1

1. Принцип абстрагирования:

а. В чем заключается принцип абстрагирования?

Абстрагирование - это способ выделить набор значимых характеристик объекта, исключая из рассмотрения незначимые. Соответственно, абстракция - это набор всех таких характеристик.

б. Что такое барьер абстракции, уровни абстракции?

Барьер абстракции отделяет существенные стороны от несущественных.

в. В чем заключается принцип наименьшего удивления?

Принцип наименьшего удивления - абстракция не должна содержать ничего, что лежит вне сферы ее применения. Другими словами: ожидаемое предназначение или поведение должно совпадать с реальностью.

г. Что такое контрактная модель программирования?

Это модель, предполагающая, что проектировщик должен определить формальные, точные и верифицируемые спецификации интерфейсов для компонентов системы.

д. Что такое сигнатура операции?

Это часть общего объявления функции, которая позволяет средствам трансляции выполнять идентификацию этой самой функции среди других.

е. Что такое инвариант?

Это либо (чаще) условие, которое остается истинным после вызова любых методов объекта в любой последовательности, либо (реже) выражение которое сохраняет свое значение после вызова любых методов.

2. Исключения:

а. Как определить пробный блок и обработчики исключений?

Инструкция, генерирующая исключение, должна исполняться внутри блока try. Вызванные из блока try функции также могут генерировать исключения. Всякое исключение должно быть перехвачено инструкцией catch, которая непосредственно следует за инструкцией try, сгенерировавшей исключение.

б. Как осуществить возбуждение исключительной ситуации?

+

в. Как происходит обработка исключений?

Мы пытаемся (try) выполнить блок кода, и если при этом возникает ошибка, система возбуждает (throw) исключение, которое в зависимости от его типа мы можем перехватить (catch) или передать умалчиваемому (finally) обработчику.

3. Инкапсуляция:

а. В чем заключается принцип инкапсуляции?

Это свойство, позволяющее объединить данные и методы в классе, скрыв детали реализации от пользователя. По сути, всё то, что не входит в интерфейс, инкапсулируется в классе.

б. В чем смысл разделения класса на интерфейс и реализацию?

Отделение интерфейса от реализации позволяет в полной мере использовать полиморфизм.

в. Что такое класс? Как он определяется? Как определяются операции для класса?

Класс - это шаблон кода, по которому создаётся какой-то объект.
Класс определяется как список своих членов, а именно полей (свойств) и методов/функций/процедур. В зависимости от языка программирования к этому списку могут добавиться константы, атрибуты и внешние определения.

г. Как определяется открытая и закрытая части тела класса?
Открытые члены (или «члены public») - это члены структуры или класса, к которым можно получить доступ извне этой же структуры или класса.
Закрытые члены (или «члены private») - это члены класса, доступ к которым имеют только другие члены этого же класса.

д. Что такое друг класса?
Дружественная функция - это функция, которая имеет доступ к закрытым членам класса, как если бы она сама была членом этого класса. Один класс может быть дружественным другому классу. Это откроет всем членам первого класса доступ к закрытым членам второго класса.

4. Модульность:
а. Что такое модуль?
Функционально законченный фрагмент программы.

б. В чем заключается принцип модульности?
Каждый класс должен составлять отдельный модуль. Члены класса, к которым не планируется обращение извне, должны быть инкапсулированы.

в. Модульность на языке C++?
На уровне языка, C++ представляет интерфейс в виде объявлений. Объявление указывает все необходимые части интерфейса, которые необходимы например для функции: возвращаемое значение, аргументы, имя функции.
Зачастую объявление функций осуществляется в заголовочных файлах с расширением .h, а реализация осуществляется в файлах исходных кодов .cpp.

5. Иерархичность:
В чем заключается принцип иерархичности?
Иерархия - это ранжированная или упорядоченная система абстракций. Принцип иерархичности предполагает использование иерархий при разработке программных систем.

6. Типизация:
а. В чем заключается принцип типизации?
Типизация - это ограничение, накладываемое на свойства объектов и препятствующее взаимозаменяемости абстракций различных типов (или сильно сужающее возможность такой замены). В языках с жесткой типизацией для каждого программного объекта (переменной, подпрограммы, параметра и т. д.) объявляется тип, который определяет множество операций над соответствующим программным объектом.

б. Что такое полиморфизм?
Это свойство использовать объекты с одинаковым интерфейсом без информации о типе и внутренней структуре объекта. Множество реализаций одного интерфейса.

в. Что такое принудительное приведение?
Принудительное приведение задаётся программистом в тексте программы с помощью конструкции языка или функции, принимающей значение одного типа и возвращающей значение другого типа.

г. Что такое перегрузка?
Перегрузки - методы с таким же названием, но принимающие другие аргументы.

д. Какие операторы приведения вы знаете?

const_cast

static_cast

dynamic_cast

reinterpret_cast

7. В чем заключается принцип параллелизма?
Параллелизм - свойство нескольких абстракций одновременно находиться в активном состоянии, т.е. выполнять некоторые операции.

8. В чем заключается принцип сохраняемости?
Сохраняемость - способность объекта существовать во времени, переживая породивший его процесс, и (или) в пространстве, перемещаясь из своего первоначального адресного пространства.

2

1. Что такое состояние объекта, чем оно определяется?
Состояние - это информация, которую хранит объект.
Состояние объекта определяется значениями его свойств (атрибутов) и связями с другими объектами, оно может меняться со временем.

2. Что такое поведение объекта, посредством чего оно реализуется?
Поведение определяет действия объекта и его реакцию на запросы от других объектов.
Поведение представляется с помощью набора сообщений, воспринимаемых объектом (операций, которые может выполнять объект).

3. Какие виды типичных операций над объектами вы можете назвать?
Модификатор (Операция, изменяющая свойства объекта)
Селектор (Операция, позволяющая читать значение свойств объекта)
Итератор (Операция, позволяющая получать последовательный доступ к свойствам объекта)
Конструктор (Операция, создающая объект)
Деструктор (Разрушающая объект)

4. Каким образом могут создаваться объекты?
При помощи конструктора.

5. Для чего используется ключевое слово explicit?
Конструктор без explicit позволяет осуществлять неявное преобразование типа аргумента в тип класса, которому принадлежит конструктор.
Если конструктор помечен ключевым словом explicit, то такое преобразование типа применяется только при наличии явной операции приведения типа.

6. Что такое инициализатор конструктора, в каких случаях его использование обязательно?
Инициализатор позволяет задавать класс и подлежащий вызову конструктор.

7. Что такое свободная подпрограмма?
Это простая, не связанная с объектом функция.

8. Что такое идентичность объекта?
Это то, что отличает один объект класса от другого объекта класса.

9. Что такое структурная зависимость, какие проблемы она порождает?
Это зависимость, использующая ссылки между объектами.
В случае использования структурной зависимости важно придерживаться какой-то договоренности относительно создания и уничтожения объектов, ссылки на которые могут содержаться в разных местах.

10. Что такое конструктор копирования, какие проблемы могут возникнуть при использовании конструктора копирования, предоставляемого по умолчанию?
Это особый тип конструктора, который используется для создания нового объекта через копирование существующего объекта. В случае с конструктором по умолчанию, если мы не предоставим конструктор копирования для своих классов самостоятельно, то C++ создаст public-конструктор копирования автоматически.

11. Какие проблемы могут возникнуть при использовании оператора присваивания, предоставляемого по умолчанию?
Самоприсваивание. Когда используется динамическое выделение памяти, самоприсваивание может быть даже опасным.

12. Что такое связь?
Отношение между двумя объектами, описанное при помощи агрегации или композиции.

13. Что такое агрегация (отношение между объектами)?
Агрегация - отношение «часть-целое» между двумя равноправными объектами, когда один объект (контейнер) имеет ссылку на другой объект. Оба объекта могут существовать независимо: если контейнер будет уничтожен, то его содержимое - нет.
Пример: профессора - факультеты, профессора остаются жить после разрушения факультета.

3

1. Ассоциация:
а. Что такое ассоциация?
Это отношение между классами объектов, которое позволяет одному экземпляру объекта вызвать другой, чтобы выполнить действие от его имени (объекты ссылаются друг на друга).
Агрегация и композиция являются частными случаями ассоциации.

б. Что такое кратность ассоциации? На какие типы делятся ассоциации в связи с понятием кратности?
Двойные ассоциации - соединяют два классовых блока. Ассоциации более высокой степени имеют более двух концов.

в. Что такое рефлексивная ассоциация?
Рефлексивная ассоциация - частный случай бинарной, связывает класс с самим собой.

2. Агрегация:
а. Что такое агрегация (отношение между классами)?
Агрегация - отношение «часть-целое» между двумя равноправными объектами, когда один объект (контейнер) имеет ссылку на другой объект. Оба объекта могут существовать независимо: если контейнер будет уничтожен, то его содержимое - нет.
Пример: профессора - факультеты, профессора остаются жить после разрушения факультета.

б. Что такое композиция?
Композиция (агрегирование по значению) - более строгий вариант агрегирования, когда включаемый объект может существовать только как часть контейнера. Если контейнер будет уничтожен, то и включенный объект тоже будет уничтожен.
Пример: университет - факультеты, факультеты без университета погибают.

3. Что такое зависимость?
Зависимости выражают связи или отношения между классами указанных объектов.

4. Наследственная иерархия:

а. Что такое наследование?

Это свойство, позволяющее описать новый класс на основе уже существующего с частично или полностью заимствующейся функциональностью.

б. Что такое виртуальная и чисто виртуальная функция?

Виртуальная функция - это функция, которая определяется в базовом классе, а любой порожденный класс может ее переопределить. Отличается об обычной функции тем, что для обычной связывание вызова функции с ее определением осуществляется на этапе компиляции. Для виртуальных функций это происходит во время выполнения программы.
Чистая виртуальная функция - это метод класса, тело которого не определено.

в. Какие классы называют конкретными, абстрактными?

Абстрактный класс не предполагает создания экземпляров. Конкретный, соответственно, наоборот.

г. Какие классы называют корневыми, листовыми?

Листовой класс - класс, который не может быть унаследован. Корневой класс - главный предок классов.

д. Для чего вводится защищенная часть класса?

Для того, чтобы сделать методы/свойства недоступными из объекта, реализующего класс, но доступными к использованию в дочерних классах.

5. Наследование и типизация:

а. Что такое открытое (public) наследование?

+

б. Что такое закрытое (private) наследование?

+

в. Что такое защищенное (protected) наследование?

Пусть B унаследован от A. Во всех случаях класс B получает доступ к открытым и защищённым членам A.
Открытое - Открытые члены A становятся открытыми в B. Защищённые A - защищёнными B. Говорят, что класс наследует как интерфейс, так и реализацию предка.
Закрытое - Открытые и защищённые члены A становятся закрытыми в B. Применяют когда класс должен наследовать только реализацию.
Защищённое - открытые и защищённые члены A становятся защищёнными в B.

г. Допустимо ли присваивание объекту класса-родителя значения класса-потомка? Если да, то каким образом оно реализуется?

+

д. Допустимо ли присваивание объекту класса-потомка значения класса-родителя? Если да, то каким образом оно реализуется?

По иерархии только дочерние классы могут наследоваться от родителя, наоборот - ошибка. С точки зрения интерфейсов, каждый производный класс полностью реализует интерфейс родительского класса.

е. Что такое чистый полиморфизм?

В случае чистого полиморфизма имеется одна функция (тело кода) и несколько ее интерпретаций. Чистый полиморфизм позволяет реализовывать обобщенные алгоритмы.

ж. Что такое повышающее приведение?

+

з. Что такое понижающее приведение?

Преобразования типов разделяют по направлению. О направлении преобразования говорят при наследовании. Возможно направление от родителей к потомкам (понижающее) и от потомков к родителям (повышающее).

6. Множественное наследование:

а. Что такое множественное наследование?

Множественное наследование позволяет одному дочернему классу иметь несколько родителей.

б Что такое конфликт имен?

Конфликт имен возникает, когда два одинаковых идентификатора находятся в одной области видимости, и компилятор не может понять, какой из этих двух следует использовать в конкретной ситуации.

в. Что такое повторное наследование?

Это способность элементов служить для построения многих различных приложений.

г. Что такое ромбовидная структура наследования?

Это когда два класса B и C наследуют от A, а класс D наследует от обоих классов B и C. При этом может возникнуть неоднозначность: если объект класса D вызывает метод, определенный в классе A (и этот метод не был переопределен в классе D), а классы B и C по-своему переопределили этот метод, то от какого класса его наследовать: B или C?

д. Что такое виртуальное наследование?

Вариант наследования, который нужен для решения проблем, порождаемых наличием возможности множественного наследования (особенно «ромбовидного наследования»), путём разрешения неоднозначности того, методы которого из суперклассов (непосредственных классов-предков) необходимо использовать.