Help: настройка Visual Studio.Net для создания консоль-приложения на основе Intel C++ с применением OpenMP. Инструменты

«Практическое параллельное программирование
в системах с общей памятью»

Калинина А.П.

Содержание

1. Последовательность установки программ

2. Создание консоль-приложения на C++ с применением OpenMP (Microsoft C++, Intel C++)

3. Intel Thread Checker (тестирование правильности выполнения многопоточного приложения)

4. Intel Thread Profiler (тестирование производительности многопоточных вычислений)

Последовательность установки

1. Visual Studio.Net
2. Intel C++
3. Intel VTune Performance Analyzer
4. Intel Thread Checker или Intel Thread Profiler

2. Создание консоль-приложения на C++ с применением OpenMP (Microsoft C++, Intel C++)

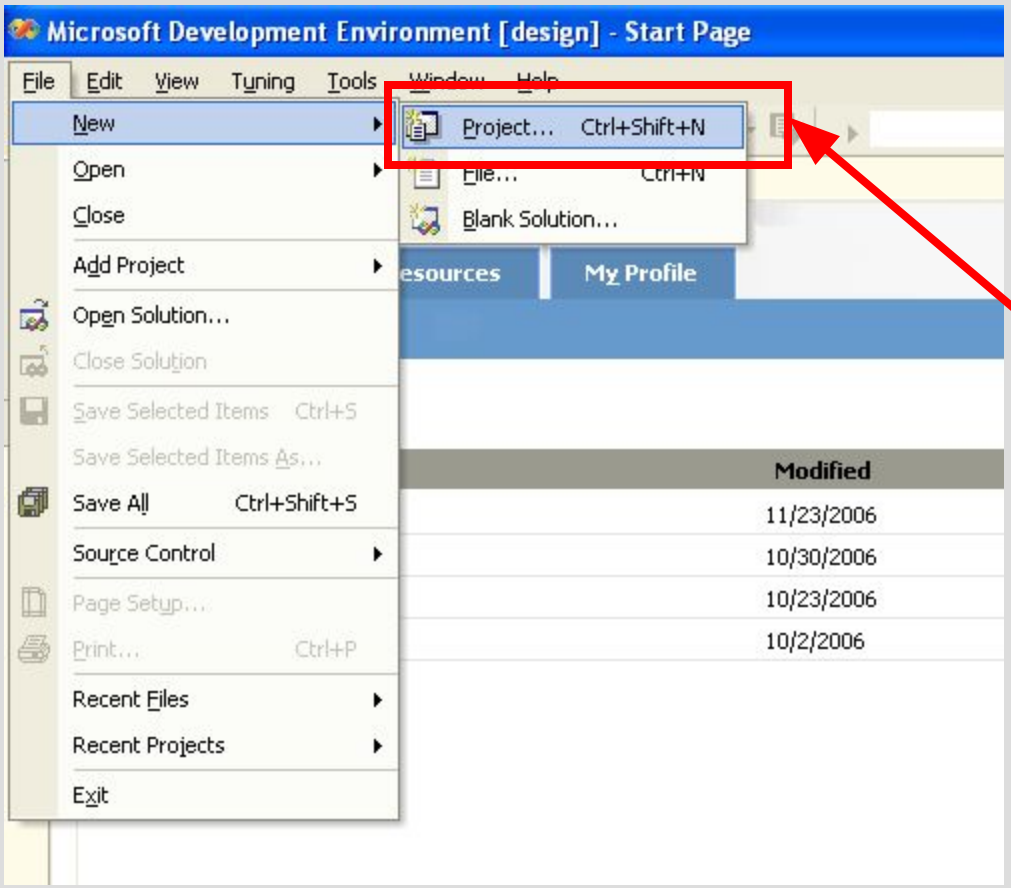
2.1. Создание консоль-приложения на основе C++

2.2. Преобразование в проект на основе Intel C++

2.3. Установка поддержки директив OpenMP

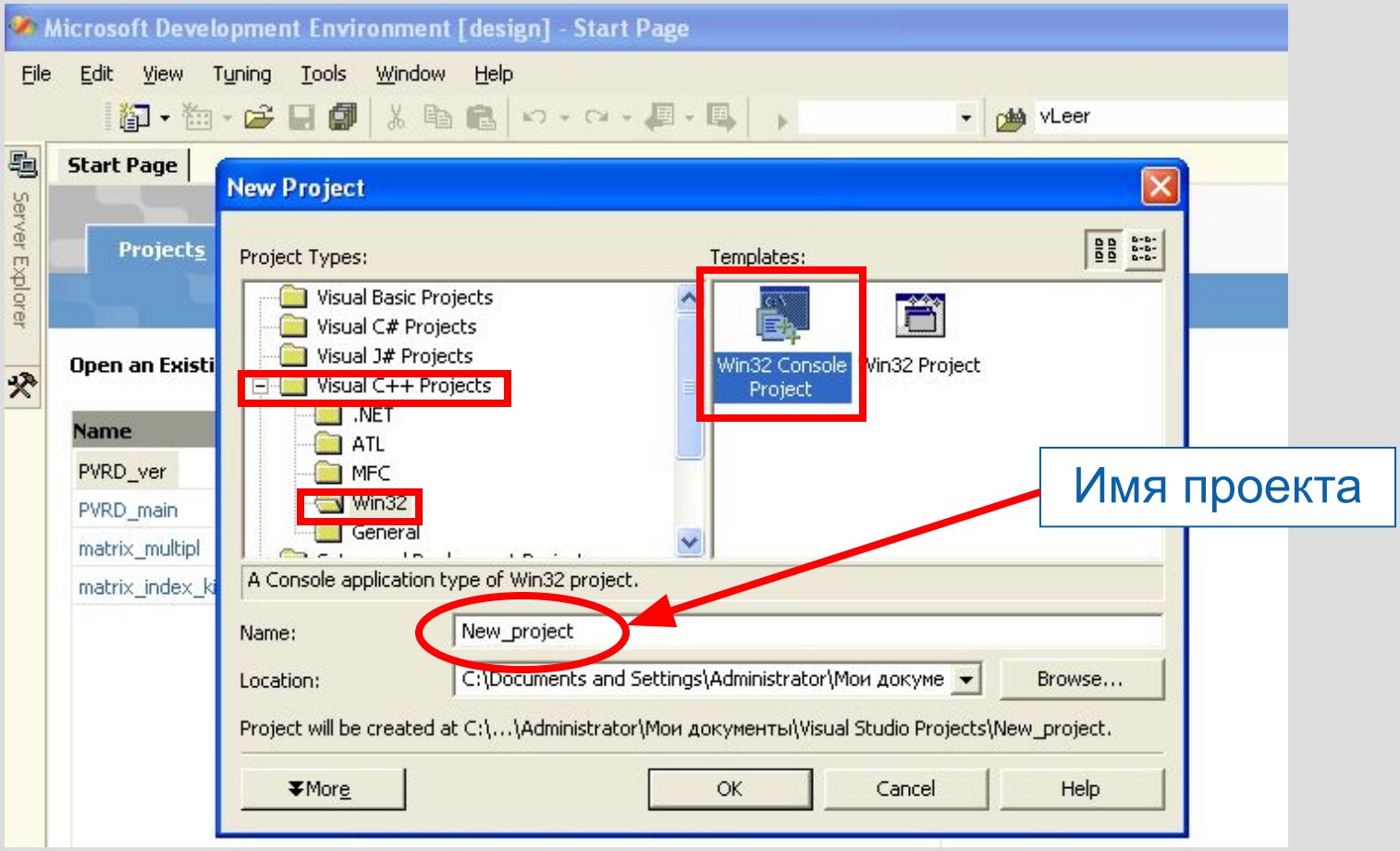
2.3. Настройка на многопоточно-безопасные библиотеки

2.1. Создание консоль-приложения на C++



Новый проект

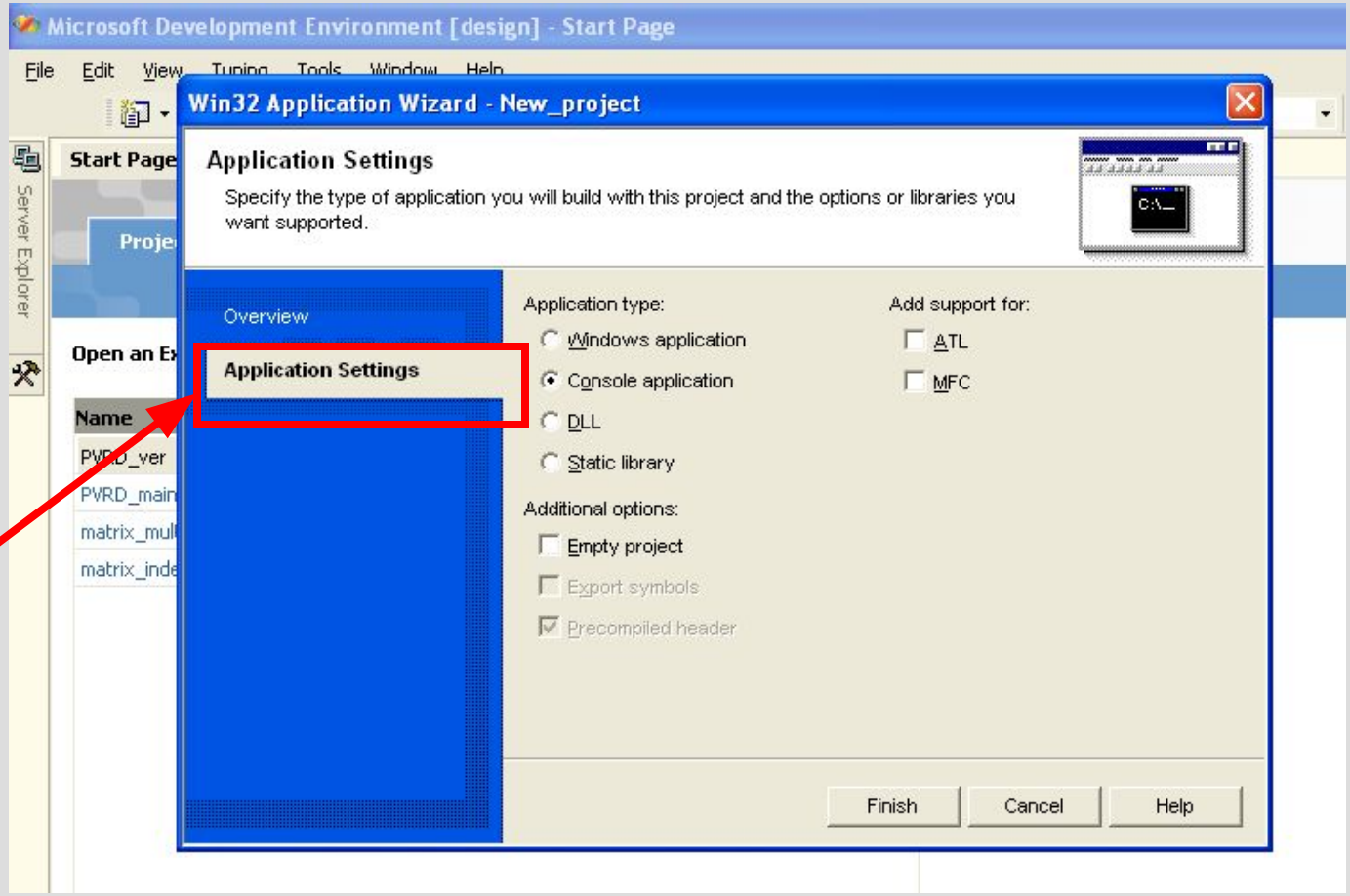
2.1. Создание консоль-приложения на C++ (слайд 2)



2.1. Создание консоль-приложения на C++ (слайд 3)

Выбрать

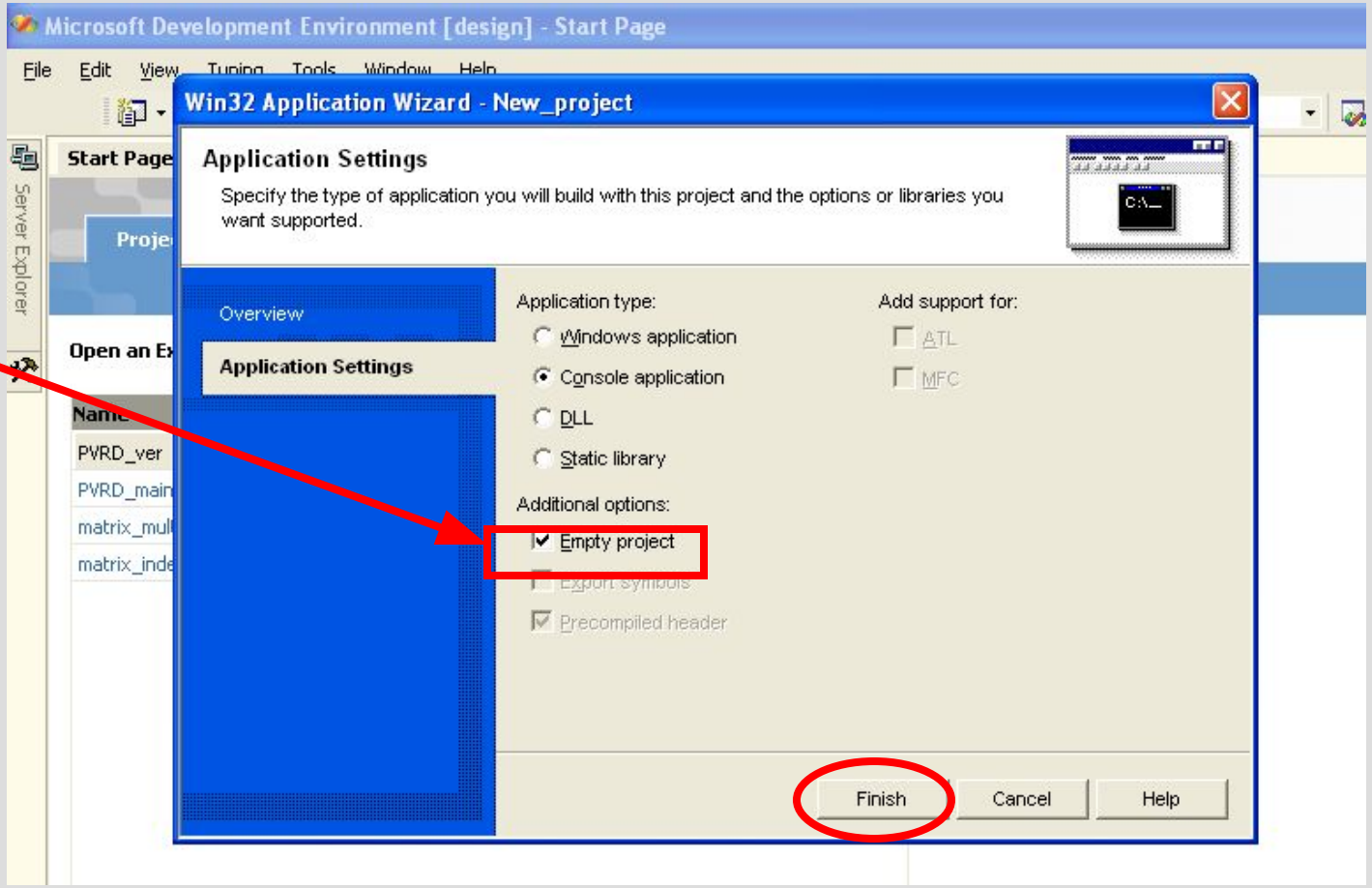
«Application Settings»



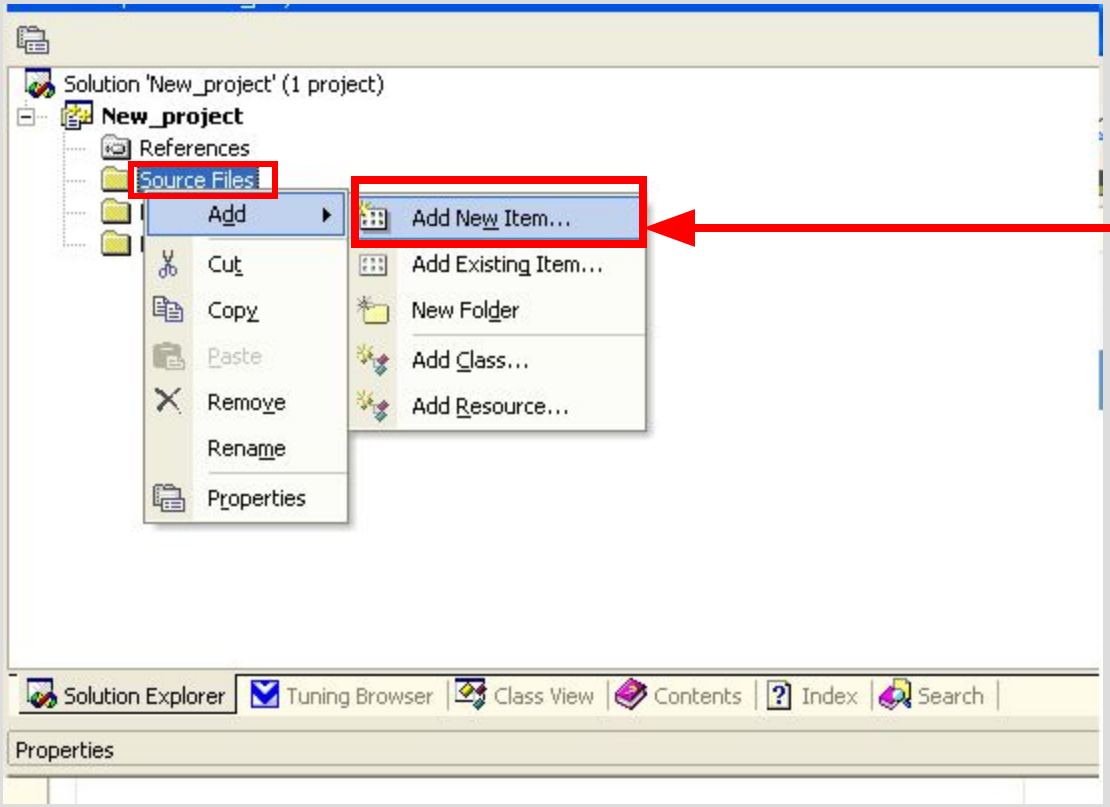
2.1. Создание консоль-приложения на C++ (слайд 4)

Установить

«пустой проект»

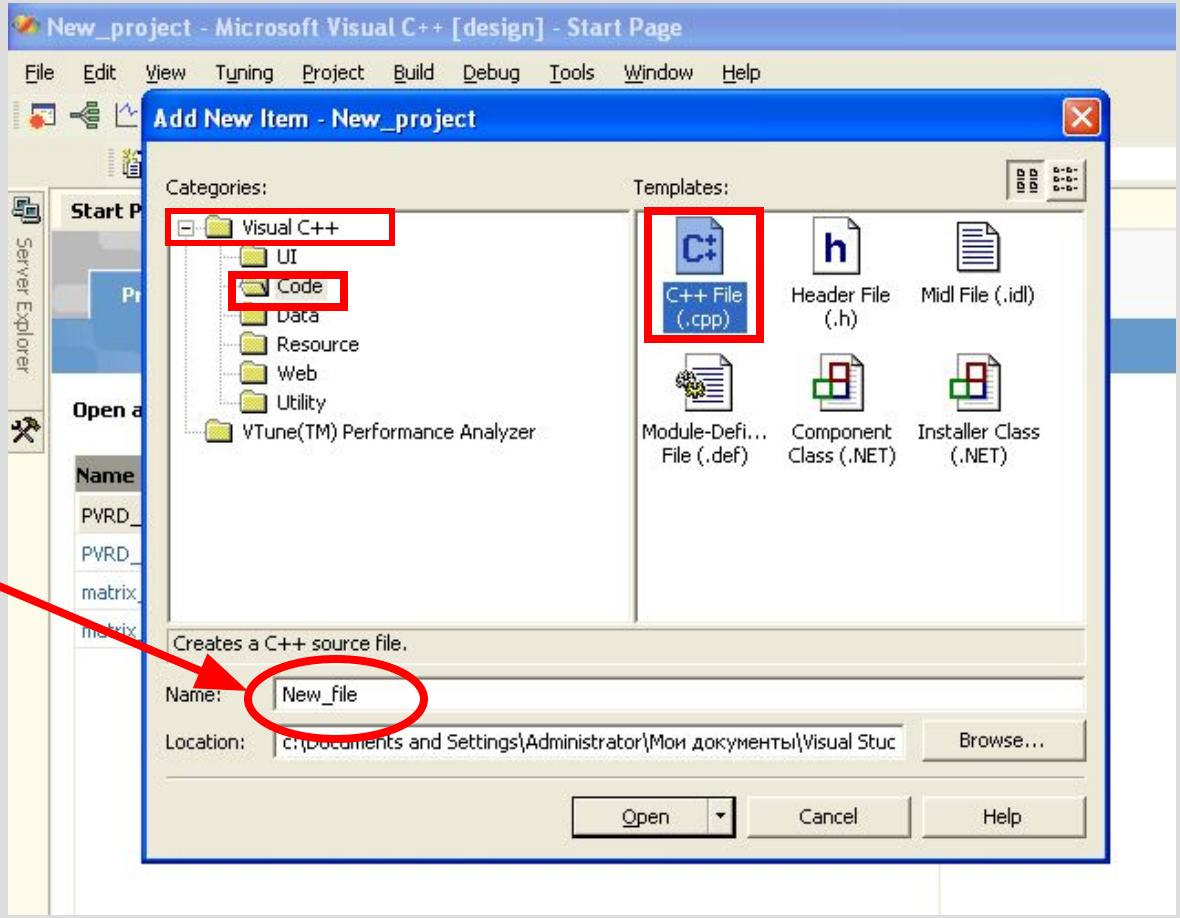


2.1. Создание консоль-приложения на C++ (слайд 5)



Создание *.cpp

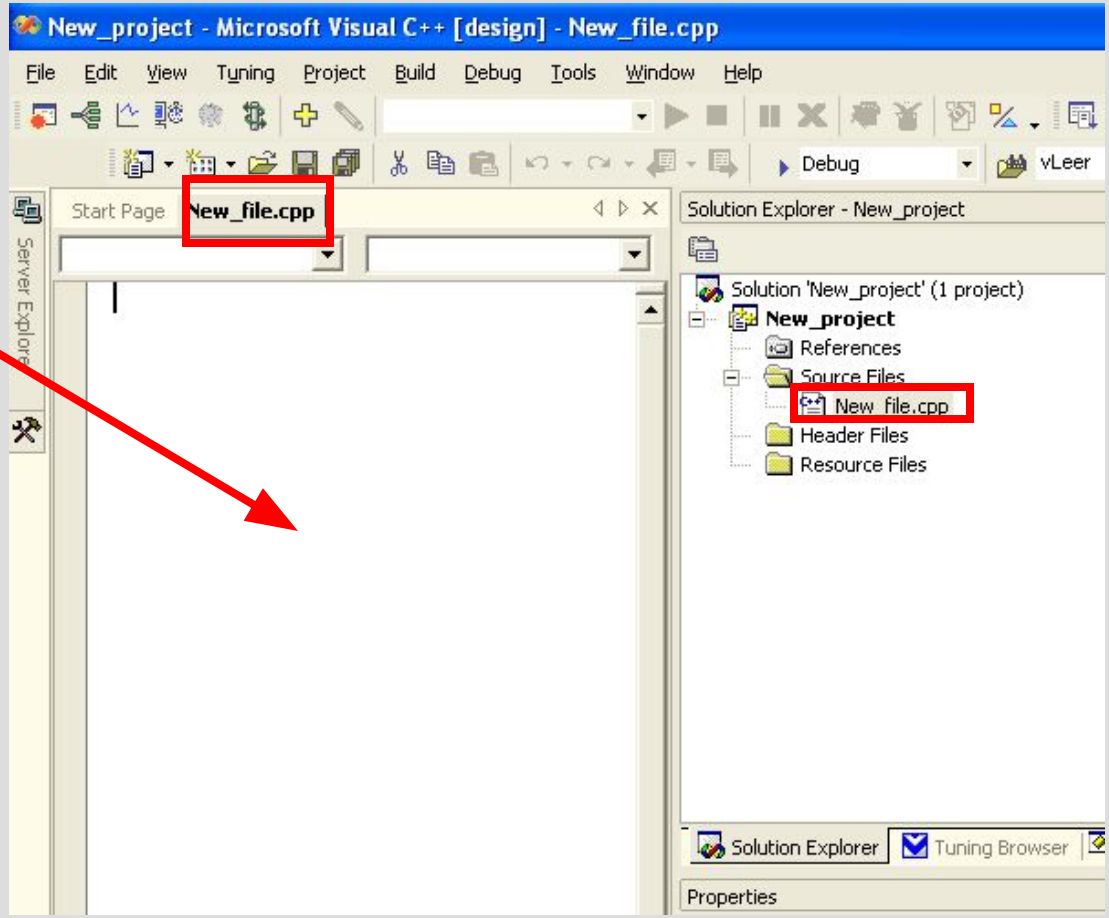
2.1. Создание консоль-приложения на C++ (слайд 6)



Ввести имя файла

2.1. Создание консоль-приложения на C++ (слайд 7)

В открывшемся окне набрать текст новой C++ - программы или скопировать в это окно текст уже имеющегося *.cpp

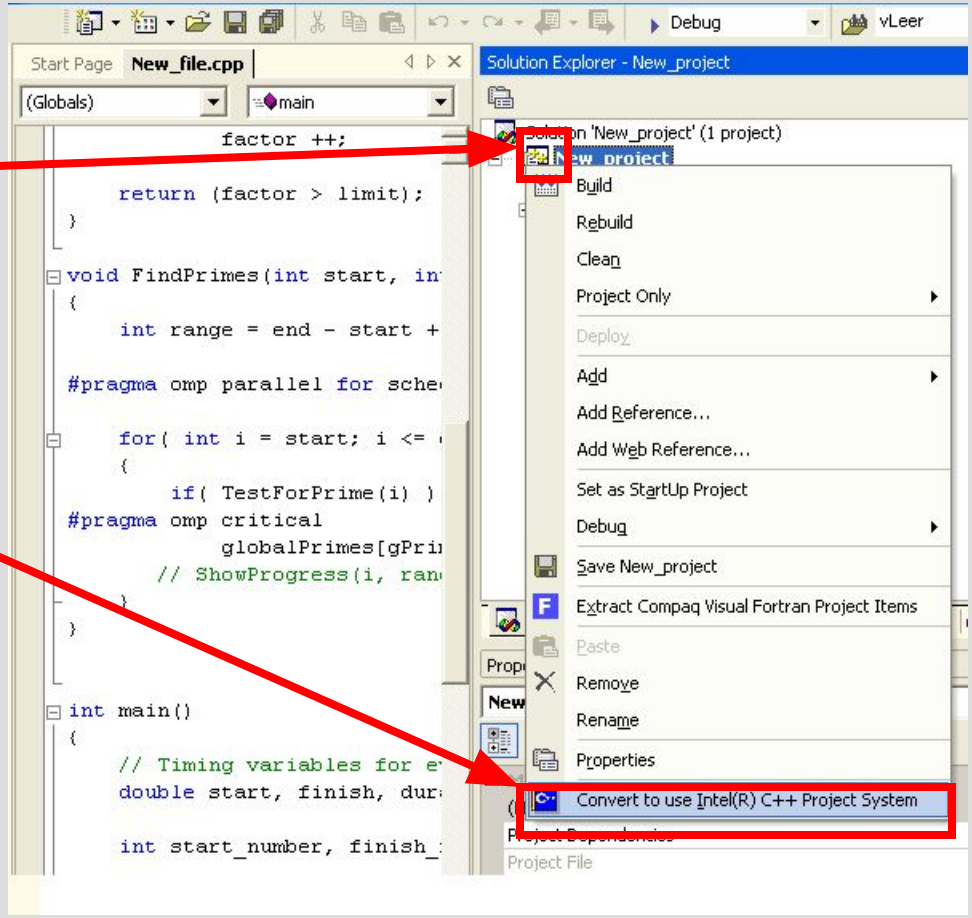


2.2. Преобразование в проект на основе Intel C++

С помощью щелчка правой кнопки мыши на значке проекта открыть контекстное меню и выбрать самый нижний пункт меню –

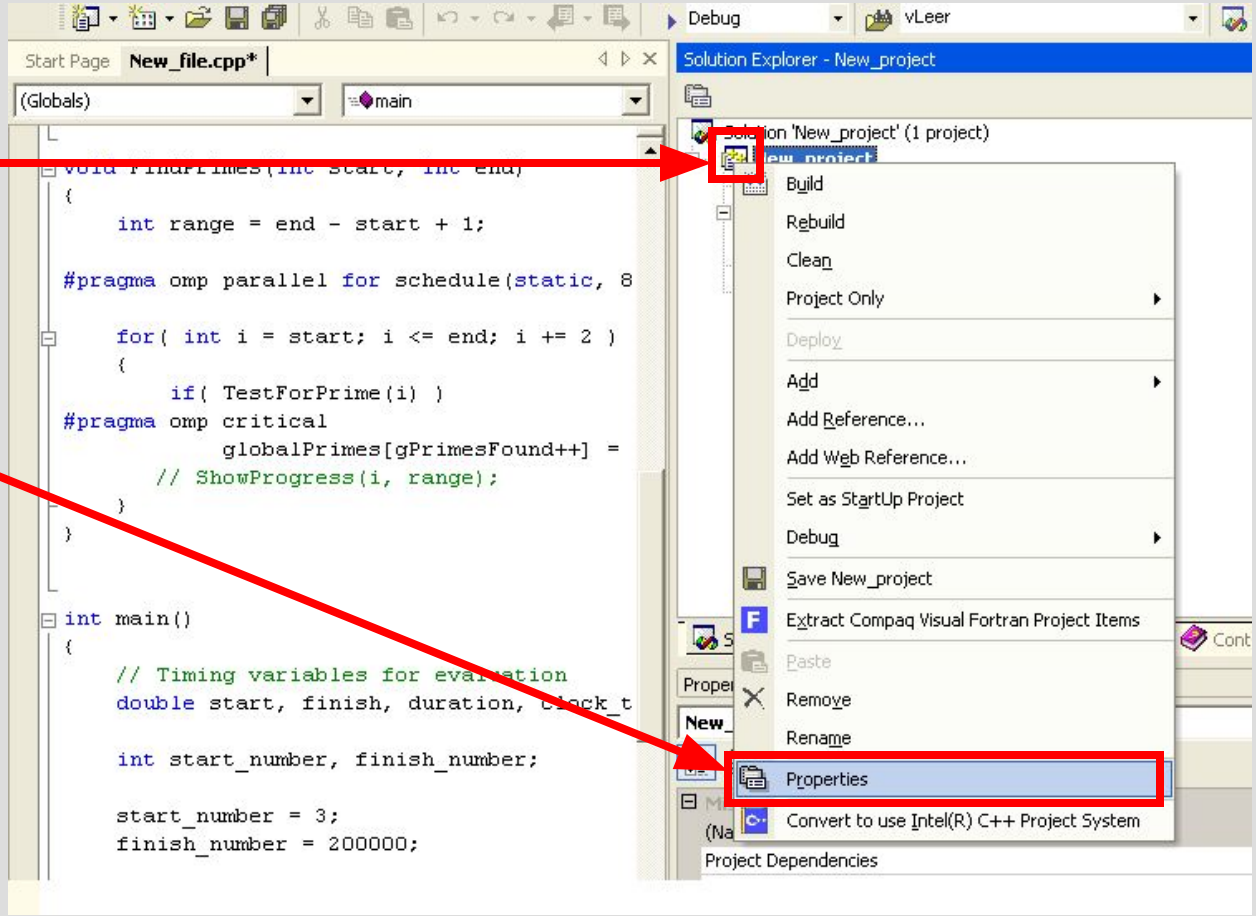
«Convert to use Intel(R) C++ project System»

- преобразование в проект на основе Intel C++



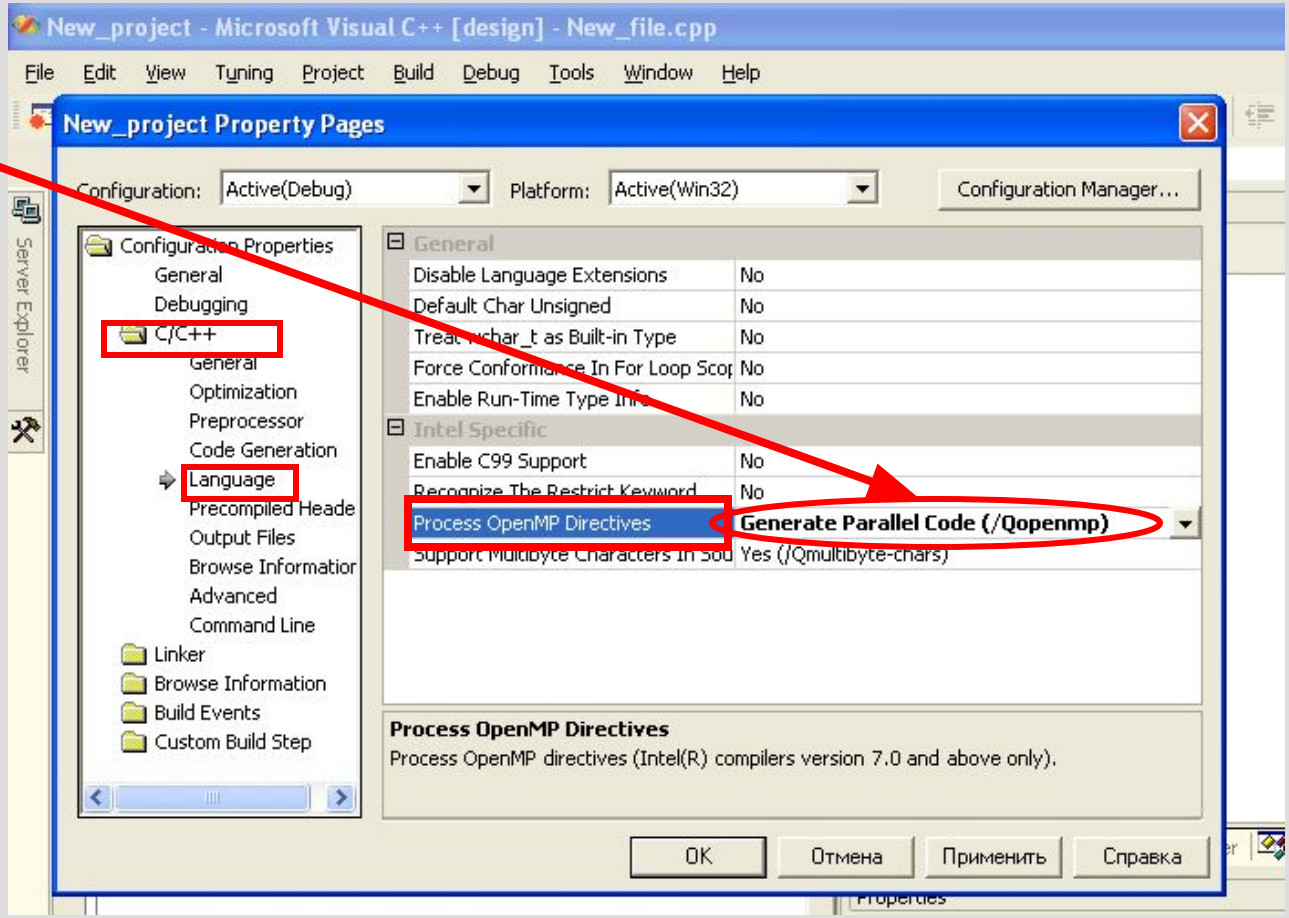
2.3. Установка поддержки директив OpenMP (слайд 1)

Правым щелчком мыши на значке проекта открыть контекстное меню и выбрать окно свойств проекта



2.3. Установка поддержки директив OpenMP (слайд 2)

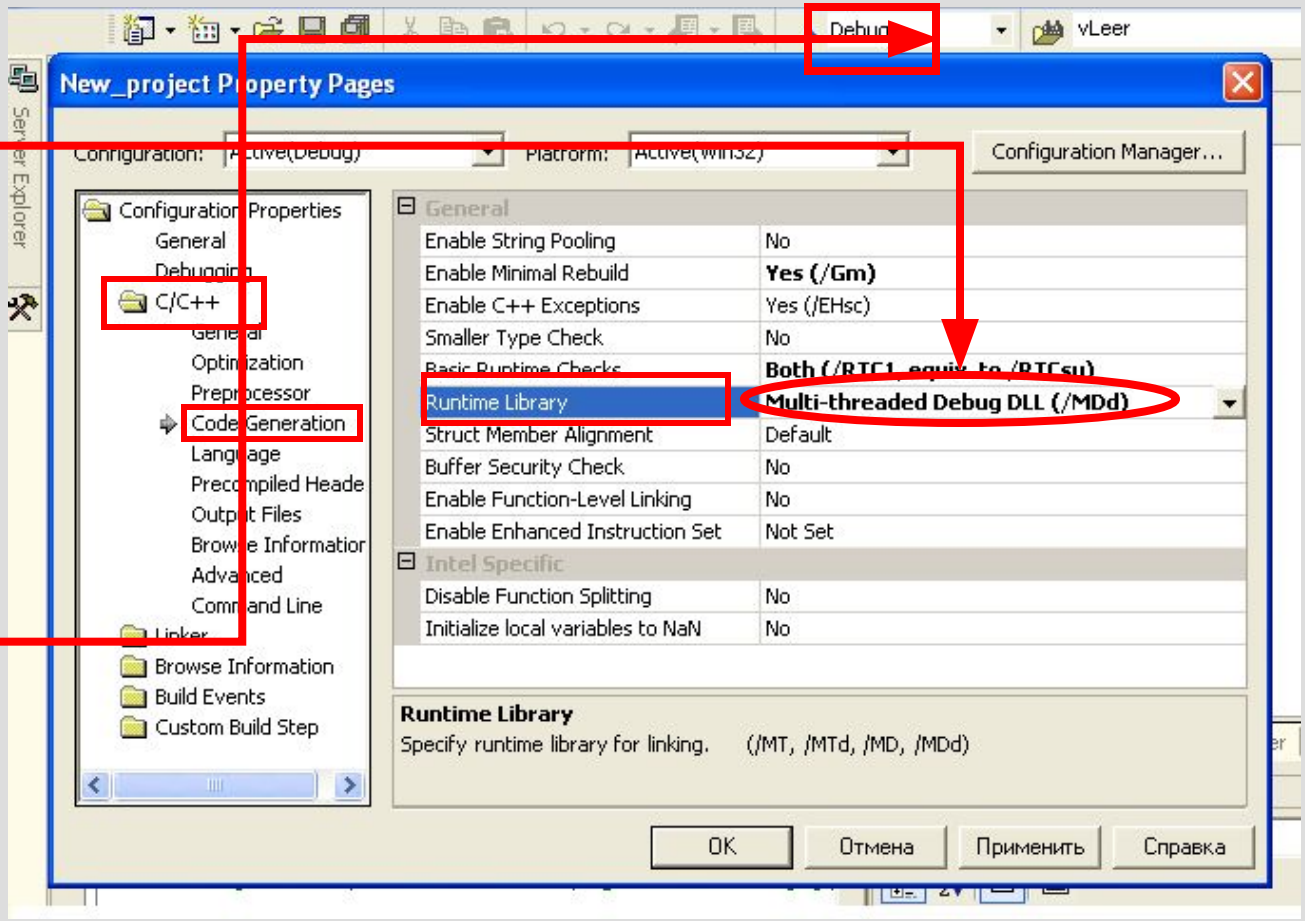
Установить поддержку директив OpenMP



2.4. Настройка на многопоточно-безопасные библиотеки

Выбрать многопоточно-безопасные библиотеки

При выборе многопоточных библиотек учитывать: «Debug» или «Release»



3. Intel Thread Checker (тестирование правильности выполнения многопоточного приложения)

3.1. Подготовка приложения для анализа Thread Checker: условия компиляции

3.2. Настройки Visual Studio.Net для компиляции приложения с целью анализа Thread Checker

3.3. Выполнение анализа Thread Checker при минимальном инструментировании приложения

Дополнительная информация о Thread Checker – в лекциях 4, 8, 9

3.1. Подготовка приложения для анализа Thread Checker: условия компиляции

Компиляция

- Используйте многопоточно - безопасные библиотеки (**/MD**, **/MDd**)
- Включите генерацию символьной информации (**/Zi**, **/ZI**, **/Z7**)
- Отключите оптимизацию (**/Od**)

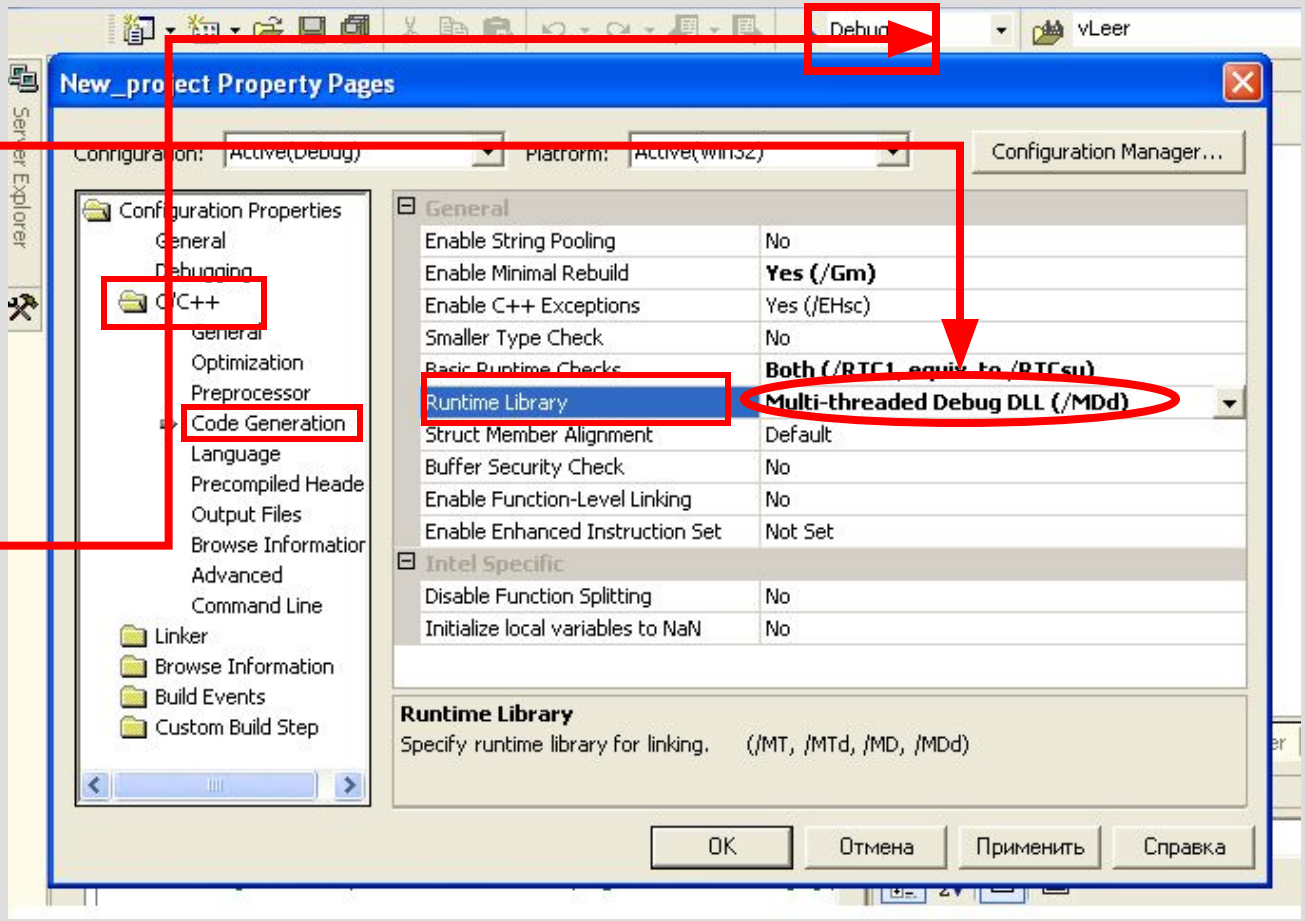
«Линкование» (Link)

- Сохранить символьную информацию (**/debug**)
- Specify relocatable code sections: **/fixed:no**)

3.2. Настройки Visual Studio.Net для компиляции приложения с целью анализа Thread Checker (слайд 1)

Выбрать многопоточно-безопасные библиотеки

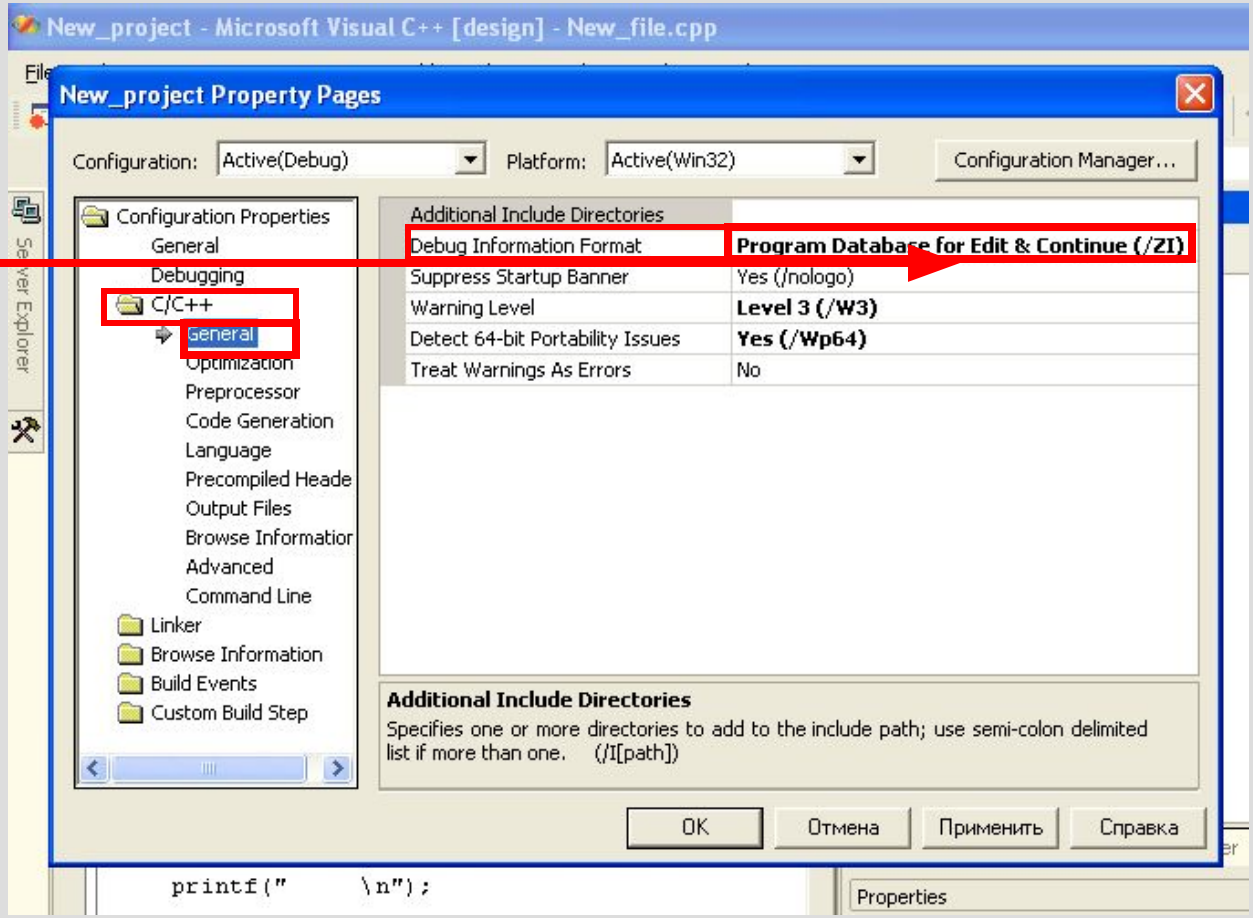
Конфигурация проекта - «Debug»



3.2. Настройки Visual Studio.Net для компиляции приложения с целью анализа Thread Checker (слайд 2)

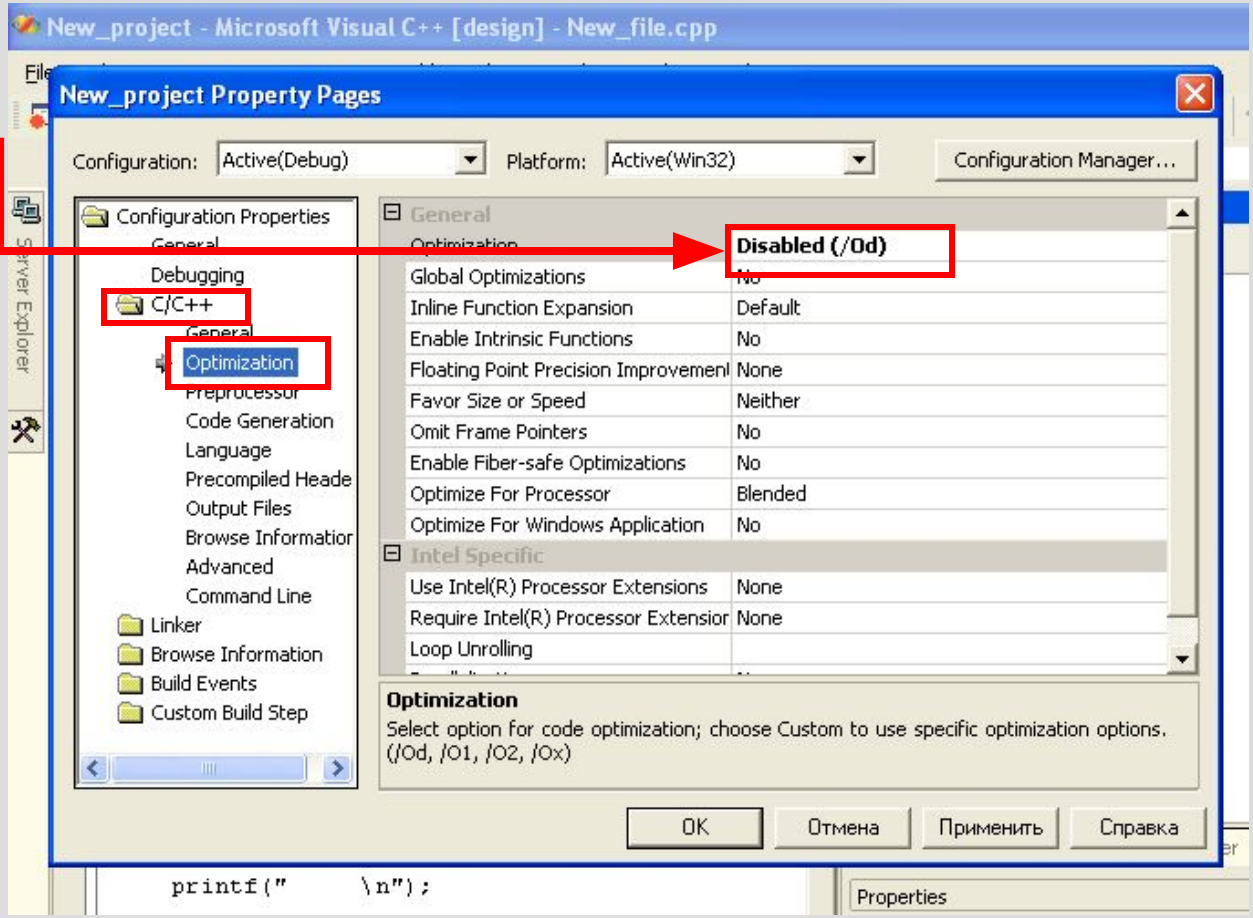
Убедитесь, что установлена генерация символьной информации

(**/Zi**, **/ZI**, **/ Z7**)



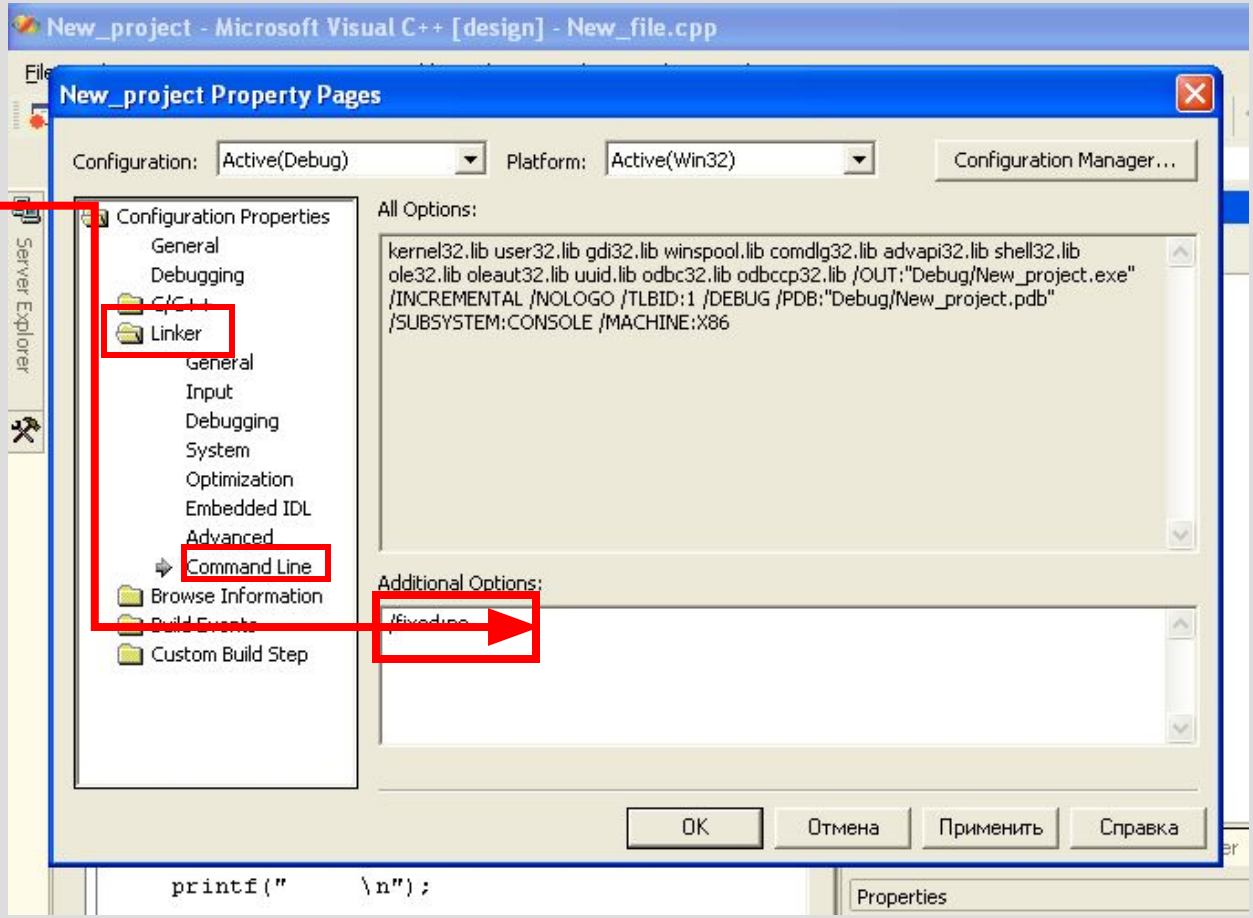
3.2. Настройки Visual Studio.Net для компиляции приложения с целью анализа Thread Checker (слайд 3)

Убедитесь, что отключена оптимизация



3.2. Настройки Visual Studio.Net для компиляции приложения с целью анализа Thread Checker (слайд 4)

Добавьте в командную строку «Linker» команду /fixed:no



3.3. Выполнение анализа Thread Checker при минимальном инструментировании приложения (слайд 1)

Запуск приложения

- Должен быть выполнен из-под Thread Checker
- Приложение инструментируется во время выполнения
- Также применяются внешние инструментированные динамические библиотеки (DLLs)

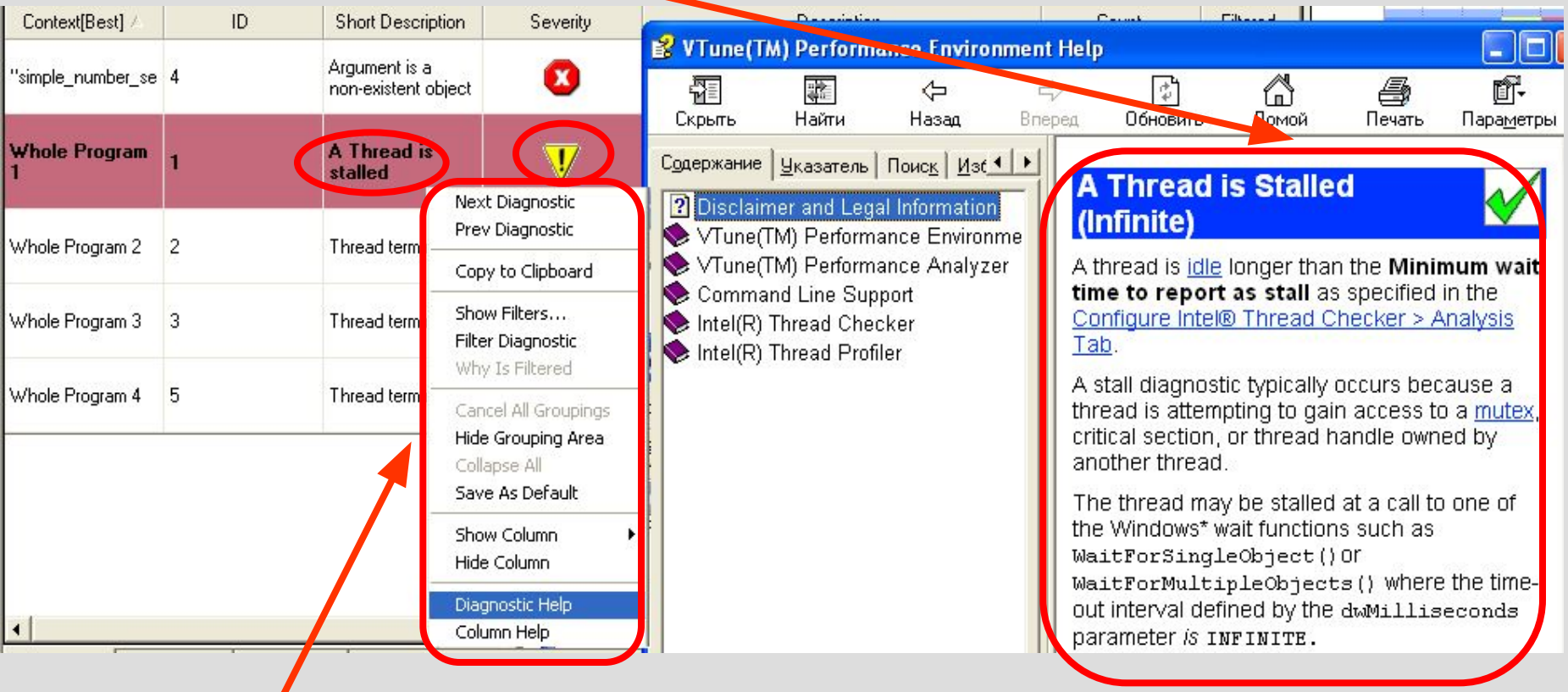
3.3. Выполнение анализа Thread Checker при минимальном инструментировании приложения (слайд 1)

Выполнение приложения

- Запуск в среде VTune™
- Запуск из-под командной строки Windows*
  - Полученные данные размещаются в файле результатов threadchecker.thr
  - Просмотр результатов (.thr file) в среде VTune

Помощь Thread Checker

Выявляет причины ошибок и предлагает способы их ликвидации. Предлагает на выбор набор API в качестве определяемых пользователем синхронизационных примитивов



Контекстное меню (щелчок правой кнопкой): выбор помощи по результатам диагностики (Diagnostic Help)

4. Intel Thread Profiler (тестирование производительности многопоточных вычислений)

4.1. Установки для выполнения минимального анализа с помощью Thread Profiler

4.1.1. Thread Profiler для Windows Threads

4.1.2. Thread Profiler для OpenMP

4.2. Немного о Thread Profiler

4.1. Установки для выполнения минимального анализа с помощью Thread Profiler

4.1.1. Thread Profiler для Windows Threads

Установки аналогичны случаю выполнения анализа с помощью Thread Checker

Дополнительная информация о Tread Profiler – в лекциях 4, 8, 9

4.1. Установки для выполнения минимального анализа с помощью Thread Profiler

4.1.2. Thread Profiler для OpenMP

- Установки аналогичны случаю выполнения анализа с помощью Thread Checker
- Дополнительная информация о Tread Profiler – в лекциях 4, 8, 9

Intel® Thread Profiler

- «Вставлен» в среду VTune™
- Сборка данных на основе работы приложения, «инструментированного» с помощью VTune
- Предназначен для анализа производительности OpenMP* приложений или многопоточных приложений с использованием потоков Win32* API и POSIX*
- Выявляет в многопоточном приложении «узкие места», которые непосредственно влияют на производительность

Как и что можно анализировать с помощью Intel® Thread Profiler

Поддерживает несколько различных компиляторов

- Компиляторы Intel® C++ и Fortran , v7 и выше
- Microsoft* Visual* C++, v6
- Microsoft* Visual* C++ .NET* 2002 & 2003 Editions
  - Интегрируется в среду Microsoft Visual Studio .NET*

Бинарное «инструментирование» приложений

Различные способы фильтрации данных и различные диаграммы для их представления с целью организации анализа

Анализ «*критического пути*» (*critical path)*