Приложение к РПД

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ

для проведения текущей и промежуточной аттестации

по учебной дисциплине (модулю)

«Операционные системы»

для направления подготовки/специальности
09.03.03 – Прикладная информатика

Профиль: «Искусственный интеллект и прикладная информатика в цифровой экономике"

1. Описание показателей (дескрипторов) и критериев оценивания компетенций на различных этапах их формирования

Контроль качества освоения дисциплины (модуля) включает в себя текущий контроль успеваемости и промежуточную аттестацию. Текущий контроль успеваемости и промежуточная аттестация обучающихся проводятся в целях установления соответствия достижений обучающихся поэтапным требованиям образовательной программы к результатам обучения и формирования компетенций.

Компетенции	Показатели* (дескрипторы)	Критерии в соответствии с уровнем освоения ОП			Оценочное средство (промежуточная аттестация)
		пороговый (удовлетворительно) 55-69 баллов	стандартный (хорошо) 70-84 балла	эталонный (отлично) 85-100 баллов	
ОПК-2	Знать	Знает современные информационные технологии и программные средства	Знает современные информационные технологии и программные средства, в том числе отечественного производства	Знает современные информационные технологии и программные средства, в том числе отечественного производства при решении задач профессиональной деятельности	Экзаменационные билеты
	Уметь	Умеет проводить анализ возможностей сквозных цифровых технологий	Умеет проводить анализ возможностей и принципов работы сквозных цифровых технологий	Умеет проводить анализ возможностей и принципов работы сквозных цифровых технологий и отраслевых решений на их основе (в том числе отечественного производства), с целью	Экзаменационные билеты

				применения для решения профессиональных задач и внедрения в прикладные сферы деятельность	
	Владеть	Владеет навыками решения профессиональных задач	Владеет навыками решения профессиональных задач с помощью современных информационных технологий и программных средств	Владеет навыками решения профессиональных задач с помощью современных информационных технологий и программных средств, в том числе отечественного производства.	Экзаменационные билеты
ОПК-5	Знать	Знает основы системного администрирования	Знает основы системного администрирования, администрирования СУБД	Знает основы системного администрирования, администрирования СУБД, современные стандарты информационного взаимодействия систем.	Экзаменационные билеты
	Уметь	Умеет выполнять параметрическую настройку информационных систем	Умеет выполнять параметрическую настройку информационных и автоматизированных систем	Умеет выполнять параметрическую настройку информационных и автоматизированных систем	Экзаменационные билеты
	Владеть	Владеет навыками инсталляции программного обеспечения информационных систем	Владеет навыками инсталляции программного и аппаратного обеспечения информационных систем	Владеет навыками инсталляции программного и аппаратного обеспечения информационных и	Экзаменационные билеты

				автоматизированных систем	
ОПК-8	Знать	Знает основные технологии создания информационных систем	Знает основные технологии создания и внедрения информационных систем	Знает основные технологии создания и внедрения информационных систем, стандарты управления жизненным циклом информационной системы.	Экзаменационные билеты
	Уметь	Умеет осуществлять организационное обеспечение выполнения работ на некоторых стадиях жизненного цикла информационной системы	Умеет осуществлять организационное обеспечение выполнения работ на всех стадиях жизненного цикла информационной системы	Умеет осуществлять организационное обеспечение выполнения работ на всех стадиях и в процессах жизненного цикла информационной системы	Экзаменационные билеты
	Владеть	Владеет навыками составления плановой и отчетной документации	Владеет навыками составления плановой и отчетной документации по управлению проектами создания информационных систем	Владеет навыками составления плановой и отчетной документации по управлению проектами создания информационных систем на стадиях жизненного цикла.	Экзаменационные билеты
ПК-10	Знать	Знает методы и модели организации ИТинфраструктуры; виды угроз и меры по обеспечению	Знает методы и модели организации ИТинфраструктуры; виды угроз и меры по обеспечению	Знает методы и модели организации ИТинфраструктуры; виды угроз и меры по обеспечению	Экзаменационные билеты

		информационной безопасности ИС;	информационной безопасности ИС; основы конфигурационного управления;	информационной безопасности ИС; основы конфигурационного управления; основы управления изменениями.	
	Уметь	Умеет применять методы и модели организации ИТ инфраструктуры	Умеет применять методы и модели организации ИТ инфраструктуры; виды угроз и меры по обеспечению информационной безопасности ИС;	Умеет применять методы и модели организации ИТ инфраструктуры; виды угроз и меры по обеспечению информационной безопасности ИС; работать с системой контроля версий;	Экзаменационные билеты
	Владеть	Владеет навыками организации ИТ- инфраструктуры и управления информационной безопасностью	Владеет навыками организации ИТ- инфраструктуры и управления информационной безопасностью, в т.ч., обеспечения и контроля соответствия технических, программных и коммуникационных средств для функционирования ИС	Владеет навыками организации ИТ- инфраструктуры и управления информационной безопасностью, в т.ч., обеспечения и контроля соответствия технических, программных и коммуникационных средств для функционирования ИС, разграничение прав доступа к ИС.	Экзаменационные билеты

2.2. Критерии и шкалы оценивания результатов обучения при проведении текущего контроля успеваемости

Текущий контроль предназначен для проверки хода и качества формирования компетенций, стимулирования учебной работы обучаемых и совершенствования методики освоения новых знаний. Он обеспечивается проведением семинаров, оцениванием заданий, проверкой конспектов лекций, выполнением индивидуальных и творческих заданий, периодическим опросом обучающихся на занятиях. Контролируемые разделы (темы) дисциплины, компетенции и оценочные средства представлены в таблице.

№ п/п	Контролируемые разделы (темы) дисциплины*	Код контролируемой компетенции (или ее части)	Наименование оценочного средства**
1.	Назначение и функции ОС. Классификация ОС.	ОПК-2	Собеседование (очная, заочная форма обучения)
2.	Планирование в системах с одним процессором, многопроцессорное планирование, планирование реального времени	ОПК-5	Собеседование (очная, заочная форма обучения)
3.	Управление памятью в распространенных ОС.	ОПК-8	Собеседование (очная, заочная форма обучения)
4.	Операционные системы реального времени	ПК-10	Собеседование (очная, заочная форма обучения)

Критерии и шкала оценивания собеседования (очная, заочная форма обучения)

Оценка	*Критерий оценки*
«отлично»	1. полно раскрыл содержание материала в объеме, предусмотренном программой; 2. изложил материал грамотным языком, точно используя математическую терминологию и символику, в определенной логической последовательности; 3. показал умение иллюстрировать теорию конкретными примерами, применять ее в новой ситуации при выполнении практического задания; 4. продемонстрировал знание теории ранее изученных сопутствующих тем, сформированность и устойчивость используемых при ответе умений и навыков;

	5. отвечал самостоятельно, без наводящих вопросов преподавателя; 6. возможны одна – две неточности при освещении второстепенных вопросов или в выкладках, которые студент легко исправил после замечания преподавателя.
«хорошо»	Ответ оценивается оценкой «*хорошо*», если удовлетворяет в основном требованиям на оценку «*отлично*», но при этом имеет некоторые из недостатков: 1. в изложении допущены небольшие пробелы, не исказившее математическое содержание ответа; 2. допущены один – два недочета при освещении основного содержания ответа, исправленные после замечания преподавателя; 3. допущены ошибка или более двух недочетов при освещении второстепенных вопросов или в выкладках, легко исправленные после замечания преподавателя.
«удовлетворительно»	1. неполно раскрыто содержание материала (содержание изложено фрагментарно, не всегда последовательно), но показано общее понимание вопроса и продемонстрированы умения, достаточные для усвоения программного материала 2. имелись затруднения или допущены ошибки в определении терминологии, выкладках, исправленные после нескольких наводящих вопросов преподавателя; 3. студент не справился с применением теории в новой ситуации при выполнении практического задания, но выполнил задания обязательного уровня сложности по данной теме; 4. при достаточном знании теоретического материала выявлена недостаточная сформированность основных умений и навыков.
«неудовлетворительно»	1. не раскрыто основное содержание учебного материала; 2. обнаружено незнание обучающимся большей или наиболее важной части учебного материала; 3. допущены ошибки в определении понятий, чертежах или графиках, в выкладках, которые не исправлены после нескольких наводящих вопросов преподавателя.

Критерии и шкала оценивания защиты лабораторных работ

На первом лабораторном занятии обучающемуся выдается перечень заданий для выполнения лабораторных работ.

Оценка	*Критерий оценки*
«Зачтено»	1. работа выполнена полностью; 2. в логических рассуждениях и обосновании решения нет пробелов и ошибок; 3. в решении нет математических ошибок (возможны некоторые неточности, которые не являются следствием незнания или непонимания учебного материала).

«не зачтено»	4. допущены существенные ошибки, показавшие, что обучающийся не обладает обязательными умениями по данной теме в полной мере.

2.3. Критерии и шкалы оценивания результатов обучения при проведении промежуточной аттестации

Промежуточная аттестация предназначена для определения уровня освоения объема учебной дисциплины за семестр. Для оценивания результатов обучения при проведении промежуточной аттестации используется 4-балльная шкала: «отлично», «хорошо», «удовлетворительно», «не удовлетворительно».

Шкала оценивания	Критерии	Уровень освоения компетенций
Отлично	Обучающийся правильно ответил на теоретические вопросы. Показал отличные знания в рамках учебного материала. Правильно выполнил практические задания. Ответил на все дополнительные вопросы. Ответ студента соответствует критериям: дано определение, формулировка рассматриваемого преобразования сигналов, процесса, закона, характеристики системы связи, названы условия их реализации, границы применения, свойства; приведены примеры применения, частные случаи; записаны соответствующие формулы, уравнения, определены в них обозначения всех физических величин; в задании выполнены необходимые преобразования уравнений и получен правильный ответ; приведены графические зависимости между описываемыми параметрами;	Эталонный
Хорошо	Обучающийся с небольшими неточностями ответил на теоретические вопросы. Показал хорошие знания в рамках учебного материала. С небольшими неточностями выполнил практические задания. Ответил на большинство дополнительных вопросов. В ответе обучающегося имеются небольшие неточности или выше названные критерии выполнены не полностью	Стандартный
Удовлетвори-тельно	Обучающийся с существенными неточностями ответил на теоретические вопросы. Показал удовлетворительные знания в рамках учебного материала. С существенными неточностями выполнил практические задания. Допустил много неточностей при ответе на дополнительные вопросы В ответе обучающегося имеются существенные неточности, ошибки или выше названные критерии выполнены частично	Пороговый

Не-удовлетворительно	Обучающийся при ответе на теоретические вопросы и при выполнении практических заданий продемонстрировал недостаточный уровень знаний и умений. При ответах на дополнительные вопросы было допущено множество неправильных ответов. В ответе обучающегося имеются грубые неточности и ошибки или большая часть выше названных критериев не выполнена	Компетенции не сформированы

3. Типовые задания или иные материалы, необходимые для оценки знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций в процессе освоения образовательной программы

3.1. Оценочные средства текущего контроля успеваемости

Вопросы к собеседованию (раздел 1, 2, 3,4 Блок 1 «Знать»):

1. Назначение и классификация операционных систем.
2. Процессы и потоки.
3. Понятие WinAPI и его применение при разработке прикладных программ.
4. Планирование в системах с одним процессором.
5. Планирование в многопроцессорных системах.
6. Переключение потоков и краткосрочное планирование.
7. Приоритеты процессов и потоков.
8. Стратегии и виды планирования.
9. Межпроцессное взаимодействие.
10. Синхронизация, механизмы синхронизации.
11. Страничная адресация и работа файла подкачки.
12. Виртуальное адресное пространство процесса.
13. Управление памятью в системах Windows.
14. Библиотеки динамической компоновки DLL.
15. Явное и неявное связывание библиотек динамической компоновки.

3.2. Оценочные средства промежуточной аттестации

В данном разделе представляются теоретические вопросы (для оценки знаний), типовые задания (для оценки умений), типовые практические задания (для оценки навыков и (или) опыта деятельности).

Варианты лабораторных работ (блок 2 «уметь», блок 3 «владеть»)

Лабораторные работы № 1 – 4

Лабораторная работа 1. *Каркасное приложение Windows*

После создания исходного текста приложения Win32 необходимо выполнить компиляцию и компоновку программы для получения исполняемого модуля (файла .EXE или .DLL). Конкретная последовательность действий зависит от типа разрабатываемого приложения (одно- или многопоточное, .EXE или .DLL), вида компоновки (статическая или динамическая), используемого компилятора (C++ Builder, Visual C++ и т.п.) и используемой технологии программирования (чистый Win32 API, библиотеки классов типа VCL, MFC и т.п.). Следует отметить, что системы быстрой разработки приложений (в частности, Visual C++ и C++ Builder), как правило, сильно автоматизируют процесс компиляции программы, скрывая от программиста отдельные его стадии. Кроме того, разработка приложений с использованием указанных средств, предполагает использование соответствующих библиотек, а, значит, и собственных приемов создания программ. Так, например, нетрудно заметить существенные различия в исходных текстах приложений, написанных на C++ Builder или на Visual C++. Между тем, все современные системы разработки поддерживают стандарт Win32 API и позволяют создавать программы, работающие только с функциями Windows, без использования библиотек классов. Такие программы зачастую проще и удобнее компилировать с помощью компиляторов командной строки, чем из IDE, хотя не исключено и обратное.

Для создания каркасного приложения Windows в среде разработки Microsoft Visual Studio 2010 необходимо создать соответствующий проект, как показано на приведенных ниже снимках экранов. Созданный проект может быть затем откомпилирован и собран в исполняемый файл средствами Visual Studio.

Для создания проекта необходимо выбрать соответствующий пункт в меню «Файл» (или нажать комбинацию клавиш Ctrl+Shift+N), в появившемся диалоге выбрать тип проекта «Проект Win32» как показано на рис. 1. Кроме того, необходимо также указать название проекта и указать место его расположения в соответствующих полях.

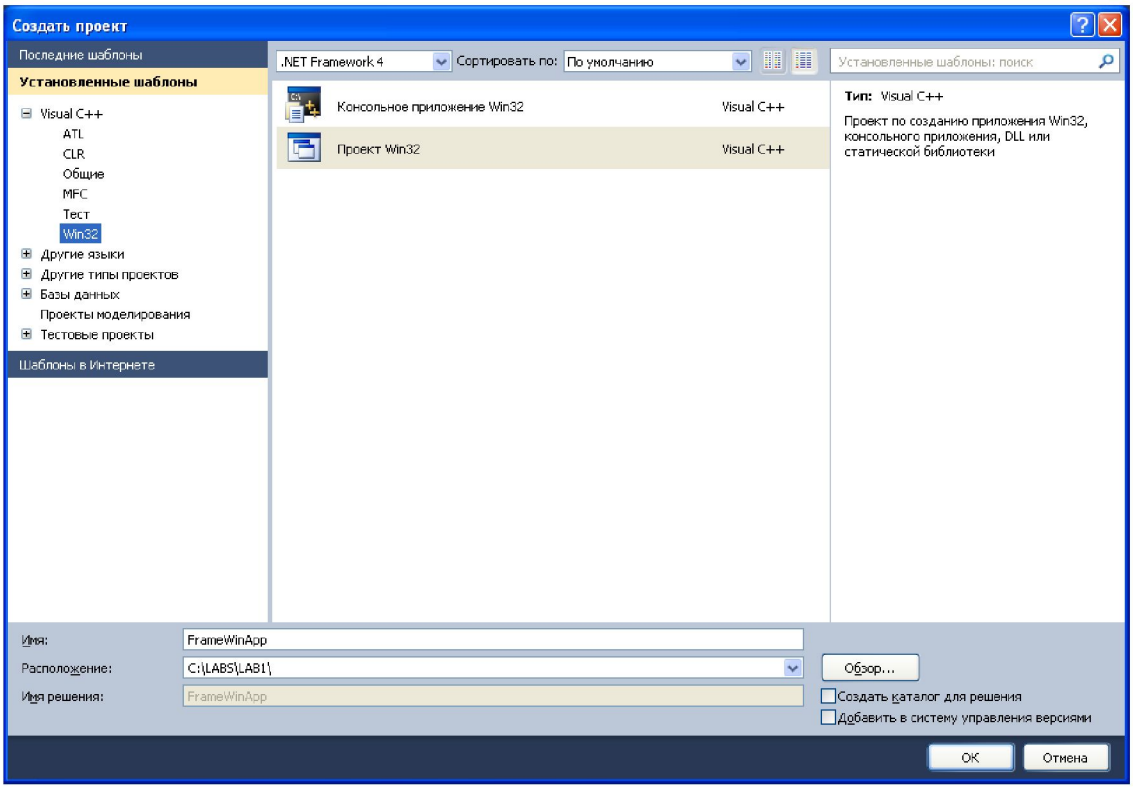

Рис. 1 – Выбор типа проекта

В появившемся окне Мастера приложений Win32 необходимо установить параметры, как показано на рис. 2.

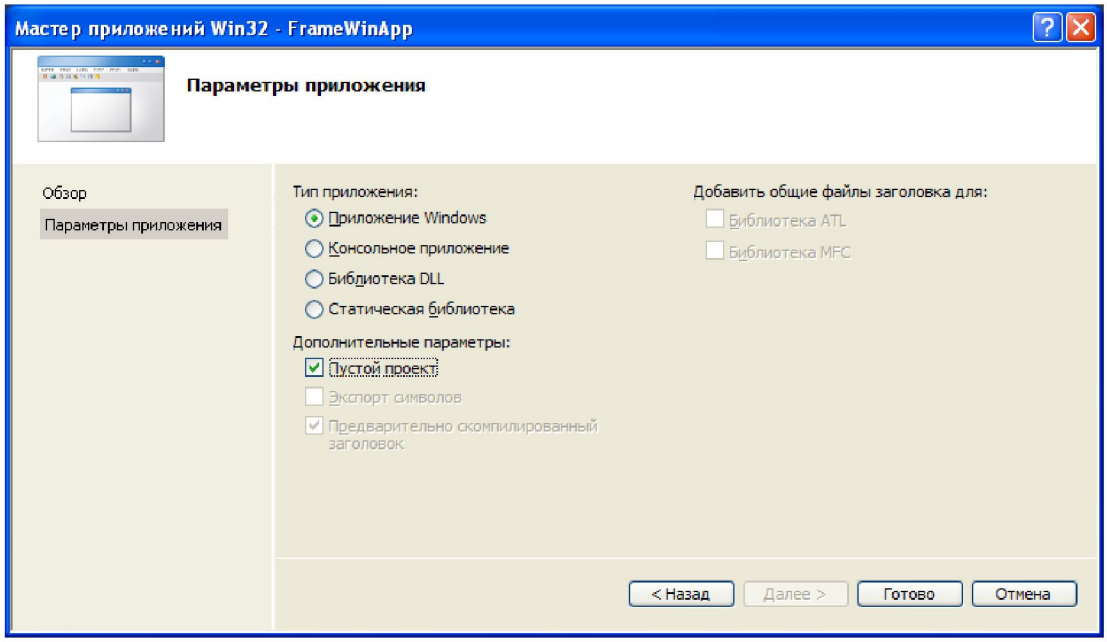

Рис. 2 – Установка параметров проекта.

После выполнения этих действий будет создан пустой проект (рис. 3), в который следует добавить файл, содержащий исходный код каркасного приложения.

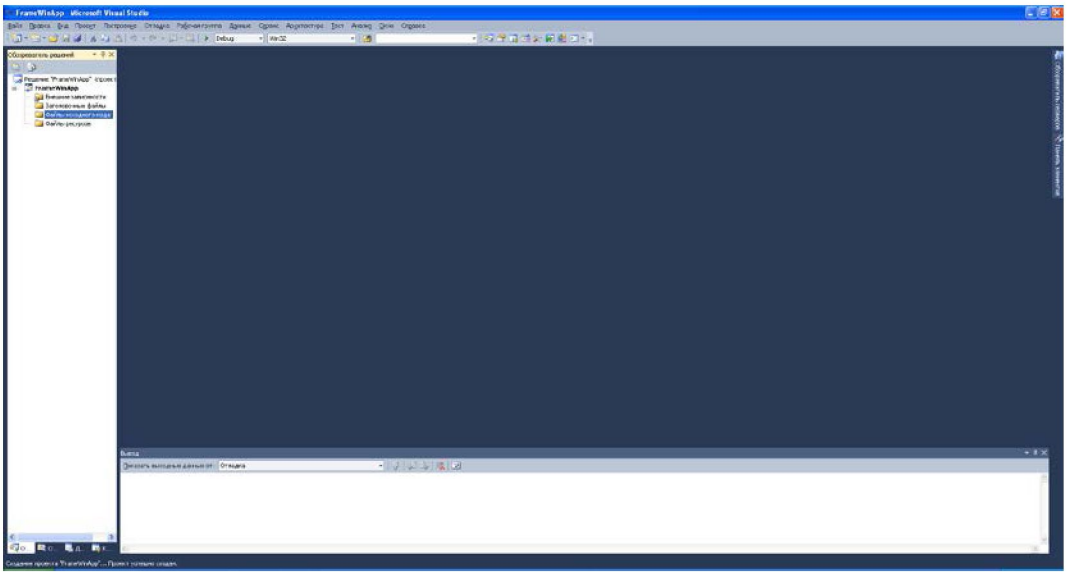

Рис. 3 – Окно созданного пустого проекта

Для добавления файла с исходным кодом следует выбрать соответствующий пункт в меню «Проект» (или нажать комбинацию клавиш Alt+Shift+A), затем в появившемся диалоге выбрать нужный файл.

После добавления файла с исходным кодом каркасного приложения (рис. 4) можно будет выполнить сборку и запуск приложения, нажав клавишу F5 или выбрав соответствующий пункт в меню «Отладка».

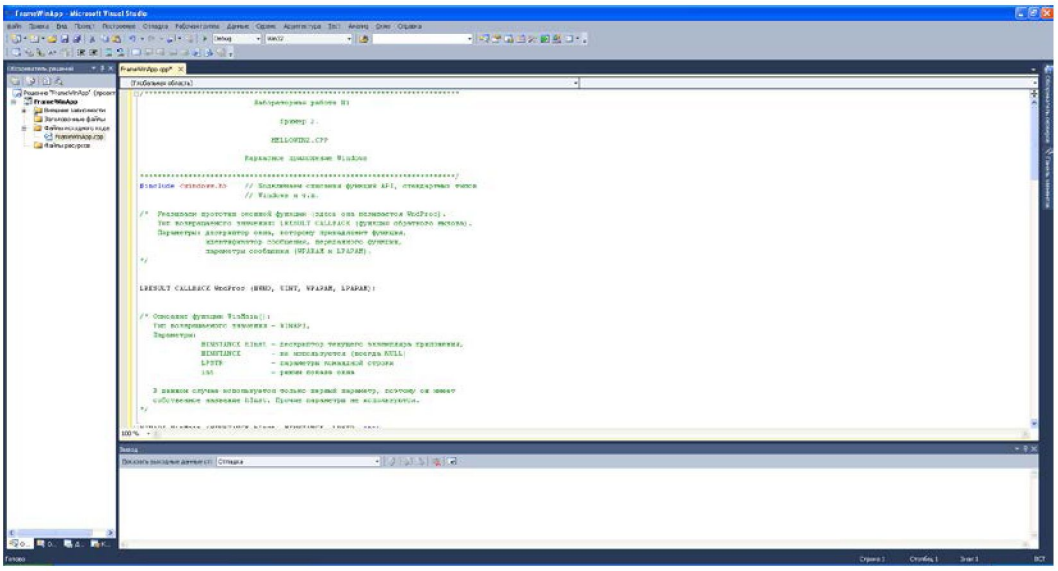

Рис. 4 – Проект готов к сборке и запуску

Выполнить модификацию каркасного приложения следующим образом:

1. Модифицировать цикл обработки сообщений для перехвата клавиши F1.
2. Модифицировать оконную функцию для вывода запроса на выход из программы.
3. Модифицировать оконную функцию для перехвата сообщения WM_RESIZE.
4. Модифицировать оконную функцию для перехвата сообщения WM_MAXIMIZE.
5. Модифицировать класс окна для использования нестандартных курсоров.

Лабораторная работа 2. *Процессы и потоки*

После создания исходного текста приложения Win32 необходимо выполнить компиляцию и компоновку программы для получения исполняемого модуля (файла .EXE или .DLL). Конкретная последовательность действий зависит от типа разрабатываемого приложения (одно- или многопоточное, .EXE или .DLL), вида компоновки (статическая или динамическая), используемого компилятора (C++ Builder, Visual C++ и т.п.) и используемой технологии программирования (чистый Win32 API, библиотеки классов типа VCL, MFC и т.п.). Следует отметить, что системы быстрой разработки приложений (в частности, Visual C++ и C++ Builder), как правило, сильно автоматизируют процесс компиляции программы, скрывая от программиста отдельные его стадии. Кроме того, разработка приложений с использованием указанных средств, предполагает использование соответствующих библиотек, а, значит, и собственных приемов создания программ. Так, например, нетрудно заметить существенные различия в исходных текстах приложений, написанных на C++ Builder или на Visual C++. Между тем, все современные системы разработки поддерживают стандарт Win32 API и позволяют создавать программы, работающие только с функциями Windows, без использования библиотек классов. Такие программы зачастую проще и удобнее компилировать с помощью компиляторов командной строки, чем из IDE, хотя не исключено и обратное.

Для создания каркасного приложения Windows в среде разработки Microsoft Visual Studio 2010 необходимо создать соответствующий проект, как показано на приведенных ниже снимках экранов. Созданный проект может быть затем откомпилирован и собран в исполняемый файл средствами Visual Studio.

Для создания проекта необходимо выбрать соответствующий пункт в меню «Файл» (или нажать комбинацию клавиш Ctrl+Shift+N), в появившемся диалоге выбрать тип проекта «Проект Win32» как показано на рис. 1. Кроме того, необходимо также указать название проекта и указать место его расположения в соответствующих полях.

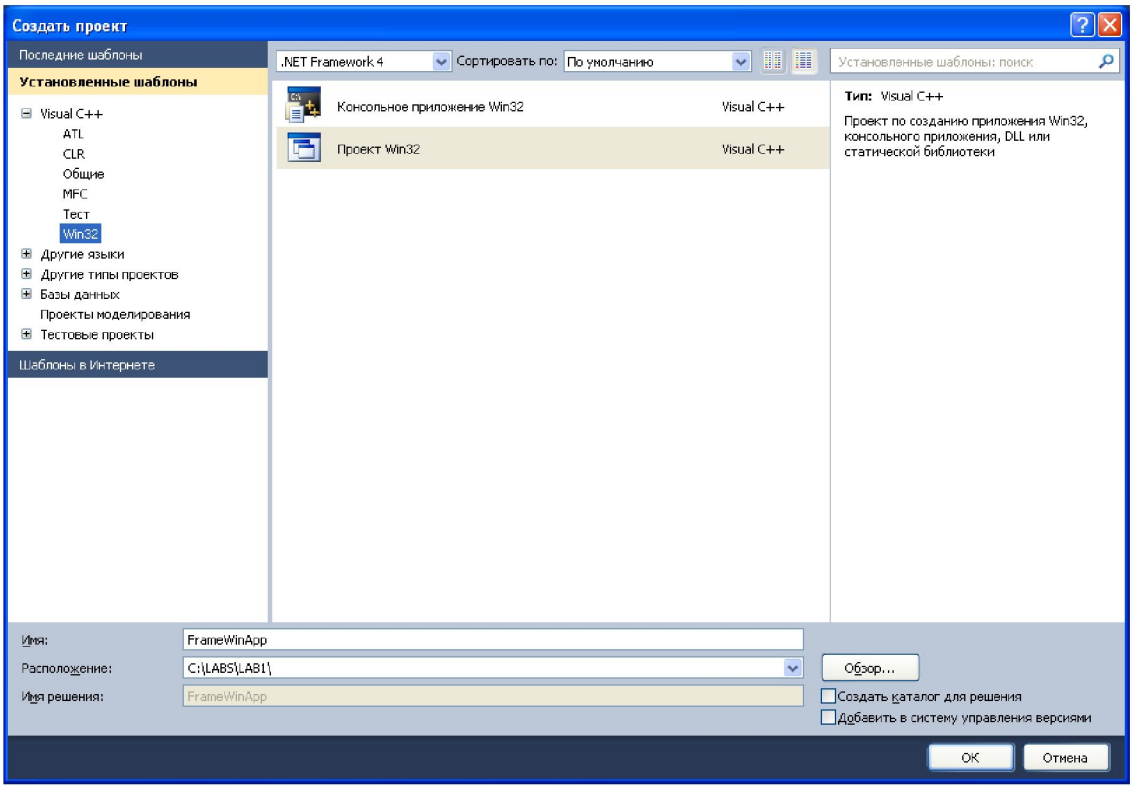

Рис. 1 – Выбор типа проекта

В появившемся окне Мастера приложений Win32 необходимо установить параметры, как показано на рис. 2.

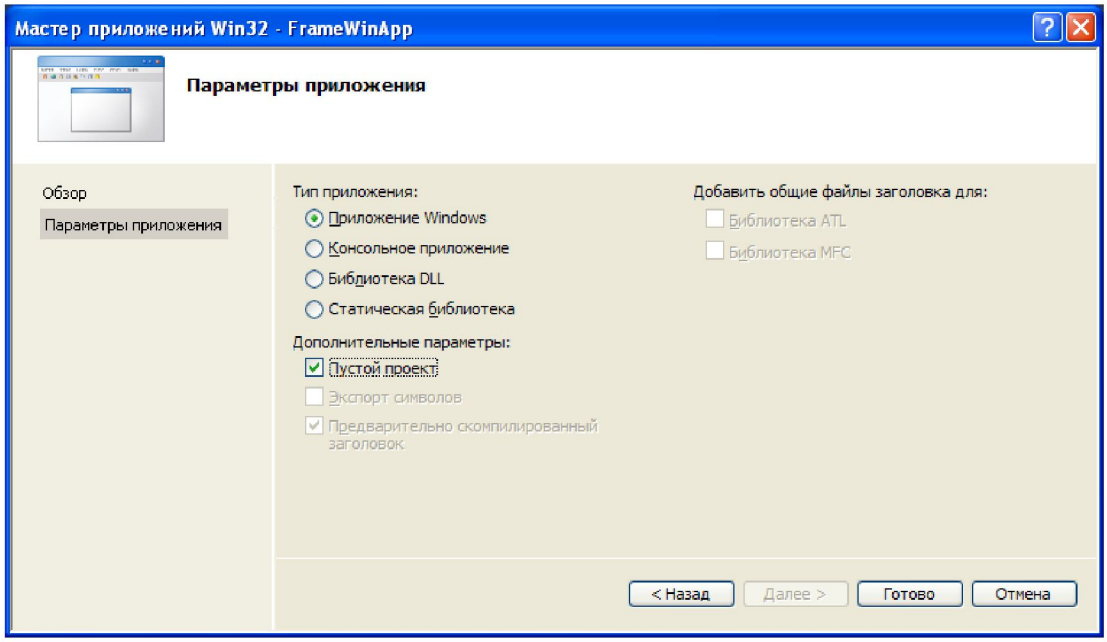

Рис. 2 – Установка параметров проекта.

После выполнения этих действий будет создан пустой проект (рис. 3), в который следует добавить файл, содержащий исходный код каркасного приложения.

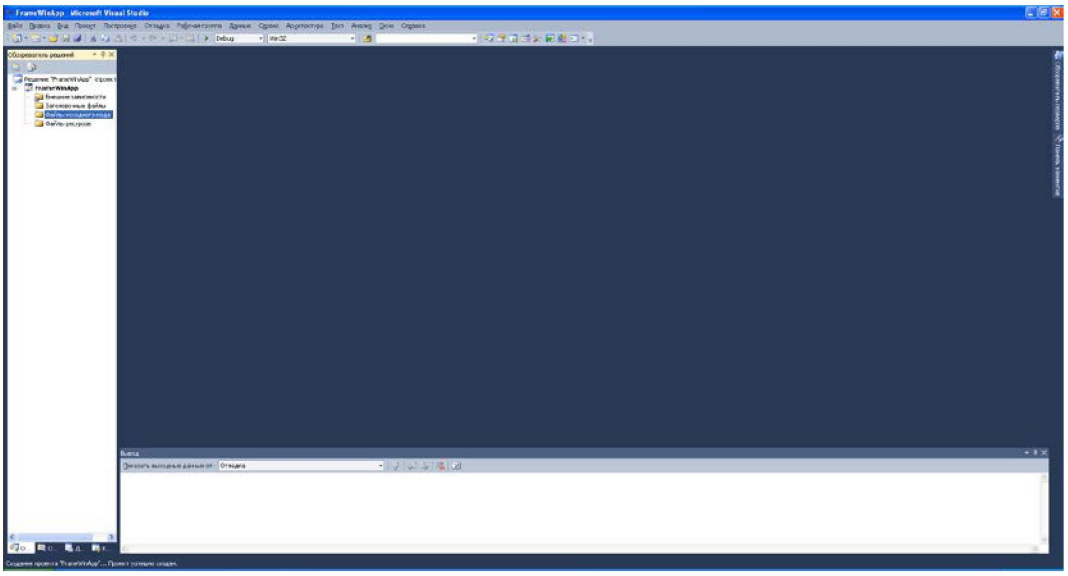

Рис. 3 – Окно созданного пустого проекта

Для добавления файла с исходным кодом следует выбрать соответствующий пункт в меню «Проект» (или нажать комбинацию клавиш Alt+Shift+A), затем в появившемся диалоге выбрать нужный файл.

После добавления файла с исходным кодом каркасного приложения (рис. 4) можно будет выполнить сборку и запуск приложения, нажав клавишу F5 или выбрав соответствующий пункт в меню «Отладка».

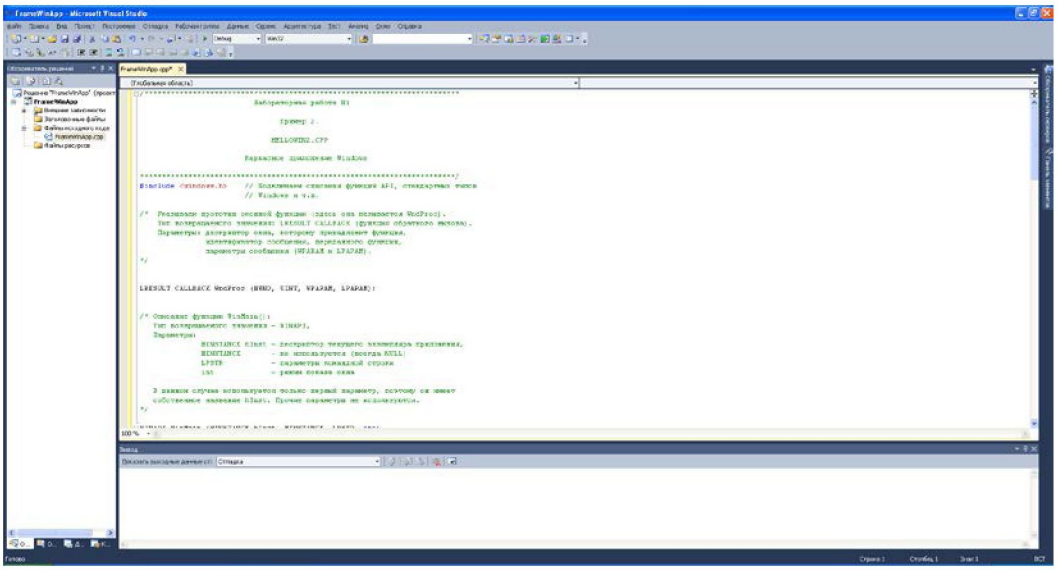

Рис. 4 – Проект готов к сборке и запуску

Выполнить модификацию приложений следующим образом:

1. Добавить вывод информации о дескрипторах потоков.
2. Добавить возможность увеличения/уменьшения количества рабочих потоков;
3. Добавить возможность создания нового процесса;
4. Добавить возможность управления приоритетами потоков;
5. Добавить возможность перевода потоков в состояния SUSPEND;

Лабораторная работа 3. *Библиотеки динамической компоновки*

Для создания демонстрационного приложения, использующего динамическую библиотеку DLL, в среде разработки Microsoft Visual Studio 2010 необходимо создать соответствующее решение (solution), включающее в себя два проекта: проект собственно DLL-библиотеки и проект вызывающего приложения, как показано на приведенных ниже снимках экранов. Проекты, составляющие решение, могут быть откомпилированы и собраны в соответствующие исполняемые файлы средствами Visual Studio.

Для создания решения необходимо выбрать соответствующий пункт в меню «Файл» (или нажать комбинацию клавиш Ctrl+Shift+N), в появившемся диалоге выбрать тип проекта «Проект Win32» как показано на рис. 1. Кроме того, необходимо также указать название проекта библиотеки DLL, место его расположения и наименование решения, в которое входит проект, в соответствующих полях.

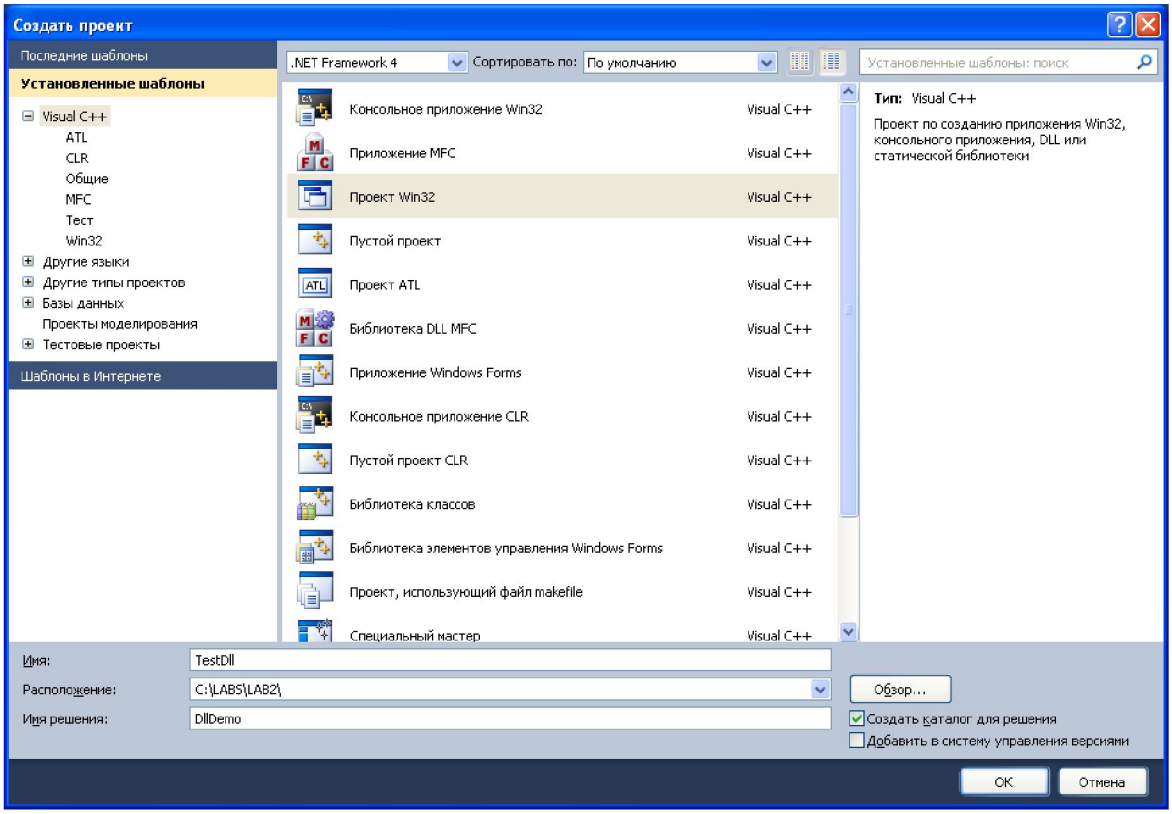

Рис. 1 – Выбор типа проекта для библиотеки DLL

В появившемся окне Мастера приложений Win32 необходимо установить параметры, как показано на рис. 2.

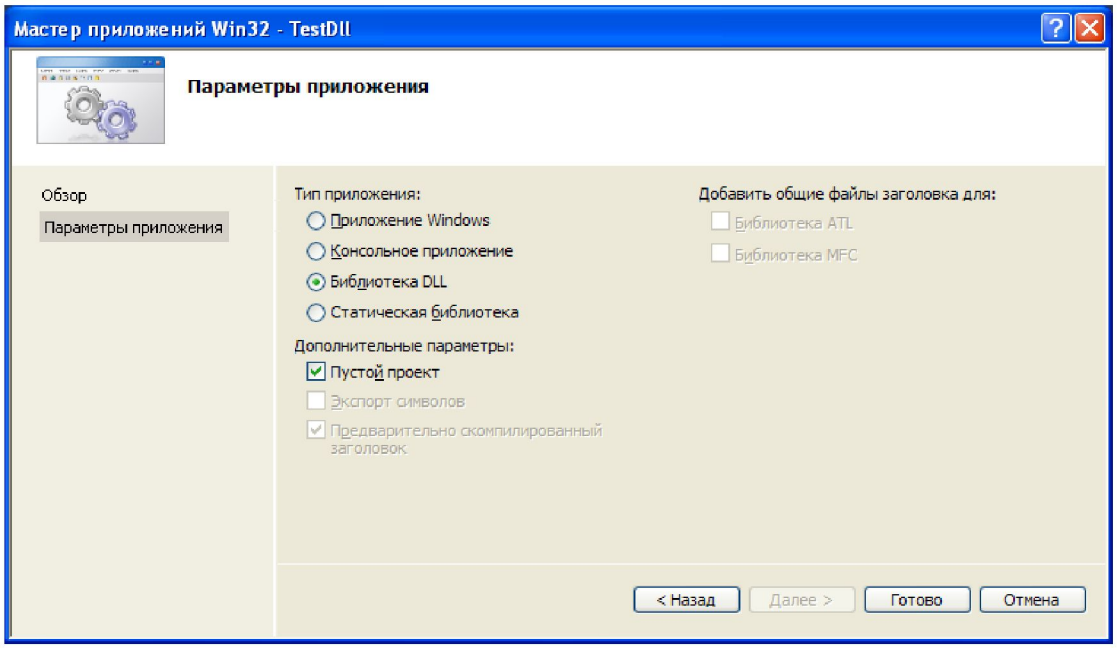

Рис. 2 – Установка параметров проекта для библиотеки DLL

После выполнения этих действий будет создан пустой проект, в который следует добавить файл, содержащий исходный код демонстрационной библиотеки DLL, либо создать такой файл непосредственно в проекте и написать там исходный код библиотеки DLL.

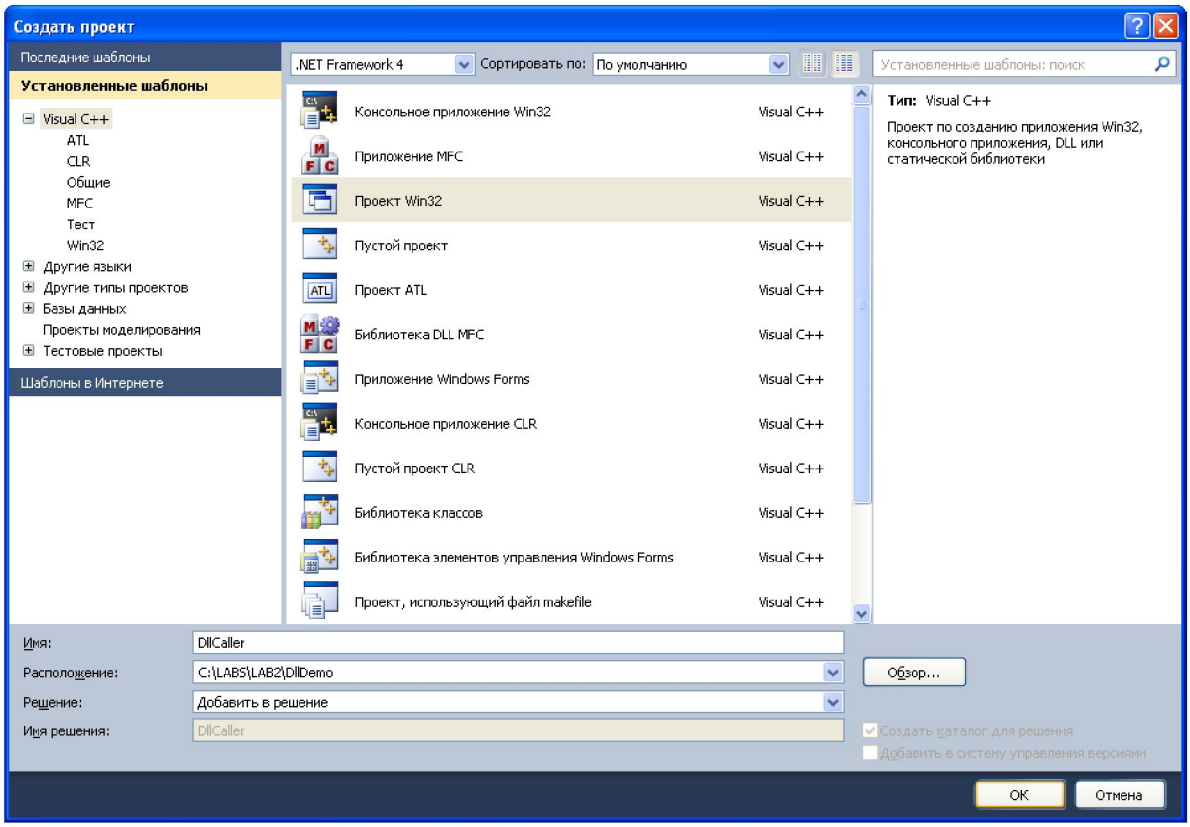

Рис. 3 – Выбор типа проекта для приложения, вызывающего DLL

Для создания проекта демонстрационного приложения, использующего демонстрационную библиотеку DLL, и добавления его в существующее решение следует выбрать соответствующий пункт в меню «Файл» (или нажать комбинацию клавиш Ctrl+Shift+N). В появившемся диалоге создания проекта следует указать имя проекта и его принадлежность к ранее созданному решению, как показано на рис. 3.

В появившемся окне Мастера приложений Win32 необходимо установить параметры, как показано на рис. 4. Далее необходимо добавить в созданный проект файл, содержащий исходный код приложения, либо создать такой файл непосредственно в проекте.

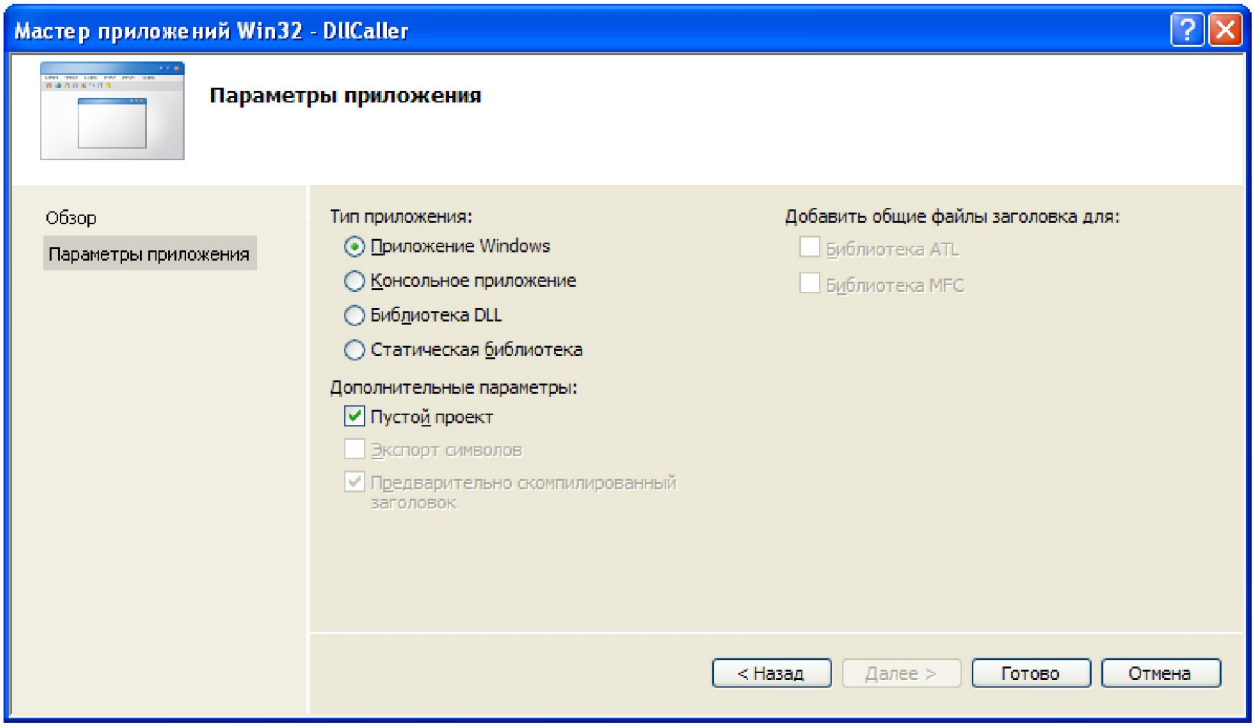

Рис. 4 – Установка параметров проекта для приложения, вызывающего DLL

После выполнения всех указанных действий будет создано решение (solution), содержащее два проекта – собственно DLL и вызывающего приложения. Далее необходимо установить порядок сборки проектов в решении и указать проект, который будет запускаться на выполнение после сборки (очевидно, что это должен быть проект вызывающего приложения). Для этого следует щелкнуть правой кнопкой на имени проекта вызывающего приложения и выбрать в появившемся меню пункт «Назначить запускаемым проектом». Также в пункте меню «Порядок построения объектов» можно указать порядок, в котором будет выполняться сборка проектов. Рекомендуется вначале выполнять сборку проекта DLL, а затем сборку проекта вызывающего приложения. Типичный вид готового к сборке решения показан на рис. 5.

После настройки можно выполнить собственно сборку и запуск демонстрационного приложения, нажав клавишу F5 или выбрав соответствующий пункт в меню.

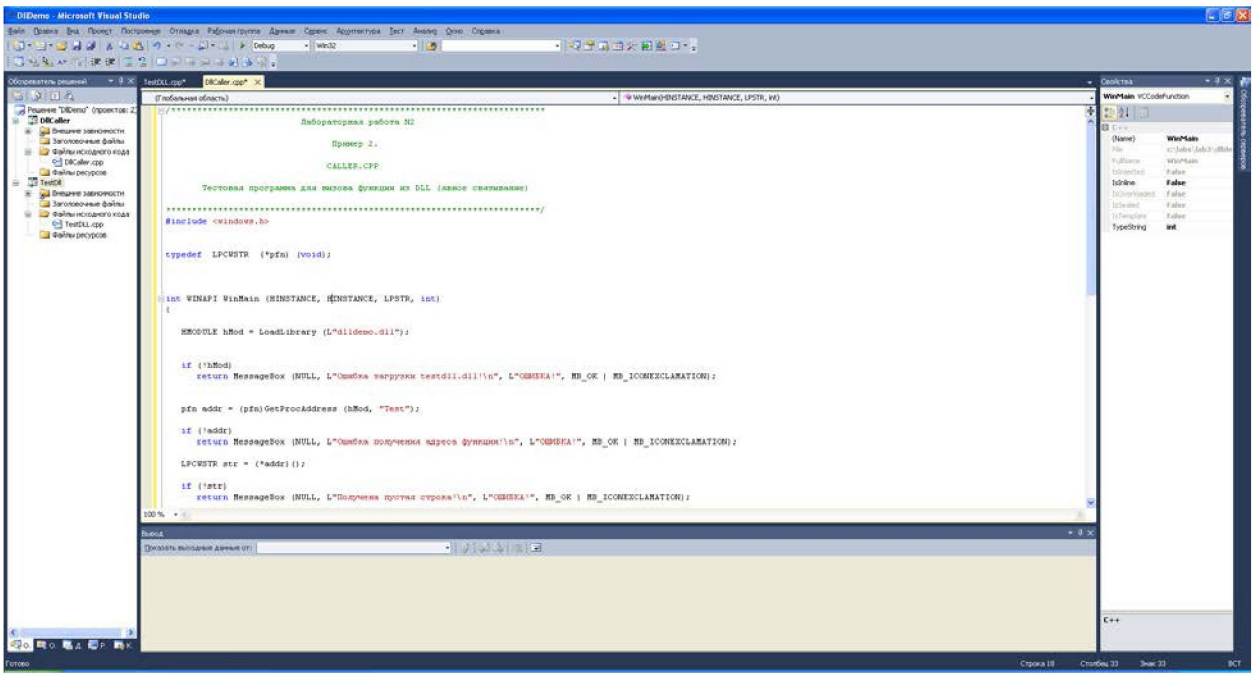

Рис. 5 – Решение настроено для сборки

Сборка приложения и DLL, демонстрирующих неявное связывание, производится аналогично сборке демонстрационных приложения и DLL для явного связывания за исключением важного нюанса. Так как загрузка библиотеки DLL должна производиться автоматически при запуске приложения (в отличие от явного связывания), то вызывающее приложение должно иметь информацию о функциях, экспортируемых из DLL. Для этого необходимо в проекте вызывающего приложения поставить ссылку на проект библиотеки DLL как показано на рис. 6, что позволит использовать информацию об экспортируемых из DLL функциях при сборке вызывающего приложения и корректно разрешить внешние ссылки на функции DLL.

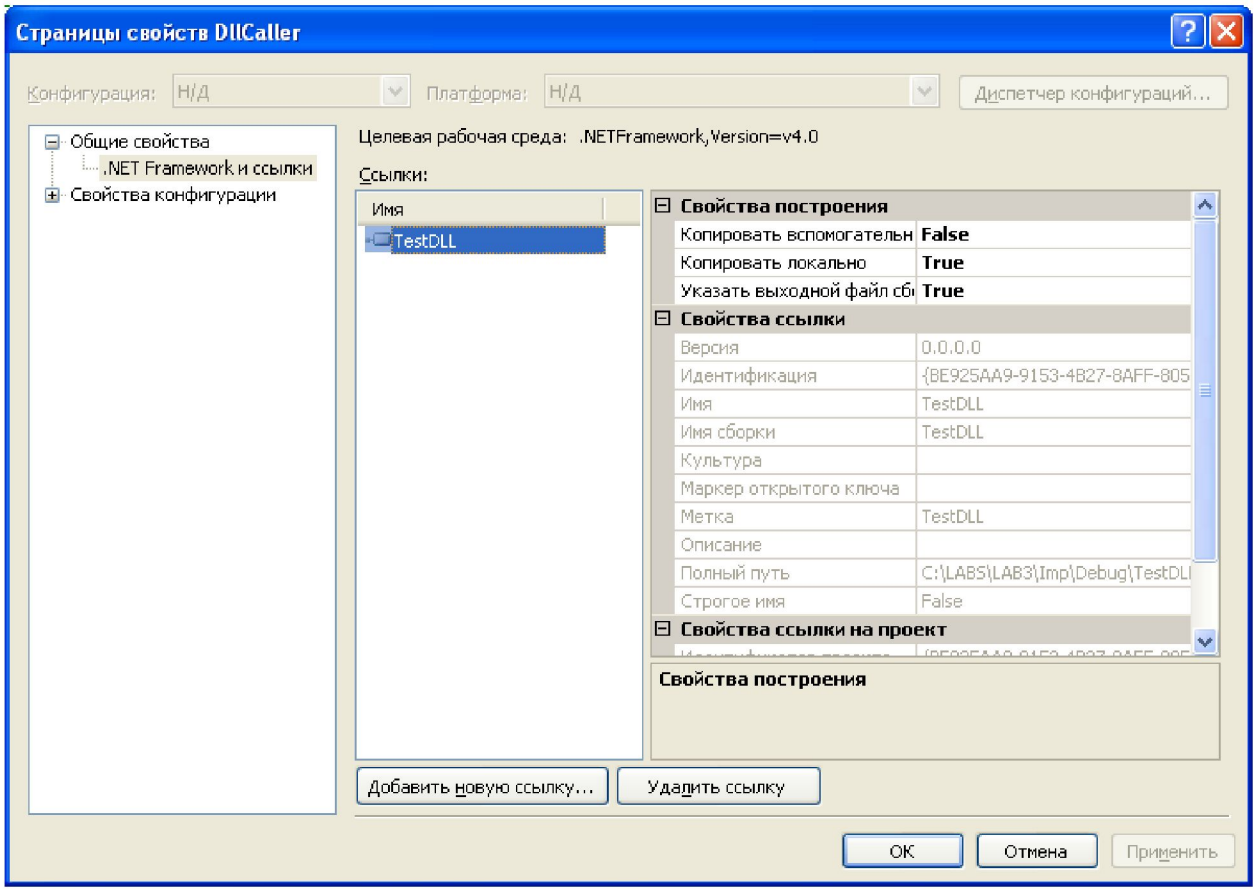

Рис. 6 – Ссылка на DLL в приложении с явным связыванием

Лабораторная работа 4. *Драйверы режима ядра*

В силу того, что драйверы режима ядра представляют собой особый вид программ, сборка драйвера выполняется специальным образом, без использования среды разработки Visual Studio.

Для использования в коде драйвера структур данных и функций, применяемых в ядре Windows, а также для корректной компоновки с внешними библиотеками необходимо осуществлять разработку и сборку драйвера с использованием пакета Windows Driver Kit (WDK). Данный пакет доступен для скачивания с сайта компании Microsoft.

После установки пакета WDK разработчику доступны т.н. build environments – среды для сборки, которые подразделяются по типу процессорной архитектуры, под которую выполняется сборка драйвера: x86, x64 и ia64, а также по версии Windows, для которой разрабатывается драйвер. Следует также отметить, что предусмотрена сборка драйверов отдельно для отладочных (checked) и релизных (free) версий Windows. Например, среда Windows 7 x86 Free Build Environment позволяет выполнить сборку драйвера для 32-х разрядной релизной версии Windows 7, а среда Windows 2008 x64 Checked Build Environment – для 64-хразрядной отладочной версии Windows Server 2008. Выбор среды для сборки драйвера после установки пакета WDK осуществляется выбором из меню «Пуск»: Win-

dows Driver Kits -> WDK <номер версии пакета> -> Build Environments -> <версия Windows> -> <нужный тип среды>, например: Пуск → Windows Driver Kits → WDK 7600.16385.1 → Build Environments → Windows 7 → x86 Checked Build Environment.

При выборе соответствующей среды сборки запускается сеанс командной строки, в которой определенным образом настроены переменные окружения, в основном – пути к различным библиотекам и инструментам WDK, что позволяет без дополнительных настроек применять различные средства командной строки: компилятор C/C++, компоновщик, утилиту Build и другие средства разработки, входящие в состав WDK.

Собственно сборка программ в среде WDK выполняется специальной утилитой Build, которая на основе информации, содержащейся в файлах makefile и sources, а также на основе переменных окружения, установленных средой построения, формирует сценарий сборки. Этот сценарий определяет, какой компилятор и с какими настройками следует вызывать для каждого из файлов с исходным кодом, каким образом следует выполнять компоновку объектных модулей, полученных в результате компиляции, где следует размещать собранный файл программы и т.п. Далее утилита Build применяет указанный сценарий к исходным файлам программы, расположенным в текущем каталоге. В случае отсутствия ошибок в сценарии сборки и исходных файлах происходит построение программы из исходного кода и ее размещение в месте, указанном в сценарии. Подобным образом можно выполнить построение любого типа программ, поддерживаемых соответствующей версией Windows: приложений различной разрядности, библиотек DLL и, в частности, драйверов.

Для сборки драйвера DemoDriver.sys в среде WDK необходимо наличие в каталоге для сборки трех файлов: makefile, sources и файла DemoDriver.cpp с исходным кодом драйвера.

Файл **makefile** имеет следующее стандартное содержание, которое не следует изменять во избежание возможных проблем при сборке:

```
#####################################################
# DO NOT EDIT THIS FILE!!! Edit .\sources. if you want to add a
# new source file to this component. This file merely indirects # to
the real make file that is shared by all the driver
# components of the Windows NT DDK
#
!INCLUDE $(NTMAKEENV)\makefile.def
#####################################################
```

Файл **sources** в простейшем случае имеет следующий вид:

```
#################################################
TARGETNAME=DemoDriver
TARGETTYPE=DRIVER
SOURCES=DemoDriver.cpp
#################################################
```

Строки TARGETNAME, TARGETTYPE и SOURCES определяют соответственно имя результирующего файла (DemoDriver.sys), тип собираемого приложения (драйвер) и файлы, содержащие исходный код драйвера (в данном случае – единственный файл с исходным кодом DemoDriver.cpp).

Файл **DemoDriver.cpp** содержит исходный код драйвера DemoDriver, рассмотренный выше.

При наличии указанных файлов, расположенных в одном каталоге, сборка драйвера выполняется простым запуском утилиты Build, по результатам выполнения которой в каталоге с исходными файлами появляется подкаталог, содержащий готовый драйвер. Название этого подкаталога определяется средой сборки, в которой происходило построение драйвера. Полученный в результате сборки файл DemoDriver.sys, содержащий бинарный код драйвера, можно использовать для дальнейшей работы.

Для создания решения необходимо выбрать соответствующий пункт в меню «Файл» (или нажать комбинацию клавиш Ctrl+Shift+N) и в появившемся диалоге выбрать тип проекта «Консольное приложение Win32», как показано на рис. 5. Кроме того, необходимо также указать название проекта приложения, место его расположения и наименование решения, в которое входит проект, в соответствующих полях.

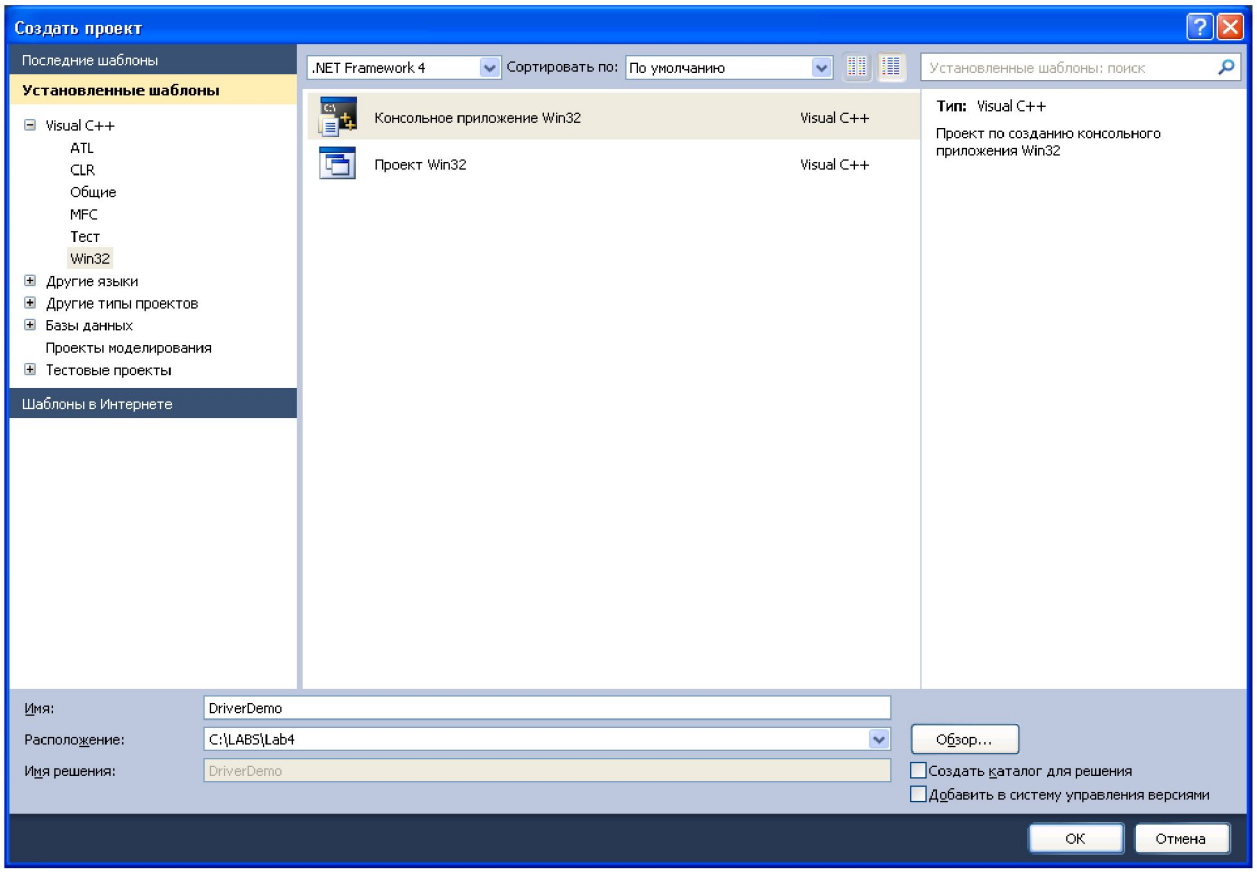

Рис. 5 – Выбор типа проекта для приложения, работающего с драйвером

В появившемся окне Мастера приложений Win32 необходимо установить параметры, как показано на рис. 6.

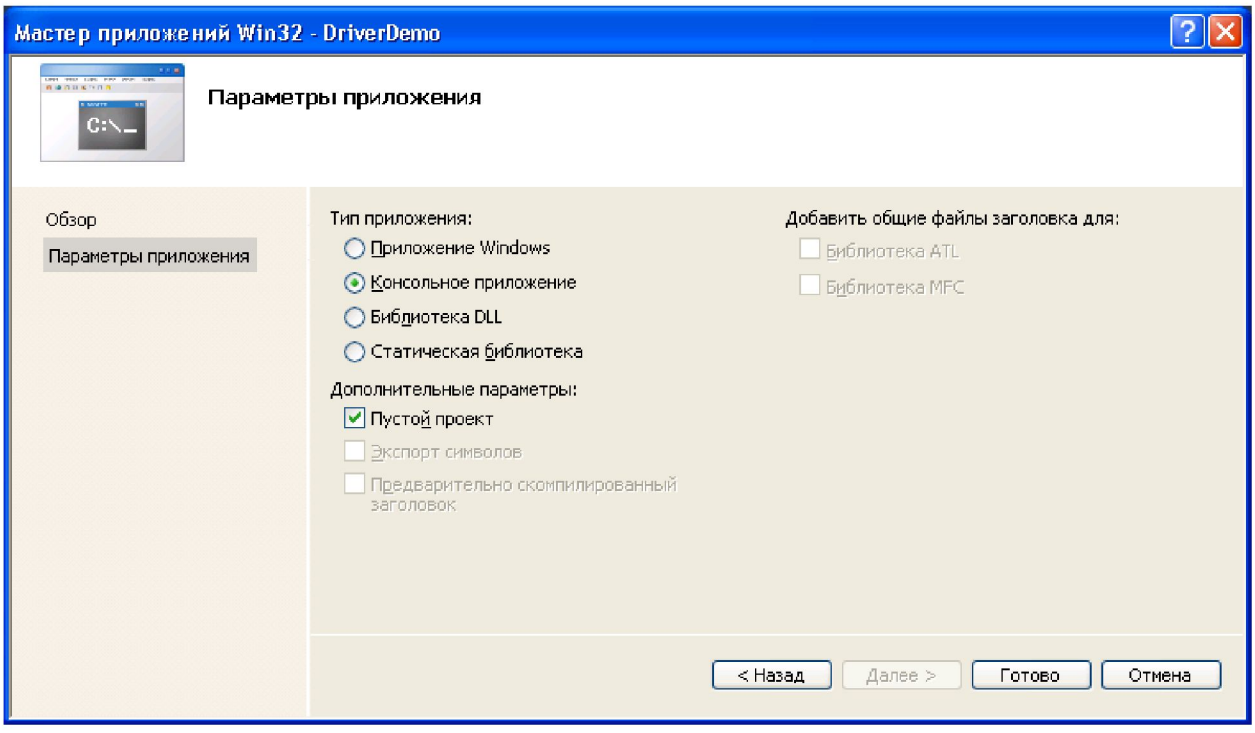

Рис. 6 – Установка параметров проекта для приложения

После выполнения этих действий будет создан пустой проект, в который следует добавить файл, содержащий исходный код демонстрационного приложения для работы с драйвером, либо создать такой файл непосредственно в проекте и написать там исходный код приложения. По завершении этих действий можно будет выполнить сборку приложения, выбрав соответствующий пункт из меню или нажав клавишу F7.

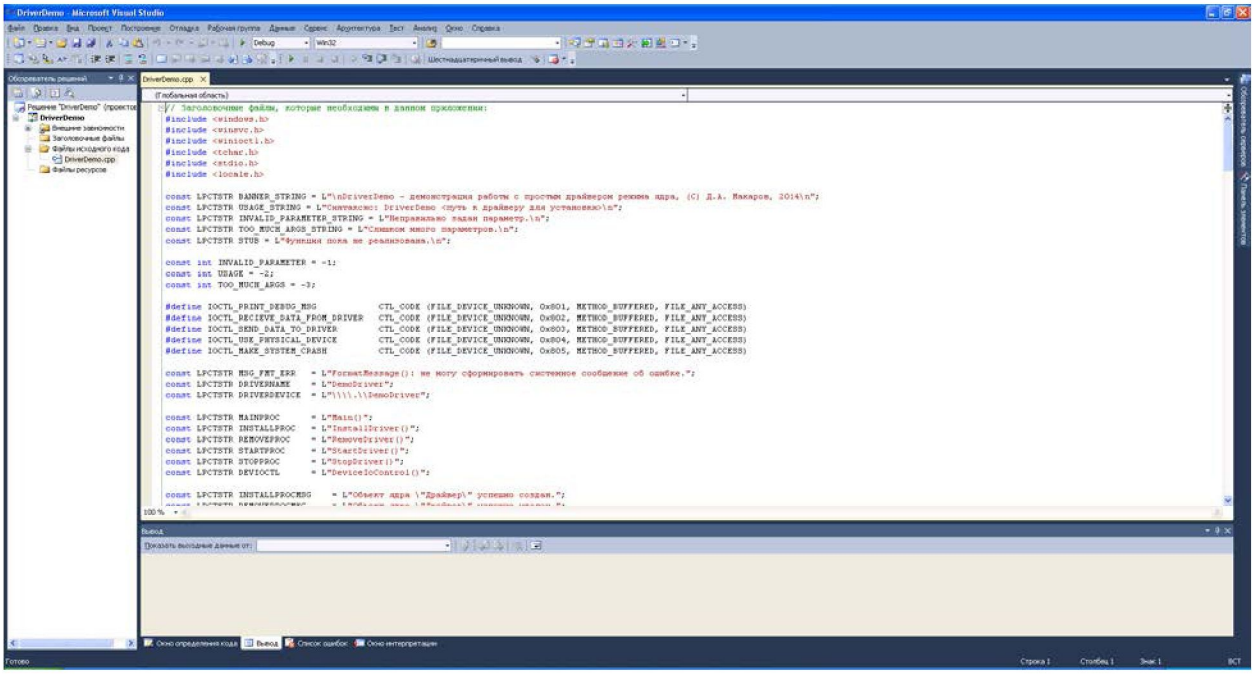

Рис. 7 – Решение настроено для сборки

3.2. Оценочные средства промежуточной аттестации

В данном разделе представляются теоретические вопросы (для оценки знаний).

Перечень теоретических вопросов к экзамену (для оценки знаний):

1. Назначение и классификация операционных систем.
2. Процессы и потоки.
3. Понятие WinAPI и его применение при разработке прикладных программ.
4. Планирование в системах с одним процессором.
5. Планирование в многопроцессорных системах.
6. Переключение потоков и краткосрочное планирование.
7. Приоритеты процессов и потоков.
8. Стратегии и виды планирования.
9. Межпроцессное взаимодействие.
10. Синхронизация, механизмы синхронизации.
11. Страничная адресация и работа файла подкачки.
12. Виртуальное адресное пространство процесса.
13. Управление памятью в системах Windows.
14. Библиотеки динамической компоновки DLL.
15. Явное и неявное связывание библиотек динамической компоновки.

Пример экзаменационного билета

МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РОССИЙСКОЙ ФЕДЕРАЦИИ
Федеральное государственное бюджетное образовательное учреждение высшего образования
«Забайкальский государственный университет»

ЭКЗАМЕНАЦИОННЫЙ БИЛЕТ № 1
по дисциплине Операционные системы
специальность 09.03.01 «Информатика и вычислительная техника»
семестр 7

1. Планирование в системах с одним процессором.
2. Управление памятью.

Составил: Макаров Д.А.

«12» мая 2019 г.

УТВЕРЖДАЮ:
Зав. кафедрой ИВТ и ПМ
_____ Валова О.В.
«12» мая 2019 г.

4. Методические материалы, определяющие процедуру оценивания знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций

4.1. Описание процедур проведения текущего контроля успеваемости студентов

В таблице представлено описание процедур проведения контрольно-оценочных мероприятий текущего контроля успеваемости студентов, в соответствии с рабочей программой дисциплины, и процедур оценивания результатов обучения с помощью запланированных оценочных средств.

Наименование оценочного средства	Описания процедуры проведения контрольно-оценочного мероприятия и процедуры оценивания результатов обучения
Собеседование (очная, заочная форма обучения)	Собеседование проводится по результатам освоения разделов дисциплины во время внеаудиторных занятий. Во время проведения собеседования пользоваться учебниками, справочниками, конспектами лекций, тетрадями для практических занятий не разрешено. Преподаватель на лекционном занятии, предшествующем занятию проведения собеседования, доводит до обучающихся: темы, количество вопросов, время и место проведения собеседования.
Защита лабора-	Указания для выполнения лабораторных работ выдаются студенту на

торной работы (очная, заочная форма обучения)	первом лабораторном занятии по указанной дисциплине. Преподаватель знакомит студентов с критериями оценивания и указывает дату сдачи конкретного задания из лабораторных работ.

4.2. Описание процедур проведения промежуточной аттестации

Экзамен

При определении уровня сформированности компетенций обучающихся на экзамене учитывается:

- знание программного материла дисциплины (блок 1 «знать»);
- знания, необходимые для выполнения типовых заданий (блок 2 «уметь»);
- владение методологией дисциплины, умение применять теоретические и практические знания в нестандартных ситуациях при решении типовых практических заданий, обосновывать свои действия (блок 3 «владеть»).

Экзамен проводится в письменной форме по билетам. Билет состоит из двух теоретических вопросов. Время подготовки заранее оговаривается преподавателем. Каждый вопрос билета оценивается отдельно по четырехбалльной шкале оценок, а далее вычисляется среднее арифметическое оценок, полученных за каждый вопрос. При выставлении оценки учитывается активность студента во время аудиторных занятий, и результаты собеседований по лекционному материалу и материалу практических занятий. В процессе ответа обучающегося на вопросы и задания билета, преподаватель может задавать дополнительные вопросы.

При определении уровня достижений обучающихся на экзамене обращается особое внимание на следующее:

1. дан полный, развернутый ответ на поставленный вопрос;
2. показана совокупность осознанных знаний об объекте, проявляющаяся в свободном оперировании понятиями, умении выделить существенные и несущественные признаки, причинно-следственные связи;
3. знание об объекте демонстрируются на фоне понимания его в системе данной дисциплины и междисциплинарных связей;
4. ответ формулируется в терминах дисциплины, изложен литературным языком, логичен, доказателен, демонстрирует авторскую позицию студента;
5. теоретические постулаты подтверждаются примерами из практики.