Черник Матвей 212 группа

Отличие структуры от класса?

Класс и структура в чем-то похожи между собой, а в чем-то совсем нет. С уверенностью можно сказать, что любую структуру в языке си++ можно представить в виде класса. Давайте в этом убедимся. И то, и другое по своей сути является новым объектом, грубо говоря, мы можем сказать, что новый созданный нами объект - новый тип данных как int или double, только устроен он сложнее. Итак, рассмотрим пример структуры двумерного вектора Vector2d

Struct Vector2d{double x ; double y;};

В языке си++ внутри структуры мы можем также функции-члены, которые могут работать с этими данными. Например, функция, вычисляющая длину вектора или прибавляющая к вектору другой вектор. Однако принято структуру, которая содержит функции-члены называть классами(при это используется уже ключевое слово class, а не struct), а сами ф-ч методами. Более серьезное отличие это модификаторы доступа private, public, protected. Модификатор доступа public говорит о том, что поле или метод могут быть использованы кем угодно. Модификатор доступа private говорит о том, что поле или метод могут быть использованы только методами того же самого класса В классах (class) по умолчанию доступ к элементам private, а в структурах (struct) — public.

Это единственное их отличие.

Классы в C++ могут содержать специальные методы, которые вызываются при создании объекта (конструкторы) и при разрушении объекта (деструкторы).

Как искать классы?

При исходных данных найти класс очень просто, но очень часто это можно сделать разными способами. Пусть для примера нам нужно найти класс подстановок. Мы представляем этот объект и понимаем, какие ключевые данные нам нужны (выбираем из тех что нам предоставил заказчик).Для начала разберемся с полями класса в нашем случае это будут размер подстановки и массив из подмножества натуральных чисел, который будет содержать элементы цикла. Подчеркну, что уже на этом этапе зачастую возможны несколько вариантов, чтобы задать поля класса. Следующим этапом будет реализация методов и перегрузок операций, все это зависит от задач, которые нам необходимо выполнить с данным объектом. Например, если в дальнейшем мы хотим посчитать определитель с помощью наших подстановок нам обязательно нужна перегрузка операции композиция. Самое важное, что нужно нам это составить конструктор и деструктор для выделения памяти под объект а затем под его освобождение.

Что такое namespace (пространство имен)?

Пространством имен в языке си++ мы называем множество, в котором хранятся определенные функции, классы и другие взаимосвязанные между собой объекты. Допустим, я хочу использовать уже готовую функцию из библиотеки, например, iostream, я просто прописываю в заголовке программы #include<iostream> и начинаю использовать эту функцию, и после компиляции у меня вдруг вылезает ошибка. В чем проблема? Дело в том, что у меня может возникнуть конфликт имен. Когда я подключаю библиотеку, может возникнуть ситуация, что и в моей программе, и в библиотеке оказались функции, отличающиеся по своему предназначению, но сигнатура у них вдруг либо очень похожа, либо неотличима. На этот случай и было придумано пространство имен. Теперь, когда мне необходимо использовать конкретную функцию из библиотеки iostream, я припишу к

ней приставку std:: .Есть и другой способ избежать ошибки: в заголовке файла программы сразу после #include<…> написать using namespace …(пространство имен, которое нам необходимо, в данном конкретном случае это std, поскольку все стандартные библиотечные функции языка си++ вынесены в это пространство имен). Но использовать это стоит на свой страх и риск, поскольку в такой ситуации высокий риск словить конфликт имен.