Санкт-Петербургский
государственный
университет

Вопросы и ответы к зачёту 14 декабря 2023 г. № 2

Санкт-Петербургский государственный университет

Вопросы

1. Что такое система контроля версий? Какие есть виды?
2. Какие есть команды git и что они делают?
3. Что такое git status и какие виды файлов есть у гита?
4. Как сделать коммит?
5. Что такое fork? Как создать локальную копию удаленного репозитория? (2 способа)
6. Как обновить локальный репозиторий на последнюю версию удалённого репозитория?
7. Что такое кодировка? Приведите примеры кодировок
8. Что такое таблица ASCII?
9. 8-битные кодировки, которые содержат русские буквы?
10. Приведите примеры несоответствия кодировок.

Санкт-Петербургский государственный университет

Вопросы

11. Что такое Unicode?
12. Что такое utf-8?
13. Как устроена модель памяти процесса (программы)?
14. В чём разница между глобальной и статической глобальной переменные?
15. Как хранятся строковые литералы?
16. Чем NULL отличается от nulltr?
17. Что такое анонимное пространство имён?
18. Что такое extern? Приведите примеры использования.
19. Что такое struct?
20. Что такое выравнивание в структурах?

Санкт-Петербургский государственный университет

Вопросы

21. Как определить размер структур?
22. Что такое указатель на функцию?
23. Для чего нужен указатель на функцию?
24. Что такое enum? И какой у него размер в памяти?
25. Что такое union? И какой у него размер в памяти?
26. Что такое класс? Чем класс отличается от объекта?
27. В чем заключаются основные принципы ООП?
28. В чём отличие структуры от класса?
29. Что такое public, protected, private?
30. Сколько занимает памяти объект пустой класс class A{}; ?

Санкт-Петербургский государственный университет

Вопросы

31. Что такое конструктор?
32. Что такое конструктор копирования?
33. Что такое this? И что такое инициализатор?
34. В чём особенность константных методов и константных объектов?
35. В чём особенность ключевого слова static с полями и методами класса?
36. Что такое деструктор и когда его надо реализовывать?
37. Что такое перегрузка функций/методов?

Санкт-Петербургский государственный университет

Что произойдёт в коде?

38

```cpp
#include <iostream>

void print_number(int x) {
    std::cout << x;
}

int main()
{
    print_number(5);
    print_number(5.5);
}
```

На консоль выведется: 55

А должно быть 55.5. Надо реализовать перегрузку функции print_number для вещественных чисел.

```cpp
#include <iostream>

void print_number(int x) {
    std::cout << x;
}

void print_number(double x) {
    std::cout << x;
}

int main()
{
    print_number(5);
    print_number(5.5);
}
```

Санкт-Петербургский государственный университет

Что произойдёт в коде?

39

```cpp
#include <iostream>

class Player {
    int id;
    int score;
};

int main()
{
    Player p1 = { 1, 2000 };
    std::cout << p1.id;
}
```

Ошибка компиляции в строке 10, а потом 11:

1. Нет конструктора для параметров (int, int)

2. Поле id private

Надо реализовать конструктор и метод get_id()

```cpp
#include <iostream>

class Player {
    int id;
    int score;

    public:
    Player(const int id, const int score) {
        this->id = id;
        this->score = score;
    }
    int get_id() const {
        return id;
    }
};

int main()
{
    Player p1 = { 1, 2000 };
    std::cout << p1.get_id();
}
```

Санкт-Петербургский государственный университет

Что произойдёт в коде?

40

```
1 #include <iostream>
2
3 class Player {
4     const int id;
5     int score;
6
7     public:
8     Player(const int id, const int score) {
9         this->id = id;
10        this->score = score;
11    }
12    int get_id() const {
13        return id;
14    }
15 };
16
17 int main()
18 {
19    Player p1 = { 1, 2000 };
20    std::cout << p1.get_id();
21 }
```

Ошибка компиляции в строке 19:

Поле id const и оно должно быть инициализировано до входа в тело конструктора.

Это надо сделать с помощью "инициализатора"

(решение строка 8 и 9)

```
1 #include <iostream>
2
3 class Player {
4     const int id;
5     int score;
6
7     public:
8     Player(const int id, const int score) : id(id) {
9         this->score = score;
10    }
11    int get_id() const {
12        return id;
13    }
14 };
15
16 int main()
17 {
18    Player p1 = { 1, 2000 };
19    std::cout << p1.get_id();
20 }
```

Санкт-Петербургский государственный университет

Что произойдёт в коде?

41

```cpp
#include <iostream>

class Player {
    int id;
    int score;

    public:
    Player(const int id, const int score) {
        this->id = id;
        this->score = score;
    }
    int get_id() const {
        score++;
        return id;
    }
};

int main()
{
    Player p1 = { 1, 2000 };
    std::cout << p1.get_id();
}
```

Ошибка компиляции в строке 21 (и 13):

Метод get_id объявлен как константный метод, и поэтому в нём не могут меняться поля класса.

score++; из 13-ой строки надо удалить или объявить score как mutable: 5 ..... mutable int score;

42

```cpp
#include <iostream>

class Player {
    public:
    static const int PLAYERS_COUNT = 59;

    private:
    int id;
    int score;

    public:
    Player(const int id, const int score) {
        this->id = id;
        this->score = score;
    }
    int get_id() const {
        return id;
    }
};

int main()
{
    Player p1 = { 1, 2000 };
    std::cout << p1.PLAYERS_COUNT;
    std::cout << Player::PLAYERS_COUNT;
}
```

На консоль выведется: 5959

Статические поля можно вызывать и через имя объекта, и через имя класса. Корректно это делать через имя класса.

Санкт-Петербургский государственный университет

Что произойдёт в коде?

43

```cpp
#include <iostream>

class Player {
    int id;
    int score;

    public:
    Player(const int id, const int score) {
        this->id = id;
        this->score = score;
    }
    int get_id() const {
        return id;
    }
};

int main()
{
    Player p1 = { 1 };
}
```

Ошибка компиляции в строке 19:

Нет такого конструктора Player(int)

Надо перегрузить конструктор и создать ещё один с аргументом (int)

```cpp
#include <iostream>

class Player {
    int id;
    int score;

    public:
    Player(const int id) {
        this->id = id;
        this->score = 0;
    }
    Player(const int id, const int score) {
        this->id = id;
        this->score = score;
    }
    int get_id() const {
        return id;
    }
};

int main()
{
    Player p1 = { 1 };
}
```

Санкт-Петербургский государственный университет

Что произойдёт в коде?

44

```cpp
#include <iostream>

class Player {
    int id;
    int score;

    public:
    Player(const int id) {
        this->id = id;
        this->score = 0;
    }
    Player(const int id, const int score) {
        this->id = id;
        this->score = score;
    }
    int get_id() const {
        return id;
    }
};

int main()
{
    Player p1 = { 1 };
    Player p2;
    p2 = p1;
    std::cout << p2.get_id();
}
```

Ошибка компиляции в строке 24:

Отсутствует конструктор по умолчанию.

Компилятор создаёт сам конструктор по умолчанию, если в классе нет ни одного конструктора. Надо создать конструктор по умолчанию.

```cpp
#include <iostream>

class Player {
    int id;
    int score;

    public:
    Player() {
        this->id = 0;
        this->score = 0;
    }
    Player(const int id) {
        this->id = id;
        this->score = 0;
    }
    Player(const int id, const int score) {
        this->id = id;
        this->score = score;
    }
    int get_id() const {
        return id;
    }
};

int main()
{
    Player p1 = { 1 };
    Player p2;
    p2 = p1;
    std::cout << p2.get_id();
}
```

Санкт-Петербургский государственный университет

Ответы

1 Что такое система контроля версий? Какие есть виды?

Система контроля версий — это ПО, позволяющее хранить много версий одних и тех же файлов; определять кто и когда делал изменения и многое другое. СКВ используются при разработке ПО несколькими разработчиками.

Есть централизованные и распределённые (децентрализованные) СКВ.

ЦСКВ используют единственный сервер, содержащий все версии файлов. Клиенты в этом случае берут из хранилища те файлы, которые хотят изменить. ЦСКВ легко управлять, но их большим минусом является единая точка отказа. Если сервер выйдет из строя на час, то в течение этого часа никто не сможет пользоваться СКВ. Пример ЦСКВ — это Subversion.

ДСКВ — это СКВ, когда каждый клиент и сервер, используемый в проекте, содержат полную копию репозитория проекта. Все копии являются равноправными и могут синхронизироваться между собой. Если не станет какой-либо копии репозитория, то это не приведет к потере кодовой базы, т.к. она будет восстановлена с компьютера любого разработчика. Пример ДСКВ — это git.

2 Какие есть команды git и что они делают?

git init инициализирует git репозиторий в текущем каталоге

git clone делает копию удалённого репозитория на локальной машине

git status показывает статус файлов рабочей копии

git add добавляет изменённые файлы в индекс

git commit создаёт снимок изменений в истории

git log показывает список всех коммитов

git push отправляет новые коммиты в удалённый репозиторий

git pull выкачивает из удалённого репозитория все изменения, которых нет в рабочей копии

git remote add добавляет знание в гит об удалённом репозитории

Санкт-Петербургский государственный университет

Ответы

3 Что такое git status и какие виды файлов есть у гита?

git status показывает статус файлов рабочей копии

Гит изменённые файлы делит на две большие группы: Unstaged и Untracked.

Unstaged файлы — это файлы, которые существовали в предыдущих коммитах, а сейчас были как-то изменены.

Untracked файлы — это новые созданные файлы, никакой информации о которых в гите нет.

4 Как сделать коммит?

1.) Выполнить команду **git status**. Она покажет изменённые файлы.

Если у вас была создана новая папка с новыми файлами, то, чтобы отобразился список её файлов, но выполнить **git status -uall**.

2.) Добавить в индекс нужные файлы командой **git add .** или имена файлов через пробел.

После этой команды надо выполнить git status и увидеть, что все нужные для коммита файлы стали зелёного цвета.

3.) Создать коммит из проиндексированных файлов командой

git commit -m "[ibusko] Create function func"

Санкт-Петербургский государственный университет

Ответы

5 Что такое fork? Как создать локальную копию удаленного репозитория? (2 способа)

fork — это полная копия проекта, которая находится на удалённом сервере и принадлежит конкретному разработчику.

2 способа создания локальной копии удаленного репозитория:

1.) **git clone <url репозитория>**

2.) Этот способ мы использовали для Visual Studio.

2.1. Создать пустой каталог (в котором мы создали проект Visual Studio и потом все файлы добавили руками).

2.2. **git init**

2.3. **git remote add <имя репозитория> <url репозитория>**

Имя для своего fork-а надо давать origin

2.4. Выкачать файлы удалённого репозитория командой **git pull <имя репозитория> <имя ветки>**.

<имя репозитория> — из какого репозитория

<имя ветки> — из какой ветки этого репозитория

Санкт-Петербургский государственный университет

Ответы

6 Как обновить локальный репозиторий на последнюю версию удалённого репозитория?

git pull <имя репозитория> <имя ветки>

<имя репозитория> — из какого репозитория

<имя ветки> — из какой ветки этого репозитория

Данный удалённый репозиторий должен находится в списке репозиториев, о которых знает git: **git remote -v**

7 Что такое кодировка? Приведите примеры кодировок

Кодировка — это набор числовых значений, которые ставятся в соответствие группе алфавитно-цифровых символов, знаков пунктуации и специальных символов.

ASCII, OEM (DOS) 866, ANSI 1251, UTF-8

8 Что такое таблица ASCII?

ASCII — American Standard Code for Information Interchange — Американская стандартная кодировка для обмена информацией. Создана в США в 1963 г.

ASCII — это таблица кодировки символов, в которой каждой букве, числу или знаку соответствует определенное число. В стандартной таблице ASCII 128 символов, пронумерованных от 0 до 127 (7 бит). В них входят латинские буквы, цифры, знаки препинания и управляющие символы.

Существуют национальные расширения ASCII, которые кодируют буквы и символы, принятые в других алфавитах. Т. к., char — это 8 бит, то в него можно поместить ещё 128 символов. Т.о., char может кодировать и русский язык.

Санкт-Петербургский государственный университет

Ответы

9 8-битные кодировки, которые содержат русские буквы?

Все кодировки основаны на ASCII. Т.е., первые 128 символов у всех кодировок одинаковые и соответствуют символам ASCII. А дальше идут символы конкретного языка. При этом, кодировки могут быть 1, 2 и 4 байтными.

8-битными кодировками, которые содержат русские буквы являются: CP 866 для ОС MS-DOS и dos программ в windows, например, cmd; CP 1251 (ANSI 1251) для Windows, КОИ-8 (KOI8-R или CP 20866) для Unix систем, Macintosh для Mac ОС.

Каждая кодировка имеет тип и номер. Тип относится к ОС, а номер — к локали. Например, ANSI — это ОС Windows, а номер 1251 означает русский алфавит (русскую локаль). Если номер будет 1252, то это будет латинский алфавит (западно-европейские языки).

10 Приведите примеры несоответствия кодировок.

1. Если русский текст, подготовленный в одной русской кодировке, попытаться прочитать другой русской кодировкой, то получатся «козяблики».

2. Если сохранить русский текст в кодировке, которая не содержит русских символов, то вместо русского текста будут вставлены ????? или квадратики.

3. Если строковые литералы в скомпилированном коде находятся в одной кодировке, а консоль, где они будут выводится, работает в другой кодировке, то будут выведены «козяблики».

Санкт-Петербургский государственный университет

Ответы

11 Что такое Unicode?

Unicode (Universal Coded Character Set) — это стандарт кодировки, который был предложен в 1991 году. Основная идея стандарта в увеличении количества кодируемых символов за счёт увеличения кодового пространства. Вместо одного байта в стандарте Юникод символы могут занимать 2 и более байт. При этом, символы с кодами от 0 до 127 соответствуют символам набора ASCII. Для практического использования было создано несколько семейств кодировок: UTF-8, UTF-16, UTF-32.

12 Что такое utf-8?

UTF8 (Unicode Transform Protocol 8-bit — «формат преобразования Юникода, 8-бит») — протокол кодирования кодовых пунктов в Unicode. Существуют разные типы UTF. Они различаются количеством байтов, используемых для кодировки одного пункта. В UTF-8 используется (минимум) один байт на пункт, в UTF-16 — (минимум) два байта, в UTF-32 — (всегда) четыре байта.
Но если у нас есть три разные кодировки, то как узнать, какая из них применяется в конкретном файле? Для этого используют маркер последовательности байтов (Byte Order Mark, BOM), который ещё называют сигнатурой кодировки (UTF8 с сигнатурой). BOM — это двухбайтный маркер в начале файла, который говорит о том, какая именно кодировка применена.

UTF-8 обеспечивает наибольшую компактность и обратную совместимость с 7-битной системой ASCII. Текст, состоящий только из символов с номерами меньше 128, при записи в UTF-8 превращается в обычный текст ASCII и может быть отображён любой программой, работающей с ASCII. Остальные символы Юникода в UTF-8 изображаются последовательностями длиной от 2 до 4 байт.

Санкт-Петербургский государственный университет

Ответы

13 Как устроена модель памяти процесса (программы)?

Модель оперативной памяти программы имеет следующие части:

1. Стек
2. Куча
3. BSS
4. Data
5. Text (Code)

1. Стек хранит переменные, размер которых в памяти можно вычислить во время компиляции.
2. Куча хранит переменные, размер которых в памяти можно вычислить только во время выполнения. Память выделяется оператором new и очищается оператором delete.
3. BSS хранит статические и глобальные переменные, которые не инициализированы.
4. Data хранит статические и глобальные переменные, которые инициализированы.
5. Text (Code) хранит скомпилированный текст кода.

Санкт-Петербургский государственный университет

Ответы

14 В чём разница между глобальной и статической глобальной переменные?

Глобальная переменная одна на всю программу.

Глобальная статическая — одна на текущий файл компиляции. Например, если объявить статическую переменную в заголовочном файле и подключить этот файл в двух разных cpp-файлах, каждый из них будет работать со своей переменной. Это важно во избежание неконтролируемого изменения глобальных переменных из разных файлов программы.

Глобальные переменные стараются не использовать.

15 Как хранятся строковые литералы?

Строковые литералы считаются глобальными статическими константными переменными, инициализированными при объявлении.

Если в коде будет написано два одинаковых строковых литерала "ABC" и "ABC", то в памяти будет сохранена только одна переменная. И она будет сохранена в части памяти Text.

16 Чем NULL отличается от nulltr?

NULL — это макрос, эквивалент записи 0 (ноль) и инициализирован в большом количестве библиотек.

nullptr — ключевое слово, введенное в C++11 для описания константы нулевого указателя.

Санкт-Петербургский государственный университет

Ответы

17 Что такое анонимное пространство имён?

namespace {…}

Анонимное пространство имён — это возможность исключить ошибки повторного определения глобальных переменных/функций с одним и тем же именем во время линковки, когда они используются только в своих единицах трансляции.

Если в двух cpp файлах будет две глобальных переменных/функции с одним и тем же именем, то ошибка повторного определения возникнет на этапе линковки, когда линкер будет объединять все имена в программе.

Компилятор анонимному пространству имён присваивает своё имя, отличное от всех, которое будет использовано только в текущем cpp файле.

18 Что такое extern? Приведите примеры использования.

extern — это ключевое слово, которое означает объявление переменной, определение которой находится в другом файле. Объявлений может быть много, а определений только одно.

Примеры применения:

1. Объявление любой функции по умолчанию является extern.

2. Позволяет объявлять константные переменные в одном файле (а иначе константные переменные должны быть инициализированы сразу в момент объявления), а определять в другом. Это может улучшить читабельность кода.

3. Объявление и определение переменных (как и функций) в разных файлах уменьшает количество кода при include, уменьшает время компиляции каждого cpp (т.к. не нужно в каждом cpp выделять память под переменные, определять их)

Санкт-Петербургский государственный университет

Ответы

19 Что такое struct?

Структура — это пользовательский тип данных, представляющий собой определённую сущность.

Когда нам надо объединить данные разных типов для облегчения хранения и использования, то применяют структуры.

Структура очень похожа на класс. И даже технически с ней можно делать почти всё, что делают с классом (Так в C++, но не в C). Но класс, с точки зрения логики, описывает совокупность однотипных объектов, которые имеют жизненный цикл и поведение. А структуру используют только для облегчения хранения разнородных данных, относящихся к какой-то сущности.

Можно сказать так (с точки зрения логики применения): «Структура — это класс, у которого все поля публичные, даже неизменяемые, и которая не имеет методов.»

20 Что такое выравнивание в структурах?

Выравнивание в структурах — это способ обеспечить эффективный доступ в памяти к данным в структурах (и классах).

Компилятор выравнивает данные в памяти по размеру переменной максимального типа. (Размер переменной максимального типа в 64-битной машине равен 8 байт. И такое может быть даже в случае с long double)

Размер структуры выравнивается до размера, кратного размеру его максимального в памяти элемента.

Компилятор читает поля структуры последовательно как они записаны в коде. И там, где требуется их выровнять при выделении памяти, добавляет недостающее количество пустых ячеек.

Процесс оптимизации последовательности полей: достаточно расположить поля в порядке убывания размера их типа.

Санкт-Петербургский государственный университет

Ответы

21 Как определить размер структур?

В коде это можно сделать с помощью оператора sizeof(имя структуры).

Размер структуры определяется согласно правилам выравнивания памяти полей структуры. Про особенности этих правил можно сказать следующее:

1. Размер структуры выравнивается до размера, кратного размеру его максимального в памяти элемента.

2. Чтобы выровнять поля достаточно расположить их в порядке убывания их размера.

3. Если поля не выровнены порядком расположения, то компилятор читает поля структуры последовательно, как они записаны в коде. И там, где требуется их выровнять при выделении памяти, добавляет недостающее количество пустых ячеек.

```
1 struct size_is_8_bytes {
2   int i;
3   char c1;
4   char c2;
5 }
```

```
1 struct size_is_12_bytes {
2   char c1;
3   int i;
4   char c2;
5 }
```

```
1 struct size_is_16_bytes {
2   char* c1ptr;
3   int i;
4   char c2;
5 }
```

```
1 struct size_is_24_bytes {
2   int i;
3   char* c1ptr;
4   char c2;
5 }
```

Санкт-Петербургский государственный университет

Ответы

22 Что такое указатель на функцию?

В C++ есть возможность работать с памятью через указатели, и поэтому мы можем хранить указатели на функцию.

Указатель на функцию хранит адрес функции. А адрес функции — это адрес первого байта в памяти, по которому располагается выполняемый код. Указателем на функцию является её имя.

Объявление переменной, содержащей адрес функции имеет вид:

[возвращаемый тип] (* имя переменной)(список аргументов)

23 Для чего нужен указатель на функцию?

Указатель на функцию позволяет:

1. Передавать функции в качестве параметров другим функциям.

2. Выбирать функцию из однотипных (одинаковая сигнатура) во время выполнения, а не компиляции. Это позволяет избежать в коде больших ветвлений.

Т.е. указатель на функцию даёт программисту большую гибкость в написании решения программ.

24 Что такое enum? И какой у него размер в памяти?

Enum (перечисление) — это один из способов определения пользовательского типа. Он служит для создания перечисления вариантов чего-то. Каждое имя имеет своё числовое значение. Если именам не присвоены значения, то по умолчанию они начинаются с 0 и идут по увеличению. Часто enum используют для задания типов объектов или видов действий. Enum могут использоваться в конструкциях switch/case.

Размер в памяти enum равен размеру int.

Санкт-Петербургский государственный университет

Ответы

25 Что такое union? И какой у него размер в памяти?

Union — это определяемый пользователем тип, в котором все члены совместно используют одно расположение памяти. Т.е., union одновременно может содержать не более одного объекта из списка своих элементов. Т.о., размер памяти union равен размеру памяти самого большого его элемента.

26 Что такое класс? Чем класс отличается от объекта?

Класс — это пользовательский тип данных, в котором задаются свойства и поведение какого-либо объекта/набора однотипных объектов. Существенным отличием класса от структуры является то, что детали его реализации скрыты за интерфейсом. А интерфейсом класса являются его методы.

Объект — это экземпляр класса.

27 В чем заключаются основные принципы ООП?

Идея классов является основой ООП. Из этого следует, что основными принципами ООП являются: инкапсуляция, наследование, полиморфизм.

Инкапсуляция — это объединение данных с функциями их обработки в сочетании со скрытием ненужной для использования этих данных информации.

Наследование — это возможность создания иерархии классов, когда потомки наследуют все свойства своих предков, могут их изменять и добавлять свои.

Полиморфизм — это возможность использовать в различных классах иерархии одно имя для обозначения сходных по смыслу действий и гибко выбирать требуемое действие во время выполнения программы.

Санкт-Петербургский государственный университет

Ответы

28 В чём отличие структуры от класса?

1. По умолчанию поля и методы классы private, а у структуры public.

2. Если мы наследуем один класс от другого и не указываем модификатор доступа, то поля и методы класса наследуются как private. А у структуры как public.

Это технические моменты, а есть исторические и, соответственно, логические:

Структура — это элемент языка C, в котором не было: модификаторов доступа, наследования, полиморфизма и функций-методов. Но потом появился C++, в котором появился класс и ООП. Т.к. C++ включает в себя C, то в нём оставили структуры, наделив их свойствами классов, с той разницей, что у классов по умолчанию модификатор доступа private, а у структуры — public.

Поэтому когда надо облегчить хранение разнородных данных, относящихся к какой-то сущности, то используем структуру. А если надо создать абстракцию для объектов с поведением, то использует класс.

Или можно применить еще один признак отличия: «Структура — это класс, у которого все поля публичные, даже неизменяемые, и которая не имеет методов.»

29 Что такое public, protected, private?

public, protected, private — это модификаторы доступа. Используются для: полей, методов, для самих классов, при наследовании.

Public — доступ открыт всем, кто видит данный класс.

Protected — доступ открыт классам, производным от данного.

Private — доступ открыт только самому классу и дружестсвенным классам и функциям.

Надо помнить, что модификаторы доступа всего лишь дают указания компилятору контролировать обращение к полям и функциям класса. Т.е., это не означает, что нельзя получить доступ к памяти, по которой хранится private поле.

Санкт-Петербургский государственный университет

Ответы

30 Сколько занимает памяти объект пустой класс class A{}; ?

Размер любого объекта программы на C++ по стандарту не может быть меньше 1 байта (точнее одного элемента памяти на данной платформе).

31 Что такое конструктор?

Конструктор предназначен для инициализации объекта и вызывается автоматически при его создании. Свойства конструктора:

1. Не возвращает значения.
2. Класс может иметь несколько конструкторов с разными параметрами.
3. Если не указан ни один конструктор, то компилятор создаёт его автоматически.

32 Что такое конструктор копирования?

КК — это специальный вид конструктора, получающий в качестве единственного параметра указатель на объект этого же класса: T::T(const T&) {тело конструктора}, где T — имя класса.

Этот конструктор вызывается тогда, когда новый объект создаётся путём копирования существующего:

1. При описании нового объекта с инициализацией другим.
2. При передаче объекта в функцию по значению.
3. При возврате объекта из функции.

Если не указан ни один конструктор копирования, то компилятор создаёт его автоматически. Если класс содержит указатели или ссылки, то конструктор копирования надо создавать. Т.к. это будет неправильно, что поля оригинала и копии будут указывать на одну и ту же область памяти.

Санкт-Петербургский государственный университет

Ответы

33 Что такое this? И что такое инициализатор?

this — это указатель на поля и методы текущего объекта.

Инициализатор — это значение, которое присваивается переменной при её объявлении.

Если надо проинициализировать поля-константы в классе, то надо это делать с помощью списка инициализаторов (специальной конструкции), расположенных после двоеточия между заголовком и телом конструктора:

MyClass(const int x) : x(x) {тело конструктора}

34 В чём особенность константных методов и константных объектов?

Константный метод — это метод, который гарантирует, что не будет изменять объект или вызывать неконстантные методы класса.

Объявление константного метода: тип имя_метода(список аргументов) const {…}

Если мы хотим возвратить из константной функции указатель или ссылку, то они должны указывать на константу, а ссылка должна быть константной:

const GoMove& get_move() const {}

const GoMove* get_move() const {}

Константность объекта запрещает вызывать у этого объекта неконстантные методы. В противном случае будет ошибка компиляции.

Иногда бывает необходимо, чтобы какие-то данные константного объекта все-таки можно было менять. В этом случае для поля, которое необходимо менять, можно использовать ключевое слово mutable при объявлении.

Санкт-Петербургский государственный университет

Ответы

35 В чём особенность ключевого слова static с полями и методами класса?

Статические поля класса относятся не к объекту, а к классу. Они применяются для хранения данных, общих для всех объектов.

Статические методы — это методы, которые выполняют действия не зависимо от данных объекта. Поэтому они могут иметь доступ только к статическим полям и другим статическим методам класса. Им не передаётся this.

Статические поля и методы класса доступны и через имя объекта, и через имя класса. Но корректно вызывать их через имя класса.

36 Что такое деструктор и когда его надо реализовывать?

Деструктор — это особый вид метода, применяющийся для освобождения памяти, занимаемой объектом.

Деструктор вызывается автоматически, когда объект выходит из области видимости. Если объект задан через указатель (new), то деструктор вызывается неявно при использовании delete.

Если деструктор явным образом не определён, то компилятор автоматически создаёт его. Описывать деструктор в классе надо, когда объект содержит указатели на память, выделяемую динамически.

```
Vector::~Vector() {…}
```

37 Что такое перегрузка функций/методов?

Перегрузка функций — это использование нескольких функций с одним и тем же именем, но с различными типами параметров. Это позволяет писать более понятный код, когда один и тот же (или почти) алгоритм надо реализовать для разного типа параметров.

В классах перегрузка используется для:
1. Конструкторов
2. Операций (унарных и бинарных)
3. Операции присваивания
4. Методов