Министерство образования и науки Российской Федерации

Федеральное государственное бюджетное образовательное
учреждение высшего образования
«Комсомольский-на-Амуре государственный университет»

Кафедра «Математическое обеспечение и применение ЭВМ»

УТВЕРЖДАЮ
Первый проректор
_____ И.В. Макурин
«_28_» _03_____ 2018 г.

РАБОЧАЯ ПРОГРАММА

дисциплины «Компоненты операционных систем»

основной профессиональной образовательной программы
подготовки бакалавров
по направлению 09.03.01 - «Информатика и вычислительная техника»
профиль «Программное обеспечение средств вычислительной техники
и автоматизированных систем»

Форма обучения заочная
Технология обучения традиционная

Комсомольск-на-Амуре 2018

Автор рабочей программы _____ В.А. Тихомиров
профессор, к.т.н.

«11» 05 2018 г.

СОГЛАСОВАНО

Директор библиотеки _____ И.А. Романовская

«18» 05 2018г.

Заведующий кафедрой «МОПЭВМ» _____ В.А. Тихомиров

«16» 05 2018 г.

Заведующий выпускающей кафедрой «МОПЭВМ» _____ В.А. Тихомиров

«16» 05 2018 г.

Декан «ФЗДО» _____ М.В. Семибратова

«21» 05 2018г.

Начальник учебно-методического управления _____ Е.Е. Поздеева

«23» 05 2018г.

Введение

Рабочая программа дисциплины «Компоненты операционных систем» составлена в соответствии требованиями федерального государственного образовательного стандарта, утвержденного приказом Министерства образования и науки Российской Федерации от 12.01.2016 № 5, и образовательной программы подготовки бакалавров по направлению 09.03.01 «Информатика и вычислительная техника».

1 Аннотация дисциплины

<table>
<tr><td>Наименование дисциплины</td><td colspan="6">Компоненты операционных систем</td></tr>
<tr><td>Цель дисциплины</td><td colspan="6">ознакомление студентов с принципами построения и организации функционирования операционных систем защищенного режима работы процессоров типа Intel.</td></tr>
<tr><td>Задачи дисциплины</td><td colspan="6">• научить студентов уверенному администрированию операционной системы Windows на уровне системного программиста;
• дать студентам навыки программирования с использованием системных ресурсов ОС Windows.</td></tr>
<tr><td>Основные разделы дисциплины</td><td colspan="6">Устройство операционной системы и ее компонентов. Системное программирование.</td></tr>
<tr><td rowspan="3">Общая трудоемкость дисциплины</td><td colspan="6">4 з.е. / 144 академических часов</td></tr>
<tr><td colspan="5">Аудиторная нагрузка, ч</td><td rowspan="2">Всего за семестр, ч</td></tr>
<tr><td>Семестр</td><td>Лекции</td><td>Лаб. работы</td><td>СРС, ч</td><td>Контроль</td></tr>
<tr><td>5</td><td>4</td><td>10</td><td>126</td><td>4</td><td>144</td></tr>
<tr><td colspan="5">ИТОГО:</td><td></td><td>144</td></tr>
</table>

2 Перечень планируемых результатов обучения по дисциплине (модулю), соотнесенных с планируемыми результатами образовательной программы

Дисциплина «Компоненты операционных систем» нацелена на формирование компетенций, знаний, умений и навыков, указанных в таблице 1.

Таблица 1 – Компетенции, знания, умения, навыки

Наименование и шифр компетенции, в формировании которой принимает участие дисциплина	Перечень формируемых знаний, умений, навыков, предусмотренных образовательной программой		
	Перечень знаний (с указанием шифра)	Перечень умений (с указанием шифра)	Перечень навыков (с указанием шифра)
способностью разрабатывать компоненты аппаратно-программных комплексов и баз данных, используя современные инструментальные средства и технологии программирования (ПК- 2)	Возможности существующей программно-технической архитектуры З1(ПК-2-4)	Проводить оценку и обоснование рекомендуемых решений при разработке программного обеспечения У1(ПК-2-4)	Навыками разработки ПО на базе функций ядра операционной системы Н4(ПК-2-4)

3 Место дисциплины (модуля) в структуре образовательной программы

Дисциплина является вариативной дисциплиной входит в состав блока 1 «Дисциплины (модули)» и относится к вариативной части.

Для освоения дисциплины (модуля) необходимы компетенции, сформированные при изучении следующих дисциплин:

- информатика;
- программирование;
- современные программные средства

4 Объем дисциплины (модуля) в зачетных единицах с указанием количества академических часов, выделенных на контактную работу обучающихся с преподавателем (по видам учебных занятий) и на самостоятельную работу обучающихся

Общая трудоемкость (объем) дисциплины составляет 4 зачетных единицы, 144 академических часа.

Распределение объема дисциплины (модуля) по видам учебных занятий представлено в таблице 2.

Таблица 2 – Объем дисциплины (модуля) по видам учебных занятий

Объем дисциплины	**Всего академических часов**		
	Очная форма обучения		**Заочная форма обучения**
Общая трудоемкость дисциплины			144
Контактная работа обучающихся с преподавателем (по видам учебных занятий), всего			18
В том числе:			
занятия лекционного типа (лекции и иные учебные занятия, предусматривающие преимущественную передачу учебной информации педагогическими работниками)			4
занятия семинарского типа (семинары, практические занятия, практикумы, лабораторные работы, коллоквиумы и иные аналогичные занятия)			8

Объем дисциплины	Всего академических часов		
	Очная форма обучения		Заочная форма обучения
самостоятельная работа обучающихся и контактная работа, включающая групповые консультации, индивидуальную работу обучающихся с преподавателями (в том числе индивидуальные консультации); взаимодействие в электронной информационно-образовательной среде вуза			128 (из низ 4 час. индив. консульт.)
Промежуточная аттестация обучающихся			4

5 Содержание дисциплины (модуля), структурированное по темам (разделам) с указанием отведенного на них количества академических часов и видов учебных занятий

Таблица 3 – Структура и содержание дисциплины (модуля) для заочного обучения

Наименование разделов, тем и содержание материала	Компонент учебного плана	Часов	Планируемые (контролируемые) результаты освоения		
			Форма проведения	Компетенции	Знания, умения, навыки
Тема Установочная лекция. Общие сведения о целях, предмете и задачах дисциплины. Методическое обеспечение дисциплины. Обзор заданий на контрольные работы. Методические рекомендации по выполнению самостоятельной работы студента.	Лекция	2	Традиционная	ПК-2	З1(ПК-2-4)
Тема Принципы построения операционных систем (ОС). Примеры выполнения заданий контрольных работ. Основные функции ОС; обзор современных ОС и операционных оболочек. Структура ОС Windows, порядок запуска, распределение памяти ПЭВМ после запуска ОС. Вычислительный процесс и его реализация с помощью ОС Windows, управление вычислительными процессами, вводом-выводом, оперативной памятью. Системные области операционной системы, взаимодействие компонентов ОС при выполнении вычислительных процессов.	Лекция	2	Презентационная	ПК-2	З1(ПК-2-4)
Тема **Системные функции оконного интерфейса** Отрабатываются умения и навыки построения простейших приложений на базе использования функций ядра операционной системы	Лабораторная работа	1	Традиционная	ПК-2	У1(ПК-2-4), Н4(ПК-2-4)
Тема Инструменты дизассемблирования и отладки компонент операционной системы	Лабораторная работа	1	Традиционная	ПК-2	У1(ПК-2-4), Н4(ПК-2-4)
Тема Применение объектов синхронизации при программировании многопоточных приложений.	Лабораторная работа	2	Традиционная	ПК-2	У1(ПК-2-4), Н4(ПК-2-4)

Наименование разделов, тем и содержание материала		Компонент учебного плана	Часов	Планируемые (контролируемые) результаты освоения		
				Форма проведения	Компе-тенции	Знания, умения, навыки
Проводится разработка консольного многопоточного приложения с использо-ванием различных объектов системной синхронизации.						
Тема Объекты синхронизации в многопоточном приложении.		Лаборатор-ная работа	2	Традици-онная	ПК-2	У1(ПК-2-4), Н4(ПК-2-4)
Тема **Файловая система NTFS** проводится исследование структуры системы управ-ления файлами в операционной системе Windows		Лаборатор-ная работа	2	Традици-онная	ПК-2	У1(ПК-2-4), Н4(ПК-2-4)
Тема-1 функции оконного интерфейса Тема-2 инструменты отладки Тема-3 объекты синхронизации Тема-4 управление памятью Тема-5 управление процессами Тема-6 файловая система NTFS		Самостоя-тельная рабо-та обучаю-щихся	128	Традици-онная	ПК-2	З1(ПК-2-4) У1(ПК-2-4), Н4(ПК-2-4)
ИТОГО по дисци-плине	Лекции		4	-	-	-
	Лабораторные работы		8	-	-	-
	Самостоятельная работа обучающихся		128	-	-	-
	Промежуточная аттестация		4			
Всего по дисциплине:			144			

6 Перечень учебно-методического обеспечения для самостоятельной работы обучающихся по дисциплине (модулю)

Самостоятельная работа обучающихся, осваивающих дисциплину «Компоненты операционных систем», состоит из следующих компонентов: изучение теоретических разделов дисциплины; подготовка и оформление курсовой работы.

Для успешного выполнения всех разделов самостоятельной работы обучающимся рекомендуется использовать следующее учебно-методическое обеспечение:

- Пособие Тихомиров В.А. Компоненты операционных систем защищенного режима работы процессора/Комсомольск-на-Амуре, КнГАТУ, 2013. - 205 с.
- Тихомиров В.А. Комплект электронных УММ для выполнения лабораторных работ и КР по дисциплине «Компоненты операционных систем» в локальной сети ФКТ по адресу \\3k316m04\ Share\МОП_ЭВМ\3. Заочное\Бакалавры\ОС.

Рекомендуемые графики выполнения самостоятельной работы для семестров с количеством недель 18 и 17, представлены в таблицах 4.1-4.2.

Общие рекомендации по организации самостоятельной работы

Выполнение курсовой работы

Основным содержание курсовой работы является разработка программного модуля под операционную систему Windows, выполняющего заданные системные функции или опирающегося на системную поддержку заданной ОС.

В связи с высокой сложностью и трудоемкостью освоения данного материала с нуля, студенту предоставляется «ПРОТОТИП» его курсовой работы, выполненный на языке BASIC.

Изучая «ПРОТОТИП», студент может понять сценарий работы его курсового задания, набор системных функций, используемых для его выполнения, порядок подачи параметров в эти функции.

Полученная по «ПРОТОТИПУ» информация позволит студенту резко сократить объем работ по предварительному поиску и изучению информации о нужных для курсовой работы функциях ядра операционной системы и позволит более эффективно направить усилия на изучение и разработку собственного задания курсовой работы.

«ПРОТОТИПЫ» курсовой работы находятся в методическом обеспечении, указанном в п.6 выше.

Таблица 4.1 – Рекомендуемый график выполнения самостоятельной работы студентами очного обучения при 18-недельном семестре

Вид самостоятельной работы	**Часов в неделю**																		**Итого по видам работ**
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	
Выполнение курсовой работы										6	6	6	6	6	6	6	6	6	54
Самостоятельное изучение теоретических разделов курса	3	3	0	3	3	0	3	3	0	3	3	0	3	3	0	3	5	0	38
Подготовка отчёта о выполнении лабораторной работы			6			6			6			6			6			6	36
Итого 4 семестр	**3**	**3**	**6**	**3**	**3**	**6**	**3**	**3**	**6**	**9**	**9**	**12**	**9**	**9**	**12**	**9**	**11**	**12**	128

Таблица 4.2 - Рекомендуемый график выполнения самостоятельной работы студентами очного обучения при 17-недельном семестре

Вид самостоятельной работы	**Часов в неделю**																		**Итого по видам работ**
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17		
Выполнение курсовой работы										7	7	7	7	7	7	7	7		56
Самостоятельное изучение теоретических разделов курса	3	3	0	3	3	0	3	3	0	3	3	0	4	4	0	4	6		42
Подготовка отчёта о выполнении лабораторной работы			6			6			6			6			6				30
Итого 4 семестр	**3**	**3**	**6**	**3**	**3**	**6**	**3**	**3**	**6**	**10**	**10**	**13**	**11**	**11**	**13**	**11**	**13**		128

7 Фонд оценочных средств для проведения текущего контроля и промежуточной аттестации обучающихся по дисциплине (модулю)

Таблица 5 – Паспорт фонда оценочных средств

Контролируемые разделы (темы) дисциплины	Код контролируемой компетенции (или ее части)	Наименование оценочного средства	Показатели оценки
Программные функции операционной системы Windows.	ПК-2_4	Защита лабораторных работ	Умеет использовать системные функции для построения прикладных программ
Дизассемблеры и отладчики.	ПК-2_4	Защита лабораторных работ	Умеет проводить анализ программного кода с использованием системного отладчика
Процессы и потоки в операционной системе.	ПК-2_4	Защита лабораторных работ	Умеет управлять памятью компьютера через системные функции
Объекты синхронизации потоков.	ПК-2_4	Защита лабораторных работ	Умеет создавать, запускать и использовать потоки в операционной системе
Структура и управление NTFS.	ПК-2_4	Защита лабораторных работ	Умеет программировать многопоточные приложения
Все разделы	ПК-2_4	Курсовая работа	Умеет разрабатывать приложения с использованием функций ядра операционной системы

Промежуточная аттестация проводится в форме **итоговой оценки.**

Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности, характеризующих этапы формирования компетенций, представлены в виде технологической карты дисциплины (таблица 6).

Таблица 6 – Технологическая карта

	Наименование оценочного средства	Сроки выполнения	Шкала оценивания	Критерии оценивания
5 семестр ***Промежуточная аттестация в форме итоговой оценки***				
1	Защита лабораторных работ (5 работ)	В течение семестра	10 баллов/за одну работу	10 баллов - студент правильно выполнил практическое задание. Показал отличные знания и умения в рамках освоенного учебного материала. 6 баллов - студент выполнил практическое задание с небольшими неточностями. Показал хорошие знания и уме-

	Наименование оценочного средства	Сроки выполнения	Шкала оценивания	Критерии оценивания
				ния в рамках освоенного учебного материала. 4 балла - студент выполнил практическое задание с существенными неточностями. Показал удовлетворительные знания и умения в рамках освоенного учебного материала. 2 балла - при выполнении практического задания студент продемонстрировал недостаточный уровень знаний и умений. 0 баллов – задание не выполнено.
	Итого	-	50 баллов	
Критерии оценки результатов обучения по дисциплине: 0 – 64 % от максимально возможной суммы баллов – «неудовлетворительно» (недостаточный уровень для текущей аттестации по дисциплине); 65 – 74 % от максимально возможной суммы баллов – «удовлетворительно» (пороговый (минимальный) уровень); 75 – 84 % от максимально возможной суммы баллов – «хорошо» (средний уровень); 85 – 100 % от максимально возможной суммы баллов – «отлично» (высокий (максимальный) уровень)				
2	Курсовая работа	В конце семестра	5 баллов	ОТЛИЧНО- студент правильно выполнил курсовую работу. Показал отличные владения навыками применения полученных знаний и умений при решении профессиональных задач в рамках усвоенного учебного материала. Ответил на все дополнительные вопросы на защите. ХОРОШО - студент выполнил курсовую работу с небольшими неточностями. Показал хорошие владения навыками применения полученных знаний и умений при решении профессиональных задач в рамках усвоенного учебного материала. Ответил на большинство дополнительных вопросов на защите. УДОВЛЕТВОРИТЕЛЬНО - студент выполнил курсовую работу с существенными неточностями. Показал удовлетворительное владение навыками применения полученных знаний и умений при решении профессиональных задач в рамках усвоенного учебного материала. При ответах на дополнительные вопросы на защите было допущено много неточностей. НЕУДОВЛЕТВОРИТЕЛЬНО - при выполнении курсовой работы студент продемонстрировал недостаточный уровень владения навыками применения полученных знаний и умений при решении профессиональных задач в рамках усвоенного учебного материала. При ответах на дополнительные вопросы на защите было допущено множество неточностей.

Задания для текущего контроля

ЗАДАНИЕ НА ЛАБОРАТОРНУЮ РАБОТУ 1

Задание 1.1

1. Изучите теоретическую часть и примеры, описанные в step1_Win32.doc методической разработке;
2. Ознакомьтесь с SDK по API функциям операционной системы Windows;
3. Реализуйте на ПЭВМ в ОС Windows пример 32-х разрядного приложения, приведенные в методической разработке step1_Win32.doc;
4. Доработайте пример созданного приложений согласно варианта, заданного в таблице:

№ вар.	Задание на доработку программы
1	Добавить на выводимое сообщение три кнопки и системную пиктограмму с предупреждающим значком. При нажатии на кнопки должна меняться пиктограмма. А при нажатии на одну и ту же кнопку два раза - заканчиваться программа.
2	Поместить на выводимом сообщении две кнопки «ДА» и «НЕТ». При нажатии на кнопку «ДА» должно появляться новое сообщение, а старое - исчезать. При нажатии на кнопку «НЕТ» заканчивается работа программы.
3	Поместить на выводимом сообщении три кнопки «ДА», «НЕТ», «Отмена». При нажатии на кнопку «ДА» должно появляться новое сообщение. При нажатии на кнопку «НЕТ» - меняться пиктограмма на сообщении, при нажатии на «Отмена» - заканчивается работа программы.
4	Поместить на выводимом сообщении три кнопки «ДА», «НЕТ», «Отмена». При нажатии на кнопку «ДА» должен меняться основной текст сообщения. При нажатии на кнопку «НЕТ» - должен меняться текст заголовка окна сообщения, при нажатии на «Отмена» - заканчивается работа программы.
5	С помощью выводимых сообщений составить диалог с пользователем, в ходе которого на сообщениях появляются и используются в диалоге разные кнопки и разные пиктограммы. Сценарий диалога разработать самостоятельно.
6	Поместить на выводимом сообщении три кнопки «ДА», «НЕТ», «Отмена». При нажатии на кнопку «ДА» должно появляться новое сообщение. При нажатии на кнопку «НЕТ» - меняться пиктограмма на сообщении, при нажатии на «Отмена» - заканчивается работа программы.
7	Поместить на выводимом сообщении три кнопки «ДА», «НЕТ», «Отмена» и составить программу - тест: пользователь отвечает на вопросы «да» или «нет» и в конце ему выдается некоторый результат. При нажатии на «Отмена» - заканчивается работа программы.
8	Поместить на выводимом сообщении три кнопки «ДА», «НЕТ», «Отмена». При нажатии на кнопку «ДА» должен меняться основной текст сообщения. При нажатии на кнопку «НЕТ» - должен меняться текст заголовка окна сообщения, при нажатии на «Отмена» - заканчивается работа программы.

Задание 1.2

1. Изучите теоретический материал по организации программ со стандартным оконным интерфейсом в среде Windows (файл Теория_SDI_лаб2.DOC).
2. Изучите текст программы, изложенной в методической разработке Step2_W32.
3. Наберите текст этой программы и текст файла ее определения.
4. Измените строку компоновки в BAT файле компиляции, восстановив в ней параметр %1. DEF.
5. Проведите компиляцию, компоновку и, при необходимости, отладку этой программы до рабочего состояния.
6. Проследите работу этого приложения в отладчике OllyDebugger 1.1 сделайте выводы о порядке организации вызова системных процедур из Вашего приложения.
7. Запустите повторно несколько копий Вашего приложения, проследите их совместную работу.
8. Проведите доработку данного примера согласно варианта сценария, приведенного в таблице ниже:

№ вар.	Содержание сценария
1	Сделайте так, чтобы при каждом повторном запуске Вашего приложения случайным образом задавался цвет шрифта в окне.
2	Сделайте так, чтобы при щелчке правой клавишей мыши в окне рисовался бы круг произвольного (случайного) диаметра и цвета
3	Комбинация клавиш Shift+M должна вызывать появление системного сообщения окна MessageBox с числом нажатий кнопки мыши
4	Сделайте так, чтобы при щелчке правой клавишей мыши открывалось системное окно открытия файлов и имя выбранного файла печаталось крупными буквами в центре основного окна программы
5	Сделайте так, чтобы при щелчке правой клавишей мыши открывалось системное окно выбора цвета и текст в окне начинал печататься выбранным цветом
6	Сделайте так, чтобы при щелчке правой клавишей мыши, если нажата клавиша Ctrl - текст в окне увеличивался на 5пт, а если нажата клавиша Alt - уменьшался бы на такую же величину
7	Сделайте так, чтобы при щелчке правой клавишей мыши открывалось бы новое окно, в котором прорисовывалась бы сетка из горизонтальных и вертикальных линий, разнообразного случайного цвета
8	Сделайте так, чтобы при нажатии комбинации клавиш Ctrl- левый + Alt-правый + Shift в окне программы прорисовывался бы овал с зеленым толстым контуром и светло-голубой заливкой

9	Сделайте так, чтобы по правому краю главного окна программы имелись бы 10-ть стандартных системных кнопок (друг под другом) с надписями «кнопка -1» ... «кнопка-10», а при щелчке правой клавишей мыши - они исчезали бы (на каждый щелчок - одна кнопка), начиная с десятой.
10	Сделайте так, чтобы при щелчке правой клавишей мыши в окне программы в точку нахождения курсора мыши рисовалась бы вертикальная черная линия 2пт толщиной, а при удержании клавиши Ctrl - аналогичная горизонтальная линия.
11	При запуске окно должно анимироваться слева направо, а если программа запускается повторно - возникать из ничего.
12	Добавить к окну меню с привольными выпадающими позициями
13	Добавить к окну инструментальную линейку с двумя произвольными стандартными кнопками

Задание на лабораторную работу 2:

Вам выданы рабочие модули (2 штуки) программ (EXE файл PE формат) по вариантам в папке «Варианты заданий». При запуске программ производится запрос пароля (или организуется регистрация программы), после чего дается заключение о правильности прохода через пароль. Ваша задача:

1) Знакомство с отладчиком уровня ПРИЛОЖЕНИЯ

Используя отладчик OlleDebuger пройти по программе, найти место, где происходит дешифровка пароля и расшифровать его, после чего снова запустить программу, ввести правильный пароль и получить подтверждение о правильности прохода через пароль.

2) Изучить назначение рабочих окон отладчика;

3) Изучить инструменты поиска функций и их расположения в коде;

4) Изучить инструменты просмотра содержимого памяти машины по заданным адресам и содержимого регистров процессора;

5) Изучить приемы редактирования памяти машины и регистров процессора;

6) Изучить приемы редактирования кода программы и фиксации исправленного кода в исполняемом файле.

7) Оформить электронный отчет, в котором отразить:

- титульный лист;
- тему лабораторной работы и задание на лабораторную работу;
- описание этапов дешифровки пароля на OLLEDEDUGER
- описать инструменты отладчика, примененные для решения задачи, привести экранные формы из дизассемблера, использованные при дешифровке пароля;
- вид экрана после срабатывания откорректированной программы;

Электронный отчет представить на защиту. Знать приемы работы с отладчиком OlleDebuger.

Задание на лабораторную работу 3

- внимательно изучить теоретический материал;
- набрать и проверить работоспособность программных модулей, представленных на листингах 1 -3. Тексты программ вместе с результатами их тестирования вставить в отчет;
- выполнить индивидуальное задание в соответствии с выданным преподавателем вариантом

№ вар.	Содержание задания
1	Выделить память под массив 1000x1000 элементов типа double и заполнить его случайными числами в интервале от 1 до 10, защитить страницы памяти с массивом от записи, выдать сумму и среднее арифметическое элементов массива, дать команду на обнуление элементов и получить системное предупреждение о невозможности записи в массив.
2	Зарезервировать память в размере 10 Мбт. Произвести физическое выделение памяти 10 раз сегментами по 1 Мбт. Вывести на экран адреса выделенных сегментов. По полученным данным построить карту проведенного выделения памяти.
3	Выделить память под два массива 1000x500 и 500x1000 элементов типа int и заполнить их случайными числами в интервале от 0 до 100. Защитить от записи страницы первого массива. Перемножить эти два массива и выдать результат на экран. Перехватить обработчик исключений. Дать команду на обнуление массивов. На исключение о защите памяти выдать сообщение с указанием названия массива, который не удалось обнулить.
4	Выделить 300 Кбт памяти и скопировать в нее содержимое ПЗУ машины. Повторить это действие еще пять раз. Распечатать на экране адреса всех выделенных блоков памяти. По полученным данным построить карту проведенного выделения памяти
5	Зарезервировать память 30 Мбт. Выделяя порциями необходимую память считать в нее последовательно от 1 до 20 дискет. Найти контрольную сумму считанных байтов. Записать эти дискеты на винчестер в файлы в виде отдельных образов.
6	Составить программу, которая бы читала и выводила на экран содержимое своего кодового сегмента.

7	Выделить память 10 Мбт. Все ячейки заполнить единицами. Генератором случайных чисел создать и случайным образом «разбросать» по байтам выделенной памяти десять целых чисел в интервале от 2 до 200. Просканировать память и все страницы, содержащие только единицы пометить как запрещенные к доступу. Произвести суммирование всех байтов выделенной памяти. При возникновении исключения при обращении к запрещенной странице программа должна «понимать», что на данной странице содержатся одни единицы. Провести исследование загруженности памяти на всех этапах выполнения программы.
8	Необходимо выделить участки памяти размером 32 Кбт по адресам: 900 000, 1 000 000 и 3 000 000. Заполнить эти участки единицами. Защитить от записи и попробовать очистить. При возникновении ошибки - перехватить ее и выдать об этом сообщение.
9	Составить программу, которая во время своей работы модифицировала бы собственный кодовый сегмент.
10	Составить программу для исследования наличия и содержания у запущенного модуля области PSP.
11	Составить программу для определения размера страницы памяти, выделяемых MMU процессам на данном компьютере, размера свободной памяти в текущий момент и процент загрузки процессора в текущий момент.

Задание на лабораторную работу 4

- внимательно изучить ниже прилагаемый теоретический материал; набрать и проверить работоспособность программных модулей, представленных на листингах 1-7 методических указаний. Тексты программ вместе с результатами их тестирования вставить в отчет;
- выполнить индивидуальное задание в соответствии с выданным пре-

№ вар.	Содержание задания
1	Программа должна запускать другую программу, путь которой указан в командной строке и анализировать код ее завершения.
2	Составить программу, которая считает количество счастливых билетов в трамвайном рулоне и выводит их на экран в десять потоков. Каждый поток заведует своим диапазоном номеров: 000000 - 199999, 200000-299999, 300000-399999 ... 900000 - 999999.
3	Программа создает три потока, в которых делятся друг на друга случайные числа пока их число не станет 100000 или не произойдет деление на 0. Анализируется код возврата.
4	Программа создает десять потоков. Каждый поток создает свою матрицу размером 1000x1000 элементов и заполняет ее случайными целыми числами. Далее каждый поток находит среднее арифметическое среди чисел матрицы и выводит результат на экран.

5	Программа создает три потока и каждом запускает расчет разных сложных функций. После завершения потоков в главную программу возвращаются характеристики рабочего набора этих потоков и результаты выполнения расчетов. Вся информация выводится на экран.
6	Главный поток создает матрицу 10000*10000 элементов и заполняет ее случайными числами. Затем поражается три потока, один - считает среднее арифметическое матрицы, второй - среднее геометрическое, а третий – средне- квадратичное. Результаты выводятся на экран. Так же на экран выводится метка достижения каждого потока середины своих вычислений.
7	Составить программу для ответа на вопрос: как много потоков может создать приложение, прежде чем ОС начнет «беспокоиться». Загрузку процессора контролировать системным монитором.
8	Пять потоков генерируют случайные целые числа в диапазоне от 1 до 49. Шестой поток - анализирует эти числа. Если появляется пара одинаковых чисел - выводится сообщение ПАРА и номера потоков этой пары. Если три числа одинаковы - ТРИ, Четыре - ЧЕТЫРЕ и пять - ПЯТЬ. Через 60 секунд работы программа завершается. Если не было ни одного совпадения - выводится об этом сообщение.
9	Главный поток преобразует себя в нить. Затем создает еще пять нитей. Главная нить по очереди передает управления каждой из пяти нитей. Те, выполняя действия (любые) возвращают управление главной нити.
10	Создать три потока, контролирующие бюджет Смитта, Ганса и Ивана через локальную память потоков (TLS). Потоки обращаются к функции назначения выплат (генератор случайных чисел в интервале от-1000 до +1000). Функция пере-
11	Создать три потока и, используя технологию APC, выполнить по очереди назначенные этим потокам функции APC.

Задание на лабораторную работу 5

- внимательно изучить ниже прилагаемый теоретический материал;
- набрать и проверить работоспособность программных модулей, представленных на листингах 1 -4. Тексты программ вместе с результатами их тестирования вставить в отчет;
- выполнить индивидуальное задание в соответствии с выданным преподавателем вариантом:

№ вар.	Содержание задания
1	Один поток готовит матрицу в памяти M = 1000x1000 байт со случайными числами от 0 до 255. Другой поток в это время принимает с клавиатуры два числа X, Y, а третий поток - готовит на экране окно для вывода результатов расчетов. Как только данные с клавиатуры введены - из подготовленного массива выбирается байт с индексом (X,Y) и выводится на экран в окне третьего

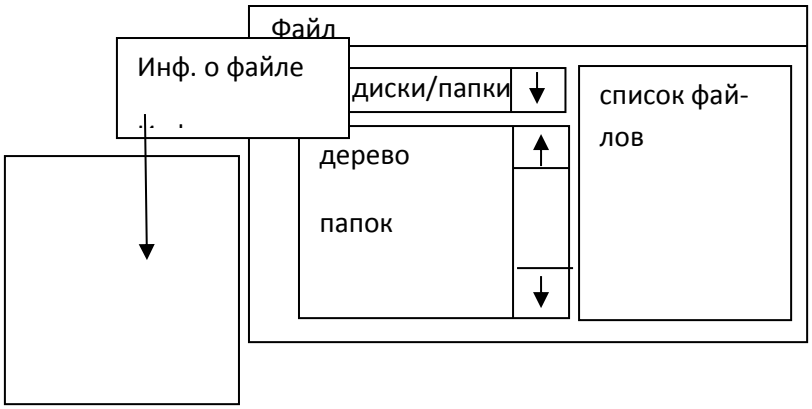
2	Имеется файл F1, в котором записано 20 слов (можно больше). Три потока генерируют случайные числа в диапазоне 0 - 100. Если сгенерированное число больше 90, поток генерирует случайное число n от 1 до 20, открывает файл F1, берет из него слово под номером n, открывает файл F2 и вписывает взятое слово в него файлы F1 и F2 закрываются. Каждый поток должен записать в файл F2 по три слова. Полученное предложение приложите в отчет.
3	Три потока генерируют случайные числа в диапазоне от 0 до 1000. Если в потоке число попадется больше 900, поток выводит на экран (в случайном месте) окно, в котором непрерывно создаются одним потоком - закрашенные окружности случайных радиусов, другим потоком - прямоугольники, а третьим - треугольники. Одновременно на экране может быть не более пяти окон. Пользователь может закрыть одно из них. Тут же появляется другое.
4	Пять потоков генерируют случайные числа в интервале от 0 до 1000. Если свободен мьютекс, поток вызывает функцию BIZNES и передает ей свой номер и сгенерированное число. Функция присуммирует полученное число к счету под номером обратившегося потока. Когда все потоки обратятся к функции BIZNES по 10 раз, программа выводит на экран итоговые суммы, накопившиеся на всех пяти счетах.
5	Когда пользователь нажимает на клавишу t по экрану снизу вверх двигается значок *. При нажатии на —► слева направо двигается значок >. Одновременно на экране может быть не более 3 -х значков каждого типа.
6	Приложение запускается три раза и создает три окна (последующие запуски к созданию окон не приводят). Когда приложение запускается с ключом /R все окна закрываются.
7	Создать приложение 1, после запуска которого оно ждет, пока не будет запущено приложение 2 и дважды не будет запущено приложение 3, только тогда оно выводит сообщение «УСЛОВИЕ ВЫПОНЕНО»
8	Один поток читает файл, указанный в первом параметре программы, второй поток читает файл, указанный во втором параметре программы. Третий поток сравнивает файлы и выводит на экран номера и значения несовпадающих байтов.
9	1) Создать поток расчёта сложной функции по случайным входным данным. 2) Создать поток вывода результатов расчёта через каждые 3 сек. 3) Синхронизировать потоки. 4) Реализовать операции полного останова потоков, перевода потоков в режим
10	Главный поток создает массив A - 60 * 60 элементов и массив B - 60 * 60 элементов и обнуляет их. После сего создаются пять потоков, которые заполняют массив A случайными значениями в интервале от 1 до 65535. Необходимо так синхронизовать потоки, чтобы они не затирали данные, введенные друг другом. В массив B при этом помещаются номера потоков, заполнивших соответствующую ячейку. Массив B вывести на экран.

Задания на курсовую работу

Разработать программу по технологии Windows API согласно варианта:

1. Приложение Информации о файле

В окне приложения можно выбрать исполняемый файл, DLL или нестандартный элемент и получить ВСЕ данные для выбранного файла. Команды меню позволяют прочитать список доступных принтеров из секции устройств файла WIN.INI или из системного реестра.



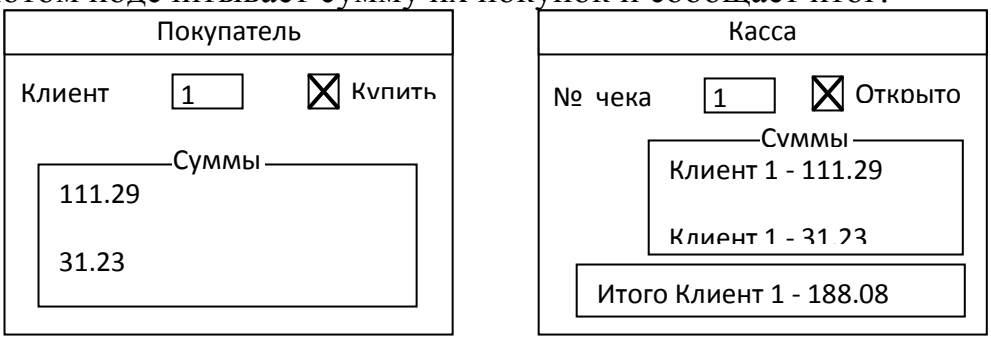
Получение данных версии делится на два этапа. Команда меню «Информация о файле» обращается к структуре VS_FIXEDFILEINFO в ресурсе версии файла и читает тип файла, а также номера версий файла и продукта и пр., затем выделяет в ресурсе версии строки StringFileInfo и читает из них название организации, описание файла и сведения об авторских правах.

Команда меню «Информация о файле» обеспечивает сбор данных о принтерах так, что вместе с именами принтеров выводится и дополнительная информация. Используются функции библиотеки динамической компоновки APIGID32.DLL.

2. Взаимодействие приложений

На основе функций, управляющих общей памятью приложений создать схему клиент/сервер, имитирующую работу супермаркета.

Сервер выполняет функции кассира, который ожидает прихода покупателей, потом подсчитывает сумму их покупок и сообщает итог.



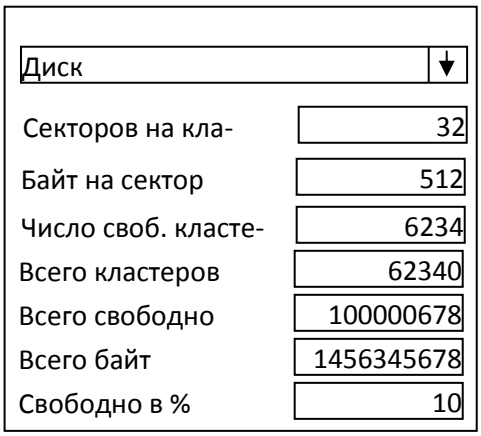
Клиент является аналогом покупателя, который ожидает открытия кассы и предъявляет свои покупки кассиру. На рисунке показаны рабочие окна двух приложений, Покупатель и Касса. В окне приложения («покупателя») имеется текстовое поле для ввода идентификатора клиента и флажок, при установке которого выдается запрос на обслуживание.

Список заполняется итоговыми суммами всех успешно обработанных заявок. В окне приложения («кассира») также присутствует текстовое поле, в котором вводится идентификатор «кассы», и флажок, установка которого означает готовность сервера к обслуживанию. В списке перечислены все успешно обработанные операции (разумеется, суммы у «покупателя» и «кассира» должны совпадать). Каждое приложение содержит простейший управляющий механизм, срабатывающий при обработке событий таймера. Это сделано для того, чтобы замедлить работу приложения и более наглядно продемонстрировать происходящее.

В программе создается структура в общей области памяти, которая используется при обмене информацией между клиентским и серверным приложением. Клиентское приложение проверяет состояние поля Total и узнает, свободен ли сервер. Если значение поля равно нулю, клиент загружает набор цен (массив Prices) покупаемых товаров и заполняет поле Done, тем самым сообщая серверу о своей готовности. Сервер суммирует цены отдельных товаров, после чего заполняет поле Total и сбрасывает поле Done. Клиент сбрасывает поле Total, сообщая, что обслуживание «покупателя» завершено и «касса» готова обработать новую заявку. Клиент также заполняет поле Client, чтобы сервер мог занести имя клиента в свой список.

3. Определение свободного места на диске

Программа является простым примером того, как получить объем свободного места на диске функцией GetDiskFreeSpace. Главное окно программы изображено на рисунке.

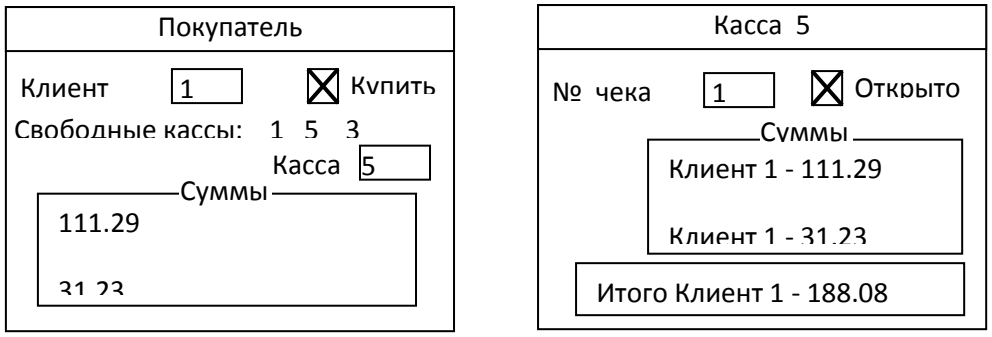


4. Большой супермаркет

На основе функций, управляющих общей памятью приложений создать схему клиент/сервер, имитирующую работу супермаркета.

Сервер выполняет функции кассира, который ожидает прихода покупателей, потом подсчитывает сумму их покупок и сообщает итог.

Клиент является аналогом покупателя, который ожидает открытия кассы и предъявляет свои покупки кассиру. На рисунке ниже показаны рабочие окна двух приложений, Покупатель и Касса. В окне приложения («покупателя») имеется текстовое поле для ввода идентификатора клиента и флажок, при установке которого выдается запрос на обслуживание.



Реализуйте программы, поддерживающие работу нескольких «касс». Разумеется, в распоряжение клиента необходимо предоставить некий механизм выбора. Возможны разные варианты. Например, можно просто выбрать случайный номер кассы или последовательно опробовать разные серверы. Поскольку каждый сервер обладает уникальным именем (имена файлового отображения и мутекса), вам будет нетрудно проверить, существует ли сервер с конкретным номером.

После реализации схемы с несколькими серверами попробуйте воспользоваться семафорами для ограничения количества клиентов, поддерживаемых каждым сервером (количества клиентов, ожидающих в одной очереди).

5. Запуск приложений

Разработать приложение, запускающее другое приложение тремя разными способами.

Запускаемое приложение – небольшая программка, выводящая на экран свой идентификатор процесса, идентификатор нити и хендел окна формы. Основная программа, после запуска одного экземпляра дочернего процесса, переходит в состояние ожидания его завершения. При этом момент завершения приложения для каждого вида запуска определяется особо.

На главной форме проекта находятся три кнопки, соответствующие различным способам запуска программ:

- с помощью функции WinExec
- с помощью функции CreateProcess
- с помощью функции ShellExecute

Завершение запущенного процесса должно регистрироваться родителем и сообщать пользователю код завершения процесса.

6. Использование анонимных каналов

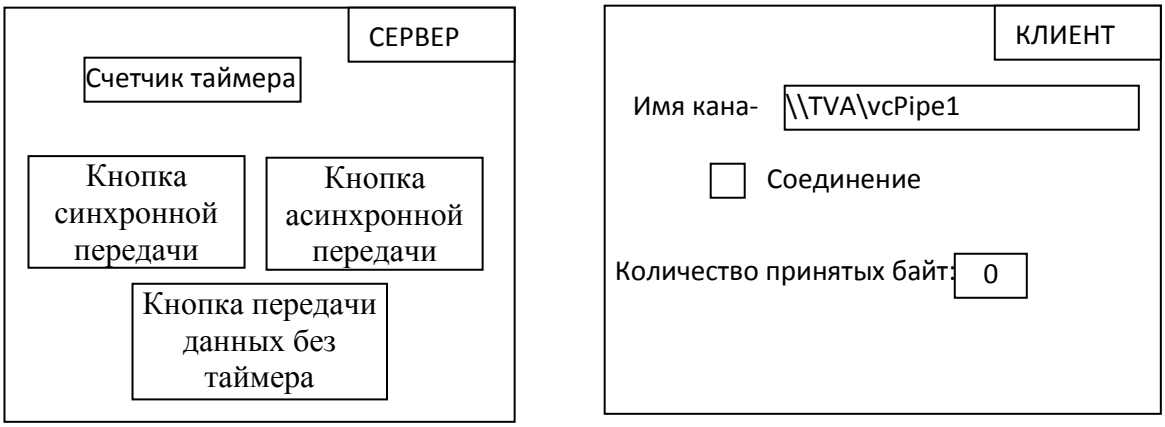
Создать два приложения, показывающих, как организовать взаимодействие между родительским и дочерним процессом с применением анонимных каналов. Оба приложения состоят из одной формы с текстовым полем и надписью. Программа 1 запускает приложение 2 и открывает канал для этой программы. Передайте фокус ввода текстовому полю программы 1. Символы, вводимые в текстовом поле» появляются а текстовом поле программы 2.

7. Использование именованных каналов

Создать приложение, демонстрирующее обмен данными через именованные каналы — как локально, так и по сети. При этом выполнять асинхронную пересылку данных с применением перекрывающихся операций записи.

Программа создает именованный канал, в который любой подключившийся канал может записать данные. В программе имеется таймер, который периодически увеличивает счетчик и выводит его обновленное значение. Счетчик является признаком того, что главная нить приложения продолжает работать. На форме приложения 1 (сервер) находятся три кнопки, соответствующие разным режимам пересылки данных, и надпись для вывода информации о текущем состоянии (см. рисунок).

При нажатии каждой кнопки приложение пересылает блок из 200 байтов, который читается клиентским приложением (программа 2) по одному байту (чтобы замедлить процесс и таким образом подчеркнуть разницу между разными типами пересылки данных). Программа будет успешно работать только в Windows NT, поскольку Windows 95 не позволяет создавать именованные каналы.



При установке флажка «соединение» клиент в течение 10 секунд ждет, пока сервер создаст канал. Узнав о том, что канал доступен, клиент

открывает канал. Если открытие удачно, клиент включает таймер для чтения данных из канала и читает данные.

В каждом событии таймера из канала читается всего один байт. Конечно, данные читаются из канала чрезвычайно медленно, но это сделано для наглядной демонстрации отличий между синхронным и асинхронным режимом пересылки, а не для эффективного обмена данными между процессами.

Программа должна уметь поочередно обслуживать несколько клиентов.

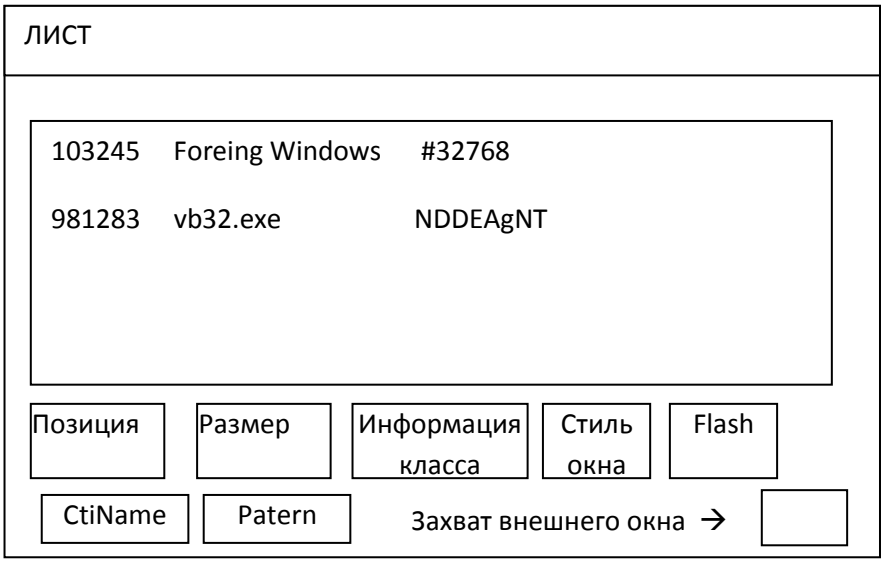
Программа должна уметь одновременно передавать данные нескольким клиентам (вам придется создать несколько экземпляров канала).

8. Просмотр информации об окнах

Программа предназначена для просмотра информации об окнах. Операции программы делятся на две категории: поиск/выбор окна и просмотр информации об окне.

Для поиска и выбора окон используется меню ЛИСТ, состоящее из пяти команд. Команда ВЕРХНИЙ УРОВЕНЬ заполняет список перечнем всех окон верхнего уровня в системе (см. рисунок).

Для каждого окна выводится хендел, имя приложения и имя класса. Команда ДОЧЕРНИЕ заполняет список перечнем всех дочерних окон текущего выделенного окна. Команда СОБСТВЕННЫЕ заполняет список перечнем всех собственных окон текущего выделенного окна. Поскольку собственные окна также могут быть окнами верхнего уровня, они могут присутствовать и в другом списке.



Команда УКАЗАТЬ позволяет навести курсор на любое окно вашего приложения и щелкнуть на нем, чтобы включить его в список.

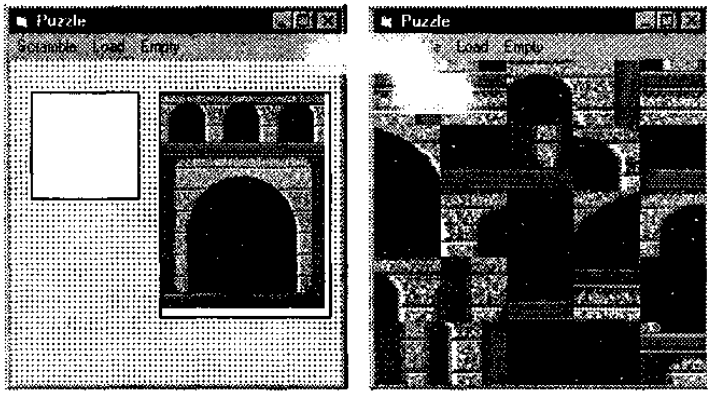
В элементе Label 1 выводится информация об окне, находящемся под курсором. Если отпустить кнопку мыши, данные об окне в текущей позиции включаются в список.
Любое окно, находящееся в списке, может быть выделено. Щелкая на кнопках, вы получаете информацию о выделенном окне. Команда ОЧИСТИТЬ стирает содержимое списка.

9. Графическая головоломка

Приложение имитирует головоломку, напоминающую известную игру «пятнадцать»!. Произвольно выбранный растр делится на 25 плиток, которые затем перемешиваются. Одна из плиток удаляется. Игрок должен воссоздать исходный растр, последовательно передвигая плитки на пустое место.

Работать с приложением очень просто. При загрузке программы в окне выводится стандартный растр (рисунок «арки» из поставки Windows). Один из квадратов сетки 5x5 закрашивается черным цветом; этот квадрат считается пустым. Если щелкнуть мышью на любой плитке, прилегающей к пустому квадрату, эта плитка перемещается на пустое место. Цель игры — восстановить исходное изображение.

Меню состоит из трех команд. Команда Scramble переставляет плитки изображения в случайном порядке. Команда Load вызывает форму для выбора нового файла растрового изображения. Выбранный растр масштабируется таким образом, чтобы он заполнял все окно приложения. Команда Empty позволяет выбрать внешний вид пустого квадрата. Он может закрашиваться черным цветом (по умолчанию), белым цветом или случайным узором.
На рисунке показана форма приложения Puzzle в режиме конструирования (слева) и эта же форма изображена во время работы (справа).



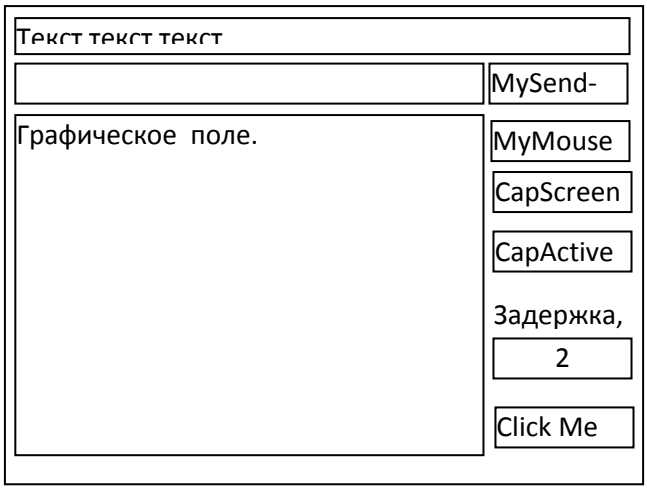
Слева расположено графическое поле Picturel, а справа — Picture2. Расположение и размеры этих полей несущественны, поскольку программа сама масштабирует их так, как потребуется.

10. Имитация нажатий клавиш и событий мыши

На рисунке изображено главное окно. На форме находятся два текстовых поля. В верхнем поле выводится исходная строка, символы которой бу-

дут имитироваться программой. Функция SendKeys работает на очень низком уровне и работает непосредственно с виртуальными клавишами. Нижнее текстовое поле всего лишь является удобным «приемником» для имитируемых клавиш — никакими другими функциями оно не обладает.

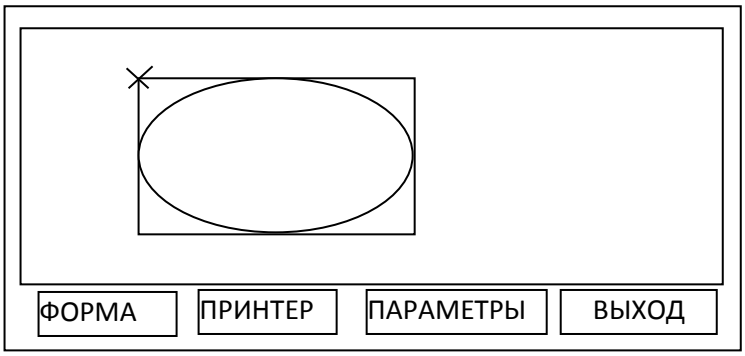
В графическом поле отображается содержимое буфера обмена (clipboard) после сохранения экрана. На форме имеются четыре кнопки, реализующие четыре разных операции. Все операции могут выполняться немедленно или после указанной задержки. По умолчанию в программе используется двухсекундная задержка. Это сделано для того, чтобы продемонстрировать общесистемный характер этих операций. Например, вы можете нажать кнопку MySendKeys с пятисекундной задержкой, переключиться в текстовый редактор и увидеть, как в нем появляются имитируемые символы.



Кнопка MyMouse имитирует операции с мышью. Курсор перемещается на кнопку Click Me и щелкает на ней. На экране появляется сообщение о щелчке. Кнопки CapScreen и CapActive управляют операцией сохранения экрана, которая также может выполняться с задержкой. Обратите внимание: на рисунке эти кнопки заблокированы. Дело в том, что сохранение экрана может быть довольно длительной операцией, и блокировка кнопок на время ее выполнения предотвращает новые щелчки и постановку в очередь новых команд.

11. Программа просмотра информации об устройстве

Программа выводит информацию о заданном устройстве. В ее выходные данные включается ВСЯ информация, предоставляемая функцией GetDeviceCaps. На рисунке изображена программа в действии. Рисунок демонстрирует расположение элементов на форме.

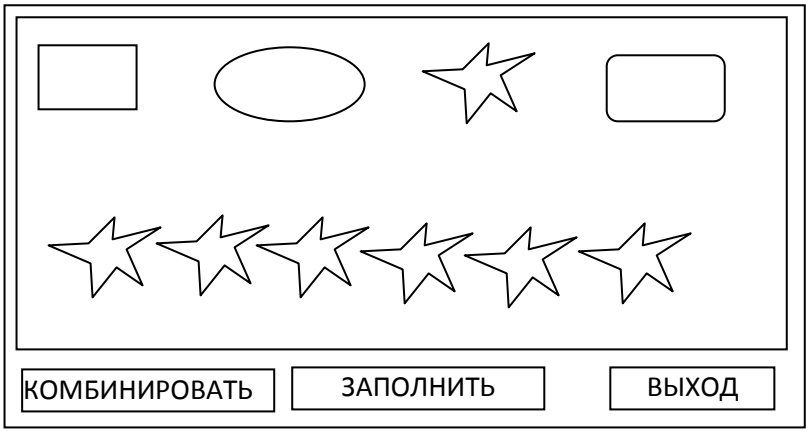


На форме этой простой программы находятся всего два элемента. Кнопка ФОРМА получает контекст устройства для формы и выводит сведения по этому контексту. Кнопка ПРИНТЕР делает то же самое для принтера по умолчанию. Кнопка ПАРАМЕТРЫ вызывает диалоговое окно позволяющее изменить параметры по умолчанию: координаты левого верхнего и правого нижнего угла области прорисовки, координаты этих же углов области просмотра и координаты этих же углов логического окна.

12. Отсечение_1

Программа должна показать, как создавать регионы, как комбинировать их и выполнять отсечение для создания интересных графических эффектов. На рисунке изображена главная форма программы.

В главном окне программы создается пять разных регионов: простой прямоугольник, эллипс, многоугольник (в нашем примере — звезда) и прямоугольник с закругленными углами. Кроме того, создается сложный регион, состоящий из нескольких многоугольников. Любой из этих регионов можно выделить щелчком мыши — выделенный регион выделяется толстой рамкой. Если вы щелкаете на сложном регионе, будьте терпеливы. Прорисовка границ может потребовать некоторого времени, особенно на медленном компьютере.



Если щелкнуть на кнопке КОМБИНИРОВАТЬ, все выделенные регионы объединяются в комплексный регион, который выбирается в качестве области отсечения формы. Обработчик события Paint этой формы рисует на

форме серию горизонтальных линий. Линии автоматически обрезаются по границам области отсечения. При нажатии кнопки ЗАПОЛНИТЬ регионы объединяются и заполняются цветной заливкой.

13. Отсечение_2

Очень простое приложение строит траекторию на основе текста и преобразует ее в область отсечения контекста устройства. Затем в контексте устройства рисуется узор, который отсекается по границам текста. На главной форме проекта находятся четыре кнопки; три из них заполняют область отсечения различными узорами, а четвертая обводит траекторию при помощи функции API StrokePath.

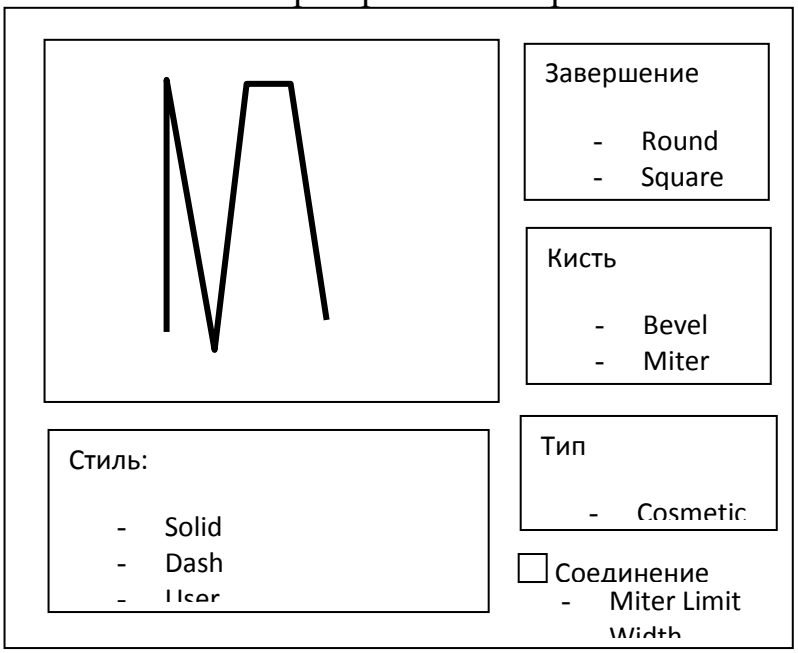
14. Перья

На рисунке показана главная форма проекта . Расширенные перья обладают некоторыми характеристиками, с которыми можно поэкспериментировать.

Стиль: сплошная линия, стандартный или пользовательский пунктирный рисунок. ExtPen позволяет выбрать сплошное или пунктирное перо и демонстрирует применение пользовательского стиля, который изменяется посредством модификации программного кода.

Тип: косметическое или геометрическое перо. Толщина косметического пера всегда равна единице. Ширина геометрического пера задается при помощи полосы прокрутки Width.

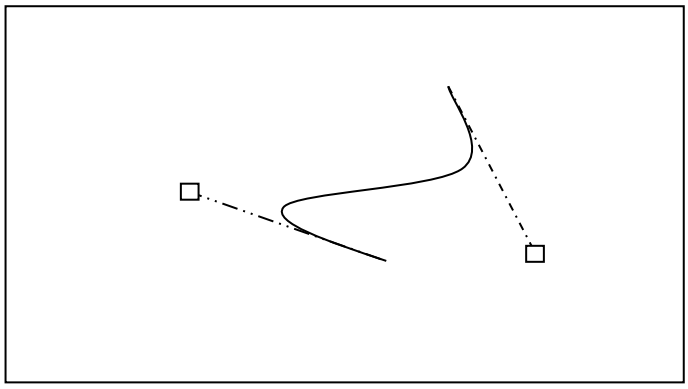
Кисть: однородная или с заданным узором. ExtPen позволяет выбрать кисть со стандартным штриховым рисунком, для этого следует установить флажок Cross Pattern. Вы можете отредактировать программу и выбрать другую стандартную кисть. Также можно воспользоваться специальной методикой, и создать кисть на основе любого растрового изображения.



Завершение: завершение линии, нарисованной геометрическим пером, может быть закругленным, квадратным или плоским. Программа позволяет выбрать нужный вариант завершения.
Соединение: Программа позволяет выбрать внешний вид соединений отрезков. Чтобы два отрезка считались соединенными, они должны быть частью либо одного объекта (например, прямоугольника), либо одной траектории. При простом рисовании линий от конечной точки отрезка эффект соединения не создается — отрезки должны быть включены в траекторию и затем прорисованы функцией StrokePath или StrokeAndFillPath.

15. Кривые Безье

Кривые Безье составляют особый тип сплайновых кривых, определяемых четырьмя точками. Поддержка кривых Безье относится к числу самых полезных новых возможностей Win32 GDI. Две точки определяют начало и конец кривой линии. Две другие (контрольные) точки определяют ее направление и кривизну. На рисунке показана главная форма программы с кривой Безье.

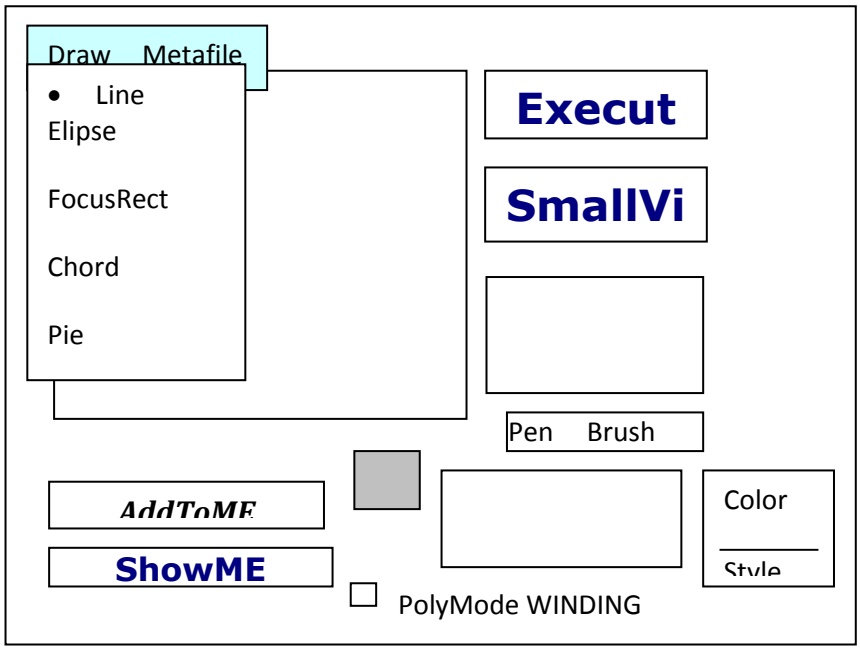


Голубые маркеры изображают начальную и конечную точку кривой Безье, а малиновые — ее контрольные точки. Пунктирные линии соединяют контрольные точки с соответствующими концами кривой. Направление пунктирной линии определяет направление, под которым кривая входит в конечную точку, а ее длина — крутизну входа. В приложении Bezier на любом маркере можно щелкнуть и перетащить его мышью, чтобы поэкспериментировать с внешним видом кривой.

16. МикроГраф

Программа реализует многие приемы, встречающиеся в полноценных графических редакторах.

На рисунке показано окно программы на стадии конструирования.



Меню Draw приложения позволяет выбрать графическую команду Windows API, выполняемую при нажатии кнопки Execute. Каждой команде в качестве параметров передаются две и более точек. Координаты точек задаются щелчками мыши в большом графическом поле. Максимальное количество точек зависит от типа выбранной фигуры; например, для линии требуются всего две точки.

Кнопка Execute рисует текущую выбранную фигуру. Если щелкнуть в большом графическом поле после нажатия кнопки Execute, приложение переходит к определению нового объекта.

При нажатии кнопки Small View текущая фигура рисуется в малом графическом поле. Фигура масштабируется так, чтобы в малом графическом поле помещалось все содержимое большого графического поля. Этот пример показывает, как преобразования системы координат используются для масштабирования изображений.

Пять полос прокрутки в правой нижней части экрана предназначены для выбора стиля пера и кисти, цвета пера и кисти, а также толщины пера для рисования. Текущая фигура рисуется с применением этих атрибутов. Изменив атрибуты, нажмите кнопку Execute или Small View, чтобы текущая фигура была нарисована с новыми атрибутами. В маленьком прямоугольном графическом поле (Pictures) рисуется прямоугольник с применением текущих графических объектов (пера и кисти).

Кнопка AddToMF добавляет текущую фигуру в глобальный метафайл. Кнопка ShowMF воспроизводит метафайл в двух графических полях, большом и малом. При помощи этих двух кнопок можно построить более сложное изображение, состоящее из нескольких фигур. Кнопка Del eteMF удаляет текущий метафайл. Флажок PolyMode WINDING выбирает режим заполнения многоугольников.

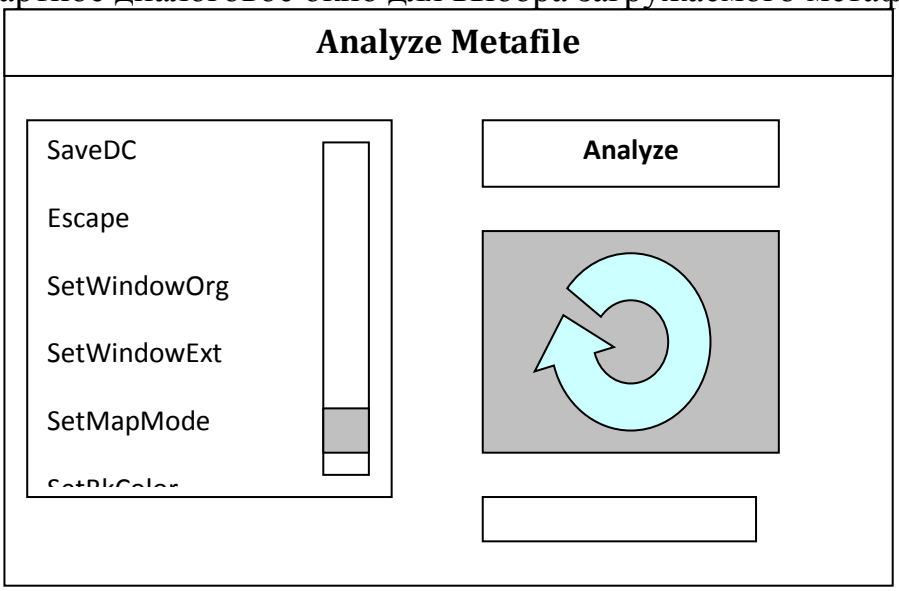
Меню Metafile содержит команды загрузки и сохранения метафайлов, а также копирования текущего глобального метафайла в буфер обмена.

17. Анализатор структуры метафайла

В программе применяется функция перечисления. Программа позволяет выбрать метафайл на диске и воспроизвести его в графическом поле. Вы можете получить полный список всех команд метафайла и даже воспроизводить его по одной команде, отбрасывая те команды, которые не должны входить в изображение.

На рисунке показана программа в действии.

В программе используется пять файлов. На главной форме, находится список, заполняемый командами метафайла, и маленькое графическое поле для его воспроизведения. Флажок Single Step определяет режим воспроизведения файла — по одной команде или все сразу. Кнопка Analyze вызывает стандартное диалоговое окно для выбора загружаемого метафайла.



Программа позволяет ввести шестнадцатеричный код любой растровой операции и быстро увидеть ее результаты для 16 доступных цветов и произвольного цвета кисти. Поскольку приемный растр после выполнения очередной операции не очищается, вы можете выполнить серию операций для получения определенного результата.

Кроме того программа позволяет экспериментировать с функцией MaskBlt.

18. Информация о системе

Программа позволяет определить текущие системные цвета, системные метрики и другие системные параметры Windows.

Список и связанная с ним надпись отображаются на экране только в режиме вывода системных цветов командой меню СИСТЕМА → ЦВЕТА.

При выделении любой строки в списке фон надписи окрашивается в выбранный системный цвет.

На форме присутствуют два текстовых поля. Когда фокус принадлежит верхнему полю, при нажатии любой клавиши в поле отображается ее имя. Если фокус ввода находится в нижнем поле, в нем создается и отображается квадратная каретка.

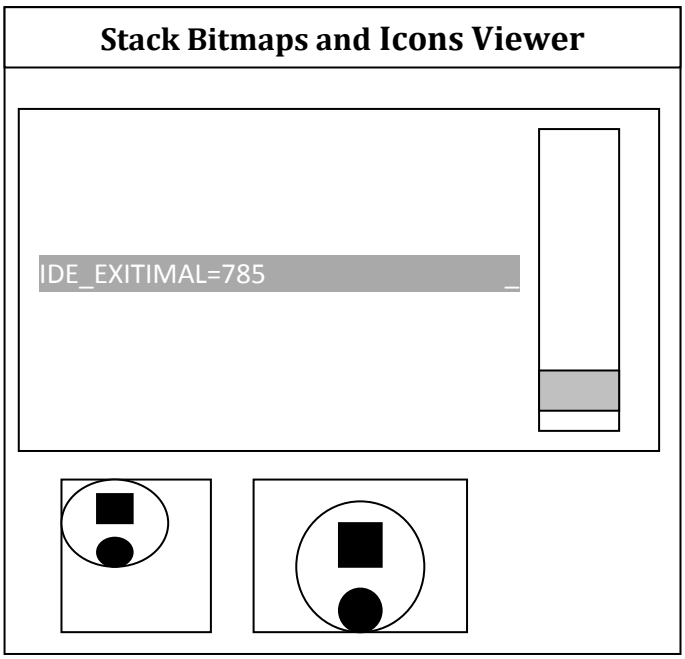
Надпись в правом нижнем углу формы используется в сочетании с таймером для вывода текущего состояния клавиш CapsLock, NumLock и ScrollLock.

На форме программы для каждой категории системных данных сделаны закладки. В каждой закладке выводится вся доступная информация по категории;

Реализуйте в программе возможность редактирования системных параметров.

19. Просмотр встроенных растров и значков

Программа предназначена для просмотра встроенных значков и растров, поддерживаемых Windows. Кроме того, она демонстрирует процесс загрузки и вывода растров и значков. На рисунке показано окно программы во время работы.

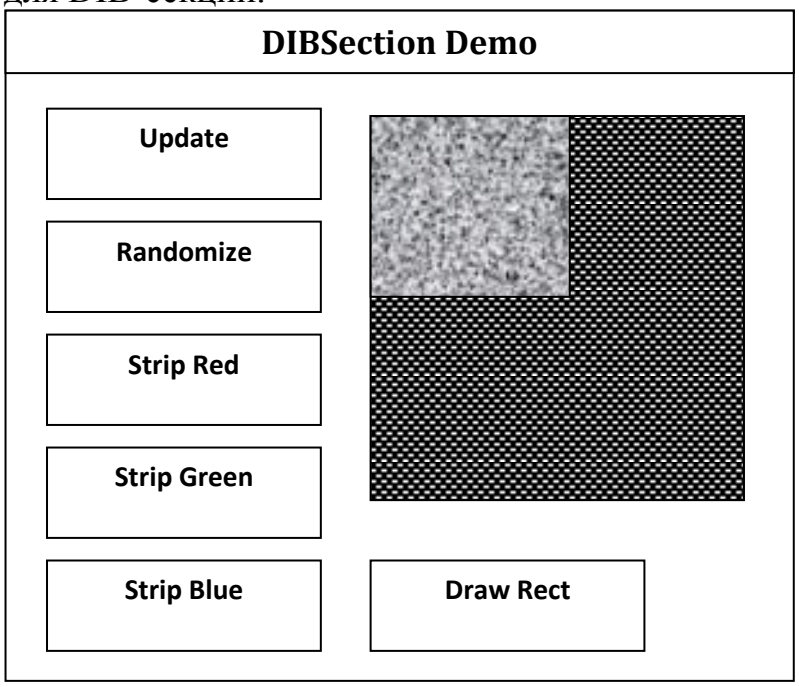


Окно программы содержит три элемента. Список заполняется определениями констант встроенных растров и значков Windows. При выделении строки списка соответствующий значок или растр загружается.и выводится в левом графическом поле. Если выделен встроенный значок, программа при помощи функции Loadlmage загружает увеличенное изображение значка и выводит его в правом графическом поле.

20. Работа с DIB-секциями

Простая программа применяет DIB-секции — объектов GDI, по своему внутреннему формату аналогичных аппаратно-независимым растрам.
На рисунке показано рабочее окно программы (хотя черно-белый со сканированный рисунок заметно снижает впечатление).

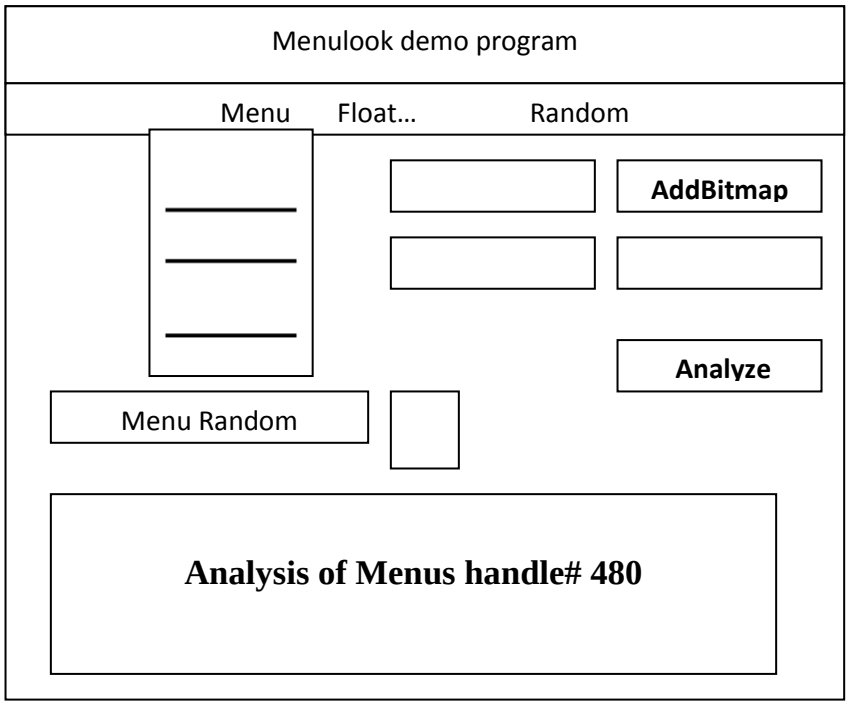
В окне программы находится одно графическое поле. При нажатии кнопки Randomize программа раскрашивает отдельные пикселы DIB-секции в случайные цвета, для этого она напрямую обращается к памяти DIB. Затем функция BitBIt переносит изображение в графическое поле. Кнопка DrawRect рисует прямоугольник в совместимом контексте устройства, в котором выбран растр DIB-секции; она демонстрирует возможность вызова графических функций GDI для DIB-секций.



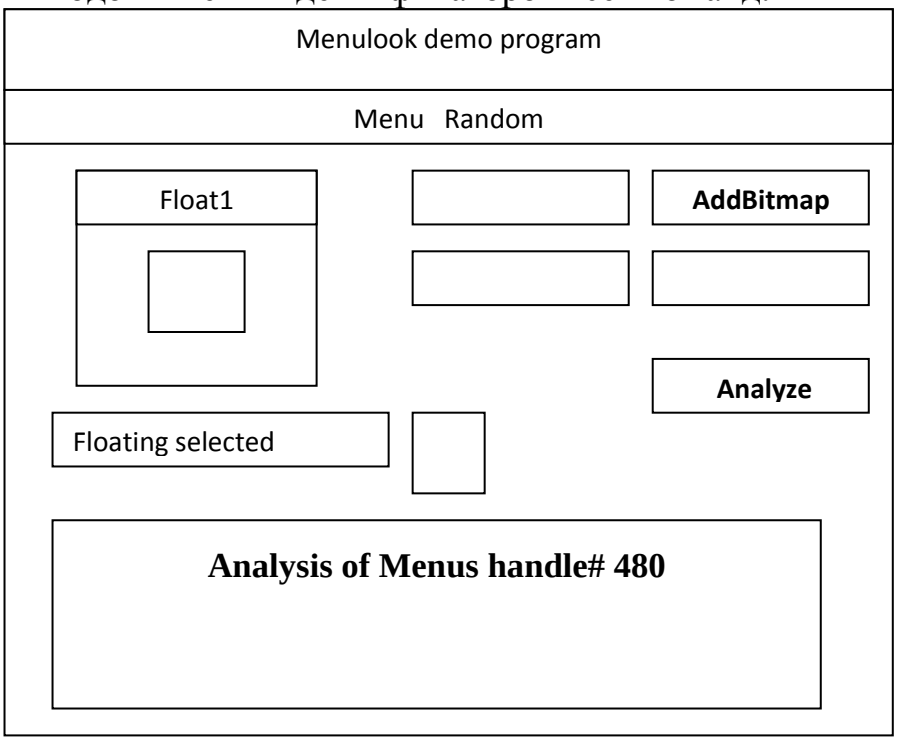
Кнопки Strip Red, Strip Green и Strip Blue выполняют простейшую обработку изображения, удаляя из него соответственно красную, зеленую и синюю цветовые составляющие. Эта задача без особых хлопот (и с лучшим быстродействием) решается применением растровых операций.

21. Просмотрщик структур меню.

В рабочем окне программы изображенном на рисунке, демонстрируются разнообразные приемы работы с меню. При нажатии кнопки Analyze производится анализ существующей структуры меню, а его результаты заносятся в список List l.



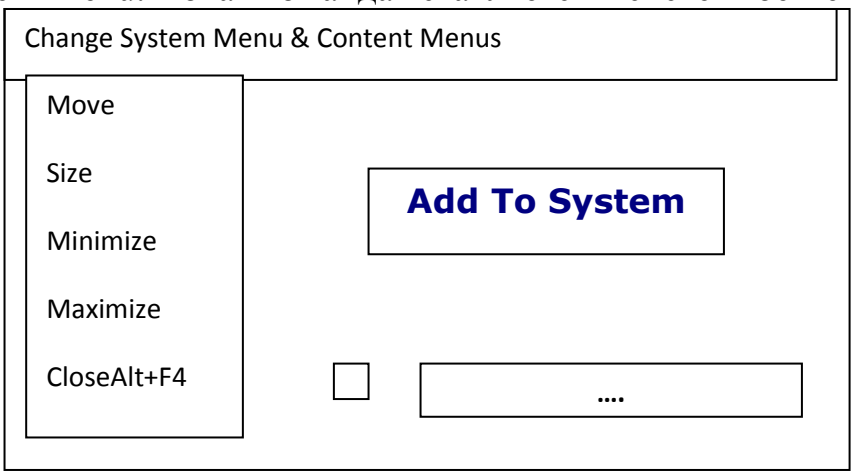
В результатах анализа приводится хендел и описание каждой команды меню. Если команда открывает подменю, его хендел выводится вместе с именем. Хендел всегда выводятся в шестнадцатеричной системе, а все остальные числа — в десятичной. Приложение поочередно анализирует все подменю с выводом имен и идентификаторов всех команд.



Кнопка Add Bitmap добавляет команду с растровым изображением в меню Floating. Используемый растр берется из свойства Picture элемента Picturel.

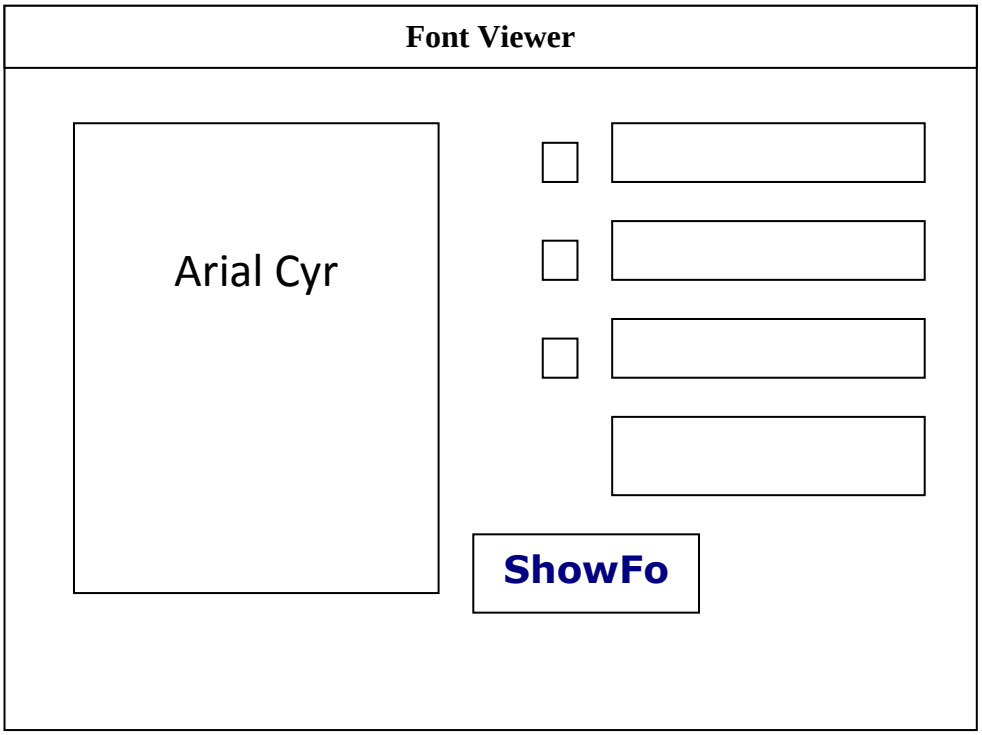
22. Редактор системного меню

Рабочее окно программы показано на рисунке. Как видите, в системном меню появилась новая команда. В текстовом поле (которое на рисунке частично скрыто раскрытым меню) можно ввести любой текст и нажать кнопку Add To System Menu. Новая команда вставляется в системное меню.

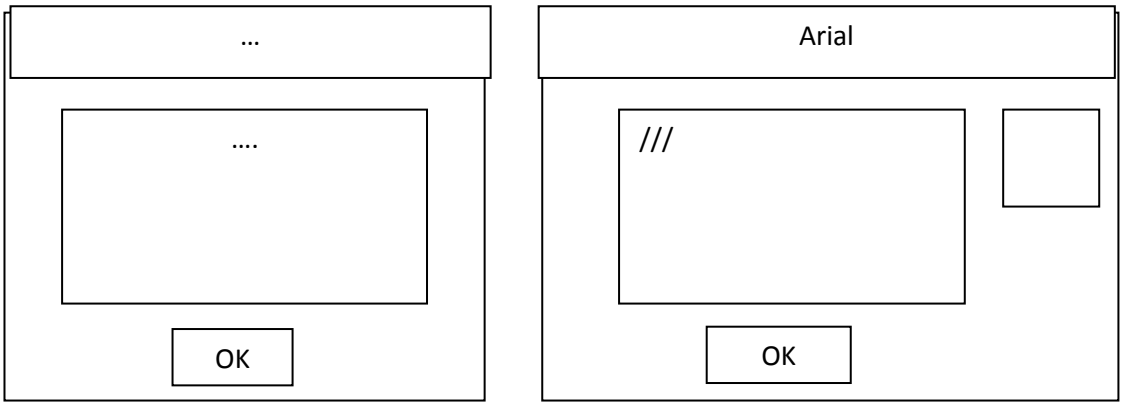


В программе используется субклассирование для перехвата сообщений WM_CONTEXTMENU формы и текстового поля, что позволяет создавать контекстные меню для элементов, не имеющих собственных меню, и переопределять контекстные меню для текстовых полей.

23. Просмотрщик системных шрифтов



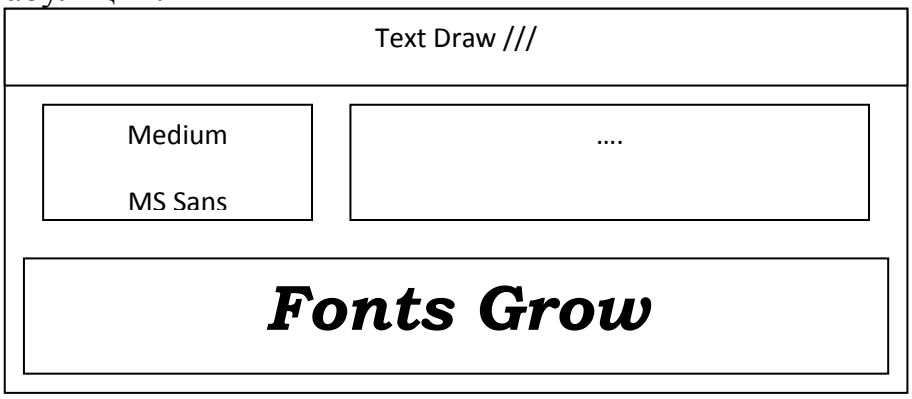
При нажатии кнопки ShowFont в графическом поле выбирается логический шрифт и выводится текст образца.



Кнопка ShowMetrics вызывает окно сообщения с информацией о физическом шрифте, который был выбран по введенным атрибутам (. Кнопка ShowInfo выводит дополнительные данные о шрифте, как показано на рисунке.

24. Вывод текста на форму

На рисунке показано рабочее окно программы. Список заполняется именами гарнитур доступных экранных шрифтов. В нижнем графическом поле выводится строка текста, в которой шрифт данного элемента, выбранный по умолчанию, масштабируется до различных размеров. В графическом поле, расположенном в правой верхней части текстовая строка выводится с автоматическим переносом слов. Здесь же продемонстрировано применение табуляций.



Единственное, что может сделать пользователь в этой программе, — выбрать отображаемую гарнитуру.

8 Перечень основной и дополнительной учебной литературы, необходимой для освоения дисциплины (модуля)

8.1 Основная литература

1 Вавренюк, А. Б. Компоненты операционных систем. Основы UNIX [Электронный ресурс] : учеб. пособие / Вавренюк А.Б., Курышева О.К., Кутепов С.В. - М. :НИЦ ИНФРА-М, 2015. - 184 с. // ZNANIUM.COM : электронно-библиотечная система. – Режим доступа: http://znanium.com/catalog.php#, ограниченный. – Загл. с экрана.

8.2 Дополнительная литература

1 Астахова, И. Ф. Компьютерные науки. Деревья, Компоненты операционных систем, сети [Электронный ресурс] / И.Ф. Астахова, И.К. Астанин, И.Б. Крыжко. - М. : ФИЗМАТЛИТ, 2013. - 88 с. // ZNANIUM.COM : электронно-библиотечная система. – Режим доступа: http://znanium.com/catalog.php#, ограниченный. – Загл. с экрана.

2 Синицын, С.В. Компоненты операционных систем: Учебник для вузов / С. В. Синицын, А. В. Батаев, Н. Ю. Налютин. - М. : Академия, 2010. - 297с.

3 Олифер, В.Г. Сетевые Компоненты операционных систем: Учебное пособие для вузов / В. Г. Олифер, Н. А. Олифер. - СПб. : Питер, 2003; 2002; 2001. - 538с.

9 Перечень ресурсов информационно-телекоммуникационной сети «Интернет» (далее – сеть «Интернет»), необходимых для освоения дисциплины (модуля)

1 Вавренюк, А. Б. Компоненты операционных систем. Основы UNIX [Электронный ресурс] : учеб. пособие / Вавренюк А.Б., Курышева О.К., Кутепов С.В. - М. :НИЦ ИНФРА-М, 2015. - 184 с. // ZNANIUM.COM : электронно-библиотечная система. – Режим доступа: http://znanium.com/catalog.php#, ограниченный. – Загл. с экрана.

2 Астахова, И. Ф. Компьютерные науки. Деревья, Компоненты операционных систем, сети [Электронный ресурс] / И.Ф. Астахова, И.К. Астанин, И.Б. Крыжко. - М. : ФИЗМАТЛИТ, 2013. - 88 с. // ZNANIUM.COM : электронно-библиотечная система. – Режим доступа: http://znanium.com/catalog.php#, ограниченный. – Загл. с экрана.

10 Методические указания для обучающихся по освоению

дисциплины (модуля)

Текущий контроль учебной деятельности студентов осуществляется на лабораторных занятиях. Студент обязан выполнить выданные ему лабораторные работы. Итоги выполнения лабораторных работ записываются в систему «Лабдиспетчер», что позволяет отследить динамику обучения каждого студента и группы в целом, а также способствует обеспечению ритмичности учебной деятельности студентов. По результатам сдачи каждой лабораторной работы присваиваются баллы. Максимальное число баллов за одну лабораторную - 10.

Студент, не выполнивший к концу сессии все лабораторные работы, не допускается до зачета. Зачет проводится по итогам текущей сдачи отчетов по выполненным лабораторным. Критерии оценки результатов обучения по дисциплине представлены в технологической карте (таблица 6).

Примерный перечень теоретических разделов курса для самостоятельного изучения:

1. Использование отладчика защищенного режима работы процессора.
2. Применение инструментария для контроля оперативной памяти ПЭВМ.
3. Изучение методов работы с утилитами контроля и администрирования дискового пространства ПЭВМ.
4. Ознакомление с принципами программирования многопоточных приложений.
5. Анализ стандартных шаблонов драйверов операционной системы.
6. Ознакомления со стандартными листингами программных модулей Разработки и управления службами операционной системы.
7. Изучение документации по организации и сопровождению файловой системы NTFS.

Примерные требования к оформлению и сдаче отчетов по лабораторным работам:

По каждой лабораторной работе должен быть составлен отчет в виде документа MS Word, содержащий следующие разделы:

- титульный лист;
- задание;
- теоретический материал, содержащий описание методики выполнения лабораторной работы;
- листинг программы или результаты лабораторных измерений и исследований;
- экранные формы;
- список использованной литературы.

Отчет в электронном виде должен быть представлен преподавателю на контроль с последующей защитой выполненной лабораторной работы на лабораторном занятии.

И дополнительная рекомендация: для тех, кто самостоятельно не может решить задачи в предложенной курсовой работе, рекомендуем воспользоваться книгой Эпплман Д. WIN32 API и VISUAL BASIC; Питер, 2001, 1120 с.

Все 24 варианта заданий там разобраны и запрограммированы, но только на языке Visual Basic. При наличии этой книги студенту только остается разобрать решение к нужному варианту и переложить его на язык С++.

11 Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), включая перечень программного обеспечения и информационных справочных систем (при необходимости)

Освоение дисциплины «Компоненты операционных систем» основывается на активном использовании пакета Visual Studio в процессе изучения теоретических разделов дисциплины и подготовки к практическим занятиям.

С целью повышения качества ведения образовательной деятельности в университете создана электронная информационно-образовательная среда. Она подразумевает организацию взаимодействия между обучающимися и преподавателями через систему личных кабинетов студентов, расположенных на официальном сайте университета в информационно-телекоммуникационной сети «Интернет» по адресу https://student.knastu.ru. Созданная информационно-образовательная среда позволяет осуществлять взаимодействие между участниками образовательного процесса посредством организации дистанционного консультирования по вопросам выполнения практических заданий. В учебном процессе по дисциплине активно используется контрольно-рейтинговая система факультета компьютерных технологий по контролю уровня выполнения лабораторных работ «ЛабДиспетчер», расположенная по адресу **http://biblserver/LD** в локальной сети ФКТ.

При изучении дисциплины, для выполнения лабораторных работ используется лицензионное программное обеспечение – пакет Visual Studio 2010.

12 Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине (модулю)

Для реализации программы дисциплины «Компоненты операционных

систем» используется материально-техническое обеспечение, перечисленное в таблице 7.

Таблица 7 – Материально-техническое обеспечение дисциплины

Аудитория	Наименование аудитории (лаборатории)	Используемое оборудование	Назначение оборудования
228/1, 303/3, 303А/3, 305/3, 312/3, 321/3	Компьютерные классы ФКТ	Компьютеры IBM PC Corel-3, 4Мб ОЗУ, 11 шт. в классе, проектор	Выполнение лабораторных работ, проведение лекций

13 Перечень программных продуктов, используемых при изучении дисциплины

Для полноценного изучения курса необходимо использование следующих лицензионных и бесплатных программных продуктов:

1. Компоненты операционных систем: Windows (Лицензионный сертификат № 46243844 от 09.12.2009)
2. Утилиты операционной системы (компоненты операционной системы).
3. Среды программирования: Visual Studio Express (LiteVare, бесплатное ПО).
4. Отладчик OlleDebuger (бесплатное ПО).
5. Виртуальная машина VMWare (Лицензия № 954630047 от 8.11.2010, государственный контракт 53-АЭ061, лицензионные ключи).
6. Текстовый процессор типа MS Word (Лицензионный сертификат № 45806198 от 19.08.2009, лицензионный сертификат № 45286522 от 25.03.2009).
7. Браузер Internet Explorer (компонент операционной системы).

Лист регистрации изменений к РПД

№ п/п	Содержание изменения/основание	Кол-во стр. РПД	Подпись автора РПД
1	Изменение листа подписей в связи со сменой декана ФКТ /пр.№ 271-ЛС «к» от 29.12.2016	1	
2	Изменение КУГ/пр. № 326-О «а» от 04.09.2017	7	
3	Изменение титульного листа в связи с переименованием вуза/пр. №997-О от 03.11.2017	1	
4	Актуализация литературы/ 28.11.2017	2	