Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Курганский государственный университет»
(КГУ)

Кафедра «Программное обеспечение автоматизированных систем»

УТВЕРЖДАЮ:
Первый проректор
_____ / Т.Р. Змызгова/
«31» августа 2022 г.

Рабочая программа учебной дисциплины

ОСНОВЫ ПРОГРАММИРОВАНИЯ

образовательной программы высшего образования –
программы бакалавриата

09.03.00 Информатика и вычислительная техника

09.03.03 Прикладная информатика
Направленность:
Интеллектуальные информационные системы и технологии

09.03.04 Программная инженерия
Направленность:
Программное обеспечение автоматизированных систем

Форма обучения: очная, заочная

Курган 2022

Рабочая программа дисциплины «Основы программирования» составлена в соответствии с учебными планами по программе бакалавриата «Прикладная информатика» (Интеллектуальные информационные системы и технологии), «Программная инженерия» (Программное обеспечение автоматизированных систем), утвержденными для очной формы обучения «30» августа 2022 года, для заочной формы обучения «30» августа 2022 года.

Рабочая программа дисциплины одобрена на заседании кафедры «Программное обеспечение автоматизированных систем» «30» августа 2022 года, протокол № 1.

Рабочую программу составил:

к.п.н., доцент _____ А.А. Медведев

Согласовано:

Заведующий
кафедрой ПОАС _____ В.К. Волк

Начальник управления
образовательной
деятельности _____ И.В. Григоренко

Специалист
по учебно-методической работе
учебно-методического отдела _____ Г.В. Казанкова

1. ОБЪЕМ ДИСЦИПЛИНЫ

Всего: 9 зачетных единиц трудоемкости (324 академических часа)

Очная форма обучения

Вид учебной работы	На всю дисциплину	Семестр	
		1	2
Аудиторные занятия (контактная работа с преподавателем), всего часов в том числе:	**136**	**64**	**72**
Лекции	56	32	24
Лабораторные работы	64	32	32
Практические работы	16	-	16
Аудиторные занятия в интерактивной форме, часов	50	12	38
Самостоятельная работа, всего часов в том числе:	**188**	**80**	**108**
Подготовка к экзамену	27	-	27
Подготовка к зачету	18	18	-
Подготовка контрольной работы	18	18	-
Подготовка курсовой работы	36	-	36
Другие виды самостоятельной работы (самостоятельное изучение тем (разделов) дисциплины)	89	44	45
Вид промежуточной аттестации	**Зачет Экзамен**	**Зачет**	**Экзамен**
Общая трудоемкость дисциплины и трудоемкость по семестрам, часов	**324**	**144**	**180**

Заочная форма обучения

Вид учебной работы	На всю дисциплину	Семестры	
		2	3
Аудиторные занятия (контактная работа с преподавателем), всего часов в том числе:	**26**	**12**	**14**
Лекции	8	4	4
Лабораторные работы	12	6	6
Практические занятия	6	2	4
Аудиторные занятия в интерактивной форме, часов	-	-	-
Самостоятельная работа, всего часов в том числе:	**298**	**132**	**166**
Подготовка к зачету, экзамену	45	18	27
Другие виды самостоятельной работы (самостоятельное изучение тем (разделов) дисциплины)	199	96	103
Контрольная работа	18	18	-
Курсовая работа	36		36
Вид промежуточной аттестации	**Зачет Экзамен**	**Зачет**	**Экзамен**
Общая трудоемкость дисциплины и трудоемкость по семестрам, часов	**324**	**144**	**180**

2. МЕСТО ДИСЦИПЛИНЫ В СТРУКТУРЕ ОБРАЗОВАТЕЛЬНОЙ ПРОГРАММЫ

Дисциплина «Основы программирования» относится к дисциплинам обязательной части блока 1, дисциплина модуля «Информатика и программирование».

Изучение дисциплины базируется на знаниях, умениях и навыках, приобретенных студентами в средней школе.

Результаты обучения по дисциплине необходимы для изучения дисциплин: «Информатика», «Объектно-ориентированное программирование», «Технологии разработки Web-приложений».

3. ПЛАНИРУЕМЫЕ РЕЗУЛЬТАТЫ ОБУЧЕНИЯ

Цель освоения дисциплины: изучение современных методов создания качественного программного обеспечения, удовлетворяющего заданным требованиям, знакомство с современными методами разработки простейших программных приложений. Формирование общепрофессиональных и специальных компетентностей посредством знакомства студентов с базовыми понятиями программирования; формирование умения анализировать поставленную задачу и на основе анализа выбирать соответствующие средства языка программирования для ее реализации.

Задачи дисциплины: используя ресурсы образовательной программы, университетского образовательного пространства, профессионального сообщества способствовать формированию у студентов навыков решения стандартных задач профессиональной деятельности на основе информационной культуры с применением информационно-коммуникационных технологий и с учетом основных требований информационной безопасности; заложить основы программирования приложений и создания программных прототипов решения прикладных задач.

Компетенции, формируемые в результате освоения дисциплины.

Для 09.03.03:

- Способен понимать принципы работы современных информационных технологий и программных средств, в том числе отечественного производства, и использовать их при решения задач профессиональной деятельности; (ОПК-2);
- Способен разрабатывать алгоритмы и программы, пригодные для практического применения; (ОПК-7).

В результате изучения дисциплины обучающийся должен

знать:

- современные методы и программные средства в профессиональной деятельности (ОПК-2);

уметь:

- использовать современные методы и программные средства в профессиональной деятельности (ОПК-2);
- уметь разрабатывать, внедрять и адаптировать прикладное программное обеспечение (ОПК-7);

владеть:

- основами языков программирования и библиотек для разработки приложений и создания программных прототипов решения прикладных задач (ОПК-7);
- необходимым математическим аппаратом, применяемым в профессиональной деятельности (ОПК-2).

Для 09.03.04, 09.03.00:

- Способен понимать принципы работы современных информационных технологий и программных средств, в том числе отечественного производства, и использовать их при решения задач профессиональной деятельности (ОПК-2);
- Способен разрабатывать алгоритмы и программы, пригодные для практического использования, применять основы информатики и программирования к проектированию, конструированию и тестированию программных продуктов (ОПК-6).

В результате изучения дисциплины обучающийся должен

знать:

- принципы работы современных информационных технологий и программных средств (ОПК-2);

уметь:

- использовать современные методы и программные средства в профессиональной деятельности (ОПК-2);
- уметь использовать современные технологии и программное обеспечение для решения задач профессиональной деятельности (ОПК-6);

владеть:

- основами языков программирования и библиотек для разработки приложений и создания программных прототипов решения прикладных задач (ОПК-6);
- необходимым математическим аппаратом, применяемым в профессиональной деятельности (ОПК-2).

4. СОДЕРЖАНИЕ ДИСЦИПЛИНЫ

4.1. Учебно-тематический план

Очная форма обучения

Рубеж	Номер раздела, темы	Наименование раздела, темы	Количество часов контактной работы с преподавателем		
			Лекции	Практич. занятия	Лабораторные работы
1 семестр					
Рубеж 1	1.	Переменные. Типы данных. Операторы.	4		-
	2.	Условные и циклические конструкции	2		4
	3.	Строки.	4		4
	4.	Структуры данных (списки, кортежи, множества, диапазоны)	6		8
	5.	Функции.	4		2
		Рубежный контроль № 1	-		2
Рубеж 2	6.	Регулярные выражения	4		2
	7.	Структуры данных. Словари	2		2
	8.	Работа с файлами и каталогами	4		4
	9.	Работа с изображениями	2		2
		Рубежный контроль № 2			2
		Всего	**32**		**32**
2 семестр					
Рубеж 3	10.	Общие сведения о программах, лексемах и алфавите.	2		
	11.	Основные управляющие конструкции C++.	4		6
	12.	Указатели и массивы в C++.	2	4	4
	13.	Функции в C++.	2	4	
	14.	Строки и векторы в C++.	6		6
		Рубежный контроль №3		2	
Рубеж 4	15.	Структуры и объединения в C++.	2		8
	16.	Работа с файлами в C++.	2		8
	17.	Обработка исключительных ситуаций в C++.	2	4	
	18.	Основные компоненты, используемые для создания приложений Windows Forms	2		-
		Рубежный контроль № 4		2	
		Всего	**24**	**16**	**32**
		Итого	**56**	**16**	**64**

Заочная форма обучения

Номер раздела, темы	Наименование раздела, темы	Количество часов контактной работы с преподавателем		
		Лекции	Практич. занятия	Лабораторные работы
2 семестр				
1.	Переменные. Типы данных. Операторы.	1	-	-
2.	Условные и циклические конструкции.	1	-	2
3.	Строки.	2	-	4
4.	Структуры данных (списки, кортежи, множества, диапазоны)		2	
Всего		**4**	**2**	**6**
3 семестр				
10.	Общие сведения о программах, лексемах и алфавите	1	-	-
11.	Основные управляющие конструкции C++.	1	-	2
12.	Указатели и массивы в C++.	2	-	4
14.	Строки и векторы в C++.		4	
Всего		**4**	**4**	**6**
Итого		**8**	**6**	**12**

4.2. Содержание лекционных занятий

1 семестр

Тема 1 Переменные. Типы данных. Операторы

Присваивание значений переменным. Проверка типа данных. Преобразование типов. Основные операторы: двоичные, работы с последовательностями, присваивания.

Тема 2 Условные и циклические конструкции

Операторы сравнения. Условная конструкция. Циклы. Функции range() и enumerate(). Операторы continue и break.

Тема 3 Строки

Основные операции со строками. Функции и методы для работы со строками: поиск и замена в строке, изменение регистра символов, работа с символами, проверка типа содержимого строки.

Тема 4 Структуры данных (списки, кортежи, множества, диапазоны)

Создание списка. Основные операции над списками. Многомерные списки. Перебор элементов списка. Генераторы списков. Методы работы со списка-

ми (добавление и удаление элементов списка, поиск элемента, получение сведений о значениях, входящих в список, сортировки списка).

Кортежи, их отличия от списков. Множества, особенности определения и использования. Основные операции над множествами. Диапазоны, способы задания и область применения.

Словари. Создание словаря. Операции над словарями. Перебор элементов словаря. Основные методы работы со словарями.

Тема 5 Функции

Описание функции и обращение к ней. Особенности использования функций. Анонимные функции. Декораторы функций. Рекурсия. Глобальные и локальные переменные. Вложенные функции.

Тема 6 Регулярные выражения

Синтаксис регулярных выражений. Поиск первого совпадения с шаблоном. Поиск всех совпадений с шаблоном. Замена в строке.

Тема 7 Структуры данных. Словари

Словари. Создание словаря. Операции над словарями. Перебор элементов словаря. Основные методы работы со словарями.

Тема 8 Работа с файлами и каталогами

Открытие файла. Основные функции и методы работы с файлами. Права доступа к файлам и каталогам. Функции работы с каталогами. Сохранение объектов в файл.

Тема 9 Работа с изображениями

Библиотека Pillow, ее базовые возможности. Загрузка изображений. Создание нового изображения. Получение информации об изображении. Рисование линий и фигур.

2 семестр

Тема 10 Общие сведения о программах, лексемах и алфавите

Лексические основы языка C++. Алфавит. Лексемы. Скалярные типы и выражения. Примеры. Использование простейших типов. Примеры. Функции форматного ввода и вывода.

Операции. Знаки операций. Арифметические операции. Примеры. Инкремент и декремент. Примеры. Операции присваивания и отношения. Примеры. Логические операции. Примеры. Условная операция. Примеры. Операции преобразования типов. Примеры.

Тема 11 Основные управляющие конструкции C++

Условные конструкции. Оператор if…else. Примеры. Оператор switch. Примеры.

Циклические конструкции. Цикл while. Примеры. Цикл do…while. Примеры. Цикл for. Примеры. Оператор безусловного перехода. Примеры. Оператор принудительного выхода из цикла или переключателя. Примеры. Оператор завершения выполнения текущего шага тела цикла. Примеры. Оператор возвращающий значение из функции. Примеры.

Тема 12 Указатели и массивы в C++

Указатели и адреса. Понятие указателя и адреса. Операция косвенной адресации и нахождения адреса. Адресная арифметика. Операция sizeof.

Массивы. Общие сведения о массивах. Одномерные массивы. Имя массива. Двумерные массивы. Многомерные массивы. Работа с массивами с помощью указателей. Примеры. Массивы указателей. Примеры. Массивы динамической памяти. Примеры.

Тема 13 Функции в C++

Функции. Общие сведения о функциях. Определение функции. Аргументы функции. Примеры. Передача нескольких значений в функцию. Возврат нескольких значений из функции. Примеры.

Ссылки. Общие сведения о ссылках. Ссылки в качестве параметров функций. Ссылки в качестве результатов функций. Классы памяти. Рекурсия. Функции с переменным числом параметров. Подставляемые (inline) функции. Способы передачи массивов в функции и их возврата. Примеры.

Тема 14 Строки и векторы в C++

Строки. Строковая константа. Инициализация строк. Строки и указатели. Функции для работы со строками. Использование строковых функций. Использование строк в командной строке. Примеры.

Класс string: назначение и способы задания объектов. Основные операции над объектами класса string. Функции для работы с отдельными символами строки. Основные методы класса string.

Класс vector: назначение и способы задания объектов. Понятие итератора. Арифметические операции над итераторами. Основные методы работы с векторами.

Тема 15 Работа с файлами в C++

Понятие файла. Ввод/вывод файлов в языке С. Основные функции для работы с файлами. Стандартный ввод/вывод. Функции стандартного ввода/вывода в языке С.

Ввод/вывод в языке C++. Пространства имен. Классы fstream, ofstream, ifstream; работа с файлами с использованием этих классов. Бинарные файлы. Стандартный ввод/вывод в C++.

Тема 16 Структуры и объединения в C++

Структуры. Описание структур. Операции над структурами. Использование полей битов в качестве полей структур.

Объединения. Понятие объединения. Совместное использование объединений и структур. Переменные структуры.

Тема 17 Обработка исключительных ситуаций в C++

Общие принципы механизма обработки исключений. Синтаксис и семантика генерации и обработки исключений.

Операторы try, catch и throw. Генерация исключений. Основные классы типов исключений. Функции, выдающие исключения.

Тема 18 Основные компоненты, используемые для создания приложений Windows Forms

Понятие CLR-приложения. Описание массивов и структур в CLR-приложениях. Указатели в CLR-приложениях.

Пространство имен System. Основные типы переменных, используемые при создании приложений Windows Forms.

Структура приложения Windows Forms, назначение некоторых файлов.

Понятие компонента, свойства, метода, события. Основные компоненты, используемые при создании приложений Windows Forms (Form, Button, TextBox, Label, ListBox, ComboBox, MenuStrip, Panel, GroupBox, CheckBox, RadioButtom), их основные свойства, методы и события, примеры использования.

4.3. Лабораторные занятия

Номер раздела, темы	Наименование раздела, темы	Наименование лабораторной работы	Норматив времени, час.	
			Очная форма обучения	Заочная форма обучения
1, 2	Переменные. Типы данных. Операторы. Условные и циклические конструкции	Лабораторная работа № 1. Действия с числами. Условные и циклические конструкции	4	2
3	Строки	Лабораторная работа № 2. Преобразование символьных величин.	4	4
4, 7	Структуры данных (списки, кортежи, словари, множества, диапазоны)	Лабораторная работа № 3. Структуры данных. Списки.	6	
		Лабораторная работа № 4. Структуры данных. Множества	2	
		Лабораторная работа № 7. Структуры данных. Словари	2	
	Рубежный контроль 1.		2	
5	Функции	Лабораторная работа № 5. Использование функций.	2	
6	Регулярные выражения	Лабораторная работа № 6. Регулярные выражения	2	
8	Работа с файлами и каталогами	Лабораторная работа № 8. Файлы	4	
9	Работа с изображениями	Лабораторная работа № 9. Графика. Библиотека Pillow	2	
	Рубежный контроль 2.		2	

		Всего	**32**	**6**
10, 11	Общие сведения о программах, лексемах и алфавите. Основные управляющие конструкции С++	Лабораторная работа №10. Действия с числами	6	2
12,13	Указатели и массивы в С++. Функции в С++.	Лабораторная работа №11. Работа с массивами	4	4
14	Строки и векторы в С++.	Лабораторная работа №12. Обработка строковой информации	6	
15	Структуры и объединения в С++.	Лабораторная работа №13. Графика	2	
		Лабораторная работа №14. Структуры	4	
		Лабораторная работа №15. Объединения	2	
16	Работа с файлами в С++	Лабораторная работа №16. Работа с файлами	8	
		Всего	**32**	**6**
		Итого	**64**	**12**

4.4. Практические занятия

Номер раздела, темы	**Наименование раздела, темы**	**Наименование практической работы**	**Норматив времени, час.**	
			Очная форма обучения	**Заочная форма обучения**
1 семестр				
4, 7	Структуры данных (списки, кортежи, словари, множества, диапазоны)	Практическое занятие № 1. Структуры данных.	-	2
		Всего	-	**2**
14	Строки и векторы в С++.	Практическое занятие № 2. Обработка строк в С++		4
12	Указатели и массивы в С++.	Практическое занятие № 3. Особенности работы с указателями в С++.	4	
13	Функции в С++.	Практическое занятие № 4. Особенности использования функций	4	
	Рубежный контроль 3.		2	

17	Обработка исключительных ситуаций в C++.	Практическое занятие № 5. Использование средств обработки исключительных ситуаций	4	
	Рубежный контроль 4.		2	
		Всего	**16**	**4**
		Итого	**16**	**6**

4.5. Контрольная работа

Контрольная работа во 1 семестре (для очной формы обучения) и во 2 семестре (для заочной формы обучения) используется для углубления знаний по особенностям использования языка программирования Python. Требования к оформлению указаны в разделе 4.6.1.

Темы контрольных работ

1. Модуль NumPy, его назначение, основные функции и методы. Примеры использования.
2. Работа с форматом CSV средствами Python. Примеры использования.
3. Работа с форматом XML средствами Python. Примеры использования.
4. Работа с форматом JSON средствами Python. Примеры использования.
5. Web-библиотека requests, ее назначение, основные функции и методы. Примеры использования.
6. Использование методов линейной алгебры в Python. Пакет NumPy. Примеры использования.
7. Работа с электронными таблицами в MSExcel в Python. Примеры использования.
8. Работа с документами MSWord в Python. Примеры использования.
9. Работа с документами PDF в Python. Примеры использования.
10. Работа с изображениями средствами Python. Примеры.
11. Модуль pyautogui, его использование для управления клавиатурой и мышью. Основные функции модуля. Примеры использования.
12. Модуль sys, его применение для получения информации о среде выполнения программы и интерпретаторе Python. Основные функции модуля. Примеры использования.
13. Модуль os, его применение для получения информации оо операционной системе. Основные элементы модуля. Примеры использования.
14. Организация работы с XML-документами средствами Python. Примеры.
15. Модуль collections – набор специальных типов данных, дополняющих встроенные типы. Основные элементы этого модуля, примеры ипользования.
16. Основные сетевые протоколы в Python. Примеры использования.
17. Модуль array, предназначенный для организации массивов в Python. Его содержание и примеры использования.

18. Модуль calendar, предназначенный для работы с календарями в Python. Его содержание и примеры использования.
19. Модуль decimal, предназначенный для работы с числами в Python. Его содержание и примеры использования.
20. Модуль html, предназначенный для разработки сайтов в Python. Его содержание и примеры использования.
21. Работа с очередями средствами Python. Модуль Queue, его содержание и примеры использования.
22. Перечисления в Python. Модуль enum. Основные элементы модуля, примеры использования.

4.6 Курсовая работа

Курсовая работа во 2 семестре (для очной формы обучения) и в 3 семетре (для заочной формы обучения) используется для углубления знаний по особенностям использования языка программирования.

4.6.1 Требования к оформлению курсовой работы

Работа должна быть выполнена на одной стороне листа белой бумаги формата А4 (210х297 мм). Интервал межстрочный - полуторный. Цвет шрифта - черный. Гарнитура шрифта основного текста — «Times New Roman» или аналогичная. Кегль (размер) от 12 до 14 пунктов. Размеры полей страницы (не менее): правое — 30 мм, верхнее, и нижнее, левое — 20 мм. Формат абзаца: полное выравнивание («по ширине»). Отступ красной строки одинаковый по всему тексту.

Страницы должны быть пронумерованы с учётом титульного листа, который не обозначается цифрой. Интервал между строками текста — 1,5. Размер шрифта для названия главы — 16 (полужирный), подзаголовка — 14 (полужирный), текста работы — 14. Точка в конце заголовка, располагаемого посередине листа, не ставится. Заголовки не подчёркиваются. Абзацы начинаются с новой строки и печатаются с отступом в 1,25 сантиметра. Оглавление (содержание) должно быть помещено в начале работы, сформировано автоматически.

Заголовки разделов и подразделов следует печатать на отдельной строке с прописной буквы без точки в конце, не подчеркивая, например: ВВЕДЕНИЕ, ЗАКЛЮЧЕНИЕ. Выравнивание по центру. Отбивка: перед заголовком — 12 пунктов, после — 6 пунктов. Расстояние между названием главы и последующим текстом должно быть равно двум междустрочным интервалам. Такое же расстояние выдерживается между заголовками главы и параграфа. Расстояния между строками заголовка принимают таким же, как и в тексте. Подчеркивать заголовки и переносить слова в заголовке не допускается.

При **цитировании** необходимо соблюдать следующие правила:

- текст цитаты заключается в кавычки и приводится без изменений, без произвольного сокращения цитируемого фрагмента (пропуск слов, предложений или абзацев допускается, если не влечет искажения всего фрагмента, и обозначается многоточием, которое ставится на месте пропуска) и без искажения смысла;

- каждая цитата должна сопровождаться ссылкой на источник, библиографическое описание которого должно приводиться в соответствии с требованиями библиографических стандартов.

Обучающиеся должны провести самостоятельный поиск в сети Интернет необходимой информации по выбранной теме, провести аналитический обзор и сравнительный анализ, подготовить выступление на 5-7 мин. с докладом и презентацией.

Нумерация. Страницы следует нумеровать арабскими цифрами, соблюдая сквозную нумерацию по всему тексту (титульный лист и оглавление включают в общую нумерацию). На титульном листе номер не проставляют. Номер страницы проставляют в центре нижней части листа без точки.

Титульный лист. В верхней части титульного листа пишется, в какой организации выполняется работа, далее буквами увеличенного кегля указывается тип («Курсовая работа») и вариант работы, ниже в правой половине листа — информация, кто выполнил и кто проверяет работу. В центре нижней части титульного листа пишется город и год выполнения.

Библиография

Библиографические ссылки в тексте курсовой работы оформляются в виде номера источника в квадратных скобках. Библиографическое описание (в списке источников) состоит из следующих элементов:

основного заглавия;

обозначения материала, заключенного в квадратные скобки;

сведений, относящихся к заглавию, отделенных двоеточием;

сведений об ответственности, отделенных наклонной чертой;

при ссылке на статью из сборника или периодического издания — сведений о документе, в котором помещена составная часть, отделенных двумя наклонными чертами с пробелами до и после них;

места издания, отделенного точкой и тире;

имени издателя, отделенного двоеточием;

даты издания, отделенной запятой.

Примеры (см. Примечание).

ПРИМЕЧАНИЕ

Список элементов библиографической записи сокращен

Книга, имеющая не более трех авторов:

Максимов, Н. В. Архитектура ЭВМ и вычислительных систем [Текст]: учеб. для вузов / Н. В. Максимов, Т. Л. Партыка, И. И. Попов. — М.: Инфра, 2005.

Книга с четырьмя и более авторами, сборник и т. п.:

Мировая художественная культура [Текст]: в 2-х т. / Б. А. Эренгросс [и др.]. — М.: Высшая школа, 2005. — Т. 2.

Статья из сборника:

Цивилизация Запада в 20 веке [Текст] / Н. В. Шишова [и др.] // История и культурология: учеб. пособие для студентов. — М, 2000. — Гл. 13. — С. 347-366.

Статья из журнала:

Мартышин, О. В. Нравственные основы теории государства и права [Текст] / О. В. Мартышин // Государство и право. — 2005. — № 7. — С. 5-12.

Электронное издание:

Сидыганов, Владимир Устинович. Модель Москвы [Электронныи ресурс]: электронная карта Москвы и Подмосковья / Сидыганов В. У., Толмачев С. Ю., Цыганков Ю. Э. — Версия 2.0. — М.: Formoza, 1998.

Интернет-ресурс:

Бычкова, Л. С. Конструктивизм / Л. С. Бычкова // Культурология 20 век. — (http//www.philosophy.ru/edu/ref/enc/k.html).

Презентация должна соответствовать требованиям эргономики.

4.6.2 Темы курсовых работ

Полный перечень тем работ приводится в учебно-методическом комплексе.

1. Задача Прима-Краскала (жадный алгоритм)

Дана плоская страна и в ней n городов. Нужно соединить все города телефонной связью так, чтобы общая длина телефонных линий была минимальной.

Уточнение задачи. В декартовой системе координат положение i-го города, i = 1,...,n, задано парой координат (x[i],y[i]). d[i,j] - декартово расстояние между i-ым городом и j-ым городом , j=1,...,n. В задаче речь идет о телефонной связи, т. е. подразумевается транзитивность связи: если i-й город связан с j-ым, а j-ый с k-ым, то i-й связан с k-ым.

Подразумевается также, что телефонные линии могут разветвляться только на телефонной станции, а не в чистом поле. Наконец, требование минимальности (вместе с транзитивностью) означает, что в искомом решении не будет циклов. В терминах теории графов задача Прима-Краскала выглядит следующим образом:

Дан граф с n вершинами; длины ребер заданы матрицей (d[i,j]), i,j = 1,...,n. Найти остовное дерево минимальной длины. Как известно, дерево с n вершинами имеет n-1 ребер. Оказывается, каждое ребро надо выбирать жадно (лишь бы ни возникали циклы).

Алгоритм Прима-Краскала (краткое описание)

В цикле n-1 раз делай:

выбрать самое короткое еще не выбранное ребро при условии, что оно не образует цикл с уже выбранными.

Выбранные таким образом ребра образуют искомое остовное дерево.

Напишите программу для решения задачи Прима-Краскала.

2. Шифр Цезаря

Юлий Цезарь был, якобы первым, кто придумал собственно шифр. Алфавит размещается на круге по часовой стрелке (при этом в русском алфавите, после А идет Б, а после Я - А). Для зашифровки буквы текста заменяются буквами, отстоящими по кругу на заданное число букв дальше по часовой стрелке.

Если, скажем, сдвиг на 3, то вместо i-й используется (i+3)-я буква, например, вместо А пишется Г а вместо Я пишется В. При расшифровке наоборот берут букву на заданное число букв ближе, т. е. двигаясь против часовой стрелки.

Шифр Цезаря расшифровать легко. Известны вероятности букв p[i],i =1,2,...,n, в языке сообщения (n - число букв в алфавите). подсчитаем частоты букв f[i] в зашифрованном сообщении. Если оно не очень короткое, то f[i] должны сравнительно хорошо согласовываться с p[i]: f[i] = p[i-s] для некоторого сдвига s. Затем начнем делать перебор по сдвигам. Когда сдвиг не угадан, общее различие между p[i] и f[i+s] , равное D(s) = S | p[i] -f[i+s] | (суммирование берется по всем i от 1 до n), будет велико, а когда сдвиг угадан - мало.

Минимизация D(s) по всем s = 1,2,...,n дает ключ к расшифровке кода Цезаря.

Напишите и испытайте программу взлома шифра Цезаря.

3. Игра Жизнь

Это игра создана в 1970 г., ее автор - английский математик Дж. Конвей (Conway). В этой игре партнер не нужен - в неё можно играть одному. Возникающие в процессе игры ситуации очень похожи на реальные процессы, происходящие при зарождении, развитии и гибели колонии живых организмов.

Правила игры. Вообразите бесконечное поле, разделенное на клетки. На каждой клетке поля живет, рождается или погибает животное. Это зависит от условий Среды, т. е. от того, сколько соседей у него на ближайших восьми клетках (четырех по сторонам и четырех по углам).

Действуют три правила существования животных:

1. Каждое животное, у которого два или три соседа, живет и сохраняется до следующего поколения.

2. Животное погибает, если у него более нежели три соседа (от недостатка места), совсем нет соседей или только один сосед (от одиночества).

3. Когда рядом с какой-нибудь клеткой есть три животных (соседа), то на этой клетке рождается новое животное.

4. Важно понять, что животные погибают и рождаются одновременно. Они образуют одно поколение. За один ход в игре в соответствии с упомянутыми правилами осуществляется переход от одного поколения к другому.

Дж. Конвей рекомендует следующий способ осуществления ходов (при наличии клетчатой доски и косточек двух цветов):

1) начать с желаемой конфигурации (колонии животных), состоящей из черных косточек;

2) найти все косточки, которые должны погибнуть, и на каждую из них одеть по одной черной косточке;

3) найти все свободные клетки, на которых должно родиться животное, и положить на них по одной белой косточке;

4) удалить с доски всех погибших животных (т. е. столбики из двух косточек), а новорожденных (белые косточки) заменить черными. Выполнив эти операции, т. е. после первого хода, получим второе поколение. Аналогичным

образом происходит и все остальные ходы в игре. Так получаются все новые поколения.

Тема. Напишите программу, моделирующую колонию Жизни. Исходными данными служит начальное расположение животных (заданное пользователем или получаемое случайно - реализовать оба случая), а в качестве результата нужно получить вид сверху в графическом режиме всех поколений колонии.

Некоторые колонии разрастаются невероятным образом при весьма скромных начальных размерах. Есть другие колонии, которые медленно перемещаются по пустыне, переходя на все новые и новые территории. Ваша программа должна обрабатывать большие колонии без чрезмерной траты памяти или времени. Многократный просмотр большого массива для построения следующих поколений - это банальный подход; здесь хороший программист выбрал бы более экономичные структуры данных и алгоритмы. Вам, возможно, захочется испытать какой-либо метод, отслеживающий только занятые квадраты. В программе нельзя определить бесконечно большое поле. Должно хватать поля некоторой известной величины m*n. Что делать, если эволюция достигает границ поля? Один из возможных выходов - прервать эволюцию. Однако эволюция могла бы продолжаться, если бы мы устранили границы поля: соединили бы любые два противоположных края поля, затем концы полученного цилиндра. Полученная фигура, имеющая форму бублика, называется тор. Используйте этот подход в вашей программе (для этого достаточно более аккуратно определить соседей клетки, находящейся на краю поля). История колонии Жизнь зачаровывает, если её просматривать как фильм, но она будет еще более увлекательней, если предстанет в цвете. Каждой клетке при рождении может быть приписан некоторый цвет, определяемый, возможно, её поколением или генами, переданными ей родителями.

Циклические, но при этом движущиеся колонии (а таких немало) великолепны в своем сверкающем многоцветном наряде.

4. Солнечная система

Тема: программирование астрономической модели солнечной системы. Модель описывает Солнце и планеты Меркурий, Венеру, Землю, Марс и их спутники. Программа работает следующим образом: на экране изображается Солнце и планеты со своими спутниками располагаются вокруг Солнца на своих астрономических местах. Планеты начинают вращаться вокруг Солнца по своим орбитам с правильным соотношением скоростей. В то же время спутники начинают вращаться вокруг своих планет по траекториям, складывающимся из двух вращательных движений: вращение планеты вокруг Солнца и вращение спутника вокруг планеты. Чтобы обобщить определения разных небесных, определите объект Tbody. Планеты и спутники так же, как и Солнце, - это небесные тела. Их надо определить как объекты-наследники от Tbody. Объекты-наследники должны содержать поля: 1)текущие координаты тела; 2) центр, вокруг которого тело вращается; 3) радиус орбиты; 4) список спутников; 5) скорость вращения; 6) размер; 7) цвет тела.

Вращение как планет, так и спутников вокруг центрального тела происходит по одним и тем же законом природы. Для планет телом, вокруг которого они вращаются, является Солнце, а для каждого спутника некоторая планета. Это движение для всех небесных тел можно определить одним методом - Вращайся! Идея метода состоит в осуществлении движения тела наращиванием углового перемещения с шагом в 10 градусов.

Перемещение каждого тела вычисляется в виде относительной величины, зависящей от значения его скорости. При каждом изменении угла вычисляются новые координаты положения тела. Каждая планета, начав вращаться должна запустить соответствующий метод вращения для своих спутников.

Относительные параметры для планет и спутников

Название	Радиус	Скорость	Размер
Меркурий	58	0.416	3
Венера	108	0.416	5
Земля	150	0.1	6
Марс	228	0.053	4
Луна	15	1.3	2
Фобос	7	114.4	1
Деймос	12	30.4	1

5. Множество Мандельброта

В Книге рекордов Гиннеса самым сложным математическим объектом названо множество Мандельброта. Это плоское множество является ярким примером фрактала.

Фракталы - это математические объекты, имеющие дробную размерность в отличие от традиционных геометрических фигур целой размерности (например, одномерных линий или двухмерных поверхностей). Фракталы - это нечто больше, чем математический курьез. Они дают чрезвычайно компактный способ описания объектов и процессов. Многие структуры обладают фундаментальным свойством геометрической регулярности, известной как инвариантность по отношению к масштабу, или самоподобие. Если рассматривать эти объекты в различном масштабе, то постоянно обнаруживаются одни и те же фундаментальные элементы. Эти повторяющиеся закономерности определяют дробную, или фрактальную, размерность структуры. В природе все фрактально: облака, изрезанная линия побережья, кромка листа, нервные и кровяные сосуды и т. д.

Множество Мандельброта описывает поведение динамического процесса, определенного на комплексных числах формулой $z(n+1) = z(n)*z(n) + c$. (1)

Опишем алгоритм построения окрашенного в черный цвет множества Мандельброта с окружением, раскрашенным в разные цвета. Для произвольного комплексного числа $c = x + i*y$ положим $z(0) = 0$ и устроим итерацию по формуле 1. Максимальное число итераций Max =150. Для последовательности $z(n)$ имеются две возможности:

1. Числа становятся все большими и большими, стремясь к бесконечности.

2. Точки находятся и продолжают оставаться на расстоянии меньшим 2 от 0.

Так вот, множество Мандельброта - это множество тех чисел c, для которых выполняется вторая возможность. Граница множества сильно изрезанна. Причем под лупой она выглядит столь же изломанной, как и без нее. Она напоминает линию морского берега, многие естественные границы, которые становятся тем длиннее, чем более мелкий масштаб используется для измерения. Одной из характерных особенностей этой границы является её самоподобие. Если взглянуть на любой из её поворотов или заливов, то можно обнаружить, что одна и та же форма встречается в различных местах и имеет разные размеры.

В программе каждая точка (пиксель) экрана представляет соответствующее комплексное число c. Если число Max увеличить, то граница множества определится точнее, так как для некоторых точек c последовательности z(n) уйдут всё-таки на бесконечность. Все точки множества Мандельброта отметим черным цветом. Для всех других точек с соответствующая последовательность z(n) уходит на бесконечность, причем скорость ухода оценивается соответствующим цветом точки c, пропорциональным количеству итераций, достаточным для того, чтобы z(n)*z(n) стало большим 4. Всего используется цветовая палитра из 16 цветов и так как Max-1 значительно больше 15, то цвета периодически повторяются. Окраска внешности множества Мандельброта позволяет более точно увидеть границу множества.

Множество Мандельброта строится в прямоугольнике с координатами xmin= -2.25, xmax=0.75, ymin=-1.5, ymax=1.5 (т. е. пиксель с координатами (0,0) представляет комплексное число -2.25 + i*1.5). Программа должна позволять пользователю менять координаты вершин прямоугольника, чтобы можно было изобразить в увеличенном виде отдельные фрагменты множества Мандельброта. Так как множество Мандельброта строится достаточно медленно, то необходимо предусмотреть возможность записи создаваемого изображения в файл. Такой файл просто хранит цвета всех пикселей экрана. Необходима так же вспомогательная программа, которая сразу же извлечет предварительно созданное изображение из файла и изобразит на экране. Динамическое изменение цветовой палитры в цикле позволит получить более красочные изображения множества Мандельброта.

6. Перенос слов

Как показывают многочисленные эксперименты, разбиение русского слова на части для переноса с одной строки на другую с большой вероятностью выполняются правильно, если пользоваться следующими простыми приемами:

1) Две идущие подряд гласные можно разделить, если первой из них предшествует согласная, а за второй идет хотя бы одна буква (буква й при этом рассматривается вместе с предшествующей гласной как единое целое).

2) Две идущие подряд согласные можно разделить, если первой из них предшествует гласная, а в той части слова, которая идет за второй согласной,

имеется хотя бы одна гласная (буквы ь, ъ вместе с предшествующей согласной рассматриваются как единое целое).

3) Если не удается применить пункты 1), 2), то следует попытаться разбить слово так, чтобы первая часть содержала более чем одну букву и оканчивалась на гласную, а вторая содержала хотя бы одну гласную. Вероятность правильного разбиения увеличивается, если предварительно воспользоваться хотя бы неполным списком приставок, содержащих гласные, и попытаться прежде всего выделить из слова такую приставку.

Дан текст на русском языке. Выполнить форматирование его строк по длине с помощью переноса слов.

7. Морской бой

На поле 10 на 10 позиций стоят невидимые вражеские корабли: 4 корабля по одной клетке, три корабля по 2 клетки, 2 корабля по 3 клетки, 1 корабль в 4 клетки. Позиции указываются русскими буквами от А до К (по строкам) и цифрами от 1 до 10 (по столбцам).

Конфигурация и положение кораблей на поле выбираются с помощью датчика случайных чисел. Если клетка корабля угадана играющим верно, она отмечается крестиком; в противном случае точкой.

Написать программу для игры против компьютера в односторонний морской бой.

8. Линейные фракталы

Фракталы - это математические объекты, имеющие дробную размерность в отличие от традиционных геометрических фигур целой размерности (например, одномерных линий или двухмерных поверхностей). Фракталы - это нечто больше, чем математический курьез.

Они дают чрезвычайно компактный способ описания объектов и процессов. Многие структуры обладают фундаментальным свойством геометрической регулярности, известной как инвариантность по отношению к масштабу, или самоподобие. Если рассматривать эти объекты в различном масштабе, то постоянно обнаруживаются одни и те же фундаментальные элементы. Эти повторяющиеся закономерности определяют дробную, или фрактальную, размерность структуры. В природе все фрактально: облака, изрезанная линия побережья, кромка листа, нервные и кровяные сосуды и т. д.

Некоторые из фракталов называются линейными, потому что строятся с помощью линейных (аффинных) преобразований плоскости. Такие преобразования изображение перемещают, сжимают, отражают, вращают и трансформируют произвольным образом при условии, что прямые линии на изображении остаются прямыми после преобразования.

Описывая фракталы посредством аффинных преобразований, мы можем значительно уменьшить количество данных, необходимых для передачи изображения по линиям связи или для хранения его в памяти компьютера. Сложная форма, подобная форме листа папоротника, может быть полностью описана линейным алгоритмом, основанным лишь на 28 числовых параметрах. Заметим,

что представление того же листа в точечном виде, как телевизионное изображение, требует несколько сотен тысяч числовых величин.

Алгоритм построения линейных фракталов.

Аффинное преобразование евклидовых координат задается с помощью двух линейных уравнений:

$x\_ = a\,x + b\,y + e$

$y\_ = c\,x + d\,y + f$

Для построения фрактала поступают следующим образом:

Выбирается несколько афинных преобразований и каждому преобразованию сопоставляется некоторая вероятность его использования. Вид аффинных преобразований, их количество и вероятность зависят от конкретного фрактала.

Так, например, для листа папоротника используются 4 преобразования:

a	b	c	d	e	f	p
0.00	0.00	0.00	0.16	0.00	0.00	0.01
0.85	0.04	-0.04	0.85	0.00	1.60	0.85
0.20	-0.26	0.23	0.22	0.00	1.60	0.07
0.15	0.28	0.26	0.24	0.00	0.44	0.07

Берется начальная точка с координатами x = 0 и y = 0. Затем применяется одно из данных преобразований координат для определения новых значений и, какое именно преобразование применить определяется случайным образом в соответствии с заданной вероятностью p. Полученная точка изображается на плоскости цветом связанным с примененным преобразованием. Этот процесс повторяется достаточное число раз, пока не построится достаточно реалистическое изображение. Чтобы изображение удачно было расположено на экране нужно задать логический экран - в случае листа папоротника :

по оси x от -4 до 0

по оси y от 6 до 10

Замена значений коэффициентов b и c для листа папоротника во втором уравнении соответственно на 0.06 и - 0.06 увеличивают кривизну стебля папоротника. Замена их на 0.02 и -0.02 - уменьшают кривизну. Если исключить первое уравнение, то пропадает стебель.

Программа должна строить несколько фракталов.

9. Одномерные клеточные автоматы

Рассмотрим клетки, расположенные вдоль прямой. Каждая из клеток может иметь состояние 0 или 1. На каждом шаге по времени новое состояние клетки вычисляется из старого состояния клетки и состояний её соседей. На экране каждый горизонтальный ряд показывает состояния клеток с помощью соответствующего цвета в очередной момент времени. Временная ось идет сверху вниз и каждый ряд вычисляется на основе предыдущего ряда. Различные классы линейных клеточных автоматов могут быть определены в зависимости от того, как много различных значений (состояний) клетка может иметь (k), и как много соседей на каждой стороне клетки используются при вычислении нового состояния ®. Правила для определения следующего состояния клетки используют сумму состояний самой клетки и клеток-соседей. Сумма со-

стояний отображается правилом на новое состояние клетки. Если k=2, r=1, то только ближайшие соседи клетки используются. Если каждая из клеток имеет состояние 1, то максимальная сумма 1+1+1 равна 3 и сумма может меняться от 0 до 3, т. е. имеем четыре величины. Поэтому правило преобразования можно задать с помощью строки из четырех двузначных цифр. Например, правило 1010, начиная с правой цифры, задает отображение суммы 0 -> 0, 1 -> 1, 2 -> 0, 3 ->1 (таким образом, если сумма равна 2, то новое состояние равно 0). Для k=2 и r=2 каждая клетка (на экране она изображается пикселем) новое значение получает в зависимости от значений двух соседних клеток с каждой стороны. Соответствующие правила должны иметь 6 двузначных цифр. Если k=3 и r=1, то каждая клетка может иметь значение 0, 1 или 2.

Учитывается только единственный сосед с каждой стороны, поэтому результат преобразования задается правилом из 7 цифр и т. д.

Напишите программу для построения линейных клеточных автоматов следующих классов kr = 21, 31, 41, 22, 32, 42.

k	r	количество цифр в правиле	пример правила
2	1	4	1010
3	1	7	1211001
4	1	10	3311100320
2	2	6	0110110
3	2	11	21212002010
4	2	16	2300331230331001

Начальная строка задается либо случайным образом, либо пользователем.

Как ведут себя клеточные линейные автоматы? Происходит ли переход к однородному состоянию независимо от начальных данных? Существуют ли классы автоматов с локализованными стационарными или периодическими конфигурациями? Есть ли автоматы с хаотическим временным поведением? Есть ли автоматы, которые ведут себя по разному при различных начальных данных? Необходимо ответить на эти вопросы.

10. Инженерный калькулятор

Написать программу, которая бы вычисляла арифметическое выражение, введенное с клавиатуры. Арифметическое выражение может содержать числа (в том числе и в экспоненциальной форме, например 1.2e-10), символы арифметических операций, скобки, функции синуса, косинуса, тангенса, логарифма, экспоненты.

Разбор арифметического выражения рекомендуется проводить следующим образом. Создается рекурсивная функция gettoken(). В зависимости от текущего символа входной строки она производит следующие действия:

+, -, /, * → gettoken(); выполнить операцию

цифра → положить в стек цифру

(→ gettoken(); пропустить)

символ → выяснить что за функция; gettoken(); вычислить значение

5. МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ОБУЧАЮЩИХСЯ ПО ОСВОЕНИЮ ДИСЦИПЛИНЫ

Лекционный курс базируется на пассивном методе обучения, реализующем традиционную объяснительно-иллюстративную образовательную технологию, в рамках которой студенты выступают в роли слушателей, воспринимающих учебный материал, и участвующих в дискуссиях и экспресс-опросах.

При прослушивании лекций рекомендуется в конспекте отмечать все важные моменты, на которых заостряет внимание преподаватель, в частности те, которые направлены на качественное выполнение соответствующей практической работы.

Преподавателем запланировано использование при чтении лекций технологии учебной дискуссии. Поэтому рекомендуется фиксировать для себя интересные моменты с целью их активного обсуждения на дискуссии в конце лекции.

Залогом качественного выполнения практических работ является самостоятельная подготовка путем повторения материалов лекций. Рекомендуется подготовить вопросы по неясным моментам и обсудить их с преподавателем в начале каждого занятия.

Преподавателем запланировано применение на лабораторных занятиях технологий развивающейся кооперации, коллективного взаимодействия, разбора конкретных ситуаций.

Для текущего контроля успеваемости по очной форме обучения преподавателем используется балльно-рейтинговая система контроля и оценки академической активности. Поэтому настоятельно рекомендуется тщательно прорабатывать материал дисциплины при самостоятельной работе, участвовать во всех формах обсуждения и взаимодействия, как на лекциях, так и на практических занятиях в целях лучшего освоения материала и получения высокой оценки по результатам освоения дисциплины.

Выполнение самостоятельной работы подразумевает самостоятельное изучение разделов дисциплины, подготовку к практическим занятиям, к рубежным контролям (для очной формы обучения), курсовой работы, подготовку к зачету и экзамену.

Рекомендуемая трудоемкость самостоятельной работы представлена в таблице.

Наименование вида самостоятельной работы	**Рекомендуемая трудоемкость, акад. час.**	
	Очная форма обучения	**Заочная форма обучения**
1 семестр (очная форма обучения) *2 семестр (заочная форма обучения)*		
Самостоятельное изучение тем дисциплины:	**28**	**88**
1. Модуль datetime, его назначение, основные функции и методы. Примеры использования.	4	12

2 Понятие модуля в Python. Общие требования к создаваемым модулям. Примеры создания модуля.	2	12
3.Различные степени копирования объектов. Модуль copy. Создание копий различных уровней. Примеры использования.	4	12
4. Шифрование данных средствами Python. Модуль crypt. Примеры использования	6	20
5. Различные способы форматирования данных. Оператор format. Примеры использования этого оператора.	6	14
6. Взаимодействие с Интернетом. Разбор URL-адреса. Обмен данными по протоколу HTTP. Кодирование и декодирование строки запроса. Модуль urllib.request.	6	18
Подготовка к лабораторным занятиям (по 1 часу на каждое занятие (очная форма обучения) и по 2 часа (заочная форма обучения))	**14**	**6**
Подготовка к рубежным контролям (по 1 часу на каждый рубеж)	**2**	**-**
Подготовка к практическим занятиям (по 2 часа на каждое занятие)	**-**	**2**
Выполнение контрольной работы	**18**	**18**
Подготовка к зачету	**18**	**18**
Всего:	**80**	**132**
2 семестр (для очной формы обучения) *3 семестр (для заочной формы обучения)*		
Самостоятельное изучение тем дисциплины:	**21**	**93**
1. Настройка среды программирования Visual Studio. Структура проектов различных типов. Настройка различных параметров.	6	28
2. Построение графиков и фигур в среде Microsoft Visual C++. Крандаш. Кисть.	5	20
3. Изучение дополнительных компонентов среды программирования Visual Studio (ErrorProvider, MonthCalendar, NumericUpDown, DomainUpDown и др.)	6	28
4. Различные способы преобразования между регулируемыми и нерегулируемыми указателями. Маршаллинг.	4	17
Подготовка к лабораторным занятиям (по 1 часу на каждое занятие (очная форма обучения) и по 2 часа (заочная форма обучения))	**16**	**6**
Подготовка к рубежным контролям (по 1 часу на каждый рубеж)	**2**	**-**
Подготовка к практическим занятиям (по 1 часу на каждое занятие (очная форма обучения) и по 2 часа (заочная форма обучения))	**6**	**4**
Выполнение курсовой работы	**36**	**36**
Подготовка к экзамену	**27**	**27**
Всего:	**108**	**166**

6. ФОНД ОЦЕНОЧНЫХ СРЕДСТВ ДЛЯ АТТЕСТАЦИИ ПО ДИСЦИПЛИНЕ

6.1. Перечень оценочных средств

1. Балльно-рейтинговая система контроля и оценки академической активности студентов в КГУ (для очной формы обучения).

2. Отчеты студентов по лабораторным и практическим занятиям.

3. Контрольная работа.

4. Банк тестовых заданий к рубежным контролям № 1 - № 4 (для очной формы обучения).

5. Курсовая работа.

6. Список вопросов к зачету.

7. Список вопросов к экзамену.

6.2. Система балльно-рейтинговой оценки работы студентов по дисциплине

1 семестр

№	Наименование	Содержание						
1	Распределение баллов за семестры по видам учебной работы, сроки сдачи учебной работы **(доводятся до сведения студентов на первом учебном занятии)**	Распределение баллов для зачета						
		Посещение лекций	Посещение лабораторных занятий	Выполнение лабораторных работ	Выполнение контрольной работы	Рубежный контроль № 1	Рубежный контроль № 2	Зачет
		До16 баллов (16 * 1 балл = 16 б.)	До 8 баллов (16 * 0,5 балл = 8 б.)	До 28 баллов (3 * 4 балла + 1 * 6 баллов + 5 * 2 балла = 28 б.)	До 9 баллов	До 4 баллов	До 5 баллов	До 30 баллов
2	Критерий пересчета баллов в традиционную оценку по итогам работы в семестре и зачета	- 60 и менее баллов – не зачтено (компетенции не освоены); - 61…73 – зачтено (пороговый уровень освоения компетенций); - 74…90 – зачтено (базовый уровень освоения компетенций); - 91…100 – зачтено (продвинутый уровень освоения компетенций).						
3	Критерии допуска к промежуточной аттестации, возможности получения автоматического зачета (экзаменационной оценки) по дисциплине, возможность получения бонусных баллов	Для допуска к промежуточной аттестации (зачету) студент должен набрать по итогам текущего и рубежного контроля не менее 50 баллов, выполнить рубежный контроль № 1 и 2, выполнить и защитить 9 лабораторных работ. Для получения зачета автоматом студенту необходимо набрать за семестр минимум 61 балл. По согласованию с преподавателем студенту могут быть добавлены дополнительные (бонусные) баллы за активное участие на консультациях, оригинальность принятых решений в ходе выполнения лабораторных работ, за участие в значимых учебных и внеучебных мероприятиях кафедры и получен зачет «автоматически».						

№	Наименование	Содержание
4	Формы и виды учебной работы для неуспевающих (восстановившихся на курсе обучения) студентов для получения недостающих баллов в конце семестра	В случае если к промежуточной аттестации набрана сумма менее 50 баллов, не выполнены все задания, то студенту необходимо выполнить дополнительные задания, до конца последней (зачетной) недели семестра. При этом необходимо проработать материал всех пропущенных лабораторных занятий. Формы дополнительных заданий (назначаются преподавателем): - выполнение и защита невыполненных студентом заданий лабораторных занятий – до 2 баллов; - прохождение рубежного контроля – до 6 баллов; - выполнение письменных работ по теме, предложенной преподавателем – до 10 баллов. Ликвидация академических задолженностей, возникших из-за разности в учебных планах при переводе или восстановлении, проводится путем выполнения дополнительных заданий, форма и объем которых определяется преподавателем.

2 семестр

№	Наименование	Содержание							
1	Распределение баллов за семестры по видам учебной работы, сроки сдачи учебной работы **(доводятся до сведения студентов на первом учебном занятии)**	Распределение баллов для экзамена							
		Посещение лекций	Посещение лабораторных занятий	Выполнение лабораторных работ	Посещение практических занятий	Выполнение практическихработ	Рубежный контроль № 3	Рубежный контроль № 4	Экзамен
		До12 баллов (12 * 1 балл = 12 б.)	До 8 баллов (16 * 0,5 баллов = 8 б.)	До 24 баллов (2 * 4 балла, 1 * 6 баллов, 2 * 3 балла, 2 * 2 балла = 24 б.)	До 4 баллов (8 * 0,5 балл = 4 б.)	До 12 баллов (3 * 4 балла = 12 б.)	До 5 баллов	До 5 баллов	До 30 баллов
2	Критерий пересчета баллов в традиционную оценку по итогам работы в семестре и зачета	60 и менее баллов – неудовлетворительно; 61…73 – удовлетворительно; 74… 90 – хорошо; 91…100 – отлично							

№	Наименование	Содержание
3	Критерии допуска к промежуточной аттестации, возможности получения автоматического зачета (экзаменационной оценки) по дисциплине, возможность получения бонусных баллов	Для допуска к промежуточной аттестации (экзамену) студент должен набрать не менее 50 баллов и принять активное участие в лекциях, выполнить рубежный контроль №№ 3, 4 и все практические и лабораторные работы. Для получения «автоматически» экзаменационной оценки «удовлетворительно» студенту необходимо набрать за семестр минимум 68 баллов. По согласованию с преподавателем студенту, набравшему минимум 68 баллов, могут быть добавлены дополнительные (бонусные) баллы за активное участие на консультациях, оригинальность принятых решений в ходе выполнения практических и лабораторных работ, за участие в значимых учебных и внеучебных мероприятиях кафедры и выставлена за экзамен «автоматически» оценка «хорошо» или «отлично».
4	Формы и виды учебной работы для неуспевающих (восстановившихся на курсе обучения) студентов для получения недостающих баллов в конце семестра	В случае если к промежуточной аттестации набрана сумма менее 50 баллов, не выполнены все задания, то студенту необходимо выполнить дополнительные задания, до конца последней (зачетной) недели семестра. При этом необходимо проработать материал всех пропущенных практических и лабораторных занятий. Формы дополнительных заданий (назначаются преподавателем): - выполнение и защита невыполненных студентом заданий лабораторных занятий – до 3 баллов; - выполнение и защита невыполненных студентом заданий практических занятий – до 2 баллов; - прохождение рубежного контроля – до 6 баллов; - выполнение письменных работ по теме, предложенной преподавателем – до 10 баллов. Ликвидация академических задолженностей, возникших из-за разности в учебных планах при переводе или восстановлении, проводится путем выполнения дополнительных заданий, форма и объем которых определяется преподавателем.

6.3. Процедура оценивания результатов освоения дисциплины

Рубежные контроли проводятся в форме письменных ответов на вопросы. Каждому студенту предлагается по 2 вопроса, за ответ на который студент может получить до 2 баллов (рубеж №1) и до 2,5 баллов (рубежи №№ 2, 3, 4).

Перед проведением каждого рубежного контроля преподаватель прорабатывает со студентами основной материал соответствующих разделов дисциплины в форме краткой лекции-дискуссии. Примерные варианты заданий для 1 - 4 рубежного контроля приведены ниже. На каждый рубежный контроль студенту отводится 2 академических часа.

Преподаватель оценивает в баллах результаты каждого студента и заносит в ведомость учета текущей успеваемости.

Зачет состоит из 3 вопросов. Вопросы к зачету доводятся до студентов на последней лекции в семестре. Каждый вопрос оценивается до 10 баллов. На подготовку ответа студенту отводится 1 астрономический час.

Билеты на экзамен состоят из 2 вопросов и практического задания. Ответы на каждый вопрос оцениваются до 10 баллов, выполнение практического задания оценивается до 10 баллов. Время, отводимое студенту на подготовку к ответу на экзаменационный билет, составляет 1 астрономический час.

Результаты текущего контроля успеваемости, зачета и экзамена заносятся преподавателем в зачетно-экзаменационные ведомости, которые сдаются в организационный отдел института в день зачета и экзамена, а также выставляются в зачетную книжку студента.

6.4. Примеры оценочных средств для рубежных контролей, зачета и экзамена

Примерные вопросы для рубежных контролей

Рубежный контроль 1: Список вопросов

1. Назовите основные типы данных
2. Из чего состоит любое выражение? К чему сводится любое выражение?
3. В чем различие между выражением и инструкцией?
4. Перечислите операторы сравнения.
5. В чем суть различия между оператором равенства и оператором присваивания?
6. Объясните, что такое условие и где используются условия.
7. Какую комбинацию клавиш нужно нажать, чтобы вывести программу из бесконечного цикла?
8. Чем различаются инструкции break и continue?
9. Чем различаются вызовы функций range (10), range (0,10) и range (0, 10, 1) в цикле for?
10. Напишите короткую программу, выводящую числа от 1 до 10 с помощью цикла for. Затем напишите аналогичную программу, в которой используется цикл while.
11. Что означают эти скобки: []?
12. Как бы вы присвоили значение 'hello' в качестве третьего элемента списка, хранящегося в переменной spam? (Предполагается, что в переменной spam содержится список [2, 4, 6, 8, 10].)
13. Переменная spam содержит список ['a', 'b', 'c', 'd']. Каково значение выражения spam[int ('3' * 2 /11] ?
14. Переменная spam содержит список ['a', 'b', 'c', 'd']. Каково значение выражения spam[-1]?
15. Переменная spam содержит список ['a', 'b', 'c', 'd']. Каково значение выражения spam[:2]?
16. Переменная bacon содержит список [3.14, 'cat', 11, 'cat', True]. Каково значение выражения bacon.index ('cat')?
17. Переменная bacon содержит список [3.14, 'cat', 11, 'cat', True]. Как будет выглядеть список, хранящийся в переменной bacon, после следующего вызова: bacon.append (98) ?

18. Переменная bacon содержит список [3.14, 'cat', 11, 'cat', True]. Как будет выглядеть список, хранящийся в переменной bacon, после следующего вызова: bacon.remove ('cat') ?
19. Какие операторы используются для конкатенации списков?
20. В чем состоит различие между предусмотренными для списков методами append() и insert()?
21. Назовите несколько общих признаков списков и строк.
22. Чем кортежи отличаются от списков?
23. Как бы вы записали кортеж, содержащий единственное значение в виде целого числа 42?
24. Как преобразовать список в кортеж? Как преобразовать кортеж в список?
25. Переменные, которые «содержат» список, на самом деле не содержат непосредственно сам список. Что же тогда они содержат?
26. Что такое экранированные символы?
27. Что представляют собой экранированные символы \n и \t?
28. Как добавить символ обратной косой черты (\) в строку?
29. Строковое значение "How's Moving Castle" — это допустимая строка. Почему она не вызовет ошибку, несмотря на наличие неэкранированного символа апострофа в слове How's?
30. Если вы не хотите вставлять символ \n и свою строку; то как вы напишете строку, содержащую символы новой строки?
31. Каковы будут результаты вычисления приведенных ниже выражений?
- 'Hello world!' [1]
- 'Hello world!' [0:5]
- 'Hello world!' [:5]
- 'Hello world!' [3:]

32. Каковы будут результаты вычисления приведенных ниже выражений?
- 'Hello'.upper()
- 'Hello'.upper().isupper()
- 'Hello'.upper().lower()

33. Каковы будут результаты вычисления приведенных ниже выражений?
- 'Remember, remember, the fifth of November.'.split()
- '-'.join('There can be only one.'.split())

34. Как удалить пробельные символы в начале и конце строки?

Рубежный контроль 2: Список вопросов

1. Что дает использование функций в программах?
2. Когда именно выполняется код функции: когда она определяется или когда вызывается?
3. С помощью какой инструкции создаются функции?
4. Чем отличается определение функции от ее вызова?
5. Сколько глобальных областей видимости может иметь программа на языке Python? Сколько локальных?

6. Что происходит с переменными, находящимися в локальной области видимости, при возврате из функции?
7. Что такое возвращаемое значение? Может ли возвращаемое значение быть частью выражения?
8. Каково возвращаемое значение функции, если в ней отсутствует инструкция return?
9. Как заставить переменную в функции ссылаться на глобальную переменную?
10. Что такое тип данных None?
11. Какой код помещается в блок try? Какой код помещается в блок except?
12. Относительно чего задается относительный путь?
13. С чего начинается абсолютный путь?
14. Каково назначение функций os.getcwd() и os.chdir())?
15. Что собой представляют папки . и ..?
16. Назовите три возможных значения аргумента, задающие режим открытия файла, которые могут передаваться функции open().
17. Что происходит при открытии существующего файла в режиме записи?
18. Чем различаются методы read() и readlines()?
19. Какая функция используется для переименования файлов?
20. Что такое класс? Что такое объект (экземпляр класса)? Что такое поле класса?
21. Общая структура описания класса. Назначение аргумента self.
22. Как создается экземпляр класса?
23. Что такое конструктор класса?
24. Что такое атрибуты объекта класса? Когда их можно добавлять, изменять?
25. Как осуществляется вызов метода класса?
26. Каково назначение наследования? Как соотносятся между собой базовый и производный классы?
27. Можно ли изменять унаследованные методы?
28. Как загрузить готовое изображение?
29. Как создать новое изображение?
30. Перечислите основные методы, предназначенные для работы с изображением.
31. Класс Draw, его назначение. Основные методы этого класса.
32. Вывод текста на графическое изображение.
33. Как выглядит код для пустого словаря?
34. Как выглядит элемент словаря с ключом “foo” и значением 42?
35. Опишите основные различия между словарем и списком.
36. Что произойдет при попытке получения доступа к элементу spam[“foo”], если spam —это {“bar”: 100}?
37. Если в переменной spam хранится словарь, то в чем состоит разница между выражениями ‘cat’ in spam и ‘cat’ in span.keys ()?

38. Если в переменной spam хранится словарь, то в чем состоит разница между выражениями ‘cat’ in spam и ‘cat’ in span.values ()?
39. Что возвращает метод search () ?
40. В синтаксисе регулярных выражений круглые скобки и точки имеют особый смысл. Как бы вы указали в регулярном выражении, что символы круглых скобок и точки сами являются объектом поиска?
41. Что означает символ | в регулярных выражениях?
42. Какие две функции выполняет символ ? в регулярных выражениях?
43. В чем разница между символами + и * в регулярных выражениях?
44. В чем разница между записями {3} и {3,5} в регулярных выражениях?
45. Что означают сокращенные символьные классы \d, \w и \s в регулярных выражениях?
46. Что означают сокращенные символьные классы \D, \W к \S в регулярных выражениях?
47. Как сделать регулярные выражения нечувствительными к регистру?

Рубежный контроль 3: Список вопросов

1. Что напечатает следующая программа?

```
#include<iostream.h>
main ()
{
  int x;
  x = -3+4*5-6;  cout<<x<<" ";  /* Операция 1 */
  x = 3+4%5-6;   cout<<x<<" ";  /* Операция 2 */
  x = -3*4%-6/5; cout<<x<<" ";  /* Операция 3 */
  x = (7+6)%5/2; cout<<x<<" ";  /* Операция 4 */
}
```

2. Что напечатает следующая программа.

```
#include<iostream.h>
#include<stdio.h>
main()
{
  int a,b,c,v,k;
  cout<<"Задайте целое число: ";
  cin>>v;
  k=v;
  cout<<"        До    Во время    После"<<endl;
  v=k;a=v;b=v++;c=v;printf("v++%8d%8d%8d\n",a,b,c);
  v=k;a=v;b=v--;c=v;printf("v--%8d%8d%8d\n",a,b,c);
  v=k;a=v;b=++v;c=v;printf("++v%8d%8d%8d\n",a,b,c);
  v=k;a=v;b=--v;c=v;printf("--v%8d%8d%8d\n",a,b,c);
}
```

3. Проиллюстрируйте в программе применение логических операций и операций увеличения.

4. Выдать на печать в обратном порядке цифры целого положительного числа N.
5. Напишите программу для замены в слове X всех букв «а» на сочетание «ку».
6. Напишите программу, удваивающую каждую букву слова X.
7. Вычеркните из слова X буквы, стоящие на четных местах.
8. Составить массив, каждый элемент которого равен максимальному из соответствующих значений двух других массивов.

Рубежный контроль 4: Список вопросов

1. Что располагается между служебным словом struct и открывающей фигурной скобкой «{»? В каком случае эта конструкция может отсутствовать?
2. Каков в общем виде шаблон структуры?
3. Способы задания экземпляра структуры.
4. Способы обращения к элементам структур.
5. В чем заключается разница между ссылками и указателями?
6. Перечислить функции, осуществляющие чтение из файла.
7. Перечислить функции, осуществляющие записсь в файл.
8. В чем заключается отличие форматного ввода/вывода от обычного?
9. Организация стандартного ввода/вывода
10. Назначение класса fstream. Его основные методы.
11. Назначение класса ofstream. Его основные методы.
12. Назначение класса ifstream. Его основные методы.
13. Понятие бинарного файла. Примеры работы с ним.

Примерный список вопросов к зачету.

1 семестр

1. Основные типы переменных в Python. Именование переменных. Преобразование типов. Удаление переменной.
2. Операторы. Двоичные операторы. Операторы работы с последовательностями. Приоритет выполнения операторов.
3. Условные конструкции.
4. Циклические конструкции.
5. Операторы break и continue.
6. Строки. Создание строки. Основные операции над строками.
7. Поиск и замена в строке.
8. Регулярные выражения. Синтаксис регулярных выражений.
9. Регулярные выражения. Поиск первого совпадения с шаблоном.
10. Регулярные выражения. Поиск всех совпадений с шаблоном.
11. Регулярные выражения. Замена в строке.
12. Списки. Различные способы задания списков.
13. Основные операции над списками.
14. Добавление и удаление элементов списка.

15. Поиск элемента в списке. Переворачивание и сортировка списка. Преобразование списка в строку.

16. Множества. Создание множества. Основные операции над множествами.

17. Словари. Создание словаря. Основные операции над словарями.

18. Кортежи. Создание кортежа. Основные операции с кортежами.

19. Функции, определяемые пользователем. Особенности определения функций. Локальные и глобальные параметры.

20. Анонимные функции. Рекурсия.

21. Декораторы функций. Анонимные функции.

22. Модули, их назначение. Стандартные модули, их краткая характеристика.

23. Понятие класса и объекта. Создание класса и объекта в Python.

24. Наследование. Реализация наследования.

25. Перегрузка операторов.

26. Обработка исключений. Инструкция try… except. Классы встроенных исключений.

27. Понятие файла. Основные методы работы с файлами.

28. Основные методы работы с каталогами. Права доступа к файлам и каталогам.

29. Библиотека Pillow. Создание нового изображения и загрузка готового изображения.

30. Библиотека Pillow. Рисование линий и фигур.

2 семестр

1. Алфавит языка C++. Лексемы языка C++.
2. Знаки операций в языке C++.
3. Скалярные типы и выражения в языке C++. Примеры использования простейших типов в языке C++.
4. Арифметические операции в языке C++. Примеры.
5. Инкремент и декремент. Примеры.
6. Операции присваивания и отношения. Примеры.
7. Логические операции. Примеры.
8. Условная операция. Примеры.
9. Операция sizeof. Примеры.
10. Операции преобразования типов. Примеры.
11. Оператор if…else. Примеры.
12. Оператор switch. Примеры.
13. Цикл while. Примеры.

14. Цикл do…while. Примеры.
15. Цикл for. Примеры.
16. Оператор безусловного перехода. Примеры.
17. Оператор принудительного выхода из цикла или переключателя. Примеры.
18. Оператор завершения выполнения текущего шага тела цикла. Примеры.

Примерный список вопросов к экзамену.

1. Одномерные и многомерные массивы.
2. Работа с массивами с помощью указателей.
3. Массивы указателей.
4. Массивы динамической памяти.
5. Определение функции, ее сигнатура, возврат значений из функции.
6. Рекурсивные функции, особенности использования
7. Функции с переменным числом аогументов
8. Подставляемые функции.
9. Способы передачи массивов в функции и их возврата. Примеры.
10. Строковая константа. Инициализация строк. Строки и указатели. Примеры.
11. Функции работы со строками.
12. Основные функции работы с файлами в языке С.
13. Основные классы работы с файлами в языке С++.
14. Форматный ввод/вывод в файл.
15. Описание структур.
16. Основные операции над структурами.
17. Использование полей битов в качестве полей структур. Примеры.
18. Понятие объединения.
19. Переменные структуры.
20. Регулируемые и нерегулируемые указатели.
21. Класс string. Его назначение и основные методы.
22. Класс vector. Его назначение и основные методы.
23. Особенности задания и использования массивов в CLR-приложениях.
24. Приложение Windows Forms. Основные файлы проекта.
25. Форма. Ее назначение и основные свойства, методы и события.
26. Компонент Button. Его назначение и основные свойства, методы и события.
27. Компонент TextBox. Его назначение и основные свойства, методы и события.
28. Компонент MenuStrip. Его назначение и основные свойства, методы и события.
29. Компонент ListBox. Его назначение и основные свойства, методы и события.
30. Компонент ComboBox. Его назначение и основные свойства, методы и события.

Примерный список заданий к экзамену.

Пример 1. Действия с числами.

1. Найти несколько простых чисел Фибоначчи.
2. Для заданного целого числа m найти среди первых m*m-1 чисел Фибоначчи хотя бы одно, делящееся на m.
3. Вычислить (N)!!, где (2N)!!=2*4*6*…*2N, (2N+1)!!=1*3*5*..*(2N+1).

Пример 2. Работа с массивами

1. Из заданного множества точек на плоскости выбрать две различные точки так, чтобы количества точек, лежащих по разные стороны прямой, проходящей через эти две точки, различались наименьшим образом.
2. Определить радиус и центр окружности, на которой лежит наибольшее число точек заданного на плоскости множества точек.
3. Определить радиус и центр окружности минимального радиуса, проходящей хотя бы через три различные точки заданного множества точек на плоскости.

Пример 3. Обработка символьной информации.

1. Перечислить все слова заданного предложения, которые состоят из тех же букв, что и первое слово предложения.
2. В заданном предложении найти пару слов, из которых одно является обращением другого.
3. Для каждого из слов заданного предложения указать, сколько раз оно встречается в предложении.

Пример 4. Файлы.

1. Дан файл f, компоненты которого являются целыми числами. Найти: 1) сумму компонент файла, 2) произведение компонент файла, 3) сумму квадратов компонент файла, 4) модуль суммы и квадрат произведения компонент файла, 5) последнюю компоненту файла.
2. Дан символьный файл f. Получить копию файла в файле g.
3. Дан символьный файл f. Получить в файле g компоненты файла f в обратном порядке.

Пример 5. Структуры.

1. Опишите, используя структуру, телефонную книгу.

6.5. Фонд оценочных средств

Полный банк заданий для текущего, рубежных контролей и промежуточной аттестации по дисциплине, показатели, критерии, шкалы оценивания компетенций, методические материалы, определяющие процедуры оценивания образовательных результатов, приведены в учебно-методическом комплексе дисциплины.

7. ОСНОВНАЯ И ДОПОЛНИТЕЛЬНАЯ УЧЕБНАЯ ЛИТЕРАТУРА

7.1. Основная учебная литература

1 Андреева, О. В. Основы алгоритмизации и программирования на языке Python : учебник / О. В. Андреева, О. И. Ремизова. - Москва : Издательский Дом НИТУ «МИСиС», 2021. - 149 с. - ISBN 978-5-907560-22-2. - Текст : электронный. - URL: https://znanium.com/catalog/product/. – Режим доступа: по подписке.

2. Златопольский, Д.М. Основы программирования на языке Python / Д.М. Златопольский. - Москва : ДМК Пресс, 2021. - 284 с. - ISBN 978-5-97060-552-3. - Текст : электронный. - URL: https://znanium.com/catalog/product/1028147. – Режим доступа: по подписке.

3 Медведев А.А. Изучение языка программирования Python [Электронный ресурс]: методические рекомендации для подготовки бакалавров и специалистов направлений 09.03.03, 09.03.04, 10.05.01, 10.05.03 «Прикладная информатика», «Программная инженерия», «Компьютерная безопасность», «Информационная безопасность» / Министерство науки и высшего образования Российской Федерации, Курганский государственный университет, Кафедра программного обеспечения автоматизированных систем; [сост.: А.А. Медведев]. - Электрон. текстовые дан. (тип файла: pdf; размер: 488 Kb). - Курган: Издательство Курганского государственного университета, 2019. - 43, [1] с.: рис. - Библиогр.: с. 43.

4. Гуриков, С. Р. Основы алгоритмизации и программирования на Visual C++ : учебное пособие / С.Р. Гуриков. — Москва : ИНФРА-М, 2021. — 515 с. — (Высшее образование: Бакалавриат). — DOI 10.12737/1039154. - ISBN 978-5-16-015500-5. - Текст : электронный. - URL: https://znanium.com/catalog/product/1039154. – Режим доступа: по подписке.

5 Гуриков, С. Р. Основы алгоритмизации и программирования на Visual C++ : учебное пособие / С.Р. Гуриков. — Москва : ИНФРА-М, 2021. — 515 с. — (Высшее образование: Бакалавриат). — DOI 10.12737/1039154. - ISBN 978-5-16-015500-5. - Текст : электронный. - URL: https://znanium.com/catalog/product/1039154. – Режим доступа: по подписке.

7.2 Дополнительная литература

1. Жуков, Р. А. Язык программирования Python: практикум : учебное пособие / Р.А. Жуков. — Москва : ИНФРА-М, 2021. — 216 с. + Доп. материалы [Электронный ресурс]. — (Высшее образование: Бакалавриат). — DOI 10.12737/textbook_5cb5ca35aaa7f5.89424805. - ISBN 978-5-16-016971-2. - Текст : электронный. - URL: https://znanium.com/catalog/product/1412168. – Режим доступа: по подписке.

2. Русанова, Я. М. C++ как второй язык в обучении приемам и технологиям программирования: учеб. пособие / Я. М. Русанова. - Ростов-на-Дону: Издательство ЮФУ, 2010. - 200 с. - ISBN 978-5-9275-0749-8. - Текст : электронный. - URL: https://znanium.com/catalog/product/550811. – Режим доступа: по подписке.

3. Гридчин, А. В. Информационные технологии. Программирование на C++ : учебно-методическое пособие / А. В. Гридчин. - Новосибирск : Изд-во НГТУ,

2020. - 68 с. - ISBN 978-5-7782-4174-9. - Текст : электронный. - URL: https://znanium.com/catalog/product/1866900. – Режим доступа: по подписке.

4. Ашарина, И. В. Язык С++ и объектно-ориентированное программирование в С++. Лабораторный практикум: Учебное пособие для вузов / Ашарина И.В., Крупская Ж.Ф. - Москва :Гор. линия-Телеком, 2016. - 232 с. - ISBN . - Текст : электронный. - URL: https://new.znanium.com/catalog/product/973780

8. УЧЕБНО-МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ САМОСТОЯТЕЛЬНОЙ РАБОТЫ ОБУЧАЮЩИХСЯ

1. Задачи по программированию / Под ред. Окулов С.М., - 3-е изд. - Москва :Лаборатория знаний, 2017. - 826 с.: ISBN 978-5-00101-448-5. - Текст : электронный. - URL: https://new.znanium.com/catalog/product/541059

2. Кувшинов, Д. Р. Компьютерные науки : Основы программирования: Учебное пособие / Кувшинов Д.Р., - 2-е изд., стер. - Москва :Флинта, Изд-во Урал. ун-та, 2017. - 102 с. ISBN 978-5-9765-3144-4. - Текст : электронный. - URL: https://new.znanium.com/catalog/product/948144

3. Культин, Н. Б. С/С++ в задачах и примерах / Н. Б. Культин. — Санкт-Петербург : БХВ-Петербург, 2015. — 285 с. - ISBN 978-5-9775-3322-5. - Текст : электронный. - URL: https://new.znanium.com/catalog/product/940363

9. РЕСУРСЫ СЕТИ «ИНТЕРНЕТ», НЕОБХОДИМЫЕ ДЛЯ ОСВОЕНИЯ ДИСЦИПЛИНЫ

1. Сайт дистанционного обучения в НОУ (Национальный Открытый Университет) «ИНТУИТ» содержит бесплатные курсы, программы повышения квалификации и профессиональной переподготовки, интересные доклады и другую полезную информацию http://www.intuit.ru.
2. Федеральный портал «Российское образование» http://www.edu.ru/
3. Информационный сайт, содержащий справочные материалы по информатике, которые включают в себя курс лекций, схемы, презентации, рефераты и др. informatikaplus.narod.ru.
4. Постоянно обновляемый электронный учебник (свободный доступ), содержащий полную информацию о языке программирования Python. https://docs.python.org/3/tutorial/index.html
5. Сайт, содержащий необходимые дистрибутивы и полную информацию для языка программирования Python https://www.python.org/
6. Сайт кафедры ПОАС КГУ «Информатика и программирование: шаг за шагом» http://it.kgsu.ru/.

10. ИНФОРМАЦИОННЫЕ ТЕХНОЛОГИИ, ПРОГРАММНОЕ ОБЕСПЕЧЕНИЕ И ИНФОРМАЦИОННЫЕ СПРАВОЧНЫЕ СИСТЕМЫ

При чтении лекций используются слайдовые презентации.

Минимальные требования к программному обеспечению компьютера, используемого при показе слайдовых презентаций: Wondershare PDF Reader (свободно распространяемое программное обеспечение).

11. МАТЕРИАЛЬНО-ТЕХНИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ДИСЦИПЛИНЫ

Материально-техническое обеспечение дисциплины включает в себя учебные лаборатории и классы, оснащенные современными компьютерами (все – в стандартной комплектации для практических занятий и самостоятельной работы), объединенными локальными вычислительными сетями с выходом в Интернет, мультимедийное оборудование (переносной персональный компьютер, мультимедийный проектор, мультимедийный экран).

Программные средства обеспечения учебного процесса должны включать: Python 3.8 или выше (свободно распространяемое программное обеспечение), Visual Studio Code 1.5 или выше (это свободно распространяемое программное обеспечение, которое может использоваться для разработки приложений на языке программирования C++); вспомогательные LibreOffice (программы презентационной графики; текстовые редакторы; графические редакторы – все программы свободно распространяемые).

12. ДЛЯ СТУДЕНТОВ, ОБУЧАЮЩИХСЯ С ИСПОЛЬЗОВАНИЕМ ДИСТАНЦИОННЫХ ОБРАЗОВАТЕЛЬНЫХ ТЕХНОЛОГИЙ

При использовании электронного обучения и дистанционных образовательных технологий (далее ЭО и ДОТ) занятия полностью или частично проводятся в режиме онлайн. Объем дисциплины и распределение нагрузки по видам работ соответствует п. 4.1. Распределение баллов соответствует п. 6.2, либо может быть изменено в соответствии с решением кафедры, в случае перехода на ЭО и ДОТ в процессе обучения. Решение кафедры об используемых технологиях и системе оценивания достижений обучающихся принимается с учетом мнения ведущего преподавателя и доводится до сведения обучающихся.

ПРИЛОЖЕНИЕ 1

Аннотация к рабочей программе дисциплины
«ОСНОВЫ ПРОГРАММИРОВАНИЯ»

образовательной программы высшего образования –
программы бакалавриата

09.03.00 Информатика и вычислительная техника

09.03.03 Прикладная информатика
Направленность:
Интеллектуальные информационные системы и технологии

09.03.04 Программная инженерия
Направленность:
Программное обеспечение автоматизированных систем

Трудоемкость дисциплины: 9 з.е. (324 академических часа)
Семестр: 1, 2 (очная), 2, 3 (заочная)
Форма промежуточной аттестации: зачет, экзамен

Содержание дисциплины

Краткая характеристика языка программирования Python. Условные и циклические конструкции. Работа со строками. Списки, множества, кортежи, диапазоны. Организация функций. Модули. Файлы. Графика.

Краткая характеристика языка программирования C/C++. Условные и циклические конструкции. Массивы. Работа со строками. Функции. Файлы. Создание консольных приложений и приложений Windows Forms. CLR-приложения. Регулируемые или нерегулируемые указатели. Основные компоненты, используемые при создании приложений Windows Forms.