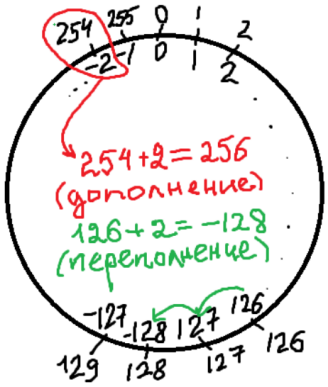
Лекция 1. Углубленный C/C++.
Введение в C



- Задания сдавать можно строго **ДО ДЕДЛАЙНА**
- Логический тип отсутствует. Значение “0” (и приводимые к нему) считается ложью, все остальное - истина.
- На примере char. Первый бит знаковый. Дополнительный код числа: все информационные биты инвертируются, и к полученному результату добавляется 1.
  -1 в дополнительном коде – это:
  00000001 → 11111110 → 11111111
- **!!a** – нормализация истинного выражения к 1, а ложного к 0
- Тернарный оператор: (x < y) ? x + 15 : y – 25
- В си, при отсутствие аргументов функция может принимать произвольное кол-во аргументов → def foo(void) – отсутствие аргументов
- `scanf(“%*[^\n]”)` – считывание из буфера ввода всех мусорных символов до перевода строки
- `define MY_CONST 100` – объявление константы. “define” не знает языка и не проверяет типы и приоритеты
- Именованные перечисления (enum) позволяют задать именованную группу констант. При выхождении за рамки в C ошибки не происходит.
- Описание – declaration, определение – definition. Определение подразумевает наличие реализации
- Как стоит размещать код в программе:
  - раздел описаний (константы, интерфейсы)
  - main (точка входа в программу)
  - раздел определений (реализации)
- SRP – single responsobility principle
- switch { case () : }. !После захождения в один кейс, продолжается обход остальных!
- Этапы выполнения программы на языке Си:
  - Обработка препроцессором
    1. обработка программы как текста
    2. выполнение директив программиста для преобразования текста в код
  - Компиляция
    1. Превращает файл в объектный файл
  - Линковка
    1. связывание сущностей из разных объектных файлов,
    2. объединение программы в единый исполняемый файл

- Обработка препроцессором - выполнение #include #define #pragma. Для инклуда, “” - означает сначала поиск в текущей дериктории. <> - означает сначала поиск в путях компилятора и системных путях.
- Классы памяти:
  - Автоматический: располагается на стеке
  - Регистровый
  - Статический без связывания на уровне блока: **static,** будет равна нулю а не мусору. Привязана к запуску функции → для функции единственный экземпляр
  - Статический с внешним связыванием на уровне файла: **extern.**
  - Статический с внутренним связыванием: static → функция или переменная будет приватная на уровне этого файла
- Если создать переменную на стеке в функции а потом выйти из этой функции сохранив ссылку на эту переменную. То при вызове следующих функций эта переменная может перетереться расширяющимся стеком.
- Проверить указывает ли указатель на реальный участок памяти или мусор – невозможно. Только попробовав прочитать и обработать ошибку. Поэтому удаленный указатели нужно занулять
- Указатели на константы и константные указатели:
  - `const int * ptr1;`
  - `int *const ptr2.`
  - Читается справа налево, как в английском
- Приведение типов указателей: можно только ужесточать ограничения.
- `Int a[1000] = {0};` – массив из тысячи нулей
- Имя массива является константным указателем на нулевой элемент.
- Для многомерных массивов в качестве первого параметра размерности при передаче в функцию можно указывать любое число
- Массивы расположены последовательно непрерывно в памяти. Структуры – нет
- Point a = {0, 0} – инициализировать структуру можно только при определении.
- Анонимная структура – структура, используемая один раз
- В С нужно всегда писать struct → используем синоним `typedef struct Poin Point`
- `union` Объединение – тип данных, позволяющий интерпретировать область памяти как переменные разных типов