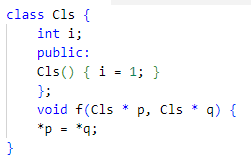
2. Перегрузка операций

Задание 1

1. Даны описание класса и функции:

Дополните описание класса Cls (не изменяя описание функции f )так, чтобы только описание f стало ошибочным.



class Cls {

int i;

public:

Cls() { i = 1; }

Cls& operator=(const Cls&) = delete;

};

void f(Cls \* p, Cls \* q) {

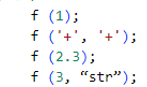
\*p = \*q;

}

Задание 2

Описать прототипы двух перегруженных функций f из некоторой области видимости, для которых будут верны следующие обращения к ним:

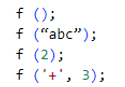
А)



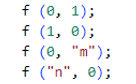
void f(double v=0,string g="");

void f(char z, char v);

б)



void f(char v = ' ', int z = 0);

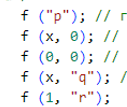
 void f(string v);

с)

void f(string g, int v);

void f(int a, string g);

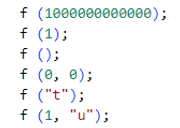
д)



void f(X, const char\* c);

void f(const char\* c);

Е)



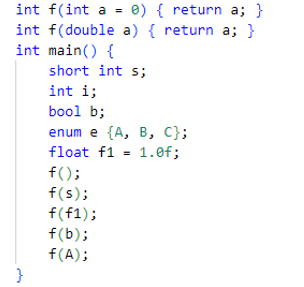
void f(int z = 0, string a = "");

void f(const char\* c);

Задание 3

Для каждого вызова перегруженной функции с одним параметром укажите, какая функция и на каком шаге алгоритма будет выбрана.

А)



f(); //1

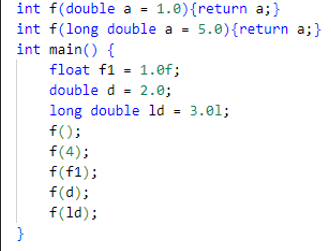
f(s); //1

f(f1); //2

f(b); //1

f(B); //1

б)



f(); //1 и 2 подходят, нужно явно указать какую

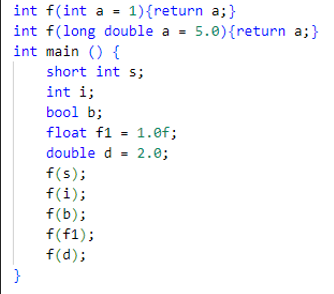
f(4); //1 и 2 подходят, нужно явно указать какую

f(f1); //1

f(d); //1

f(ld); //2

с)



f(s); //1

f(i); //1

f(b); //1

f(f1); //1 и 2 подходят, нужно явно указать какую

f(d); //1 и 2 подходят, нужно явно указать какую

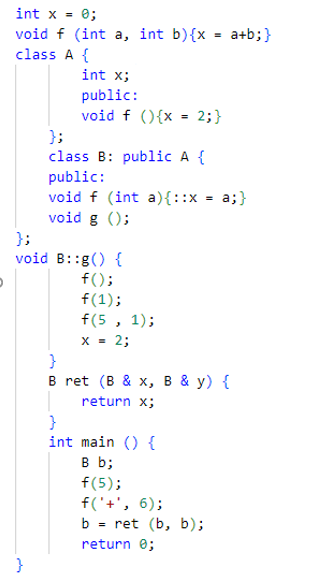
1. Наследование

Задание 1

1. Если есть ошибки в реализации функций B::g () и main (), объясните, в чем они заключаются.

Для всех правильных операторов этих функций с помощью операции разрешения области видимости «::» укажите, из какой области видимости выбираются участвующие в их записи имена.

Какие конструкторы и деструкторы и в каком порядке будут вызываться при работе данной программы?



А)

4. Виртуальные функции

Задание 1

Описать условия включения механизма виртуальности для метода класса. Привести пример записи виртуальной функции и обращения к ней.

Функция для переопределния имеет virtual

Функция переопределяющая имеет тот же возвращаемый тип и тот же конструктор

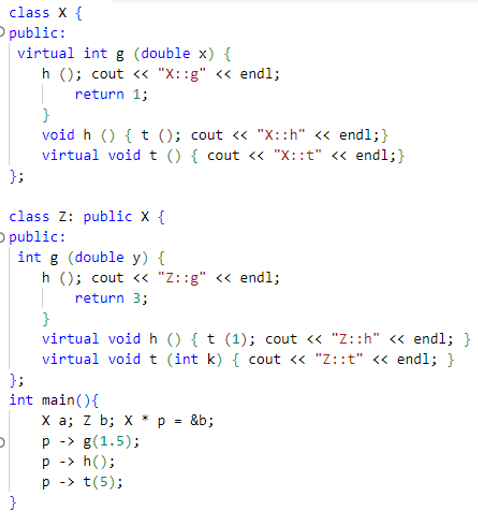
Для явного указания переопределения нужно указать override

virual void print(string x) {cout<< "Вывод "<<x;}

void print(string x) override{cout<< "Вывод "<<x; cout<< x.size() <<" символов";}

Задание 2

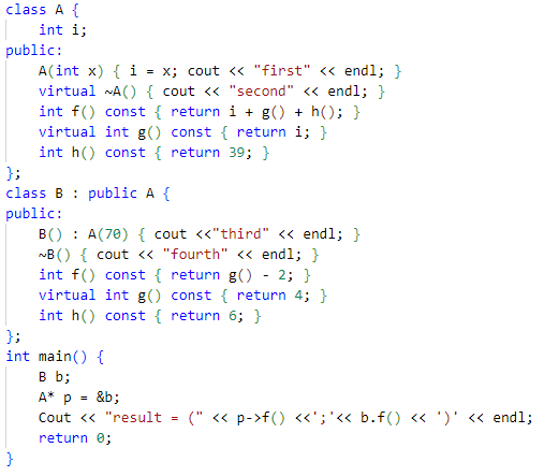
Есть ли ошибки в приведенном фрагменте программы? Если есть, то объясните, в чем они заключаются. Ошибочные конструкции вычеркнуть из текста программы. Что будет выдано в стандартный канал вывода при работе программы?



Добавляем в классе X к h virtual для переопределения, поменять последний t в main

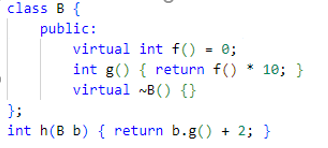
Задание 3

Что напечатает следующая программа?

firstthirdresult = (113;2)fourthsecond

Задание 4

Дан фрагмент программы, есть ли в этом фрагменте ошибки, если да то в чем они заключаются?



Создаётся объект из абстрактного класса

5. Утилиты

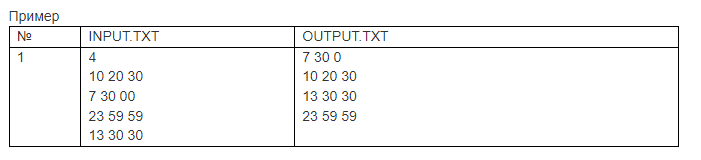
Требуется выполнить сортировку временных моментов, заданных в часах, минутах и секундах.

**Входные данные**

Во входном файле INPUT.TXT в первой строке записано число N (1 ≤ N ≤ 100), а в последующих N строках N моментов времени. Каждый момент времени задается 3 целыми числами - часы (от 0 до 23), минуты (от 0 до 59) и секунды (от 0 до 59).

**Выходные данные**

В выходной файл OUTPUT.TXT выведите моменты времени, упорядоченные в порядке возрастания без ведущих нулей.



ifstream in("INPUT.txt");

ofstream out("OUTPUT.txt");

int n;

in >> n;

int\* h = new int[n];

int\* m = new int[n];

int\* s = new int[n];

for (int i = 0; i < n; i++) {

in >> h[i] >> m[i] >> s[i];

}

for (int i = 0; i < n - 1; ++i) {

for (int j = 0; j < n - i - 1; ++j) {

if (h[j] > h[j + 1] ||

(h[j] == h[j + 1] && m[j] > m[j + 1]) ||

(h[j] == h[j + 1] && m[j] == m[j + 1] && s[j] > s[j + 1])) {

int h2 = h[j];

h[j] = h[j + 1];

h[j + 1] = h2;

int m2 = m[j];

m[j] = m[j + 1];

m[j + 1] = m2;

int s2 = s[j];

s[j] = s[j + 1];

s[j + 1] = s[j];

}

}

}

for (int i = 0; i < n; i++) {

out << h[i] << ' ' << m[i] << ' ' << s[i] << "\n";

}

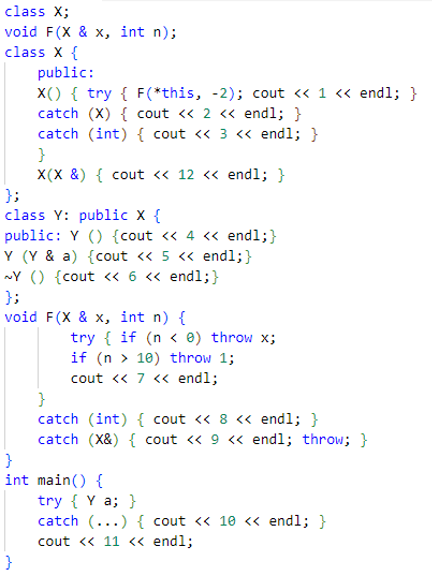
in.close();

out.close();

6. Аппарат исключений

Задание 1

Что будет выдано в стандартный канал вывода при работе следующей программы?

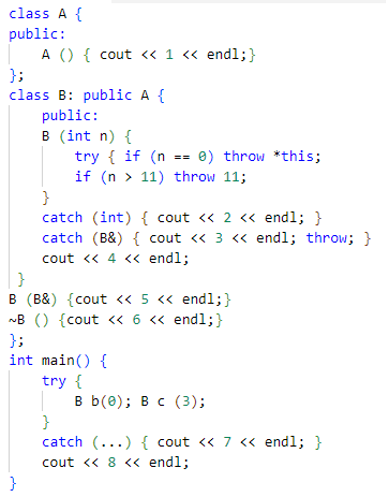


12 9 1 4 6 11

Задание 2

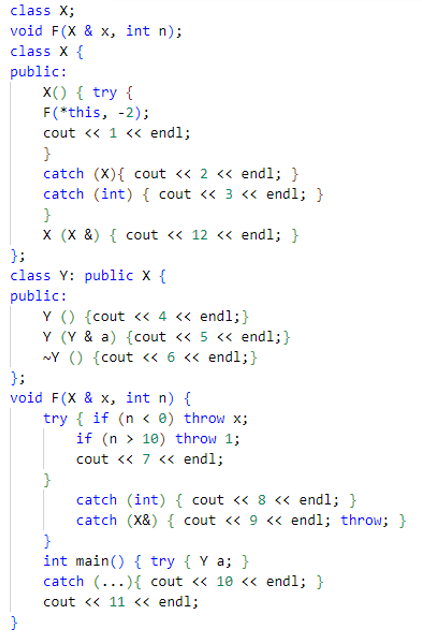
Что будет выдано в стандартный канал вывода при работе следующей программы?

1 1 5 3 7 6 8

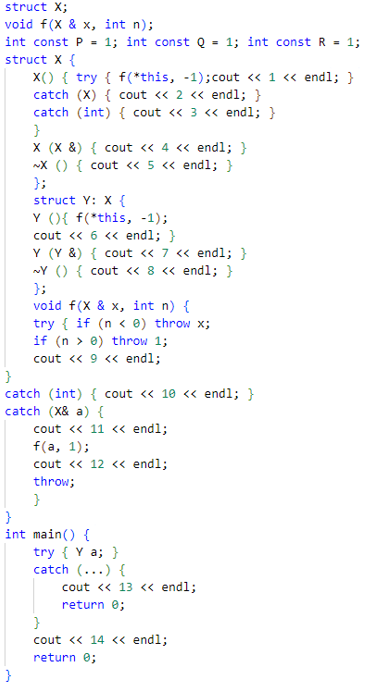


Задание 3

Что будет выдано в стандартный канал вывода при работе следующей программы?

12 9 1 4 6 11

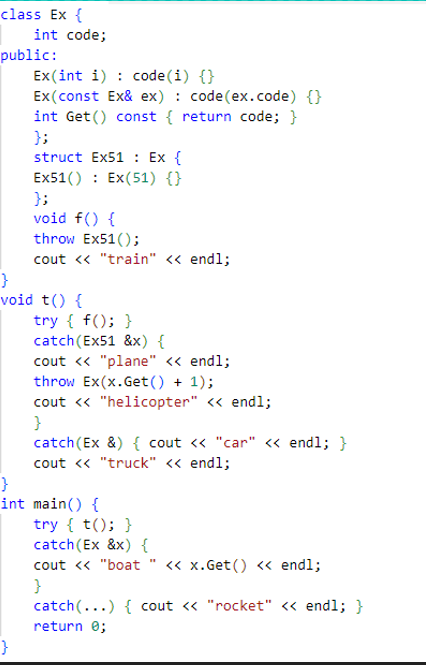
Задание 4



Что будет выдано в стандартный канал вывода при работе следующей программы?

4 11 10 12 2 5 4 11 10 12 5 13 5

Задание 5

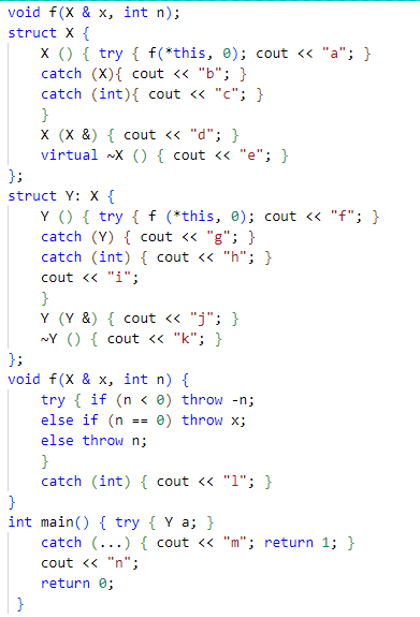


Что напечатает следующая программа?

plane boat 52

Задание 6

Что будет выдано в стандартный поток вывода при работе следующей программы?

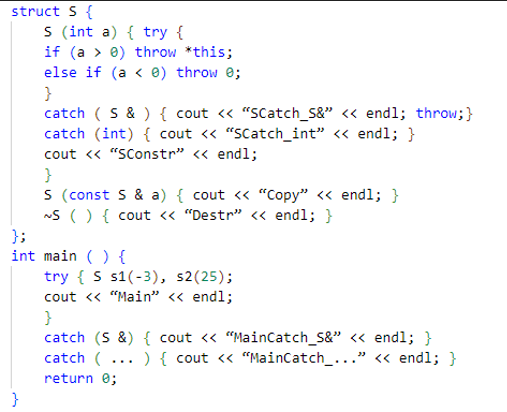


d b e d e m e

Задание 7

Что напечатает следующая программа?

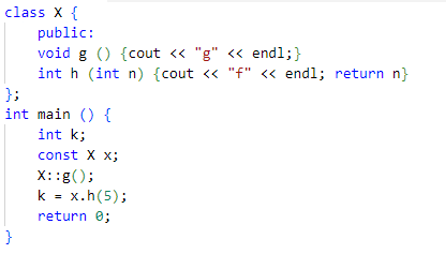
Scatch\_int SConstr Copy Scatch\_S& Destr MainCatch\_S& Destr



7. Константные и статические члены класса

Задание 1

Есть ли ошибки в тексте приведенной программы? Можно ли исправить описание класса, не вводя дополнительных членов, чтобы программа стала верной? Если да, то как?



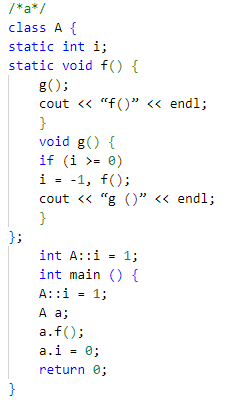
static const void g() { cout << "g "; } и убрать const при создании объекта класса

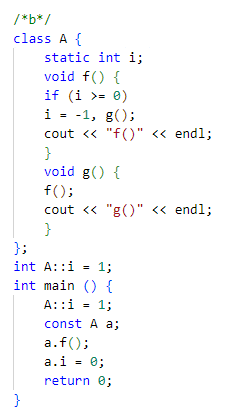
Задание 2

Есть ли синтаксические ошибки в тексте приведенной программы? Можно ли исправить описание класса, не вводя дополнительных членов и не убирая имеющиеся, чтобы программа стала верной? Если да, то как?

А)

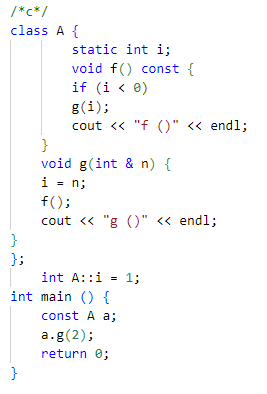
добавить public: классу и static g()



б)

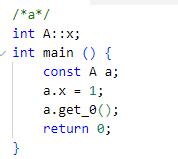
добавить public: классу и static обоим методам

С)

добавить public: классу, static void f(), static void g(const int& n)

Задание 3

Опишите класс А таким образом, чтобы были верными все конструкции следующего фрагмента программы:

А)

class A {

public:

static int x;

static int get\_0() {

if (x != 0) {

x = 0;

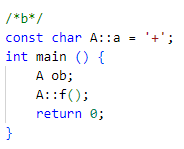
return x;

}

return x;

}

};

Б)

class A {

public:

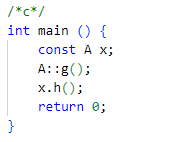
static const char a;

static void f() {

cout << "f() " << a;

}

};

С)

class A {

public:

static void g() {

cout << "g() ";

}

void h() const {

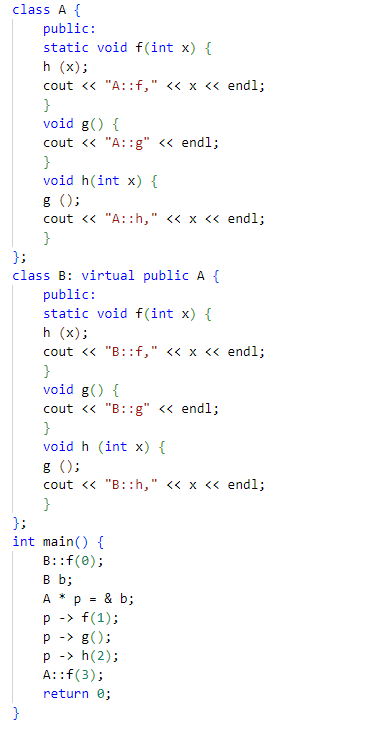
cout << "h() ";

}

};

Задание 4

Если есть ошибки в приведенной программе, то объясните, в чем они заключаются. Ошибочные операторы или ключевые слова вычеркните (допускается не более двух вычеркиваний). Что будет выдано в стандартный поток вывода при работе получившейся программы?



B::f(0);

B b;

A\* p = &b;

p->f(1);

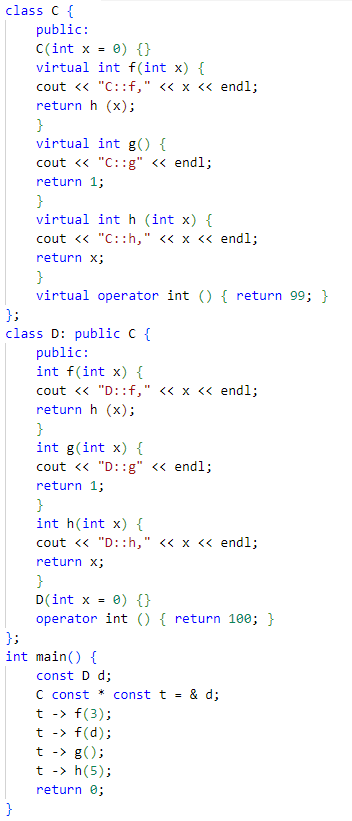
p->g();

p->h(2);

A::f(3);

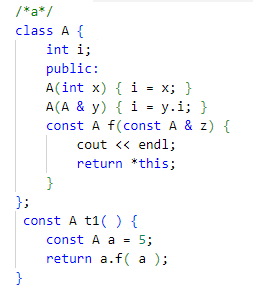
Задание 5

Есть ли ошибки в интерфейсах классов C и D программы? Если есть, то объясните, в чем они заключаются и внесите нужные исправления, оставив без изменения реализацию классов и функции main (). Что будет выдано в стандартный поток вывода при работе получившейся программы?



Задание 6

Добавить (если нужно) в класс А служебные слова const, так, чтобы заданный фрагмент программы был верным.



А)

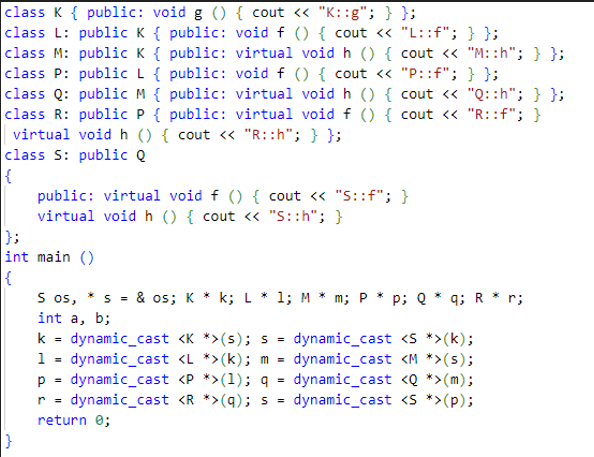
A(const A& y), A a = 5;

Б)

8. Динамическая идентификация и приведение типов

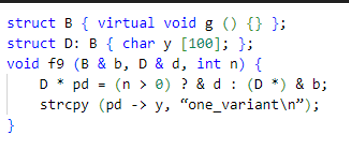
Задание 1

Укажите лишние и ошибочные операции динамического приведения типа, если таковые имеются в функции main (). Дайте необходимые пояснения своим исправлениям.



<S\*>(k), <L\*>(k) k не является полиморфным, тк не содержит виртуальных функций, <P\*>(l) и <S\*>(p) также

Задание 2

Добавить в функцию f9 () использование механизма приведения типов так, чтобы ее выполнение всегда завершалось нормально.

void f9(B & b, D & d, int n) {

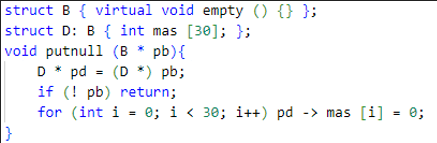
D\* pd = (n > 0) ? &d : (D\*)&b;

pd = dynamic\_cast<D\*>(&b);

strcpy\_s(pd->y, "one variant\n");

}\*

Задание 3

Добавить в функцию putnull () использование механизма приведения типов так, чтобы ее выполнение всегда завершалось нормально.

void putnull(B \* pb) {

D\* pd = (D\*)pb;

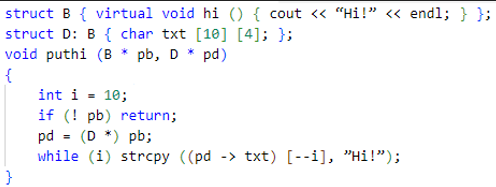
if (!pb) return;

pd = dynamic\_cast<D\*>(pb);

for (int i = 0; i < 30; i++)pd->mas[i] = 0;

}

Задание 4

Добавить в функцию puthi () использование механизма приведения типов, так, чтобы ее выполнение всегда завершалось нормально.

void puthi(B \* pb, D \* pd) {

int i = 10;

if (!pb)return;

pd = (D\*)pb;

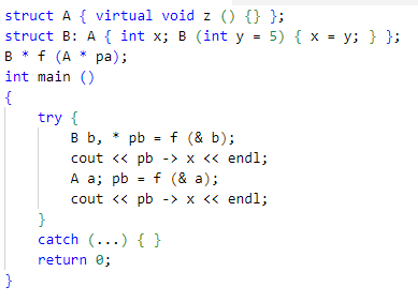
pb = dynamic\_cast<B\*>(pd);

while (i)strcpy((pd->txt)[--i], "Hi!");

}

Задание 5

Для приведённой ниже программы описать функцию f (), которая, получая в качестве параметра указатель типа A\*, возвращает его значение, наиболее безопасным образом преобразованное к типу B\*, а в случае невозможности преобразования корректно завершает работу программы.



B\* f(A \* pa) {

B\* b;

b = dynamic\_cast<B\*>(pa);

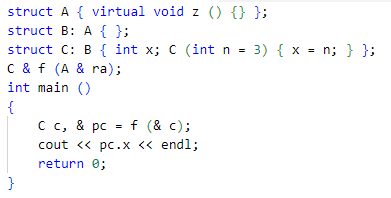
if (!b) { throw nullptr; }

return b;

}

Задание 6

Для приведённой ниже программы описать функцию f (), которая, получая в качестве параметра ссылку на объект базового класса A, возвращает ссылку на объект производного класса C, полученную наиболее безопасным образом, а в случае невозможности приведения типов корректно завершает программу.



C& f(A & ra) {

C& c = dynamic\_cast<C&>(ra);

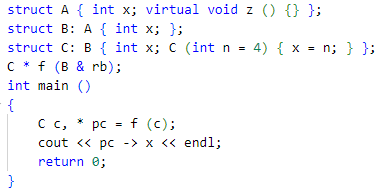
if (!&c) { throw nullptr; }

return c;

}

Задание 7

Для приведённой ниже программы описать функцию f (), которая, получая в качестве параметра ссылку на объект класса B, возвращает указатель на объект класса C, полученный наиболее безопасным образом, а в случае невозможности приведения типов корректно завершает программу.



C\* f(B & rb) {

C\* c;

c = dynamic\_cast <C\*>(&rb);

if (!c) { return 0; }

return c;

}