Тема 1-3.

Отладка

для АСУб и ЭВМб

Настройка среды разработки: Ссылки на внешние ресурсы в коде

1. On the **Tools** menu, click **Options**.
2. Click **Text Editor**.
3. To change the option for only one language, expand the folder for that language and choose **General**.

1. —or—
2. To change the option for all languages, expand the **All Languages** folder and choose **General**.
4. Under **Display**, select **Enable single-click URL navigation**.

Отладка программы

- ***Корректность*** – это способность программной системы работать в строгом соответствии со своей спецификацией.

- Понятие корректности программной системы имеет смысл только тогда, когда задана ее спецификация. В зависимости от того, как формализуется спецификация, уточняется понятие корректности.

- ***Отладка*** – процесс, направленный на достижение корректности.

- ***Отладка*** (debug, debugging) – **процесс** поиска, локализации и исправления ошибок в программе.

Отладка программы

- Часть ошибок программы ловится автоматически еще на этапе компиляции. Сюда относятся
  - все синтаксические ошибки,
  - ошибки несоответствия типов
  - некоторые другие.

- Это простые ошибки и их исправление, как правило, не вызывает трудностей.

- В отладке нуждается **синтаксически корректная** программа, результаты вычислений которой получены, но не соответствуют требуемым спецификациям.

- Чаще всего еще не отлаженная программа на одних исходных данных работает правильно, на других – дает ошибочный результат.

- **Искусство** отладки состоит в том, чтобы обнаружить все ситуации, в которых работа программы приводит к ошибочным вычислениям.

Классификация средств отладки

- Средства, позволяющие контролировать **ход** вычислительного **процесса**:
  - порядок следования операторов в функциях/методах,
  - порядок вызова самих функций/методов,
  - условия окончания циклов,
  - правильность переходов.

- Средства, позволяющие отслеживать изменение **состояния** вычислительного **процесса** (значения переменных) в процессе выполнения

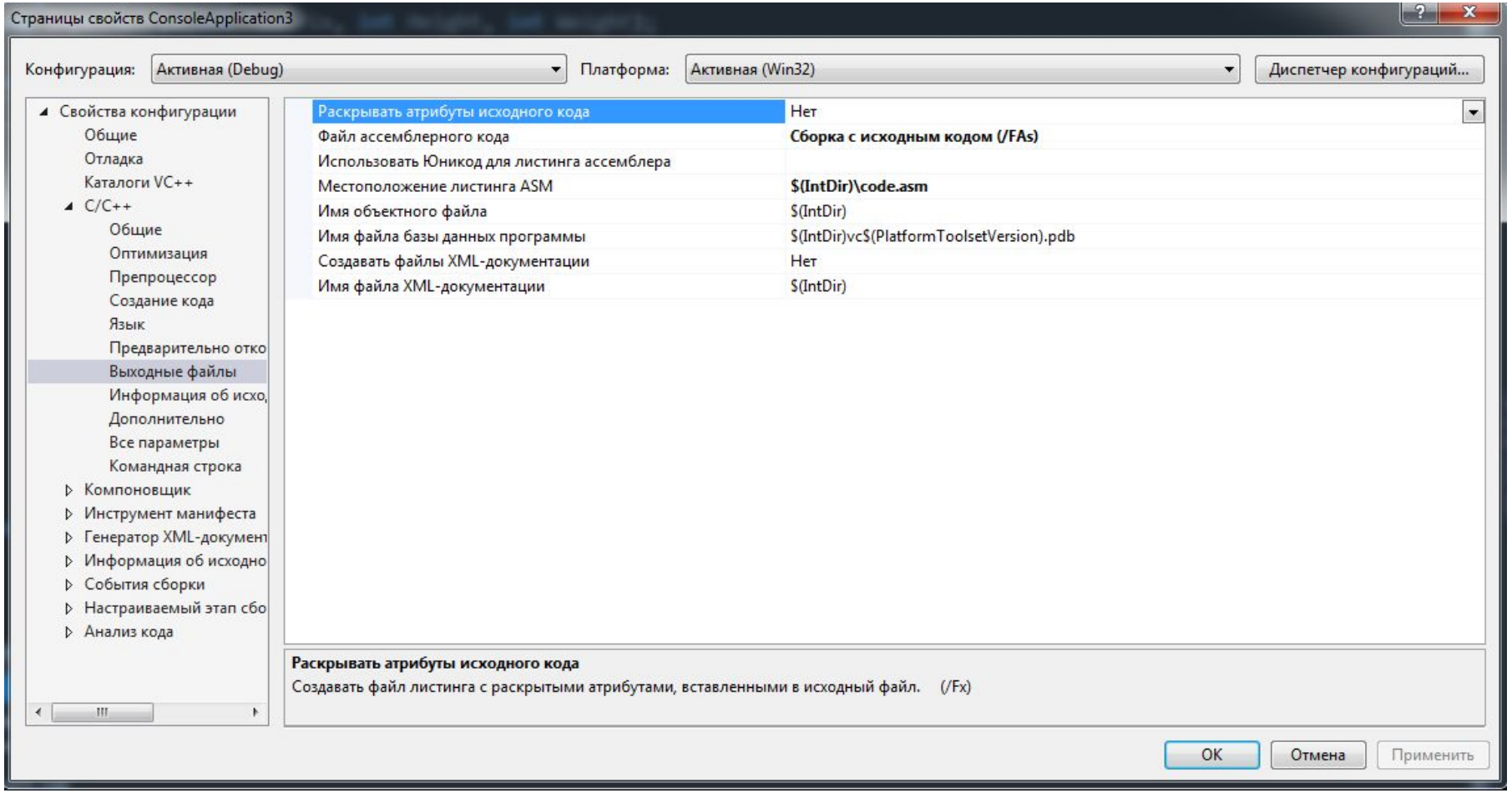
Классификация средств отладки

- Точки останова:
  - Способы создания
  - Безусловные и Условные
  - Список точек
- Дизассемблирование
- Состояния
- Иерархия вызовов

Дизассемблирование

- Поставьте точку останова на рассматриваемый код, и когда отладчик попадает в нее:
  - щелкните правой кнопкой мыши и найдите «Перейти к сборке» или «К дизассемблированному коду»;
  - нажмите CTRL + ALT + D или **ALT + G**
- Генерировать листинги сборки во время компиляции
  - настройки проекта (Проект->Свойства), далее «C / C ++» -> «Выходные файлы»
  - Выберите раздел «Файл ассемблированного кода» и какие-либо значения, содержащие «машинный код»
  - Выберите раздел «Местоположение листинга ASM» и введите имя файла.
  - В каталоге отладки после компиляции появится файл

Дизассемблирование



Отладка кода C++ на примере средства Visual Studio

- https://habr.com/post/102178/
- https://msdn.microsoft.com/ru-ru/library/sc65sadd.aspx
- **https://docs.microsoft.com/ru-ru/visualstudio/debugger/quickstart-debug-with-cplusplus?view=vs-2019**

Replit: сборка проекта

- Когда вы нажимаете кнопку **Выполнить** 

,

то система Replit пытается скомпилировать все файлы C++ в одну программу.

- Если существует несколько точек входа в программу, т.е. есть несколько функций **main** в разных (или одном) файлах, то будет ошибка сборки.
  - clang: error: linker command failed with exit code 1
  - multiple definition of `main’
- Для решения этой проблемы необходимо осуществить сборку через командную строку

Replit: сборка проекта в командной строке

- Пример

```
clang++ -pthread -std=c++17 -o Exe/lab5.exe
Lab5/Lab5.cpp Lab5/StructSaver.cpp
```

- **Здесь есть неизменяемая часть**

  ```
  clang++ -pthread -std=c++17 -o
  ```

- Далее после -o идёт **имя исполняемого файла** (может быть вместе с путём), который будет скомпилирован.
- В конце указывается список файлов (только cpp, не h), которые будут участвовать в текущей сборке.
- После компиляции исполняемый файл появится в списке

Replit: сборка проекта в командной строке

- Запустить программу можно также из командной строки
  - Если файл в корне, то команда будет такая: `./имя_файл`
  - Если нет, то команда будет такая: `./Путь к файлу/имя_файл`
- Советы для работы в командной строке

  Для ускорения ввода пути и имени используйте автозаполнения с помощью `Tab`

  Чтобы вернуться к предыдущим и следующим командам используйте клавиши «вверх» и «вниз»