МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ
РОССИЙСКОЙ ФЕДЕРАЦИИ
Саратовский государственный университет имени Н.Г. Чернышевского

Факультет компьютерных наук и информационных технологий

УТВЕРЖДАЮ
_____
"__" _____20__ г.

Рабочая программа дисциплины

ТЕХНОЛОГИИ ПРОГРАММИРОВАНИЯ

Направление подготовки

230100 Информатика и вычислительная техника

Профиль подготовки
Вычислительные машины, комплексы, системы и сети

Квалификация (степень) выпускника
Бакалавр

Форма обучения
очная

Саратов,
2011 год

1. Цели освоения дисциплины

Целями освоения дисциплины «Технологии программирования» являются обучение систематизированному представлению о принципах построения и проектирования сложных программных систем, приобретение соответствующих навыков проектирования, с использованием современных средств совместной работы и поддержки проектов, и разработки программ на процедурно-ориентированных и объектно-ориентированных языках программирования с применением методологии объектно-ориентированного программирования.

2. Место дисциплины в структуре ООП бакалавриата

Данная учебная дисциплина входит в раздел «Профессиональный цикл. Вариативная часть» ФГОС-3.

Для изучения дисциплины необходимы компетенции, сформированные у обучающихся в результате изучения дисциплин «Информатика», «Программирование», «Математическая логика и теория алгоритмов».

Сформированные в процессе изучения дисциплины «Технологии программирования» компетенции необходимы студенту при изучении дисциплин «Вычислительная математика», «Объектно-ориентированные базы данных», «Стандартизация программного обеспечения».

3. Компетенции обучающегося, формируемые в результате освоения дисциплины «Технологии программирования»

Данная дисциплина способствует формированию следующих компетенций:

- владеет культурой мышления, способен к обобщению, анализу, восприятию информации, постановке цели и выбору путей ее достижения (ОК-1);
- готов к кооперации с коллегами, работе в коллективе (ОК-3);
- стремится к саморазвитию, повышению своей квалификации и мастерства (ОК-6);
- осваивать методики использования программных средств для решения практических задач (ПК-2);
- разрабатывать модели компонентов информационных систем, включая модели баз данных (ПК-4);
- разрабатывать компоненты программных комплексов и баз данных, использовать современные инструментальные средства и технологии программирования (ПК-5);
- знание международных стандартов в области разработки программного обеспечения, понимание методов управления жизненным циклом и качеством программного обеспечения (ПК-15);

– понимание основных парадигм и методологий программирования, особенностей языков программирования, наиболее широко используемых средств программирования (ПК-20).

В результате освоения дисциплины обучающийся должен:

Знать:

– принципы проектирования программных систем;
– методы организации процесса проектирования программного обеспечения;
– методологию объектно-ориентированного программирования;
– методы и технологические средства разработки программного обеспечения;
– методы отладки и тестирования программ;
– методы защиты программ и данных.

Уметь:

– использовать методы декомпозиции и абстракции при проектировании ПО;
– применять средства разработки программного обеспечения: инструментальные среды разработки, средства поддержки проекта, отладчики;
– документировать и оценивать качество программных продуктов;
– проектировать пользовательские интерфейсы.

Владеть:

– методами представления сопроводительной и проектной документации к программным проектам;
– навыками коллективной работы над исходными кодами, с применением систем контроля версий;
– методами использования шаблонов проектирования;
– методами объектно-ориентированного программирования;
– навыками разработки консольных и графических объектно-ориентированных приложений;
– навыками сборки комплексных программных решений.

4. Структура и содержание дисциплины «Технологии программирования»

Общая трудоемкость дисциплины составляет 5 зачетных единиц, 180 часов.

№ п/п	Раздел дисциплины	Семестр	Неделя семестра	Виды учебной работы, включая самостоятельную работу студентов и трудоемкость (в часах)			Формы текущего контроля успеваемости *(по неделям семестра)* Формы промежуточной аттестации *(по семестрам)*
				Лекции	Лабораторная работа	Самостоятельная работа	
1	**2**	**3**	**4**	**5**	**6**	**7**	**8**
1	Проблемы разработки сложных программных систем. Исторический и социальный контекст программирования.	3	1	2		1	
2	Объектно-ориентированный подход к проектированию программных систем	3	2 - 3	4	12	12	
3	Жизненный цикл и процессы разработки ПО	3	4 - 5	4	4	6	
4	Унифицированный процесс разработки и экстремальное программирование	3	6 - 7	4		4	Контрольная работа № 1 на 7 неделе
5	Анализ предметной области и требования к ПО	3	8	2		4	
6	Качество ПО и методы его контроля	3	9 - 10	4	4	8	
7	Архитектура программного обеспечения	3	11 - 12	4	8	10	
8	Образцы проектирования	3	13 - 14	4	4	8	
9	Принципы создания пользовательского интерфейса	3	15	2	4	4	
10	Компонентные технологии и разработка распределённого ПО	3	16	2		2	
11	Управление разработкой ПО	3	17 - 18	4		4	Контрольная

							работа № 2 на 18 неделе
	Промежуточная аттестация						Экзамен
	ИТОГО:		**180**	**36**	**36**	**63**	**45**

РАЗДЕЛ 1. Проблемы разработки сложных программных систем

Тема 1.1. Разработка сложных программных систем

Понятие сложной программы и отличия сложных программ от простых. Основные проблемы разработки сложных программ. Понятие информационной среды процесса обработки данных. Программа как формализованное описание процесса. Понятие о программном средстве.

Тема 1.2. Критерии оценки программных продуктов

Понятие ошибки в программном средстве. Не конструктивность понятия правильной программы. Надёжность программного средства. Технология программирования как технология разработки надёжных программных средств. Роль в обществе компьютеров и программирования, информатизация общества.

РАЗДЕЛ 2. Объектно-ориентированный подход к проектированию программных систем

Тема 2.1. Объектно-ориентированное проектирование

Введение в объектно-ориентированный подход к разработке и реализации прикладных программных систем. Использование абстракций уровня предметной области при проектировании.

Тема 2.2. Объектно-ориентированное программирование

Языки программирования, предназначенные для реализации программ в рамках объектно-ориентированного подхода. Особенности реализации объектно-ориентированного подхода не объектно-ориентированных языках программирования – сущность возможностей предоставляемых объектно-ориентированным подходом.

Тема 2.3. Механизмы объектно-ориентированного программирования

Объекты и классы. Методы и операторы. Инкапсуляция, наследование и полиморфизм, как встроенные в объектно-ориентированные языки шаблоны проектирования. Интерфейсы и множественное наследование.

Тема 2.4. Особенности объектно ориентированного программирования

Область видимости. Статические переменные и функции, статические методы и члены данные классов. Перегрузка методов и операторов. Спецификаторы

доступа в объявлениях классов. Абстрактные и конкретные классы.

РАЗДЕЛ 3. Жизненный цикл и процессы разработки ПО

Тема 3.1. Жизненный цикл программных систем

Специфика разработки программных средств. Жизненный цикл ПО, виды деятельности, роли заинтересованных лиц, процессы жизненного цикла, процесс разработки ПО

Тема 3.2. Стандарты и модели жизненных циклов ПО

Стандарты жизненного цикла ПО, модель зрелости возможностей модели жизненного цикла ПО, каскадная модель жизненного цикла, итеративная модель жизненного цикла, спиральная модель жизненного цикла. Набор стандартов, регулирующих процессы разработки ПО в целом.

РАЗДЕЛ 4. Унифицированный процесс разработки и экстремальное программирование

Тема 4.1. Процессы разработки ПО

«Тяжелые» процессы разработки, «живые» методы разработки. Проблемы долгосрочного планирования крупных программных проектов.

Тема 4.2. Формализация процессов разработки ПО

Унифицированный процесс разработки Rational (RUP). Архитектура, как основа для получения качественного ПО. Архитектура, как база для планирования работ и оценок проекта в терминах времени и ресурсов. Представление архитектуры в виде набора графических моделей на языке UML. Основные фазы жизненного цикла RUP.

Тема 4.3. Повышение эффективности процессов разработки ПО

Экстремальное программирование (XP). Живое планирование. Простые проектные решения. Разработка на основе тестирования. Парное программирование. Коллективное владение кодом.

РАЗДЕЛ 5. Анализ предметной области и требования к ПО

Тема 5.1. Анализ предметной области

Роль системного аналитика в процессе разработки ПО, схема Захмана, модели предметной области.

Тема 5.2. Формальное представление моделей предметной области

Диаграммы потоков данных, диаграммы сущностей и связей, функции ПО, требования к ПО, варианты использования, действующие лица, диаграммы

вариантов использования.

РАЗДЕЛ 6. Качество ПО и методы его контроля

Тема 6.1. Понятие качества ПО и критерии его оценки

Качество разработки. Внутреннее и внешнее качество ПО. Качество По при использовании. Стандарты систем управления качеством. Характеристики качества ПО многоуровневой модели стандарта ISO 9126 (ГОСТ Р ИСО-МЭК 9126-93): Функциональность, надёжность, удобство использования, производительность, удобство сопровождения, переносимость.

Тема 6.2. Методы контроля качества ПО

Тестирование, верификация и валидация. Виды тестирования. Схема процесса тестирования. Модульное, интеграционное и системное тестирование. Проверка свойств ПО на моделях, ошибки в ПО. Анализ и инспекция результатов в процессе разработки ПО.

РАЗДЕЛ 7. Архитектура программного обеспечения

Тема 7.1. Анализ области решений

Роль архитектора ПО при анализе области решений. Первичный выбор решений и технологий при проектировании.

Тема 7.2. Архитектура ПО

Понятия архитектуры ПО, компонентов архитектуры, представления архитектуры. Сценарий использования, методы оценки архитектуры.

Тема 7.3. Представление архитектуры ПО графическими моделями UML

Статические и динамические диаграммы. Составление сценариев использования на основе диаграмм вариантов использования. Диаграммы классов и объектов. Диаграммы компонентов и диаграммы развёртывания. Диаграммы взаимодействия и диаграммы сценариев.

РАЗДЕЛ 8. Образцы проектирования

Тема 8.1. Определение образцов проектирования.

Понятие образца или шаблона проектирования, классификация образцов проектирования. Основные свойства описывающие образец проектирования.

Тема 8.2. Образцы анализа и архитектурные стили

Концептуальные модели разработки ПО в виде образцов анализа и архитектурных стилей. Виды образцов анализа архитектурных стилей. Примеры образцов анализа архитектурных стилей. Архитектурный стиль «каналы и

фильтры», архитектурный стиль «многоуровневая система». архитектурный стиль «данные–представление–обработка».

Тема 8.3. Образцы проектирования

Виды образцов проектирования. Идиомы. Примеры образцов проектирования. Образец проектирования «подписчик».Идиома «шаблонный метод». Образцы организации. Инспекция программ по Фагану.

РАЗДЕЛ 9. Принципы создания пользовательского интерфейса

Тема 9.1. Критерии удобства использования ПО

Основные факторы удобства использования ПО. психофизиологические особенности человека. Методика проектирования, ориентированного на удобство использования.

Тема 9.2. Методы разработки пользовательских интерфейсов

Проектирование, ориентированное на удобство использования. Модель ролей пользователей, модель задач, модель содержимого интерфейса, эвристическое инспектирование интерфейса. Тестирование удобства использования

РАЗДЕЛ 10. Компонентные технологии и разработка распределённого ПО

Тема 10.1. Программные компоненты и интерфейсы

Программный компонент. Программный интерфейс. Программный контракт, предусловия и постусловия.

Тема 10.2. Компонентная модель разработки ПО

Компонентная модель, компонентная среда, базовые службы компонентной среды, распределённое ПО, прозрачность, открытость, масштабируемость, безопасность, синхронное и асинхронное взаимодействие, удалённый вызов процедур, транзакция.

РАЗДЕЛ 11. Управление разработкой ПО

Тема 11.1. Роль управления при разработке ПО

Роль и компетенция руководителя при разработке ПО. Основные деятельности, входящие в компетенцию руководителей проектов.

Тема 11.2. Задачи управления при разработке ПО

Аспекты управления ресурсами, персоналом, рисками и коммуникациями проекта. Особенности управления проектами по созданию ПО.

На лабораторных занятиях студенты получают индивидуальные и групповые задания, связанные с тематикой соответствующей занятию недели, пример которых приведен в разделе 6 настоящей программы. Задания выполняются в компьютерном классе с использованием программного обеспечения, указанного в разделе 7. Результатом выполнения индивидуальных и групповых заданий являются проектная документация с использованием графического языка моделирования UML, а также соответствующий программный код, представленный в системе контроля версий.

5. Образовательные технологии

В учебном процессе, при реализации компетентностного подхода, используются такие активные и интерактивные формы проведения занятий как модельный метод обучения, метод развивающей кооперации, разбор конкретных ситуаций, командное выполнение заданий с распределением ролей, тестирование. Широко используются мультимедийные презентации при представлении лекционного материала, а также технологии для совместного взаимодействия через интернет.

6. Учебно-методическое обеспечение самостоятельной работы студентов. Оценочные средства для текущего контроля успеваемости, промежуточной аттестации по итогам освоения дисциплины

Самостоятельная работа студентов заключается в углубленном изучении материала курса по соответствующей тематике недели с использованием научной и учебно-методической литературы. В рамках самостоятельной работы студент также готовит отчет о проделанных лабораторных работах.

Примеры тем для лабораторных работ по курсу «Технологии программирования»

1 Компиляция и компоновка (сборка) консольных приложений в различных средах разработки и языках программирования
2 Изучение особенностей сборки и работы графических приложений в C++ и Java.
3 Изучение основных механизмов объектно-ориентированного программирования (инкапсуляция, наследование, полиморфизм)
4 Работа с библиотечными классами-контейнерами
5 Особенности работы работы с файловыми потоками
6 Изучение реализации различных программных идиом и шаблонов проектирования на примере учебных задач

Примеры заданий для лабораторных работ по курсу «Технологии программирования»

1 Написать графическое приложение, демонстрирующее, на примере тестового набора классов, свойство полиморфизма.
2 Создать класс-потомок стандартного элемента управления (button, radiobutton, listbox, checkbox, combobox...), модифицировать метод отрисовки объекта этого класса.
3 Используя распределённую систему контроля версий Git, создать репозиторий файлов проекта из заданий №1 и №2. Репозитарий должен содержать минимальный набор файлов необходимых для компиляции проекта.
4 Написать приложение реализующие набор классов ипользующих шаблон проектирования Фабрика на Java со встроенным сборщиком мусора и на языке C++, в котором нет встроенного сборщика мусора.

Примеры контрольных вопросов по курсу «Технологии программирования»

1. Истина или Ложь?
   (а) И / Л : Диаграммы вариантов использования полностью отражают архитектуру будущей программной системы
   (б) И / Л : Принцип абстракции позволяет ограничить концептуальные границы программных объектов, выделяя их отличительные характеристики
   (в) И / Л : Объектно-ориентированный подход может быть пременён только в решениях, реализованных с помощью объектно-ориентированных языков программирования
   (г) И / Л : Стражи включения появились в языке программирования C++, как результат его развития по отношению к языку C
2. Приведите пример объявления класса без конструктора по умолчанию на языке C++. Реализуйте алгоритм инициализации массива из 100 элементов, объявленного класса.
3. Приведите список основных спецификаторов видимости классов, укажите особенности их применения в объявлении и наследовании классов.
4. Тестирование. Найдите и исправьте синтаксические ошибки в коде объявления класса. Напишите набор функций максимально покрывающий тестами данный класс, в соответствии с тем, как вы поняли его спецификацию на основе объявления.

```
class Driver
{
public:
        enum Mode
        {
                ReadOnly = 0;
                ReadWrite,
        };
```

```
        Driver(std::string);
        Driver(const Driver&);
        virtual ~Driver();

        bool Open(std::string, Mode);
        void Close();

        std::string Read();
        void Write(std::string);
}
```

5. Опишите принципы простоты и ортогональности интерфейсов сложных систем. Приведите пример простого и адекватного интерфейса для модуля хранения данных системы библиотечного хранилища.
6. Дайте определение классу образцов проектирования «идиома», приведите несколько примеров такого образца.
7. Приведите и опишите семь различных видов тестирования. Укажите в деталях разницу между модульным, интеграционным и системным тестированием.
8. Приведите список диаграмм UML, с которыми вы знакомы, и дайте каждому из них краткое описание.

Для получения допуска к экзамену по данной дисциплине обучающемуся необходимо выполнить четыре лабораторных и две контрольных работы

Лабораторные работы оформляются в виде компилируемых и рабочее приложение или библиотеку исходных кодов в распределённой системе контроля версий Git. Репозиторий с исходными кодами публикуется на кафедральном сервере (http://git.toiit.sgu.ru), в интернете или предоставляется локально на внешнем накопителе. Уведомление о публикации результатов проводится через систему дистанционного образования на основе Moodle (http://course.sgu.ru), через почтовую рассылку курса в интернет, либо во время проведения занятий по выполнению лабораторных работ.

Контрольные работы оформляются в виде теста с использованием контрольных вопросов.

Экзаменационные вопросы по курсу «Технологии программирования»

1 Сложные и «большие» программные продукты. Отличительные особенности и основные свойства сложных программных продуктов.
2 Программная инженерия. Организационные, инженерные и технические аспекты разработки ПО
3 Модули и подсистемы. Понятие интерфейса модуля и подсистемы. Внутреннее и внешнее окружения.
4 Контекст времени исполнения при взаимодействии модулей и подсистем. Предусловия и постусловия в интерфейсах.
5 Принципы работы со сложными системами: абстракция, модульность,

переиспользование. Разбиение системы на модули.

6 Принципы работы со сложными системами: адекватность, полнота, ортогональность и простота интерфейсов, разделение ответственности.
7 Жизненный цикл ПО: виды деятельности, артефакты и роли.
8 Модели жизненных циклов ПО: каскадная, итеративная и спиральная модели
9 Унифицированный процесс разработки Rational. Основные модели и диаграммы UML
10 Экстремальное программирование. Принципы «живой» разработки ПО
11 Анализ предметной области. Схема Захмана.
12 Диаграммы потоков данных. Диаграммы сущностей и связей
13 Выделение и анализ требований. Потребности, функции и требования к ПО
14 Варианты использования и действующие лица. Диаграммы вариантов использования.
15 Качество ПО. Набор стандартов ISO для оценки качества разработки ПО Стандарт качества технологических процессов разработки ПО.
16 Методы контроля качества ПО. Верификация, валидация, тестирование. Ошибки ПО.
17 Архитектура ПО, компоненты и представление архитектуры, методы оценки архитектуры.
18 Виды UML диаграмм. Диаграммы классов, сценариев, компонентов, Диаграммы взаимодействия и развёртывания
19 Образцы проектирования и их классификация. Шаблоны образцов проектирования.
20 Примеры образцов анализа и архитектурных стилей: образец анализ «величина», архитектурные стили «каналы и фильтры», «многоуровневая система, «данные–представление–обработка»
21 Образец проектирования – идиома «шаблонный метод»
22 Образцы организации и образцы процессов, инспекция программ по Фагану
23 Объектно-ориентированная разработка ПО. Инкапсуляция, наследование, полиморфизм.
24 Объектно-ориентированные языки программирования.
25 Агрегация, обобщение, наследование.
26 Объекты, классы, методы, операторы, перегрузка.
27 Область видимости. Статические переменные и функции, статические методы и члены данные классов.
28 Спецификаторы доступа в классах. Спецификаторы доступа при наследовании.
29 Абстрактные и конкретные классы, множественное наследование.
30 Выявление асинхронного параллелизма.
31 Распределение модулей и подсистем по процессам и задачам.
32 Управление глобальными ресурсами и программным обеспечением.

7. Учебно-методическое и информационное обеспечение дисциплины «Технологии программирования»

а) основная литература

1. *Кулямин В. В.* Технологии программирования. Компонентный подход. — М.: Интернет-Ун-т Информ. Технологий, 2007 ; М.: БИНОМ. Лаб. знаний, 2007.
2. *Синельников Е. А.* Курс. Технологии программирования. — 2010.) http://course.sgu.ru/course/view.php?id=16 (доступен гостевой вход)

б) дополнительная литература

1. *Терехов А. Н.* Технологии программирования: учебное пособие — М.: Интернет-Ун-т Информ. Технологий, 2007 ; М.: БИНОМ. Лаб. знаний, 2007.
2. *Липаев В. В.* Программная инженерия. Методологические основы. — М.: ТЕИС, 2006.
3. *Вязовик Н. А.* Программирование на Java. — М.: Интернет-Университет Информ. Технологий, 2003. http://www.intuit.ru/department/pl/javapl/
4. *Мацяшек Л.А., Лионг Б.Л.* Практическая программная инженерия на основе учебного примера. — М.: БИНОМ. Лаб. знаний, 2009.
5. *Жоголев Е. А.* Технология программирования. — М.: Науч. Мир, 2004.
6. *Непейвода Н. Н.* Основания программирования. — Ижевск: Изд. Ин-та компьютер. исслед., 2003.
7. *Леоненков А. В.* Самоучитель UML. — СПб.: БХВ-Петербург, 2004.
8. *К. Бек.* Экстремальное программирование: разработка через тестирование .. — СПб.: Питер, 2003.
9. *Кулямин В. В.* Технологии программирования. Компонентный подход. — 2006. http://panda.ispras.ru/~kuliamin/sdt-course.html http://www.intuit.ru/department/se/compprog/
10. *J. Gosling, B. Joy, G. Steele, and G. Bracha.* Java Language Specification, 3-rd edition. Addison Wesley Professional, 2005. http://people.toiit.sgu.ru/Sinelnikov/PT/Java/langspec-3.0.pdf http://java.sun.com/docs/books/jls/download/langspec-3.0.pdf
11. C# Language Specification. Standard ECMA-334. 4-th edition, June 2006. http://people.toiit.sgu.ru/Sinelnikov/PT/dotNet/Ecma-334.pdf http://www.ecma-international.org/publications/files/ECMA-ST/Ecma-334.pdf
12. *Жоголев Е. А.* Введение в технологию программирования (конспект лекций). — М.: ДИАЛОГ-МГУ, 2004. http://lib.mexmat.ru/books/69295
13. *Леоненков А. В.* Нотация и семантика языка UML. — 2005.

http://www.intuit.ru/department/pl/umlbasics/
14. *Мейер Б.* Основы объектно-ориентированного программирования. — 2005. http://www.intuit.ru/department/se/oopbases/
15. Guide to the Software Engineering Body of Knowledge — 2004. http://www.computer.org/portal/web/swebok
16. *G. Keefer.* Extreme Programming Considered Harmful for Reliable Software Development. AVOCA Technical Report, 2002. http://people.toiit.sgu.ru/Sinelnikov/PT/Keefer/ExtermeProgramming.pdf
17. Kroll, The Spirit of the RUP. — 2001. http://people.toiit.sgu.ru/Sinelnikov/PT/IBM/developerworks/rational/TheSpiritoftheRUPDec01.pdf http://www.ibm.com/developerworks/rational/library/content/RationalEdge/dec01/TheSpiritoftheRUPDec01.pdf
18. *M. C. Paulk, B. Curtis, M. B. Chrissis, and C. V. Weber.* Capability Maturity Model for Software, Version 1.1, SEI Technical Report CMU/SEI-93-TR-024, Software Engineering Institute, Pittsburgh, Feb. 1993. http://people.toiit.sgu.ru/Sinelnikov/PT/SEI/reports/93tr024.pdf http://www.sei.cmu.edu/reports/93tr024.pdf
19. *M. C. Paulk, C. V. Weber, S. M. Garcia, M. B. Chrissis, and M. Bush.* Key Practices of the Capability Maturity Model, Version 1.1, SEI Technical Report CMU/SEI-93-TR-025, Software Engineering Institute, Pittsburgh, Feb. 1993. http://people.toiit.sgu.ru/Sinelnikov/PT/SEI/reports/93tr025.pdf http://www.sei.cmu.edu/reports/93tr025.pdf
20. Capability Maturity Model Integration (CMMI), Version 1.1. CMMI for Systems Engineering, Software Engineering, Integrated Product and Process Development, and Supplier Sourcing (CMMI-SE/SW/IPPD/SS, V1.1). Continuous Representation. SEI Technical Report CMU/SEI- 2002-TR-011, Software Engineering Institute, Pittsburgh, March 2002. http://people.toiit.sgu.ru/Sinelnikov/PT/SEI/reports/02tr011.pdf http://www.sei.cmu.edu/reports/02tr011.pdf
21. Capability Maturity Model Integration (CMMI), Version 1.1. CMMI for Systems Engineering, Software Engineering, Integrated Product and Process Development, and Supplier Sourcing (CMMI-SE/SW/IPPD/SS, V1.1). Staged Representation. SEI Technical Report CMU/SEI-2002-TR-012, Software Engineering Institute, Pittsburgh, March 2002. http://people.toiit.sgu.ru/Sinelnikov/PT/SEI/reports/02tr012.pdf http://www.sei.cmu.edu/reports/02tr012.pdf

в) программное обеспечение и Интернет-ресурсы

1 ОС Linux или ОС Windows;
2 консольный файловый менеджер (Far под Windows или mc под Linux)
3 компилятор C++ (GNU C++ Compiler, опционально Microsoft Visual C++)

4 комплект средств разработчика для платформы Java — Java Development Kit (JDK)
5 комплект средств разработчика для платформы QT — QT SDK (Включает в себя, под Windows, сборку MinGW с GNU C++ Compiler).
6 среды разработки (Qt Creator, NetBeans, Eclipse, Visual Studio 2008);
7 система контроля версий Git (средства интеграции со средами разработки);
8 браузер Mozilla Firefox (версии не ниже 3.6.13) с установленным плагином FireBug;
9 браузер Google Chromium.

8. Материально-техническое обеспечение дисциплины

Лекционная аудитория, оснащенная мультимедийным оборудованием для организации презентаций (компьютер с проектором и акустической системой). Лабораторная аудитория, оснащенная персональными компьютерами с необходимым программным обеспечением, подключенными к локальной сети и имеющими доступ в глобальную сеть Интернет.

Программа составлена в соответствии с требованиями ФГОС ВПО с учетом рекомендаций и Примерной ООП ВПО по направлению и профилю подготовки «Вычислительные машины, комплексы, системы и сети».

Автор
Директор саратовского подразделения
ООО «Этерсофт» _____ Е. А. Синельников

Программа одобрена на заседании кафедры дискретной математики и информационных технологий от «___» _____ 2011 года, протокол № ____.

Заведующий кафедрой
дискретной математики и
информационных технологий,
доцент _____ Л. Б. Тяпаев

Декан факультета КНиИТ,
доцент _____ А. Г. Федорова