

**Министерство науки и высшего образования
Российской Федерации**

**Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Тульский государственный университет»**

Интернет-институт ТулГУ

Кафедра ИБ

ОТЧЕТ ПО ЛАБОРАТОРНЫМ РАБОТАМ

по дисциплине

«Программирование 2»

Выполнил:

студент группы ИБ262521-ф
Артемов Александр Евгеньевич

Проверил:

канд. техн. наук, доц.
Сафронова Марина Алексеевна

Тула, 2025

Лабораторная работа № 1.

Название работы: Знакомство со средой программирования Microsoft Visual C++.

Цели работы: Изучение меню и настроек интегрированной среды Visual C++.

Задание:

Установить среду программирования Visual C++ и изучить ее интерфейс.

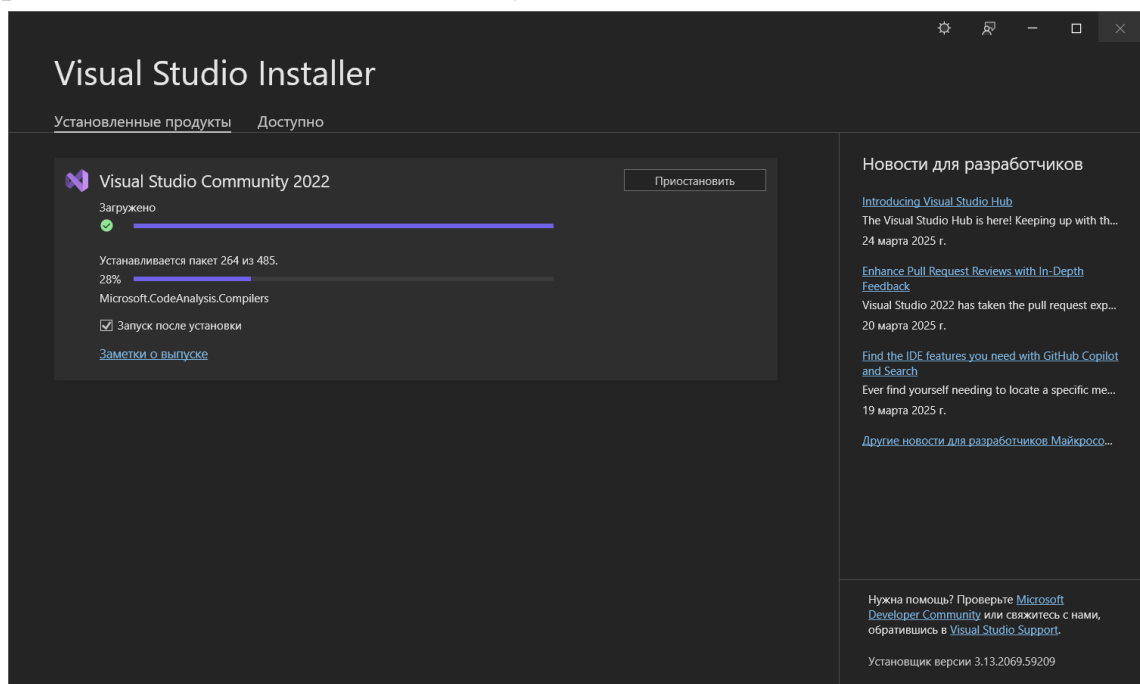
Выполнение лабораторной работы.

Порядок выполнения:

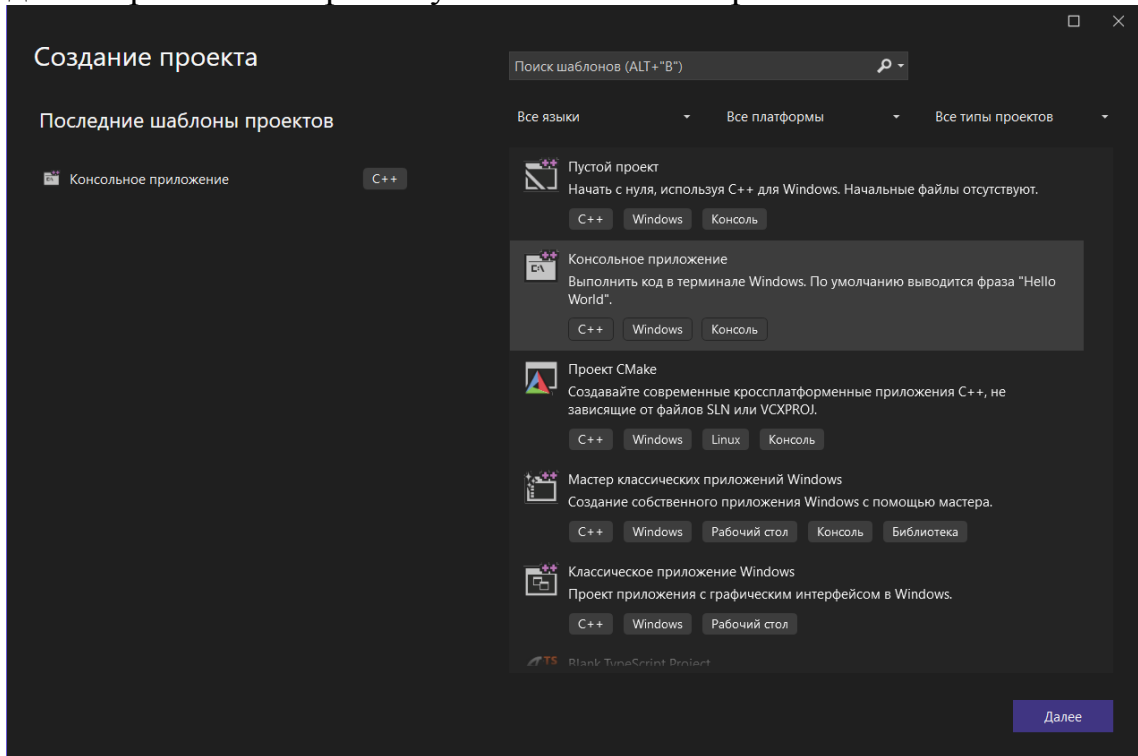
1. Изучить теоретические положения
2. Запустить Visual C++ и создать консольное приложение.
3. Запустить программу на выполнение (сочетание клавиш Ctrl+F5)
4. Внести в программу ошибки
5. Скомпилировать программу и проанализировать результат
6. Оформить отчет

1. Изучены теоретические положения работы.
2. Запуск Visual C++ и создание консольного приложения.

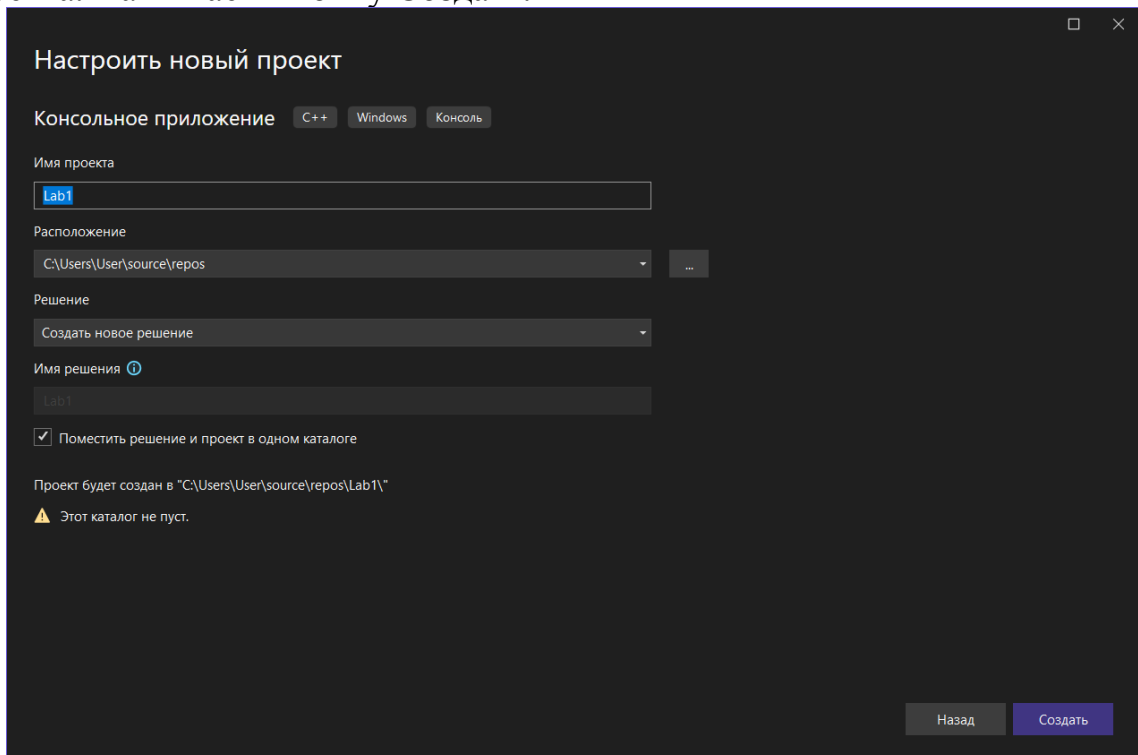
Для знакомства с языком программирования C++ установлена среда разработки Visual Studio Community 2022:



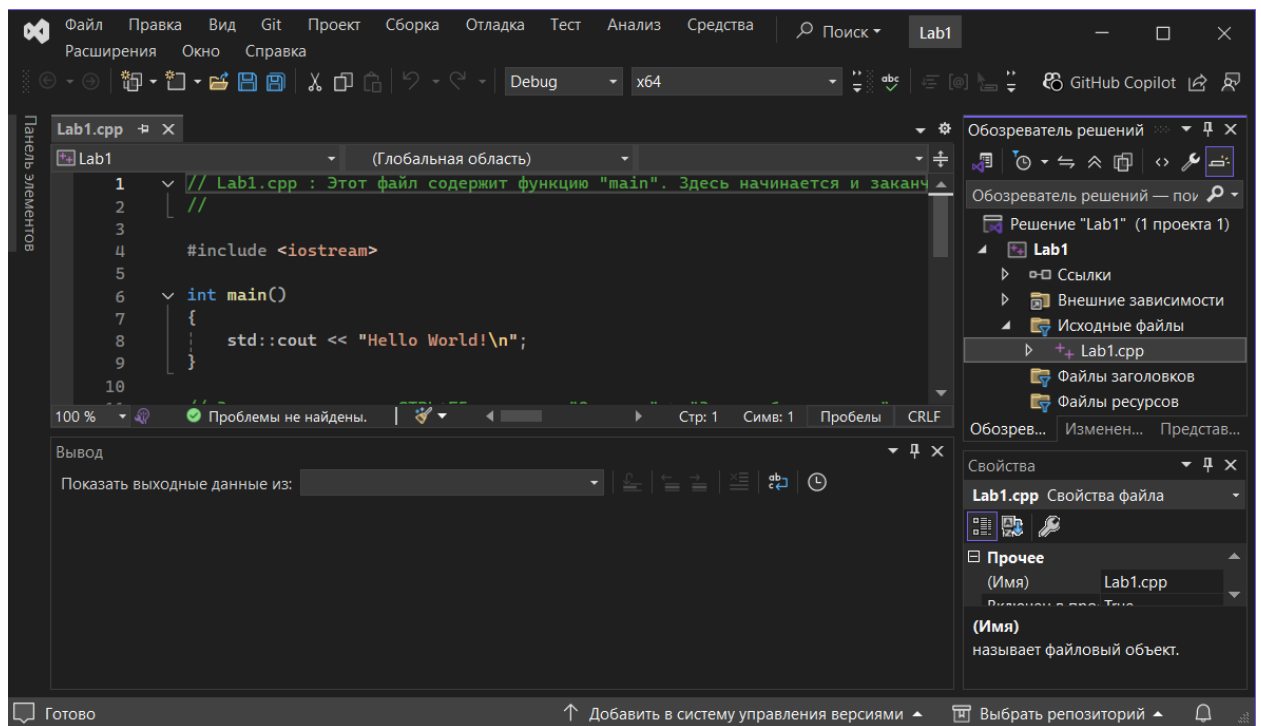
После установки среды перезагружаем ОС, запускаем среду Visual Studio. В меню Файл выбираем Создать → Проект, в диалоговом окне Создание проекта выбираем пункт Консольное приложение.



Нажимаем кнопку Далее, вводим название проекта Lab1 в поле Имя проекта. Нажимаем кнопку Создать.

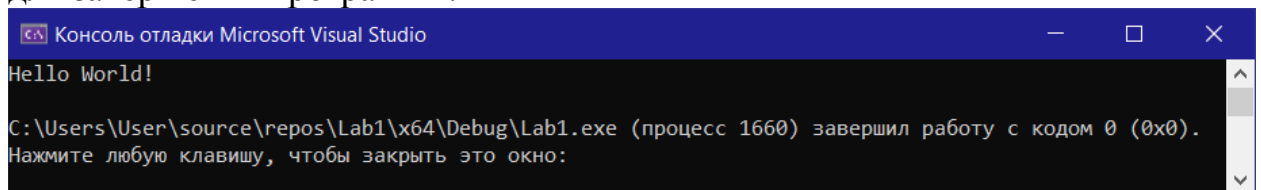


Среда создает новый консольный проект — классический «Hello World!». В редакторе открывается файл исходного кода Lab1.cpp, содержащий точку входа в приложение — функцию main.

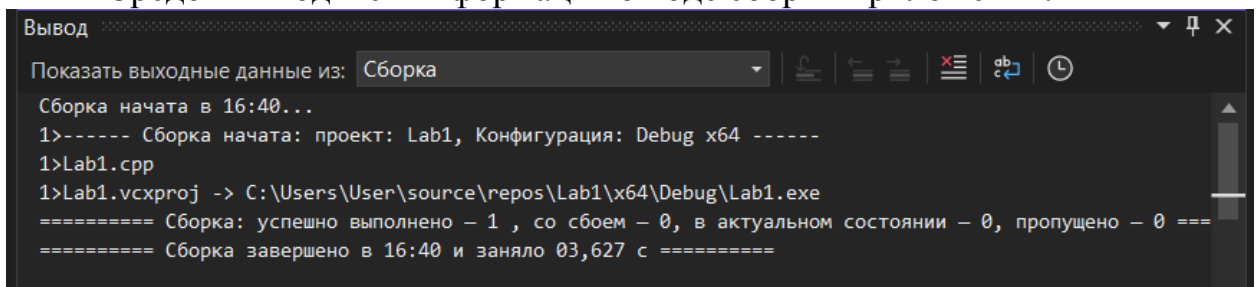


3. Запуск программы на выполнение (сочетание клавиш Ctrl+F5)

Нажатием сочетания клавиш Ctrl+F5 запускаем приложение без отладки. В результате работы программы запускается окно командной строки с выводом строки «Hello World!» и предложением нажать любую клавишу для завершения программы.



Средой выводится информация о ходе сборки приложения.



4. Внесение в программу ошибки

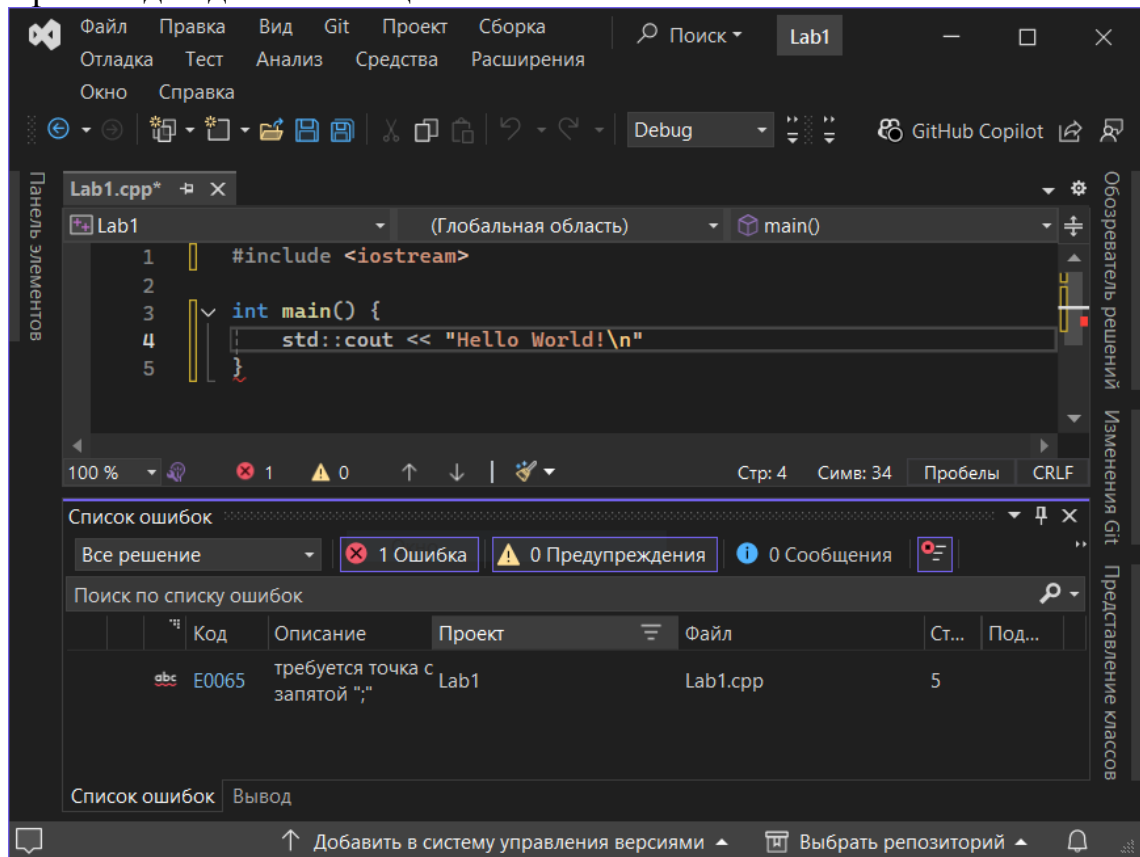
Классическая ошибка на C++ - отсутствие точки с запятой в конце строки: уберем точку с запятой в конце строки вывода «Hello World!».

```
1  #include <iostream>
2
3  int main() {
4      std::cout << "Hello World!\n"
5  }
```

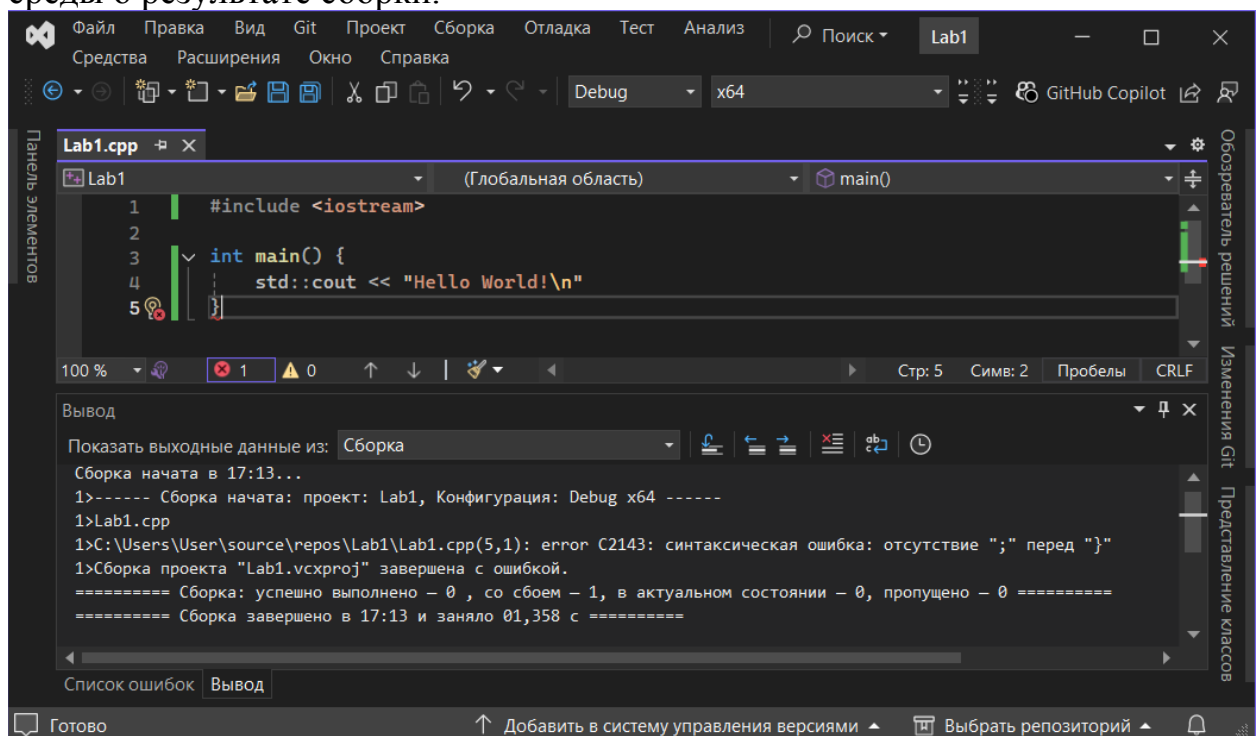
5. Компилирование программы и анализ результата

Еще до компиляции в редакторе кода подсвечена красным закрывающая фигурная скобка функции main, что говорит о наличии

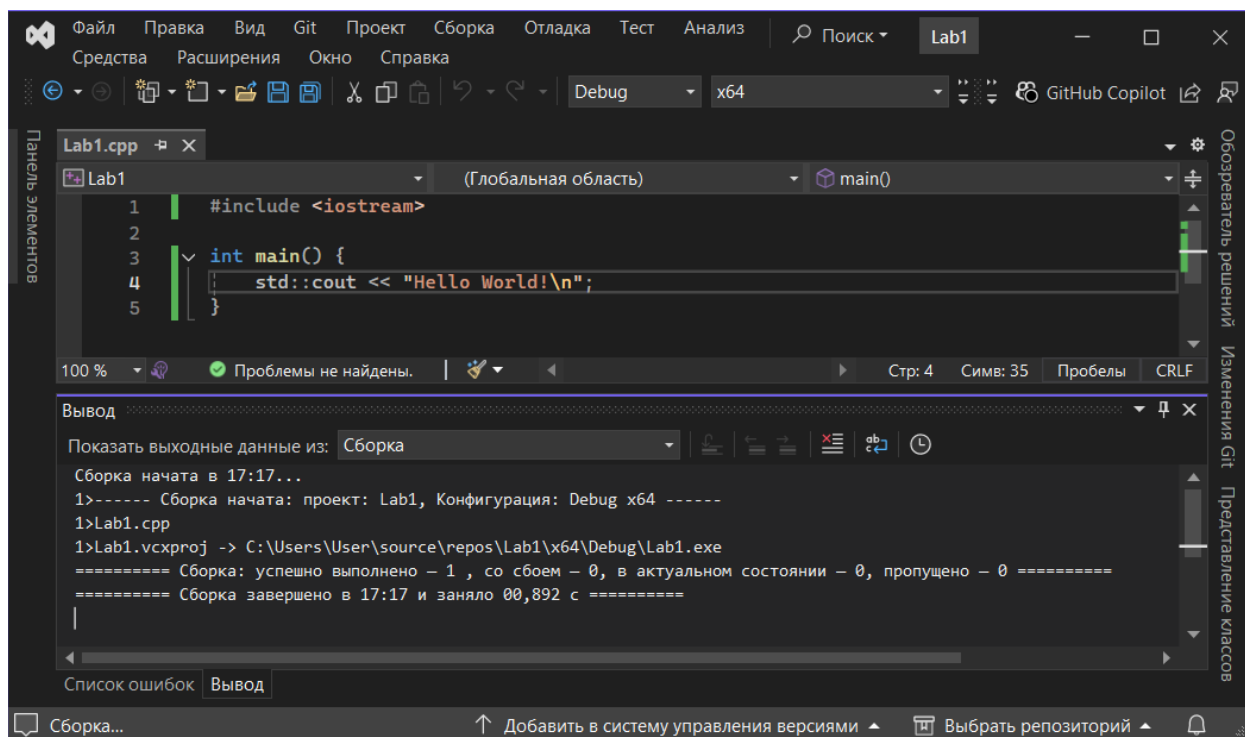
ошибки. Так же анализатор кода среды выводит сообщение об ошибке. И это все происходит до компиляции!



Тем не менее запускаем программу и наблюдаем следующий вывод среды о результате сборки.



Исправляем ошибку и проверяем: сборка выполнена успешно.



6. Оформлен отчет с указанием номера и названия лабораторной работы, ее целей и задач. Текст программы не приводится ввиду его тривиальности.

Visual Studio 2022 Community – это бесплатная интегрированная среда разработки от Microsoft, предназначенная для создания приложений не только на языке C++, но и C#, Python и других. Она предлагает мощные инструменты для профессиональной разработки, включая продвинутый отладчик, умное автодополнение кода, интеграцию с системами контроля версий и поддержку сборщика CMake. Версия «Community» полностью бесплатна для индивидуальных разработчиков, студентов и небольших команд (с некоторыми ограничениями на коммерческое использование).

Среди ключевых преимуществ Visual Studio 2022 для C++ – улучшенная производительность (особенно при работе с крупными проектами), поддержка стандартов C++20/23, встроенный анализатор кода и удобные инструменты для работы с библиотеками. Среда также поддерживает кросс-платформенную разработку (Windows, Linux, Android). Это отличный выбор как для обучения, так и для профессиональных проектов. Для личного использования и небольших команд Visual Studio 2022 Community остаётся одной из лучших бесплатных сред разработки на C++.

Лабораторная работа № 2.

Название работы: Основы языка C/C++. Линейные программы.

Цели работы: Изучение меню и настройка интегрированной среды. Получение навыков по составлению и отладке простейших программ на языке C. Изучение организации простейшего ввода-вывода на языке C.

Задание:

Отладить, откомпилировать программу (создать объектный файл) и выполнить программу:

```
/*          Моя первая программа на Си          */
# include <stdio.h>
main()
{
    int x;
    float y, z;
    printf("Введите значение переменной x: ");
    scanf ("%d", &x);
    printf("Введите значение переменной y: ");
    scanf ("%f", &y);
    z=x+y;
    printf("Сумма введенных переменных=%f", z);
    z=x*y;
    printf("Произведение введенных переменных=%7.3f", z);
    return(0);
}
```

Выполнение лабораторной работы.

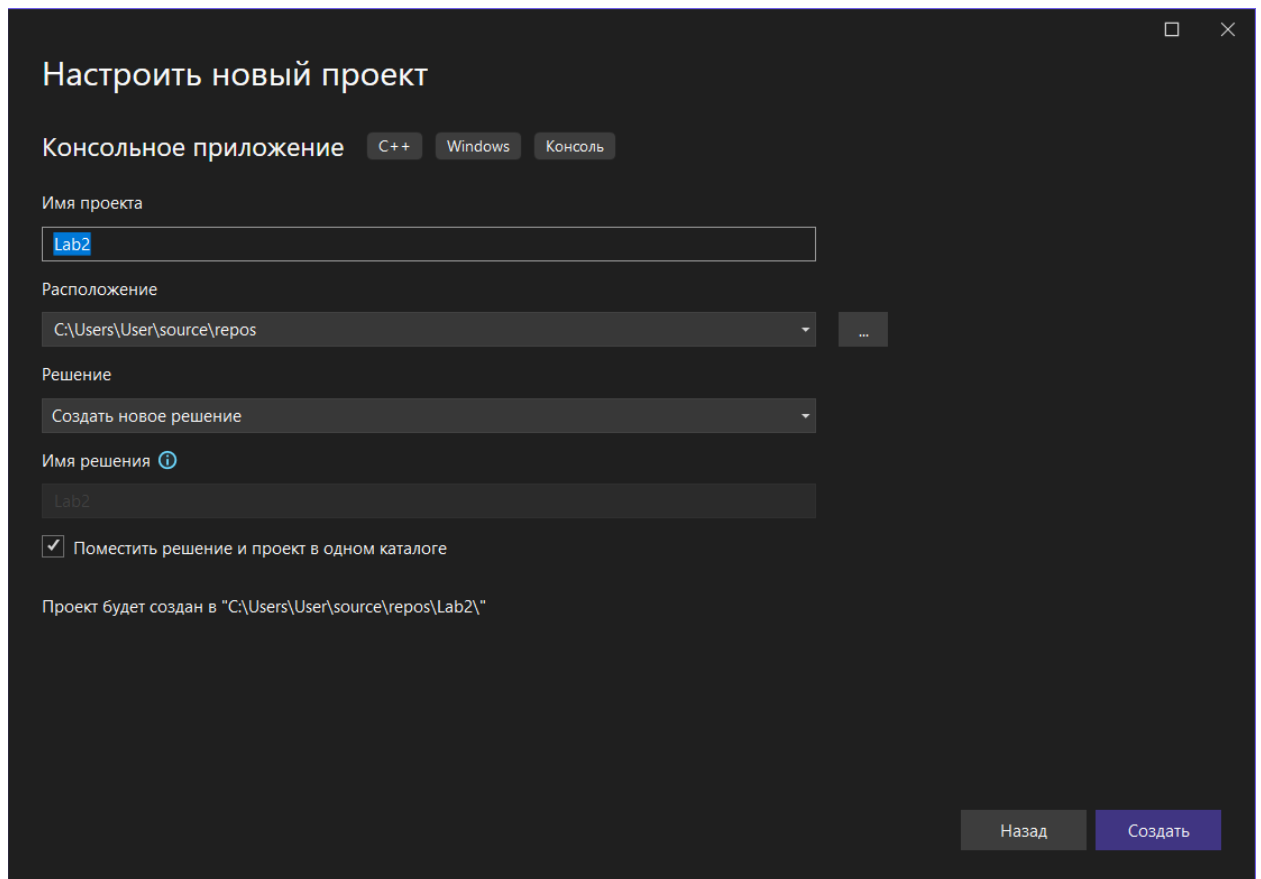
Порядок выполнения:

1. Изучить теоретические положения;
2. Ввести представленную выше программу с клавиатуры;
3. Отладить программу. Результаты работы программы показать преподавателю;
4. Оформить отчет;
5. Защитить лабораторную работу перед преподавателем.

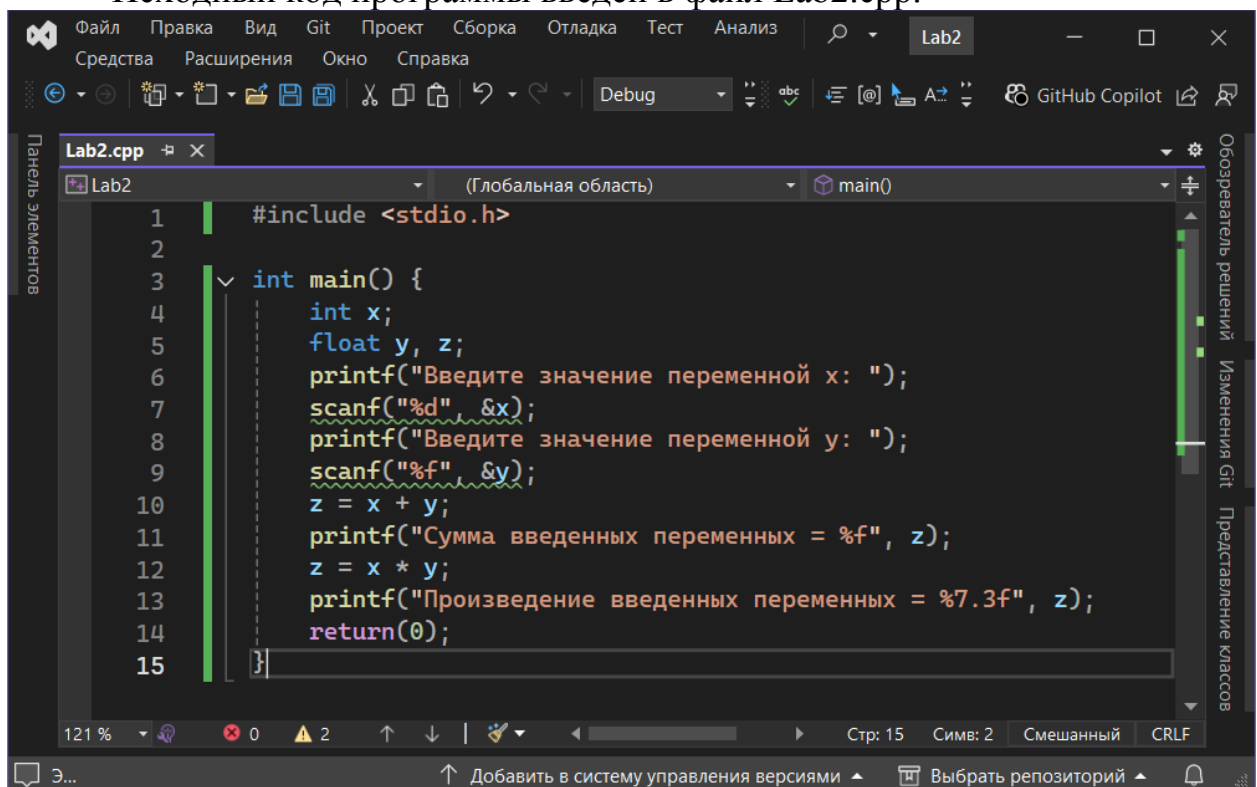
1. Изучены теоретические положения работы.

2. Ввод программы с клавиатуры.

Для выполнения лабораторной работы создан новый проект Lab2.



Исходный код программы введен в файл Lab2.cpp.



3. Отладка программы.

При запуске программы среда выдает одну и ту же ошибку 2 раза. В сообщении об ошибке говорится, что функция `scanf` объявлена устаревшей. Предлагается использовать вместо нее функцию `scanf_s`, либо включать макрос `_CRT_SECURE_NO_WARNINGS`.

Код	Описание	Проект	Файл	Ст...
C4996	'scanf': This function or variable may be unsafe. Consider using scanf_s instead. To disable deprecation, use _CRT_SECURE_NO_WARNINGS. See online help for details.	Lab2	Lab2.cpp	7
C4996	'scanf': This function or variable may be unsafe. Consider using scanf_s instead. To disable deprecation, use _CRT_SECURE_NO_WARNINGS. See online help for details.	Lab2	Lab2.cpp	9

Включим макрос перед подключением файла stdio.h.

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>

int main() {
    int x;
```

Запустим программу на выполнение.

```
Консоль отладки Microsoft Visual Studio
ТТХФШЕх чэрүхэшх яхЁхххээюш х: 5
ТТХФШЕх чэрүхэшх яхЁхххээюш у: 4
Тееьр ттхфхээвї яхЁхххээвї = 9.000000
C:\Users\User\source\repos\Lab2\x64\Debug\Lab2.exe (процесс 1612) завершил работу с кодом 0 (0x0).
Нажмите любую клавишу, чтобы закрыть это окно:
```

В принципе, программа работает верно, но кодировка отображается не корректная. Так же стоит добавить перевод строки перед выводом произведения.

Исправим это. Добавим в начало функции main строку «setlocale(LC_ALL, "Rus");», а так же подключим заголовочный файл «clocale», где эта функция объявлена. Функция setlocale задает языковой стандарт. Так же добавим перевод строки перед выводом произведения.

В результате имеем следующий исходный код:

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <clocale>

int main() {
    setlocale(LC_ALL, "Rus");
    int x;
    float y, z;
    printf("Введите значение переменной x: ");
    scanf("%d", &x);
    printf("Введите значение переменной y: ");
    scanf("%f", &y);
    z = x + y;
    printf("Сумма введенных переменных = %f", z);
    z = x * y;
    printf("\nПроизведение введенных переменных = %7.3f", z);
    return(0);
}
```

При запуске приложения имеем следующий вывод:

```
Консоль отладки Microsoft Visual Studio
Введите значение переменной x: 5
Введите значение переменной y: 4
Сумма введенных переменных = 9,000000
Произведение введенных переменных = 20,000
C:\Users\User\source\repos\Lab2\x64\Debug\Lab2.exe (процесс 7916) завершил работу с кодом 0 (0x0).
Нажмите любую клавишу, чтобы закрыть это окно:
```

4. Оформлен отчет с указанием номера и названия лабораторной работы, ее целей и задач. Приведен откорректированный исходный код программы и результаты его работы.

В данной работе видно, что устаревшие части языка, такие как наследие языка C, все еще могут использоваться при написании программ с использованием современных стандартов, хотя и с ограничениями. Так же видно, что при написании приложений есть возможность использовать различные локализации.

Контрольные вопросы.

1. Какие символы входят в алфавит языка C?

Прописные и строчные буквы латинского алфавита, цифры 0-9, знаки препинания и операторы ~ ! @ # % ^ & * () _ + - = { } [] | \ : ; " ' < > , . ? /, пробел, знаки табуляции (\t) и новой строки (\n).

2. Какова структура программы на языке C?

Общая структура программы на C содержит следующие разделы:

раздел подключения библиотек (#include);

раздел описания констант (#define);

раздел описания глобальных переменных;

раздел объявления функций;

точка входа (функция main);

раздел описания локальных переменных (в теле функции main);

раздел определения функций.

3. Как создать новый файл?

Через меню Файл → Создать → Файл, либо сочетанием клавиш Ctrl+N.

4. Как сохранить программу на внешнем носителе?

Файл исходного кода программы создается и сохраняется на диске при создании проекта. Для сохранения файла на другом диске или под другим именем нужно выбрать меню Файл → Сохранить как... и указать имя файла или место сохранения.

5. Как запустить программу на выполнение?

Запустить программу на выполнение можно через меню Отладка → Запуск без отладки, либо с сочетанием клавиш Ctrl+F5.

6. Как просмотреть результат выполнения программы?

При запуске программы из среды разработки откроется окно командной строки, в котором будет отображен результат выполнения программы.

7. Что выделяется значками /* */?

Многострочный комментарий.

8. Что выполняется в следующей строке – #include <stdio.h>?

Строка #include <stdio.h> в языке C выполняет директиву препроцессора, которая подключает к программе содержимое стандартного заголовочного файла stdio.h (Standard Input/Output).

9. Для чего используется оператор – printf()?

Оператор `printf()` в языке C используется для форматированного вывода данных на стандартное устройство вывода (обычно — консоль).

10. Для чего необходима запись `– main()`?

Функция `main()` в языке C (и C++) — это главная и обязательная точка входа в программу. Без неё выполнение программы невозможно, так как именно с `main()` начинается работа кода после запуска.

11. Для чего необходимы `{ }`?

Фигурные скобки `{ }` в языке C выполняют две ключевые роли: определение блоков кода и определение области видимости переменных.

12. Для чего используется оператор `return(0)`?

Оператор `return(0)` используется для завершения работы функции `main()` (всей программы), а так же подает сигнал операционной системе об успешном завершении работы программы.

13. Что означает выражение `stdio.h`?

`stdio.h` — это стандартный заголовочный файл (header) в языке C, название которого расшифровывается как "Standard Input/Output Header" (заголовок стандартного ввода-вывода). Он содержит объявления функций, макросов и типов данных, необходимых для работы с вводом и выводом данных в программе.

14. Какие операторы ввода Вы знаете?

`scanf()`; `scanf_s()`; `gets()`.

15. Какие операторы вывода Вы знаете?

`printf()`; `puts()`; `putchar()`.

16. В чем особенность оператора `scanf`?

Обязательно использовать `&` перед переменными, т. к. работает с указателями.

17. В чем особенность оператора `printf`?

Можно указывать точность, ширину и другие модификаторы, использовать управляющие символы.

18. В чем особенность оператора `gets`?

Выполняет считывание символов из стандартного входного потока. Если входной поток прерывается символом перехода на новую строку `«\n»`, то этот символ отбрасывается и не попадает в строку.

19. Какие спецификаторы необходимо использовать при работе с целыми числами?

`%d` или `%i` — `int` (знаковое) — стандартный вывод/ввод целых чисел.

`%u` — `unsigned int` — для беззнаковых целых чисел.

`%ld` — `long int` — длинные целые числа.

`%lu` — `unsigned long int` — беззнаковые длинные целые.

`%lld` — `long long int` — очень длинные целые (C99 и новее).

`%llu` — `unsigned long long int` — беззнаковые очень длинные целые.

`%hd` — `short int` — короткие целые числа.

`%hu` — `unsigned short int` — беззнаковые короткие целые.

20. Какие спецификаторы необходимо использовать при работе с вещественными числами?

`%f` – float / double – стандартный вывод/ввод. По умолчанию 6 знаков после точки.

`%lf` – double – только для `scanf()` (в `printf()` `%f` и `%lf` эквивалентны для double).

`%Lf` – long double – для чисел типа long double.

`%.<N>f` – округление до N знаков после точки.

`%<M>.<N>f` – ширина поля M символов, N знаков после точки (выравнивание вправо).

`%e` / `%E` – экспоненциальная запись (научная нотация).

`%g` / `%G` – автоматический выбор между `%f` и `%e` для компактного вывода.

21. Какие спецификаторы необходимо использовать при работе с отдельными символами?

`%c` – char – выводит/считывает один символ (включая пробелы и управляющие символы).

`%hhd` – char – выводит ASCII-код символа (целое число).

`%hhx` – char – выводит ASCII-код в шестнадцатеричном формате (например, 41 для 'A').

22. Какие спецификаторы необходимо использовать при работе со строками символов?

`%s` – выводит/считывает строку до первого нуль-терминатора (`\0`).

`%[^\n]` – считывает строку включая пробелы (до символа `\n`).

`%Ns` – считывает не более N символов (для защиты от переполнения буфера).

Лабораторная работа № 3.

Название работы: Язык C/C++. Условный оператор.

Цели работы: Получение навыков использования управляющих условных операторов `if`, `if-else`, `if-else-if`, `switch` и получение навыков использования условного выражения.

Задание:

Составить программу (в соответствии с вариантом задания), работающую в двух режимах: в первом режиме производится вычисление функции M целого типа, во втором – функции Y вещественного типа. Выбор режима осуществить оператором `switch`. Исходные данные задаются с учетом типов переменных: A, B, C – целого типа, X, K – вещественного.

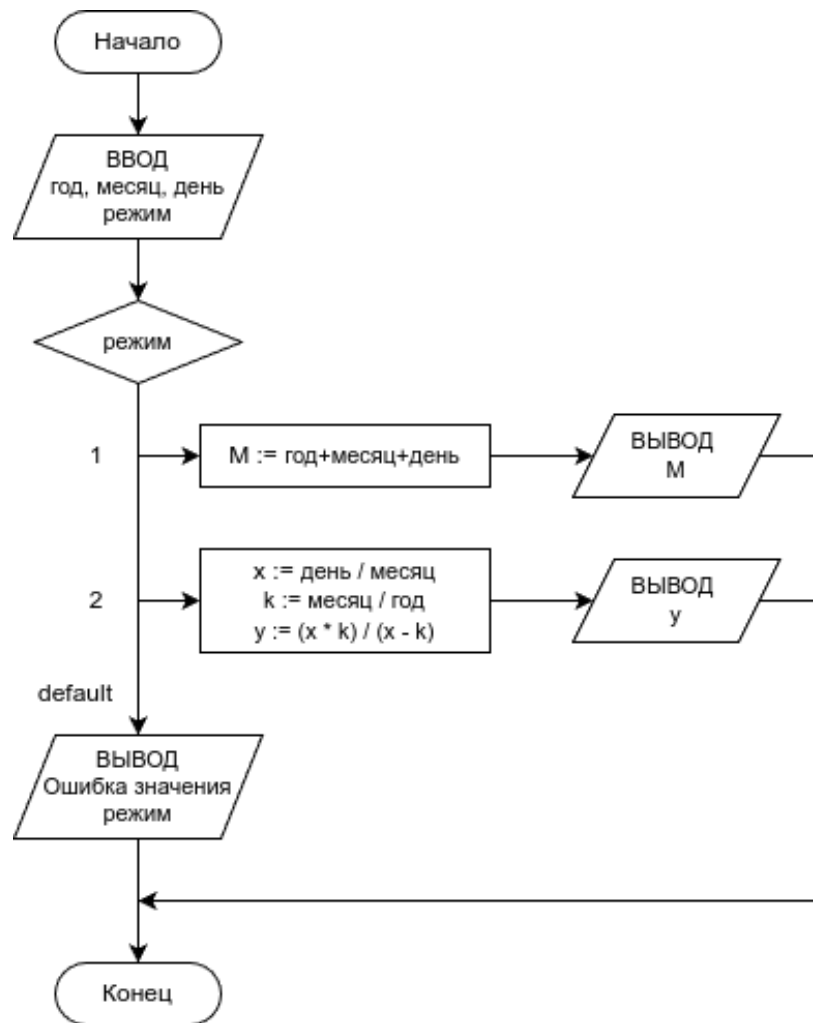
Варианты: $M = A + B + C$, где A, B, C – год, месяц и число рождения студента соответственно. $Y = (X \cdot K)/(X - K)$, где X – отношение даты рождения к месяцу рождения студента; K – отношение месяца рождения студента к году рождения студента.

Выполнение лабораторной работы.

Порядок выполнения:

1. Изучить теоретические положения;
 2. Разработать схему программы;
 3. Составить программу;
 4. Отладить программу. Результаты работы программы показать преподавателю;
 5. Оформить отчет;
-
1. Изучены теоретические положения работы.
 2. Разработка схемы программы.

В ходе работы программа должна вычислить некоторое значение на основании введенных данных. Так же во входных данных задается режим, который определяет способ вычисления данных. Во входных данных указывается год, месяц и день рождения студента, а так же режим — число 1 или 2. При вводе режима 1 вычисляется целое значение как сумма введенных года, месяца и дня. При вводе режима 2 вычисляется вещественное значение по более сложной формуле: $(X \cdot K)/(X - K)$, где X – отношение даты рождения к месяцу рождения студента; K – отношение месяца рождения студента к году рождения студента. Вычисленные значения, как в 1-ом, так и во 2-ом режиме выводятся на экран.



3. Составление исходного кода программы.

```

#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <locale>

int main() {
    setlocale(LC_ALL, "Rus");
    int a, b, c, r, M;
    float x, k, y;
    printf("Введите год рождения: ");
    scanf("%d", &a);
    printf("Введите месяц рождения: ");
    scanf("%d", &b);
    printf("Введите день рождения: ");
    scanf("%d", &c);
    printf("Выберите режим (1 или 2): ");
    scanf("%d", &r);
    printf("Режим %d.\n", r);
    switch (r) {
        case 1:
            M = a + b + c;
            printf("M = %d + %d + %d = %d.\n", a, b, c, M);
            break;
        case 2:

```

```

        x = float(c) / float(b);
        k = float(b) / float(a);
        y = (x * k) / (x - k);
        printf("Y = (%.3f * %.3f) / (%.3f - %.3f) = %.3f.\n",
               x, k, x, k, y);
        break;
default:
    printf("Введено %d вместо 1 или 2.\n", r);
    break;
}

return 0;
}

```

Данный код выводит приглашения для ввода данных и записывает введенные данные в заранее объявленные переменные. В зависимости от введенного значения режима (переменной *r*) проводятся вычисления: если $r = 1$, то вычисляется значение $M = a + b + c$; если $r = 2$, то вычисляются значения $x = \text{float}(c) / \text{float}(b)$; $k = \text{float}(b) / \text{float}(a)$; $y = (x * k) / (x - k)$. При вводе значения отличного от 1 или 2 при вводе режима, программа выводит сообщение «Введено %d вместо 1 или 2.», где %d - введенное значение.

При вычислении переменных *x* и *k* переменные *a*, *b* и *c* приводятся к вещественному типу для того, чтоб результат деления был так же вещественным.

4. Отладить программу. Результаты работы программы показать преподавателю.

```

PowerShell 7 (x64)
PS C:\Users\User\source\repos\Lab3\x64\Debug> .\Lab3.exe
Введите год рождения: 2000
Введите месяц рождения: 6
Введите день рождения: 29
Выберите режим (1 или 2): 1
Режим 1.
M = 2000 + 6 + 29 = 2035.
PS C:\Users\User\source\repos\Lab3\x64\Debug> .\Lab3.exe
Введите год рождения: 2000
Введите месяц рождения: 6
Введите день рождения: 29
Выберите режим (1 или 2): 2
Режим 2.
Y = (4,833 * 0,003) / (4,833 - 0,003) = 0,003.
PS C:\Users\User\source\repos\Lab3\x64\Debug>
PS C:\Users\User\source\repos\Lab3\x64\Debug> .\Lab3.exe
Введите год рождения: 2000
Введите месяц рождения: 6
Введите день рождения: 29
Выберите режим (1 или 2): 5
Режим 5.
Введено 5 вместо 1 или 2.
PS C:\Users\User\source\repos\Lab3\x64\Debug>

```

При выполнении программы видно, что она работает корректно в обоих режимах. При вводе не корректного режима выводится соответствующее сообщение.

5. Оформлен отчет с указанием номера и названия лабораторной работы, ее целей и задач. Разработана схема программы. В соответствии со

схемой написан исходный код. Сборка и запуск программы произведены без ошибок; во время выполнения программы ошибки не возникали.

Контрольные вопросы.

1. В чем особенность работы оператора if?

Если значение выражение истинно, то выполняется действие и программа продолжается, начиная с оператора, следующего за этим действием.

2. В чем особенность работы оператора if-else?

В этом операторе, если выражение истинно, выполняется действие после if; если же выражение ложно, выполняется действие после else.

3. В чем особенность работы оператора if-else-if?

Комбинация операторов if-else-if часто используется для выполнения многочисленных последовательных сравнений.

4. Какой тип может иметь константа в операторе switch?

В switch можно использовать только целочисленные (int, char, short и т.д.) и перечислимые (enum) типы.

5. В чем особенность работы оператора switch?

В возможности сравнить некоторую переменную или выражение с несколькими значениями.

6. В чем особенность использования оператора break в работе оператора switch?

Оператор break останавливает выполнение инструкций выбранного случая и передает управление коду, идущему далее после оператора switch. Если забыть оператор break, то выполнение кода «провалится» в следующий случай, описанный в операторе switch.

7. Когда можно использовать условное выражение?

Условное выражение используется в коде, когда необходимо выполнить те или иные действия в зависимости от значений данных.

8. В чем особенность использования условного выражения?

Особенность условного выражения заключается в том, что в зависимости от значений данных некоторые части кода могут пропускаться и не использоваться при выполнении программы.

Лабораторная работа № 4.

Название работы: Язык C/C++. Операторы циклов.

Цели работы: Освоение навыков работы с операторами цикла `for`, `while`, `do-while`.

Задание:

Написать 3 программы, выполняющие решение задачи 3-мя способами:

1. первая программа решает задачу с применением оператора цикла с параметром (в соответствии с вариантом);
2. вторая программа решает задачу с применением оператора цикла с предусловием (в соответствии с вариантом);
3. третья программа решает задачу с применением оператора цикла с постусловием (в соответствии с вариантом);

Варианты задач: вычислить значение функции $y = x^n$, если $x = [-F; +F]$, где F – дата рождения студента (число); n – произвольное натуральное число от 2 до 8.

Выполнение лабораторной работы.

Порядок выполнения:

1. Изучить теоретические положения;
2. Разработать схему программы;
3. Составить программу;
4. Отладить программу. Результаты работы программы показать преподавателю;
5. Оформить отчет.

1. Изучены теоретические положения работы.

2. Разработка схемы программы.

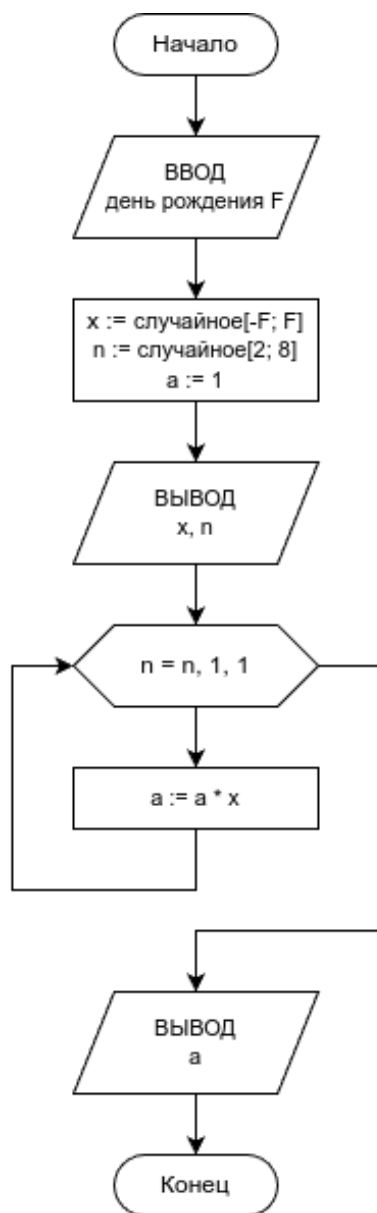
В ходе работы программа должна вычислить значение функции $y = x^n$. Значение x получаем случайным образом из интервала $[-F; +F]$, где F – дата рождения студента (число). Значение n получаем так же случайным образом из интервала от 2 до 8. Так как x^n – это x перемноженный сам на себя n раз, то значение x^n получим выполнив цикл n раз, в теле которого будем умножать итоговое произведение на x . Значение итогового произведения до цикла умножений примем как равное 1.

В первом способе используем цикл с параметром. В качестве параметра используем переменную n , которую будем уменьшать на 1 на каждой итерации, пока ее значение не достигнет 0. В теле цикла итоговое произведение будем умножать на значение x .

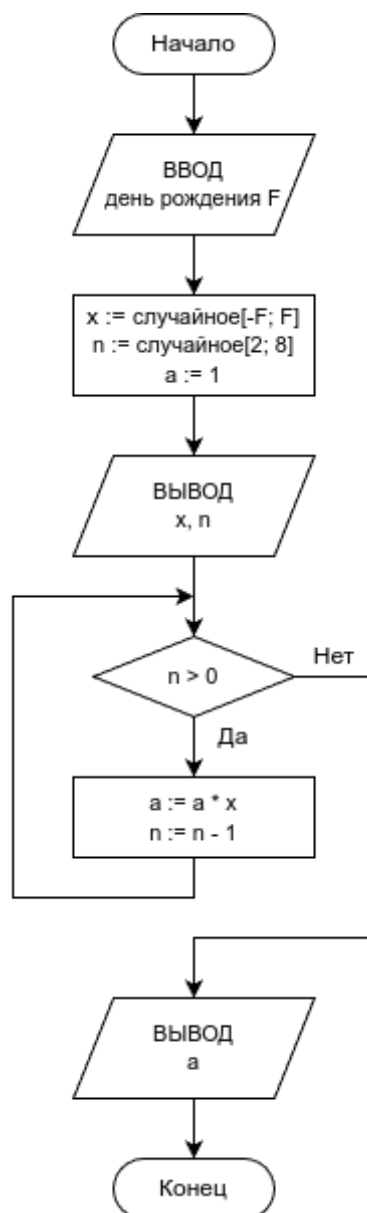
Во втором способе используем цикл с предусловием. В качестве счетчика так же используем переменную n . В теле цикла итоговое произведение умножаем на значение x , а счетчик уменьшаем на 1. Пока

значение счетчика положительное, цикл выполняется. Выполнение цикла завершается, когда значение счетчика становится равным 0. Особенность данного способа в том, что количество итераций цикла зависит от счетчика. Например, если начальное значение счетчика будет равно 0, то тело цикла не выполнится ни разу.

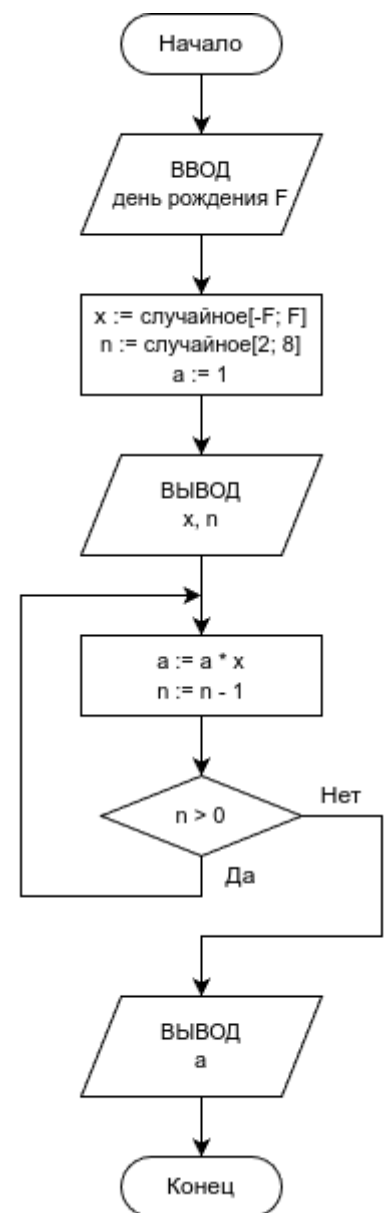
В третьем способе используем цикл с постусловием. В качестве счетчика опять же используем переменную n . В теле цикла итоговое произведение умножаем на значение x , а счетчик уменьшаем на 1. Пока значение счетчика положительное, цикл выполняется. Выполнение цикла завершается, когда значение счетчика становится равным 0. Особенность данного способа в том, что в независимости от значения счетчика тело цикла выполнится хотя бы один раз.



Способ 1



Способ 2



Способ 3

3. Составление исходного кода программы.

Способ 1.

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <stdlib>
#include <ctime>
#include <locale>

int main() {
    setlocale(LC_ALL, "Rus");

    int f, a{1}, n, x;
    printf("Введите дату рождения: ");
    scanf("%d", &f);

    std::srand(std::time(nullptr));

    x = (std::rand() % (2 * f + 1)) - f;
    n = (std::rand() % 7) + 2;

    printf("Случайный x = %d", x);
    printf("\nСлучайное n = %d", n);
    printf("\n%d^%d = ", x, n);

    for (n; n > 0; n--) {
        a *= x;
    }

    printf("%d", a);

    return 0;
}
```

Данный код объявляет переменные f (дата рождения), x и n , a (итоговое произведение, т. е. результат возведения в степень), и инициализирует переменную a значением 1. Далее выводится приглашения для ввода даты рождения и запись введенного значения в переменную f . Далее, генерацией случайных чисел задаются значения для переменных x и n ; выводится сообщение « x^n », где вместо x и n выводятся их текущие значения. Далее выполняется цикл с параметром, в качестве которого используется переменная n . Параметр на каждой итерации декрементируется. Цикл завершается, когда значение параметра становится равным либо меньшим 0. В теле цикла выполняется умножение итогового произведения на значение x . Перед завершением тела программы выводится значение итогового произведения, т. е. результат возведения в степень.

Способ 2.

```
while (n > 0) {
    a *= x;
    n--;
}
```

Данный код аналогичен коду в способе 1 за исключением цикла, поэтому приведен только код цикла. Цикл выполняется с предусловием, в качестве счетчика используется переменная n . В предусловии счетчик сравнивается с 0: пока значение счетчика больше 0 цикл выполняется. В теле цикла выполняется умножение итогового произведения на значение x , а так же декремент счетчика.

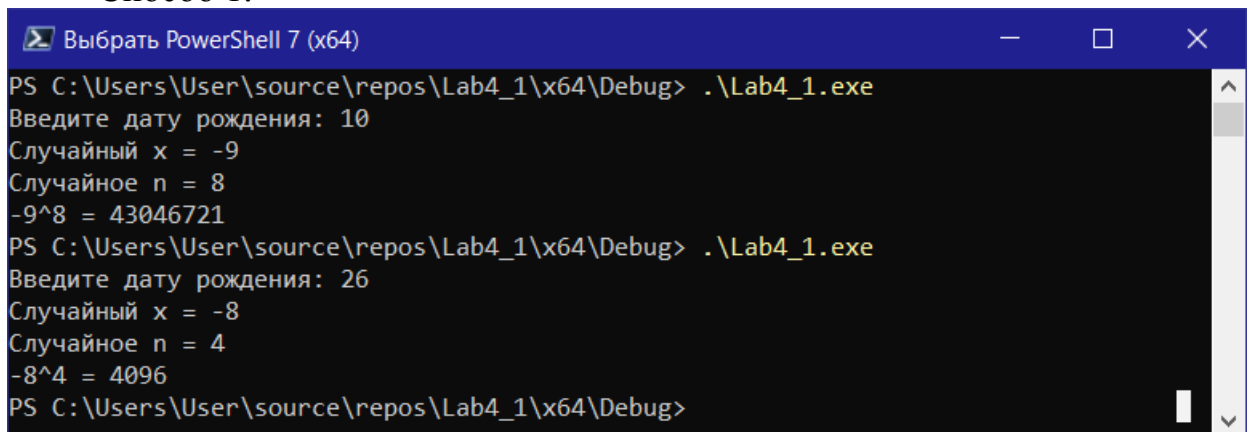
Способ 3.

```
do {  
    a *= x;  
    n--;  
} while (n > 0);
```

Данный код так же аналогичен коду в способе 1 за исключением цикла. Цикл выполняется с постусловием, в качестве счетчика используется переменная n . В постусловии счетчик сравнивается с 0: пока значение счетчика больше 0 цикл выполняется. В теле цикла выполняется умножение итогового произведения на значение x , а так же декремент счетчика.

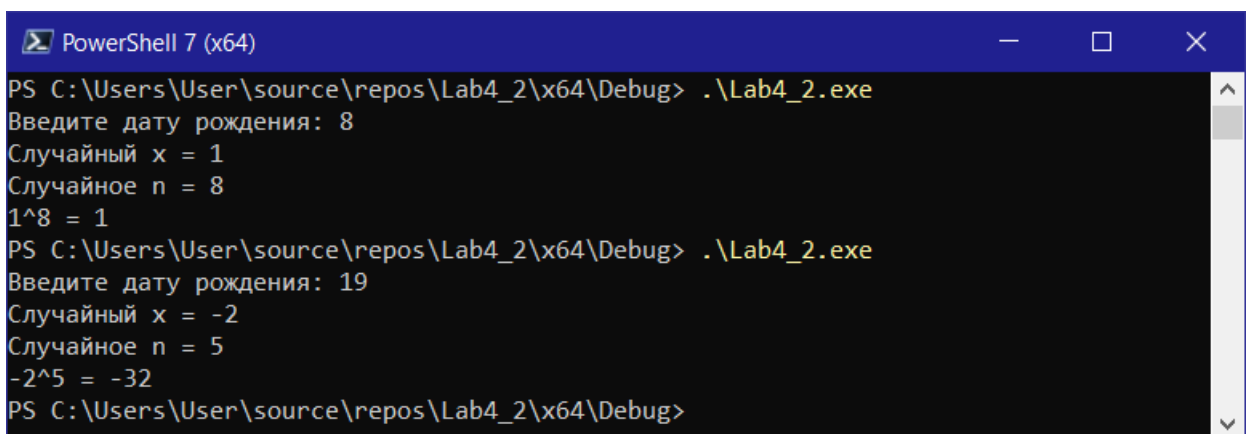
4. Отладить программу. Результаты работы программы показать преподавателю.

Способ 1.



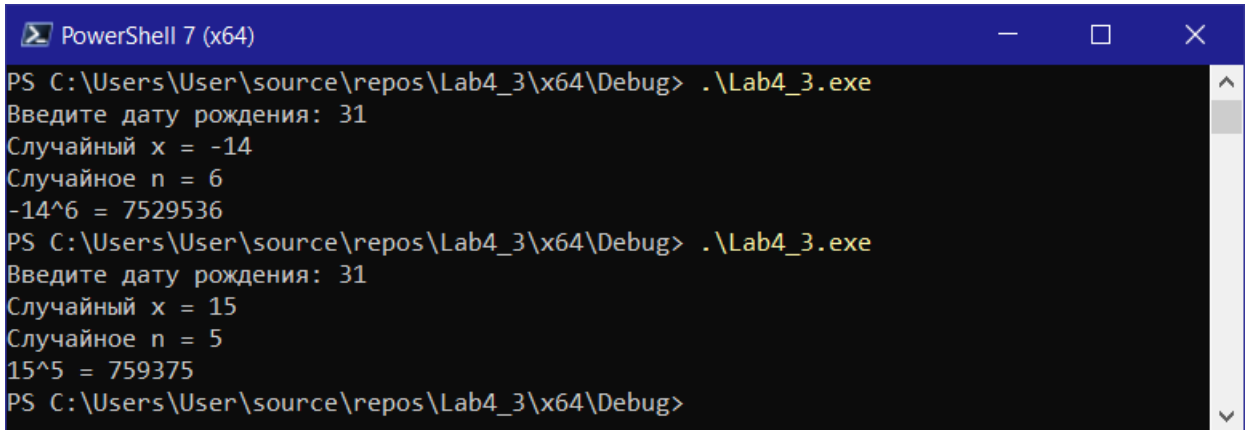
```
Выбрать PowerShell 7 (x64)  
PS C:\Users\User\source\repos\Lab4_1\x64\Debug> .\Lab4_1.exe  
Введите дату рождения: 10  
Случайный x = -9  
Случайное n = 8  
-9^8 = 43046721  
PS C:\Users\User\source\repos\Lab4_1\x64\Debug> .\Lab4_1.exe  
Введите дату рождения: 26  
Случайный x = -8  
Случайное n = 4  
-8^4 = 4096  
PS C:\Users\User\source\repos\Lab4_1\x64\Debug>
```

Способ 2.



```
PowerShell 7 (x64)  
PS C:\Users\User\source\repos\Lab4_2\x64\Debug> .\Lab4_2.exe  
Введите дату рождения: 8  
Случайный x = 1  
Случайное n = 8  
1^8 = 1  
PS C:\Users\User\source\repos\Lab4_2\x64\Debug> .\Lab4_2.exe  
Введите дату рождения: 19  
Случайный x = -2  
Случайное n = 5  
-2^5 = -32  
PS C:\Users\User\source\repos\Lab4_2\x64\Debug>
```

Способ 3.



```
PowerShell 7 (x64)
PS C:\Users\User\source\repos\Lab4_3\x64\Debug> .\Lab4_3.exe
Введите дату рождения: 31
Случайный x = -14
Случайное n = 6
-14^6 = 7529536
PS C:\Users\User\source\repos\Lab4_3\x64\Debug> .\Lab4_3.exe
Введите дату рождения: 31
Случайный x = 15
Случайное n = 5
15^5 = 759375
PS C:\Users\User\source\repos\Lab4_3\x64\Debug>
```

При выполнении программ видно, что они работают корректно: генерированные случайные значения в пределах заданного отрезка, итоговые значения вычислены верно.

5. Оформлен отчет с указанием номера и названия лабораторной работы, ее целей и задач. Разработаны схемы выполнения программы тремя способами. В соответствии со схемами написан исходный код. Сборка и запуск программ произведена без ошибок; во время выполнения программ ошибки не возникали.

Контрольные вопросы.

1. Какой оператор цикла является циклом с параметром?
Оператором цикла с параметром является цикл `for`.
2. Какой оператор цикла является циклом с предусловием?
Оператором цикла с постусловием является цикл `while`.
3. Какой оператор цикла является циклом с постусловием?
Оператором цикла с постусловием является цикл `do-while`.
4. Каков формат цикла `for`?
`for` (инициализация; проверка-условия; коррекция) оператор;
5. В каком случае выполняется тело цикла `for`?
Тело цикла `for` выполняется только тогда, когда условие продолжения цикла истинно. Цикл `for` может не выполнить ни одной итерации.
6. В каком случае выполняется тело цикла `while`?
Тело цикла `while` выполняется только тогда, когда предусловие цикла истинно. Цикл `while` может не выполнить ни одной итерации.
7. В каком случае выполняется тело цикла `do-while`?
Тело цикла `do-while` выполняется несколько раз когда постусловие цикла истинно или один раз когда постусловие ложно. Цикл `do-while` выполняется хотя бы 1 раз в независимости от истинности постусловия.

Лабораторная работа № 5.

Название работы: Язык C/C++. Массивы и указатели.

Цели работы: Освоение навыков работы с массивами в языке C/C++.

Задание:

Написать программу в соответствии с вариантом задания.

Варианты задания:

Определить и вывести на экран элемент массива:

- если дата рождения студента – четное число, то минимального;
- если дата рождения студента – нечетное число, то максимального.

Размерность массива определяется так:

- если дата рождения студента – четное число, то массив содержит 4 столбца и 5 строк;
- если дата рождения студента – нечетное число, то массив содержит 5 столбцов и 4 строки.

Элементы массива – натуральные числа, причем:

- если дата рождения студента – четное число, то в массиве исключены числа 2 и 8;
- если дата рождения студента – нечетное число, то в массиве исключены числа 3 и 7.

Иные числа в массиве произвольные.

Выполнение лабораторной работы.

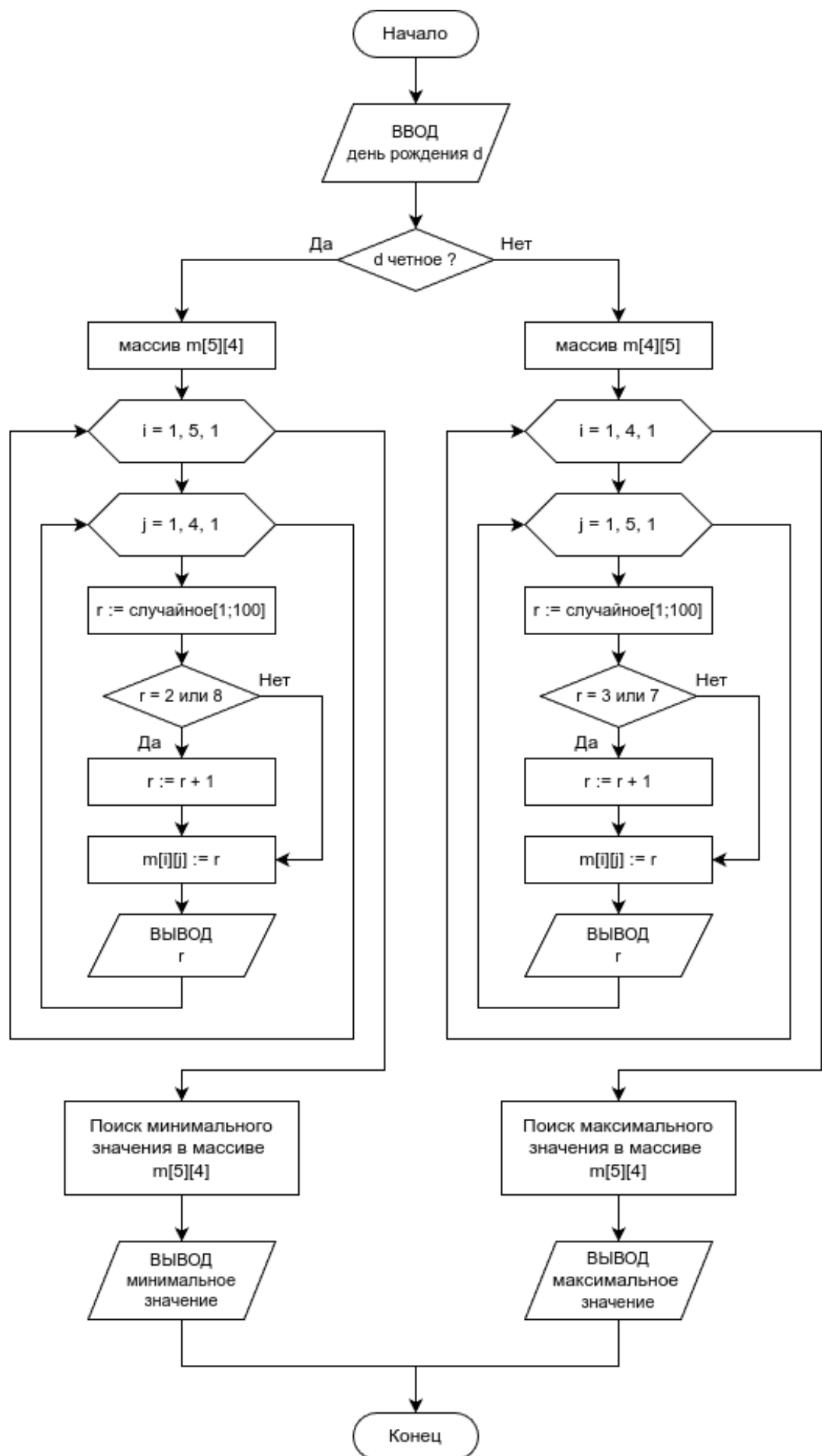
Порядок выполнения:

1. Изучить теоретические положения;
2. Разработать схему программы;
3. Составить программу;
4. Отладить программу. Результаты работы программы показать преподавателю;
5. Оформить отчет.

1. Изучены теоретические положения работы.

2. Разработка схемы программы.

В ходе работы программа получает значение даты рождения и записывает его в переменную. Далее программа создает двумерный массив, заполняет его случайными целыми числами, выводит на экран и выполняет действия над элементами массива. В зависимости от четности даты рождения определяется размерность массива, исключение записи определенных значений в массив и тип действий.



3. Составление исходного кода программы.

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <cstdlib>
#include <ctime>
#include <clocale>
#include <algorithm>

int main() {
    setlocale(LC_ALL, "Rus");

    int d;
    printf("Введите дату рождения: ");
    scanf("%d", &d);

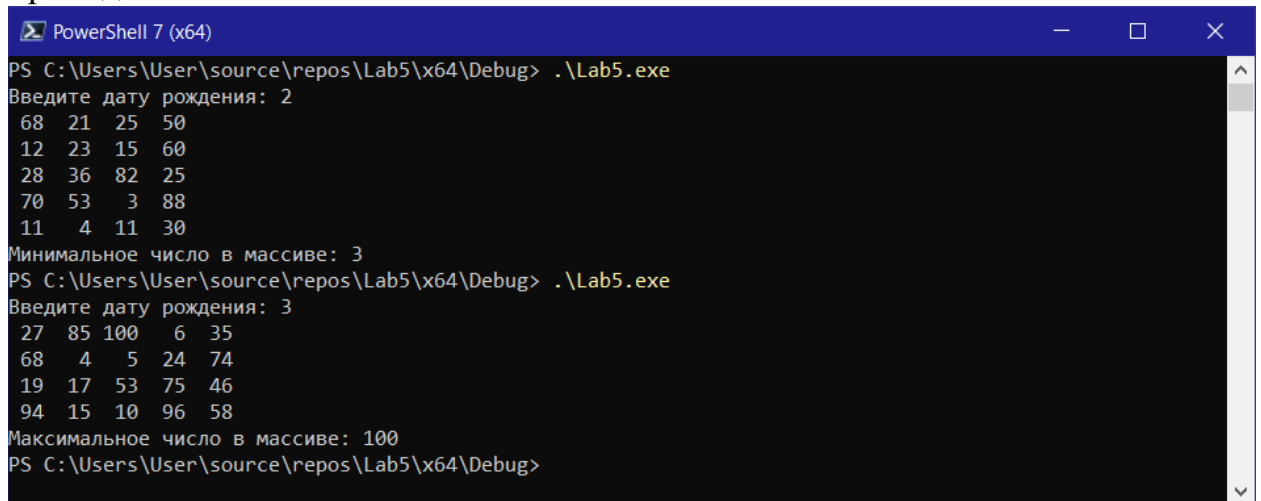
    std::srand(std::time(nullptr));

    if (d % 2 == 0) {
        // четное число - ищем минимум, размер 4 столб 5 строк,
        // число от 1 до 100 кроме 2 и 8
        int m[5][4];
        for (size_t i = 0; i < 5; i++) {
            for (size_t j = 0; j < 4; j++) {
                int r = 1 + std::rand() % 100;
                if (r == 2 || r == 8) r++;
                m[i][j] = r;
                if (r < 10) printf(" %d ", r);
                else if (r < 100) printf(" %d ", r);
                else printf("%d ", r);
            }
            printf("\n");
        }
        auto min_val = *std::min_element(&m[0][0], &m[0][0] + 20);
        printf("Минимальное число в массиве: %d", min_val);
    }
    else {
        // нечетное число - ищем максимум, размер 5 столб 4 строк,
        // число от 1 до 100 кроме 3 и 7
        int m[4][5];
        for (size_t i = 0; i < 4; i++) {
            for (size_t j = 0; j < 5; j++) {
                int r = 1 + std::rand() % 100;
                if (r == 3 || r == 7) r++;
                m[i][j] = r;
                if (r < 10) printf(" %d ", r);
                else if (r < 100) printf(" %d ", r);
                else printf("%d ", r);
            }
            printf("\n");
        }
        auto max_val = *std::max_element(&m[0][0], &m[0][0] + 20);
        printf("Максимальное число в массиве: %d", max_val);
    }

    return 0;
}
```

Данный код выводит приглашение для ввода даты рождения и записывает введенное значение в переменную *d*. Далее, если значение даты четное объявляется массив 5×4 , запускается цикл с параметром по строкам (5), внутри вложенный цикл с параметром по столбцам (4), в теле цикла по столбцам генерируется случайное значение от 1 до 100, сравнивается с 2 и 8, при совпадении значение инкрементируется, записывается в массив по текущим индексам и выводится на экран. После прохода обоих циклов определяется минимальное значение в массиве функцией `min_element` из заголовочного файла `algorithm`. Минимальное значение выводится на экран и программа завершает выполнение. При вводе даты с нечетным значением производятся аналогичные действия со следующими отличиями: объявляется массив 4×5 ; случайное значение сравнивается с 3 и 7; определяется и выводится максимальное значение в массиве функцией `max_element` из заголовочного файла `algorithm`.

4. Отладить программу. Результаты работы программы показать преподавателю.



```
PowerShell 7 (x64)
PS C:\Users\User\source\repos\Lab5\x64\Debug> .\Lab5.exe
Введите дату рождения: 2
68 21 25 50
12 23 15 60
28 36 82 25
70 53 3 88
11 4 11 30
Минимальное число в массиве: 3
PS C:\Users\User\source\repos\Lab5\x64\Debug> .\Lab5.exe
Введите дату рождения: 3
27 85 100 6 35
68 4 5 24 74
19 17 53 75 46
94 15 10 96 58
Максимальное число в массиве: 100
PS C:\Users\User\source\repos\Lab5\x64\Debug>
```

При выполнении программы видно, что она работает корректно: четность введенного значения определена верно, генерированные случайные значения в пределах заданного отрезка, итоговые значения определены верно, размерность массива соблюдена.

5. Оформлен отчет с указанием номера и названия лабораторной работы, ее целей и задач. Разработана схема выполнения программы. В соответствии со схемой написан исходный код. Сборка и запуск программы произведена без ошибок; во время выполнения программы ошибки не возникали.

Контрольные вопросы.

1. В чем особенность объявления и инициализации массива?

Массив объявляется с фиксированным размером, который должен быть известен на этапе компиляции (константа или значение). Объявление можно совмещать с инициализацией или проводить инициализацию отдельно, после объявления.

2. С какого значения начинаются индексы в массивах?

В языке C индексы массива всегда начинаются с 0.

Лабораторная работа № 6.

Название работы: Язык C/C++. Функции.

Цели работы: Получение навыков написания и использования функций.

Задание:

Составить программу с использованием функции, разработанной самостоятельно в соответствии с вариантом задания. Варианты задания:

- если дата рождения студента – четное число, то функция содержит результат деления даты рождения студента на месяц рождения;
- если дата рождения студента – нечетное число, то функция содержит результат умножения месяца рождения студента на дату его рождения.

Выполнение лабораторной работы.

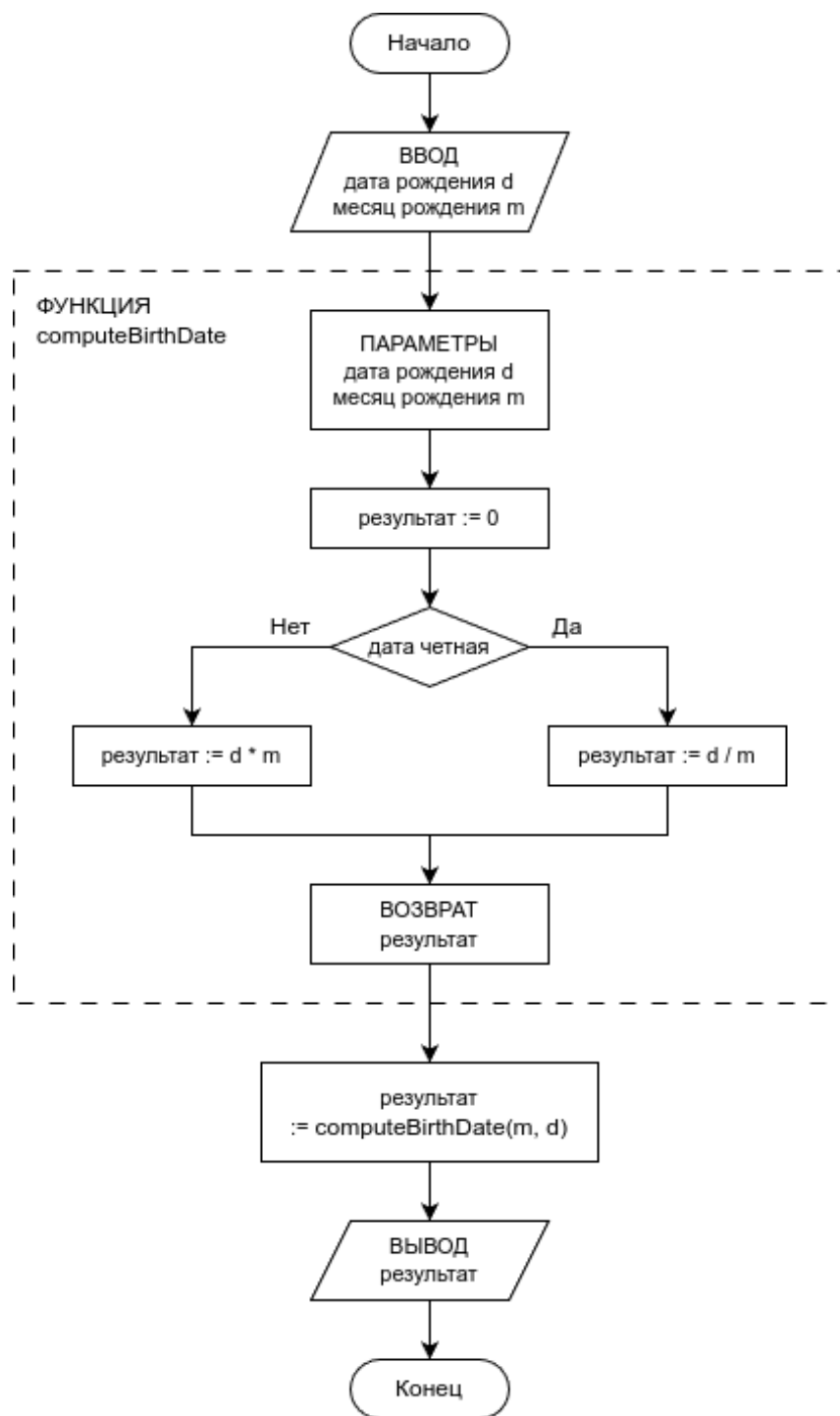
Порядок выполнения:

1. Изучить теоретические положения;
2. Разработать схему программы;
3. Составить программу;
4. Отладить программу. Результаты работы программы показать преподавателю;
5. Оформить отчет.

1. Изучены теоретические положения работы.

2. Разработка схемы программы.

В ходе работы программа получает значения даты и месяца рождения и записывает их в переменные. Далее программа передает полученные значения в функцию. В случае если дата рождения является четным числом функция возвращает результат деления даты на месяц; в случае нечетной даты рождения функция возвращает произведение даты на месяц. Полученное значение из функции выводится на экран.



3. Составление исходного кода программы.

```
#define _CRT_SECURE_NO_WARNINGS
#include <stdio.h>
#include <locale>

double computeBirthDate(int date, int month);

int main() {
    setlocale(LC_ALL, "Rus");

    int d, m;
    printf("Введите дату рождения: ");
    scanf("%d", &d);
    printf("Введите месяц рождения: ");
    scanf("%d", &m);

    auto result = computeBirthDate(d, m);
    printf("Рассчитанное значение: %.2f", result);

    return 0;
}

double computeBirthDate(int date, int month) {
    double result{ 0 };

    if (date % 2 == 0) result = double(date) / double(month);
    else result = double(date) * double(month);

    return result;
}
```

Данный код выполняет следующие действия:

- объявляет функцию `computeBirthDate(int date, int month)` для действий с днем и месяцем рождения;
- в функции `main()` выводятся приглашения ввода данных и записываются введенные значения в соответствующие переменные. Далее вызывается функция `computeBirthDate`, которой передаются полученные данные. Функция `computeBirthDate` возвращает результат типа `double`, который выводится на экран;
- определяет функцию `computeBirthDate(int date, int month)`. В случае если переменная `date` имеет четное значение, функция возвращает результат деления переменных `date` на `month`; в случае нечетного значения функция возвращает произведение `date` на `month`.

4. Отладить программу. Результаты работы программы показать преподавателю.

При выполнении программы видно, что она работает корректно: четность введенного значения определена верно, итоговые значения вычислены верно.

```
PowerShell 7 (x64)
PS C:\Users\User\source\repos\Lab6\x64\Debug> .\Lab6.exe
Введите дату рождения: 10
Введите месяц рождения: 7
Рассчитанное значение: 1,43
PS C:\Users\User\source\repos\Lab6\x64\Debug> .\Lab6.exe
Введите дату рождения: 9
Введите месяц рождения: 7
Рассчитанное значение: 63,00
PS C:\Users\User\source\repos\Lab6\x64\Debug>
```

5. Оформлен отчет с указанием номера и названия лабораторной работы, ее целей и задач. Разработана схема выполнения программы. В соответствии со схемой написан исходный код. Сборка и запуск программы произведена без ошибок; во время выполнения программы ошибки не возникали.

Контрольные вопросы.

1. В каком месте программы объявляются (описываются) функции?
Прототипы могут располагаться либо в самой программе на С, либо в заголовочном файле. Обычно функции объявляются в глобальной области видимости до функции `main()` и до их первого использования.
2. Каков формат записи прототипов функции?
Возможно использовать следующий стиль записи прототипов:
возвращаемый-тип имя-функции(тип-аргумента имя-аргумента);
3. В каком месте программы определяются функции?
Функции в программе могут помещаться в любом порядке, они считаются глобальными для всей программы, включая функции, определенные раньше текстуально.
4. Как определяется функция?
Функция может иметь вид:
возвращаемый-тип имя-функции (тип-арг1 имя-арг1,..., тип-аргN имя-аргN)
{
.
/* объявления данных и тело функции) */
.
return();
}
5. Каким образом определяется тип функции?
Тип функции определяется по типу возвращаемого ей значения.
6. Каким образом осуществляется возврат из функции?
Возврат из функции осуществляется при помощи оператора `return()`;

Лабораторная работа № 7.

Название работы: Создание оконных приложений.

Цели работы: Научиться создавать оконные приложения с помощью конструктора форм WindowsForms.

Задание:

Составить программу в соответствии с вариантом задания. Варианты задания:

– Если дата рождения студента заключена в интервал от 1 до 10, то программа должна иметь оконный интерфейс и должна определять количество элементов в произвольном двумерном массиве. Размер массива – $n \times n$, где n – натуральное число в интервале от 5 до 10.

– Если дата рождения студента заключена в интервал от 11 до 20, то программа должна иметь оконный интерфейс и должна определять сумму элементов в произвольном двумерном массиве. Размер массива – $n \times n$, где n – натуральное число в интервале от 5 до 10.

– Если дата рождения студента заключена в интервал от 21 до 31, то программа должна иметь оконный интерфейс и должна определять произведение отрицательных элементов в двумерном массиве. Размер массива – $n \times n$, где n – натуральное число в интервале от 5 до 10.

Выполнение лабораторной работы.

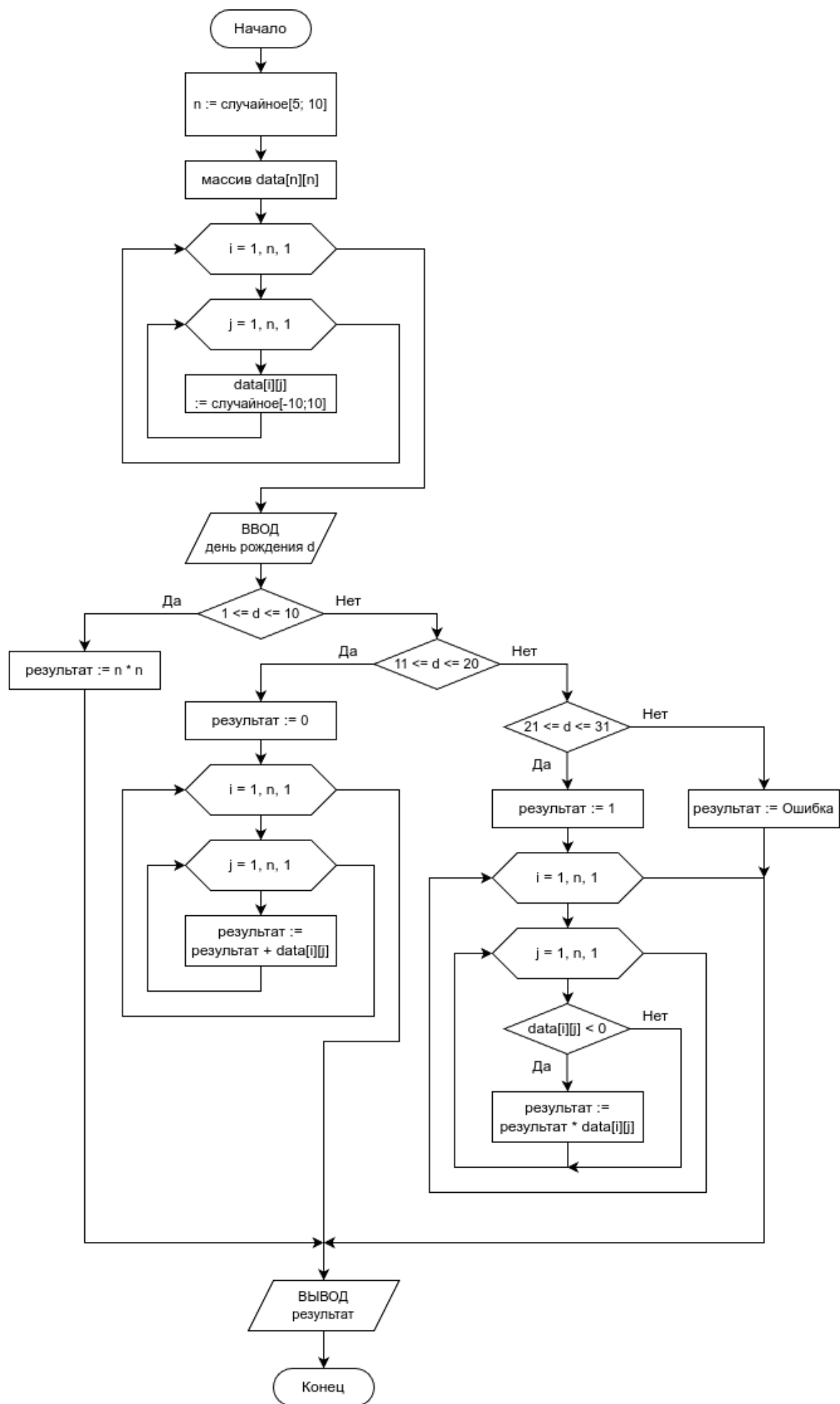
Порядок выполнения:

1. Изучить теоретические положения;
2. Разработать схему программы;
3. Составить программу;
4. Отладить программу. Результаты работы программы показать преподавателю;
5. Оформить отчет.

1. Изучены теоретические положения работы.

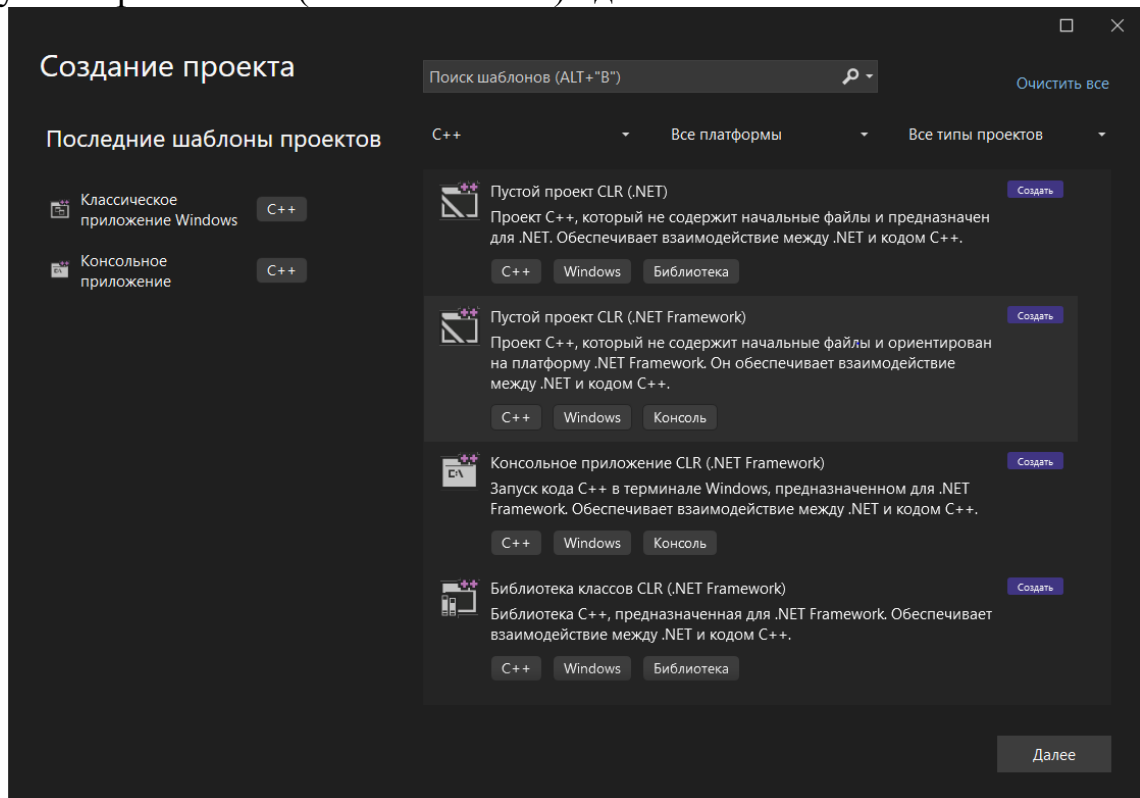
2. Разработка схемы программы.

В ходе работы программа генерирует случайное число n в диапазоне от 5 до 10, определяющее размерность двумерного массива. Далее генерируется массив размерностью $n \times n$, заполненный случайными числами от -10 до 10. Массив выводится на экран. Далее программа получает значение даты рождения и определяет интервал, в котором находится дата (от 1 до 10, от 11 до 20 или от 21 до 31). В случае если дата попадает в интервал от 1 до 10, вычисляется количество элементов массива; если дата попадает в интервал от 11 до 20, вычисляется сумма элементов массива; если дата попадает в интервал от 21 до 31, вычисляется произведение отрицательных элементов массива. Полученное значение выводится на экран.

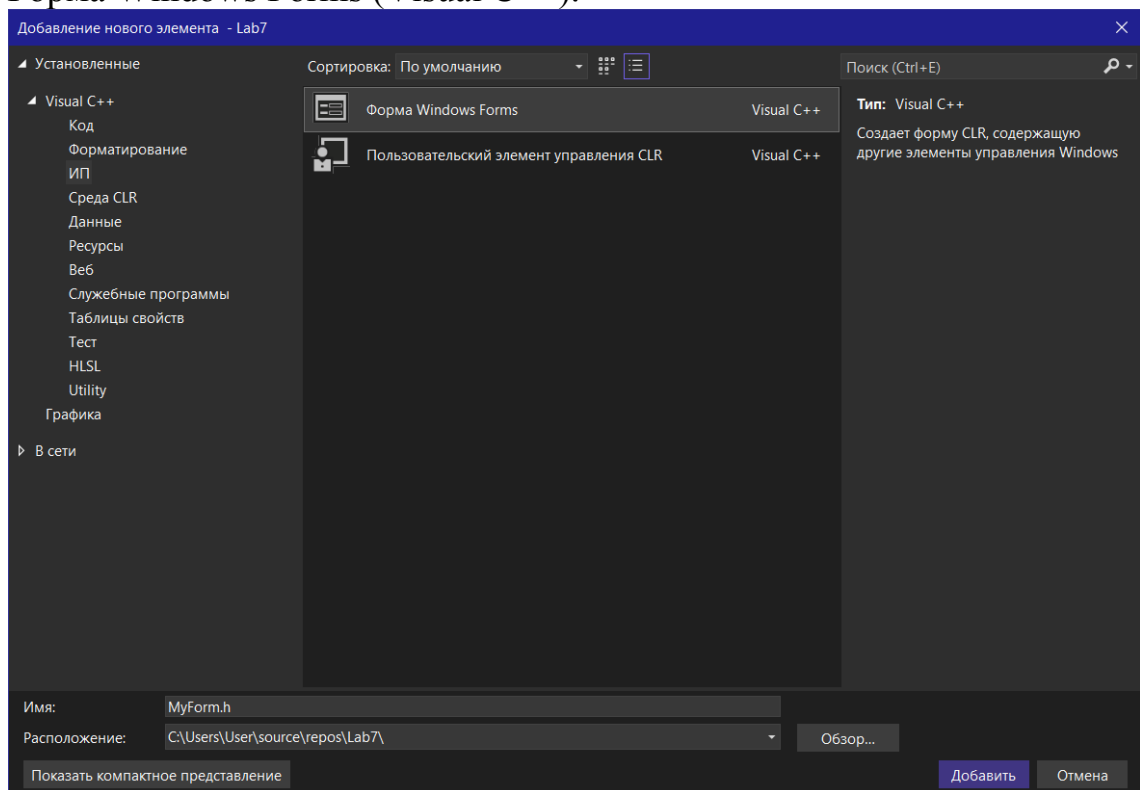


3. Составление исходного кода программы.

Для создания оконного приложения создаем новый проект. Выбираем «Пустой проект CLR (.Net Framework)» для C++.



Для добавления формы выбираем в главном меню Проект → Добавить новый элемент... В появившемся диалоге добавления элемента выбираем ИП → Форма Windows Forms (Visual C++).



В файл MyForm.cpp пишем следующий код:

```

#include "MyForm.h"

#include <Windows.h>

using namespace Lab7;

int WINAPI WinMain(HINSTANCE, HINSTANCE, LPSTR, int) {
    Application::EnableVisualStyles();
    Application::SetCompatibleTextRenderingDefault(false);
    Application::Run(gcnew MyForm);
    return 0;
}

```

В конструкторе форм редактируем файл MyForm.h: накидываем элементы на форму: 2 кнопки Button, 11 полей Label, поле ввода TextBox, панель Panel. Располагаем элементы по своему вкусу. Далее добавляем необходимые функционал вручную.

```

#pragma once
#include <vector>
#include <ctime>

namespace Lab7 {
    using namespace System;
    using namespace System::ComponentModel;
    using namespace System::Collections;
    using namespace System::Windows::Forms;
    using namespace System::Data;
    using namespace System::Drawing;

    public ref class MyForm : public System::Windows::Forms::Form {
    public:
        MyForm(void) {
            InitializeComponent();

            generate_data();
            print_array();
        }

    protected:
        ~MyForm() {
            if (components) {
                delete components;
            }
            delete data;
        }

    private: System::Windows::Forms::TextBox^ textBox1;
    private: System::Windows::Forms::Button^ button1;
    private: System::Windows::Forms::Panel^ panel1;
    private: System::Windows::Forms::Label^ label10;
    private: System::Windows::Forms::Label^ label9;
    private: System::Windows::Forms::Label^ label8;
    private: System::Windows::Forms::Label^ label7;
    private: System::Windows::Forms::Label^ label6;
    private: System::Windows::Forms::Label^ label5;
    private: System::Windows::Forms::Label^ label4;
}

```

```

private: System::Windows::Forms::Label^ label3;
private: System::Windows::Forms::Label^ label2;
private: System::Windows::Forms::Label^ label1;
private: System::Windows::Forms::Label^ label11;
private: int size;
private: std::vector<std::vector<int>*> *data;
private: System::Windows::Forms::Button^ button2;
private: System::ComponentModel::Container^ components;

#pragma region Windows Form Designer generated code
void InitializeComponent(void){
    this->textBox1 = (gcnew System::Windows::Forms::TextBox());
    this->button1 = (gcnew System::Windows::Forms::Button());
    this->panel1 = (gcnew System::Windows::Forms::Panel());
    this->label10 = (gcnew System::Windows::Forms::Label());
    this->label9 = (gcnew System::Windows::Forms::Label());
    this->label8 = (gcnew System::Windows::Forms::Label());
    this->label7 = (gcnew System::Windows::Forms::Label());
    this->label6 = (gcnew System::Windows::Forms::Label());
    this->label5 = (gcnew System::Windows::Forms::Label());
    this->label4 = (gcnew System::Windows::Forms::Label());
    this->label3 = (gcnew System::Windows::Forms::Label());
    this->label2 = (gcnew System::Windows::Forms::Label());
    this->label1 = (gcnew System::Windows::Forms::Label());
    this->label11 = (gcnew System::Windows::Forms::Label());
    this->button2 = (gcnew System::Windows::Forms::Button());
    this->panel1->SuspendLayout();
    this->SuspendLayout();
    // textBox1
    this->textBox1->Location = System::Drawing::Point(12, 12);
    this->textBox1->Name = L"textBox1";
    this->textBox1->Size = System::Drawing::Size(404, 30);
    this->textBox1->TabIndex = 0;
    this->textBox1->Text = L"1";
    // button1
    this->button1->Location = System::Drawing::Point(422, 12);
    this->button1->Name = L"button1";
    this->button1->Size = System::Drawing::Size(140, 30);
    this->button1->TabIndex = 1;
    this->button1->Text = L"Рассчитать";
    this->button1->UseVisualStyleBackColor = true;
    this->button1->Click += gcnew System::EventHandler(this,
&MyForm::button1_Click);
    // panel1
    this->panel1->Controls->Add(this->label10);
    this->panel1->Controls->Add(this->label9);
    this->panel1->Controls->Add(this->label8);
    this->panel1->Controls->Add(this->label7);
    this->panel1->Controls->Add(this->label6);
    this->panel1->Controls->Add(this->label5);
    this->panel1->Controls->Add(this->label4);
    this->panel1->Controls->Add(this->label3);
    this->panel1->Controls->Add(this->label2);
    this->panel1->Controls->Add(this->label1);

```

```

        this->panel1->Font = (gcnew System::Drawing::Font(L"Cascadia
Mono", 12, System::Drawing::FontStyle::Regular,
System::Drawing::GraphicsUnit::Point, static_cast<System::Byte>(204)));
        this->panel1->Location = System::Drawing::Point(12, 48);
        this->panel1->Name = L"panel1";
        this->panel1->Size = System::Drawing::Size(550, 277);
        this->panel1->TabIndex = 2;
        // label10
        this->label10->AutoSize = true;
        this->label10->Location = System::Drawing::Point(3, 246);
        this->label10->Name = L"label10";
        this->label10->Size = System::Drawing::Size(0, 27);
        this->label10->TabIndex = 9;
        // label9
        this->label9->AutoSize = true;
        this->label9->Location = System::Drawing::Point(3, 219);
        this->label9->Name = L"label9";
        this->label9->Size = System::Drawing::Size(0, 27);
        this->label9->TabIndex = 8;
        // label8
        this->label8->AutoSize = true;
        this->label8->Location = System::Drawing::Point(3, 192);
        this->label8->Name = L"label8";
        this->label8->Size = System::Drawing::Size(0, 27);
        this->label8->TabIndex = 7;
        // label7
        this->label7->AutoSize = true;
        this->label7->Location = System::Drawing::Point(3, 165);
        this->label7->Name = L"label7";
        this->label7->Size = System::Drawing::Size(0, 27);
        this->label7->TabIndex = 6;
        // label6
        this->label6->AutoSize = true;
        this->label6->Location = System::Drawing::Point(3, 138);
        this->label6->Name = L"label6";
        this->label6->Size = System::Drawing::Size(0, 27);
        this->label6->TabIndex = 5;
        // label5
        this->label5->AutoSize = true;
        this->label5->Location = System::Drawing::Point(3, 111);
        this->label5->Name = L"label5";
        this->label5->Size = System::Drawing::Size(0, 27);
        this->label5->TabIndex = 4;
        // label4
        this->label4->AutoSize = true;
        this->label4->Location = System::Drawing::Point(3, 84);
        this->label4->Name = L"label4";
        this->label4->Size = System::Drawing::Size(0, 27);
        this->label4->TabIndex = 3;
        // label3
        this->label3->AutoSize = true;
        this->label3->Location = System::Drawing::Point(3, 57);
        this->label3->Name = L"label3";
        this->label3->Size = System::Drawing::Size(0, 27);
        this->label3->TabIndex = 2;
        // label2

```

```

        this->label2->AutoSize = true;
        this->label2->Location = System::Drawing::Point(3, 30);
        this->label2->Name = L"label2";
        this->label2->Size = System::Drawing::Size(0, 27);
        this->label2->TabIndex = 1;
        // label1
        this->label1->AutoSize = true;
        this->label1->Location = System::Drawing::Point(3, 3);
        this->label1->Name = L"label1";
        this->label1->Size = System::Drawing::Size(0, 27);
        this->label1->TabIndex = 0;
        // label11
        this->label11->AutoSize = true;
        this->label11->Font = (gcnew System::Drawing::Font(L"Microsoft
        Sans Serif", 12, System::Drawing::FontStyle::Bold,
        System::Drawing::GraphicsUnit::Point, static_cast<System::Byte>(204)));
        this->label11->ForeColor =
        System::Drawing::Color::FromArgb(static_cast<System::Int32>(static_cast<System::Byte>(192)), static_cast<System::Int32>(static_cast<System::Byte>(0)),
        static_cast<System::Int32>(static_cast<System::Byte>(0)));
        this->label11->Location = System::Drawing::Point(7, 328);
        this->label11->Name = L"label11";
        this->label11->Size = System::Drawing::Size(0, 25);
        this->label11->TabIndex = 3;
        // button2
        this->button2->Location = System::Drawing::Point(422, 332);
        this->button2->Name = L"button2";
        this->button2->Size = System::Drawing::Size(139, 43);
        this->button2->TabIndex = 4;
        this->button2->Text = L"Генерация";
        this->button2->UseVisualStyleBackColor = true;
        this->button2->Click += gcnew System::EventHandler(this,
        &MyForm::button2_Click);
        // MyForm
        this->AutoScaleDimensions = System::Drawing::SizeF(12, 25);
        this->AutoScaleMode =
        System::Windows::Forms::AutoScaleMode::Font;
        this->ClientSize = System::Drawing::Size(574, 387);
        this->Controls->Add(this->button2);
        this->Controls->Add(this->label11);
        this->Controls->Add(this->panel1);
        this->Controls->Add(this->button1);
        this->Controls->Add(this->textBox1);
        this->Font = (gcnew System::Drawing::Font(L"Microsoft Sans
        Serif", 12, System::Drawing::FontStyle::Regular,
        System::Drawing::GraphicsUnit::Point, static_cast<System::Byte>(204)));
        this->Margin = System::Windows::Forms::Padding(4, 5, 4, 5);
        this->Name = L"MyForm";
        this->Text = L"Лабораторная работа № 7";
        this->panel1->ResumeLayout(false);
        this->panel1->PerformLayout();
        this->ResumeLayout(false);
        this->PerformLayout();
    }
#pragma endregion

```

```

        private: System::Void button1_Click(System::Object^ sender,
System::EventArgs^ e) {
            int d;
            if (Int32::TryParse(this->textBox1->Text, d)) {
                this->label11->Text = "" + d;
                if (d >= 1 && d <= 10) {
                    this->label11->Text = "Количество элементов равно " + size
* size;
                }
                else if (d > 10 && d <= 20) {
                    this->label11->Text = "Сумма элементов равна " +
data_sum();
                }
                else if (d > 20 && d <= 31) {
                    this->label11->Text = "Произведение отрицательных \
nэлементов равно " + data_prod();
                }
                else if (d > 31) {
                    this->label11->Text = "Ошибка ввода!!";
                    return;
                }
            } else {
                this->label11->Text = "Ошибка ввода!!";
            }
        }

        private: void generate_data() {
            std::srand(std::time(nullptr));
            size = 5 + std::rand() % 6;
            this->label11->Text = "" + size;

            data = new std::vector<std::vector<int>*>();

            for (size_t i = 0; i < size; i++) {
                auto row = new std::vector<int>();
                for (size_t j = 0; j < size; j++) {
                    row->push_back(std::rand() % 21 - 10);
                }
                data->push_back(row);
            }
        }

        private: void print_array() {
            clear();
            for (size_t i = 0; i < size; i++) {
                System::Windows::Forms::Label^ label;
                if (i == 0) label = this->label1;
                else if (i == 1) label = this->label2;
                else if (i == 2) label = this->label3;
                else if (i == 3) label = this->label4;
                else if (i == 4) label = this->label5;
                else if (i == 5) label = this->label6;
                else if (i == 6) label = this->label7;
                else if (i == 7) label = this->label8;
                else if (i == 8) label = this->label9;
            }
        }
    }
}

```

```

        else if (i == 9) label = this->label10;
        std::vector<int>* row = data->at(i);
        String ^text = "";
        for (size_t j = 0; j < size; j++) {
            int el = row->at(j);
            if (el >= 0 && el < 10) text += " " + el + " ";
            else if ((el >= -9 && el < 0) || (el > 9)) text += " " +
el + " ";
            else text += "" + el + " ";
        }
        label->Text = text;
    }
}

void clear() {
    this->label1->Text = "";
    this->label2->Text = "";
    this->label3->Text = "";
    this->label4->Text = "";
    this->label5->Text = "";
    this->label6->Text = "";
    this->label7->Text = "";
    this->label8->Text = "";
    this->label9->Text = "";
    this->label10->Text = "";
    this->label11->Text = "";
}

int data_sum() {
    int sum = 0;
    for (size_t i = 0; i < size; i++) {
        std::vector<int>* row = data->at(i);
        for (size_t j = 0; j < size; j++) {
            sum += row->at(j);
        }
    }
    return sum;
}

int data_prod() {
    int prod = 1;
    for (size_t i = 0; i < size; i++) {
        std::vector<int>* row = data->at(i);
        for (size_t j = 0; j < size; j++) {
            int el = row->at(j);
            if (el < 0) prod *= el;
        }
    }
    return prod;
}

private: System::Void button2_Click(System::Object^ sender,
System::EventArgs^ e) {
    generate_data();
    print_array();
}

```



```
};  
}
```

Функция `InitializeComponent()` создает компоненты интерфейса формы и задает их свойства и расположение на форме. На форме расположены: поле ввода текста для ввода даты рождения; кнопка `Расчитать` для расчета значения из массива; 10 текстовых полей для вывода строк массива; текстовое поле для вывода результата; кнопка `Генерация` для генерации нового массива.

В функции `generate_data()` выполняется генерация размерности массива и его элементов. Значения элементов массива сохраняются в переменной типа `std::vector<std::vector<int>*>*`.

Функция `print_array()` выводит значения элементов массива в 10 текстовых полей, каждая строка записывается в отдельное текстовое поле.

При нажатии кнопки `Расчитать` вызывается обработчик `button1_Click`, в котором определяется значение введенной даты, ее интервал и производятся соответствующие вычисления.

При нажатии кнопки `Генерация` вызывается обработчик `button2_Click`, в котором выполняются функции `generate_data()` и `print_array()`.

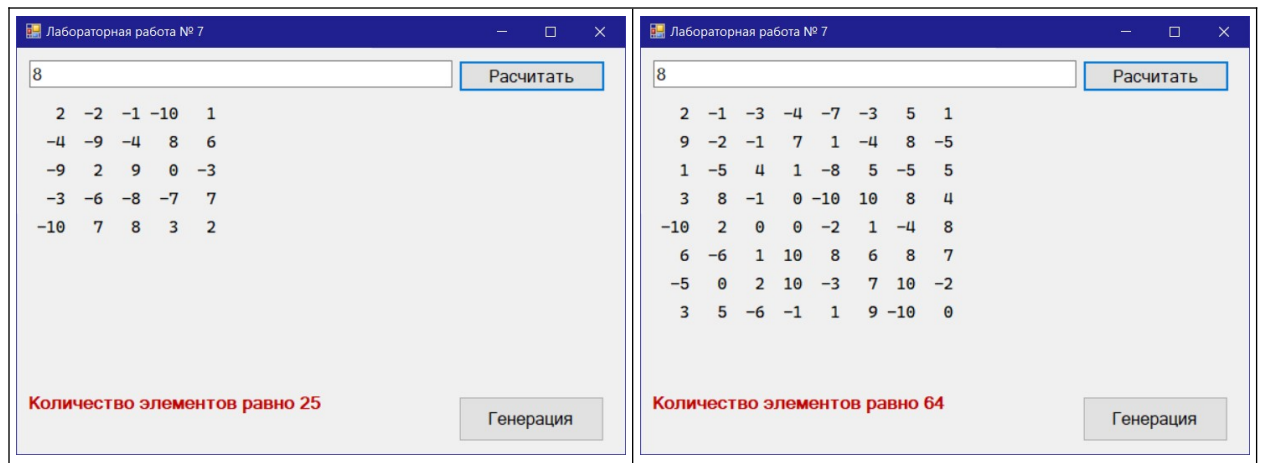
Функция `clear()` очищает 10 текстовых полей для отображения массива.

Функция `data_sum()` вычисляет и возвращает сумму элементов массива.

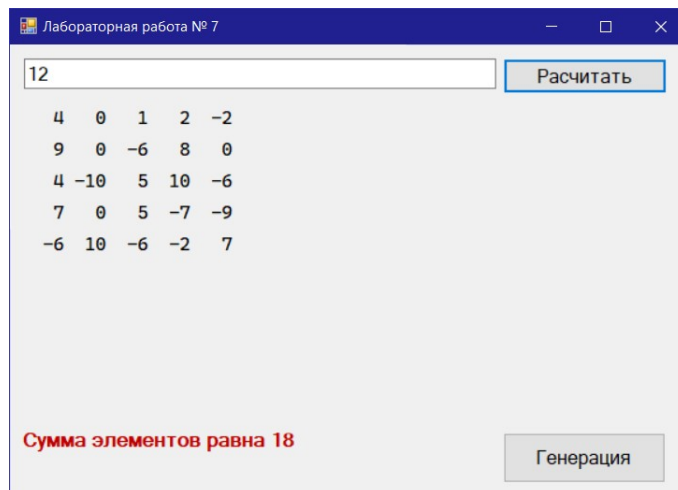
Функция `data_prod()` вычисляет и возвращает произведение отрицательных элементов массива.

4. Отладить программу. Результаты работы программы показать преподавателю.

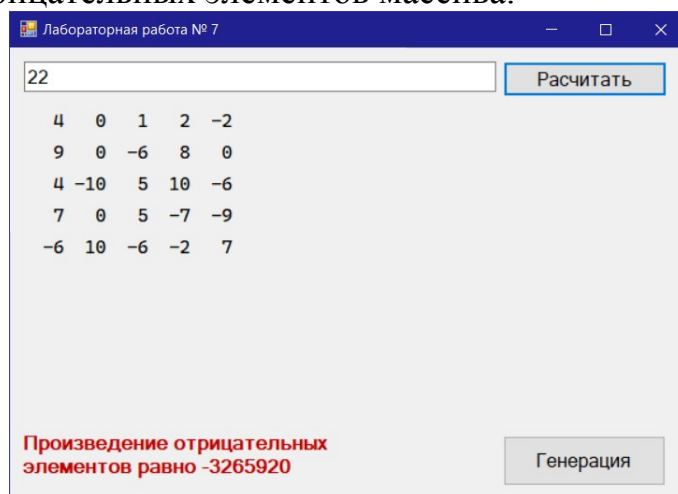
При значении даты рождения от 1 до 10 программа определяет количество элементов массива.



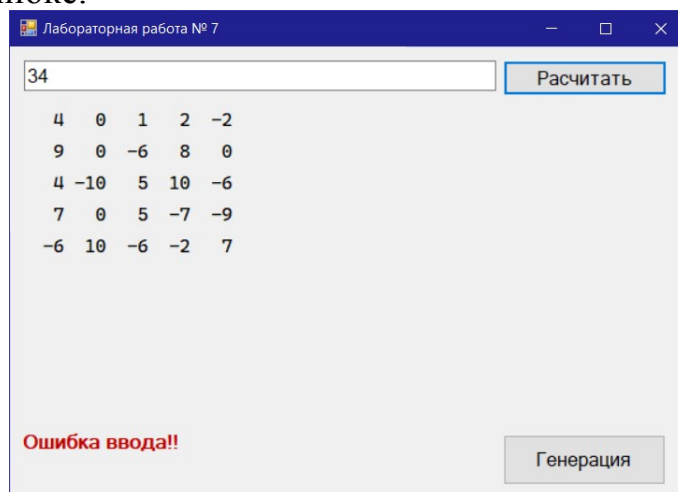
При значении даты рождения от 11 до 20 программа вычисляет сумму элементов массива.



При значении даты рождения от 21 до 31 программа вычисляет произведение отрицательных элементов массива.



При значении даты рождения больше 31 программа выводит сообщение об ошибке.



5. Оформлен отчет с указанием номера и названия лабораторной работы, ее целей и задач. Разработана схема выполнения программы. В соответствии со схемой написан исходный код. Сборка и запуск программы произведена без ошибок; во время выполнения программы ошибки не возникали.