Министерство науки и высшего образования

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» 12 2017 г.

РАБОЧАЯ ПРОГРАММА

дисциплины «Стандартизация, сертификация и качество программного обеспечения»

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 2017

Автор рабочей программы
профессор, к.т.н. _____ В.А. Тихомиров
« 15 » 05 2017 г.

СОГЛАСОВАНО

Директор библиотеки _____ И.А. Романовская
« 18 » 05 2017г.

Заведующий кафедрой «МОПЭВМ» _____ В.А. Тихомиров
« 16» 05 2017г.

Заведующий выпускающей кафедрой «МОПЭВМ» _____ В.А. Тихомиров
« 16 » 05 2017г.

Декан «ФЗДО» _____ М.В.Семибратова
« 21 » 05 2017г.

Начальник учебно-методического управления _____ Е.Е. Поздеева
« 23 » 05 2017г.

Введение

Рабочая программа дисциплины «Стандартизация, сертификация и качество программного обеспечения» составлена в соответствии с 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». 2017, 2018 годы.

1 Аннотация дисциплины

<table>
<tr><td>Наименование дисциплины</td><td colspan="8">Стандартизация, сертификация и качество программного обеспечения</td></tr>
<tr><td>Цель дисциплины</td><td colspan="8">формирование у студентов знаний, умений и навыков в указанных областях деятельности применительно к программному обеспечению.</td></tr>
<tr><td>Задачи Дисциплины</td><td colspan="8">-познакомить студентов с существующими стандартами качества ПО;
-познакомить студентов со сложившимися на сегодня различными взглядами на проблему надежности ПО и с существующими моделями надежности;
-освоение студентами программных систем, позволяющих собрать ряд статистических данных о работе программ, написанных на языке высокого уровня.</td></tr>
<tr><td>Основные разделы дисциплины</td><td colspan="8">Качество программного обеспечения
Надежность программного обеспечения
Метрология, стандартизация, сертификация программного обеспечения
Оптимизация программных модулей</td></tr>
<tr><td rowspan="3">Общая трудоемкость дисциплины</td><td colspan="8">3 з.е. / 108 академических часов</td></tr>
<tr><td rowspan="2">Семестр</td><td colspan="4">Аудиторная нагрузка, ч</td><td rowspan="2">СРС, ч</td><td rowspan="2">Промежуточная аттестация, ч</td><td rowspan="2">Всего за семестр, ч</td></tr>
<tr><td>Число недель</td><td>Лекции</td><td>Лаб. работы</td><td>Курсовое проектирование</td></tr>
<tr><td>5</td><td>17</td><td>4</td><td>6</td><td></td><td>89</td><td>9</td><td>108</td></tr>
<tr><td colspan="2">ИТОГО:</td><td>17</td><td>4</td><td>6</td><td></td><td>89</td><td>9</td><td>108</td></tr>
</table>

Трудоемкость дисциплины в таблице представлена для учебного года, когда учебный семестр составляет 17 недель.

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

Дисциплина «Стандартизация, сертификация и качество программного обеспечения» 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 обосновывать принимаемые проектные решения, осуществлять постановку и выполнять эксперименты по проверке их корректности и эффективности (ПК-3)	Нормативные материалы и стандарты по разработке ПО, З1(ПК-3-4)	Применять нормативные и стандартизованные материалы при разработке ПО, У1(ПК-3-4)	Навыками использования стандартов, справочников и нормативной документации при разработке ПО, Н1(ПК-3-4)

3 Место дисциплины (модуля) в структуре образовательной программы

Дисциплина(модуль) «Стандартизация, сертификация и качество программного обеспечения» изучается на 3 курсе в 5 семестре.

Дисциплина является обязательной, входит в состав блока Б1 «Дисциплины (модули)» и относится к вариативной части.

Для освоения дисциплины необходимы общие знания, умения, навыки работы с ПЭВМ, навыки программирования на языках высокого уровня.

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 3 зачетных единиц, 108 академических часов.

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**
Число недель в семестре номер 1/номер 2				**17**
Общая трудоемкость дисциплины				108
Контактная аудиторная работа обучающихся с преподавателем (по видам учебных занятий), всего				10
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)				6
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				89
Промежуточная аттестация обучающихся				9

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)

Наименование разделов, тем и содержание материала	Компонент учебного плана	Трудоемкость (в часах)			Форма проведения	Планируемые (контролируемые) результаты освоения	
			Для графика 17 недель в семестре			Компетенции	Знания, умения, навыки
Раздел 1 ***Стандартизация и сертификация программного обеспечения***							
Тема 1 Установочная лекция. Стандартизация ПО, метрология ПО - наука об измерениях. Виды измерений. Физические величины как объект измерений. Средства измерений. Порядок выполнения лабораторных работ. Порядок выполнения контрольной работы. Подготовка к экзамену.	Лекция		2		Традиционная	ПК-3	З1(ПК3-4)
Тема 2 Сущность стандартизации. Нормативные документы по стандартизации и виды стандартов. Цели, принципы, функции и методы стандартизации. Качество ПО. Современные модели качества: CMM, SPICE, ISO/IEC 12207. Другой взгляд на проблему создания качественного ПО – CBP. Надежность ПО. Надежность аппаратуры и надежность программного обеспечения. статистическая модель Миллса; интуитивная модель; модель последовательности испытаний Бернулли.	Лекция		2		Традиционная	ПК-3	З1(ПК3-4)

Наименование разделов, тем и содержание материала	Компонент учебного плана	Трудоемкость (в часах)			Форма проведения	Планируемые (контролируемые) результаты освоения	
			Для графика 17 недель в семестре			Компетенции	Знания, умения, навыки
Изучение системы True Time Оптимизация программного модуля с помощью системы True Time	Лабораторная работа		2		Традиционная	ПК-3	У1(ПК3-4) Н1(ПК3-4)
Изучение системы AQtime	Лабораторная работа		2		Традиционная	ПК-3	У1(ПК3-4) Н1(ПК3-4)
Оптимизация программного модуля с помощью системы AQtime	Лабораторная работа		10		Активная	ПК-3	У1(ПК3-4) Н1(ПК3-4)
Тенденции и перспективы развития систем оценки качества и надежности ПО.	Самостоятельная работа обучающихся		67		Чтение основной и дополнительной литературы, конспектирование	ПК-3	У1(ПК3-4) Н1(ПК3-4)
			22		Выполнение контр.работы	ПК-3	У1(ПК3-4) Н1(ПК3-4)
Промежуточная аттестация			9				
Итого по разделу 1			108				
ИТОГО по дисциплине	Лекции		4		-	-	-
	Лабораторные работы		6		-	-	-
	Самостоятельная работа обучающихся		89		-	-	-
	Промежуточная аттестация		9 (экзамен)				
ИТОГО: общая трудоемкость дисциплины (часов) 108 в том числе с использованием активных методов обучения 10 часов							

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

Самостоятельная работа обучающихся, осваивающих дисциплину «Стандартизация, сертификация и качество программного обеспечения», состоит из следующих компонентов: - изучение теоретических разделов дисциплины;

- подготовка к лабораторным занятиям;
- подготовка и оформление контрольной работы.

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

Комплект электронных УММ для выполнения лабораторных работ и РГР по дисциплине «Стандартизация, сертификация и качество программного обеспечения» в локальной сети ФКТ по адресу **\\3**k316m04\ Share\МОП_ЭВМ\1.Заочное\Бакалавры\ССиК.

Рекомендуемый график выполнения самостоятельной работы студента, для варианта семестра в 17 недель, представлен в таблице 4.

Самостоятельная работа студентов, реализуемая вне рамок аудиторных занятий, имеет следующую структуру:

- подготовка к лекциям;
- теоретическая подготовка к лабораторным занятиям;
- самостоятельное изучение отдельных теоретических разделов дисциплины «Стандартизация, сертификация и качество программного обеспечения»;
- выполнение контрольной работы и подготовка к ее сдаче;

При подготовке к лекциям студент должен восстановить в памяти материал, разобранный в предыдущих лекциях, и освежить навыки практического использования этого материала на лабораторных работах.

Теоретическая подготовка к лабораторным занятиям требует знания пройденного лекционного материала, предварительного изучения методического материала по предстоящей к выполнению (или защите) лабораторной работы. Необходимо дополнительно ознакомиться с аналогичными темами, проиллюстрированными в интернет сообществе и на порталах вузов – аналогов.

КнРб выполняется, начиная с середины семестра, когда у студента накапливается необходимый набор знаний и умений для её выполнения.

Таблица 4 – Рекомендуемый график выполнения самостоятельной работы студентов при 17-недельном семестре

Вид самостоятельной работы	Часов в неделю в семестре 5																				Итого по видам работ
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17				
Подготовка к лабораторным занятиям	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**	**2**				**34**
Изучение теоретических разделов дисциплины						**2**	**2**	**2**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**	**3**				**33**
Подготовка, оформление и защита контрольной работы												**2**	**4**	**4**	**4**	**4**	**4**				**22**
ИТОГО в 5 семестре	**2**	**2**	**2**	**2**	**2**	**4**	**4**	**4**	**5**	**5**	**5**	**7**	**9**	**9**	**9**	**9**	**9**				**89**

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
Изучение системы True Time Оптимизация программного модуля с помощью системы True Time	ПК-3	Лабораторная работа № 1	Умеет проводить анализ деятельности пользователя за ПВМ Умеет использовать стандартные элементы интерфейса
Изучение системы Aqtime	ПК-3	Лабораторная работа № 2	Умеет выполнять визуализацию интерфейса в среде программирования
Оптимизация программного модуля с помощью системы Aqtime	ПК-3	Лабораторная работа № 3	Умеет формировать эргономичный интерфейс
Тенденции и перспективы развития систем оценки качества и надежности ПО.	ПК-3	Выполнение КнРб	Умеет строить и тестировать программный интерфейс
Все темы	ПК-3	Вопросы к экзамену	Умеет вести сертификацию ПО и проверку его качества

Промежуточная аттестация в семестре проводится в форме экзамена.

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 экзамена***				
1	Отчеты по лабора-	В течение се-	10 баллов	10 баллов – студент правильно выполнил и защитил лабораторную работу. Показал отличные знания в рамках

	Наименование оценочного средства	Сроки выполнения	Шкала оценивания	Критерии оценивания
	торным работам (4 работы)	местра	За каждую работу	освоенного учебного материала. 5 баллов – студент выполнил и защитил лабораторную работу с небольшими неточностями. Показал хорошие знания в рамках освоенного учебного материала. 2 баллов – студент выполнил и защитил лабораторную работу с существенными неточностями. Показал удовлетворительные знания в рамках освоенного учебного материала. 0 баллов – задание не выполнено.
2	Отчет по выполнению контр.работы	В конце семестра	10 баллов	10 баллов – студент правильно выполнил задание. 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. 5 баллов – студент выполнил задание 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. 2 балла – студент выполнил задание 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. 0 баллов – при выполнении задания 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.
Текущий контроль:		-	50 баллов	
3	Экзаменационные билеты	Сессия	50 баллов	50 баллов - студент правильно выполнил практическое задание билета. Показал отличные умения в рамках освоенного учебного материала. Ответил на все дополнительные вопросы. 30 баллов - студент выполнил практическое задание билета с небольшими неточностями. Показал хорошие умения в рамках освоенного учебного материала. Ответил на большинство дополнительных вопросов. 15 баллов - студент выполнил практическое задание билета с существенными неточностями. Показал удовлетворительные умения в рамках освоенного учебного материала. При ответах на дополнительные вопросы было

	Наименование оценочного средства	Сроки выполнения	Шкала оценивания	Критерии оценивания
				допущено много неточностей. 0 баллов - при выполнении практического задания билета студент продемонстрировал недостаточный уровень умений. При ответах на дополнительные вопросы было допущено множество неправильных ответов.
Экзамен:		-	50 балл	-
ИТОГО:		-	100 балл	-
Критерии оценки результатов обучения по дисциплине: 0 – 64 % от максимально возможной суммы баллов – «неудовлетворительно» (недостаточный уровень для промежуточной 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)				

Задания для текущего контроля

Задания на лабораторные работы

Лабораторная работа 1

Провести изучение системы True Time и выполнить измерения быстродействия процедур программного модуля, выданного преподавателем по вариантам. Изучить инструменты оптимизации программного модуля с помощью системы True Time и выполнить оптимизацию программного кода процедур в модуле, выданном преподавателем по вариантам.

Лабораторная работа 2

Провести изучение системы AQtime и выполнить измерения быстродействия процедур программного модуля, выданного преподавателем по вариантам.

Лабораторная работа 3

Изучить инструменты оптимизации программного модуля с помощью системы AQtime и выполнить оптимизацию программного кода процедур в модуле, выданном преподавателем по вариантам.

Задание на контрольную работу

Задание: Оптимизировать с помощью системы Turbo Profiler программный модуль, написанный на языке СИ.

Замечание. В качестве оптимизируемой программы можно взять одну из программ, разработанных вами в курсе «Программирование на языках высокого уровня» либо в курсе «Вычислительная математика».

Типовые вопросы на экзамен

Экзамен проводится в виде тестирования. Билет имеет вид листа с тестовыми вопросами:

Примеры вопросов на экзамен:

1. Сертификация в переводе с латыни означает

а) «Официальное признание».

б) «Письменная гарантия».

в) «Сделано верно».

2. Основной формой государственного надзора является

а) выборочная проверка.

б) соблюдение обязательных требований государственных стандартов.

в) контроль за качеством и безопасностью потребительских товаров.

3. Основным достоверным способом доказательства соответствия продукции заданным требованиям является _____.

4. Процедура, посредством которой третья дает письменную гарантию, что продукция, процесс или услуга соответствуют заданным требованиям, называется _____.

5. Установить соответствие

Понятие	Определение
1) Испытание 2) Аккредитация 3) Аттестация	а) официальное признание права испытательной лаборатории осуществлять конкретные испытания. б) техническая операция, заключающаяся в определении характеристик продукции в соответствии с установленной процедурой. в) оценка состояния дел в испытательной лаборатории по определенным параметрам и критериям.

6. Установить соответствие

Данные сессии в True Time	Назначение
1) % in Function 2) % with Children 3) % in Image	а) процент времени, потраченного на выполнение данной функции от всех функций данного файла. б) процент времени, потраченного на функцию от времени, потраченного на сессию. в) процент времени, потраченного на функцию и ее дочерние функции от времени, потраченного на сессию .

7. Порядок проведения сертификации:

-Оценка производства. Затем выдача сертификата соответствия.

-Подача заявки на сертификацию. Затем отбор, идентификации образцов и их испытания .

-Корректирующие мероприятия.

-Применение знака соответствия. Затем инспекционный контроль за сертифицированной продукцией.

8. Уровни возможностей процесса в стандарте SPICE в порядке возрастания:

-Выполняемый процесс.

-Предсказуемый процесс.

-Установленный процесс.

-Управляемый процесс.

-Оптимизирующий процесс.

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

8.1 Основная литература

1 Ананьева, Т. Н. Стандартизация, сертификация и управление качеством программного обеспечения [Электронный ресурс]: учеб. пособие / Т.Н. Ананьева, Н.Г. Новикова, Г.Н. Исаев. — М. : ИНФРА-М, 2017. — 232 с. // ZNANIUM.COM : электронно-библиотечная система. – Режим доступа: http://znanium.com/catalog.php, ограниченный. – Загл. с экрана.

2 Трусов, Б. Г. Программная инженерия: Учебник для вузов / Под ред. Б.Г.Трусова. - М. : Академия, 2014. - 282с.

8.2 Дополнительная литература

1 Исаев, Г.Н. Теоретико-методологические основы качества информационных систем : монография [Электронный ресурс] / Г.Н. Исаев. — М. : ИНФРА-М, 2018. — 293 с. // ZNANIUM.COM : электронно-библиотечная

система. – Режим доступа: http://znanium.com/catalog.php, ограниченный. – Загл. с экрана.

2 Антипов, В. А. Бубнов, А. А. Пылькин, А. Н. Столчнев, В. К. Введение в программную инженерию [Электронный ресурс] : учебник / В. А. Антипов, А. А. Бубнов, А. Н. Пылькин, В. К. Столчнев. — М.: КУРС: ИНФРА-М, 2017. — 336 с. // ZNANIUM.COM : электронно-библиотечная система. – Режим доступа: http://znanium.com/catalog.php, ограниченный. – Загл. с экрана.

3 Царев, Р. Ю. Мультиверсионное программное обеспечение. Алгоритмы голосования и оценка надёжности [Электронный ресурс] : монография / Р. Ю. Царев, А. В. Штарик, Е. Н. Штарик. - Красноярск: Сиб. федер. ун-т, 2013. - 120 с. // ZNANIUM.COM : электронно-библиотечная система. – Режим доступа: http://znanium.com/catalog.php, ограниченный. – Загл. с экрана.

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

1 Единое окно доступа к образовательным ресурсам // Электронный ресурс [Режим доступа: свободный] **http://window.edu.ru/.**

10 Методические указания для обучающихся по освоению дисциплины (модуля)

Для повышения качества выживаемости знаний, задачи лабораторных работ и КнРб должны подбираться с учетом необходимости применения знаний в последующих дисциплинах.

С начала семестра необходимо на первом лабораторном занятии проводить тестирование остаточных знаний студентов. Для тестирования можно использовать задачи для лабораторных работ, включающие в качестве вспомогательных элементов, действия, связанные с тематикой программирования на языках высокого уровня.

Проведение контроля текущей успеваемости, с одной стороны, позволяет получать адекватную информацию о степени усвоения учебного материала, с другой стороны, стимулирует ритмичность учебной деятельности.

Информация о выполненных лабораторных работах отражается в рейтинговых таблицах, выкладываемых на соответствующем сайте ФКТ. Уровень рейтинга студента рассчитывается и отображается на сайте автоматически.

Контрольная работа способствует лучшему освоению практических навыков по данному предмету. Студент получает задания в начале изучаемо-

го раздела, а сдает выполненное задание после прохождения основных лабораторных занятий по данному разделу.

Качество освоения учебного материала и выполнения контрольной работы контролируется преподавателем в виде защиты лабораторных работ. На защите контрольной работы преподаватель в устной (или письменной) форме проверяет знание основных определений и положений дидактической единицы, являющейся темой КнРб, а также проверяет навыки практического использования современного программного обеспечения.

Пример порядка решения задания контрольной работы

Рассматривается программа, вычисляющая коэффициенты a[0], a[1], …, a[n-1] многочлена: $p(x) = a[0] + a[1]x + \ldots + a[n-1]x^{n-1} + x^n$, если известны его корни x[0], x[1] ,…, x[n-1].

При сборе статистических данных в Turbo Profiler число запусков программы равнялось десяти.

Исходный текст программы(v_1.cpp):

```
#include <math.h>
#include <stdlib.h>
#include <stdio.h>
#include <conio.h>
#define n 3
#define d 10
int a[n],x[n],i;
void nachalo()
{
 clrscr();
 printf("\tp(x)=a0");
 randomize();
 for(i=0;i<n;i++)
        {
         x[i]=random(d);
         if (i==0) continue;
         printf(" + a%d*x**%d",i,i);
        }
 printf(" + x**%d\n",n);
}
double p(int x,int i)
 {
  if (x==0) return a[0];
  if (i==n) return pow(x,n);
  double k,l;
  k=pow(x,i);
  l=p(x,i+1);
  return a[i]*k+l;
 }
int rec(int l)
 {
  int i,j;
  if (l==n)
        {
         int flag=1;
         double pp;
         for (i=0;i<n;i++)
                if (p(x[i],0)!=0)
                 {
                  flag=0;
                  break;
                 }
         return flag;
        }

 for(i=-d;i<d;i++)
        {
         a[l]=i;
         int r=rec(l+1);
         if (r) return 1;
        }
 }
void main(void)
 {
 nachalo();
  for(i=0;i<n;i++)
        printf("\tx%d=%d \n ",i,x[i]);
        printf("\n\n");
  if(rec(0)==0)
        {
        printf("\tNo result.");
        return;
        }
  printf("a= { ");
  for(i=0;i<n;i++)
        printf("\t%d ",a[i]);
```

```
    printf("}\n");                                  }
```

Статистические данные исходной программы(v_1.cpp):

```
Execution Profile
Total time: 78.865 sec
% of total: 8  %
Run: 10 of 10
Filter: All
Show: Time and counts
Sort: Frequency      Pass count: +++   Time: ***

#v_1#27   307605      12%  |++++++++++++++++++++++++++++++++++++++++++++++
          10.559 sec  13%  |**********************************************
#v_1#33   307605      12%  |++++++++++++++++++++++++++++++++++++++++++++++
          9.4544 sec  11%  |****************************************
#v_1#28   307584      12%  |++++++++++++++++++++++++++++++++++++++++++++++
          9.2548 sec  11%  |****************************************
#v_1#30   230688       9%  |++++++++++++++++++++++++++++++++
          8.7664 sec  11%  |**************************************
#v_1#32   230688       9%  |++++++++++++++++++++++++++++++++
          7.7772 sec   9%  |**********************************
#v_1#31   230688       9%  |++++++++++++++++++++++++++++++++
          5.9124 sec   7%  |*************************
#v_1#42   76305        3%  |++++++++++
          3.0141 sec   3%  |**********
#v_1#54   80312        3%  |++++++++++
          2.6927 sec   3%  |**********
#v_1#53   80312        3%  |++++++++++
          2.6767 sec   3%  |**********
#v_1#48   76305        3%  |++++++++++
          2.6320 sec   3%  |**********
#v_1#55   80312        3%  |++++++++++
          2.6284 sec   3%  |**********
#v_1#38   80322        3%  |++++++++++
          2.5752 sec   3%  |**********
#v_1#57   80322        3%  |++++++++++
          2.4536 sec   3%  |**********
#v_1#43   76917        3%  |++++++++++
          2.3346 sec   2%  |**********
#v_1#46   76304        3%  |++++++++++
          2.1975 sec   2%  |*********
#v_1#40   76305        3%  |++++++++++
          2.0231 sec   2%  |********
#v_1#45   76304        3%  |++++++++++
          1.7976 sec   2%  |*******
```

Из этих данных видно, что в рассматриваемой программе есть «узкие места». Данные, полученные профилировщиком, показывают пути уменьшения времени работы программы и упрощения её структуры, но некоторые строки изменить нельзя.

Например, точку выхода из функции – }:

```
#v_1#33   307605      12%  |++++++++++++++++++++++++++++++++++++++++++++++
          9.4544 sec  11%  |****************************************
```

или цикл – for (i=0;i<n;i++):

```
#v_1#42   76305        3%   |++++++++++
          3.0141 sec   3%   |**********
```

Первое изменение:

В файле v_1.cpp можно заменить следующие строки:

Номер строки	Время выполнения	Количество вызовов	Строка
54	2.6927	80312	int r=rec(l+1);
55	2.6284	80312	if (r) return 1;

Эти строки можно объединить в строку (v_2.cpp):

```
if (rec(l+1)) return 1;
```

Статистические данные, полученные после выполнения первого преобразования программы (v_2.cpp):

```
Execution Profile
Total time: 73.450 sec
% of total: 1  %
Run: 10 of 10
Filter: All
Show: Time and counts
Sort: Frequency      Pass count: +++   Time: ***

#v_2#54   84200        3%   |++++++++++
          2.0782 sec   2%   |*********
```

Строка, которая получилась в результате объединения, выполняется за 2.0782 sec и вызывается 84200 раз. Общее время выполнения программы (v_2.cpp) теперь составляет 73.450 sec, т.е. замена двух строк на одну позволила уменьшить время выполнения программы на 5,415 sec.

Второе изменение:

Теперь в файле (v_2.cpp) заменим операторы printf на cprintf, уберем в них символы табуляции и перевода строки, а также объединим строки 62 и 63; 68 и 69

Номер строки	Время выполнения	Количество вызовов	Строка
62	0.0025	30	printf("\tx%d=%d \n ",i,x[i]);
63	0.0005	10	printf("\n\n");
68	0.0001	3	printf("\t%d ",a[i]);
69	0.0000	1	printf("}\n");

на строки (v_3.cpp):

```
cprintf("x%d=%d ",i,x[i]);
```

и

```
cprintf("%d }",a[i]);
```

Статистические данные, полученные после выполнения второго преобразования программы(v_3.cpp):

```
Execution Profile
Total time: 69.191 sec
% of total: 47 %
Run: 10 of 10
Filter: All
```

Show: Time and counts
Sort: Frequency Pass count: +++ Time: ***

```
#v_3#62   30          <1%  |
          0.0000 sec  <1%  |
#v_3#72   6           <1%  |
          0.0000 sec  <1%  |
```

Время работы программы (v_3.cpp) уменьшилось. Строка, которая получилась в результате объединения строк с номерами 62 и 63, теперь выполняется за 0.0000 sec и вызывается 30 раз, а строка, которая получилась в результате объединения строк с номерами 68 и 69, теперь выполняется за 0.0000 sec и вызывается 6 раз. Внесенные изменения позволили сократить время выполнения программы (v_3.cpp) на 4,259 sec.

Третье изменение:

Теперь поместим функцию nacalo() в основное тело программы, для того чтобы программа не тратила время на вызовы данной функции.

Исходный фрагмент программы(v_3.cpp):

Номер строки	Время выполнения	Количество вызовов	Строка
11	0.0000	10	void nachalo()
12			{
13	0.0044	10	clrscr();
14	0.0000	10	cprintf("p(x)=a0");
15	0.0000	10	randomize();
16	0.0000	10	for(i=0; i<n; i++)
17			{
18	0.0000	30	x[i]=random(d);
19	0.0000	30	if (i==0) continue;
20	0.0000	20	cprintf(" + a%d*x**%d",i,i);
21			}
22	0.0000	10	cprintf(" + x**%d ",n);
23	0.0000	10	}

Преобразованный фрагмент программы(v_4.cpp):

Номер строки	Время выполнения	Количество вызовов	Строка
44	0.0000	10	void main(void)
45			{
46	0.0044	10	clrscr();
47	0.0000	10	cprintf("p(x)=a0");
48	0.0000	10	randomize();
49	0.0000	10	for(i=0; i<n; i++)
50			{
51	0.0000	30	x[i]=random(d);
52	0.0000	30	if (i==0) continue;
53	0.0000	20	cprintf(" + a%d*x**%d",i,i);
54			}
55	0.0000	10	cprintf(" + x**%d ",n);

Статистические данные, полученные после выполнения третьего преобразования программы (v_4.cpp):

Execution Profile
Total time: 66.059 sec
% of total: 45 %
Run: 10 of 10
Filter: All
Show: Time and counts
Sort: Frequency Pass count: +++ Time: ***

```
p          244710      10%  |++++++++++++++++++++++++++++++++++++++++++++
           7.6194 sec  11%  |********************************************
#v_4#14    240264      10%  |++++++++++++++++++++++++++++++++++++++++++++
           6.7483 sec  10%  |****************************************
#v_4#19    244710      10%  |++++++++++++++++++++++++++++++++++++++++++++
           6.5808 sec   9%  |****************************************
#v_4#13    244710      10%  |++++++++++++++++++++++++++++++++++++++++++++
           6.5725 sec   9%  |****************************************
#v_4#17    180198       8%  |++++++++++++++++++++++++++++++++
           6.1985 sec   9%  |************************************
#v_4#16    180198       8%  |++++++++++++++++++++++++++++++++
           5.0413 sec   7%  |******************************
#v_4#18    180198       8%  |++++++++++++++++++++++++++++++++
           4.8864 sec   7%  |******************************
#v_4#29    64512        2%  |++++++++++++
           2.9127 sec   4%  |******************
#v_4#40    66988        2%  |++++++++++++
           2.7993 sec   4%  |*****************
#v_4#24    66998        2%  |++++++++++++
           2.6911 sec   3%  |*****************
#v_4#31    63639        2%  |++++++++++
           2.1391 sec   3%  |*************
#v_4#28    63643        2%  |++++++++++
           2.0343 sec   3%  |*************
rec        66998        2%  |++++++++++++
           1.9814 sec   2%  |************
#v_4#39    66988        2%  |++++++++++++
           1.8181 sec   2%  |***********
#v_4#34    63643        2%  |++++++++++
           1.5902 sec   2%  |*********
#v_4#26    63643        2%  |++++++++++
           1.5902 sec   2%  |*********
#v_4#32    63639        2%  |++++++++++
           1.4341 sec   2%  |********
#v_4#42    66998        2%  |++++++++++++
           1.2138 sec   1%  |*******
#v_4#37    3355        <1%  |
           0.1675 sec  <1%  |*
```

Выполненное изменение позволило сократить время выполнения программы на 3,132 sec, и теперь программа (v_4.cpp) выполняется за 66.059 sec.

Четвертое изменение:

Проанализировав полученные данные, можно заметить, что больше всего времени программа тратит на вызов функции p():

```
p           244710      10%   |++++++++++++++++++++++++++++++++++++++++++
            7.6194 sec  11%   |******************************************
```

Мы знаем, что функция p() вызывается только в одном месте программы, а именно в рекурсивной функции, в её цикле:

Исходный фрагмент программы(v_4.cpp):

```
double p(int x,int i)
{
        if (x==0) return a[0];
        if (i==n) return pow(x,n);
        double k,l;
        k=pow(x,i);
        l=p(x,i+1);
        return a[i]*k+l;
}
int rec(int l)
{
         int i,j;
         if (l==n)
          {
           int flag=1;
           double pp;
           for (i=0;i<n;i++)
                if (p(x[i],0)!=0)
                {
                flag=0;
                break;
                }
           return flag;
           }
           for(i=-d;i<d;i++)
           {
           a[l]=i;
           if (rec(l+1)) return 1;
           }
}
```

Поместим функцию p() в основное тело программы (v_5.cpp):

```
int rec(int l)
{
      int i,j;
      if (l==n)
      {
            int flag=1,rez;
            double pp,x;
            for (i=0;i<n;i++)
            if (x==0)  rez=a[0];
            if (i==n) rez=pow(x,n);
            double k,l;
            k=pow(x,i);
            if ((a[i]*k+l)!=0)
            {
                  flag=0;
            }
            return flag;
      }
      for(i=-d;i<d;i++)
      {
            a[l]=i;
            if (rec(l+1)) return 1;
      }
}
```

Статистические данные, полученные после выполнения четвертого преобразования (v_5.cpp):

```
Execution Profile
Total time: 31.612 sec
% of total: 99 %
Run: 10 of 10
Filter: All
Show: Time and counts
Sort: Frequency      Pass count: +++   Time: ***

#v_5#20   240000      21%
          6.3644 sec  20%
          |++++++++++++++++++++++++++++++++++++++++++++++
          |**********************************************
```

```
#v_5#33   84200        7%  |+++++++++++++++
          2.8151 sec   8%  |*******************
#v_5#24   80000        7%  |+++++++++++++++
          2.7021 sec   8%  |*******************
#v_5#36   84210        7%  |+++++++++++++++
          2.6504 sec   8%  |*******************
#v_5#15   84210        7%  |+++++++++++++++
          2.4864 sec   7%  |*****************
#v_5#17   80000        7%  |+++++++++++++++
          2.3180 sec   7%  |***************
#v_5#23   80000        7%  |+++++++++++++++
          2.2128 sec   6%  |**************
#v_5#21   80000        7%  |+++++++++++++++
          2.1578 sec   6%  |**************
#v_5#26   80000        7%  |+++++++++++++++
          2.1027 sec   6%  |**************
#v_5#28   80000        7%  |+++++++++++++++
          2.0499 sec   6%  |**************
#v_5#34   84200        7%  |+++++++++++++++
          1.8865 sec   5%  |************
#v_5#19   80000        7%  |+++++++++++++++
          1.8237 sec   5%  |************
#v_5#64   10          <1%  |
          0.0190 sec  <1%  |
#v_5#40   10          <1%  |
          0.0044 sec  <1%  |
#v_5#31   4210        <1%  |
          0.0035 sec  <1%  |
```

Выполненное изменение позволило нам уменьшить время выполнения программы за счет того, что программа теперь не тратит время на вызов функции p(). Теперь программа (v_5.cpp) выполняется за 31.612 sec, что на 34,447 sec быстрее по сравнению с предыдущей версией программы.

Пятое изменение:

Последнее преобразование, которое мы сделаем, – это заменим строку:
if (i==n) rez=pow(x,n);
на строку:
if (i==n) rez=x*x*x;.

Статистические данные, полученные после выполнения пятого преобразования(v_6.cpp):

```
Execution Profile
Total time: 30.649 sec
% of total: 99 %
Run: 10 of 10
Filter: All
Show: Time and counts
Sort: Frequency      Pass count: +++   Time: ***

#v_6#20   240000       19%
          5.2167 sec   16%
```

```
          |++++++++++++++++++++++++++++++++++++++++++++++++
          |**************************************
#v_6#34       84200  6%     |+++++++++++++++++
        4.7078 sec     15%    |**************************************
#v_6#28       80000  6%     |+++++++++++++++++
        2.2131 sec     7%     |*****************
#v_6#23       80000  6%     |+++++++++++++++++
        2.2128 sec     7%     |*****************
#v_6#33       84200  6%     |+++++++++++++++++
        2.0535 sec     6%     |*****************
#v_6#36       84210  6%     |+++++++++++++++++
        2.0503 sec     6%     |*****************
#v_6#17       80000  6%     |+++++++++++++++++
        1.9989 sec     6%     |****************
#v_6#24       80000  6%     |+++++++++++++++++
        1.8306 sec     5%     |***************
#v_6#15       84210  6%     |+++++++++++++++++
        1.7767 sec     5%     |***************
#v_6#19       80000  6%     |+++++++++++++++++
        1.7743 sec     5%     |***************
#v_6#21       80000  6%     |+++++++++++++++++
        1.7726 sec     5%     |***************
rec     84210  6%     |+++++++++++++++++
        1.4502 sec     4%     |***********
#v_6#26       80000  6%     |+++++++++++++++++
        1.3336 sec     4%     |**********
#v_6#31       4210   <1%    |
        0.2232 sec     <1%    |*
```

До преобразования программы (v_5.cpp) рассматриваемая строка выполнялась за 2.1578 sec и вызывалась 80000 раз, после преобразования программы (v_6.cpp) строка выполняется за 1.7726 sec, хотя и вызывается такое же количество раз. Общее время работы программы уменьшилось на 0,963 sec.

Вывод. Поэтапно производя изменения в программе и сравнивая статистические данные, мы уменьшали время выполнения программы и находили те места в программе, где ещё возможны изменения в лучшую сторону. В итоге, в результате пяти изменений, время работы программы уменьшилось с 78,865 sec до 30,649 sec, т.е. примерно в 2,6 раза.

Студент, успешно выполнивший и защитивший плановые лабораторные работы и контрольную работу допускается к экзамену.

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

Освоение дисциплины «Стандартизация, сертификация и качество программного обеспечения» основывается на активном использовании систем тестирования, трассировки и отладки программ.

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 свободно распространяемое программное обеспечение – True Time (https://freesoft.ru/time_tracker) и демо-версия AQtime (https://smartbear.com/product/aqtime-pro/free-trial/). Могут также, использоваться учебные версии профилировщиков фирмы Microsoft.

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

Для реализации программы дисциплины «Стандартизация, сертификация и качество программного обеспечения» используется материально-техническое обеспечение, перечисленное в таблице 7.

Таблица 7 – Материально-техническое обеспечение дисциплины

Аудитория	Наименование аудитории (лаборатории)	Используемое оборудование	Назначение оборудования
Аудитория с компьютерами и проектором	Компьютерные классы ФКТ	Компьютеры IBM PC Corel-3, 4Мб ОЗУ, 11 шт. в классе, проектор	Выполнение лабораторных работ, проведение лекций

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	9	
3	Изменение титульного листа в связи с переименованием вуза/пр. №997-О от 03.11.2017	1	
4	Актуализация литературы/ 28.11.2017	2	