1. **Определение функции**: Определение функции в C++ представляет собой описание того, что функция делает и какие аргументы принимает. Это включает в себя имя функции, тип возвращаемого значения, список параметров (формальные аргументы) и тело функции.
2. **Формальные и фактические аргументы функции**:
   * Формальные аргументы - это параметры, объявленные в определении функции.
   * Фактические аргументы - это значения, передаваемые в функцию при ее вызове. Пример:
3. **Объявление функции**: Объявление функции - это указание компилятору на существование функции с определенной сигнатурой. Оно предшествует вызову функции и обычно содержит только имя функции, тип возвращаемого значения и список параметров, но без тела функции.
4. **Вызов функции**: Вызов функции - это исполнение кода функции в программе.
5. **Передача параметров в функцию**: Параметры могут передаваться в функцию различными способами, включая передачу по значению, по ссылке и по указателю.
6. **Способы передачи параметров**:
   * Передача по значению: Значение параметра копируется в функцию.
   * Передача по ссылке: Адрес переменной передается в функцию, что позволяет функции изменять значение переменной.
   * Передача по указателю: Функция получает указатель на переменную, что также позволяет ей изменять значение переменной.
7. **Модель памяти С/С++ (классы памяти) - назначение стека**: В C/C++ используется модель памяти, включающая стек, кучу и статическую область памяти. Стек используется для хранения локальных переменных, аргументов функций и адресов возврата из функций.
8. **Открытие окна стека вызовов**: В большинстве сред разработки (IDE) окно стека вызовов доступно через отладочные инструменты во время выполнения программы.
9. **Информация в окне стека вызовов**: Окно стека вызовов отображает текущий стек вызовов программы, показывая последовательность вызовов функций и их текущие состояния. Это может включать имена функций, параметры функций, адреса возврата и другую отладочную информацию.
10. **Функция Checking**:
    * Она принимает один аргумент - целое число **x**, представляющее год.
    * Функция проверяет, является ли год високосным, и возвращает **true**, если год високосный, и **false** в противном случае.
11. **Функция quantity**:
    * Принимает три аргумента: **x** - месяц, **y** - день и **z** - год.
    * Функция вычисляет порядковый номер дня в году (от 1 до 365 или 366 для високосного года).
    * Использует массив **Mounthes**, чтобы определить количество дней в каждом месяце.
12. **Функция nameMounth**:
    * Принимает два аргумента: **mes** - номер месяца и **Months** - массив с количеством дней в месяцах.
    * Функция выводит название месяца на основе его номера.
13. **Функция day\_prog**:
    * Принимает один аргумент - год.
    * Функция выводит дату Дня программиста в зависимости от типа года.
14. **Функция Finding\_number\_day**:
    * Принимает три аргумента: **day**, **month** и **year**.
    * Функция вычисляет порядковый номер дня в году, учитывая, является ли год високосным.

Также в коде встречается директива **#include "stdafx.h"**, которая может использоваться в Visual Studio для предварительно скомпилированных заголовочных файлов, но в данном случае она не обязательна и можно было бы обойтись без нее.

Используется директива **#pragma once**, которая предотвращает многократное включение заголовочных файлов.

Также определен макрос **chislo**, который определяет значение 9, но в данном фрагменте кода он не используется.

Начало формы