Lab2

Этапы обработки:

1. Препроцессинг

2. Компилирование

3. Ассемблирование

4. Компоновка

5. Загрузка

В Visual Studio конфигурации Debug и Release представляют два различных режима сборки проекта.

Конфигурация Debug используется во время разработки приложения. В этом режиме проект компилируется с отладочной информацией, что позволяет использовать отладчик для пошагового выполнения кода, установки точек останова и просмотра значений переменных во время выполнения. Также в режиме Debug могут быть включены дополнительные проверки и предупреждения, чтобы облегчить отладку и обнаружение ошибок.

Конфигурация Release предназначена для окончательной сборки и выпуска приложения. В этом режиме проект компилируется без отладочной информации и с оптимизацией кода, чтобы достичь максимальной производительности приложения. В режиме Release также могут быть применены другие оптимизации, такие как удаление неиспользуемого кода или объединение файлов ресурсов, чтобы уменьшить размер и улучшить производительность приложения.

Отличие между конфигурациями Debug и Release в Visual Studio заключается в том, что Debug предоставляет дополнительные возможности для отладки и обнаружения ошибок, в то время как Release оптимизирует производительность и размер приложения для выпуска.

Файлы с расширением .exe образуются на этапе линковки проекта в Visual Studio. Линковщик объединяет все файлы .obj, а также статические и динамические библиотеки, которые используются в проекте, и создает исполняемый файл (.exe), который может быть запущен на компьютере.

Файлы с расширением .obj образуются на этапе компиляции проекта в Visual Studio. Они содержат объектный код, который представляет собой низкоуровневое представление исходного кода программы после его трансляции компилятором. Файлы .obj содержат инструкции процессора, данные и другую информацию, необходимую для создания исполняемого файла или библиотеки.

1) Общие: Вкладка "Общие" содержит основные настройки проекта, такие как имя проекта, расположение файлов проекта, тип проекта и целевую платформу.

2) Дополнительно: Вкладка "Дополнительно" содержит дополнительные настройки проекта, такие как настройки компилятора, определение макросов, настройки оптимизации и другие параметры компиляции.

3) Отладка: Вкладка "Отладка" содержит настройки для отладки проекта, такие как параметры запуска, настройки символов отладки, режимы работы отладчика и другие параметры отладки.

4) Каталоги VC++: Вкладка "Каталоги VC++" содержит пути к заголовочным файлам и библиотекам, используемым проектом. Здесь можно добавлять или изменять пути к необходимым зависимостям проекта.

5) С/C++: Вкладка "С/C++" содержит настройки компилятора C/C++, такие как опции компиляции, настройки предупреждений и другие параметры компиляции.

6) Компоновщик: Вкладка "Компоновщик" содержит настройки компоновщика, который объединяет объектные файлы в исполняемый файл или библиотеку. Здесь можно настроить параметры компоновки, включая опции компоновщика и ссылки на внешние библиотеки.

7) Инструмент манифеста: Вкладка "Инструмент манифеста" содержит настройки для создания и управления манифестом приложения. Манифест определяет зависимости и требования к запуску приложения.

8) Генератор XML-Документов: Вкладка "Генератор XML-Документов" содержит настройки для создания XML-документов в проекте. Здесь можно добавлять и настраивать генераторы XML-документов, которые могут использоваться в процессе сборки проекта.

9) Информация об исходном коде: Вкладка "Информация об исходном коде" содержит настройки для отображения информации об исходном коде во время отладки. Здесь можно настроить пути к файлам исходного кода, указать символические имена для отладки и другие параметры связанные с исходным кодом.

10) События сборки: Вкладка "События сборки" содержит настройки для выполнения пользовательских действий во время сборки проекта. Здесь можно настроить команды, которые будут выполняться перед или после сборки проекта.

11) Настраиваемый этап сборки: Вкладка "Настраиваемый этап сборки" содержит настройки для настраиваемых этапов сборки проекта. Здесь можно добавить пользовательские скрипты или команды, которые будут выполняться во время сборки проекта.

12) Code analysis: Вкладка "Code analysis" содержит настройки для статического анализа кода. Здесь можно настроить параметры анализа кода, такие как включение или отключение определенных правил анализа и настройки уровня предупреждений.

Lab3

1. Что такое таблица кодировки?

2. Расшифруйте аббревиатуру ASCII.

3. Поясните структуру кодировки Windows - 1251.

4. Что такое UNICODE?

5. Поясните структуру UNICODE.

6. Что такое UTF-8 и UTF-16?

7. Определите разницу значений кодов следующий символов UTF-16: F и f, S и s, L и l, Б и б, Г и г, Э и э. Каким способом можно получить из кода буквы нижнего регистра букву верхнего регистра?

1. Таблица кодировки - это таблица, которая содержит соответствие между символами или значениями в одной системе счисления и их кодами или представлением в другой системе счисления.

2. ASCII расшифровывается как American Standard Code for Information Interchange (Американский стандартный код для обмена информацией).

3. Структура кодировки Windows-1251 основана на ASCII, что означает, что первые 128 символов (от 0 до 127) идентичны ASCII. Оставшиеся 128 символов (от 128 до 255) в кодировке Windows-1251 содержат дополнительные символы, которые не были представлены в ASCII. В этой части кодировки содержатся символы русского алфавита, а также специальные символы и знаки препинания, используемые в русском и других славянских языках. Windows-1251 является однобайтовой кодировкой, что означает, что каждый символ представлен одним байтом. В кодировке Windows-1251 всего 256 возможных значений байта, поэтому она может представить только ограниченный набор символов. Это также означает, что кодировка Windows-1251 несовместима с многобайтовыми кодировками, такими как UTF-8.

4. UNICODE - это стандарт кодирования символов, который предоставляет уникальный числовой код для каждого символа, используемого в практически всех письменных языках мира.

5. UNICODE организован в виде плоскостей, каждая из которых содержит набор кодовых точек для определенного набора символов. Каждая плоскость в UNICODE имеет фиксированное количество кодовых точек. Кодовая точка в UNICODE обозначается шестнадцатеричным числом, например, U+0041 для символа "A".

6. UTF-8 (Unicode Transformation Format - 8-bit) - это переменная длина кодировка, которая использует от 1 до 4 байтов для представления символов. В UTF-8 символы из основной многоязычной плоскости (BMP) кодируются одним байтом, а символы из дополнительных плоскостей кодируются последовательностью нескольких байтов.

UTF-16 (Unicode Transformation Format - 16-bit) - это фиксированная длина кодировка, которая использует 2 байта для представления символов. UTF-16 может представлять символы из любой плоскости UNICODE, включая как основную многоязычную плоскость (BMP), так и дополнительные плоскости.

7. Разница значений кодов для символов UTF-16 между верхним и нижним регистром составляет 20.

Для получения буквы верхнего регистра из кода буквы нижнего регистра нужно увеличить значение кода буквы нижнего регистра на 20.

№3.

Минимальная единица хранения информации - байт.

Минимальная адресуемая единица хранения информации - сектор диска.

№4.

Бит может принимать только два взаимоисключающих значения: «да» или «нет», «1» или «0», «включено» или «выключено», и тд.

№5.

В одном байте содержится 8 бит.

№6.

дс. дв. ш.

0 00000000 00

1 00000001 01

2 00000010 02

3 00000011 03

4 00000100 04

5 00000101 05

6 00000110 06

7 00000111 07

8 00001000 08

9 00001001 09

10 00001010 A

11 00001011 B

12 00001100 C

13 00001101 D

14 00001110 E

15 00001111 F

16 00010000 10

17 00010001 11

18 00010010 12

19 00010011 13

20 00010100 14