

Преобразования*

Лекция 4

* значений и
функций

План лекции

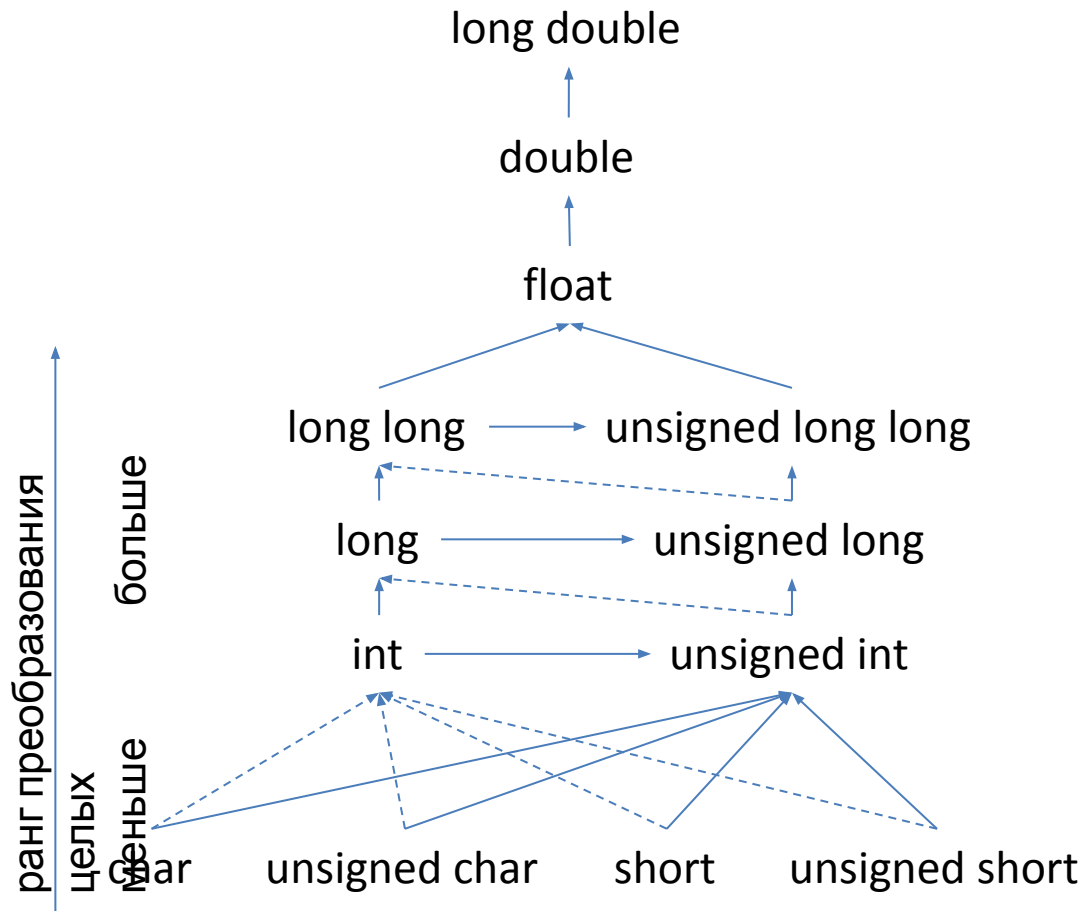
- Преобразования
 - Целых и типов с плавающей точкой
 - l-value
 - Массивов
 - Функциональных типов
 - С типом void
 - Указателей

Простая часть

- Неявные (автоматические)
- Явные
- Преобразование в совместимый тип сохраняет значение и представление значения
- Конец простой части

Непростая часть

Неявные арифметические преобразования



- При выполнении операции x op y , кроме операций присваивания и сдвига, с операндами числовых типов $T1$ и $T2$, значения выражений x и y преобразуются к типу T , который удовлетворяет условиям
 - Есть путь из $T1$ в T
 - Есть путь из $T2$ в T
 - T – наименьший из возможных (если есть путь из $T1$ в TT и из $T2$ в TT , то есть путь из T в TT для любого TT)
- Пунктирная стрелка выбирается, если множество значений нижнего типа $\subseteq$ множество значений верхнего типа; иначе выбирается сплошная стрелка
- Переход от битового поля, `char`, `unsigned char`, `short`, `unsigned short` к `int` или `unsigned int` называется целочисленное повышение
- Для присваивания и сдвига результат всегда преобразуется к типу левого операнда

Преобразования целых

- Если значение представимо в T , то преобразование к T сохраняет значение
 - И может изменить представление
- Если T беззнаковый, то к значению добавляется или вычитается $1 + \text{max}(\text{диапазон } T)$ до тех пор, пока результат не попадет в диапазон T
- Иначе результат преобразования зависит от реализации (implementation-defined)
 - Например, преобразование может привести к возникновению «исключительной ситуации» (exception)

Преобразования целых и с плавающей точкой

- Преобразование конечного числа с плавающей точкой в целое = округление к нулю
 - Если целая часть выходит за диапазон целого типа, то поведение не определено
- Если целое представимо в типе с плавающей точкой точно, то значение сохраняется
 - Представление может измениться
- Если целое попадает в диапазон типа с плавающей точкой, то ближайшее к нему меньшее и большее значение с плавающей точкой
 - Выбор зависит от реализации
- Если целое не попадает в диапазон типа с плавающей точкой, то результат не определен

Преобразования для типов с плавающей точкой

- Преобразования по цепочке float --> double --> long double сохраняют значение
 - Представление может измениться
- Преобразование к меньшему типу T с плавающей точкой
 - Если значение представимо в T точно, то оно сохраняется
 - Если значение попадает в диапазон T, то выбирается ближайшее меньшее или большее значение T
 - Зависит от реализации
 - Если не попадает в диапазон T, то поведение не определено

Преобразования других целых типов

- Ранг расширенного целого типа ниже ранга стандартного типа той же ширины
 - Соотношение рангов расширенных целых типов одинаковой ширины является implementation-defined
- C99: Ранг `_Bool` ниже ранга любого другого целого типа

Преобразование l-value в обычное значение

1/2

- L-value – это выражение имеющее полный тип или неполный тип, отличный от void, и обозначающее значение в памяти
- Изменяемое l-value – это l-value, которое
 - Не является массивом
 - Имеет полный тип без квалификатора const
 - Если это struct или union, то все его элементы (рекурсивно) имеют тип без квалификатора const

Преобразование l-value в обычное значение

2/2

- Если l-value не является массивом и не является операндом sizeof, унарного &, ++, --, ., оператора присваивания, то оно преобразуется в обычное значение
 - Тип результата = тип l-value без квалификаторов
- Если l-value имеет неполный тип, не является массивом и является операндом, то поведение не определено

Преобразование массивов

- Массив типа T преобразуется в указатель на нулевой элемент массива
 - Этот указатель не является l-value
- Кроме массивов, являющихся
 - Операндом sizeof
 - Операндом унарного &
 - Строковым литералом, инициализирующим массив

Преобразование функциональных типов

- Именуемое выражение функции (function designator) – это выражение, имеющее функциональный тип
- «Функция, возвращающая T», преобразуется к типу «указатель на функцию, возвращающую T»
- Кроме именуемых выражений функции, являющихся
 - Операндом sizeof
 - При этом операнду sizeof запрещено иметь функциональный тип
 - Как вы это понимаете?
 - Унарного &

Преобразования типа void

- Тип void нельзя преобразовывать
- Любой тип можно преобразовать к void при этом значение
 - Вычисляется
 - Так как вычисление может иметь побочные эффекты
 - Становится недоступным

Безопасные преобразования указателей

- Целое 0 в указатель любого типа
 - Получается нулевой указатель, отличный от всех остальных указателей
- `void*` в любой `T*`
- Любой `T*` в `void*`
- `T*` в `const T*` и `volatile T*`
- Во всех случаях:
 - Меняется только тип выражения
 - Значение указателя не меняется
 - well-defined

Другие преобразования указателей

- Целое в указатель
 - Implementation-defined, результат может быть «негодным» указателем
- Указатель T^* в целое типа T_1
 - Implementation-defined, если $\text{sizeof}(T^*) \leq \text{sizeof}(T_1)$
 - Undefined behavior иначе
- Любой T_1^* в T_2^*
 - Меняется тип выражения, значение указателя сохраняется
 - Undefined behavior, если значение указателя не выровнено для типа T_2
- Указатель на функцию в указатель на любую другую функцию
 - Меняется тип выражения, значение указателя сохраняется
 - Undefined behavior, если при вызове тип именуемого выражения функции не совместим с типом вызываемой функции

Заключение

- Преобразования
 - Целых и типов с плавающей точкой
 - l-value
 - Массивов
 - Функциональных типов
 - С типом void
 - Указателей