекс 0)

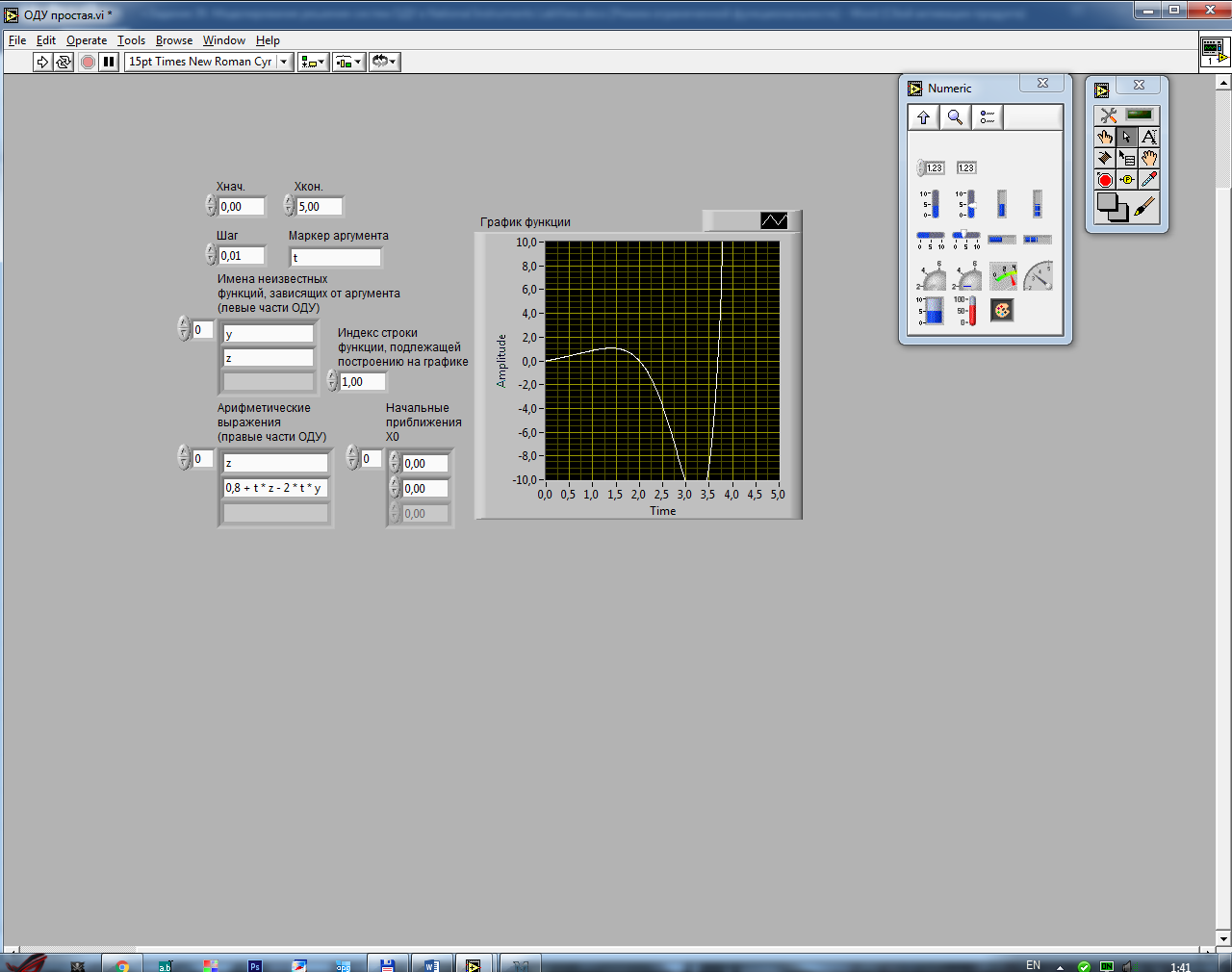


Рисунок 9 – Решение дифференциального уравнения второго порядка в пакете прикладных программ *National Instruments LabView* с заменой функций для производной функции (индекс 1)

**Варианты индивидуального задания:**

Таблица 1 – Варианты индивидуального задания

|  |
| --- |
| **1.**  Реальное дифференцирующее звено.  Начальное приближение h(0) = 0.  Входной сигнал x(t) = t.  **РК, Э, КК с выводом каждого результата в свой XY Graph.** |
| **2.**  Колебательное звено.  Начальное приближение h(0) = 0, h'(0) = 0.  Входной сигнал x(t) = 1.  **РК и КК объединить на одном XY Graph, Э отдельно.** |
| **3.**  Инерционное звено с запаздыванием.  Начальное приближение h(0) = 0.  Входной сигнал x(t) = t.  **РК и Э объединить на одном XY Graph, КК отдельно.** |
| **4.**  Инерционное звено.  Начальное приближение h(0) = 0.  Входной сигнал x(t) = 1.  **Э и КК объединить на одном XY Graph, РК отдельно.** |
| **5.**  Начальное приближение h(0) = 0.  Входной сигнал x(t) = sin(2t).  **РК, Э, КК объединить на одном XY Graph.** |
| **6.**  Начальное приближение h(0) = -2500, h'(0) = -2500.  Входной сигнал x(t) = et.  **РК, Э, КК с выводом каждого результата в свой XY Graph.** |
| **7.**  Интегрирующее звено.  Начальное приближение h(0) = 0.  Входной сигнал x(t) = 1.  **РК и КК объединить на одном XY Graph, Э отдельно.** |
| **8.**  Инерционное звено.  Начальное приближение h(0) = 0.  Входной сигнал x(t) = 1.  **РК, Э, КК объединить на одном XY Graph.** |
| **9.**  Реальное дифференцирующее звено.  Начальное приближение h(0) = 6.  Входной сигнал x(t) = 1.  **РК и Э объединить на одном XY Graph, КК отдельно.** |
| **10.**  Колебательное звено.  Начальное приближение h(0) = 0, h'(0) = 0. Входной сигнал x(t) = t.  **Э и КК объединить на одном XY Graph, РК отдельно.** |
| **11.**  Инерционное звено с запаздыванием.  Начальное приближение h(0) = 0.  Входной сигнал x(t) = t2.  **РК, Э, КК объединить на одном XY Graph.** |
| **12.**  Инерционное звено.  Начальное приближение h(0) = 0.  Входной сигнал x(t) = t.  **РК, Э, КК с выводом каждого результата в свой XY Graph.** |
| **13.**  Начальное приближение h(0) = 0.  Входной сигнал x(t) = sin(t).  **РК и Э объединить на одном XY Graph, КК отдельно.** |
| **14.**  Начальное приближение h(0) = 0, h'(0) = 0.  Входной сигнал x(t) = sin(t2).  **РК, Э, КК объединить на одном XY Graph.** |
| **15.**  Интегрирующее звено.  Начальное приближение h(0) = 0.  Входной сигнал x(t) = t2.  **Э и КК объединить на одном XY Graph, РК отдельно.** |

**Список литературы:**

1. Сафронов, А.И. Составление отчётной документации по решённым задачам алгоритмизации и программирования: Учебно-методическое пособие для проведения аудиторных занятий по Учебной практике // А.И. Сафронов, Н.Н. Зольникова, В.Г. Новиков. – М.: РУТ (МИИТ), 2018. – 83 с.

2. Зольникова Н.Н., Работа в среде *Microsoft Excel*: Методические указания к лабораторным работам по дисциплине «Информатика» // Н.Н. Зольникова, Л.Н. Логинова. – М.: МИИТ, 2012. – 53 с.

3. Зольникова Н.Н., Логинова Л.Н., Сафронов А.И. Построение графиков и диаграмм в Excel: Учебно-методическое пособие к лабораторным работам и практическим занятиям по дисциплине «Информационные технологии». – М.: РУТ (МИИТ), 2017. – 54 с.

# Приложение 1. Типовая структура отчёта по работе, классифицируемой как расчётная задача

Отчёт по выполненной расчётной работе должен содержать:

0. Титульный лист.

1. Формулировку цели работы.

2. Описание задачи согласно выданному варианту.

3. Расчётную часть (с подпунктами, если они необходимы):

4. Формулировку вывода о проделанной работе (обезличено – исключить из вывода местоимения, такие как «я», «мы» и другие).

# Приложение 2. Типовая структура отчёта по работе, классифицируемой как задача программирования и алгоритмизации

Отчёт по выполненной работе программирования и алгоритмизации должен содержать:

0. Титульный лист.

1. Формулировку цели работы.

2. Описание задачи согласно выданному варианту.

2.1. Общая часть.

2.2. Индивидуальная часть (если есть).

3. Составление блок-схемы алгоритма программы.

3.1. Общая часть.

3.2. Индивидуальная часть (если есть).

4. Подбор и расчёт тестовых примеров.

4.1. Общая часть.

4.2. Индивидуальная часть (если есть).

5. Листинг кода составленного программного обеспечения (иллюстрация блок-диаграммы *National Instruments LabView*).

5.1. Общая часть.

5.2. Индивидуальная часть (если есть).

6. Графический пользовательский интерфейс программного обеспечения (передняя панель виртуального прибора *National Instruments* *LabView*) и его описание.

6.1. Общая часть.

6.2. Индивидуальная часть (если есть).

7. Расчёт тестовых примеров с использованием составленного программного обеспечения.

7.1. Общая часть.

7.2. Индивидуальная часть (если есть).

8. Формулировку вывода о проделанной работе (обезличено – исключить из вывода местоимения, такие как «я», «мы» и другие).

# Приложение 3. Перечень типовых рекомендаций для демонстрации самостоятельности выполненной работы обучающимися

Далее сформулирован краткий перечень типовых рекомендаций, уместных для оформления отчёта по каждой из решённых обучающимися задач, изложенных в сборнике:

1. Блок-схема алгоритма: размер для всех блоков по ширине подбирается исходя из ширины наиболее наполненного текстом блока.

2. Блок-схема алгоритма: для случаев, когда в блоке, согласно пункту 1, необходимо разместить довольно большое количество текста, который в итоге обусловит неоправданно громоздкую схему, рекомендуется часть текста оформить в виде комментария к блоку; размеры блоков-комментариев могут превышать принятые размеры основных блоков (комментарии не нормируются).

3. Блок-схема алгоритма: блоки по высоте не нормируются, однако рекомендуется выполнять увеличение их высоты только пропорционально количеству записанных в них действий. **Иначе**: для всей блок-схемы принимается некоторая средняя высота одного блока, достаточная для описания одного действия, и если блок содержит описание двух действий, то его высота может быть увеличена вдвое, трёх действий – втрое, и так далее.

4. Блок-схема алгоритма: в тех случаях, когда блок-схема, размещаемая на странице А4, масштабируется настолько, что текст в блоках становится нечитаемым, рекомендуется распределить блоки на нескольких страницах А4, используя ссылки для демонстрации переходов между страницами по линиям связи.

5. Листинг кода программы: код программы должен содержать значимые имена переменных или таблицу соответствия, раскрывающую смысловое содержание введённых автором переменных, не обладающих значащими именами.

6. Листинг кода программы: для удобства навигации по коду автора программы и проверяющего во время защиты работы, код необходимо снабжать комментариями. Комментарии, выполненные в коде, могут также играть роль связки с блок-схемой алгоритма и упрощать проверку отчёта.

7. Подбор тестовых примеров: минимальное количество тестовых примеров, доказывающих работоспособность составленного программного обеспечения, должно быть достаточным для покрытия всех ветвей разветвляющегося вычислительного процесса.

8. Общее: необходимо выполнение дополнительных скриншотов/иллюстраций для случаев, когда текстовое описание проделанных автором работы действий становится громоздким или трудным для подробного текстового описания.

9. Общее: рекомендуется выполнять нумерацию рисунков (если они есть) с подписями, содержащими наименование этих рисунков, например, «Рисунок 1 – Пользовательский интерфейс калькулятора *Microsoft Windows*».

# Приложение 4. Перечень требований, предъявляемых к именам отчётных файлов, направляемых на удалённую проверку (по электронной почте)

**Общий вид формата имени файла:** «*Дата. Задание. Фамилия.docx*»

**Формат записи даты:** «*ГГГГММДД*», где *ГГГГ* – четыре цифры текущего года, *ММ* – две цифры текущего месяца, *ДД* – две цифры текущего дня.

**Формат записи задания:** «Задание *NNk*», где *NN* – две цифры номера задания, *k* – обозначение «о», если файл содержит общую часть; обозначение «и», если файл содержит индивидуальную часть; обозначение «ои», если файл содержит как общую, так и индивидуальную части.

**Если устранить замечания по работе удаётся в тот же день:** после фамилии ставится пробел и в круглых скобках записывается номер попытки исправления.

**Примеры правильных имён файлов, которые сдаются на проверку впервые:**

«*20180919. Задание 02о. Иванов.docx*»

«*20180926. Задание 03и. Иванов.xlsx*»

«*20181013. Задание 05ои. Иванов.vi*»

**Примеры правильных имён файлов, которые сдаются на проверку повторно в тот же день:**

«*20180919. Задание 02о. Иванов (1).docx*»

«*20180926. Задание 03и. Иванов (1).xlsx*»

«*20181013. Задание 05ои. Иванов (1).vi*»

**Приложение (примеры некоторых разложений тригонометрических функций в ряд):**

1. Разложение синуса в ряд Маклорена:



2. Разложение косинуса в ряд Маклорена:



3. Разложение арксинуса в ряд Маклорена:



4. Разложение арккосинуса в ряд Маклорена:

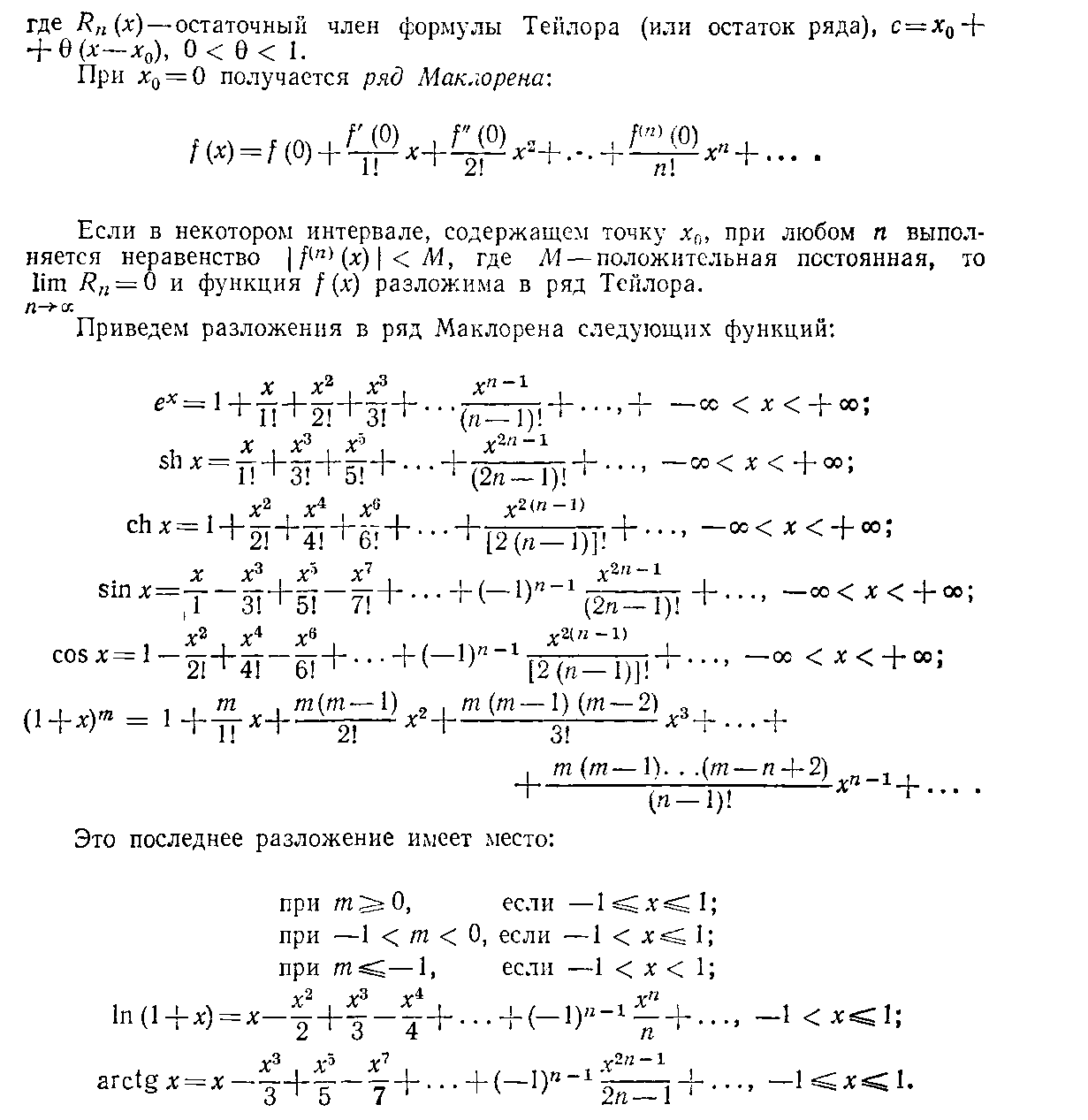


5. Разложение экспоненты в ряд Маклорена:



6. Разложение арктангенса в ряд Маклорена:





Оглавление

# Введение 3