МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФГБОУ ВО «КАБАРДИНО-БАЛКАРСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ им. Х.М. БЕРБЕКОВА (КБГУ)»

Институт информатики, электроники и робототехники

Кафедра «Мехатроника и робототехника»

СОГЛАСОВАНО

Руководитель образовательной программы

_____ Х.М. Сенов

«30» 05 2023 г.

УТВЕРЖДАЮ

И.о. директора ИИЭиР

_____ Р.Ш. Тешев

«30» 05 2023 г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ (МОДУЛЯ)

Б1.О.08.06 «ПРОГРАММИРОВАНИЕ В ТЕХНИЧЕСКИХ СИСТЕМАХ»

Направление подготовки
15.03.06 Мехатроника и робототехника

Профиль подготовки
Промышленная робототехника и робототехнические системы

Квалификация (степень) выпускника
Бакалавр

Форма обучения
очная

Нальчик 2023

Рабочая программа предназначена для преподавания дисциплины вариативной части блока 1 студентам очной формы обучения по направлению подготовки 15.03.06 Мехатроника и робототехника в 4 семестре.

Рабочая программа составлена в соответствии с рабочим учебным планом и Федеральным государственным образовательным стандартом высшего образования по направлению подготовки 15.03.06 «Мехатроника и робототехника» № 1046, утвержденного приказом Министерства образования и науки РФ «17» августа 2020 г.

СОДЕРЖАНИЕ

1 Цели и задачи освоения дисциплины

Цель учебной дисциплины «Программирование в технических системах» является подготовка студентов к решению задач по компьютерному управлению мехатронными и робототехническими системами, формирование навыков по разработке и отладке программного обеспечения для технических систем.

Курс «Программирование в технических системах» ставит перед собой следующие задачи:

- ознакомление студентов с современными подходами к разработке и отладке программного обеспечения мехатронных и робототехнических систем;
- формирование навыков создания программного обеспечения для микроконтроллеров и операционной системы Windows;
- формирование навыков программирования на языках высокого (C, C++, C#) и низкого уровня (ассемблер) для управления мехатронными и робототехническими системами.

2 Место дисциплины в структуре ОПОП ВО

Дисциплина «Программирование в технических системах» относится к вариативной части дисциплин.

Дисциплина является базой для изучения последующих дисциплин профессионального цикла.

Изучение дисциплины базируется на фундаментальных знаниях в области математики, информатики и дискретной математики.

Полученные при изучении данной дисциплины знания используются при изучении дисциплины «Проектирование роботов и РТС», а также в дипломном проектировании.

3 Требования к результатам освоения содержания дисциплины

Процесс изучения дисциплины направлен на формирование элементов следующих компетенций по данному направлению подготовки:

ОПК-14 Способен разрабатывать алгоритмы и компьютерные программы, пригодные для практического применения.

ОПК-14.1 Способен разрабатывать алгоритмы и блок-схемы алгоритмов для решения задач в области мехатроники и робототехники. для программирования мехатронных и робототехнических устройств и систем..

ОПК-14.2 Способен разрабатывать компьютерные программы для управления роботами и решения задач в области мехатроники и робототехники.

В результате изучения дисциплины студент должен:

Знать:

- основы синтаксических конструкций современных языков программирования (**З2**);
- шаблоны проектирования высокоуровневого программного обеспечения, применяющихся для управления и моделирования РТС (**З3**);

Уметь:

- применять основные методы проектирования сложных систем программного обеспечения с использованием объектно-ориентированного подхода (**У2**);

Владеть:

- навыками работы в комплексных средах создания программного обеспечения (**В1**);
- навыками написания алгоритмов и на языках программирования высокого уровня (**В2**);
- навыками проектирования сложных систем с использованием объектно-

ориентированного подхода (**В3**).

4 Содержание и структура дисциплины (модуля)

4.1 Содержание разделов дисциплины

№ разд.	Наименование раздела	Содержание раздела	Формируемая компетенция (часть компетенции)	Форма текущего контроля
1.	Основные принципы и методология разработки прикладного программного обеспечения.	Основные принципы и методология разработки прикладного программного обеспечения мехатронных и робототехнических систем на базе алгоритмических языков программирования различного уровня.	ОПК-14.1	Тестирование Вопросы на экзамене Практические занятия
2.	Языки программирования, классификация. Языки программирования низкого и высокого уровней, как инструмент для реализации управляющих функций в мехатронных и робототехнических системах.	Обзор языков программирования. Языки программирования низкого уровня «ассемблер», языки высокого уровня С++, С#. Обзор средств разработки и отладки программ - Microsoft Visual Studio 2010, описание интерфейса программы. Инсталляция и настройка прикладного программного обеспечения.	ОПК-14.2	Тестирование Вопросы на экзамене Практические занятия Лабораторные работы
3.	Синтаксис языков программирования	Синтаксис языков программирования С++, С#. Структура языков, обзор основных операторов и конструкций, на примере написания программ.	ОПК-14.2	Тестирование Вопросы на экзамене Практические занятия Лабораторные работы
4.	Подход объектно-ориентированного программирования при разработке ПО для управления мехатронными и робототехническими системами	Объектно-ориентированное программирование, назначение и основные понятия. Инкапсуляция, наследование, полиморфизм. Особенности написания программ на языке объектно-ориентированного и объектного программирования.	ОПК-14.2	Тестирование Вопросы на экзамене Практические занятия Лабораторные работы

4.2 Структура дисциплины

Общая трудоемкость дисциплины составляет 3 зачетные единицы (108 часов)

Вид работы	6 семестр	Всего
Общая трудоемкость	**108**	**108**
Аудиторная работа:	**45**	**45**
Лекции (Л)	15	15
Лабораторные занятия (ЛР)	15	15
Практические занятия (ПЗ)	15	15
Самостоятельная работа:	**63**	**63**
Самостоятельное изучение разделов	31	31

Самоподготовка (проработка и повторение лекционного материала и материала учебников и учебных пособий, подготовка к лабораторным и практическим занятиям, коллоквиумам, рубежному контролю и т.д.),	32	32
Контроль (подготовка к сдаче экзамена)		
Вид промежуточной аттестации	зачет	экзамен

№ раздела	Наименование разделов	Количество часов				
		Всего	Контактная работа			Внеауд. работа СР
			Л	ЛР	ПЗ	
1.	Основные принципы и методология разработки прикладного программного обеспечения	14	2	-	-	11
2.	Языки программирования, классификация. Языки программирования низкого и высокого уровней, как инструмент для реализации управляющих функций в мехатронных и робототехнических системах.	27	3	2	2	16
3.	Синтаксис языков программирования.	41	5	8	8	16
4.	Подход объектно-ориентированного программирования при разработке ПО для управления мехатронными и робототехническими системами	35	5	5	5	10
Итого:		117	15	15	15	63

4.3 Лабораторные занятия

№	Темы занятий	Кол. часов
1.	Изучение среды разработки VISUAL STUDIO	1
2.	Линейные алгоритмы	1
3.	Разветвляющиеся алгоритмы	2
4.	Циклические алгоритмы	2
5.	Классы и объекты	2
6.	Строки	2
7.	Одномерные массивы	1
8.	Многомерные массивы	2
9.	Графики функций	2
	ИТОГО	15

4.4 Практические занятия

№	Тема	Кол. часов.
1	Обзор средств разработки и отладки программ Microsoft Visual Studio 2010, описание интерфейса программы.	2
2	Синтаксис языков программирования.	7
3	Подход объектно-ориентированного программирования при разработке ПО для управления мехатронными и робототехническими системами	6
ИТОГО		15

4.5 Самостоятельное изучение разделов дисциплины

№	Вопросы, выносимые на самостоятельное изучение	Кол-во часов
1	2	3
1.	Развитие языков программирования, современные тенденции.	11
2.	Языки программирования, классификация. Языки программирования низкого и высокого уровней, как инструмент для реализации управляющих функций в мехатронных и робототехнических системах.	26
3.	Синтаксис языков программирования C++, C#.	26
4.	Итого	63

5 Оценочные материалы для текущего контроля успеваемости и промежуточной аттестации

5.1 Оценочные материалы для текущего контроля успеваемости

В рамках балльно-рейтинговых мероприятий студент трижды проходит тестирование на компьютере. В зависимости от процента правильных ответов компьютер выставляет от 0 до 6 баллов. Примеры тестовых заданий, приведены ниже.

Тесты:

I:

S: Язык C#, первым символом идентификатора C# может быть …

-: только цифра

-: любой символ

+: только буква

I:

S: Язык C#, длина идентификатора …

-: ограничена тремя символами

+: не ограничена

-: ограничена десятью символами

-: ограничена одним символом
I:
S: Язык C#, преобразование встроенных типов…
-: допускается только неявное преобразование типов
-: допускается только явное преобразование типов
-: объекты одного типа не могут быть преобразованы в объекты другого типа
+: объекты одного типа могут быть преобразованы в объекты другого типа неявно или явно
I:
S: Язык C#, переменные …
+: перед использованием, любая переменная должна быть объявлена и инициализирована
-: перед использованием, любая переменная должна быть только объявлена
-: перед использованием нет необходимости объявлять и инициализировать переменные
I:
S: Язык C#, константа …
-: это переменная целого типа, значение которой можно изменять
+: это переменная, значение которой не может быть изменено
-: это переменная, значение которой может быть изменено
-: это массив переменных целого типа, значение которых может быть изменено
I:
S: Язык C#, перечисления это …
+: особый тип значений, который состоит из набора именованных констант
-: набор данных целого типа
-: массив данных
-: особый вид класса

Задания к лабораторным работам

При выполнении лабораторной работы, студент изучает теоретическую часть, пишет программу. Студент представляет отчет и демонстрирует работоспособность написанной программы. За выполнение и защиту лабораторных работ студент может набрать 18 баллов (по 3 балла в каждую рейтинговую точку).

Лабораторная работа № 1. Изучение среды разработки VISUAL STUDIO

Цель работы: Изучить среду быстрой разработки приложений Visual Studio. Научиться размещать и настраивать внешний вид элементов управления на форме.

Лабораторная работа №2. Линейные алгоритмы

Цели работы: Научиться составлять каркас простейшей программы в среде Visual Studio. Написать и отладить программу линейного алгоритма.

Лабораторная работа №3. Разветвляющиеся алгоритмы

Цель работы: Научиться пользоваться элементами управления для организации переключений (RadioButton). Написать и отладить программу разветвляющегося алгоритма.

Лабораторная работа №4. Циклические алгоритмы

Цель работы: Изучить простейшие средства отладки программ в среде Visual Studio. Составить и отладить программу циклического алгоритма.

Лабораторная работа №5. Классы и объекты

Цель работы: Изучить основные понятия, относящиеся к классам и объектам, освоить динамическое создание объектов в программном коде.

Лабораторная работа №6. Строки

Цель работы: Изучить правила работы с элементом управления ListBox. Написать программу для работы со строками.

Лабораторная работа №7. Одномерные массивы

Цель работы: Изучить способы получения случайных чисел. Написать программу для работы с одномерными массивами.

Лабораторная работа №8. Многомерные массивы

Цель работы: Изучить свойства элемента управления DataGridView. Написать программу с использованием двухмерных массивов.

Лабораторная работа №9. Графики функций

Цель работы: Изучить возможности построения графиков с помощью элемента управления Chart. Написать и отладить программу построения на экране графика заданной функции.

Пример одной из лабораторных работ

ЛАБОРАТОРНАЯ РАБОТА. КЛАССЫ И ОБЪЕКТЫ

Цель лабораторной работы: изучить основные понятия, относящиеся к классам и объектам, освоить динамическое создание объектов в программном коде.

5.1. Классы и объекты

В объектно-ориентированном подходе существуют понятия *класс* и *объект*.

Класс – это программная единица, которая задает общий шаблон для конкретных объектов. Класс содержит все необходимые описания переменных, свойств и методов, которые относятся к объекту. Примером класса в реальной жизни является *понятие «автомобиль»*: как правило, автомобиль содержит некоторое количество колес, дверей, имеет какой-то цвет, но эти конкретные детали в классе не описываются.

Объект – это экземпляр класса. Свойства объекта содержат конкретные данные, характерные для данного экземпляра. В реальной жизни примером объекта будет конкретный экземпляр автомобиля с 4 колесами, 5 дверками и синего цвета.

5.2. Динамическое создание объектов

Чаще всего для размещения на форме кнопки, поля ввода или других управляющих элементов используется дизайнер среды Visual Studio: нужный элемент выделяется в панели элементов и размещается на форме. Однако иногда создавать элементы нужно уже в процессе выполнения программы. Поскольку каждый элемент управления представляет собой отдельный класс, его помещение на форму программным способом включает несколько шагов:

1. Создание экземпляра класса.

2. Привязка его к форме.
3. Настройка местоположения, размеров, текста и т. п.

Например, чтобы создать кнопку, нужно выполнить следующий код (его следует разместить в обработчике сообщения Load или в каком-либо другом методе):

```
Button b = new Button();
```

Здесь объявляется переменная b, относящаяся к классу Button, как и в предыдущих лабораторных работах. Однако дальше идет нечто новое: с помощью оператора new создается экземпляр класса Button, и ссылка на него присваивается переменной b. При этом выполняется целый ряд дополнительных действий: выделяется память под объект, инициализируются все свойства и переменные. Далее нужно добавить объект на форму. Для этого служит свойство Parent, которое определяет родительский элемент, на котором будет размещена кнопка:

```
b.Parent = this;
```

Ключевое слово this относится к тому объекту, в котором размещен выполняемый в данный момент метод. Поскольку все методы в лабораторных работах размещаются в классе формы, то и this относится к этому конкретному экземпляру формы. Вместо формы кнопку можно поместить на другой контейнер. На пример, если на форме есть элемент управления Panel, то можно поместить кнопку на него следующим образом:

```
b.Parent = panel1;
```

Чтобы задать положение и размеры кнопки, нужно использовать свойства Location и Size:

```
b.Location = new Point(10, 20);
b.Size = new Size(200, 100);
```

Обратите внимание, что Location и Size – это тоже объекты. Хотя внутри у Location содержатся координаты x и y, задающие левый верхний угол объекта, не получится поменять одну из координат, нужно менять целиком весь объект Location. То же самое относится и к свойству Size. На самом деле, каждый раз, когда на форму помещается новый элемент управления или, вносятся какие-то изменения в свойства элементов управления, Visual Studio генерирует специальный служебный код, который проделывает приведенные выше операции по созданию и настройке элементов управления. Попробуйте поместить на форму кнопку, изменить у нее какие-нибудь свойства, а затем найдите в обозревателе решений ветку формы Form1, разверните ее и сделайте двойной щелчок по ветке Form1.Designer.cs. Откроется файл с текстом программы на языке C#, которую среда создала автоматически. Менять этот код вручную крайне не рекомендуется! Однако можно его изучить, чтобы понять принципы создания элементов управления в ходе выполнения программы.

5.3. Область видимости

Переменные, объявленные в программе, имеют область видимости. Это значит, что переменная, описанная в одной части программы, не обязательно будет видна в другой. Вот наиболее часто встречающиеся ситуации:

1. Переменные, описанные внутри метода, не будут видны за пределами этого метода. Например:

```
void MethodA()
{
// Описываем переменную delta
int delta = 7;
}
void MethodB()
{
// Ошибка: переменная delta в этом методе неизвестна!
int gamma = delta + 1;
}
```

2. Переменные, описанные внутри блока или составного оператора, видны только внутри этого блока. Например:

```
void Method()
{
if (a == 7)
{
int b = a + 5;
}
// Ошибка: переменная b здесь уже неизвестна!
MessageBox.Show(b.ToString());
}
```

3. Переменные, описанные внутри класса, являются *глобальными* и доступны для всех методов этого класса, например:

```
class Form1 : Form
{
int a = 5;
void Method()
{
// Переменная a здесь действительна
MessageBox.Show(a.ToString());
}
}
```

5.4. Операции is и as

Часто бывает удобно переменные разных классов записать в один список, чтобы было легче его обрабатывать. Чтобы проверить, к какому классу принадлежит какой-либо объект, можно использовать оператор if: он возвращает истину, если объект принадлежит указанному классу. Пример:

```
Button b = new Button();
if (b is Button)
MessageBox.Show("Это кнопка!");
else
MessageBox.Show("Это что-  то другое...");
```

Как правило, в общих списках объекты хранятся в «обезличенном» состоянии, так, чтобы у всех у них был лишь минимальный общий для всех набор методов и свойств. Для того чтобы получить доступ к расширенным свойствам объекта, нужно *привести* его к исходному классу с помощью *операции приведения* as:

```
(someObject as Button).Text = "Это кнопка!";
```

Следует помнить, что операция приведения сработает только в том случае, если объект изначально принадлежит тому классу, к которому его пытаются привести (или совместим с ним), в противном случае оператор as выбросит исключение и остановит выполнение программы. Поэтому более безопасный подход состоит в комбинированном применении операторов as и is: сначала проверяем совместимость объекта и класса, и только потом выполняем операцию приведения:

```
if (someObject is Button)
(someObject as Button).Text = "Это кнопка!";
```

В качестве практического примера использования этих операций рассмотрим пример программы, которая перебирает все элементы управления на форме, и у кнопок (но не у других элементов управления!) заменяет текст на пять звездочек «*****»:

```
private void Form1_Load(object sender, EventArgs e)
{
// Перебираем все элементы управления
```

```
foreach (Control c in this.Controls)
if (c is Button) // Кнопка?
(c as Button).Text = "*****"; // Да!
}
```

5.5. Сведения, передаваемые в событие

Когда происходит какое-либо событие (например, событие Click при нажатии на кнопку), в обработчик этого события передаются дополнительные сведения об этом событии в параметре e.

Например, при щелчке кнопки мыши на объекте возникает событие MouseClick. Для этого события параметр e содержит целый ряд переменных, которые позволяют узнать информацию о нажатии:

- Button – какая кнопка была нажата;
- Clicks – сколько раз была нажата и отпущена кнопка мыши;
- Location – координаты точки, на которую указывал курсор в момент нажатия, в виде объекта класса Point;
- X и Y – те же координаты в виде отдельных переменных.

Индивидуальные задания

Если в индивидуальном задании используется элемент Panel, измените его цвет, чтобы он визуально выделялся на форме. Если используется элемент Label, не забудьте присвоить ему какой-либо текст, иначе он не будет виден на форме.

1. Разработать программу, динамически порождающую на окне кнопки. Левый верхний угол кнопки определяется местоположением курсора при щелчке. Вывести надпись на кнопке с координатами ее левого верхнего угла.

2. Разработать программу, динамически порождающую на окне кнопки и поля ввода. Левый верхний угол элемента управления определяется местоположением курсора при щелчке. Кнопка порождается, если курсор находится в левой половине окна, в ином случае порождается поле ввода.

3. На форме размещен элемент управления Panel. Написать программу, которая при щелчке мыши на элементе управления Panel добавляет в него кнопки Button, а при щелчке на форме в нее добавляются поля ввода TextBox.

4. На форме размещены 3 панели (элемент управления Panel). Написать программу, которая при щелчке мыши на первой панели добавляет во вторую панель кнопки Button, при щелчке на второй панели добавляет в третью панель поля ввода TextBox, а при щелчке на третьей панели добавляет на первую панель метки Label.

5. Написать программу, добавляющую на форму кнопки. Кнопки добавляются в узлы прямоугольной сетки. Расстояния между кнопками и расстояния между крайней кнопкой и границей окна должны быть равны как по горизонтали, так и по вертикали.

6. Разработать программу, при щелчке мыши динамически порождающую на окне кнопки или поля ввода. Каждый четный элемент управления является кнопкой, нечетный – полем ввода. Левый верхний угол кнопки определяется местоположением курсора при щелчке. Для поля ввода положение курсора определяет координаты *правого нижнего* угла.

7. Создать программу с кнопкой, меткой и полем ввода. При щелчке на соответствующий элемент на форме динамически должен создаваться подобный ему элемент. Предусмотреть возможность вывода количества кнопок, меток и полей ввода.

8. Создать программу, добавляющую различные элементы управления на форму и на панель Panel. Тип элементов управления выбирается случайным образом. Предусмотреть возможность вывода информации о количестве элементов по типам и информацию о расположении элементов.

9. Разработать программу, добавляющую на форму последовательность элементов управления случайной длины. Тип элементов управления задается случайным образом. Предусмотреть возможность вывода информации о количестве элементов по типам.

10. Написать программу, динамически порождающую на окне кнопки или метки. Левый верхний угол элемента управления определяется местоположением курсора при щелчке. При нажатии правой кнопки мыши на форме с нее удаляются все кнопки.

11. Написать программу, динамически порождающую на окне поочередно кнопки или поля ввода. Левый верхний угол элемента управления определяется местоположением курсора при щелчке. При нажатии правой кнопки мыши на форме с нее удаляются все порожденные элементы.

12. Разработать программу с двумя кнопками на форме. При нажатии на первую на форму добавляется одна панель Panel. При нажатии на вторую кнопку в каждую панель добавляется поле ввода.

13. Разработать программу с двумя кнопками на форме. При нажатии на первую на форму добавляется одна кнопка или поле ввода. При нажатии на вторую кнопку каждое поле увеличивается по вертикали в два раза.

14. Написать программу с кнопкой и тремя полями ввода. При нажатии на кнопку программа анализирует содержимое первого поля и динамически порождает элемент управления. Если в первом поле ввода содержится буква «К», то на форму добавляется кнопка, если «П» – поле ввода, если «М» – метка. Во втором и третьем поле ввода содержатся координаты левого верхнего угла будущего элемента управления.

15. Разработать программу, добавляющую на форму метки с текстом. Местоположение и размеры меток определяются в программе динамически через поля ввода. В заголовок окна, анализируя размер всех меток, вывести количество маленьких и больших меток. Маленькой меткой считается метка размером менее 50 пикселей по горизонтали и вертикали.

16. Создать программу с двумя кнопками на форме, динамически порождающую на окне метки или поля ввода. При нажатии на первую кнопку каждая метка увеличивается по горизонтали в два раза. При нажатии на вторую кнопку каждое поле уменьшается по вертикали в два раза.

17. Разработать программу, динамически порождающую на окне кнопки и поля ввода. Координаты элемента управления определяются случайным образом. Элементы управления не должны накладываться друг на друга. Если нет возможности добавить элемент управления (нет места для размещения элемента), то предусмотреть вывод информации об этом.

18. Разработать программу, динамически порождающую на окне кнопки и поля ввода. Координаты элемента управления определяются случайным образом. При наведении курсора на элемент управления он должен быть удален с формы.

19. Разработать программу, динамически порождающую при щелчке на окне различные элементы (поля ввода, кнопки, метки). Тип элементов определяется с помощью радиокнопок. Все элементы располагаются горизонтально в ряд. При достижении правой границы окна начинается новый ряд элементов.

20. Разработать программу, динамически порождающую или поле ввода (при нажатии на окне левой кнопкой мыши), или кнопку (при нажатии на окне правой кнопкой мыши). Все элементы располагаются наискосок, начиная с левого верхнего угла окна. Реализовать обработчик события изменения размера окна, в котором удалить все порожденные элементы.

Экзаменационные вопросы

1.	Языки программирования низкого и высокого уровня, их отличие
2.	Язык C++, основные типы переменных, преобразование типов, явное и неявное.

3.	Язык С++, операторы управления потоком выполнения программы – if – else, switch.
4.	Язык С++, операторы цикла.
5.	Язык С++, указатели и ссылки, назначение и применение.
6.	Язык С++, массивы и указатели.
7.	Язык С++, операторы new и delete, их назначение.
8.	Язык С++, многомерные массивы, динамические массивы.
9.	Язык С++, определение, описание и вызов функций, рекурсивные функции.
10.	Язык С++, перегрузка функций.
11.	Язык С++, структуры и объединения.
12.	Язык С++, понятие класса, методы и члены класса.
13.	Язык С++, класс, конструктор и деструктор, доступность компонентов класса.
14.	Понятия – инкапсуляция, наследование, полиморфизм.
15.	Объектно-ориентированное программирование, назначение и основные понятия.
16.	Понятие класса в объектно-ориентированном программировании.
17.	Перегрузка конструктора класса.
18.	Язык С++,С#. Обработка исключений.
19.	Передача в функцию параметров по значению и по ссылке.
20.	Многомерные массивы.
21.	Язык С#, класс, конструктор и деструктор, доступность компонентов класса.
22.	Язык С++,С#. Классы, наследование, множественное наследование.
23.	Язык С#, перегрузка функций.
24.	Язык С#, перегрузка конструктора класса.
25.	Язык С#, понятие класса, методы и члены класса.
26.	Язык С#, основные типы переменных, преобразование типов, явное и неявное.
27.	Язык С#, операторы цикла.
28.	Язык С#, структуры и объединения.
29.	Язык С#, класс, конструктор и деструктор, доступность компонентов класса.
30.	Язык С#, массивы.

6 Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности

6.1 Результаты освоения учебной дисциплины, подлежащие проверке

Контролируемые компетенции (часть компетенций)	**Результаты обучения (объекты оценивания)**	**Основные показатели оценки результатов**	**Оценочные средства**
готовностью собирать, обрабатывать, анализировать и систематизировать научно-техническую информацию по тематике исследования, использовать достижения отечественной и зарубежной науки, техники и техноло-	**З1** Знать основные архитектуры устройств управления роботов и РТС	Знание основных архитектур устройств управления роботов и РТС	практическое занятие, лабораторная работа, тестирование, экзамен
	З2 Знать основы синтаксических конструкций современных языков программирования	Знание основ синтаксических конструкций современных языков программирования	практическое занятие, лабораторная работа, тестирование, экзамен
	З3 Знать шаблоны проектирования высокоуровневого программного обеспечения,	Знание шаблонов проектирования высокоуровневого программного обеспечения, применяющихся для управ-	практическое занятие, лабораторная работа, тестирование, экзамен

гии в своей профессиональной деятельности (ОПК-4).	применяющихся для управления и моделирования РТС	ления и моделирования РТС	
	У1 Уметь анализировать архитектуру устройств управления роботов и РТС	Умение анализировать архитектуру устройств управления роботов и РТС	практическое занятие, лабораторная работа, экзамен
	У2 Уметь применять основные методы проектирования сложных систем программного обеспечения с использованием объектно-ориентированного подхода	Умение применять основные методы проектирования сложных систем программного обеспечения с использованием объектно-ориентированного подхода	практическое занятие, лабораторная работа, экзамен
	В1 Владеть навыками работы в комплексных средах создания программного обеспечения	Владение навыками работы в комплексных средах создания программного обеспечения	практическое занятие, лабораторная работа
	В2 Владеть навыками написания алгоритмов и на языках программирования высокого уровня	Владение навыками написания алгоритмов и на языках программирования высокого уровня	практическое занятие, лабораторная работа
	В3 Владеть навыками проектирования сложных систем с использованием объектно-ориентированного подхода	Владение навыками проектирования сложных систем с использованием объектно-ориентированного подхода	практическое занятие, лабораторная работа

6.2 Шкала оценивания планируемых результатов обучения

6.2.1 Текущий и рубежный контроль

В рамках текущего и рубежного контроля по дисциплине студент может набрать до 70 баллов

Семестр	Шкала оценивания			
	0-35 баллов	36-50 баллов	51-60 баллов	61-70 баллов
6	Частичное посещение аудиторных занятий. Неудовлетворительное выполнение лабораторных и практических работ. Плохая подготовка к балльно-рейтинговым мероприятиям. Студент не допускается к промежуточной аттестации	Полное или частичное посещение аудиторных занятий. Частичное выполнение и защита лабораторных и практических работ. Выполнение контрольных работ, тестовых заданий на оценки «удовлетворительно».	Полное или частичное посещение аудиторных занятий. Полное выполнение и защита лабораторных и практических работ. Выполнение контрольных работ, тестовых заданий на оценки «хорошо».	Полное посещение аудиторных занятий. Полное выполнение и защита лабораторных и практических занятий. Выполнение контрольных работ, тестовых заданий на оценки «отлично».

6.2.2 Промежуточная аттестация

Оценка результатов освоения учебной дисциплины в 6 семестре проводится по шкале, используемой на экзамене:

Семестр	Шкала оценивания			
	Неудовлетворительно (36-60 баллов)	Удовлетворительно (61-80 баллов)	Хорошо (81-90 баллов)	Отлично (91-100 баллов)
6	Студент имеет 36-60 баллов по итогам текущего и рубежного контроля, на экзамене не дал полного ответа ни на один вопрос. Студент имеет 36-45 баллов по итогам текущего и рубежного контроля, на экзамене дал полный ответ только на один вопрос	Студент имеет 36-50 баллов по итогам текущего и рубежного контроля, на экзамене дал полный ответ на один вопрос и частично (полностью) ответил на второй. Студент имеет 46-60 баллов по итогам текущего и рубежного контроля, на экзамене дал полный ответ на один вопрос или частично ответил на оба вопроса. Студент имеет по итогам текущего и рубежного контроля 61-70 баллов на экзамене не дал полного ответа ни на один вопрос.	Студент имеет 51-60 баллов по итогам текущего и рубежного контроля, на экзамене дал полный ответ на один вопрос и частично (полностью) ответил на второй. Студент имеет 61 – 65 баллов по итогам текущего и рубежного контроля, на экзамене дал полный ответ на один вопрос и частично ответил на второй. Студент имеет 66-70 баллов по итогам текущего и рубежного контроля, на экзамене) дал полный ответ только на один вопрос.	Студент имеет 61-70 баллов по итогам текущего и рубежного контроля, на экзамене дал полный ответ на один вопрос и частично (полностью) ответил на второй.

7 Учебно-методическое обеспечение дисциплины (модуля)

7.1 Основная литература

1. Ревич Ю.В. Практическое программирование микроконтроллеров Atmel AVR на языке ассемблера. –СПб.: БВХ-Петербург, 2008. -384 с.
2. Хартов В.Я. Микроконтроллеры AVR. Практикум для начинающих. –М.: Изд-во МГТУ им. Н.Э. Баумана, 2007, -240 с.
3. Шпак Ю.А. Программирование на языке С для AVR и PIC микроконтроллеров. –К.: «МК-Пресс», 2006. -400 с.
4. Шилдт Г. Полный справочник по C#. –М.: «Вильямс», 2004. -752 с.
5. Подбельский В.В. Язык С++. –М.: Финансы и статистика, 2007. -560 с.

6. Брагин В.Б., Войлов Ю.Г., Жаботинский Ю.Д., и др. Системы очувствления и адаптивные промышленные роботы. –М.: Машиностроение, 1985. -256 с.
7. Куафе Ф. Взаимодействие робота с внешней средой: Пер. с франц. –М.: Мир, 1985. -285 с.
8. Биллиг В.А. Основы объектного программирования на C# (C# 3.0, Visual Studio 2008) [Электронный ресурс]: учебное пособие/ Биллиг В.А.— Электрон. текстовые данные.— Москва, Саратов: Интернет-Университет Информационных Технологий (ИНТУИТ), Вузовское образование, 2017.— 583 c.— Режим доступа: http://www.iprbookshop.ru/72339.html.— ЭБС «IPRbooks»
9. Кариев Ч.А. Разработка Windows-приложений на основе Visual C# [Электронный ресурс]: учебное пособие/ Кариев Ч.А.— Электрон. текстовые данные.— Москва, Саратов: Интернет-Университет Информационных Технологий (ИНТУИТ), Вузовское образование, 2017.— 768 c.— Режим доступа: http://www.iprbookshop.ru/72340.html.— ЭБС «IPRbooks»
10. Камлюк В.С. Мехатронные модули и системы в технологическом оборудовании для микроэлектроники [Электронный ресурс]: учебное пособие/ Камлюк В.С., Камлюк Д.В.— Электрон. текстовые данные.— Минск: Республиканский институт профессионального образования (РИПО), 2016.— 384 c.— Режим доступа: http://www.iprbookshop.ru/67660.html.— ЭБС «IPRbooks»
11. Разработка Windows-приложений в среде программирования Visual Studio.Net [Электронный ресурс]: учебно-методическое пособие по дисциплине Информатика и программирование/ — Электрон. текстовые данные.— М.: Московский технический университет связи и информатики, 2016.— 20 c.— Режим доступа: http://www.iprbookshop.ru/61536.html.— ЭБС «IPRbooks»

7.2 Дополнительная литература

1. Яценков В.С. Основы спутниковой навигации. Системы GPS NAVSTAR и ГЛОНАС. – М.: Телеком, 2005. – 272 с.
2. Жданов А.А. Автономный искусственный интеллект/ -М.: Бином. 2008. -359с.
3. Интеллектуальные роботы: учебное пособие для вузов/ под ред. Е.И. Юревича. –М.: Машиностроение, 2007 -360с.
4. Робототехнические системы и комплексы: Учеб. Пособие для вузов/ Мачульский И.И, Запятой В.П., Майоров Ю.П. и др. М.: Транспорт 1999. 446 с.

7.3 Интернет-ресурсы

1. Wikipedia – свободная энциклопедия. - http://ru.wikipedia.org/.
2. http://www.atmel.com
3. https://msdn.microsoft.com
4. http://www.iprbookshop.ru/

7.4 Методические указания к лабораторным занятиям

Комплект учебного оборудования «Микропроцессорные системы управления электроприводов» ПО1033 Методические указания к выполнению лабораторных работ. ООО ТД «ПрофОбразование» 2015.

7.5 Программное обеспечение современных информационно-коммуникационных технологий

1. Microsoft Windows.
2. Пакет Microsoft Office.
3. Программные продукты: Atmel Studio.

4. Программный продукт: Microsoft Visual Studio 2010

8 Материально-техническое обеспечение дисциплины

Требования к условиям реализации дисциплины:

№ п/п	Вид аудиторного фонда	Требования
1.	Лекционная аудитория	Оснащение специализированной учебной мебелью. Оснащение техническими средствами обучения: настенный экран с дистанционным управлением, мультимедийное оборудование.
2.	Кабинет для практических занятий	Оснащение специализированной учебной мебелью. Оснащение техническими средствами обучения: подвижная маркерная доска, считывающее устройство для передачи информации в компьютер; настенный экран с дистанционным управлением, мультимедийное оборудование.
3.	Компьютерные классы	Оснащение специализированной учебной мебелью. Оснащение техническими средствами обучения: ПК с возможностью подключения к локальным сетям и Интернету. Наличие ВТ из расчета один ПК на два студента.

Перечень материально-технического обеспечения дисциплины:

№ п/п	Вид и наименование оборудования	Вид занятий	Краткая характеристика
1.	IBM PC - совместимые персональные компьютеры.	Практические занятия.	Процессор серии не ниже Pentium IV. Оперативная память не менее 512 Мбайт. ПК должны быть объединены локальной сетью с выходом в Интернет.
2.	Мультимедийные средства.	Лекционные и практические занятия.	Демонстрация с ПК электронных презентаций, документов Word, электронных таблиц, графических изображений.

№ работ	Материальное обеспечение лабораторных занятий
1	2
	1. Комплект учебного оборудования «Микропроцессорные системы управления электроприводов». 2. Микроконтроллеры фирмы Atmel. 3. Программаторы для микроконтроллеров. 4. Макетная плата для микроконтроллера, датчики различных параметров, средства индикации, шаговые электродвигатели и электродвигатели постоянного тока.

9 Особенности реализации дисциплины для инвалидов и лиц с ограниченными возможностями здоровья

Для студентов с ограниченными возможностями здоровья созданы специальные условия для получения образования. В целях доступности получения высшего образования по образовательным программам инвалидами и лицами с ограниченными возможностями здоровья университетом обеспечивается:

1. Альтернативная версия официального сайта в сети «Интернет» для слабовидящих;

2. Для инвалидов с нарушениями зрения (слабовидящие, слепые)

- присутствие ассистента, оказывающего обучающемуся необходимую помощь, дублирование вслух справочной информации о расписании учебных занятий; наличие средств для усиления остаточного зрения, брайлевской компьютерной техники, видеоувеличителей, программ невизуального доступа к информации, программ-синтезаторов речи и других технических средств приема-передачи учебной информации в доступных формах для студентов с нарушениями зрения;

- задания для выполнения на экзамене зачитываются ассистентом;

- письменные задания выполняются на бумаге, надиктовываются ассистенту обучающимся;

3.Для инвалидов и лиц с ограниченными возможностями здоровья по слуху (слабослышащие, глухие):

- на зачете/экзамене присутствует ассистент, оказывающий студенту необходимую техническую помощь с учетом индивидуальных особенностей (он помогает занять рабочее место, передвигаться, прочитать и оформить задание, в том числе записывая под диктовку);

- зачет/экзамен проводится в письменной форме;

4. Для инвалидов и лиц с ограниченными возможностями здоровья, имеющих нарушения опорно-двигательного аппарата, созданы материально-технические условия обеспечивающие возможность беспрепятственного доступа обучающихся в учебные помещения, объекту питания, туалетные и другие помещения университета, а также пребывания в указанных помещениях (наличие расширенных дверных проемов, поручней и других приспособлений).

- письменные задания выполняются на компьютере со специализированным программным обеспечением или надиктовываются ассистенту;

- по желанию студента экзамен проводится в устной форме.

Обучающиеся из числа лиц с ограниченными возможностями здоровья обеспечены электронными образовательными ресурсами в формах, адаптированных к ограничениям их здоровья.