Министерство образования Республики Беларусь

УЧРЕЖДЕНИЕ ОБРАЗОВАНИЯ
«БРЕСТСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»

Кафедра интеллектуальных информационных технологий

РАБОТА В ИНТЕГРИРОВАННОЙ СРЕДЕ BORLAND C++

Методические указания

к лабораторным работам по дисциплине
"Конструирование программ и языки программирования"
для студентов
специальности "Искусственный интеллект"

БРЕСТ 2008

УДК 681.3 (075.8)
ББК с57

Методические указания предназначены для студентов, изучающих язык С в системе программирования Borland C++. В работе даются рекомендации по овладению практическими навыками настройки и использования системы для разработки программ. Материал служит базой для проведения лабораторных работ по курсу " Конструирование программ и языки программирования".

Составители: Г.Л. Муравьев, доцент, к.т.н., С.В. Мухов, доцент, к.т.н., В.Н. Шуть, доцент, к.т.н.

Рецензент: доцент кафедры математического моделирования Брестского государственного университета им. А.С. Пушкина, к.т.н., доцент Пролиско Е.Е.

ТЕМА 1. УСТАНОВКА BORLAND C++

1. Справочная информация

1.1. Дистрибутив Borland C++

Здесь рассматриваются инструментальные средства распространенной интегрированной среды разработки программ (системы программирования) Borland C++ 3.1, которая позволяет изучать и использовать как классический С, так и расширение языка в части поддержки прогрессивной технологии объектно-ориентированного программирования C++.

Система программирования Borland C++ предназначена для использования на IBM-совместимых ПЭВМ. Borland C++ в целом соответствует стандарту ANSI (Американского института национальных стандартов) и включает в себя как подмножество реализацию классического С в версии K&R (Кернигана и Ритчи).

Поставка (дистрибутив) системы включает в себя компилятор Borland C++, а также набор сервисных программ (утилит), заголовочных файлов (header) и библиотеки функций.

Компилятор Borland C++ представлен в диалоговом варианте как интегрированная программная среда (запускается, например, как файл bc.exe) и пакетном как автономный компилятор, управляемый командной строкой (запускается, например, как файл bcc.exe).

Утилиты, управляемые командной строкой, включают редактор связей (компоновщик, "линкер") tlink.exe, препроцессор cpp.exe, обработчик проекта make.exe, являющийся более мощной версией Project-Make, которая перед компоновкой и связыванием объектных файлов перекомпилирует файлы, требующие обновления, и объединяет их с теми, которые не требуют повторной компиляции.

Заголовочные файлы служат для подключения библиотек, реализующих функции: alloc.h - управления памятью; bios.h - работы с базовой системой ввода-вывода; conio.h - функции консольного ввода-вывода; ctype.h – подключает макросы классификации символов; dir.h - функции работы с каталогами; dos.h - функции поддержки интерфейса с MS-DOS и с процессором 8086; float.h - подключает функции работы с плавающей точкой; math.h подключает математические функции; mem.h функции манипулирования памятью; stdlib.h - "стандартные" процедуры; stdio.h - функции стандартного ввода-вывода; string.h - функции работы со строковыми данными.

Borland C++ включает: - библиотеки работы с сопроцессором плавающей точки FP87.LIB либо программный эмулятор EMU.LIB при отсутствии сопроцессора; библиотеки математических процедур MATHx.LIB, библиотеки процедур Cx.LIB и стартовые файлы загрузчика C0x.OBJ для различных моделей памяти ЭВМ. Символ x принимает значения из множества { T, S, C, M, L, H }, что соответствует первой букве используемой в системе программирования модели памяти, в том числе – минимальной (tiny), малой (small), компактной (compact), средней (medium), большой (large) или максимальной (huge).

1.2. Установка Borland C++

Для установки системы программирования (пакета Borland C++) надо запустить на выполнение с установочного диска (дистрибутива) программу INSTALL.EXE. При этом дистрибутив может быть скопирован предварительно на жесткий диск компьютера или сетевой диск.

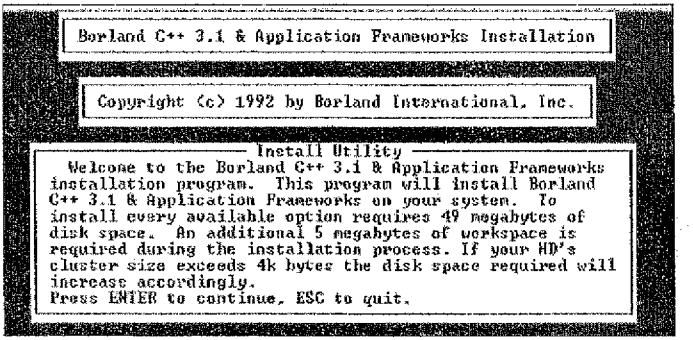

Рис. 1. Установка Borland C++ 3.1

При установке необходимо:

- указать диск с дистрибутивом Borland C (рисунок 2);

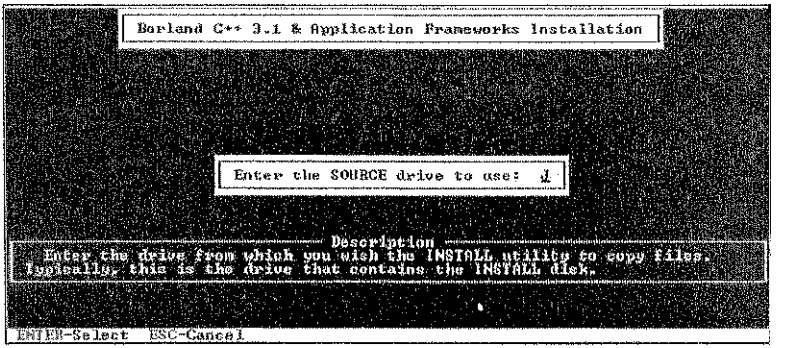

Рис. 2. Установка Borland C++ 3.1. Указание диска с дистрибутивом

-указать местонахождение дистрибутива Borland C на диске, т.е. указать путь доступа к папке (рисунок 2);

-определить параметры установки пакета, т.е. настроить процедуру установки (рисунок 3);

-запустить установку пакета.

Параметры установки пакета включают:

1. Directories - каталог (папка, директория), в который будет устанавливаться пакет.

2. Windows Dir - каталог с установленным Windows, где находится файл win.ini для выполнения полной установки Borland C++.

3. Install Options - опции (режимы) установки, в том числе:

а) Command Line – установка необходимых файлов для работы компилятора командной строки;

б) IDE – установка необходимых файлов для работы интегрированной среды разработки (IDE);

в) Turbo Debugger – предусматривает установку необходимых файлов для работы отладчика программ;

г) Turbo Assembler – предусматривает установку необходимых файлов для работы Турбо Ассемблера;

д) Turbo Profiler – установка необходимых файлов для работы Turbo Profiler;

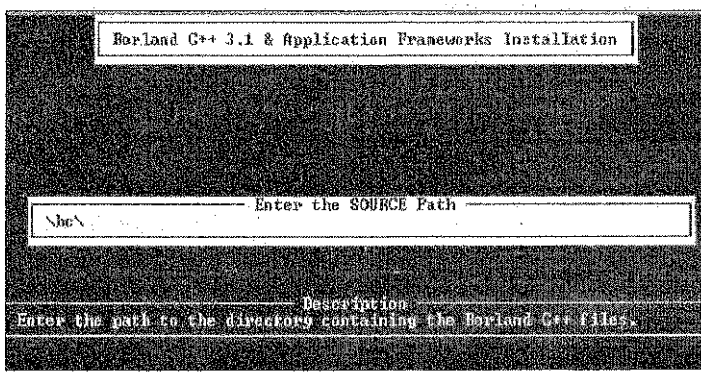

Рис. 3. Установка Borland C++ 3.1. Указание места дистрибутива на диске

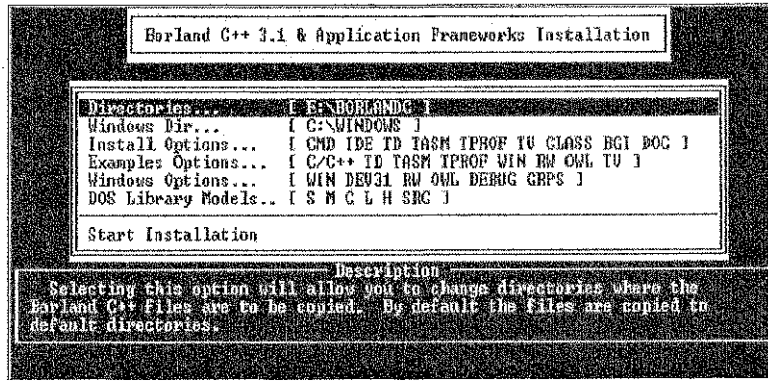

Рис. 4. Установка Borland C++ 3.1. Настройка процедуры установки пакета

е) Turbo Vision – установка необходимых файлов для работы Turbo Vision;

ж) Class Libraries – установка необходимых файлов для поддержки библиотек классов, включая источники, заголовочные файлы, библиотеку и файлы с примерами;

з) BGI Library – установка необходимых файлов для поддержки работы в графическом интерфейсе, включая драйверы графики и шрифтов;

и) Documentation – установка документации, необходимой для работы с C++, отладчиком, Турбо Ассемблером и другими компонентами системы.

4. Example Options – позволяет определить, какие файлы с примерами устанавливать на жёсткий диск.

5. Windows Options – определяет конфигурацию установки Borland C++ для работы под управлением Windows, в том числе:

а) Windows Capability – разрешает установку файлов Borland C++, поддерживающих работу с Windows;

б) Windows 3.1 – разрешает установку файлов Borland C++, поддерживающих работу с Windows 3.1;

в) Resource Workshop – определяет установку файлов, требуемых для работы мастера ресурсов и т.п.

6. Dos Library Models – определяет устанавливаемые модели памяти, каждая из которых требует около 300 Кб свободного места на диске, в том числе: - small/tiny минимальную-малую; - compact компактную; - medium среднюю; - large большую; - huge максимальную.

По умолчанию в инструментальную среду разработки включены все компоненты пакета.

1.3. Запуск интегрированной среды Borland C++

Для запуска интегрированной среды Borland C++ надо ввести в командной строке путь доступа к соответствующему файлу. Например, для запуска файла интегрированной среды BC.exe, находящегося в папке E:\BORLANDC

E:\BORLANDC \BIN\BC

или запустить файл BC.exe с помощью проводника.

1.4. Настройка Borland C++ с помощью опций главного меню

При запуске Borland C++ появляется основное меню. Для дальнейшей работы необходимо настроить среду согласно специфике использования пакета, для чего используются опции меню и файл конфигурации tcconfig.tc. Для получения подсказок применяется клавиша F1.

В файле конфигурации, создаваемом автоматически при включении опции Options-Save, запоминается информация о настройке Borland C++ (опции компилятора, компоновщика, среды, проекта и специфические опции, связанные только с интегрированной средой). При очередном пуске среды опции главного меню принимают ранее выбранные значения.

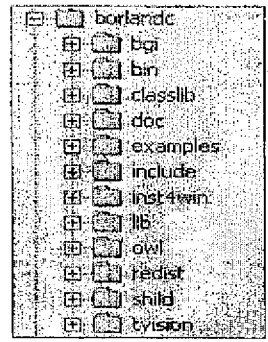

Рис. 5. Дерево каталогов Borland C++ 3.1

При запуске системы Borland C++ файл конфигурации ищется в текущем, а затем в каталоге Borland C++. Если он не найден, то устанавливаются режимы по умолчанию.

Настройка интегрированной среды выполняется в первую очередь заданием функций (опций) пункта главного меню Options. Основные функции, группы функций описаны ниже.

Пункт **Application** (Приложение) открывает диалоговое окна Set Application Options, где могут быть выбраны начальные установки параметров, определяющих режимы трансляции программ (для ОС Windows и ОС Dos). Выбор режима производится из набора Standard, Overlay или Library.

Пункт **Compiler** (Компилятор) служит для выбора режимов, касающихся трансляции исходного кода программы. Настройка компилятора предполагает установку следующих основных параметров.

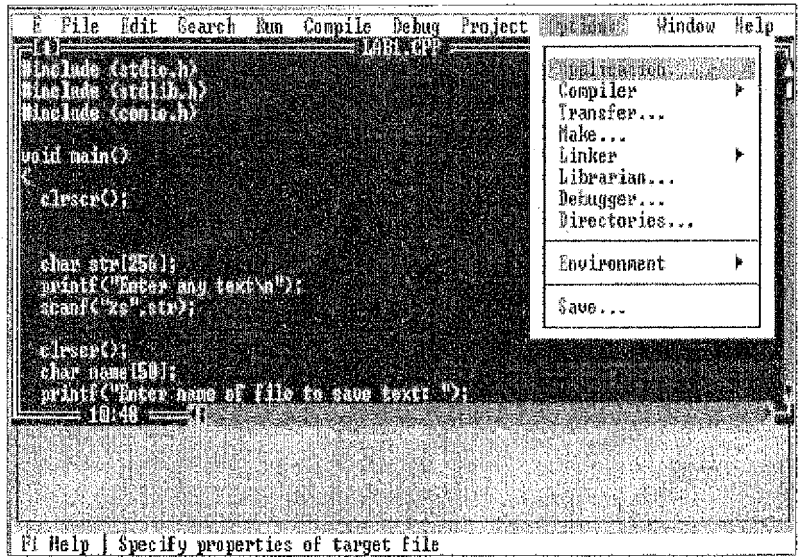

Рис. 6. Пункт меню Options

1. Code generation – задание параметров генерируемого объектного кода программы. Параметры задаются в диалоговом окне Code Generation, где определяется используемая модель памяти, определяющая способ использования памяти компьютера, характер размещения программы и данных.

Параметр Model задает выбранную модель памяти. При этом выбор производится из вариантов моделей памяти, представленных в таблице 1.

Модель памяти Tiny (крошечная) – используется для приложений небольшого размера. Здесь все сегменты программы (CS, DS, ES, SS), используемые для хранения кода, данных и стека используют один и тот же сегмент памяти в 64 K. Применяются соответственно только указатели типа NEAR. Программы такого типа с моделью памяти Tiny могут быть преобразованы в COM-файлы.

Модель памяти Small (малая) – используется для приложений среднего размера. Сегмент кода и сегмент данных разделены. Каждый использует отдельный сегмент в 64 K. Стек имитируется на сегменте данных. Применяются соответственно только указатели типа NEAR.

Модель памяти Medium (средняя) – является наиболее подходящей моделью для программ с большим кодом, не использующих больших объемов данных, размещаемых в оперативной памяти. Для размещения данных и стека выделяется один сегмент в 64 K, а для размещения кода программы используется память объемом до 1M. Здесь применяются FAR указатели для кода, NEAR – для данных.

Модель памяти Compact (компактная) – используется для программ компактного размера, обрабатывающих большие объёмы данных, размещаемых в оперативной памяти. Для размещения динамических данных и организации стека выделяется памятью в 1 M, код программы размещается в отдельном сегменте в 64 K. Модель памяти Compact является противоположностью модели типа Medium. Здесь указатели типа FAR использу-

ются для работы с данными, указатели типа NEAR используются для работы с кодом программы.

Модель памяти Large (большая) – используется для приложений как с большим программным кодом, так и с большим количеством оперативно обрабатываемых данных. Для размещения программы, динамических данных и стека выделяется память объемом до 1М. Здесь применяются FAR указатели как при работе с кодом, так и при работе с данными.

Модель памяти Huge (огромная) – используется для больших приложений. В отличие от модели памяти Large и других моделей памяти позволяет использовать для размещения статичных данных память объёмом более 64 К. При использовании этой модели применяют FAR указатели и для кода и для данных.

Таблица 1. Модели памяти

Содержимое памяти	Типы памяти: Tiny	Small	Compact	Medium	Large	Huge	Адреса
Векторы прерываний, DOS	~ К						
Прикладная программа: Коды программы (CS)	CS,DS,SS,ES, 64K	CS,64K "near"	CS,1M "far"	CS,64K "near"	CS,1M "far"	CS,1M "far"	младшие
Данные: -статические (инициализир., неинициал., конст.) (DS)	"near"	DS,SS,ES, 64K "near"	DS,SS ES, 64K "near"	DS,64K "far"	DS,64K "far"	DS,1M "huge"	
-динамические ("куча") (ES)	v SP-->	v SP-->	v SP-->	SS,64K SP-->	SS,64K SP-->	SS,64K SP-->	
-локальные и аргументы функций, стек (SS,SP)	^	^	^	^	^	^	v
-динамические (heap, удаленная "куча") (ES)	не используется			ES,1M	ES,1M	ES,1M	старшие
Видео-память	~ 112 К						

Параметр Options задает варианты создания объектного кода.

Параметр Assume SS Equals DS задает варианты интерпретации компилятором стекового сегмента. Здесь значение Default for Memory Model означает, что компилятор в соответствии с моделью памяти определяет, равен ли размер сегмента стека размеру сегмента данных. Значение Never означает, что компилятор принимает, что размер сегмента стека не равен размеру сегмента данных. Значение Always означает, что компилятор принимает, что размер сегмента стека равен размеру сегмента данных.

Режим Defines служит для задания макроопределений (директив), обрабатываемых программой препроцессором.

2. Advanced code generation – служит для установки дополнительных опций, влияющих на процесс создания кода. В том числе:

- опция Floating Point определяет, как Borland C++ обрабатывает числа с плавающей запятой. Для отмены обработки устанавливается значение None. Значение Emulation устанавливается, чтобы Borland C++ автоматически определял наличие сопроцессора обработки данных в формате с плавающей точкой и использовал его. В противном случае система будет эмулировать его команды подключением соответствующих библиотек;

- опция Instuction set настраивает компилятор на генерацию команд процессора Intel 8088/8086 или 80186/80286, включая расширенный набор команд 80186;

- дополнительные настройки, собранные в Options, позволяют уточнить особенности процесса создания загрузочного кода. Например, настройка Line number включает режим нумерации строк объектного файла для дальнейшего использования отладчиком и т.д.

3. Entry/Exit code – служит для конкретизации типа создаваемого приложения.

В том числе настройка Prolog/Epilog Code Generation задает создание приложений с ориентацией на ОС Windows или Dos, с использованием или без использования динамически подключаемых библиотек DLL.

Настройка Calling Convention Выбор определяет правила генерации команд вызова функций, отличающихся способом использования стека. Это стили вызова процедур, включая C-стиль и Pascal-стиль.

Настройка Stack Options определяет работу стека. Значение Standart stack frame генерирует стандартные коды событий вызова функций, что при отладке с помощью отладчика упрощает просмотр хранящихся в стеке вызванных процедур. Значение Test stack overflow проверяет переполнение стека во время выполнения программы.

4. C++ options – служит для дополнительной настройки компилятора для случая создания объектного кода приложений, написанных с использованием средств языка C++.

Настройка Use C++ Compiler определяет, компилирует Borland C++ программу как код C или как код C++.

Настройка Template Generation определяет, как Borland C++ создает шаблоны, используемые в C++.

Пункт Options позволяет задать дополнительные настройки.

Настройка C++ Virtual Tables определяет настройки контроля виртуальных таблиц в C++.

5. Advanced C++ options - служит для задания дополнительных режимов процесса генерации объектного кода приложения, не вошедших в предыдущие пункты, в случае использования языка C++.

6. Optimizations (Optimization Options) – служит для установки параметров, которые определяют режим работы компилятора с точки зрения оптимизации загрузочного кода.

Настройка Optimize For задает критерии оптимизации программы. Параметр Size вызывает минимизацию размера модуля, параметр Speed максимизацию скорости выполнения программы.

Настройка Register Variables запрещает или разрешает использовать регистровые переменные.

Настройка Optimizations: оптимизирует использование регистров, исключая избыточные загрузочные операции, запоминая содержимое регистров и вторично используя их при возможности; параметр Jump optimization уменьшает размер программы за счет удаления лишних передач управлений, goto, реорганизации циклов и операторов switch и др.

7. Source (Source Options) – служит для установки режимов работы с исходным текстом и определяет ограничения к его стандарту.

Настройка Keywords ограничивает круг используемых ключевых слов: ANSI ограничивает круг ключевых слов стандартом ANSI, игнорируя специфические слова Borland C++ как near, far, huge, asm, cdecl, pascal, interrupt, _es, _ds, _cs, _ss, регистровые псевдопеременные (_AX, _BX,...); - Borland C++ использует ключевые слова в полном объеме. Кроме этого используются наборы UNIX и Kernigan and Ritchie.

Настройка Source options устанавливает число значащих символов для идентификаторов (параметр Identifier Length - от 1 до 32 символов) и разрешает или запрещает использование вложенных комментариев (параметр Nested comments).

8. Messages – служит для настройки режимов обработки сообщений об ошибках.

Настройка Display устанавливает реакцию компилятора на ошибки. Параметр Display warnings задает режимы вывода предупреждений (All – вывод всех сообщений, Selected – вывод только определенных сообщений, None – без вывода).

Параметр Errors: stop after ... вызывает прекращение компиляции после выявления заданного числа ошибок (25 по умолчанию).

Параметр Warnings: stop after вызывает прекращение компиляции после заданного числа предупреждений (100 по умолчанию).

Настройка ANSI violations вызывает вывод на экран предупреждений о нарушении стандарта ANSI. Настройка C++ warnings управляет выводом на экран предупреждений о специфических нарушениях C++. Настройка "Frequent errors" вызывает вывод на экран предупреждений о распространенных ошибках, а "Less frequent errors " о менее распространенных ошибках.

9. Names – служит для изменения заданных по умолчанию имен сегментов, групп и классов для Code, Data и BSS секций.

Пункт **Make** (Управление Проектами) служит для установки режимов обработки проектов приложений.

Настройка Break Make On определяет условия прерывания обработки проекта в зависимости от характера возникших ошибок.

Настройка After Compiling определяет виды дальнейшей обработки проекта после завершения компиляции модулей исходного текста.

Настройка Generate Import Library предопределяет использование DEF или DLL файлов.

Пункт **Linker** (Компоновщик).

1. Settings - служит для задания режимов компоновки проектов.

Настройка Map File определяет генерацию таблиц распределения памяти файлов, в том числе таблицы "Segments" (для сегментов памяти), "Publics" (для глобальных переменных) или "Detailed" (для детального отчета).

Настройка Output определяет тип создаваемого приложения. Это может быть стандартное (классическое) DOS-приложение - параметр Standard DOS EXE, оверлейное DOS-приложение - параметр Overlaid DOS EXE, Windows -приложение - параметр Windows EXE, Windows –приложение с подключением динамических библиотек DLL - параметр Windows DLL.

Дополнительные настройки (группа Options) описаны ниже.

Параметр Default libraries заставляет компоновщик отыскивать любую неопределенную функцию (результат работы отличных от Borland C++ компиляторов) в их библиотеках.

Параметр Initialize segments предписывает компоновщику инициализировать неинициализированные сегменты памяти.

Параметр Warn dublicate symbols предупреждает о повторе идентификаторов.

Параметр Stack warning блокирует генерацию компоновщиком сообщения "No stack specified" (стек не специфицирован).

Параметр Case-sensitive link предопределяет выполнение компоновки модулей приложения с учетом выбранного регистра клавиатуры.

2 Libraries - служит для управления поведением компоновщика в отношении неинициализированных сегментов и библиотек. Так настройки Libraries управляют подключением системы Turbo Vision, библиотек графики Graphics library; настройки Container Class Library – управляют подключением библиотек контейнерных классов – Static или Dynamic.

Пункт **Librarian** служит для задания параметров, определяющих размер страниц в библиотеках (размер в байтах). Пункт Debugger (Отладчик) служит для задания параметров интегрированной отладки, определяет, какая информация об отладке должна быть включена в исполняемый файл.

Пункт **Directories** (Каталоги, папки) служит для указания местонахождения папок с основными типами файлов.

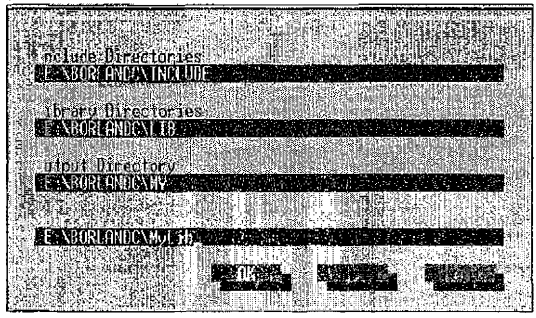

Рис 7. Меню Directories

Настройка Include Directories описывает через символ ";" названия каталогов, содержащих стандартные включаемые h-файлы.

Настройка Library Directories определяет каталоги, содержащие стартовые объектные файлы BC++ библиотечные программы, подключаемые во время компилирования программы.

Настройка Output Directory определяет каталоги, где хранятся созданные из исходных текстов программ командами Run и Make файлы типа *.OBJ, *.EXE и *.MAP.

Настройка Source Directories определяет каталоги, где интегрированный отладчик ищет исходные тексты к библиотекам, не принадлежащим открытому проекту.

Пункт **Environment** (Среда) служит для настройки пользовательского интерфейса системы программирования.

Настройка Preference служит для установки параметров, управляющих видом интерфейса. Параметр Screen Size определяет размер экрана в строках и столбцах. Параметр auto save задает автоматическое сохранение измененного файла конфигурации, проекта и т.п.

Настройка Editor служит для управления текстовым редактором. Например, параметр Tab size определяет размер табуляции (2-16). Соответственно при включенном режиме и нажатии клавиши Tab редактор вставляет в файл символ табуляции, а курсор проскакивает к месту следующего останова табуляции.

Настройка Mouse задает параметры работы манипулятора "мышь". В том числе, настраивается скорость срабатывания нажатий кнопок, определяются действия, выполняемые правой клавишей "мыши" и т.д.

Настройка Desktop служит для установки параметров сохранения: списков хронологии, содержания буфера обмена и т.п.

Настройка Startup служит для определения параметров запуска интегрированной среды.

Настройка Colors служит для уточнения и изменения цветового дизайна пользовательского интерфейса интегрированной среды, для задания цветов отдельных элементов текста.

Пункт **Save** (Сохранить) служит для выполнения принудительного сохранения измененных настроек системы программирования. Соответственно сохранение включает: - сохранение настроек среды программирования Environment в файле конфигурации tcconfig.tc; - сохранение настроек Desktop среды программирования Environment в файле prjname.dsk; - сохранение настроек проекта Project в файле *.prj.

Таким образом, для минимальной настройки среды следует:

- войти в меню Options и выбрать в нём пункт Directories. В появившемся окне задать: каталог Include Directories, содержащий включаемые стандартные файлы системы с расширением .h; каталог Library Directories, содержащий библиотечные и загружаемые файлы системы; каталог вывода Output Directory, в котором сохраняются созданные компилятором и компоновщиком .OBJ, .EXE и .MAP файлы;
- сохранить выбранные режимы в файле конфигурации, выполнив пункт Save меню Options.

Например (см. рисунки 5 и 7),

Include Directories - B:\BORLANDC\INCLUDE
Output Directory - B:\BORLANDC \ MY
Library Directories - B:\BORLANDC \LIB .

1.5. Запуск и настройка пакетного компилятора Borland C++

Режим работы пакетного (командного) компилятора BCC определяется опциями командной строки (могут отсутствовать) и опциями файла конфигурации TURBOC.CFG, аналогичными опциям главного меню среды.

При вызове командного компилятора выполняется поиск конфигуратора TURBOC.CFG и настройка компилятора на заданные в нем опции. Конфигуратор ищется сначала в текущем каталоге, а при его отсутствии там – в системном каталоге. При необходимости опции конфигуратора могут быть переопределены из командной строки (см. тему 4), имеющей более высокий приоритет по отношению к соответствующим опциям файла конфигурации.

В отличие от настройки интегрированной среды через главное меню и соответственно автоматического формирования файла конфигурации tcconfig.tc, кофигуратор TURBOC.CFG формируется вручную в рабочем разделе копированием из системного каталога с последующей правкой любым стандартным текстовым редактором ASCII (редактором ASCII).

Из списка опций TURBOC.CFG следует выделить опции типа -I. Указанные опции -I дополняются в командной строке справа, а оставшиеся опции TURBOC.CFG вставляются слева в список опций командной строки после команды BCC. Опции разделяются пробелами или размещаются каждая в своей строке. Например,

```
-LE:/BorlandC/LIB
-IE:/BorlandC/INCLUDE
-nF:/BorlandCWORK .
```

Для вызова командного компилятора BCC используется команда формата

BCC [опция...опция] имя_файла ... имя_файла,

где опции задают режим работы компилятора, а имена - список обрабатываемых файлов. Пример вызова командного компилятора приведен ниже

```
F:/BCWORK>E:/BorlandC/BCC <имя_файла>
F:/BCWORK>E:/BorlandC/BCC -LC:/LIB1 P1.CPP P2.CPP P3.ASM P4.LIB .
```

Здесь библиотеки подключаются из каталога C:/LIB1, компилируются файлы P1,P2, ассемблируется P3, при компоновке подключается библиотечный файл P4, загрузочный файл именуется P1.

1.6. Настройка Borland C++ с помощью утилиты TDINST

Утилита TDINST позволяет изменять, определять размеры экрана, окон редактирования и сообщений, режимы редактирования, цвета меню, каталоги включаемых, библиотечных, конфигурационных и других файлов и применяется для создания копий Borland C++, отличающихся дизайном и настройками. Все, указанные с помощью TDINST параметры, используются при запуске Borland C++ по умолчанию.

Запуск TDINST выполняется в формате:

TDINST [/C] [полное_имя_настраиваемого_компилятора] ,

где опция /C позволяет использовать цветной режим монитора. Если второй параметр не задан, то BC.EXE ищется в текущем каталоге. Примеры запуска приведены ниже:

```
TDINST BC.EXE
TDINST /C C:\P1\P2.EXE .
```

При запуске TDINST высвечивается основное меню генерации с соответствующими пунктами, описанными ниже.

Пункт Colors служит для установки цвета экранного объекта с помощью смотрового окна и эталонной палитры (параметр Customize предполагает установку цветов по выбору пользователя; параметр Default color set задает установку цветов по умолчанию);

Пункт Displays служит для задания видеорежима работы дисплея;

Пункт Options служит для настройки опций среды системы программирования. Подпункт Directories служит для установки используемых по умолчанию каталогов Borland C++ (включает в том числе параметры Include directories, Library directories, Output directory, Borland C++ directory). Параметр Input & prompting обеспечивает быстрый ввод, параметр Source debugging определяет язык отладки и т.д.

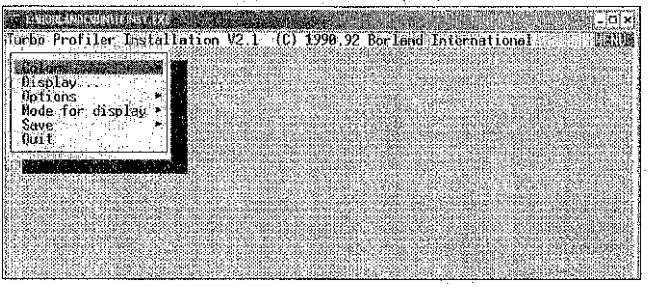

Рис. 8. Настройка Borland C++ с помощью утилиты TDINST

Пункт Mode of display служит для задания видеорежима работы дисплея (параметр Default задает режимы по умолчанию; параметр Color задает использование цветного режима; Black and white – черно-белого алфавитно-цифрового режима с 80 столбцами на экране; Monohrome – монохроматического режима; LCD – для LCD мониторов).

Сохранение настроек выполняется командой Save, выход из программы – командой Quit.

2. Порядок проведения работы

2.1. Порядок выполнения работы

1. Получить у преподавателя вариант задания.
2. Ознакомиться с каталогами ПЭВМ, где размещаются модули системы программирования Borland C++.
3. Сформировать на заданном преподавателем диске свой рабочий подкаталог.
4. Загрузить Borland C++, ознакомиться с опциями главного меню и выполнить настройку среды на каталоги Borland C++ и свой рабочий каталог. Создать конфигуратор.
5. Выйти из Borland C++ и повторным входом убедиться, что конфигуратор создан и установленные в п.4 опции сохранены.
6. Создать файл конфигурации пакетного компилятора и описать в нем минимально-необходимый набор опций.
7. Скопировать BC.EXE в свой рабочий каталог и выполнить его настройку утилитой TDINST. Сохранить указанный вариант в своем разделе. Повторным входом в систему убедиться, что конфигуратор создан и установленные в нем опции сохранены.

2.2. Форма отчётности

Отчетом по данной работе является сформированный рабочий каталог, содержащий все файлы, созданные в пп. 3, 4, 6, 7.

2.3. Вопросы для проверки

1. Состав и назначение модулей Borland C++.
2. Назначение и способы создания конфигуратора.
3. Способ задания режима работы Borland C++ по умолчанию
4. Назначение опций среды Borland C++.
5. Назначение пакетного компилятора Borland C++.

ТЕМА 2. ИНТЕГРИРОВАННАЯ СРЕДА BORLAND C++. ВСТРОЕННЫЙ РЕДАКТОР ТЕКСТА

1. Справочная информация по теме

1.1. Общая характеристика среды

Borland C++ предоставляет пользователю специальную интегрированную инструментальную среду, в рамках которой можно поэтапно выполнять все необходимые действия по разработке программ, не обращаясь к другим инструментальным средствам.

После запуска Borland C++ экран дисплея делится на 4 функциональные части: главное меню, окно редактора, окно сообщений и строка быстрой подсказки.

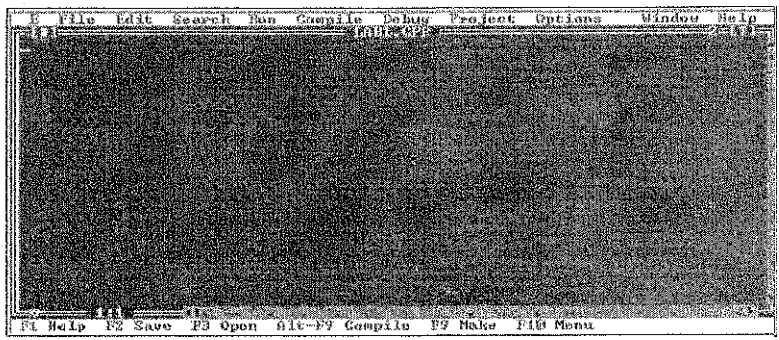

Рис. 9. Интерфейс системы Borland C++

Вход в главное меню выполняется клавишей F10, вход в подменю – комбинацией ALT-<символ>, где символ – выделенная буква в названии соответствующей команды или подменю. Перемещение внутри меню и подменю осуществляется клавишами управления курсора, клавишами Enter и ESC, выход из меню в ранее активное окно осуществляется клавишами ESC либо F6.

Для перемещения по пунктам меню Borland C++ можно использовать манипулятор "мышь". Для входа в нужный пункт меню необходимо выбрать его левой кнопкой мыши, аналогично производится перемещение внутри подменю и выбор там необходимого пункта.

Встроенный экранный редактор инициирует окно для создания и редактирования исходных текстов программ. Редактирование текста выполняется выбором из пункта меню Edit необходимых операций.

Для операций редактирования используют специальные клавиши и их комбинации (см. Приложение 2). Например:

Backspace – стереть символ слева от курсора;
Del – стереть символ, на который указывает курсор;
Ctrl-Y – стереть строку, на которой располагается курсор;
Enter – вставить новую строку в старую.

Выход из окна редактора (в главное меню) осуществляется нажатием клавиши F10 или F6 (в окно сообщений). Размер окна изменяется клавишей F5.

Аналогичным образом выполняются манипуляции с окном сообщений, которое используется при отладке для вывода диагностических сообщений и результатов трассирования. Переключение между окнами осуществляется клавишей F6.

Получение справочной информации о среде осуществляется нажатием клавиши F1 или для получения справочной информации по элементам языка С - путем подвода курсора к соответствующей языковой конструкции и нажатием комбинации CTRL-F1.

Строка быстрой подсказки показывает краткую справку о назначении основных функциональных клавиш.

Выход из системы Borland C++ производится командой (режимом) главного меню File-Quit либо комбинацией клавиш Alt-X.

В распоряжении пользователя находятся также "горячие" (функциональные) клавиши, срабатывающие по нажатию клавиши Alt и первой буквы названия соответствующего режима (пункта) основного меню или соответствующей команды (см. Приложение 1).

1.2. Структура и режимы главного меню

Все режимы (пункты) главного меню, кроме Edit, имеют соподчиненные меню в виде столбцов подменю со многими опциями (пунктами) или с последовательно отображаемыми иерархическими меню.

Режим (пункт главного меню) **File** обеспечивает операции с файлами и каталогами.

Подпункт New инициирует редактор для создания нового файла с названием NONAME.CPP.

Подпункт Open открывает новое окно редактора и загружает файл с указанным именем. Соответственно открывается диалоговое окно, в поле ввода которого можно вписать нужное имя файла. Кнопка Replace используется для замены существующего в активном окне редактора текста на текст, считанный из файла.

Подпункт Save записывает файл, находящийся в редакторе, на диск. Если имя файла NONAME.CPP, то редактор обеспечивает его переименование.

Подпункт Save as перезаписывает файл под новым или тем же именем на место его старой версии.

Подпункт Save all записывает содержимое всех окон редактора в соответствующие файлы.

Подпункт Change dir позволяет изменить текущий каталог пользователя (параметр CHDIR – сменить текущий каталог; параметр REVERT – восстановить прежний каталог).

Подпункт Print управляет выводом содержимого активного окна на принтер.

Подпункт DOS shell обеспечивает временный выход из Borland C++ с возвратом по Exit;

Подпункт Quit завершает работу Borland C++ , передавая управление ОС.

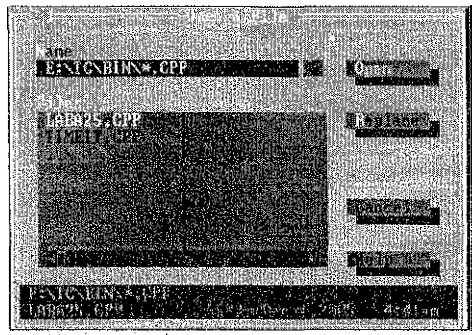

Рис. 10. Диалоговое окно Open a File

Режим (пункт главного меню) **Edit** обеспечивает операции с буфером обмена (ClipBoard).

Подпункты Undo, Redo соответственно отменяют последнее действие и отменяют действие команды Undo.

Подпункты Cut, Copy соответственно вырезают или копируют выделенный в окне редактора блок в буфер обмена.

Подпункт Paste копирует содержимое буфера обмена в окно редактора, при этом содержимое буфера не изменяется и может использоваться повторно.

Подпункт Clear удаляет из окна редактора выделенный блок, но не помещает его в буфер обмена.

Подпункт Copy example копирует выбранный пример из справочной информации.

Подпункт Show clipboard выводит содержимое буфера обмена.

Режим (пункт главного меню) **Search** обеспечивает поиск и обработку в тексте (программе) указанного фрагмента.

Подпункт Find обеспечивает поиск нужного фрагмента в активном окне редактора. Поиск ведется по заданному образцу – строке или по образцу-описателю – шаблону.

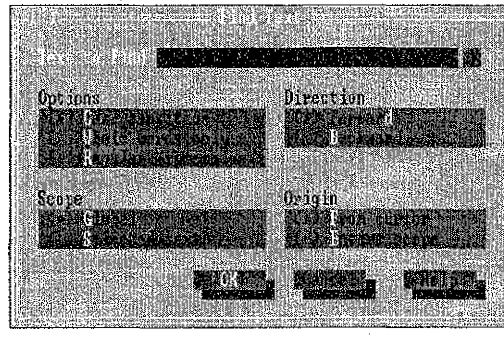

Рис. 11. Диалоговое окно Find Text

В поле выбора либо заносится слово, на которое указывал курсор в активном окне редактора, либо можно ввести новый фрагмент в поле Text to Find. После нажатия на клавишу Enter редактор отыщет этот фрагмент в тексте и установит курсор на его начало. Для оптимизации поиска используются специальные настройки, управляющие поиском. Они задается как параметры: Case sensitive – прописные буквы отличать от строчных; Whole word only – искать по совпадению целых слов; - Regular expression – искать по заданному выражению; Forward – направление поиска вниз по тексту; Backward – направление поиска вверх по тексту; Global – искать во всем тексте; Selected text – искать только в выделенном блоке; From cursor – начать поиск от текущего положения курсора; Entire scope – искать от начала текста.

Кроме этого в поле ввода можно ввести выражение–описатель текста (поисковый шаблон), которое формируется из следующих специальных символов:

символ ^ в начале выражения-описателя означает начало текстовой строки в искомом тексте;

символ $ в конце выражения-описателя означает конец текстовой строки в искомом тексте;

символ . означает, что на этом месте может стоять любой символ;

символ * после любого символа означает любое количество этих символов, которые могут стоять вместо него. Например, 1* может означать 1, 11, 111, 16, 1дщ, 1235 и т.д.;

Библиотека Государственного

символ + после символа означает один или больше этих символов, которые могут стоять вместо него. Например, bo+ означает bo, bot, boo, но не b, be и т.д.;

символ \ перед специальным символом означает сам символ. Например, * означает сам символ *.

Ввод управляющих символов производится с помощью префикса ^P. Например, можно ввести Ctrl T путем удержания клавиши Ctrl в нажатом состоянии и нажатия P, а затем T. Ограничитель строки в образце поиска задается как Ctrl-MJ (возврат каретки/перевод строки). Комбинация Ctrl-A имеет специальное значение, соответствующее любому символу и может использоваться в качестве метасимвола в строке поиска.

Подпункт Replace обеспечивает поиск нужного текстового фрагмента и замену его на новый (параметр Text to find служит для ввода искомого текста; New text служит для ввода нового текста; Prompt on Replace предписывает системе запрашивать подтверждение при замене текста; параметр Change all используется для режима поиска и безусловной замены всех обнаруженных фрагментов текста).

Подпункт Search again обеспечивает повторный поиск с ранее установленными параметрами.

Подпункт Go to line number обеспечивает переход на строку с указанным номером.

Подпункт Previous error обеспечивает переход к месту предыдущей ошибки.

Подпункт Next error обеспечивает переход к месту следующей ошибки.

Подпункт Locate function обеспечивает поиск в тексте программы функции с заданным названием.

Режим (пункт главного меню) **Run** содержит команды управления запуском и отладкой программы.

Подпункт Run вызывает автоматическую компиляцию и компоновку программы из файла редактора.

Остальные подпункты (Program Reset, GoToCursor, Trace info, StepOver используются для отладки программ в режиме трассировки).

Подпункт Program reset останавливает текущий сеанс отладки, освобождает память, выделенную программе, закрывает все файлы, которые использовала программа.

Подпункт Go to cursor запускает и выполняет программу до строки - команды, которая выделена курсором; далее возможно пошаговое выполнение программы.

Подпункт Trace Into обеспечивает пошаговое (построчное) выполнение программы и ее функций.

Подпункт Step over обеспечивает пошаговое (построчное) выполнение программы, а функции выполняются за один шаг.

Подпункт Arguments обеспечивает ввод значений аргументов для командной строки.

Режим (пункт главного меню) **Compile** содержит команды, обеспечивающие компилирование и создание объектных и выполнимых файлов.

Подпункт Compile обеспечивает компилирование программы, загруженной в активное окно редактора.

Подпункт Make создаёт выполняемый файл программы (совмещает все операции по созданию загрузочного кода).

Подпункт Link обеспечивает компоновку файлов программы с библиотечными файлами без последующего выполнения.

Подпункт Build all перекомпилирует все файлы.

Подпункт Information показывает статистику текущего файла;

Подпункт Remove messages позволяет удалить ранее полученные сообщения.

Режим (пункт главного меню) **Debug** содержит команды, обеспечивающие управление ходом отладки программ.

Подпункт Inspect позволяет задать переменную или выражение для просмотра.

Подпункт Evaluate/modify дает возможность в процессе отладки просмотреть содержимое любой переменной или найти значение любого выражения и при необходимости скорректировать полученные значения (параметр Expression задает выражение; Result – результат; параметр New Value задает новое значение).

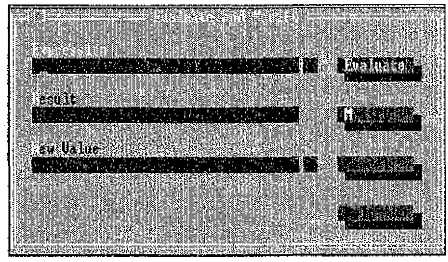

Рис. 12. Диалоговое окно Evaluate and modify

Подпункт Call stack делает активным окно отображения программного стека, где протоколируются вызовы всех функций.

Подпункт Watches позволяет наблюдать за изменением при отладке программы указанных переменных или выражений, позволяет управлять окнами, отображающими текущие значения объектов, обрабатываемых трассируемой программой.

Параметры Add watch, Delete watch позволяют соответственно добавить или удалить выделенную переменную или выражение из окна наблюдения; параметр Edit watch позволяет редактировать переменную или выражение в окне наблюдения; параметр Remove all watches очищает окно наблюдения.

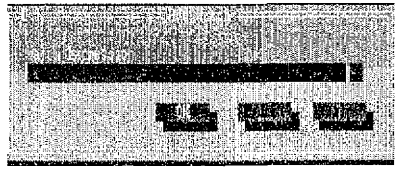

Рис. 13. Диалоговое окно Add Watch

Подпункт Toggle breakpoint позволяет переключаться между контрольными точками, расставленными в программе для отладки.

Подпункт Breakpoints позволяет просматривать все контрольные точки, при необходимости удалять, перемещать любую контрольную точку или задавать условия ее работы. Например, в качестве условия, управляющего работой контрольной точки, можно указать любое допустимое условное выражение.

Режим (пункт главного меню) **Project** содержит команды, обеспечивающие управление проектами программ.

Подпункты Open project, Close project позволяют открыть или закрыть проект.

Подпункты Add item, Delete item позволяют добавить или удалить файл из проекта.

Подпункт Local options позволяет управлять опциями проекта, а подпункт Include files просматривать файлы проекта.

Режим (пункт главного меню) **Options** позволяет выполнять настройку интегрированной среды (см. тему 1).

Режим (пункт главного меню) **Window** содержит команды, обеспечивающие управление окнами многооконной системы Borland C++ .

Подпункт Size/Move обеспечивает управление местоположением и размером активного окна. Подпункт Zoom увеличивает размер активного окна до максимальных размеров или возвращает ему прежний вид.

Подпункт Cascade располагает окна каскадом. Подпункт Tile располагает окна так, чтобы каждое было видно на экране, и все они имели приблизительно одинаковые размеры.

Подпункт Next активизирует следующее окно. Подпункт Close закрывает активное окно. Подпункт Close all закрывает все открытые окна.

Подпункт Message обеспечивает переход к окну "Message", Output к окну "Output", Watch – переход к окну "Watch".

Подпункт Register открывает окно регистров, подпункт Project показывает файлы, используемые в проекте, подпункт Project notes выводит файл примечаний проекта в окно редактора.

Подпункт List all – выводит список всех открытых окон среды.

Режим (пункт главного меню) **Help** содержит команды, обеспечивающие получение справочной информации при работе с системой.

Подпункт Contents выводит на экран содержание справочной информации системы, а подпункт Index позволяет получить алфавитный список справочной информации.

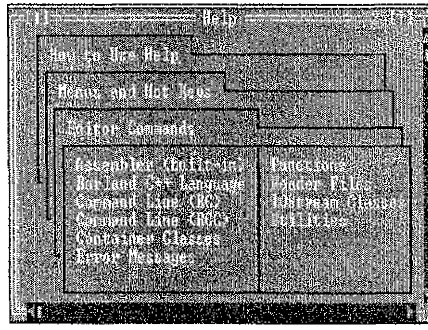

Рис. 14. Состав справочной информации (окно Help)

Справочная информация содержит разделы: инструкция по пользованию справочником; меню и "горячие" клавиши; команды редактора текста; справочник по языку и программным средствам. Последний раздел включает справочные сведения об ассемблере, языке Borland C++, опциях командного процессора, классах, функциях, утилитах, заголовочных файлах и т.п.

Состав справочной информации об языке Borland C++ представлен на рисунке ниже.

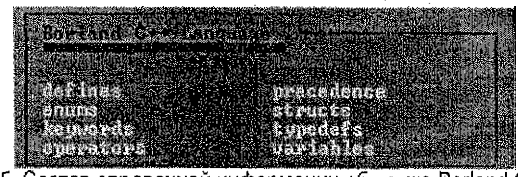

Рис. 15. Состав справочной информации об языке Borland C++

Он включает определения, перечислимые типы, ключевые слова, операторы, их приоритеты, системные структуры, типы и переменные.

Основные классы функций языка Borland C++ приведены на рисунке ниже. Они, в частности, включают графические функции, математические, функции преобразования, функции управления памятью, функции работы с временем и датами, функции для обработки строковых данных и т.д.

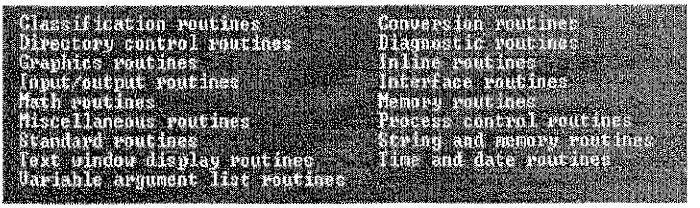

Рис. 16. Классификация функций языка Borland C++

Описания функций сгруппированы в заголовочные файлы. Справочная информация по этим файлам выглядит как на рисунке ниже.

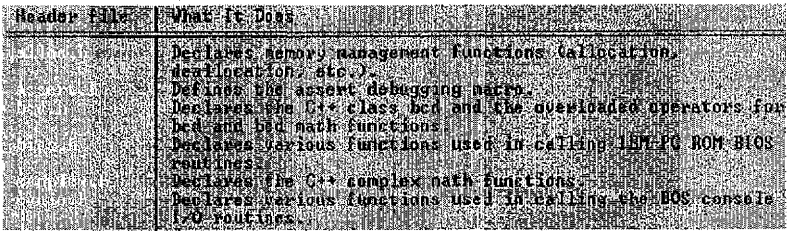

Рис. 17. Заголовочные файлы Borland C++

Подпункт Topic search обеспечивает поиск зарезервированного слова или имени стандартной процедуры. Подпункт Previous topic обеспечивает возврат к предыдущей справке.

Подпункт Help on help обеспечивает получение инструкции, справки по использованию справочной системы Borland C++.

Подпункт Active file определяет состав используемых справочных файлов.

Подпункт About сообщает информацию о текущей версии установленной системы Borland C++.

1.3. Окно редактирования

В окне редактора может содержаться текст, который не будет помещаться целиком в окне. Правая и нижняя рамки такого окна содержат указатели размеров, в которых показывается положение демонстрируемого в окне фрагмента относительно полных размеров текста. Эти поля можно использовать для того, чтобы перемещать окно относительно текста с помощью мыши.

Кроме этого перемещение по тексту можно выполнять с помощью следующих клавиш: Page Up - перемещение текста на одну страницу вверх; Page Down – перемещение текста на одну страницу вниз; Home – перемещение в начало текущей строки; End – пе-

ремещение в конец текущей строки; Ctrl-Page Up – перемещение в начало текста; Ctrl-Page Down – перемещение в конец текста.

В верхней части окна редактирования имеются два поля, которые используются при работе с мышью. Поле  служит для закрытия окна редактирования, поле  - для развертывания окна на полный экран.

При работе с текстом используют два режима ввода: режим вставки символов и режим замены.

В режиме вставки курсор имеет обычный вид. Данный режим позволяет вставлять с клавиатуры в позицию курсора новые символы в имеющийся текст, при этом текст сдвигается вправо от курсора по мере ввода нового текста.

В режиме замены текст, вводимый с клавиатуры, заменяет прежние символы, начиная с позиции курсора. В этом режиме курсор, как правило, имеет вид прямоугольника, полностью закрашивающего заменяемый символ. Переключение между режимами вставки и замены производится клавишей Ins.

1.4. Основные команды встроенного редактора текста Borland C++

Редактор использует несколько десятков команд (см. Приложение 2): для перемещения и быстрого перемещения курсора к границам строк, к началу и концу файла, а также к последней позиции курсора; команды вставки и удаления текста (символов, слов, строк, блоков).

Здесь слово определяется как последовательность символов, ограниченная одним из следующих знаков: пробел < > , ; . ( ) [ ] ^ ' * + - / $.

Характеристика групп команд приведена ниже.

1. Команды перемещения курсора в тексте по экрану – клавиши "влево", "вправо", "вверх", "вниз", "PgUp", "PgDn".

2. Команды выделения блоков, фрагментов текста. Команды Ctrl-K B, Ctrl-K K служат для отметки начала и конца участка текста - блока. Для отметки необходимо разместить курсор на помечаемое в качестве начала или конца место текста и выполнить соответствующую команду.

3. Команды копирования, удаления, пересылки блока (Ctrl-K C, Ctrl-K Y, Ctrl-K V), которые работают только тогда, когда блок выделен. При пересылке ранее отмеченный блок перемещается из исходного положения в позицию курсора и стирается в исходном месте. При копировании стирание не производится.

4. Команды чтения, записи блока на диск (Ctrl-K R, Ctrl-K W).

По команде чтения редактор запрашивает в окне имя файла для чтения, читает файл и вставляет в текущий текст в позицию курсора как блок. Прочитанный текст отмечается повышенной яркостью.

По команде записи редактор запрашивает в окне имя файла для записи, запрашивает подтверждение на перезапись существующего файла и записывает ранее отмеченный блок в файл. В текущем файле блок не меняется и остается помеченным.

5. Команды поиска (Ctrl-Q F), поиска с заменой (Ctrl-Q A) необходимого элемента текста. Команда поиска позволяет найти образец - строку длиной до 30 символов. По команде строка состояния очищается, редактор запрашивает режим поиска и образец, который может содержать любые символы, включая управляющие.

Их ввод производится с помощью префикса ^P. Например, можно ввести Ctrl T путем удержания клавиши Ctrl в нажатом состоянии и нажатия P, а затем T. Ограничитель строки в образце поиска задается как Ctrl-MJ (возврат каретки - перевод строки). Комбинация Ctrl-A имеет специальное значение, соответствующее любому символу, и может использоваться в качестве метасимвола в строке поиска.

Образец поиска можно редактировать командами "символ влево-вправо", "слово влево-вправо". "Слово вправо" вызывает для редактирования предыдущую строку поиска. Завершение операции поиска производится командой Ctrl-U.

6. Команды, используемые для структурирования текста программы.

Команда включения-выключения автоотступа (Ctrl-O I), обеспечивает автоматический отступ последовательных строк. По нажатии клавиши Ввод курсор возвращается в начальный столбец только что завершенной строки.

Команда включения-выключения режима табуляции (Ctrl-O T). При ее включении табуляция производится с фиксированными остановками по 8 позиций. При выключении табуляция будет перемещать курсор к началу каждого слова в строке.

2. Порядок проведения работы

2.1. Порядок выполнения работы

1. Получить у преподавателя вариант задания.
2. Загрузить Borland C++, ознакомиться со структурой главного меню, окном редактора и сообщений, а также подсказками среды.
3. Войти в свой каталог и просмотреть существующие там файлы.
4. Ознакомиться с клавиатурой управления редактором, для чего войти в режим редактирования и набрать текст программы.
5. Ознакомиться с командами редактора (см. Приложение 2).

2.2. Форма отчётности

Отчетом по данной работе являются сформированные исходные тексты и промежуточные файлы с копируемыми блоками текста.

2.3 Вопросы для проверки

1. Команды основного меню и их назначение.
2. Опции основного меню, их отличие от команд.
3. Способы получения подсказки.
4. Различия между командами Save и Save as.
5. Вставка текста из другого файла.
6. Различие между режимами Вставка и Замена.
7. Очистка экрана редактора.
8. Замена фрагмента текста на образец во всей программе.
9. Блочные команды.
10. Использование блочных команд при работе с файлами.
11. Использование табуляции.
12. Использование справочной системы.

ТЕМА 3. РАЗРАБОТКА ПРОГРАММ В BORLAND C++

1. Справочная информация по теме

Разработка программ включает ряд типовых этапов, направленных на последовательное преобразование алгоритма решения задачи в конечный программный продукт. Это создание исходного текста (например, в файле EXAMPLE.CPP), компилирование исходного текста и получение объектного модуля (EXAMPLE.OBJ) с разрешенными внешними ссылками. Это компоновка и получение загрузочного модуля с подключенными стандартными библиотеками (EXAMPLE.EXE). Это выполнение программы. Схематично этапы представлены ниже:

EXAMPLE.CPP -> EXAMPLE.OBJ -> EXAMPLE.EXE .

На всех указанных этапах производится отладка программы: локализация ошибок и соответствующее редактирование исходного текста.

Пример создания программы с одним исходным файлом, хранящемся под именем EX1.C, иллюстрируется рисунком 17.

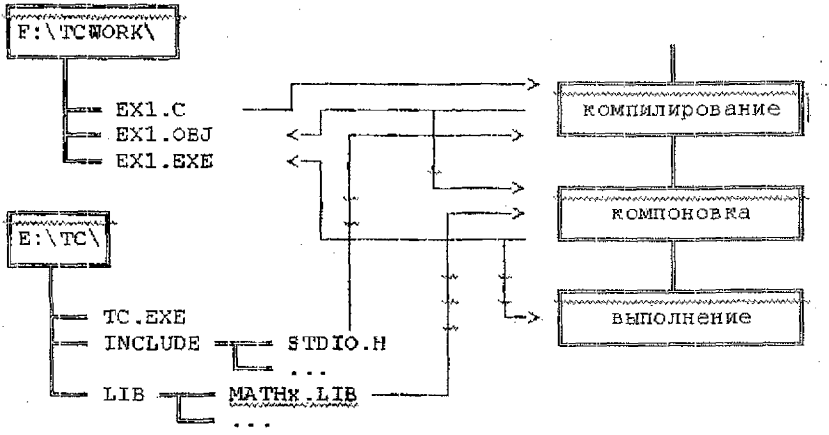

Рис. 18. Создание программы с одним исходным файлом

1.1 Разработка исходных текстов

Создание исходных текстов выполняется в режиме FILE главного меню, активизирующего окно редактора. Создание текстов включает перечисленные ниже этапы.

1. Загрузка исходного текста (файла) в редактор, если он уже существует, либо автоматическое создание пустого файла с именем NONAME.CPP для его последующего заполнения.

Для загрузки и редактирования существующего файла используется команда File-Open с вводом имени файла, вызываемого на редактирование. Либо используются команды File-Directory и File-Change Dir для просмотра каталогов и выбора имени нужного файла клавишами управления курсором.

Для создания нового исходного файла используется команда File-New, открывающая окно для редактирования файла с именем NONAME.CPP.

2. Редактирование текста (см. тему 2).

3. Обновление (сохранение) исходного текста, которое выполняется по нажатию клавиши F2 либо отработкой пунктов меню интегрированной среды File-Save. Для сохранения файла под новым именем необходимо выполнить команду File-Save as и ввести новое полное имя файла. Отметим, что конкретный исходный текст может быть сохранен только как один конкретный файл.

Пример 1. EX1.CPP

```
#include  <stdio.h>
main( )
{
   int  c;
   int  d;
   c =  2;
   d =  3;
   printf( " Сумма = %d", c+d);
}
```

Пример 2. EX2.CPP

```
#include <stdio.h>
void sum( int x, int y);
main( )
{
   int  c;
   int  d;
   c = 2;
   d = 3;
   sum( c, d);
}
void sum( int x, int y)
{
   printf( " Сумма = %d", x+y );
 }
```

Также могут создаваться программные проекты, состоящие из нескольких программных компонент, для хранения которых может быть использовано несколько файлов.

Пример 3. EX3.CPP

```
#include  <stdio.h>
#include  "sum.cpp"
void sum( int x, int y );
main( )
{
```

```
    int c;
    int d;
    c = 2;
    d = 3;
    sum ( c, d );
}
```

Программа использует функцию sum, размещенную в файле SUM.CPP

```
void sum(int x,int y)
{
    printf( " сумма = %d", x+y);
} .
```

1.2 Компиляция, компоновка и запуск программ

Компиляция включает этапы, конкретное содержание которых может меняться в зависимости от того, отлаживается простая программа (все программные компоненты доступны в одном файле) или проект. Основные этапы перечислены ниже.

1. Установка опций среды таким образом, чтобы компилятор и компоновщик могли найти нужные файлы и разместить результаты компиляции. Для этого необходимо настроить пути к соответствующим папкам (См. тему 1 – параметры Include directories, Library directory, Output directory и т.д.).

2. Загрузка исходного текста программы в редактор.

3. Компиляция и редактирование программы.

Исходный файл необходимо откомпилировать, а затем скомпоновать полученный объектный файл со стандартными библиотеками и файлом загрузки, что при разработке простых программ обеспечивается командой главного меню Compile, а при работе с проектом командами пункта главного меню Project (см. тему 3).

Пункт главного меню Compile включает ряд команд.

Команда Compile обеспечивает компиляцию исходного текста программы в объектный файл с расширением .OBJ. В качестве имени объектного файла берется имя исходного файла или, если оно не указано, имя последнего файла, загруженного в окно редактора.

Команда Make создает выполнимый файл проекта с именем, указанным в меню Project-Project name, или по правилам предыдущего пункта. При этом перекомпилируются только текущие файлы.

Команда Link компонует текущие объектные файлы с библиотечными файлами (.LIB), создавая новый выполнимый файл с расширением .EXE.

Команда Build All аналогична команде Make, но перекомпилируются все файлы независимо от даты их создания и изменения.

В случае обработки простой программы можно без построения проекта провести компиляцию, компоновку и выполнение различными способами, но проще всего нажать F9 (Make) или выбрать в меню Compile опцию Make EXE файл.

4. Запуск программы. Для запуска программы нажимается Alt-R или инициируется команда Run основного меню. Можно также запустить программу из командной строки операционной системы, набрав ее имя без расширения.

1.3 Отладка программ

При разработке программ приходится, как правило, многократно исправлять ошибки. Для отладки программ в Borland C++ существует мощный и гибкий интегрированный отладчик, являющийся частью среды и работающий на уровне исходных кодов.

Отладчик позволяет без дополнительных усилий по модификации исходного текста: трассировать (выполнять построчно) модульные, оверлейные программы, просматривать значения переменных и выражений при построчном выполнении, при необходимости корректировать полученные промежуточные значения и продолжать выполнение программы без возврата к ее программе.

Ошибки, которые наблюдаются в программах, традиционно подразделяются на ряд групп. Это синтаксические ошибки, являющиеся ошибками кодирования, ошибками записи на языке С алгоритмов программ. Семантические ошибки, ошибки выполнения, возникающие при обработке корректных по синтаксису программ при попытке совершить недопустимое действие. Например, попытка деления на 0. Возникновение ошибок выполнения отмечается сообщением RunTime error at seg:ofs. Это означает, что произошла ошибка выполнения в операторе с адресом "сегмент:смещение". При этом выполнение программы прекращается и подсвечивается ошибочный оператор. Логические ошибки, не вызывающие сообщений и не прерывающие выполнения программы, являющиеся наиболее сложными для обнаружения. Например, неинициализированная переменная может проявиться в некорректных результатах или в неустойчивом по результатам выполнении программы.

1.3.1 Исправление синтаксических ошибок

Первая группа ошибок (см. Приложение 3) фиксируется в виде сообщений компилятора и компоновщика. Это "Грубые ошибки" (Fatal error), приводящие к прекращению компиляции; "Ошибки" (Error), приводящие к завершению текущей фазы компиляции; "Предупреждения" (Warning), которые не прекращают компиляцию, но предупреждают о действиях, которые могут привести к искажениям результатов. Формат сообщения о синтаксической ошибке:

<класс_ошибки> <имя_файла> <номер_строки> <текст_сообщения> .

При появлении сообщения "Press any key" посредством нажатия любой клавиши в редактор загружается файл, содержащий ошибку. Появляется окно сообщений и светлая полоса маркера строки отмечает первую ошибку или предупреждение. Одновременно в окне редактора подсвечивается строка, к которой относится указанное сообщение, что позволяет, ознакомившись с ошибками, исправить текст исходного файла.

Просмотр других сообщений осуществляется перемещением маркера по строкам окна сообщений, например, с помощью клавиш управления курсором. Горизонтальное передвижение по строкам выполняется клавишами влево-вправо, увеличение числа одновременно просматриваемых сообщений выполняется клавишей F5.

Для исправления ошибки активизируется окно редактора. Для этого маркер строки окна сообщений помещается на анализируемом сообщении и нажимается клавиша Ввод. При этом курсор расположится на строке исходного текста, вызвавшем сообщение об ошибке. Ошибка исправляется средствами редактора.

Продолжение корректировок либо возвращение к окну сообщений выполняется нажатием клавиши F6 и выбором очередного сообщения, например, нажатием F8. При этом редактор помещает курсор на место, соответствующее очередной ошибке. По аналогии клавишей F7 можно двигаться в обратном направлении. После исправления ошибок файл необходимо перекомпилировать, например, нажатием клавиш Alt-F9.

1.3.2 Трассировка программ

Встроенный отладчик предоставляет возможность управляемого выполнения программы с возможностью просмотра данных и объектов, которые интересуют программиста.

Режим отладки (способ управляемого выполнения программы):

пошаговый - с начала программы строка за строкой с обработкой вызываемых функций за один шаг или построчно;

по контрольным точкам (по точкам останова) - "неинтересные" части программы, от текущей позиции до заранее указанной строки, помеченной курсором или точкой останова, выполняются как один оператор за один шаг, а остальные фрагменты, начиная с точки остановки, "прокручиваются" построчно.

Способ просмотра контролируемых данных и объектов определяется конкретными просматриваемыми объектами (переменными, структурами, выражениями). При этом обеспечивается интерактивная проверка результатов с помощью калькулятора вычислений (Evaluate), а также возможна модификация промежуточных значений.

Исходя из специфики отслеживаемых объектов, могут быть использованы следующие инструментальные средства:

1. Экран выполнения (Execution), доступный при выходе из среды для просмотра результатов выполнения; окно вывода (Output) для отображения копии экрана выполнения; окно редактирования исходного текста; окно просмотра (Watch) трассируемых значений и сообщений об ошибках, инициируемое клавишей F6. Для "прокрутки" сообщений используются клавиши управления курсором.

2. Пункт меню DEBUG.

Здесь используется подменю Evaluate/Modify для активизации окна калькулятора с тремя полями для автономного вычисления значений выражений-констант либо для аналогичных вычислений при отладке, когда можно ссылаться на переменные и выражения программы. Соответственно выражение задается либо ручным набором, либо позиционированием курсора на выражении в тексте и выбором режима Evaluate (Ctrl-F4).

Подменю Watches используется для добавления (Add Watch или Ctrl-F7), для редактирования (Edit Watch), для удаления (Delete Watch) выражения из окна просмотра, а также для удаления всех выражений из окна просмотра (Remove All Watches).

Подменю BreakPoints и Toggle breakpoint для работы (установки-отмены) с точки останова.

3. Пункт меню OPTIONS.

Здесь используется подменю Debugger для настройки отладки. Параметр Source Debugging служит для подготовки отладки программы в интегрированной среде с использованием Borland-отладчика путем добавления компоновщиком отладочной информации в конец EXE-файла программы (режим ON). Параметр Standalone определяет, что трасса размещается в EXE-файле, но не используется отладчиком. Параметр None определяет, что трасса не запоминается.

4. Пункт меню RUN.

Здесь для управления способом трассировки используются команды. Команда Program Reset – для завершения текущего сеанса отладки. Команда Go To Cursor – для выполнения программы до строки, где находится курсор (F4). Команда Trace Into – для выполнения каждой текущей строки, включая трассировку внутри вызываемых функций (F7). Команда Step Over аналогична Trace Into, но без трассировки вызываемых функций (F8).

Соответственно для отладки программ в Borland C++, как правило, необходимо:

1) подготовить отладчик к использованию - установить соответствующие переключатели отладки пунктов меню Debug и Options, чтобы сделать отладочную трассу доступной отладчику откомпилированного исходного текста. В частности, включить в меню Options, в пункте Compiler на вкладке Advanced Code Generation параметры Options-Line numbers debug info и Options-Debug info in OBJs;

2) определить состав трассы путем подготовки окна просмотра и записи в него трассируемых переменных. Далее отладка производится путем комбинирования действий, указанных ниже в п.3-5;

3) при трассировании с начала программы перейти на пункт 4. При трассировании с произвольного места программы предварительно пометить его, установив там точку останова. Для перехода в точку начала трассирования выполнить Ctrl-F9;

4) выполнить трассирование программы клавишами F7 или F8, одновременно используя выводимые результаты, окно просмотра для анализа трасс, команды подпункта Debug-Watches для управления-модификации состава трассы;

5) завершить отладку комбинацией клавиш Ctrl-F2 или используя режим Run-Program Reset. Для редактирования точек прерывания использовать команды подпункта Debug - BreakPoints и Toggle breakpoint.

1.4 Разработка проектов программ

1.4.1. Файл проекта

Одним из преимущества Borland C++ является обеспечение работы с большими проектами - связывание в единый проект нескольких исходных и объектных файлов с возможностью раздельной компиляции. Для этого необходимо описать файл проекта. Последний создается любым текстовым редактором и содержит перечень файлов (по одному на каждой строке), включаемых в проект. Имя файла проекта может отличаться от имени главного файла, а имя выполнимого файла (и имя любого MAP-файла, создаваемого компоновщиком) будет определяться именем файла проекта.

Для работы с проектами используется пункт главного меню Project и соответствующие команды. Работа с проектом, как правило, предполагает описание файла проекта, если он новый.

1.4.2. Особенности отладки проекта

Создание файла проекта осуществляется в разных режимах, отличающихся реакцией процедуры make на ошибки. Для прерывания процедуры make после компиляции файла при наличии предупреждений или ошибок необходимо установить параметры Warnings (предупреждения) или Errors (ошибки), а для работы с полным списком ошибок во всех

исходных файлах устанавливается режим Fatal errors (фатальные ошибки). Для этого используется пункт главного меню Options, подпункт Make, команда Break make on.

Сообщения об ошибках появляются в окне сообщений. Просмотр и исправление синтаксических ошибок выполняется аналогично тому, как это делается в однофайловых программах. Курсор должен располагаться на первом сообщении об ошибке или предупреждении в окне сообщений, при этом в файле, к которому относится сообщение, подсвечивается ошибочный оператор.

Перед созданием проекта окно сообщений можно очистить, включив Remove messages в меню Compile. При этом очищаются все текущие сообщения.

2. Порядок проведения работы

2.1. Порядок выполнения работы

1. Получить у преподавателя варианты заданий на разработку программ в Borland C++.
2. Создать простую программу.
3. Выполнить ее компиляцию и компоновку. Убедиться, что в каталоге создаются необходимые файлы.
4. При наличии исправить синтаксические ошибки (см. Приложение 3). При отсутствии внести 5-10 ошибок различных типов в исходные тексты, выполнить компиляцию, анализ сообщений и их исправление. Сохранить исходные и исправленные тексты для отчета.
5. Выполнить трассировку программы в разных режимах отладки на указанных преподавателем участках и проконтролировать изменение заданных преподавателем объектов.
6. Внести логические ошибки и ошибки выполнения в исходные тексты. Локализовать их путем трассирования. Сохранить исправленные и исходные тексты для отчета.
7. Создать новый проект. Повторить п. 3-7 для проекта.

2.2. Форма отчетности

Отчетом по данной работе являются тексты программ и файлов проектов, а также результаты пошаговой трассировки указанных преподавателем переменных.

2.3. Вопросы для проверки

1. Этапы разработки программ в Borland C++.
2. Различие в отладке простых программ и проектов.
3. Различие процедур подменю COMPILE - Make, Link, Build?
4. Отличия проектов от простых программ.
5. Способы выявления и исправления ошибок в Borland C++.
6. Включение отладчика для проведения трассировки.
7. Управление режимом трассировки, окнами просмотра, точками останова.
8. Использование встроенного калькулятора.
9. Создание исполнимого файла без отладчика.

ТЕМА 4. РАЗРАБОТКА ПРОГРАММ В BORLAND C++ С ИСПОЛЬЗОВАНИЕМ КОМАНДНОЙ СТРОКИ

1. Справочная информация по теме

1.1. Компиляция, компоновка и запуск программ из командной строки

Кроме интегрированной среды для запуска программ Borland C++ можно использовать традиционный интерфейс - "командную строку". В ряде случаев он является единственным выходом для создания программы, например, если она включает встроенный ассемблерный код.

Запуск Borland C++ из командной строки в ответ на приглашение DOS выполняется в формате BCC [опция_режима ... опция_режима] имя_файла ...имя_файла, где опции задают режим работы компилятора, а имена - список обрабатываемых файлов, составляющих проект.

BCC компилирует исходные файлы C и ASM, связывает их вместе с объектными и библиотечными в выполнимый файл. Для выполнения только компиляции используется опция -C.

Компилятор обрабатывает файлы по следующим правилам:

имя_файла или имя_файла.c или имя_файла.xyz компилируется как имя_файла.c в OBJ-файл;

имя_файла.obj включается в проект в качестве объектного файла при его компоновке;

имя_файла.lib включается в проект в качестве библиотеки при его компоновке;

имя_файла.asm ассемблируется как *.ASM в *.OBJ программой MASM.

Затем компилятор вызывает компоновщик и сообщает ему имена соответствующего стартового файла и стандартных библиотек C.

Имя файла-проекта с расширением .EXE, являющегося результатом работы BCC, определяется по имени первого исходного или объектного файла в списке файлов командной строки. Для задания любого другого имени файла-проекта используется опция -e, стоящая перед списком имен_файлов.

Пример вызова Borland C++ из командной строки DOS:

F:/ BCWORK > E:/BorlandC/BCC P1.cpp

или

F:/BCWORK> E:/BorlandC/BCC P1.CPP P2.CPP P3.ASM P4.LIB ,

где компилируются файлы P1 и P2, ассемблируется P3, при компоновке подключается библиотечный файл P4, загрузочный файл именуется P1.

В примере

TCC -eP1 P2.cpp P3.ASM P4.OBJ P5.LIB P6.cpp ,

исходные файлы P6.cpp и P2.cpp компилируются; файл P3.ASM ассемблируется с использованием MASM; объектный файла P4.OBJ и библиотечный файл P5.LIB подключаются при компоновке; результат помещается в файл с именем P1.EXE.

1.2. Опции командной строки

Режим работы BCC определяется опциями (см. тему 1 и Приложение 4) файла конфигурации TURBOC.CFG. Большинство опций командной строки имеют аналоги в меню опций интегрированной среды Borland C++.

При необходимости, заданные в конфигураторе опции переопределяются из командной строки, опции которой имеют более высокий приоритет.

Опции отделяются от команды BCC и последующих имен файлов пробелами, каждой опции предшествует тире "-". Режим, задаваемый опцией, выключается указанием после нее тире "-", например, -K- отменяет опцию K.

Для запуска программы EX1.CPP, сохраненной в папке F:\BCWORK, задается команда

```
F:\BCWORK>E:\BorlandC\BCC  EX1.CPP
```

В примере

```
BCC -IB:\INCLUDE -LB:\LIB -eP1 -mm -C -K P2.cpp P3.ASM  P4.OBJ  P5.LIB  P6
```

опции командной строки интерпретируются следующим образом: каталог B:\INCLUDE используется для включаемых файлов; каталог B:\LIB как библиотечный; используется средняя модель памяти (-mm); разрешены вложенные комментарии (-C); переменные типа char интерпретируются как беззнаковые (-K). В следующем примере

```
BCC  -a  -f  -C  -O  -Z  -eP1  P2.cpp  P3  P4.cpp
```

BCC будет компилировать P2.cpp, P3.cpp, P4.cpp в файлы .OBJ, создавая исполнимую программу в файле P1.EXE с выравниванием к границе слова (-a), эмулятором плавающей точки (-f), регистровой оптимизацией (-Z) и оптимизацией размера программы (-O).

2. Порядок проведения работы

2.1. Порядок выполнения работы

1. Сформировать конфигуратор пакетного компилятора C.
2. Из командной строки откомпилировать программы, сформированные в предыдущих работах.
3. Запустить и выполнить программы из командной строки.
4. Повторить п.2-3 с опциями, заданными преподавателем.

2.2. Форма отчетности

Отчетом по данной работе являются сформированные объектные и загрузочные файлы, а также используемые для этого командные строки, созданные как bat-файлы.

2.3. Вопросы для проверки

1. Назначение пакетной версии Borland C++.
2. Формат запуска Borland C++ из командной строки.
3. Взаимодействие опций конфигуратора и командной строки.
4. Правила обработки файлов при запуске Borland C++ из командной строки.
5. Опции командной строки и их аналоги в основном меню.
6. Создать конфигуратор для использования средней модели памяти и оптимизации программы по быстродействию.
7. Описать командную строку для запуска программы с конфигуратором п.6 с малой моделью памяти и без оптимизации.

ЛИТЕРАТУРА

1. Касаткин А.И., Вальвачев А.И. От Turbo C к Borland C++. – Мн.: Высш. школа, 1992.

2. Паппас К., Мюррей У. Visual C++. Руководство для профессионалов: Пер. с англ. – СПб: BHV -Санкт-Петербург, 1996.

3. Шилдт Г. Самоучитель C++, 3-е изд. – СПб.: БХВ-Петербург, 2003. – 688 с.

4. Демидович Е.М. Основы алгоритмизации и программирования. Язык СИ. – Мн.: Бестпринт, 2003. – 384 с.

5. А.И. Касаткин. Управление ресурсами. Минск, Вышэйшая школа, 1992.

6. Мэтт П.Секреты системного программирования в Windows 95. - Киев: Диалектика, 1996.

7. Керниган Б., Ритчи Л., Фьюер А. Язык программирования Си. Задачи по языку Си. – М.: Финансы и статистика, 1985.

8. Бахтизин В.В., Глухова Л.А., Муравьев Г.Л. Лабораторный практикум по курсам "Конструирование программ и языки программирования" и "Программирование". – Мн.: МРТИ, 1992. – 47 с.

9. Муравьев Г.Л., Хвещук В.И., Бахтизин В.В. Методические указания по работе в интегрированной среде Турбо СИ. – Брест.: БРПИ, 1994. – 48 с.

ПРИЛОЖЕНИЕ 1. Основные "горячие" клавиши

Таблица П1.1. Список "горячих" клавиш

Клавиша	Функция
F1	вызывает на экран окно помощи
F2	сохраняет редактируемый файл на диске
F3	загружает файл для работы
F5	увеличивает активное окно до размеров экрана
F6	переключает активное окно
F9	создает выполняемый файл
F10	вызывает основное меню
Alt-F1	высвечивает последний экран справочной информации, полученный в режиме HELP
Alt-F3	загружает файл
Alt-F9	компилирует файл, загруженный в редактор
Alt-F10	сообщает информацию о версии Borland C++
Alt-C	вызывает меню Compile (компиляция)
Alt-D	вызывает меню Debug (отладка)
Alt-E	вызывает среду Edit (редактор)
Alt-F	вызывает меню File (файл)
Alt-O	вызывает меню Options (режимы)
Alt-P	вызывает меню Project (проект)
Alt-R	запускает программу
Alt-X	вызывает выход из Borland C++

ПРИЛОЖЕНИЕ 2. Основные команды текстового редактора

Основные команды перемещения курсора:

символ влево/вправо	Ctrl-S или <- / -> или Ctrl-D
слово влево/вправо	Ctrl-A / Ctrl-F
строка вверх/вниз	Ctrl-E или Вверх/Ctrl-X или Вниз
кадр вверх/вниз	Ctrl-W / Ctrl-Z
страница вверх/вниз	Ctrl-R или PgUp/Ctrl-C или PgDn

Команды быстрого перемещения курсора:

начало/конец строки	Ctrl-Q S или Home/Ctrl-Q D или End
верх/низ экрана	Ctrl-Q E / Ctrl-Q X
верх/низ файла	Ctrl-Q R / Ctrl-Q C
начало/конец блока	Ctrl-Q B / Ctrl-Q K
последняя позиция курсора	Ctrl-Q P

Команды вставки и замены:

включение/выключение режима вставки	Ctrl-V или Ins
вставить/удалить строку	Ctrl-N / Ctrl-Y
удалить до конца строки	Ctrl-Q Y
удалить символ слева от курсора	Ctrl-H или Backspace
удалить символ в позиции курсора	Ctrl-G или Del
удалить слово справа от курсора	Ctrl-T

Команды обработки блоков текста:

отметить начало/конец блока	Ctrl-K B / Ctrl-K K
отметить одно слово	Ctrl-K T
копировать/удалить/переслать блок	Ctrl-K C / Ctrl-K Y / Ctrl-K V
скрыть/показать блок	Ctrl-K H
прочитать/записать блок с диска	Ctrl-K R / Ctrl-K W

Прочие команды:

сбросить операцию	Ctrl-U
автоотступ включить/выключить	Ctrl-Q I
префикс управляющего символа	Ctrl-P
поиск	Ctrl-Q F
поиск с заменой	Ctrl-Q A
поиск отметки места	Ctrl-Q N
выйти из редактора без сохранения	Ctrl-K D или Ctrl-K Q
повторить последний поиск	Ctrl-I
восстановить строку	Ctrl-Q L
сохранить и продолжать редактирование	Ctrl-K S или F2
установить отметку места	Ctrl-K N
табуляция	Ctrl-I или Tab
режим табуляции	Ctrl-Q T

ПРИЛОЖЕНИЕ 3. Коды и описание основных ошибок компилирования

1. Грубые ошибки:

Bad call of in-line function - неправильный вызов встроенной функции, взятой из макроопределения

Register allocation failure - слишком сложное выражение

2. Ошибки:

'XXXXXXXX' not an argument - в исходном файле идентификатор объявлен как аргумент функции, но отсутствует в списке аргументов функции

Ambiguous symbol 'XXXXXXXX' - имя поля используется более чем в одной структуре, но с другим смещением и типом

Argument # missing name - пропущено имя параметра в прототипе функции

Array size too large - объявленный массив слишком велик для памяти процессора

Assembler statement too long - встроенные операторы ассемблера не должны превышать 480 байтов

Bad configuration file - файл TURBOC.CFG содержит текст, не являющийся опцией. Опции в файле конфигурации должны начинаться с тире (-)

Compound statement missing - несоответствие количества открывающих и закрывающих скобок

Conflicting type modifiers - одновременное использование ключевых слов near и far для одного и того же указателя

Could not find file 'XXXXXXXX.XXX' - компилятор не может найти файл, заданный в командной строке

Division by zero - деление с нулевым делителем

Error writing output file - рабочий диск переполнен или неисправна дискета

Extra parameter in call to XXXXXXXX - обращение к указанной функции содержит слишком много аргументов

File name too long - имя файла, описанного в директиве #include, слишком длинное (более 64 символов)

Illegal charachters 'C'(0xXX) - компилятор обнаружил неверный символ в выводном файле. Шестнадцатеричный код символа дается в сообщении

Illegal pointer subtraction - попытка вычесть указатель из не указателя

Illegal use of pointer - указатели используются только в операциях сложения, вычитания, присваивания, сравнения, и адресных операциях (*) и (->)

In-line assembly not allowed - нельзя компилировать файл с ассемблерными вставками в Borland среде

Invalid pointer addition - попытка сложения двух указателей

Too much code defined in file - размер функции в текущем исходном файле превышает 64 Кбайта

Too much global data defined in file - объявленные глобальные переменные занимают более 64Кбайт памяти

Unablle to create output file 'XXXXXXXX.XXX' - рабочая дискета переполнена или защищена от записи

Unable to open include file 'XXXXXXXX.XXX' - компилятор не может найти файла с указанным именем

Unable to open input file 'XXXXXXXX.XXX' - компилятор не может найти исходного файла с указанным именем

3. Предупреждения:

Code has no effect - найден оператор, который не имеет смысла

Conversion may lose significant digits - потеря значащих разрядов при преобразовании типа long в тип int

Non portable pointer assignment - присваивание значения указателя не указателю или наоборот

Non portable pointer comparison - сравнение значения указателя и не указателя

Non-portable return type conversion - выражение в операторе return не того типа, который был в описании функции

Restarting compile using assembly - компилятор обнаружил оператор asm, но не задана опция командной строки -B или директива #pragma inline

Unknown assembler instruction - оператор ассемблера с неверным кодом операции

Void function may not return a value - текущая функция объявлена как void, но в операторе return обнаружено возвращаемое значение, которое игнорируется

Zero length structure - объявлена структура с нулевым размером

ПРИЛОЖЕНИЕ 4. Основные опции командной строки

1. Опции моделей памяти:
-mc, -mh, -ml, -mm, -ms, -mt компиляция с использованием модели памяти compact, huge, large, medium, small, tiny

2. Опции процессора:
-l компиляция с использованием команд расширенного набора процессора 80186
-a выравнивание элементов целого размера к границам машинного слова
-f87 использование команд процессора плавающей точки
-f использование эмулятора команд процессора 8087
-f- отмена режима вычислений с плавающей точкой

3. Опции исходного модуля:
-A создание ANSI-совместимого кода, когда все ключевые слова расширения Турбо Си - near, far, huge, cdecl, asm, pascal, interrupt, _es, _ds, _cs, _ss и регистровые псевдо-переменные _AX, _BX, _SX и т.д. игнорируются и могут использоваться как обычные идентификаторы
-C допускается вложенность комментариев
-i# распознавание только первых # символов идентификатора
-K интерпретация объявлений char как unsigned char

4. Опции кодов:
-d слияние одинаковых строк
-G оптимизация программы по быстродействию
-N генерация сообщения "Stack owerflow" о переполнении стека
-y включение номеров строк в объектный файл для отладчика
-O оптимизация программы по размеру
-p генерирование Паскаль-вызовов подпрограмм
-r использование регистровых переменных по умолчанию. При использовании фрагментов на ассемблере опция -r- позволяет выполнять программу на Турбо Си
-Y генерация стандартной рамки стека для отладчика

5. Опции ошибок:
-wsus запрещенное преобразование указателей
-wvoi функция типа void не должна возвращать значения
-wzst структура нулевой длины
-wrvl функция должна возвращать значение
-wamb в операциях должны быть скобки
-wpro вызов функции без прототипа
-wcln константа слишком длинна
-wcp смешение указателей signed и unsigned char

6. Опции управления компиляцией:
-b компиляция и вызов ассемблера для обработки ассемблерных строк
-c компиляция и ассемблирование .C и .ASM-файлов без компоновки
-ofilename компиляция исходного файла в заданный
-S компиляция исходного файла и выдача выходного файла .asm без ассемблирования

7. Опции компоновщика:
-efilename задает имя исполнимого файла программы
-M создание полной карты компоновки

8. Опции среды:
-I директория спецификация каталога включаемых файлов
-L директория спецификация каталога библиотечных файлов
-nxxx спецификация каталога для выходных файлов

СОДЕРЖАНИЕ

Учебное издание

Составители: Муравьев Геннадий Леонидович
Мухов Сергей Владимирович
Шуть Василий Николаевич

РАБОТА В ИНТЕГРИРОВАННОЙ СРЕДЕ BORLAND C++

Методические указания

к лабораторным работам по дисциплине "Конструирование программ и языки программирования" для студентов специальности "Искусственный интеллект"

Ответственный за выпуск: ***Г.Л. Муравьев***
Редактор: ***Т.В. Строкач***
Компьютерная вёрстка: ***Кармаш Е.Л.***
Компьютерный набор: ***Г.Л. Муравьев***
Корректор: ***Никитчик Е.В.***

Подписано в печать 30.01.2008 г. Формат 60x84 1/16
Бумага писч. Усл. п. л. 2,33. Уч. изд. л. 2,5. Тираж 80 экз. Заказ №121.
Отпечатано на ризографе УО "Брестский государственный технический университет"
224017, Брест, ул. Московская, 267