1. Что такое определение, объявление, инициализация, присваивание и неопределённое поведение?

Объявление - это сообщение о том, что объект такого типа с такми именем существует.
Определение - это создание объекта (переменной, функции, класса, объекта класса, лямбды...). Причём объект - это не обязательно данные, это объект в самом широком смысле. Любое определение также является и объявлением
Инициализация - присваивание значения в момент объявления
Неопределенное поведение - поведение, которое может возникать в результате использования ошибочных программных конструкций или некорректных данных, на которые Международный Стандарт не налагает никаких требований. Неопределенное поведение означает, что произойти может что угодно, дальнейшие события никак не регулируются и определяются случайными факторами.

2. Объясните следующие понятия: объект, переменная, литерал, оператор, выражение, инструкция.

Объект - некоторая сущность в цифровом пространстве, обладающая определённым состоянием и поведением, имеющая определённые свойства (атрибуты) и операции над ними (методы). Как правило, при рассмотрении объектов выделяется то, что объекты принадлежат одному или нескольким классам, которые определяют поведение (являются моделью) объекта.
Переменная - именованный "контейнер"для хранения определенного типа данных.
Литерал - безымянная константа, своеобразная запись в программе, которая выражена определенным фиксированным значением. Литералы не могут изменяться по мере обработки программного кода и изменить их можно только в режиме редактирования, это их основное отличие от перменных.
Оператор - команда, обозначающая определенное математическое или логическое действие, выполняемое с данными (операндами).
Выражение - это код, который после выполнения возвращает какое-либо значение. Например, 2+3. Отдельная константа или отдельная переменная также является выражением, значением которого является, соответственно, сама константа или значение переменной.
Инструкция - отдельное предложение, предписывающее компилятору выполнить некоторые действия, команда для компьютера. Из последовательных инструкций состоит код.

3. В чём заключается опасность использования оператора goto и когда его можно использовать?

Оператор goto используется для изменения нормальной последовательности выполнения программы путем передачи управления какой-либо другой части программы. Этот оператор дает возможность перейти к любой части программы, но часто он делает программу сложной и запутанной. Также нужно обращать внимание на то, что инструкция goto и соответствующая ей метка инструкции должны находиться в одной функции. Всегда в программе можно обойтись без этого оператора, причем практически всегда с использованием других конструкций код становится четче и понятнее. Goto создает "спагетти-код в

нем нет порядка, все запутано и скручено.
Тем не менее, в редких случаях альтернативы значительно ухудшают читабельность кода, и тогда стоит аккуратно использовать goto.

4. Приведите примеры неправильного или опасного использования указателей, массивов и ссылок.

4.1. Неправильное использование указателей.

```
- Неинициализированный указатель:
int* ptr;
*ptr = 10; // Некорректное обращение к неинициализированному указателю

- Ошибки при работе с динамической памятью:
int* ptr = new int;
delete ptr;
*ptr = 10; // Некорректное обращение к удаленной памяти

- Утечка памяти:
void foo(){
    int* ptr = new int;
    // ...
    return; // Забыли освободить память с помощью delete
}
```

4.2. Неправильное использование массивов.

```
- Обращение за пределы массива:
int arr[5];
arr[10] = 10; // Обращение за пределы массива

- Некорректное копирование массива:
int arr1[5] = {1, 2, 3, 4, 5};
int arr2[5];
arr2 = arr1; // Некорректное копирование массива
```

4.3. Неправильное использование ссылок.

```
- Ссылка на временный объект:
int& foo() {
    int x = 10;
    return x; // Возвращаем ссылку на локальную переменную
}

int main() {
    int& ref = foo(); // Некорректное использование ссылки
}
```

```
- Нулевая ссылка:
  int* ptr = nullptr;
  int& ref = *ptr; // Некорректное использование нулевой ссылки
```

5. Перечислите и прокомментируйте рассмотренные особенности передачи аргументов в функции.

Аргументы, которые представляют переменные или константы, могут передаваться в функцию по значению (by value) и по ссылке (by reference).
При передаче аргументов по значению функция получает копию значения переменных и констант. Поэтому после выполнения функции значение самой переменной не меняется, меняется лишь значения копии. При передаче аргументов по ссылке функция получает параметр, который связывается непосредственно с объектом, поэтому через ссылку можно менять сам объект. Таким образом, после выполнения функции изменится значение переменной, ссылку на которую мы передавали.
При передаче параметров по значению значения копируются в определенные участки памяти, которые потом использует функция, поэтому для передачи объемных параметров, например, массивов, pass by value невыгодно. В случае передачи параметров по ссылке не нужно копировать все содержимое объекта в участок памяти, за счет чего увеличивается производительность программы.
При передаче в функцию по значению C++ может автоматически преобразовывать значения одних типов в другие, в том числе если подобные преобразования сопровождаются потерей точности (например, преобразование от типа double к типу int). Но при передаче параметров по ссылке неявные автоматические преобразования типов исключены. Если при передаче по значению переданное число double успешно преобразуется в int (пусть и с потерей точности), то при передаче по ссылке мы столкнемся с ошибкой на этапе компиляции.