МИНИСТЕРСТВО ОБРАЗОВАНИЯ И НАУКИ РФ
Федеральное государственное бюджетное образовательное учреждение высшего образования

«ДАГЕСТАНСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

Современные технологии программирования

Кафедра **Информатики и информационных технологий**

факультета **Информатики и информационных технологий**

Образовательная программа

09.03.02 «Информационные системы и технологии»

Профиль подготовки
Информационные системы и технологии

Уровень высшего образования
Бакалавриат

Форма обучения
очная

Статус дисциплины: **базовая**

Махачкала, 2016

Рабочая программа дисциплины составлена в 2016 году в соответствии с требованиями ФГОС ВО по направлению подготовки 09.03.02 «Информационные системы и технологии», профиль подготовки «Информационные системы и технологии» (уровень бакалавриат), утвержденного приказом Минобрнауки РФ от 12 марта 2015 г. № 219, вступил в силу 30 марта 2015 г.

Разработчик: кафедра информатики и информационных технологий,
Абдуллаев Габид Шаванович, кандидат экономических наук, доцент

Рабочая программа дисциплины одобрена:
на заседании кафедры Информатики и информационных технологий
от «2» 07 2016 г., протокол № 1

Зав. кафедрой _____ проф. Ахмедов С.А.
(подпись)

на заседании Методической комиссии факультета Информатики и информационных технологий

от «7» 07 2016 г., протокол № 1.

Председатель _____ доц. Камилов К.Б.
(подпись)

Рабочая программа дисциплины согласована с учебно-методическим управлением «7» 07 2016 г.

Аннотация рабочей программы дисциплины

Дисциплина «Современные технологии программирования» входит в базовую часть образовательной программы бакалавриата по направлению 09.03.02 «Информационные системы и технологии»

Дисциплина реализуется на факультете Информатики и ИТ кафедрой Информатики и ИТ.

Содержание дисциплины охватывает круг вопросов, связанных с изучением современных технологий и методов программирования, основных принципов объектно-ориентированного программирования, механизмов доступа к базам данных и работы с ними, приобретением практических навыков использования современных инструментальных средств для разработки, отладки и тестирования создаваемых прикладных программ.

Дисциплина нацелена на формирование следующих компетенций выпускника: общекультурных – ОК-1, общепрофессиональных – ПК-4, профессиональных – ПК-20.

Преподавание дисциплины предусматривает проведение следующих видов учебных занятий: *лекции, практические занятия, самостоятельная работа.*

Рабочая программа дисциплины предусматривает проведение следующих видов контроля успеваемости в форме *контрольной работы или коллоквиума* и промежуточный контроль в форме экзамена.

Объем дисциплины 3 зачетных единиц, в том числе в академических часах по видам учебных занятий

Семестр	Учебные занятия							Форма промежуточной аттестации (зачет, дифференцированный зачет, экзамен
	в том числе							
	Контактная работа обучающихся с преподавателем						СРС, в том числе экзамен	
	Всего	из них						
		Лекции	Лабораторные занятия	Практические занятия	КСР	консультации		
5	108	18	18	18	2	2	50	экзамен

Место дисциплины в структуре образовательной программы:

Программа дисциплины «Современные технологии программирования»разработана в соответствии с федеральным государственным стандартом высшего образования по направлению подготовки бакалавриата 09.03.02 «Информационные системы и технологии», утвержденным приказом Минобрнауки России от 12 марта 2015 г. № 219_, вступил в силу 30 марта 2015 г.

Дисциплина относится к обязательной части профессионального цикла (ПД.Б.3.6) учебного плана образовательной программы 09.03.02 «Информационные системы и технологии» профиля «Информационные системы и технологии в образовании», изучается в 5 семестре. Объем дисциплины: 3 ЗЕ / 108 часов, в том числе 54 часов - контактная работа с преподавателем,54 часа - самостоятельная работа.

Для изучения данной учебной дисциплины необходимы следующие знания, умения и навыки, формируемые предшествующими дисциплинами:

Из курса «Алгоритмические языки и системы программирования»:

Знания: ядро языка программирования высокого уровня, его синтаксис и семантику; основы проектирования программ: типовые алгоритмы.

Умения: описывать разработанные программы посредством блок схем, тестировать и отлаживать разработанные программы; реализовывать на языке программирования высокого уровня типовые алгоритмы: табуляцию функций, формирование таблиц, нахождение сумм, среднего и т.п.; поиск экстремума, работу с датчиком случайных чисел, ввод и вывод одномерных и двумерных массивов, поиск элементов в массиве, обработку массивов с выводом таблиц, сортировку, ввод и вывод текстов, сравнение фрагментов текста, изменение фрагмента текста по определенному правилу, запись информации в файл, чтение информации из файла, поиск и изменение информации в файле по заданному условию.

Владения: приемами работы в среде программирования (составление, отладка и тестирование программ; разработка и использование интерфейсных объектов)

Из курса «Высокоуровневые методы информатики и программирования»:

Знания: основные подходы к разработке программ; основные методы программирования; принципы отношения между классами – наследование, универсализация и их роль в построении программных систем.

Умения: проводить декомпозицию; использовать средства разработки для создания и отладки программного обеспечения; использовать готовые программные решения.

Владения: приемами и методами проектирования программ, основанных на классах; приемами объектно-ориентированного анализа; приемами работы в современных средах программирования.

Перечень последующих учебных дисциплин, для которых необходимы знания, умения и владения, формируемые данной учебной дисциплиной:

- Web-программирование;
- Методы и средства проектирования информационных систем и технологий.

Цели освоения дисциплины:

Подготовка к самостоятельной профессиональной работе, ознакомление с методами и технологиями программирования, умение ориентироваться во всем многообразии технологий программирования, умение применять практические навыки использования инструментальных и прикладных технологий в различных отраслях техники, экономики, управления и бизнеса.

Планируемые результаты обучения:

Дисциплина направлена на формирование компетенций ОК-1, ПК-4, ПК-20 и планируемых результатов обучения, представленных в таблице 1.

Таблица 1– Перечень планируемых результатов обучения

Компетенции	Формулировка компетенции из ФГОС ВО	Планируемые результаты обучения (показатели достижения заданного уровня освоения компетенций)
ОК-1	владение культурой мышления, способность к обобщению, анализу, восприятию информации, постановке цели и выбору путей ее достижения, умение логически верно, аргументированно и ясно строить устную и письменную речь	Знать: методику проведения системного анализа; основные правила обобщения и логического структурирования информации Уметь: грамотно и аргументировано строить устную и письменную речь Владеть: основными методами системного анализа; навыками обеспечения процесса коммуникации в организации посредством информационных технологий процессов и систем
ПК-4	способность проводить выбор исходных данных для проектирования	Знать: возможности использования ИТ в профессиональной деятельности Уметь: разрабатывать технический проект; создавать и поддерживать актуальные базы данных; подготавливать электронные ресурсы для проектируемого процесса. Владеть: основными навыками поиска и структурирования информации; стремление к самосовершенствованию, познавательную активность.
ПК-20	способность организации работы малых коллективов исполнителей	Знать: психологические особенности малых коллективов; Уметь: грамотно и рационально организовать работу исполнителей Владеть: методикой организации процесса работы малых коллективов, методиками анализа потребностей малых групп

4. Объем, структура и содержание дисциплины.

4.1. Объем дисциплины составляет 3 зачетных единиц, 108 академических часов.

4.2. Структура дисциплины.

№ п/п	**Разделы и темы дисциплины**	**Семестр**	**Неделя семестра**	**Виды учебной работы, включая самостоятельную работу студентов и трудоемкость (в часах)**	**Самостоятел ная работа**	**Формы текущего контроля успеваемости *(по неделям семестра)* Форма промежуточной**

				Лекции	Практические занятия	Лабораторные занятия	Контроль самост. раб.		аттестации *(по семестрам)*
	Модуль 1. *Введение в программирование на C#*								
1	Введение. История языка, отличительные особенности, применение в индустрии. Основные языковые конструкции. Пример кросс - платформенного исполняемого файла и его примерный формат. Связка CLR, CIL, CTS и CLS. Сборка программ на C#: Visual Studio, использование csc из командной строки, Mono.	5	1	2	2			6	
2	Синтаксис языка. Пространства имен (namespaces). Классы и методы, модификаторы const и static	5	2	2	2	2		6	Выполнение и защита лаборат. работы 1-2
	Итого по модулю 1:			4	4	4		12	Тестирование по мод
	Модуль 2. *Объектно-ориентированные конструкции C#*								
3	Ссылочные типы (reference types) и типы - значения (value types). Ссылочная семантика. Оператор == и метод object.Equals(). Передача параметров в функцию. Ключевые слова ref, out и params. Класс String	5	3	2	2	2		6	Выполнение и защита лаборат. работы 3-6
4	Массивы, ключевое слово foreach. Перечисления (enums).	5	4	2	2	2		6	
5	Наследование. Интерфейсы, абстрактные классы. Ключевые слова interface, abstract, virtual, sealed, override, new	5	5	2	2	2		6	
6	Перегрузка операторов. Арифметические операторы и индексеры (indexers). Ключевое слово foreach и метод object.GetEnumerator(). Операторы приведения	5	6	2	2	2		6	

	типов, ключевые слова implicit и explicit.								
	Итого по модулю 2:			8	8	8		24	Тестирование по мод
	Модуль 3. *Развитые особенности программирования на C#*								
7	Исключения (exceptions). Делегаты. Частичные (partial) типы и методы. Ключевое слово using и интерфейс IDisposable.	5	7	2	2	2		6	Выполнение и защита лаборат. работы 7-9
8	Потоки (threads). Примитивы синхронизации в стандартной библиотеке. Ключевое слово lock. Асинхронные делегаты.	5	8	2	2	2		6	
9	Шаблонные типы 9. (generics). Ключевые слова where и default. Пространство имен System.Collections.Generic. Часто используемые структуры данных	5	9	2	2	2		6	
	Итого по модулю 3:			6	6	6		18	Тестирование по мод
	ИТОГО:			18	18	18		54	

4.3. Содержание дисциплины, структурированное по темам (разделам).

Содержание курса

Тема 1. Введение.

Введение. История языка, отличительные особенности, применение в индустрии. Основные языковые конструкции. Пример кросс-платформенного исполняемого файла и его примерный формат. Связка CLR, CIL, CTS и CLS. Сборка программ на C#: Visual Studio, использование csc из коммандной строки, Mono.

Тема 2. Синтаксис C#

Синтаксис языка. Пространства имен (namespaces). Классы и методы, модификаторы const и static.

Тема 3. Ссылочные типы и типы-значения

Ссылочные типы (reference types) и типы-значения (value types). Ссылочная семантика. Оператор == и метод object.Equals(). Передача параметров в функцию. Ключевые слова ref, out и params. Класс String.

Тема 4. Массивы, коллекции

Массивы, ключевое слово foreach. Перечисления (enums).

Тема 5. Элементы ООП

Наследование. Интерфейсы, абстрактные классы. Ключевые слова interface, abstract, virtual, sealed, override, new.

Тема 6. Приведение типов, перегрузка операторов

Перегрузка операторов. Арифметические операторы и индексеры (indexers). Ключевое слово foreach и метод object.GetEnumerator(). Операторы приведения типов, ключевые слова implicit и explicit.

Тема 7. Прочие особенности языка

Исключения (exceptions). Делегаты. Частичные (partial) типы и методы. Ключевое слово using и интерфейс IDisposable.

Тема 8. Сборка мусора

Сборщик мусора (garbage collector) в .NET. Поколения. Финализаторы (finalizers) и метод object.Finalize().

Тема 9. Многопоточность

Потоки (threads). Примитивы синхронизации в стандартной библиотеке. Ключевое слово lock. Асинхронные делегаты.

Тема 10. Шаблонные типы

Шаблонные типы (generics). Ключевые слова where и default. Пространство имен System.Collections.Generic. Часто используемые структуры данных. **Модуль 1.Понятия программной инженерии**

Темы практических занятий

1. Счетчик строк в проекте Command-line утилита для подсчета строк кода в проекте, отфильтровывающая пустые строки и комментарии. Опционально — сделать архитектуру плагинов, позволяющую описывать структуру комментариев для произвольного языка.

Программа должна принимать параметром командной строки тип учитываемых файлов (например, *.cs). После запуска она должна обходить текущую директорию и все вложенные директории, искать в них файлы заданного типа и подсчитывать в них количество осмысленных строк. Суммарное количество строк во всех учтенных файлах должно быть выведено на экран.

Количество баллов: 5

Ключевые слова: работа с файловой системой; для системы плагинов — рефлексия или наследование

2. JSON-сериализатор Написать свой JSON-сериализатор для произвольных объектов.

Должна быть возможность корректно сериализовать, например, объект такого класса:

```
[Serializable] class TestClass
{
public int i;
public string s;
[NonSerializable]
public string ignore; // это поле не должно сериализоваться
public int[] arrayMember;
}
```

и получить на выходе что-нибудь вроде

```
{ "i": 25, "s": "Hello world!", "arrayMember": [1, 2, 3, 4, 5] }
```

Ваш сериализатор должен ориентироваться на атрибуты System.SerializableAttribute и System.NonSerializableAttribute.
Подробнее о формате JSON можно прочитать здесь: http://en.wikipedia.org/wiki/JSON
Количество баллов: 5

Ключевые слова: атрибуты, рефлексия

3. RSS2Email Программа должна периодически проверять какой-нибудь RSS-поток и пересылать все новые записи в нём по какому-либо email-адресу. Опционально можно оформить программу в виде win32-сервиса.

Все инструменты для парсинга XML, отправки почты и осуществления HTTP-запросов есть в стандартной библиотеке .NET.

Подсказка: самый простой RSS-поток, который можно обновлять самому — лента ваших твитов на twitter.com. Еще можно попробовать Tumblr и LiveJournal.

Количество баллов: 10

Ключевые слова: XML, SMTP, HTTP, многопоточность

4. Task scheduling library Написать собственный планировщик задач, наподобие библиотеки Quartz.NET (http://quartznet.sourceforge.net).

Ваша библиотека должна уметь выполнять произвольные методы по расписанию (например, каждые N минут) и с задержкой (например, выполнить это метод ровно через M секунд). Опционально — добавить поддержку расписаний в стиле crontab (http://en.wikipedia.org/wiki/Cron#Examples).

Подсказка: интерфейс библиотеки может выглядеть следующим образом:

```
interface IJob
{
void Execute(object argument);
}
public void ScheduleDelayedJob(IJob job, TimeSpan delay);
public void SchedulePeriodicJob(IJob job, TimeSpan period);
public void SchedulePeriodicJob(IJob job, string cronExpression);
```

Ключевые слова: многопоточность, межпоточное взаимодействие

Задача №1: Hello World Revenge

Имеется база данных под управлением MS SQL 2008. В ней находится единственная таблица с полями (Id, Value). В таблице единственное значение (1, "Hello World Revenge").

Необходимо реализовать WCF-сервис, который получит доступ к этой БД через LINQ2SQL, вытянет эту строчку, отрендерит из неё картинку (обычный PNG, в котором на белом фоне будет написана строчка произвольно выбранным шрифтом), после чего вернет ее как поток.

Создать WPF-графическое приложение, в котором будет ссылка этот сервис, которое по нажатию кнопки будет обращаться к сервису, получать картинку и отображать ее на экране.

Задача №2: Склад документов

Некоторой организации понадобилась система работы с документами: заявлениями, квитанциями и прочими.

Основной use-case: пользователь может зайти на страничку, увидеть список документов, которые уже есть в системе, скачать каждый документ в виде PDF, картинки или HTML.

Второй основной use-case: пользователь может выбрать какой документ ему создать (из списка всех доступных документов), ввести данные необходимых полей и сохранить

документ в системе, после чего он появится в списке и его можно будет скачать, как и все
остальные.
Узкий момент в том, что заказчик еще сам не знает, какие документы будут в системе, так
что архитектурно надо предусмотреть возможность быстро добавить поддержку еще одного документа.
Программисты решили сделать сначала прототип без авторизации пользователей и редактирования документов, а так же без дизайна. В качестве наиболее подходящего технологического стека были выбраны ASP.NET MVC, MongoDB и Stimulsoft Reports. Для
фронтенда планируется использовать Twitter Bootstrap и jQuery.
Подумав, программисты решили отдать прототип данного проекта на реализацию студенту ФИТ, т.к. задача довольно простая.
Помогите этой команде программистов.
Ключевые слова: ASP.NET MVC, работа с документно-ориентированными БД, практика
применения атрибутов, практика reflection и написания своих атрибутов, ASHX

Задача №3: Заглушки
Некому брокеру очень хочется разработать dashboard для своих клиентов, где они смогут
увидеть свои активы, котировки, посмотреть телефон консультанта, адрес отделения на
карте, историю сделок за определенные периоды и прочие статистические данные. Брокер
заботливо подготовил мокапы web-интерфейса, однако не спешит говорить, откуда разработчикам брать данные для отображения. Говорил что-то про web-сервисы, но обсуждение и согласование подвисло на стадии определения источников данных на стороне заказчика. Несмотря на это печальное обстоятельство, клиенту уже хочется видеть схематичный интерфейс (Twitter Bootstrap) с тестовыми данными. Менеджмент
проекта принимает очевидное решение сделать слой доступа к данным на заглушках, возвращающих тестовые данные, в надежде позже просто подменить реализации. Делать
это они решили посредством IoC-контейнера (Unity Application Block, при том обязательно на xml-конфигурации). Реализуйте пожелания менеджмента.
Ключевые слова: ASP.NET MVC, n-tier архитектура, разработка с использованием IoC,
Unity

Контрольные вопросы
В некоторых случаях (например, для уточнения или повышения оценки студента) преподаватель может предложить студенту ответить на несколько контрольных вопросов.
Список таких вопросов приведен ниже.
1. Какие из следующих выражений объявляют переменные значимых типов (value type)?

```
int x = 10;
string s = "строка";
object o = 20;
int y = new int();
int? z = null;
```

```
object o2 = y;
object o3 = new object();
```
2. Что выведет представленный ниже код?
```
 static class Program
 {
 static const int NaturalMin = 1;
 static bool IsNatural(int x)
 {
 Console.Write("IsNatural({0}) ", x);
 return x >= NaturalMin;
 }
 static readonly string EmptyString = "";
 static bool IsEmpty(string s)
 {
 Console.Write("IsEmpty({0}) ", s);
 return s == EmptyString;
 }
 static void Main(string[] args)
 {
 int x = 10;
 string s = "";
 Console.WriteLine(IsNatural(x) || IsEmpty(s));
 }
 }
```
3. Что выведет представленный ниже код?
```
struct Sample
 {
 public int i;
 }
 class MyProgram
 {
 static void Main()
 {
 Sample x = new Sample();
 x.i = 10;
 fun(x);
 Console.Write(x.i + " ");
 }
 static void fun(Sample y)
 {
 y.i = 20;
 Console.Write(y.i + " ");
 }
 }
```
4. Что выведет представленный ниже код?

```
struct Sample
 {
 public int i;
 }
 class MyProgram
 {
 static void Main(string[] args)
```

```
{
Sample x = new Sample();
x.i = 10;
fun(ref x);
Console.Write(x.i + " ");
}
public static void fun(ref Sample y)
{
y.i = 20;
Console.Write(y.i + " ");
}
}
```

5. Что выведет представленный ниже код?

```
public class A
{
public A()
{
Console.WriteLine("1");
Print();
}
public virtual void Print()
{
Console.WriteLine("2");
}
}
public class B: A
{
public B()
{
Console.WriteLine("3");
Print();
}
public new void Print()
{
Console.WriteLine("4");
}
}
static class Program
{
static void Main(string[] args)
{
A b = new B();
b.Print();
}
}
```

6. Оцените неэффективность приведенного кода. Сколько лишних объектов он создаст? Как бы вы переписали этот код?

```
static void Main(string[] args)
{
string text = "Hello world!";
string reverse = string.Empty;
foreach (char c in text)
{
```

```
 reverse = c + reverse; // concantenate the characters in a reverse mode
 }
 Console.WriteLine(reverse);
}
```

7. Что произойдет при компиляции и выполнении следующего кода?

```
public class Generic<T>
{
 public T Field;
 public void TestSub()
 {
 T i = Field + 1;
 }
}
class MyProgram
{
 static void Main(string[] args)
 {
 Generic<int> gen = new Generic<int>();
 gen.TestSub();
 }
}
```

8. Какие преимущества и недостатки вы видите в следующих пяти реализациях паттерна singleton?

```
public sealed class Singleton
{
 private static Singleton instance=null;
 private Singleton()
 {
 }
 public static Singleton Instance
 {
 get
 {
 if (instance==null)
 {
 instance = new Singleton();
 }
 return instance;
 }
 }
}
public sealed class Singleton
{
 private static Singleton instance = null;
 private static readonly object padlock = new object();
 Singleton()
 {
 }
 public static Singleton Instance
 {
 get
 {
 lock (padlock)
```

```
 {
 if (instance == null)
 {
 instance = new Singleton();
 }
 return instance;
 }
 }
 }
}
public sealed class Singleton
{
 private static Singleton instance = null;
 private static readonly object padlock = new object();
 Singleton()
 {
 }
 public static Singleton Instance
 {
 get
 {
 if (instance == null)
 {
 lock (padlock)
 {
 if (instance == null)
{
 instance = new Singleton();
 }
 }
 }
 return instance;
 }
 }
}
public sealed class Singleton
{
 private static readonly Singleton instance = new Singleton();
 static Singleton()
 {
 }
 private Singleton()
 {
 }
 public static Singleton Instance
 {
 get
 {
 return instance;
 }
 }
}
public sealed class Singleton
```

```
{
 private Singleton()
 {
 }
 public static Singleton Instance { get { return Nested.instance; } }

 private class Nested
 {
 static Nested()
 {
 }
 internal static readonly Singleton instance = new Singleton();
 }
}
```

9. Не компилируя этот код, скажите, вызовет ли он ошибки или warning-и при компиляции. Если да, то какие именно?

```
using System;
 namespace Polymorphism
 {
 class A
 {
 public void Foo() { Console.WriteLine("A::Foo()"); }
 }
 class B : A
 {
 public void Foo() { Console.WriteLine("B::Foo()"); }
 }
 class Test
 {
 static void Main(string[] args)
 {
 A a;
 B b;
 a = new A();
 b = new B();
 a.Foo(); // output --> "A::Foo()"
 b.Foo(); // output --> "B::Foo()"
 a = new B();
 a.Foo(); // output --> "A::Foo()"
 }
 }
 }
```

10. Сколько байт памяти занимает объект класса Derived?

```
namespace Test
{
 class Baseclass
 {
 private int i;
 protected int j;
 public int k;
 }
 class Derived: Baseclass
 {
```

```
 private int x;
 protected int y;
 public int z;
 }
 class MyProgram
 {
 static void Main (string[ ] args)
 {
 Derived d = new Derived();
 }
 }
}
```

11. Какие ошибки вы видите в этом коде?

```
switch (s) {
 case "a":
 try {
 HitA();
 }
 catch (Exception e) {
 throw e;
 }
 break;
 case "b":
 try {
 HitB();
 }
 catch (Exception e) {
 throw e;
 }
 break;
}
```

12. Какие из следующих участков кода открывают файл для записи?

```
File.Open("somefile.txt", FileMode.Create);
File.Open("somefile.txt", FileMode.Create, FileAccess.Write);
File.Open("somefile.txt", FileMode.Create, FileAccess.Read);
FileInfo file = new FileInfo("somefile.txt");
file.Open(FileMode.Create);
```

13. Какие из следующих участков кода запускает сборку, скачанную из Интернета?

```
object [] hostEvidence = {new Zone(SecurityZone.Internet)};
Evidence e = new Evidence(hostEvidence, null);
AppDomain d = AppDomain.CreateDomain("MyDomain", e);
d.ExecuteAssembly("Assembly.exe");
object [] hostEvidence = {new Zone(SecurityZone.Internet)};
AppDomain d = AppDomain.CreateDomain("MyDomain");
Evidence e = new Evidence(hostEvidence, null);
d.Evidence = e;
d.ExecuteAssembly("Assembly.exe");
AppDomain myDomain = AppDomain.CreateDomain("MyDomain");
myDomain.ExecuteAssembly("Assembly.exe", new Zone(SecurityZone.Internet));
Evidence e = new Evidence();
e.AddHost(new Zone(SecurityZone.Internet));
AppDomain myDomain = AppDomain.CreateDomain("MyDomain");
myDomain.ExecuteAssembly("Assembly.exe", e);
```

5. **Образовательные технологии**

Основными образовательными технологиями проведения курса «Технологии программирования» являются:

- Лекции, сопровождаемые компьютерными презентациями;
- лабораторные работы, в рамках которых составляются и тестируются программы, иллюстрирующие теоретический материал лекций;
- самостоятельная работа студентов, включающая усвоение теоретического материала, поиск дополнительного материала и эффективных способов выполнения заданий, завершение выполнения лабораторных работ; оформление и подготовка к защите лабораторных работ, подготовка к текущему контролю знаний и к итоговому экзамену;
- разработанные индивидуальные задания для самостоятельной работы;
- рейтинговая технология контроля учебной деятельности студентов для обеспечения их ритмичной работы в течение семестра
- консультирование студентов по вопросам учебного материала и выполнения курсового заданий.

6. Учебно-методическое обеспечение самостоятельной работы студентов

Таблица – Технологическая карта самостоятельной работы студента

№	Темы дисциплины	Задания для самостоятельной работы	Коды результатов обучения, на которые направлены задания	Трудоёмкость задания, часы	Перечень учебно-методического обеспечения
1.	Проблемы разработки сложных программных систем	Выполнить тест №1	*ОК-1.Б.3.5-з*	2	Работа с источниками 2, 3.
2.	Жизненный цикл программного обеспечения	Выполнить тест №2	*ОК-1.Б.3.5-у*	*4*	Работа с источниками 1, 3, 5.
3.	Унифицированный процесс разработки программного обеспечения	Выполнить тест №3	*ОК-1.Б.3.5-д*	*6*	Использовать источники 1, 4, 5 и Интернет-ресурсы.

4.	Экстремальное программирование	Выполнить тест №4	*ПК-20.Б.3.5-д*	8	Использовать источники 4, 6 и Интернет-ресурсы
5.	Анализ предметной области	Выполнить тест №5	*ПК-4.Б.3.5-в*	8	Использовать источники 1,4, 5 и Интернет-ресурсы
6.	Качество программного обеспечения	Выполнить тест №6	*ПК-4.Б.3.5-у*	8	Использовать источники 3,4, 5 и Интернет-ресурсы
7.	Подготовка к экзамену. Сдача экзамена.			36	Все темы курса
Итог				72	

1.2 Контроль результатов освоения дисциплины

Текущий контроль успеваемости осуществляется путем оценки результатов выполнения заданий лабораторных, самостоятельной работ, посещения лекций.

Промежуточная аттестация осуществляется в форме экзамена, который выставляется по результатам проверки выполнения тестов и заданий.

Оценочные средства результатов освоения дисциплины, критерии оценки выполнения заданий представлены в разделе «Фонды оценочных средств для проведения промежуточной аттестации» и фонде оценочных средств образовательной программы.

7. Фонд оценочных средств для проведения текущего контроля успеваемости, промежуточной аттестации по итогам освоения дисциплины

7.1. Перечень компетенций с указанием этапов их формирования в процессе освоения образовательной программы.

Перечень компетенций с указанием этапов их формирования приведен в описании образовательной программы.

Компетенция	Знания, умения, навыки	Процедура освоения
ОК-2 ПК-1	Знать …	Устный опрос, письменный опрос
ПК-7, ПК-17	Уметь …	Письменный опрос
ПК-6, ПК-7, ПК-17, ПК-19	Владеть …	Круглый стол
	Владеть …	Мини-конференция

7.2. Описание показателей и критериев оценивания компетенций, описание шкал оценивания.

ОК-1

Схема оценки уровня формирования компетенции «_____» (приводится содержание компетенции из ФГОС ВО)

Уровень	Показатели (что обучающийся должен продемонстрировать)	Оценочная шкала		
		Удовлетворительно	Хорошо	Отлично
Пороговый				

ПК-7

Схема оценки уровня формирования компетенции «_____» (приводится содержание компетенции из ФГОС ВО)

Уровень	Показатели (что обучающийся должен продемонстрировать)	Оценочная шкала		
		Удовлетворительно	Хорошо	Отлично
Пороговый				

…

…

Если хотя бы одна из компетенций не сформирована, то положительная оценки по дисциплине быть не может.

7.3. Типовые контрольные задания

1. Задание {{ 1 }} ТЗ № 1

Признаки "небольшой" программы

□□решение четко поставленной, несущественной для практической деятельности задачи

□□решение одной или нескольких значимых для пользователей задач, не имеющих четкой постановки

□□периодически требуется доработка программы с появлением новых версий

□□для выполнения своих задач программа должна взаимодействовать с другими программами

□□есть существенная необходимость в документировании программы

2. Задание {{ 2 }} ТЗ № 2

Признаки "небольшой" программы

□□отсутствует необходимость в документировании программы

□□низкая производительность приносит существенный ущерб

□□для выполнения своих задач программа должна взаимодействовать с другими программами

□□в разработку вовлечено большое количество людей

3. Задание {{ 3 }} ТЗ № 3

Признаки сложной программной системы (программного комплекса)

□□для выполнения своих задач программа должна взаимодействовать с другими программами

□□в разработку вовлечено большое количество людей

□□неправильная работа программы наносит ощутимый ущерб

□□отсутствует необходимость в оптимизации производительности программы

4. Задание {{ 4 }} ТЗ № 4

Свойства сложных программных систем

□□низкая производительность приносит существенный ущерб

□□требуется документация для обучения пользователей

□□отсутствие проектной документации

□□ущерб от неправильной работы программы незначителен

5. Задание {{ 5 }} ТЗ № 5

Свойства сложных программных комплексов

□□удобство в использовании программы носит существенный характер

□□наличие проектной документации

□□в разработке участвует один человек

□□система решает одну четко поставленную задачу

6. Задание {{ 6 }} ТЗ № 6

Виды документации, требуемой для эксплуатации и развития программной системы

□□пользовательская

□□технический

□□инженерный

□□технологический

8. Задание {{ 8 }} ТЗ № 8

Системная инженерия изучает следующие аспекты создания программно-аппаратных систем …

□□разработка программно-аппаратных систем

□□эксплуатация программно-аппаратных систем

□□интеграция программной и аппаратной составляющих

□□разработка аппаратных устройств

9. Задание {{ 9 }} ТЗ № 9

Аспекты организации экономически эффективной работы

□□организация совместной работы коллектива разработчиков

□□учет требований к пользовательским свойствам программы

□□учет квалификации пользователя при проектировании пользовательских интерфейсов

□□создание безошибочно работающего программного продукта

10. Задание {{ 10 }} ТЗ № 10

Объективные причины отсутствия "безошибочно работающих" сложных программных систем

□□противоречие требований друг другу

□□изменение требований с течением времени

□□сложность поиска ошибок в коде программы

□□сложность исправления найденных ошибок

11. Задание {{ 11 }} ТЗ № 11

Сложные программные системы с точки зрения наличия в них ошибок условно делятся на …

□□достаточно качественные

□□недостаточно качественные

□□правильные

□□неправильные

12. Задание {{ 12 }} ТЗ № 12

Основные проблемы разработки сложных программных систем связаны с нахождением разумного компромисса между затратами на разработку и … ее результата

Правильные варианты ответа: качеством; качество;

13. Задание {{ 13 }} ТЗ № 13

К наиболее важным ресурсам при оценке затрат на программу относятся …

□□время выполнения проекта

□□бюджет проекта

□□персонал

□□стоимость оборудования

14. Задание {{ 14 }} ТЗ № 14

Функциональные возможности, надежность, гибкость, удобство внесения изменений являются аспектами … программной системы

Правильные варианты ответа: качества; качество;

15. Задание {{ 15 }} ТЗ № 15

К процессам создания программных систем относятся понятия …

□□жизненный цикл

□□качество

□□процесс разработки

□□разработка документации

1. **Перечень основной и дополнительной учебной литературы, необходимой для освоения дисциплины.**

1. Акчурин Э.А. Программирование на языке C# в MS Visual Studio .Net или SharpDevelop. Учебное пособие. Самара, ИУНЛ. ПГУТИ, 2011, 150 с.
2. Нэш. C# 2010. Ускоренный курс для профессионалов. М: ИД Вильямс, 2010. 592с.
3. Макки А. Введение в .NET 4.0 и Visual Studio 2010 для профессионалов. Пер. с англ. - М.: "Вильямс", 2010. 412с.
4. Нейгел К. и др. C# 2008 и платформа .Net 3.5 для профессионалов. – М. Диалектика, 2009, 1392 с.

Дополнительная литература

1. Рихтер Дж. CLR via C#. Программирование на платформе Microsoft .NET Framework 2.0 на языке C#. Пер. с англ. - М.: «Русская Редакция» ; СПб. : Питер , 2007. 656 стр.
2. Макаров А. и др. CIL и системное программирование в Microsoft.NET: – М. : Интернет-УИТ, 2006. 328 с.

Методические указания и материалы по видам занятий

1. Акчурин Э., Ильин А. Программирование на языке C#. ЛР в ИСР Visual C# 2010 Express или SharpDevelop. . Самара, ИУНЛ. ПГУТИ, 2011, 114 с.

Программное обеспечение по видам занятий

Программное обеспечение для выполнения лабораторных работ:

- Visual Studio .Net Express Edition 2010.

9. Перечень ресурсов информационно-телекоммуникационной сети «Интернет», необходимых для освоения дисциплины.

1. http://www.microsoft.com/msf
2. http://www.uml.org

3. http://www.wikipedia.org

10. Методические указания для обучающихся по освоению дисциплины.

Критерии и показатели сформированности компетенций

Степень (уровень) сформированности компетенций на этапе изучения дисциплины «Технологии программирования» оценивается по следующим критериям: мотивационно-ценностный, когнитивный, операционно-деятельностный. Показателями критериев являются результаты обучения по дисциплине (дескрипторы) таблицы 1. Инструментарий, этапы измерения показателей и критериев компетенции представлены в таблицах 7 – 9.

Таблица 7 – Критерии и показатели сформированности компетенции ОК-1

Критерии сформированности компетенции	Показатели критериев - контролируемые результаты обучения	Способы оценки	
		Этапы контроля	Средства оценки
Мотивационно-ценностный критерий	ОК-1.Б.3.6-д	2, 5, экзамен	1
Когнитивный критерий	ОК-1.Б.3.6-з	1, 2, экзамен	1
Операционно-деятельностный критерий	ОК-1.Б.3.6-у	2	1
	ОК-1.Б.3.6-в	2, 5, экзамен	1
Интегральная оценка		Экзамен	

Таблица 8 – Критерии и показатели сформированности компетенции ПК-4

Критерии сформированности компетенции	Показатели критериев - контролируемые результаты обучения	Способы оценки	
		Этапы контроля	Средства оценки
Мотивационно-ценностный критерий	ПК-4.Б.3.6-д	3, экзамен	1
Когнитивный критерий	ПК-4.Б.3.6-з	2, 3	1
Операционно-деятельностный критерий	ПК-4.Б.3.6-у	2, 3, экзамен	1
	ПК-4.Б.3.6-в	3, экзамен	
Интегральная оценка		Экзамен	1

Таблица 9 – Критерии и показатели сформированности компетенции ПК-20

Критерии сформированности компетенции	Показатели критериев - контролируемые результаты обучения	Способы оценки	
		Этапы контроля	Средства оценки
Мотивационно-ценностный критерий	ПК-20.Б.3.5-д	4, 6, 7, экзамен	1
Когнитивный критерий	ПК-20.Б.3.5-з	4, 5, 6, 7	1
Операционно-деятельностный критерий	ПК-20.Б.3.5-у	4, 5, 7	1
	ПК-20.Б.3.5-в	6, экзамен	
Интегральная оценка		Экзамен	1

ФОНД ОЦЕНОЧНЫХ СРЕДСТВ ДЛЯ ПРОВЕДЕНИЯ ПРОМЕЖУТОЧНОЙ АТТЕСТАЦИИ

Этапы контроля: раздел 2 (самостоятельная работа), раздел 3 (самостоятельная работа), раздел 4 (самостоятельная работа), раздел 5 (самостоятельная работа), раздел 6 (самостоятельная работа), раздел 7 (самостоятельная работа), экзамен.

Время на выполнение: 60 мин.

Метод оценивания: автоматизированный

Критерии оценки результатов выполнения: менее 50% правильных ответов - неудовлетворительно, менее 65% - удовлетворительно, менее 75% хорошо, 75% и более – отлично.

11. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине, включая перечень программного обеспечения и информационных справочных систем.

Информационные технологии

Образовательный процесс осуществляется с применением локальных и распределенных информационных технологий (таблица 4, 5).

Таблица 4 – Локальные информационные технологии

Группа программных средств	Наименование программного продукта
Офисные программы	MicrosoftOffice
	LibreOffice
Распознавание текста и речи	ABBYY FineReader
Средства разработки	Borland C++ builder 6

Таблица 5 – Распределенные информационные технологии

Группа	Наименование
Система тестирования	Система сетевого компьютерного тестирования

	ДГУ www.ts.icc.dgu.ru
Библиотеки и образовательные ресурсы	Электронная библиотека ДГУ http://www.elib.dgu.ru
	Кафедральные сайты ДГУ http://cafedra.dgu.ru
	Сайте электронных образовательных ресурсов ДГУ http://umk.dgu.ru
Система электронного обучения	Сервер электронного обучения moodle http://moodle.dgu.ru

12. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине.

Таблица 6 – Материально-техническая база

Помещения для осуществления образовательного процесса	Перечень основного оборудования (с указанием кол-ва посадочных мест)	Адрес (местоположение)
Аудитории для проведения лекционных занятий		
Лекционные аудитории	Интерактивная доска, ноутбук; проектор. Количество посадочных мест – 30.	Ауд. 3-14, 4-16, 2-10, учебный корпус № 83, г.Махачкала, ул. Джержинского, 12.
Аудитории для проведения лабораторных занятий, контроля успеваемости		
Компьютерный класс	Компьютеры с выходом в Интернет и доступом в электронную информационно-образовательную среду вуза. Количество посадочных мест – 15.	Компьютерный зал № 1 учебный корпус № 3, г.Махачкала, ул. Джержинского, 12.
Помещения для самостоятельной работы		
Компьютерные классы	Компьютеры с выходом в Интернет и доступом в электронную информационно-образовательную среду вуза. Количество посадочных мест – 15+12=27.	Компьютерный зал № 2, № 3, учебный корпус № 3, г. Махачкала, ул. Джержинского, 12.
Читальный зал библиотеки ДГУ	Компьютеры с выходом в Интернет и доступом в электронную информационно-образовательную среду вуза. Количество посадочных мест – 30.	Электронный читальный зал научной библиотеки ДГУ, г. Махачкала, ул. Батырая, 4