УЧЕБНАЯ ПРОГРАММА
по дисциплине
«Программирование научных задач на языке C++»

А.А.Быков abkov.ru, boombook@yandex.ru

Данный курс предназначен для обучения студентов физического факультета МГУ программированию на языке C++. В основу курса положен принцип практического освоения каждого изучаемого аспекта языка C++. Результат освоения каждой темы – работающий код, написанный слушателем в соответствии с заданием. Форма отчетности – зачет или экзамен по каждой главе. Зачет получает студент, выполнивший все практические задания. Экзамен состоит из двух частей, теоретической и практической. Теоретическая часть содержит ориентировочно 10 вопросов по всем изучаемым темам, на каждый из которых дается письменный ответ. Состав вопросов теоретической части по пособию [1]. Практическая часть состоит в создании проекта в среде MS VS C++ по заданию экзаменатора. Первая часть практического курса использует парадигму консольного кодивания. Ввод и вывод осуществляется только через файл. Вторая часть курса использует интерактивное кодирование на базе MFC.

Основная литература (доступна в виде документов pdf):

[1] Дейтел Х., Дейтел П. Программирование на C++.
- ✓ ***Дейтел C++, §1.11,***

[2] Либерти Дж. Освой самостоятельно C++ за 21 день.
- ✓ ***Либерти C++21, глава 1.***

[3] Круглински Д., Уингоу С., Шефферд Дж. Программирование Microsoft Visual Studio.
- ✓ ***Круглински, Уингоу, Шефферд, глава 1.***

Дополнительная литература:

[4] Страуструп Б., Программирование на C++. Специальное издание.
- ✓ ***Страуструп C++, §1.11,***

[5] Янг М. Дж. Visual C++, полное руководство в 2-х т.
- ✓ ***Янг, Том I, глава 1,***

[6] Либерти Дж. Энциклопедия C++.
- ✓ ***Либерти C++ энциклопедия, глава 1.***

Глава 1 Основные понятия C++

Лекция 1. Основные понятия C++ и Microsoft Visual Studio

1.1 Список проектов Visual studio.

1.1.1 p1n01 Hello world
1.1.2 p1n02 Арпифметика
1.1.3 p1n03 Циклы
1.1.4 p2n06 Функции

1.2 Понятие Microsoft Visual studio.

- ✓ ***Круглински, Уингоу, Шефферд, глава 1.***
- ✓ ***Янг, Том 1, глава1, 2,***
- ✓ ***Дейтел С++, §1.15,***

1.2.1 Инсталляция MS Visual Studio 2005 и MS VC++2005.
1.2.2 Среда программирования, компиляции и отладки. Понятие проекта.
1.2.3 Проекты Console Application.
1.2.4 Проекты win32.
1.2.5 Проекты MFC. Особенности проектов Dialog based, SDI, MDI.

Практика 1. Инсталляция продукта MS-VS-2008.

Задание. Инсталлируйте Microsoft Visual Studio 2008.

1.3 Понятие С++

- ✓ *Страуструп С++, §3.2.*
- ✓ *Дейтел С++, §1.14 – 1.16,*
- ✓ *Либерти С++21, глава 2.*

1.3.1 Понятие: консольное приложение.

1.3.2 Понятие: строка.

1.3.3 Понятие: файл, имена и расширения.

1.3.4 Понятие: вывод информации в файл.

1.3.5 Понятие: Предопределенные объекты myofs и myifs.

1.3.6 Понятие: манипулятор endl.

1.3.7 Техника: Вывод в файл текстовой строки. Простейшее форматирование.

Практика 2. *x01n01-* ***Вывод текста в файл.***

Листинг кода (файл my.cpp)

```
// Первая часть кода, обеспечивает вывод информации в файл:

// my.cpp : Defines the entry point for the console application.
//
#include <iostream>
#include <fstream>
#include <iomanip>
#include <math.h>
//
using namespace std;
using std::ofstream;
using std::ifstream;
using std::cout;
using std::cin;
using std::setw;
using std::endl;
//
ofstream myofs;
ifstream myifs;
//


// Функция main(), обеспечивает исполнение кода:
```

Листинг кода (файл my.cpp)

```
int main()
{
// x01n01. Откроем файл протокола для записи и поставим подпись:
  myofs.open("myproto.txt");
  myofs << "x01n01. Откроем файл протокола для записи и поставим подпись." << endl;
  myofs << "Иванов Иван Иванович, 18 февраля 2009 г." << endl;
  myofs << "Практическое задание x01n01. Простейшая программа С++" << endl << endl;
  myofs << "Hello, world!" << endl << endl;
// Закроем файл протокола:
  myofs.close();
  return 0;
}
```

(Конец листинга)

Результат работы кода, файл myproto.txt:

```
x01n01. Откроем файл протокола для записи и поставим подпись.
Иванов Иван Иванович, 18 февраля 2009 г.
Практическое задание x01n01. Простейшая программа С++

 Hello, world!
```

(конец вывода результата работы кода)

Задание. Измените код так, чтобы в начале отчета располагалась информация о Вас (Фамилия, имя отчество, номер учебной группы, номер задания, название задания, дата выполнения). В дальнейшем каждый отчет оформляйте аналогичным образом, указывая текущий номер задания и тему работы.

Задание. Экспериментально определите значение манипулятора endl.

Задание. Что получится, если в конце оператора не поставить символ ;

1.3.8 Техника: арифметические операции с int и double константами.

Практика 3. *x01n02-* ***Операции с константами.***

Функция main(), обеспечивает исполнение кода:

Листинг кода (файл my.cpp)

```
{
  myofs.open("myproto.txt", ios_base::app);
  myofs << "x01n02. Арифметические операции с константами типа int и double " << endl;
  myofs << "Попробуем вывести константы в файл: " << endl;
  myofs << " 7=" << 7 << endl;
  myofs << " 7.0=" << 7.0 << endl;
  myofs << " 7/3=" << 7/3 << endl;
  myofs << " 7.0/3.0=" << 7.0/3.0 << endl;
  myofs << " (7+6)/3=" << (7+6)/3 << endl;
  myofs << " (7+6)/3.0=" << (7+6)/3.0 << endl;
  myofs << " (7.0+6)/3=" << (7.0+6)/3 << endl;
  myofs.close();
}
```

(конец листинга)

Результат работы кода, файл myproto.txt:

```
x01n02. Арифметические операции с константами типа int и double
Попробуем вывести константы в файл:
 7=7
 7.0=7
 7/3=2
 7.0/3.0=2.33333
 (7+6)/3=4
 (7+6)/3.0=4.33333
 (7.0+6)/3=4.33333
```

(конец вывода результата работы кода)

Задание. Объясните разницу результатов выполнения операций с числовыми константами с фиксированной запятой (целочисленными) и с числовыми константами с плавающей запятой (вещественными, более точно, рациональными).

Задание. Экспериментально определите значение опции `ios_base::app`.

Лекция 2. Встроенные типы C++.

1.4 Встроенные типы C++.

- ✓ ***Страуструп C++, §4.1-4.6.***
- ✓ ***Либерти C++21, глава 3.***
- ✓ ***Дейтел C++, §1.20,***

1.4.1 Понятие объекта.

1.4.2 Основные ключевые точки кода, связанные с каждым объектом.

Объявление,
Определение,
Конструирование,
Инициализация,
Использование,
Разрушение.

1.4.3 Понятие класса. Имя класса (типа).
1.4.4 Понятие встроенного типа (предопределенного системного класса).
1.4.5 Объект как представитель класса (типа).
1.4.6 Понятие имени объекта.
1.4.7 Основные атрибуты объекта: поля данных и методы (функции).
1.4.8 Переменные и константные объекты (константы).
1.4.9 Понятие поля данных.
1.4.10 Понятие функции. Основные действия, которые выполняет функция:

Изменение значения полей данных объекта,
Изменение порядка исполнения операторов (функций),
Изменение состояния внешних устройств.

1.4.11 Понятие конструирования и инициализации.

1.5 *Арифметические типы.*

- ✓ ***Страуструп С++, §4.4-4.5.***
- ✓ ***Дейтел С++, §1.18,***
- ✓ ***Либерти С++21, глава 3.***

1.5.1 Арифметические типы.
1.5.2 Типы int, unsigned int, long, unsigned long.
1.5.3 Тип bool.

Практика 4. *x02n01.* ***Определение переменных типа*** *int.*

Функция main(), обеспечивает исполнение кода:

Листинг кода (файл my.cpp)

```
{
  myofs << "x02n01. Конструирование переменных целого типа, " << endl;
  myofs << "инициализация переменных целого типа, " << endl;
  myofs << "присваитвание значений переменной целого типа." << endl;
  myofs << "Создадим переменную m типа int, но инициализировать не будем." << endl;
  myofs << "Создадим и инициализируем переменную n типа int." << endl;
  myofs << "Создадим и инициализируем переменную k типа int." << endl;
  int m;
  int n=17;
  int k=12345678901;
  myofs << " m=" << m << endl;
  myofs << " n=" << n << endl;
  myofs << " k=" << k << endl;
  myofs << " Ошибка: Переменная m используется раньше, чем ей присвоено значение," <<
endl;
  myofs << " Ошибка: Попытка присвоить переменной k значение, выходящее за допустимые
границы." << endl << endl;
}
```

(конец листинга)

Результат работы кода, файл myproto.txt:

```
x02n01. Инициализация переменных целого типа
 m=-858993460
 n=17
 k=-539222987
 Ошибка: Переменная m используется раньше, чем ей присвоено значение,
 Ошибка: Попытка присвоить переменной k значение, выходящее за допустимые границы.
```

(конец вывода результата работы кода)

Задание. Объясните две диагностические выдачи на этапе исполнения. Исправьте код так чтобы все операции стали корректными.

1.5.4 Копирование значений переменных целого типа.

Практика 5. *x02n02.* ***Копирование переменных типа*** *int.*

Функция main(), обеспечивает исполнение кода:

Листинг кода (файл my.cpp)

```
{
  myofs << "x02n02. Копирование переменных целого типа" << endl;
  myofs << " Переменные m, n, k не инициализированы, но им присвоены значения" <<
endl;
  int m, n, k, p;
  m=123;
  n=-789;
  k=987654;
  p=n;
  myofs << " m=" << m << " n=" << n << " k=" << k << " p=" << p << endl << endl;
}
```

(конец листинга)

Результат работы кода, файл myproto.txt:

```
x02n02. Копирование переменных целого типа
 Переменные m, n, k не инициализированы, но им присвоены значения
 m=123 n=-789 k=987654 p=-789
```

(конец вывода результата работы кода)

Задание. Создайте несколько переменных, используйте операцию копирования.

1.5.5 Типы float, double.

Практика 6. *x02n03.* ***Определение и инициализация переменных типа*** *double.*

Функция main(), обеспечивает исполнение кода:

Листинг кода (файл my.cpp)

```
{
  myofs << "x02n03. Инициализация переменных типа double" << endl;
  double x, y=123.567, z=3.45e123;
  myofs << " x=" << x << endl << " y=" << y << endl << " z=" << z << endl;
  myofs << " Ошибка: Переменная x используется раньше, чем ей присвоено значение," <<
endl;
  myofs << " Ошибка: Попытка присвоить переменной z значение, выходящее за допустимые
границы." << endl << endl;
}
```

(конец листинга)

Результат работы кода, файл myproto.txt:

```
x02n03. Инициализация переменных типа double
 x=-9.25596e+061
 y=123.567
 z=3.45e+123
 Ошибка: Переменная x используется раньше, чем ей присвоено значение,
 Ошибка: Попытка присвоить переменной z значение, выходящее за допустимые границы.
```

(конец вывода результата работы кода)

Задание. Разберитесь в значении фигурных скобок, обрамляющих код каждого упражнения.

1.5.6 Арифметические операторы без преобразования типа.

Практика 7. *x02n04.* ***Арифметические операции без преобразования типов.***

Листинг кода (файл my.cpp)

```
{
  myofs << "x02n04. Арифметические операции без преобразования типов" << endl;
  int m=47, n=8;
  myofs << " int m=" << m << ", int n=" << n << ", m+n=" << m+n << ", m*n=" << m*n <<
", m/n=" << m/n << endl;
  double x=47, y=8;
  myofs << " double x=" << x << ", double y=" << y << ", x+y=" << x+y << ", x*y=" <<
x*y << ", x/y=" << x/y << endl << endl;
}
```

(конец листинга)

Результат работы кода, файл myproto.txt:

```
x02n04. Арифметические операции без преобразования типов
 int m=47, int n=8, m+n=55, m*n=376, m/n=5
 double x=47, double y=8, x+y=55, x*y=376, x/y=5.875
```

(конец вывода результата работы кода)

Задание. См. сборник задач.

1.5.7 Типы char, char[], char*.

1.5.8 Понятие длины (size). Функция sizeof(имя).

1.5.9 Переменные и константные объекты.

1.5.10 Понятие преобразования типов.

Практика 8. *x02n05.* ***Арифметические операции с явным преобразованием типов.***

Листинг кода (файл my.cpp)

```
{
  myofs << "x02n05. Арифметические операции с явным преобразованием типов" << endl;
  int m=47, n=8;
  myofs << " int m=" << m << ", int n=" << n << ", m/n=" << m/n << ",
    double(m)/n=" << double(m)/n;
  myofs << " m/double(n)=" << m/double(n) << ",
    double(m)/double(n)=" << double(m)/double(n) << endl << endl;
}
```

(конец листинга)

Результат работы кода, файл myproto.txt:

```
x02n05. Арифметические операции с явным преобразованием типов
 int m=47, int n=8, m/n=5, double(m)/n=5.875 m/double(n)=5.875,
   double(m)/double(n)=5.875
```

(конец вывода результата работы кода)

Задание. См. сборник задач.

1.6 Арифметические операторы С++.

- ✓ ***Страуструп С++, §6.2.***
- ✓ ***Дейтел С++, §1.18, §2.11, §2.12,***
- ✓ ***Либерти С++21, глава 4.***

1.6.1 Арифметические операции.

1.6.2 Операция копирования.

1.6.3 Операции сложения, умножения, вычитания, деления.

1.6.4 Операция вычисления остатка от деления нацело.

Практика 9. *x02n06.* ***Остаток от деления.***

Листинг кода (файл my.cpp)

```
{
  myofs << "x02n06. Остаток от деления" << endl;
  int m=47, n=8;
```

```
  myofs << " int m=" << m << ", int n=" << n << ", m/n=" << m/n << ", m%n=" << m%n <<
endl << endl;
}
```

(конец листинга)

Результат работы кода, файл myproto.txt:

```
x02n06. Остаток от деления
 int m=47, int n=8, m/n=5, m%n=7
```

(конец вывода результата работы кода)

Задание. См. сборник задач.

1.6.5 Операции инкремента ++ и декремента – для переменных типа int.

Практика 10. *x02n11.* ***Операции инкремента и декремента для*** *int.*

Листинг кода (файл my.cpp)

```
{
  myofs << "x02n11. Операции инкремента и декремента для int" << endl;
  int m=23;
  myofs << " int m=" << m;
  int m1=m++;
  myofs << ",  Если int m1=m++, то после этого m=" << m << ",  m1=" << m1 << endl;
  int n=23;
  myofs << " int n=" << n;
  int n1=++n;
  myofs << ",  Если int n1=++n, то после этого n=" << n << ",  n1=" << n1 << endl;
  m=77;
  myofs << " int m=" << m;
  m1=m--;
  myofs << ",  Если int m1=m--, то после этого m=" << m << ",  m1=" << m1 << endl;
  n=77;
  myofs << " int n=" << n;
  n1=--n;
  myofs << ",  Если int n1=--n, то после этого n=" << n << ",  n1=" << n1 << endl;
  m=35;
  myofs << " int m=" << m;
  m1=m+=15;
  myofs << ",  Если int m1=m++15, то после этого m=" << m << ",  m1=" << m1 << endl;
  n=35;
  myofs << " int n=" << n;
  n1=n-=15;
  myofs << ",  Если int n1=n-=15, то после этого n=" << n << ",
    n1=" << n1 << endl << endl;
}
```

(конец листинга)

Результат работы кода, файл myproto.txt:

```
x02n11. Операции инкремента и декремента для int
 int m=23,  Если int m1=m++, то после этого m=24,  m1=23
 int n=23,  Если int n1=++n, то после этого n=24,  n1=24
 int m=77,  Если int m1=m--, то после этого m=76,  m1=77
 int n=77,  Если int n1=--n, то после этого n=76,  n1=76
 int m=35,  Если int m1=m++15, то после этого m=50,  m1=50
 int n=35,  Если int n1=n-=15, то после этого n=20,  n1=20
```

(конец вывода результата работы кода)

Задание. См. сборник задач.

1.6.6 Операции инкремента ++ и декремента – для переменных типа double.

Практика 11. *x02n11.* ***Операции инкремента и декремента для*** *double.*

Листинг кода (файл my.cpp)

```
{
```

```
  myofs << "x02n12. Операции инкремента и декремента для double" << endl;
  double x=23.4;
  double xpp = x+=3;
  myofs << " double x=" << x << ",  double x1=x+=3, x1=" << xpp << ",  x=" << x << endl
<< endl;
}
```

(конец листинга)

Результат работы кода, файл myproto.txt:

```
x02n12. Операции инкремента и декремента для double
 double x=26.4,  double x1=x+=3, x1=26.4,  x=26.4
```

(конец вывода результата работы кода)

Задание. См. сборник задач.

1.6.7 Операции << и >> для int.

Практика 12. x02n07. Операция << для int.

Листинг кода (файл my.cpp)

```
{
  myofs << "x02n07. Операция << для int" << endl;
  int m=128, n=77;
  int m1=m<<1; int n1=n<<1;
  myofs << " int m=" << m << ",  m<<1 =" << m1 << ",  n=" << n << ",  n<<1 =" << n1 <<
endl;
  int m2=m<<2; int n2=n<<2;
  myofs << " int m=" << m << ",  m<<2 =" << m2 << ",  n=" << n << ",  n<<2 =" << n2 <<
endl << endl;
}
```

(конец листинга)

Результат работы кода, файл myproto.txt:

```
x02n07. Операция << для int
 int m=128,  m<<1 =256,  n=77,  n<<1 =154
 int m=128,  m<<2 =512,  n=77,  n<<2 =308
```

(конец вывода результата работы кода)

Задание. См. сборник задач.

Практика 13. x02n08. Операция >> для int.

Листинг кода (файл my.cpp)

```
{
  myofs << "x02n08. Операция >> для int" << endl;
  int m=128, n=77;
  int m1=m>>1; int n1=n>>1;
  myofs << " int m=" << m << ",  m>>1 =" << m1 << ",  n=" << n << ",  n>>1 =" << n1 <<
endl;
  int m2=m>>2; int n2=n>>2;
  myofs << " int m=" << m << ",  m>>2 =" << m2 << ",  n=" << n << ",  n>>2 =" << n2 <<
endl << endl;
  int m3=m>>3; int n3=n>>3;
  myofs << " int m=" << m << ",  m>>3 =" << m3 << ",  n=" << n << ",  n>>3 =" << n3 <<
endl << endl;
}
```

(конец листинга)

Результат работы кода, файл myproto.txt:

```
x02n08. Операция >> для int
 int m=128,  m>>1 =64,  n=77,  n>>1 =38
 int m=128,  m>>2 =32,  n=77,  n>>2 =19
 int m=128,  m>>3 =16,  n=77,  n>>3 =9
```

(конец вывода результата работы кода)

Задание. См. сборник задач.

1.6.8 Префиксные и постфиксные операторы.

1.6.9 Применение скобок.

1.6.10 Понятие функции. Основные действия, которые выполняет функция:

Изменение значения полей данных объекта,
Изменение порядка исполнения операторов (функций),
Изменение состояния внешних устройств.

1.6.11 Стандартные арифметические операторы. Функции floor(double), ceil(double), sqrt(double).

Практика 14. *x02n09.* ***Функции*** *sqrt(), floor(), ceil().*

Листинг кода (файл my.cpp)

```
{
  myofs << "x02n09. Функции sqrt(), floor(), ceil()" << endl;
  double x=17.345;
  int m=floor(x);
  int n=ceil(x);
  myofs << " double x=" << x << ",  sqrt(x)=" << sqrt(x) << ",
    floor(x)=" << m << ",  ceil(x)=" << n << endl << endl;
}
```

(конец листинга)

Результат работы кода, файл myproto.txt:

```
x02n09. Функции sqrt(), floor(), ceil()
 double x=17.345,  sqrt(x)=4.16473,  floor(x)=17,  ceil(x)=18
```

(конец вывода результата работы кода)

1.7 Преобразование арифметических типов.

- ✓ ***Страуструп С++, §В6.***
- ✓ ***Дейтел С++, §1.18,***
- ✓ ***Либерти С++21, глава 4.***

1.7.1 Преобразование типов при выполнении арифметических операций.

1.7.2 Преобразование int в double.

Практика 15. *x02n01.* ***Операции с*** *int* ***и*** *double* ***константами.***

Листинг кода (файл my.cpp)

(конец листинга)

Результат работы кода, файл myproto.txt:

(конец вывода результата работы кода)

Задание. Разберитесь в значении фигурных скобок, обрамляющих код каждого упражнения.

Задание. Исправьте код так чтобы диагностики ошибок исчезли.

Задание. Создайте несколько переменных типа int, инициализируйте, выполните операции сложения, вычитания, умножения, деления, инкремента. Выведите результаты в файл. Создайте несколько переменных типа double, инициализируйте, выполните операции сложения, вычитания, умножения, деления, инкремента, декремента. Выведите в файл протокола. Вычислите $\frac{1+2+3+4+5}{4}$, $\frac{1.0+2.0+3.0+4.0+5.0}{4.0}$. Объясните разницу.

1.7.3 Преобразование double в int. Функции double floor(double), double ceil(double).

Практика 16. *x02n10.* ***Функции*** *sqrt(), floor(), ceil()".*

Листинг кода (файл my.cpp)

```
{
  myofs << "x02n10. Функции sqrt(), floor(), ceil()" << endl;
  double x=16;
  int m=floor(x);
  int n=ceil(x);
  myofs << " double x=" << x << ",  sqrt(x)=" << sqrt(x) << ",
    floor(x)=" << m << ",  ceil(x)=" << n << endl << endl;
}
```

(конец листинга)

Результат работы кода, файл myproto.txt:

```
x02n10. Функции sqrt(), floor(), ceil()
 double x=16,  sqrt(x)=4,  floor(x)=16,  ceil(x)=16
```

(конец вывода результата работы кода)

1.7.4 Оператор typedef.

1.8 Вывод в файл значений арифметических типов.

- ✓ *Страуструп С++, §21.2.*

1.8.1 Вывод int, double.

1.8.2 Вывод с комментариями.

Лекция 3. Операторы цикла.

1.9 Операторы цикла.

- ✓ *Страуструп С++, §6.3.*
- ✓ *Дейтел С++, §2.7– §2.15, §2.17– §2.18,*
- ✓ *Либерти С++21, глава 4, 7.*

1.9.1 Цикл for. Итератор (переменная цикла).

```
{
  myofs << "Упражнение 03-01. Оператор цикла for с заданным числом итераций." << endl;
  int m, n=0;
  double d=0;
  for(m=1; m<124; m++){
    d += 1.0/double(m);
    n++;
  }
  myofs << "Число итераций n=" << n;
  myofs << ", сумма " << n << " обратных натуральных чисел=" << d << endl;
  myofs << "Значение переменной цикла m после его завершения m=" << m << endl << endl;
}
{
  myofs << "Упражнение 03-02. Оператор цикла for с заданным приращением." << endl;
  int m, n=0;
  double d=0;
  for(m=1; m<124; m+=2){
    d += 1.0/double(m);
    n++;
  }
  myofs << "Число итераций n=" << n << ", сумма " << n << " обратных нечетных
натуральных чисел=" << d << endl;
  myofs << "Значение переменной цикла m после его завершения m=" << m << endl << endl;
}
{
  myofs << "Упражнение 03-03. Оператор цикла for и оператор прерывания цикла break" <<
endl;
  int n=0; double d=0;
  for(int m=1; m<100000; m++){
    d += 1.0/double(m);
    n++;
    if (d>10.0) break;
  }
  myofs << "Число итераций n=" << n << ", сумма " << n << " обратных натуральных
чисел=" << d << endl << endl;
}
{
  myofs << "Упражнение 03-05. Оператор for и оператор continue" << endl;
  int n=0; double d=0;
  for(int m=1; m<1000; m++){
    if (m%2==1) continue;
    d += 1.0/double(m);
    n++;
  }
  myofs << "Число итераций n=" << n << ", сумма " << n << " обратных нечетных
натуральных чисел=" << d << endl << endl;
}
{
  myofs << "Упражнение 03-04. Прерывание оператора for с помощью оператора goto" <<
endl;
  int n=0; double d=0;
  for(int m=1; m<100000; m++){
    d += 1.0/double(m);
    n++;
    if(d>10.0) goto next;
  }
  next:
  myofs << "Число итераций n=" << n << ", сумма " << n << " обратных нечетных
натуральных чисел=" << d << endl << endl;
}
{
  myofs << "Упражнение 03-11. Оператор while" << endl;
  int n=0;
  double d=0;
```

```
  while(n<1000){
    n++;
    d+=1.0/double(n);
  }
  myofs << "Число итераций n=" << n << ", сумма " << n << " обратных натуральных
чисел=" << d << endl << endl;
}
{
  myofs << "Упражнение 03-12. Оператор while(true)" << endl;
  int n=0; double d=0;
  while(true){
    n++;
    d += 1.0/double(n);
    if(d>10.0)break;
  }
  myofs << "Число итераций n=" << n << ", сумма " << n << " обратных натуральных
чисел=" << d << endl << endl;
}
{
  myofs << "Упражнение 03-10. Оператор do-while" << endl;
  int n=0;
  double d=0;
  do{n++; d+=1.0/double(n);}
  while(n<1000);
  myofs << "Число итераций n=" << n << ", сумма " << n << " обратных натуральных
чисел=" << d << endl << endl;
}
{
  myofs << "Упражнение 03-13. Вложенные операторы цикла." << endl;
  const int M=12, N=5;
  int m, n;
  double a[M][N];
  for(m=0; m<M; m++)
    for(n=0; n<N; n++)
      a[m][n] = m*n;
    for(m=0; m<M; m++)
    {
      for(n=0; n<N; n++)
      {
        myofs << "  a[" << setw(2) << m << "," << setw(2) << n << "]=";
        myofs << setw(5) << setprecision(1) << a[m][n];
      }
      myofs << endl;
    }
}
{
  myofs << "Упражнение 03-14. Таблица умножения." << endl;
  const int M=12, N=5;
  int m, n;
  double a[M][N];
  for(m=0; m<M; m++)
    for(n=0; n<N; n++)
      a[m][n] = m*n;
    myofs << "    ";
  for(n=0; n<N; n++)
    myofs << setw(6) << n;
  myofs << endl;
  for(m=0; m<M; m++)
  {
    myofs << setw(3) << m << "  ";
    for(n=0; n<N; n++)
      myofs << setw(6) << setprecision(1) << a[m][n];
    myofs << endl;
  }
```

```
  myofs << endl;
}
```

(конец листинга)

Результат работы кода, файл myproto.txt:

```
Упражнение 03-14. Таблица умножения.
         0     1     2     3     4
  0     0.0   0.0   0.0   0.0   0.0
  1     0.0   1.0   2.0   3.0   4.0
  2     0.0   2.0   4.0   6.0   8.0
  3     0.0   3.0   6.0   9.0  12.0
  4     0.0   4.0   8.0  12.0  16.0
  5     0.0   5.0  10.0  15.0  20.0
  6     0.0   6.0  12.0  18.0  24.0
  7     0.0   7.0  14.0  21.0  28.0
  8     0.0   8.0  16.0  24.0  32.0
  9     0.0   9.0  18.0  27.0  36.0
 10     0.0  10.0  20.0  30.0  40.0
 11     0.0  11.0  22.0  33.0  44.0
```

(конец вывода результата работы кода)

Задание. Составьте таблицы сложения и возведения в степень .

Практика 17. *x03n15.* ***Анализ двумерной таблицы.***

Листинг кода (файл my.cpp)

```
{
  myofs << "Упражнение 03-15. Анализ двумерной таблицы" << endl;
  const int M=12, N=5;
  double a[M][N];
  double ar1[M], ar2[M], ac1[N], ac2[N];
  int ir1[M], ir2[M], ic1[N], ic2[N];
  int m, n, m1, m2, n1, n2;
  double d1, d2, d3;
  for(m=0; m<M; m++)
    for(n=0; n<N; n++)
      a[m][n] = 10.0*((double(rand())/double(RAND_MAX))*2.0-1.0);
  for(m=0; m<M; m++)
  {
    d1=d2=a[m][0];
    n1=n2=0;
    for(n=1; n<N; n++)
    {
      d3=a[m][n];
      if(d3<d1){d1=d3; n1=n;}
      if(d3>d2){d2=d3; n2=n;}
    }
  ar1[m]=d1;
  ar2[m]=d2;
  ir1[m]=n1;
  ir2[m]=n2;
  }
  for(n=0; n<N; n++)
  {
    d1=d2=a[0][n];
    m1=m2=0;
    for(m=1; m<M; m++)
    {
      d3=a[m][n];
      if(d3<d1){d1=d3; m1=m;}
      if(d3>d2){d2=d3; m2=m;}
    }
    ac1[n]=d1;
    ac2[n]=d2;
    ic1[n]=m1;
```

```
    ic2[n]=m2;
  }
  myofs << "                    ";
  for(n=0; n<N; n++){myofs << " [" << setw(2) << ic2[n] << "," << n << "]";} myofs <<
endl;
  myofs << "                    ";
  for(n=0; n<N; n++){myofs << setw(7) << setprecision(3) << ac2[n];} myofs << endl;
  myofs << "                  ------------------------------------------------------
--" << endl;
  for(m=0; m<M; m++){
    myofs << "a[" << setw(2) << m << "," << setw(1) << ir1[m] << "]=" << setw(7) <<
setprecision(3) << ar1[m] << " || ";
    for(n=0; n<N; n++){
      myofs << setw(7) << setprecision(3) << a[m][n];
    }
    myofs << " || a[" << setw(2) << m << "," << setw(1) << ir2[m] << "]=" << setw(7) <<
setprecision(3) << ar2[m] << endl;
  }
  myofs << "                  ------------------------------------------------------
--" << endl;
  myofs << "                    ";
  for(n=0; n<N; n++){myofs << " [" << setw(2) << ic1[n] << "," << n << "]";} myofs <<
endl;
  myofs << "                    ";
  for(n=0; n<N; n++){myofs << setw(7) << setprecision(3) << ac1[n];} myofs << endl;
}
myofs.close();
return 0;
}
```

(конец листинга)

Результат работы кода, файл myproto.txt:

```
Упражнение 03-15. Анализ двумерной таблицы
                    [ 4,0] [ 2,1] [10,2] [10,3] [ 1,4]
                     9.771  7.179  9.936  9.994  4.932
                  ------------------------------------------------------------
a[ 0,0]= -9.975 ||  -9.975  1.272 -6.134  6.175  1.700 || a[ 0,3]=  6.175
a[ 1,1]= -2.994 ||  -0.403 -2.994  7.919  6.457  4.932 || a[ 1,2]=  7.919
a[ 2,0]= -6.518 ||  -6.518  7.179  4.210  0.271 -3.920 || a[ 2,1]=  7.179
a[ 3,0]= -9.700 ||  -9.700 -8.172 -2.711 -7.054 -6.682 || a[ 3,2]= -2.711
a[ 4,3]= -9.907 ||   9.771 -1.086 -7.618 -9.907 -9.822 || a[ 4,0]=  9.771
a[ 5,0]= -2.442 ||  -2.442  0.633  1.424  2.035  2.143 || a[ 5,4]=  2.143
a[ 6,4]= -8.859 ||  -6.675  3.261 -0.984 -2.958 -8.859 || a[ 6,1]=  3.261
a[ 7,4]= -3.961 ||   2.154  5.666  6.052  0.398 -3.961 || a[ 7,2]=  6.052
a[ 8,4]=  0.787 ||   7.519  4.534  9.118  8.514  0.787 || a[ 8,2]=  9.118
a[ 9,0]= -7.153 ||  -7.153 -0.758 -5.293  7.245 -5.808 || a[ 9,3]=  7.245
a[10,4]=  2.230 ||   5.593  6.873  9.936  9.994  2.230 || a[10,3]=  9.994
a[11,4]= -9.525 ||  -2.151 -4.676 -4.054  6.803 -9.525 || a[11,3]=  6.803
                  ------------------------------------------------------------
                     [ 0,0] [ 3,1] [ 4,2] [ 4,3] [ 4,4]
                     -9.975 -8.172 -7.618 -9.907 -9.822
```

(конец вывода результата работы кода)

Задание. .

Лекция 4. Логические операторы

1.10 Логические операторы C++.

- ✓ ***Страуструп C++, §6.2, §6.3.***
- ✓ ***Дейтел C++, §2.4– §2.6, §2.16, §2.19– §2.20,***
- ✓ ***Либерти C++21, глава 4.***

1.10.1 Бинарная булевская функция == для встроенных типов.

1.10.2 Логический бинарный оператор ||.
1.10.3 Логический бинарный оператор &&.
1.10.4 Операторы if и if - else.
1.10.5 Оператор switch.
1.10.6 Совместное применение циклов и логических операторов.

1.11 Ввод текста и данных из файла.

- ✓ *Страуструп C++, §3.6, 21.3.*

1.11.1 Предопределенный объект myifs и его использование.
1.11.2 Ввод нескольких слов. Понятие буфера.
1.11.3 Понятие разделителя.
1.11.4 Ввод целых и вещественных значений.

1.12 Простой интерпретатор.

1.12.1 Программирование интерпретатора значений и операций, вводимых из файла.

Практика 18. p1n04 **Ввод данных из файла и анализ данных**

Задание. Сформируйте файл, содержащий несколько целых и действительных чисел. Введите значения из этого файла, используя оператор извлечения данных из потока >>. Найдите наименьшее, наибольшее. Напишите код для вывода некоторого текста, который определяется введенными значениями.

Практика 19. p1n05 **Логические операторы**

Задание. Используя оператор цикла for и логические операторы, найдите значение суммы всех натуральных чисел в пределах от 5 до 5555, которые делятся на 2, 3, 5, 11, но не делятся на 7. Напечатайте в файле протокола таблицу всех таких чисел.

Глава 2 Функции

Лекция 5. Функции

1.13 Функции.

- ✓ *Страуструп C++, §7.1, §7.2, §7.3.*
- ✓ *Дейтел C++, §3.1– §3.16,*
- ✓ *Либерти C++21, глава 5.*

1.13.1 Понятие функции.
1.13.2 Формальные параметры, возвращаемое значение.
1.13.3 Оператор вызова функции, фактические параметры.
1.13.4 Объявление (прототип) и описание функции.

Практика 20. x05n01. **Простейшая функция.**

Листинг кода (файл my.cpp)

```
double f101(double x, double y){ // Вычисляет x+y;
  double z=x+y;
  return z;
}
double f102(double x){ // Вычисляет cos(x);
  int n=0; double d=1.0; double b=1.0;
  while(n<1000)
  {
    n+=2;
    b=-b*x*x/double((n-1)*n);
    d+=b;
  }
  return d;
```

```
}
```
(конец листинга)

Результат работы кода, файл myproto.txt:
```
Упражнение 05-01. Функция, возвращающая значение
 f101(3.3,4.3)= 7.700000
Упражнение 05-02. Функция, возвращающая значение
 f102(3.1425)= -1.000000
```
(конец вывода результата работы кода)

Задание. Измените код f102(…) так, чтобы оператор цикла завершал свою работу при выполнении условия «модуль очередного слагаемого не более 10^{-10}, но не раньше чем величина n достигнет значения 10».

Задание. Создайте код функции double f102m(double x), которая вычисляет $\sin x$.

1.13.5 Практика кодирования функций.

Задание. Напишите функцию, которая вычисляет сумму всех натуральных чисел в пределах от M до N, которые делятся на m, n, k, но не делятся на p, q, r. Параметры передавайте по значению.

Задание. Напишите функцию int mymax(int m, int n), вычисляющую большую из двух переменных типа int.

Задание. Напишите аналогичную функцию, которая будет работать для 4 переменных и для 8 переменных. При этом используйте mymax(..,..).

1.13.6 Преобразование типов параметров при вызове функции.

Практика 21. *x05n02.* ***Преобразование типов параметров.***

Листинг кода (файл my.cpp)
```
{
  myofs << "Упражнение 05-03. Преобразование типов." << endl;
  myofs << " dfdd(3.1, 4.7)= " << dfdd(3.1, 4.7) << endl;
  myofs << " ifii(3.1, 4.7)= " << ifii(3.1, 4.7) << endl;
  myofs << " ifii(3.8, 4.7)= " << ifii(3.8, 4.7) << endl;
  myofs << " ifdd1(3.1, 4.7)= " << ifdd1(3.1, 4.7) << endl;
  myofs << " ifdd1(3.8, 4.7)= " << ifdd1(3.8, 4.7) << endl;
  myofs << " ifdd2(3.1, 4.7)= " << ifdd2(3.1, 4.7) << endl;
  myofs << " ifdd2(3.8, 4.7)= " << ifdd2(3.8, 4.7) << endl << endl;
}
```
(конец листинга)

Результат работы кода, файл myproto.txt:
```
 Упражнение 05-03. Преобразование типов.
 dfdd(3.1, 4.7)= 7.800000
 ifii(3.1, 4.7)= 7.000000
 ifii(3.8, 4.7)= 7.000000
 ifdd1(3.1, 4.7)= 7
 ifdd1(3.8, 4.7)= 8
 ifdd2(3.1, 4.7)= 7
 ifdd2(3.8, 4.7)= 7
```
(конец вывода результата работы кода)

Задание. Напишите код функции int factor(int n), которая вычисляет $n!=1\cdot 2\cdot 3\cdot ...\cdot n$. Напишите код функции double factor(double x), которая вычисляет $x!=1\cdot 2\cdot 3\cdot ...\cdot x$. Исследуйте результат применения этих двух функций для вычисления
1) factor(1), factor(10), factor(100), factor(1000),
2) factor(1.0), factor(10.0), factor(100.0), factor(1000.0).

Лекция 6. ***Различные способы передачи и возврата***

1.13.7 Основная ошибка при кодировании функций.

Практика 22. *x05n04.* ***Попытка возврата результата через параметр, передаваемый по значению.***

Листинг кода (файл my.cpp)

```
{
  myofs << "Упражнение 05-04. Нельзя возвратить результат через параметр, передаваемый по значению" << endl;
  double z = 7.7;
  myofs << "    До вызова f104(z) значение переменной z равно " << z << endl;
  int m = f104(z);
  myofs << " После вызова f104(z) значение переменной z равно " << z << endl << endl;
}
```

(конец листинга)

Результат работы кода, файл myproto.txt:

```
Упражнение 05-04. Нельзя возвратить результат через параметр,
                  передаваемый по значению
    До вызова f104(z) значение переменной z равно 7.700000
 После вызова f104(z) значение переменной z равно 7.700000
```

(конец вывода результата работы кода)

1.13.8 Передача параметра в функцию по значению.

Практика 23. *p2n06* ***Функции, передача параметра по значению***

Листинг кода (файл my.cpp)

(конец листинга)

Результат работы кода, файл myproto.txt:

(конец вывода результата работы кода)

1.14 *Передача и возврат по указателю.*

1.14.1 Передача входного параметра по указателю.

Листинг кода (файл my.cpp)

(конец листинга)

Результат работы кода, файл myproto.txt:

(конец вывода результата работы кода)

1.14.2 Функция с пустым списком параметров.

Практика 24. *x05n02.* ***Простейшая функция.***

Листинг кода (файл my.cpp)

(конец листинга)

Результат работы кода, файл myproto.txt:

(конец вывода результата работы кода)

Практика 25. *x05n03.* ***Попытка возврата результата через параметр, передаваемый по значению.***

Листинг кода (файл my.cpp)

(конец листинга)

Результат работы кода, файл myproto.txt:

(конец вывода результата работы кода)

1.14.3

1.14.4 Передача выходного параметра по указателю.

1.14.5 Передача нескольких параметров по указателю.

1.14.6 Значение NULL.

1.14.7 Передача значения NULL для управления работой функции.

1.14.8 Значение параметра по умолчанию. Применение сравнения указателя с NULL.

- ✓ ***Страуструп C++, §7.1, §7.2, §7.3.***

Практика 26. Функции, передача параметра по указателю

Задание. Напишите функцию f(x1, x2, x3, x4), которая осуществляет циклическую перестановку величин int x1, x2, x3, x4 (т.е. значение x1 заменяем на значение x2, значение x2 заменяем на значение x3, … значение x4 заменяем на значение x1). Параметры передавайте через указатели. Проверьте, будет ли данная функция работать для фактических параметров const, например можно ли написать нечто типа f(1, 2, 3, 4). Проверьте, будет ли данная функция работать для фактических параметров другого типа (например, double). Напишите функцию f(x1, x2, x3, x4, n), которая осуществляет n-кратный вызов функции f(x1, x2, x3, x4). Параметры x1, x2, x3, x4 передайте через указатели, параметр n по значению. Напишите функцию sort(x1, x2, x3, x4), которая осуществляет сортировку величин int типа x1, x2, x3, x4 по возрастанию значений. Способ передачи параметров выберите самостоятельно.

1.15 Передача параметра в функцию по ссылке.

- ✓ ***Страуструп C++, §7.1, §7.2, §7.3.***
- ✓ ***Дейтел C++, §3.17.***

1.15.1 Понятие ссылки. Описание, определение и инициализация.

1.15.2 Понятие типа ссылки.

1.15.3 Копирование ссылки.

1.15.4 Копирование ссылаемого объекта.

Практика 27. Функции, передача параметра по ссылке

Задание. Напишите функцию f(x1, x2, x3, x4), которая осуществляет циклическую перестановку величин int x1, x2, x3, x4 (т.е. значение x1 заменяем на значение x2, значение x2 заменяем на значение x3, … значение x4 заменяем на значение x1). Параметры передавайте по ссылке. Проверьте, будет ли данная функция работать для фактических параметров const, например можно ли написать нечто типа f(1, 2, 3, 4). Проверьте, будет ли данная функция работать для фактических параметров другого типа (например, double). Напишите функцию f(x1, x2, x3, x4, n), которая осуществляет n-кратный вызов функции f(x1, x2, x3, x4). Параметры x1, x2, x3, x4 передайте по ссылке, параметр n по значению. Напишите функцию sort(x1, x2, x3, x4), которая осуществляет сортировку величин int типа x1, x2, x3, x4 по возрастанию значений. Параметры передавайте по ссылке.

1.16 Защита операндов

- ✓ ***Страуструп C++, §7.1, §7.2, §7.3.***

1.16.1 Константные функции и константные параметры

1.17 Параметры по умолчанию.

- ✓ ***Страуструп C++, §7.5.***
- ✓ ***Дейтел C++, §3.18.***
- ✓ ***Либерти C++21, глава 5.***

1.18 Перегруженные функции.

- ✓ ***Страуструп C++, §7.4.***
- ✓ ***Дейтел C++, §3.20.***
- ✓ ***Либерти C++21, глава 10.***

1.19 Рекурсивные функции.

1.19.1 Рекурсивные функции

- ✓ ***Дейтел C++, §3.20.***

Практика 28. p2n09 Перегруженные и рекурсивные функции

Задание. Напишите функцию f(x1, x2, x3, x4), которая осуществляет циклическую перестановку величин double x1, x2, x3, x4. Пусть в одном файле определены функции f(int& x1, …) и f(double& x1, …). Включите в код функции оператор выдачи сообщения в файл протокола. Определите, какая версия функции f(…) активизируется при исполнении операторов f(int, int, int, int), f(double, double, double, double), f(int, double, int, double). Напишите функцию int g(int x1, int x2, int x3), которая возвращает x1+x2+x3. Напишите функцию int g(double x1, double x2, double x3), которая возвращает x1*x2*x3. Определите, какая версия функции g(…) активизируется при исполнении операторов g(int, int, int), g(double, double, double), g(int, double, int), g(int, double, char). Определите, какая версия функции g(…) активизируется при исполнении операторов g(3, 4, 5), g(3.0, 4.0, 5.0), g('a', 2, 6.0).

Задание. Напишите функцию s(x1, x2, x3, x4), которая вычисляет x1+x2+x3+x4 и которая может быть также активизирована операторами int m4=s(1, 2, 3, 4), int m3=s(1, 2, 3), int m2=s(1, 2), int m1=s(1), int m0=f(). Возможно ли написать две перегруженные версии данной функции, одна из которых принимает аргументы типа int, а другая типа double?

Задание. Напишите рекурсивную функцию.

Глава 3 Массивы и указатели

Лекция 7. *Одномерные массивы и указатели*

1.20 *Указатели.*

- ✓ ***Страуструп C++, §5.1, §5.2.***
- ✓ ***Дейтел C++, §5.1– §5.7.***
- ✓ ***Либерти C++21, глава 8, 12.***

1.20.1 Понятие указателя. Описание, определение и инициализация.

1.20.2 Адрес объекта и указатель на объект.

1.20.3 Понятие типа указателя.

1.20.4 Операция разыменовывания указателя.

1.20.5 Копирование указателя.

1.20.6 Копирование указываемого объекта.

1.20.7 Инкремент и декремент указателя.

1.20.8 Константные указатели и указатели на константные объекты.

1.20.9 Указатель на указатель.

Практика 29. *p3n10* *Указатели*

Задание. Создайте и инициализируйте несколько переменных разных типов. Создайте и инициализируйте указатели на эти переменные. Исследуйте операцию копирования указателя. Исследуйте операцию копирования указываемых объектов через указатели на них. Продемонстрируйте разыменовывание указателя. Продемонстрируйте разницу при выводе в файл объекта и указателя на него. Выведите в файл указатель и разыменованный указатель с комментариями. Пусть int m=7; int* pm=&m; Найдите и выведите в файл значение величины *pm. Сделайте то же самое для величины (*pm+1). Найдите и выведите в файл значение величины pm. Сделайте то же самое для величины (pm+1). Объясните. Какое значение будут иметь величины (*pm)++, *(pm++), *pm++?

1.21 *Одномерные массивы*

- ✓ ***Страуструп C++, §5.1, §5.2.***
- ✓ ***Дейтел C++, §4.1– §4.8.***
- ✓ ***Либерти C++21, глава 8, 12.***

1.21.1 Понятие одномерного массива с фиксированной размерностью.

1.21.2 Доступ к элементам одномерного массива с помощью переменной с индексом.

1.21.3 Инициализация одномерного массива.

1.21.4 Неявное указание размера массива при инициализации.

1.21.5 Вывод значения массива в файл.

1.22 Одномерные массивы и указатели

- ✓ ***Страуструп С++, §5.1, §5.3, §5.4 .***
- ✓ ***Дейтел С++, §4.1– §4.8.***
- ✓ ***Либерти С++21, глава 8, 12.***

1.22.1 Операции над указателями.

1.22.2 Операции инкремента и декремента.

1.22.3 Доступ к элементам одномерного массива с помощью указателей.

1.22.4 Взаимосвязь указателей и массивов.

1.23 Передача одномерного массива в функцию.

Практика 30. p3n11 Одномерные массивы

Задание. Создайте и инициализируйте несколько одномерных массивов разных типов с заданной размерностью. Исследуйте применение указателей и переменных с индексом для доступа к элементам массива. Выведите в файл.

Задание. Пусть имеется массив int X[10], инициализированный последовательными значениями 1000, 1100, 1200, 1300 и т.д. Какие значения будут иметь величины X, X[0], X[7], *X, *X+1, *X+7, *(X+7), (*X)+7. Какое значение будет иметь величины X+10000, *(X+10000)?

Задание. Пусть int X[10]; intY[10]; инициализируем массивы X и Y (значения придумайте сами). Что произойдет при исполнении операторов a) X[7]=Y[7], b) X[0]=Y[0], c) X=Y?

Лекция 8. Операторы new и delete.

1.24 Операторы new и delete.

- ✓ ***Страуструп С++, §6.2.6.***
- ✓ ***Дейтел С++, §4.1– §4.8.***
- ✓ ***Либерти С++21, глава 8.***

1.24.1 Применение операторов new и delete для создания переменной.

1.24.2 Применение операторов new и delete для создания одномерного массива.

1.24.3 Методика проверки корректности операции выделения памяти.

1.24.4 Понятие утечки памяти и способы диагностики.

Практика 31. p3n12 Операторы new и delete-1

Задание. Создайте и инициализируйте несколько одномерных массивов разных типов с помощью оператора new. Используйте оператор delete для разрушения. Исследуйте применение указателей и переменных с индексом для доступа к элементам массивов. Выведите в файл. Исследуйте последствия попытки использования элементов массива до создания и после разрушения.

Задание. Пусть имеется массив int* X = new int[10], инициализированный последовательными значениями 1000, 1100, 1200, 1300 и т.д. Какие значения будут иметь величины X, X[0], X[7], *X, *X+1, *X+7, *(X+7), (*X)+7. Какое значение будет иметь величины X+10000, *(X+10000)?

Задание. Пусть int X=new int[10]; intY=new int[10]; инициализируем массивы X и Y (значения придумайте сами). Что произойдет при исполнении операторов a) X[7]=Y[7], b)

X[0]=Y[0], c) X=Y? Какое значение после этого будут иметь X[7] и Y[7]?

Задание. Что произойдет в результате исполнения операторов int* X = new int[10] и int* X = new int(10)?

Задание. Изучите программу одномерной игры Life, представленной в данном проекте. Найдите место в программе, где определяются правила игры. Измените правила по своему усмотрению так, чтобы игра не вырождалась. Например, установите правило, в соответствии с которым одномерный зверь четной длины бежит влево, в зверь нечетной длины бежит вправо. Продемонстрируйте столкновение.

Задание. Измените программу игры Life так, чтобы одномерный игровой массив был не булевским, а типа int. Создайте свои правила игры.

1.25 Операции с массивами.

- ✓ ***Либерти C++21, глава 7.***
- ✓ ***Дейтел C++, §4.6.***

1.25.1 Копирование массива.

1.25.2 Сортировка массива.

Практика 32. p3n13 Сортировка массива

Задание. Создайте и инициализируйте одномерные массивы различных арифметических типов (int, double, char) с помощью оператора new (не забудьте разрушить). Создайте функцию сортировки. Продемонстрируйте правильность работы. Напишите функцию сортировки, использующую функцию сравнения (например, int compare(int x, int y){if(x<y)return -1; if(x==y)return 0; return 1;} или int compare(int x, int y){if(x<0&&y>0)return -1; if(x>0&&y<0)return 1; return 0;})

Задание. Создайте несколько массивов строк, инициализируйте их названиями ваших любимых музыкальных произведений, именами их авторов, год выпуска, название диска, продолжительность звучания и т.д. Сортируйте эти массивы одновременно по различным критериям, а) имя автора, б) год выпуска и т.д.

Лекция 9. Строки и указатели

- ✓ ***Страуструп C++, §4.3, §5.2.2, §3.5.***
- ✓ ***Дейтел C++, §5.12.***
- ✓ ***Либерти C++21, глава 3, 12.***

1.25.3 Константные строки.

1.25.4 Массивы символов char[]="…"; Символ конца строки.

1.25.5 Строка заданной длины.

1.25.6 Буфер.

1.25.7 Строка переменной длины.

1.25.8 Функции копирования, конкатенации, усечения.

1.25.9 Запрос длины строки.

1.25.10 Доступ к отдельным символам строки.

Практика 33. p3n14 Строки

Задание. Создайте несколько константных строк. Создайте буфер. Примените операции копирования и конкатенации. Выведите в файл.

Задание. Создайте строку с помощью оператора new (не забудьте разрушить). Задайте значение с помощью функции генерирования случайных чисел. Проведите сортировку символов в строке.

Задание. Напишите функцию swap(char* X, char* Y) для обмена значений X и Y. Не забудьте, что при копировании строки большей длины в буфер меньшей длины потребуется переопределить буфер (разрушить старый и создать новый). Исследуйте возможности функции swap(int** ppx, int** ppy), которая обменивает значения *ppx и *ppy.

Лекция 10. Двумерные массивы.

1.26 Двумерные массивы.

- ✓ ***Страуструп C++, §5.3.***
- ✓ ***Дейтел C++, §4.9.***
- ✓ ***Либерти C++21, глава 8, 12.***

1.26.1 Массивы указателей.

1.26.2 Понятие двумерного массива с фиксированной размерностью.

1.26.3 Доступ к элементам двумерного массива.

1.26.4 Вывод двумерного массива в файл.

Практика 34. p3n15 Двумерные массивы

Задание. Создайте и инициализируйте двумерные массивы типов int и double. Создайте функцию сортировки. Создайте функцию вычисления наименьшего и наибольшего значений по двумерному массиву и отдельно по строкам и столбцам. Продемонстрируйте правильность работы.

Задание. Напишите функцию для печати в файл содержимого массива char X[M][N]. Нарисуйте график функции y=f(x) с помощью двумерного массива char[128][128].

Задание. Изучите программу Life. Переделайте игровой массив на тип bool (данное упражнение носит чисто учебный характер и позволит Вам усовершенствовать Ваше понимание типов C++). Запрограммируйте другие правила игры, используя информацию из Wikipedia. Измените программу так, чтобы можно было задавать матрицу правил размера 5x5 (вместо 3x3 в оригинале, данное упражнение носит чисто учебный характер и позволит Вам усовершенствовать Ваше понимание типов C++). Напишите программу, которая будет отслеживать путешествие стандартного life-организма по большому полю, размеры которого много больше, чем размеры экрана. Задайте случайное распределение начальных оккупированных позиций по игровому полю. Напишите код для анализа динамики игры (ширина и высота life-организма, его вес, т.е. число оккупированных позиций, время жизни, интенсивность жизненного цикла, которую определить догадайтесь как сами). Смоделируйте большое количество случайных начальных состояний и отберите из них наиболее жизнеспособные по критерию, который установите самостоятельно. Результаты фиксируйте в файлах начальных данных. После этого исследуйте визуально наиболее жизнеспособные состояния.

Задание. * (более сложное). Смоделируйте трехмерную игру.

1.27 Операторы new и delete для двумерных массивов.

- ✓ ***Страуструп C++, §5.3.***
- ✓ ***Дейтел C++, §6.10, 6.12–6.13.***
- ✓ ***Либерти C++21, глава 8.***

1.27.1 Применение операторов new и delete для создания двумерного массива.

1.28 Передача двумерного массива в функцию.

Практика 35. Операторы new и delete-2

Задание. Создайте и инициализируйте несколько двумерных массивов разных типов с помощью оператора new. Используйте delete для разрушения. Исследуйте применение указателей и переменных с индексом для доступа к элементам массивов. Выведите в файл. Исследуйте последствия попытки использования элементов массива до создания и после разрушения.

Лекция 11. Указатели на функции

1.29 Указатели на функции.

- ✓ ***Страуструп C++, §7.7.***
- ✓ ***Дейтел C++, §5.9, §5.11.***

- ✓ ***Либерти C++21, глава 8.***

1.29.1 Указатели на функции.

1.29.2 Массив указателей на функции.

Практика 36. Указатели на функцию

Задание. Создайте несколько функций и используйте указатели для их вызова. Продемонстрируйте правильность работы.

Глава 4 Ввод и вывод

Лекция 12. Форматированный вывод в файл

1.30 Манипуляторы при выводе.

- ✓ ***Страуструп C++, §21.1-21-4.***
- ✓ ***Дейтел C++, §11.5, §11.6.***
- ✓ ***Либерти C++21, глава 16.***

1.30.1 Установка ширины поля вывода.

1.30.2 Установка точности.

1.30.3 Обзор манипуляторов.

Практика 37. p4n18 Форматированный вывод в файл

Задание. Создайте таблицу (двумерный массив) и выведите в файл с нумерацией строк, столбцов. Прочитайте текстовый документ (отрывок из книги, предоставленный преподавателем). Составьте словарь. Найдите частоту повторения каждого слова в документе. Выдайте документ в файл в форматированном виде (выровненном влево, вправо, центрированном, выровненном по ширине).

Лекция 13. Форматированный ввод из файла

1.31 Манипуляторы при вводе.

- ✓ ***Страуструп C++, §21.1-21.4.***
- ✓ ***Дейтел C++, §11.5, §11.6.***
- ✓ ***Либерти C++21, глава 16.***

1.31.1 Ввод встроенных типов.

1.31.2 Функции распознавания образов.

Практика 38. Форматированный ввод из файла

Задание. Создайте таблицу (двумерный массив) и выведите в файл с нумерацией строк, столбцов. Прочитайте и проверьте правильность. Прочитайте таблицу смешанных типов (текст-данные).

Лекция 14. Компоновка проекта

1.32 Создание проекта с несколькими файлами.

- ✓ ***Страуструп C++, §9.1-9.6, §4.9.***

1.32.1 Основные и заголовочные файлы.

1.32.2 Область видимости объекта.

1.32.3 Время жизни объекта.

1.32.4 Блок.

1.32.5 Обеспечение доступа к объекту из нескольких файлов.

1.32.6 Пять ключевых точек кода:

Объявление объекта
Определение объекта
Инициализация объекта
Использование объекта
Разрушение объекта

Практика 39. Компоновка проекта

Задание. Создайте проект с несколькими h-файлами и несколькими cpp-файлами для сортировки одномерного массива так, чтобы функция сортировки была определена в одном файле, а массив подлежащий сортировке в другом. Исследуйте последствия определения двух одноименных обьектов в разных файлах. Исследуйте применение блоков.

Глава 5 Обьектно ориентированное программирование на С++.

Лекция 15. Понятие класса

1.33 Понятие класса.

- ✓ ***Страуструп С++, §10.2.***
- ✓ ***Дейтел С++, §6.1-§6.5.***
- ✓ ***Янг, часть I, глава 4,***
- ✓ ***Либерти С++21, глава 6,***

1.33.1 Понятие класса. Встроенные классы и классы пользователя.
1.33.2 Протокол класса.
1.33.3 Данные-члены (поля) и функции-члены.
1.33.4 Открытые и закрытые члены класса.
1.33.5 Дружественные функции.

1.34 Работа с объектами.

- ✓ ***Страуструп С++, §10.4.***
- ✓ ***Дейтел С++, §6.1-§6.5.***
- ✓ ***Янг, часть I, глава 4,***
- ✓ ***Либерти С++21, глава 6,***

1.34.1 Объявление и описание объекта данного класса.
1.34.2 Доступ к данным и функциям для объектов и указателей на объекты.
1.34.3 Соглашение об именах классов, объектов и указателей.

Практика 40. p5n21-Создание простого класса

Задание. Создайте протокол своего класса (например, CMyCar , мой любимый автомобиль) с несколькими полями данных (например, фирма, название, год выпуска, пробег и т.д.). Создайте объект, указатель на объект. Инициализируйте указатель. Выдайте в файл.

1.35 Способы доступа к полям данных.

- ✓ ***Страуструп С++, §10.4.***
- ✓ ***Дейтел С++, §14.4-14.6, §6.14,***
- ✓ ***Либерти С++21, глава 6.***

1.35.1 Способы доступа к членам класса. Интерфейсные функции.
1.35.2 Интерфейсные функции типа get(…) .
1.35.3 Интерфейсные функции типа set(…) .
1.35.4 Методы валидации параметров.
1.35.5 Выдача протокола работы программы в файл.

1.36 Обеспечение безопасности

- ✓ ***Страуструп С++, §10.2.***

1.36.1 Константные объекты и константные функции-члены.

1.36.2 Статические данные-члены и статические функции-члены.

Практика 41. Функции доступа

Задание. Создайте функции доступа. Измените значение полей данных. Выдайте в файл с помощью функций доступа. Создайте friend функцию.

Лекция 16. Конструкторы и деструкторы

1.37 Конструкторы и деструкторы.

- ✓ ***Страуструп С++, §10.2, 10.4.***
- ✓ ***Янг, часть I, глава 4,***
- ✓ ***Дейтел С++, §6.10, 6.11, 6.14,***
- ✓ ***Либерти С++21, глава 6.***

1.37.1 Конструирование объекта.

1.37.2 Принципы работы с указателями на объект класса пользователя.

1.37.3 Инициализирующий конструктор (конструктор с параметрами).

1.37.4 Конструктор с аргументами по умолчанию. Перегрузка конструктора.

Практика 42. Конструктор и деструктор

Задание. Создайте конструктор и деструктор CMyCar. Добавьте инициализирующий конструктор (с параметрами).

1.38 Конструирование объектов, включающих массивы.

- ✓ ***Страуструп С++, §10.4.6.***

1.38.1 Конструирование и разрушение объектов, включающих поля данных – массивы с постоянной размерностью.

1.38.2 Конструирование и разрушение объектов, включающих поля данных – массивы с переменной размерностью.

Практика 43. Конструктор и деструктор

Задание. Создайте конструктор и деструктор класса, имеющего поле данных – массив с постоянной размерностью.

Лекция 17. Конструктор копирования

1.39 Копирование объектов.

- ✓ ***Страуструп С++, §10.2, 10.4.***
- ✓ ***Дейтел С++, §6.01-6.11, §8.1-8.3,***
- ✓ ***Либерти С++21, глава 6, 10.***

1.39.1 Применение указателя this.

1.39.2 Перегрузка оператора копирования =.

1.39.3 Операция swap(…, …).

1.39.4 Конструктор копирования.

Практика 44. Конструктор копирования

Задание. Создайте конструктор копирования CMyCar. Добавьте в протокол и реализацию класса CMyTime перегруженный оператор “=”. В класс CMyWork добавьте метод обмена значений двух полей.В класс CMyTime Добавьте метод обмена значений двух объектов класса CMyTime, используя промежуточный обьект того же класса.

Лекция 18. Массив объектов и массив–поле данных

1.40 Массивы объектов класса.

- ✓ *Страуструп С++, §10.4.*

1.40.1 Массив фиксированного размера объектов класса.

1.40.2 Доступ к элементам массива.

1.40.3 Массив переменного размера объектов класса.

1.41 Массив – поле данных класса пользователя.

- ✓ *Страуструп С++, §10.4.*

1.41.1 Массив фиксированного размера – поле данных класса.

1.41.2 Массив переменного размера – поле данных класса.

Практика 45. Массив объектов и массив-поле данных

Задание. 1. Создайте массив объектов класса CMyTime. Инициализируйте конструктором по умолчанию. Задайте значения полей (числовых и текстовых) явным образом. Создайте интерфейс для просмотра значений каждого объекта. Не забудьте разрушить массив перед завершением программы.

Задание. 2. Измените протокол и реализацию класса CMyTime так, чтобы поле данных, являющееся одномерным массивом, создавалось динамически, причем размер массива равен заданному постоянному значению. Добавьте проверку превышения размера массива с соответствующей диагностикой.

Лекция 19. Перегрузка операций

1.42 Перегрузка унарных операций.

- ✓ *Страуструп С++, §11.2.*
- ✓ *Либерти С++21, глава 10.*
- ✓ *Дейтел С++, §8.4-§8.8.*

1.42.1 Перегрузка операции сравнения.

1.42.2 Перегрузка операции сравнения, пример: сортировка.

1.42.3 Перегрузка унарных операций.

Практика 46. Перегрузка унарных операций

Задание. Добавьте в протокол и реализацию класса CMyTime перегруженные операторы сравнения “>”, и “==”, и “!=”. Добавьте в протокол и реализацию класса CMyWork метод сортировки, использующий перегруженный оператор сравнения. Проверьте правильность работы сортировки. Добавьте в протокол и реализацию класса CMyTime перегруженные операторы «!» и «++». Проверьте правильность работы. Добавьте в протокол и реализацию класса CMyTime перегруженный оператор сравнения “<”. Проверьте работу оператора для нескольких объектов класса.

1.43 Перегрузка бинарных операций

- ✓ *Страуструп С++, §11.2.*
- ✓ *Дейтел С++, §8.4-§8.8.*

1.43.1 Перегрузка бинарных операций для двух операндов одного типа.

1.43.2 Перегрузка бинарных операций для операндов смешанного типа.

Практика 47. Перегрузка бинарных операций

Задание. 1. Изучите перегруженные бинарные операторы с операндами, если оба являются объектами класса пользователя. Изучите перегруженные бинарные операторы с операндами, один из которых является объектом класса пользователя, а другой – объектом встроенного класса. Изучите перегруженные бинарные операторы с операндами, первый из которых является объектом класса пользователя, а второй – объектом другого класса пользователя. Добавьте в протокол и реализацию класса CMyTime

перегруженные friend операторы “+”, “-”, “*”. Проверьте правильность работы.

Задание. 2. Добавьте в протокол и реализацию класса CMyTime перегруженные friend операторы “+”, “-”, “*” для операндов смешанного типа, например CMyTime и int. Проверьте правильность работы.

1.44 Преобразование типов.

1.44.1 Преобразование типов для объектов встроенных классов.

1.44.2 Перегрузка операций преобразования типов.

Практика 48. Перегрузка операции преобразования типа

- ✓ ***Страуструп C++, §11.4.***
- ✓ ***Дейтел C++, §8.9.***

Задание. Добавьте в протокол и реализацию класса CMyTime перегруженные операции преобразования типа int(CMyTime) и CMyTime(double). Проверьте правильность работы.

1.45 Перегрузка операции помещения в поток и взятия из потока.

- ✓ ***Дейтел C++, §11.3,***

1.45.1 Перегрузка операции помещения в поток и взятия из потока.

Задание: Добавьте в протокол и реализацию класса CMyTime перегруженные операторы “<<”, “>>”. Проверьте правильность работы.

Литература: Дейтел C++, §8.5.

Лекция 20. Примеры классов

1.46 Примеры классов, обладающих большой степенью функциональности.

- ✓ ***Дейтел C++, §8.8, 8.10.***

1.46.1 Пример: класс array.

1.46.2 Пример: класс complex.

1.46.3 Пример: класс string для обработки строк.

Практика 49. Исследование класса array

Задание. Создайте класс array для хранения одномерного массива, оснащенный конструктором, деструктором, арифметическими операторами, операторами сравнения, копирования, сортировки, выдачи в файл. Напишите операторы вычисления минимума, максимума, среднего значения и дисперсии. Проверьте правильность работы.

Глава 6 Производные классы

Лекция 21. Классы-контейнеры

1.47 Понятие класса-контейнера.

- ✓ ***Дейтел C++, §7.9, 7.10.***

1.47.1 Принципы наследования: контейнеризация (классы-контейнеры).

Задание 1. Используя AppWizard, сформируйте solution и найдите пять критических точек (объявление, определение, инициализация, использование, разрушение).

Задание 2. Подключите h-файл потокового вывода в файл. Объявите и создайте объект потока вывода в файл, инициализируйте, проведите тестовую выдачу, обеспечьте разрушение.

Задание 3. Используя ClassVizard, создайте протокол и реализацию класса CMyPoint, включающего три поля данных встроенного типа (по указанию преподавателя) с атрибутом public. Оснастите указанный класс default-конструктором и параметрическим конструктором, конструктором копирования, перегруженным оператором вывода в поток.

Задание 4. Создайте два объекта класса CMyPoint, без инициализации и с инициализацией, проведите тестовую выдачу.

Задание 5. Создайте два указателя на объекты класса CMyPoint. Создайте два объекта класса с помощью оператора new, без инициализации и с инициализацией. Проведите тестовую выдачу.

Задание 6. Используя ClassVizard, создайте протокол и реализацию класса CMyCirc, включающего поле данных встроенного типа (по указанию преподавателя) и поле данных класса CMyPoint. Оснастите указанный класс default-конструктором и несколькими параметрическим конструкторами, в том числе с параметром типа CMyPoint. Оснастите указанный класс конструктором копирования, перегруженным оператором вывода в поток. Проведите тестовую выдачу.

Литература: Дейтел С++, §9.12, §9.13, §9.14.

Лекция 22. Классы-итераторы

1.48 Понятие класса-итератора. Наследование.

Проект: p2n53. Источник: AppWizard

✓ ***Дейтел С++, §7.9, 9.1, 9.2, 9.3.***

Архивированный проект

1.48.1 Принципы наследования: классы-итераторы.

Задание 1. Удалите поле данных типа CMyPoint и сделайте класс CMyCirc производным классом базового класса CMyPoint. Оснастите указанный класс default-конструктором и несколькими параметрическим конструкторами, в том числе с параметром типа CMyPoint. конструктором копирования, перегруженным оператором вывода в поток. Проведите тестовую выдачу.

Задание 2: Исследуйте последствия использования конструктора по умолчанию при конструировании производного класса.

Литература: Дейтел С++, §9.1, §9.2, §9.9.

1.48.2 Статические поля данных в базовом классе.

Задание 1. Добавьте статическое поле данных static int с именем count в базовый класс CMyPoint. Обеспечьте подсчет числа создаваемых любым образом объектов базового класса и подсчет числа разрушаемых объектов. Создайте поле данных для хранения номера данного объекта базового класса. Проведите тестовую выдачу. Проверьте количество объектов базового класса после разрушения всех объектов, созданных с помощью функции new().

Литература: Дейтел С++, §9.1, §9.2, §9.9.

Лекция 23. Открытые, защищенные, закрытые данные-члены.

1.49 Понятие уровня защиты поля данных и функции (метода) класса.

✓ ***Дейтел С++, §9.7, 9.8.***

1.49.1 Доступ к полям данных и методам базовых классов из объекта производного класса. Открытые, защищенные, закрытые данные-члены.

Задание 1: Добавьте в базовый класс поля данных public, protected и private. Проверьте возможность доступа к этим полям данных из объекта производного класса.

Литература: Дейтел С++, §9.1, §9.2, §9.9.

1.49.2 Открытые, защищенные, закрытые базовые классы.

Задание 1: Исследуйте последствия объявления базового класса как public, protected и private. Проверьте возможность доступа к полям данных базового класса из объекта производного класса.

Литература: Дейтел С++, §9.7.

Лекция 24. Дружественные функции и дружественные

классы.

1.50 Реализация доступа к полям данных и методам из других классов.

✓ ***Дейтел С++, §7.1-7.5.***

1.50.1 Дружественные функции и дружественные классы.

Задание 1: Изучите реализацию перегруженного оператора помещения в поток как friend функции. Изучите последствия определения выводимого поля данных как public, protected и private. Сделайте вывод о том, как правильно определить атрибуты поля данных, чтобы его можно было использовать из friend функции.

Литература: Дейтел С++, §9.1, §9.2, §9.9.

1.50.2 Приведение типов указателей базовых классов к указателям производных классов.

Задание 1: Создайте указатель на объект базового класса и инициализируйте его значением указателя на объект производного класса. Проверьте результат вызова функции f() для объекта, указываемого данным указателем.

Литература: Дейтел С++, §9.1, §9.2, §9.9.

1.50.3 Переопределение функций-членов в производных классах.

Задание 1: Добавьте в базовый класс функцию f().Добавьте в производный класс функцию f() с тем же именем и тем же набором параметров, но возвращающую другое значение. Проверьте результаты вызова каждой из этих функций из объектов базового и производного классов.

Литература: Дейтел С++, §9.1, §9.2, §9.9.

Лекция 25. Множественное наследование.

1.51 Множественное наследование.

✓ ***Дейтел С++, §9.1-9.6.***

1.51.1 Классы, представляемые простым графом типа дерево.

Проект: p2n55. Источник: p2n52. ***Архивированный проект***

Задание: Исследуйте множественное невиртуальное наследование на примере класса CMyBike, производного от CMyCirca и CMyCircb, каждый из которых невиртуально порожден от класса CMyPoint.

Литература: Дейтел С++, §9.1, §9.2, §9.9, Либерти С++21, Гл.13

1.51.2 Виртуальные базовые классы.

Проект: p2n56. Источник: p2n55. ***Архивированный проект***

Задание: Исследуйте множественное виртуальное наследование на примере класса CMyBike, производного от CMyCirca и CMyCircb, каждый из которых виртуально порожден от класса CMyPoint. Исследуйте последствия отмены инициализации полей CMyCirca и CMyCircb.

Литература: Либерти С++21, Гл.13

1.51.3 Виртуальные базовые классы.

Проект: p2n56m. Источник: p2n56. ***Архивированный проект***

Задание: Исследуйте ошибочное применение множественного наследования на примере класса CMyBike, производного от CMyCirca и CMyCircb, один из которых порожден невиртуально, а другой виртуально порожден от класса CMyPoint. Исследуйте последствия отмены инициализации полей CMyCirca и CMyCircb.

Литература: Либерти С++21, Гл.13

1.52 Конструирование и разрушение сложных объектов.

✓ ***Дейтел С++, §9.9-9.6.***

1.52.1 Порядок конструирования при одиночном наследовании.

1.52.2 Порядок конструирования при множественном наследовании.

Глава 7 Виртуальные функции и полиморфизм

Лекция 26. Виртуальные функции и полиморфизм.

1.53 Виртуальные функции и полиморфизм.

✓ ***Дейтел C++, §10.1-10.9.***

1.53.1 Виртуальные функции.

Проект: p2n61. Источник: p2n54.
Цель 1: понять разницу между виртуальной функцией базового класса и невиртуальной функцией базового класса, переопределяемыми в производном классе. ***Цель 2:*** Обратить внимание на неправильное использование деструкторов. Найдите сообщение отладчика об утечке памяти, найдите место в программе, дающее утечку памяти.
Задание: Создайте собственную версию кода с базовым классом, несколькими производными классами и виртуальными и невиртуальными переопределяемыми функциями.
Комментарий: результат см. в файле Protocol.txt.
Литература: Либерти C++21, Гл.11.

1.53.2 Виртуальные деструкторы.

Проект: p2n62. Источник: p2n61.
Цель: понять способ устранения неправильного вызова деструкторов для объектов производных классов через указатели на обьекты базового класса.
Задание: Создайте собственную версию кода с базовым классом, несколькими производными классами, с виртуальными и невиртуальными деструкторами.
Комментарий: результат см. в файле Protocol.txt.
Литература: Либерти C++21, Гл.11.

1.53.3 Абстрактные классы.

Проект: p2n63. Источник: p2n62.
Цель: понять идеологию абстрактного базового класса.
Задание: Создайте собственную версию кода с абстрактным базовым классом, несколькими производными классами, с виртуальными и невиртуальными деструкторами. Для начала замените функции базового класса в предыдущем проекте на абстрактные (чистые) виртуальные функции, получите диагностику компилятора и затем исправьте ошибку.
Комментарий: результат см. в файле Protocol.txt.
Литература: Либерти C++21, Гл.13.

1.53.4 Выполняемые абстрактные (чистые) виртуальные функции.

Для самостоятельного изучения, на экзамен не выносится.
Литература: Либерти C++21, Гл.13.

1.53.5 Иерархия абстрактных классов.

Проект: p2n65. Источник: p2n63.
Версия: 2007-12-03 (#03).
Цель: понять идеологию иерархии абстрактных классов.
Задание: Создайте собственную версию кода с абстрактным базовым классом, абстрактным производным классом, несколькими производными классами, с виртуальными и невиртуальными деструкторами.
Комментарий: результат см. в файле Protocol.txt.
Литература: Либерти C++21, Гл.13.

1.53.6 Раннее и позднее связывание.

Лекция 27. Потоки ввода-вывода.

1.54 Потоки ввода-вывода.

✓ ***Дейтел C++, §11.1-11.8.***

1.54.1 Понятие файлового потока вывода.

Проект p7n62. Источник: AppWizard. Данный проект создан MVS с опцией «Win32 App».
Цель: понять общие принципы направления ввода и вывода из потока (в поток).
Задание: Создать обьект типа ofstream и связать его с файлом на диске. Вывести в поток ofstream обьекты типа char[], int, double.
Литература: Дейтел С++, §11.1, §11.2, §11.9.

1.54.2 Вывод в файловый поток протокола работы программы.

Проект p7n63. Источник: p2n65 (библиотека классов), p7n62 (работа с потоком). Данный проект передан в p2n65
Задание: Направить протокол работы программы в поток вывода, связанный с файлом, используя для этого класс ofstream.
Литература: Дейтел С++, §11.1, §11.2, §11.9.

1.54.3 Вывод в файловый поток данных класса пользователя.

Проект p7n64. Источник: p7n63.
Задание:
Литература: Дейтел С++, §11.1, §11.2, §11.9.

1.54.4 Манипуляторы потока.

Задание:
Литература: Дейтел С++, §11.6.

1.54.5 Форматирование при выводе.

Задание:
Литература: Дейтел С++, §11.7.

1.54.6 Обработка ошибок.

Задание:
Литература: Дейтел С++, §11.8.

1.54.7 Форматированный ввод из файла.

Задание: Создать файл с помощью ofstream<<
и прочитатьь его с помощью <<ifstream.
Литература: Дейтел С++, §11.9.

1.55 Шаблоны.

✓ ***Дейтел С++, §12.1-12.8.***

1.55.1 Шаблоны классов.

Проект p2n71. Источник: p2n56.
Задание: Создайте класс CMyStack, используя как образец код Дейтел С++, §12.4. Проверьте для объектов классов CMyBike, CMyCirca, CMyPoint, CMyTime.
Литература: Дейтел С++, §12.1 - §12.4, Либерти С++21, Гл.19.

1.55.2 Шаблоны функций.

1.56 Работа с файлами.

✓ ***Дейтел С++, §14.1-14.13.***

1.56.1 Открытие и закрытие файлов.

1.56.2 Файлы последовательного доступа.

1.56.3 Файлы произвольного доступа.

1.57 Обработка исключений.

Глава 8 Библиотека шаблонов

Лекция 28. Библиотека шаблонов.

1.58 STL (Standard Template Library), абстракция данных.

✓ ***Дейтел С++, §15.1-15.7.***

1.58.1 Абстрактный тип данных «vector», реализация шаблона «vector» с параметром <int>.

Проект: p2n72. Источник: p2n31.
Цель: Изучить принципы организации STL. Изучить понятие контейнера и итератора. Изучить контейнер vector. Изучить реализацию контейнера vector для встроенного типа данных int.
Задание: Добавьте Button «Vector» в DialogBox. Добавьте создание, инициализацию, операции с контейнером vector<int> и операторы тестирования с выводом в файл. Продемонстрируйте

1) операции добавления в конец вектора,
2) действие итератора,
3) действие обратного итератора,
4) методы insert, erase, clear.
5) Действие операции копирования вектора с помощью операции =.

Комментарий: результат см. в файле Protocol.txt.
Литература: Дейтел С++, §20.1, §20.2, Янг, том I, глава 7, Либерти С++21, Гл.19.

1.58.2 Абстрактный тип данных «vector», реализация шаблона с параметром пользовательского типа <CMyTime>.

Проект: p2n73. Источник: p2n72.
Цель: Изучить контейнер vector объектов пользовательского класса.
Задание: Проделайте то же самое для контейнера объектов класса CMyTime.
Комментарий: результат см. в файле Protocol.txt.
Литература: Дейтел С++, §20.1, §20.2, Янг, том I, глава 7, Либерти С++21, Гл.19.

1.58.3 Абстрактный тип данных «deque», реализация шаблона с параметром пользовательского типа <CMyTime>.

Проект: p2n74. Источник: p2n73.
Цель: Изучить контейнер deque объектов пользовательского класса.
Задание:

6) Замените template vector на template deque для контейнера объектов класса CMyTime.
7) Одна из функций, определенная для vector, неопределена для deque. Найдите эту функцию и закомментируйте соответствующие операторы. После этого программа заработает.
8) Добавьте операцию расширения deque с начального конца.
9) Проверьте действие оператора [] для объекта deque.

Комментарий: результат см. в файле Protocol.txt.
Литература: Дейтел С++, §20.1, §20.2, Янг, том I, глава 7, Либерти С++21, Гл.19.

1.58.4 Абстрактный тип данных «list», реализация шаблона с параметром пользовательского типа <CMyTime>.

Проект: p2n75. Источник: p2n74.
Цель: Изучить контейнер list объектов пользовательского класса.
Задание:

10) Замените template deque на template list для контейнера объектов класса CMyTime.
11) Добавьте операцию сортировки sort() и операцию удаления копий unique().

Комментарий: результат см. в файле Protocol.txt.
Литература: Дейтел С++, §20.1, §20.2, Янг, том I, глава 7, Либерти С++21, Гл.19.

1.58.5 Абстрактный тип данных «стек».

Проект: p2n71. Источник: p2n56.
Цель: d.
Задание: a.
Литература: Янг, часть II, глава 16, Круглински, Уингоу, Шефферд, глава 2, Либерти

Глава 9 Графический интерфейс

Лекция 29. Графический интерфейс.

1.59 Программная оболочка для изучения обьектно ориентированного программирования на языке С++.

1.59.1 Dialog based application.

Проект: p2n01. ***Архивированный проект***
Цель: Изучить Microsoft Visual Studio, проект MFC-Dialog based.
Задание: Создайте в диалоговой панели элемент управления типа Static Text и внесите в него информацию об авторе (Ваши фамилию, имя, отчество, номер группы) Создайте еще один элемент управления того же типа и занесите в него дату выполнения. Создайте еще один элемент управления того же типа и занесите в него тему данного задания.
Литература: Янг, часть I, главы 1, 2, часть II, глава 16, Круглински, Уингоу, Шефферд, главы 1, 2.
Файл: p2n01.doc, файлы проекта с выделенными изменениями.

1.59.2 Элемент управления типа Static Text.

Проект: p2n02, источник: p2n01. ***Архивированный проект***
Цель: Изучить Microsoft Visual Studio, проект MFC-Dialog based.
Задание: Создайте в диалоговой панели элемент управления типа Static Text и внесите в него информацию об авторе (Ваши фамилию, имя, отчество, номер группы) Создайте еще один элемент управления того же типа и занесите в него дату выполнения. Создайте еще один элемент управления того же типа и занесите в него тему данного задания.
Литература: Янг, часть II, глава 16, Круглински, Уингоу, Шефферд, глава 2.

1.59.3 Элемент управления Button, button RUN, изменение значенией переменных в функции OnRun().

Проект: p2n03, источник: p2n02. ***Архивированный проект***
Цель: Изучить программирование и обработку сообщений элемента управления Button.
Задание: Добавьте Button «RUN» и напишите функцию обработчик OnRun.
Литература: Янг, часть II, глава 16, Круглински, Уингоу, Шефферд, глава 6.
Файл: p2n03.doc, файлы проекта с выделенными изменениями.

1.59.4 Создание статических элементов управления и редактируемых эленментов управления.

Проект: p2n04, источник: p2n03.
Цель: Изучить статические элементы управления и редактируемые элементы управления. Инициализация, чтение значений.
Задание: Создайте набор статических и редактируемых элементов управления, инициализируйте, допишите операторы обмена данных. Проверьте работу диалоговой панели.
Литература: Янг, часть II, глава 16, Круглински, Уингоу, Шефферд, глава 2.
Файл: p2n04.doc, файлы проекта с выделенными изменениями.

1.59.5 Инициализация переменных в функции InitDialod().

Проект: p2n04m, источник: p2n04.
Цель: Понять правильную тактику инициализации полей данных.
Задание: Перенести инициализацию ф функцию InitDialog().

- ✓ ***Янг, часть II, глава 16,***
- ✓ ***Круглински, Уингоу, Шефферд, глава 2.***

Файл: p2n04m.doc, файлы проекта с выделенными изменениями.

1.59.6 Чтение информации с диалоговой панели.

Проект: p2n5, источник: p2n04m.

Цель: Изучить метод чтения информации с диалоговой панели.
Задание: Создайте Edit Box и добавьте функцию чтения информации с диалоговой панели. Дописать операторы обмена данных.
Литература: Янг, часть II, глава 16, Круглински, Уингоу, Шефферд, глава 2.
Файл: p2n05.doc, файлы проекта с выделенными изменениями.

1.59.7 Отображение информации на диалоговой панели.

Проект p2n5m, источник: p2n05.
Цель: Изучить метод отображения информации на диалоговой панели.
Задание: Добавьте функцию отображения информации на диалоговой панели.
Литература: Янг, часть II, глава 16, Круглински, Уингоу, Шефферд, глава 2.
Файл: p2n05m.doc, файлы проекта с выделенными изменениями.

Глава 10 Программирование MFC.

Лекция 30. Программирование MFC.

1.60 Методика встраивания класса пользователя в MFC приложение.

Проект: p2n34, взаимодействие класса и MFC.

1.61 Основные операторы вывода информации.

1.61.1 Редактор ресурсов. Понятие Static box и Edit box.

Проект p2n5m, источник: p2n05. 1) Помещение данных в DialogBox, 2) извлечение отредактированных данных из DialogBox, 3) навешивание функции-обработчика на Button в DialogBox. Простейший калькулятор.

1.61.2 Простейший набор операторов форматирования.

1.61.3 Элементарные операции со строками.

1.61.4 Операторы графического вывода. Рисование линий и простейших форм.

1.61.5 Методика встраивания класса пользователя в MFC приложение, 1.

Цель: понять организацию взаимодействия класса пользователя и MFC проложения.
Задание: Добавьте обработка сообщения OnDestroy, протокол и реализация класса CMyWork, указатель на и обьявление обьект класса CMyWork. Добавьте конструктор без аргументов для класса CMyTime. Исследуйте последствия удаления из протокола класса конструктора без аргументов. Обьясните причину ошибки компиляции.
Литература: Янг, часть I, глава 4, Либерти С++21, глава 6,
Дейтел С++, §6.1-§6.5. .

1.61.6 Методика встраивания класса пользователя в MFC приложение, 2.

Цель: 1) Изучить методы чтения и отображения объекта пользовательского класса на диалоговой панели.
2) Понять характер поведения public и private полей данных.
Задание: Добавить чтение и отображение значения поля типа пользователя, CmyTime. Исследуйте последствия обьявления полей hou, min, sec как private.
Литература: Янг, часть I, глава 4, Либерти С++21, глава 6,
Дейтел С++, §6.7, §6.8.

1.62 Методика встраивания класса пользователя в MFC приложение.
1.63 Инициализация, управление данными, операторы исполнения.
1.64 Конструирование диалоговой панели.
1.65 Стандартный диалог работы с файлами.
1.66 Управления работой многопоточного приложения. Семафоры и т.д.
1.67 Графическое отображение информации.

Проект p6n43

1.67.1 Программирование простейшего графического класса.

Задание: Создайте класс для рисования графика функции одной переменной. Создайте элемент Picture с атрибутом Frame и и менем IDC_GRAPH1. Создайте ассоциированную переменную m_gr1 категории Control типа CStatic. Подключите графический класс. Нарисуйте график y=sin x. Проверьте правильность работы.
Литература: Дейтел С++, §.

1.67.2 Программирование двумерной графики.
1.67.3 Программирование квазитрехмерной графики.

1.68 Программирование таймера.

Проект p6n45 (базируется на p6n43).
Задание: Создайте класс для рисования графика функции одной переменной. Создайте элемент Picture с атрибутом Frame и и менем IDC_GRAPH1. Создайте ассоциированную переменную m_gr1 категории Control типа CStatic. Подключите графический класс. Нарисуйте график y=sin x. Проверьте правильность работы.
Литература: Дейтел С++.

1.69 Многопоточные приложения. Программирование интерфейсного потока.
1.70 Многопоточные приложения. Программирование рабочих потоков.

Глава 11 Программирование SDI и MDI приложений.

Лекция 31. Приложения SDI и MDI.

1.71 Архитектура Документ-Вид.
1.72 Создание SDI приложений.

1.72.1 Программирование простейшего SDI приложения с элементами отображения.

Проект p3n21 (базируется на AppWizard).
Задание: Создайте простейшее SDI приложение.
Литература: Круглински, Уингоу, Шефферд, гл. 3, проект Ex03a.

1.72.2 Программирование SDI приложения с диалогом.

Проект p3n22 (базируется на p3n21).
Задание: Создайте простейшее SDI приложение.
Литература: Круглински, Уингоу, Шефферд, гл. 3, проект Ex03a.

1.73 Реализация представления данных.
1.74 Реализация документа.

Проект p5n35a-Mandel. Источник: Янг, проект Mandel.

Задание: Изучите SDI приложение на базе Document-View-архитектуры, оснащенное графическим выводом двумерного массива. Данное приложение иллюстрирует применение графического вывода для изображения так называемого фрактала.
Литература: Янг.

1.75 Сериализация. Хранение документа на внешнем носителе.

1.75.1 Понятие сериализации.

Проект p3n23-ex17a. Источник: Круглински, Уингоу, Шефферд, CD, проект ex16b.
Задание: Изучите SDI приложение, оснащенное функцией сериализации документа.
Литература: Круглински, Уингоу, Шефферд, глава 17.

1.75.2 Применение сериализации для сохранения объектов класса пользователя.

Проект p3n24. Источник: p3n23-ex17a.
Задание: Изучите SDI приложение, оснащенное функцией сериализации документа.
Литература: Круглински, Уингоу, Шефферд, глава 17.

1.76 Панели инструментов и строка состояния.

1.77 Перемещаемые панели.

1.78 Создание MDI приложений.

1.79 Многопоточность в SDI и MDI приложениях.

Проект p5n35b-MandelMT. Источник: Янг, проект MandelMT.
Задание: Изучите MT-SDI приложение на базе Document-View-архитектуры, оснащенное графическим выводом двумерного массива. Данное приложение иллюстрирует применение графического вывода для изображения так называемого фрактала.
Литература: Янг.