

МИНОБРНАУКИ РОССИИ



Федеральное государственное бюджетное образовательное учреждение
высшего образования

«Российский государственный гуманитарный университет»

(ФГБОУ ВО «РГГУ»)

ИНСТИТУТ ИНФОРМАЦИОННЫХ НАУК И ТЕХНОЛОГИЙ БЕЗОПАСНОСТИ

ФАКУЛЬТЕТ ИНФОРМАЦИОННЫХ СИСТЕМ И БЕЗОПАСНОСТИ

Кафедра информационных технологий и систем

ЯЗЫКИ ПРОГРАММИРОВАНИЯ

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

Направление подготовки 10.03.01 Информационная безопасность

Направленность (профили) подготовки:

Организация и технология защиты информации,

Безопасность автоматизированных систем

Уровень квалификации выпускника – бакалавр

Форма обучения – очная

РПД адаптирована для лиц
с ограниченными возможностями
здоровья и инвалидов

Методы и средства криптографической защиты информации

Рабочая программа дисциплины

Составитель:

Кандидат технических наук, доцент кафедры КЗИ А.С. Моляков

Ответственный редактор

Кандидат технических наук, доцент, зав. кафедрой ИТиС А.А. Роганов

УТВЕРЖДЕНО

Протокол заседания кафедры
кафедры информационных технологий и
систем

№ 10 от 26 апреля 2021г.

ОГЛАВЛЕНИЕ

1. Пояснительная записка

1.1. Цель и задачи дисциплины

1.2. Перечень планируемых результатов обучения по дисциплине (модулю), соотнесённых с индикаторами достижения компетенций

1.3. Место дисциплины в структуре образовательной программы

2. Структура дисциплины

3. Содержание дисциплины

4. Образовательные технологии

5. Оценка планируемых результатов обучения

5.1. Система оценивания

5.2. Критерии выставления оценок

5.3. Оценочные средства (материалы) для текущего контроля успеваемости, промежуточной аттестации обучающихся по дисциплине

6. Учебно-методическое и информационное обеспечение дисциплины

6.1. Список источников и литературы

6.2. Перечень ресурсов информационно-телекоммуникационной сети «Интернет»

7. Материально-техническое обеспечение дисциплины

8. Обеспечение образовательного процесса для лиц с ограниченными возможностями здоровья и инвалидов

9. Методические материалы

9.1. Планы практических занятий

Приложения

Приложение 1. Аннотация дисциплины

Приложение 2. Лист изменений

1. Пояснительная записка

1.1. Цель и задачи дисциплины

Цель дисциплины: приобретение знаний, навыков и умений в области высокоуровневых языков программирования, а также освоение современных алгоритмов анализа больших данных.

Задачи дисциплины: изучение базовых принципов программирования; изучение специализированных технологий и методов программирования на языках C/C++ и Python для анализа и хранения данных; изучение главных управляющих структур языков при использовании функций Win API; приобретение навыков и умений по разработке алгоритмов в задачах анализа данных с использованием библиотек графической и потоковой обработки;

1.2. Формируемые компетенции, соотнесённые с планируемыми результатами обучения по дисциплине

Компетенция (код и наименование)	Индикаторы компетенций (код и наименование)	Результаты обучения
УК-2 Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся ресурсов и ограничений	УК-2.1 Способен применять соответствующий математический аппарат для решения профессиональных задач	Уметь: решать типовые программно-математические задачи защиты информации;
	УК-2.2 Способен использовать знания о важнейших нормах, институтах и отраслях действующего российского права для определения круга задач и оптимальных способов их решения	Владеть: навыками использования положений стандартов при разработке, настройке и оптимизации программных модулей на алгоритмических языках программирования
ОПК-7 Способен использовать языки программирования и технологии разработки программных средств для решения задач профессиональной деятельности	ОПК-7.1 Знает основные принципы построения компьютера, формы и способы представления данных; области и особенности применения языков программирования высокого уровня	Знать: основные принципы и способы представления данных; построения вычислительных блоков компьютерных систем; области и особенности применения языков программирования высокого уровня (C/C++/Python)
	ОПК-7.2 Умеет работать с интегрированной средой разработки программного обеспечения; работать с интегрированной средой разработки программного обеспечения; разрабатывать и реализовывать на языке высокого уровня алгоритмы решения типовых профессиональных задач	Уметь: работать с интегрированной средой разработки программного обеспечения; работать с интегрированной средой разработки и реализовывать алгоритмы на примере MS Visual Studio и Python Shell

	ОПК-7.3 Владеет навыками разработки, документирования, тестирования и отладки программ; разработки алгоритмов решения типовых профессиональных задач	Владеть: навыками разработки, документирования, тестирования и отладки программ; разработки алгоритмов на примере MS Visual Studio и Python Shell
--	---	---

1.3. Место дисциплины в структуре основной образовательной программы

Дисциплина «Языки программирования» относится к обязательной части блока дисциплин учебного плана.

Для освоения дисциплины необходимы компетенции, сформированные в ходе изучения следующих дисциплин: «Теория информации», «Информационные процессы и системы. Вычислительные сети», «Информационные технологии. Автоматизированные системы», «Информационные технологии. Операционные системы».

В результате освоения дисциплины формируются компетенции, необходимые для изучения следующих дисциплин и прохождения практик: «Безопасность операционных систем», «Защита информационных процессов в автоматизированных системах», «Практика по получению профессиональных умений и опыта профессиональной деятельности по технической защите информации, «Преддипломная практика».

2. Структура дисциплины

Структура дисциплины для очной формы обучения

Общая трудоемкость дисциплины – 4 з.е., 152 часа, в том числе контактная работа обучающихся с преподавателем 80 ч., промежуточная аттестация 18 ч., самостоятельная работа обучающихся 54 ч.

№ п/п	Раздел дисциплины/темы	Семестр	Виды учебной работы (в часах)						Формы текущего контроля успеваемости, форма промежуточной аттестации (по семестрам)
			контактная					Самостоятельна я работа	
			Лекции	Семинар	Практические занятия	Лабораторные занятия	Промежуточная аттестация		
1	Введение в дисциплину	1	4					2	Опрос
2	Основы теории алгоритмических языков программирования	1	4		6			4	Оценка выполнения практических заданий
3	Основные принципы и способы представления данных; построения вычислительных блоков компьютерных систем	1	4		8			8	Оценка выполнения практических и внеаудиторных заданий
4	Интегрированные среды разработки программного	1	4		8			8	Оценка выполнения практических

	обеспечения								и внеаудиторных заданий
5	Программирование на C/C++/Phyton. Работа с массивами и строковыми переменными	1	4		8			8	Оценка выполнения практических и внеаудиторных заданий
6	Программирование на C/C++/Phyton. Работа с математическими формулами	1	4		8			8	Оценка выполнения практических заданий
7	Программирование на C/C++/Phyton. Работа с графическими объектами	1	4		8			4	Оценка выполнения практических заданий
8	Базовые навыки сборки и тестирования программных модулей	1	4		2			4	Опрос. Написание контрольного теста
	Экзамен	1					18	8	Экзамен по билетам
	Итого:		32		48		18	54	

3. Содержание дисциплины

№	Наименование раздела дисциплины	Содержание
1	Введение в дисциплину	Общие положения. Базовые термины, понятия и определения. Классическая теория алгоритмов
2	Основы теории алгоритмических языков программирования	Синтаксис языков программирования . Операционная семантика Место компилятора в программном обеспечении. Машина Тьюринга. Принципы фон-Неймановской архитектуры
3	Основные принципы и способы представления данных; построения вычислительных блоков компьютерных систем	Критерии эффективности языков программирования . Простота прочтения и понимания программного языка. Простота реализации программ, то есть насколько удобен в применении язык для конкретной сферы. Критерий надёжности. Имеется в виду способность языка обеспечить минимум ошибок при выполнении программ. Общая себестоимость жизненного цикла языка программирования.
4	Интегрированные среды разработки программного обеспечения	Интегрированная среда разработки как система программных средств, используемая программистами для разработки программного обеспечения . Среда разработки включает в себя: текстовый редактор; компилятор и / или интерпретатор; средства автоматизации сборки; отладчик. Visual Studio 2003/2012/2019. Быстрое написание кода. Phyton shell. Автоматизация работы.

5	Программирование на C/C++/Python. Работа с массивами и строковыми переменными	Массивы и строковые переменные в языках C/C++/Python. Многомерные массивы. Типы операндов. Форматы данных. Оптимизация работы IntelliSense в файлах C++. Локальная разработка с поддержкой множества популярных эмуляторов.
6	Программирование на C/C++/Python. Работа с математическими формулами	Руководство. Определение и использование классов и структур. Упрощенный доступ к тестам в обозревателе решений. Интерфейс Git для создания и клонирования репозитория, управления ветвями
7	Программирование на C/C++/Python. Работа с графическими объектами	Отображение изображений с помощью .NET Framework. Рисование фигур с помощью .NET Framework. Вращение изображений с помощью .NET Framework. Преобразование форматов файлов изображений
8	Базовые навыки сборки и тестирования программных модулей	Компиляция программы на C/C++ включает взятие написанного нами исходного кода (файлы .cpp, .c, .h, .hpp) и преобразование их в исполняемый файл или библиотеку, которая может работать на указанной платформе. Этот процесс можно разделить на три основных этапа: Препроцессинг, Компиляция, Компоновка. Типичный рабочий процесс при создании программы - сборка, а затем отладка.

4. Образовательные технологии

При реализации рабочей программы дисциплины «Языки программирования» используются следующие образовательные технологии:

Образовательные технологии

№ п/п	Наименование темы	Виды учебных занятий	Образовательные технологии
1	2	3	4
1	Введение в дисциплину	Лекция 1.1 Лекция 1.2 Самостоятельная работа	Традиционная с использованием презентаций Подготовка к занятиям с использованием ЭБС
2	Основы теории алгоритмических языков программирования	Лекция 2.1 Лекция 2.2 Практическое занятие 1. Самостоятельная работа	Лекция-дискуссия Занятия с использованием специализированного ПО – MS Visual Studio и Python Shell Подготовка к занятиям с использованием ЭБС

№ п/п	Наименование темы	Виды учебных занятий	Образовательные технологии
3	<i>Основные принципы и способы представления данных; построения вычислительных блоков компьютерных систем</i>	<i>Лекция 3.1 Лекция 3.2 Практическое занятие 2. Самостоятельная работа</i>	<i>Лекция-дискуссия Занятия с использованием специализированного ПО – MS Visual Studio и Phyton Shell Подготовка к занятиям с использованием ЭБС</i>
4	<i>Интегрированные среды разработки программного обеспечения</i>	<i>Лекция 4.1 Лекция 4.2 Практические занятия 3. Самостоятельная работа</i>	<i>Проблемная лекция Традиционная с использованием презентаций Занятия с использованием специализированного ПО – MS Visual Studio и Phyton Shell Подготовка к занятиям с использованием ЭБС</i>
5	<i>Программирование на C/C++. Работа с массивами и строковыми переменными</i>	<i>Лекция 5.1 Лекция 5.2 Практическое занятие 4. Самостоятельная работа</i>	<i>Лекция с разбором конкретных ситуаций Занятия с использованием специализированного ПО – MS Visual Studio и Phyton Shell Подготовка к занятиям с использованием ЭБС</i>
6	<i>Программирование на C/C++. Работа с математическими формулами</i>	<i>Лекция 6.1 Лекция 6.2 Практическое занятие 5. Самостоятельная работа</i>	<i>Лекция-дискуссия Занятия с использованием специализированного ПО – MS Visual Studio и Phyton Shell Подготовка к занятиям с использованием ЭБС</i>
7	<i>Программирование на C/C++. Работа с графическими объектами</i>	<i>Лекция 7.1 Лекция 7.2 Практическое занятие 6. Самостоятельная работа</i>	<i>Лекция-дискуссия Занятия с использованием специализированного ПО – MS Visual Studio и Phyton Shell Подготовка к занятиям с использованием ЭБС</i>
8	<i>Базовые навыки сборки и тестирования программных модулей</i>	<i>Лекция 8.1 Лекция 8.2 Самостоятельная работа</i>	<i>Лекция-дискуссия Подготовка к занятиям с использованием ЭБС</i>

5. Оценка планируемых результатов обучения

5.1. Система оценивания

Форма контроля	Макс. количество баллов	
	За одну работу	Всего
Текущий контроль:		
- опрос (темы 1-8)	5 баллов	30 баллов
- участие в дискуссии на семинаре	5 баллов	10 баллов
- практические задания (темы 2-4)	10 баллов	20 баллов
- практические задания (темы 5-8)	10 баллов	30 баллов
Промежуточная аттестация		40 баллов
Экзамен		
Итого за семестр		100 баллов
Экзамен		

Перечень компетенций с указанием этапов их формирования в процессе освоения дисциплины представляется в виде таблицы:

№ п/п	Контролируемые разделы дисциплины	Код контролируемой компетенции	Наименование оценочного средства
1.	Темы 1 – 8	ОПК-7.1; ОПК-7.2; ОПК-7.3; УК-2.1; УК-2.2	Опрос
2.	Практические занятия 1 – 6	ОПК-7.1; ОПК-7.2; ОПК-7.3; УК-2.1; УК-2.2	План практического занятия

Полученный совокупный результат конвертируется в традиционную шкалу оценок и в шкалу оценок Европейской системы переноса и накопления кредитов (European Credit Transfer System; далее – ECTS) в соответствии с таблицей:

100-балльная шкала	Традиционная шкала		Шкала ECTS
95 – 100	отлично	зачтено	A
83 – 94			B
68 – 82	хорошо		C
56 – 67	удовлетворительно		D
50 – 55			E
20 – 49	неудовлетворительно	не зачтено	FX
0 – 19			F

5.2. Критерии выставления оценки по дисциплине

Баллы/ Шкала ECTS	Оценка по дисциплине	Критерии оценки результатов обучения по дисциплине
100-83/ A,B	«отлично»/ «зачтено (отлично)»/ «зачтено»	<p>Выставляется обучающемуся, если он глубоко и прочно усвоил теоретический и практический материал, может продемонстрировать это на занятиях и в ходе промежуточной аттестации.</p> <p>Обучающийся исчерпывающе и логически стройно излагает учебный материал, умеет увязывать теорию с практикой, справляется с решением задач профессиональной направленности высокого уровня сложности, правильно обосновывает принятые решения.</p> <p>Свободно ориентируется в учебной и профессиональной литературе.</p> <p>Оценка по дисциплине выставляется обучающемуся с учётом результатов текущей и промежуточной аттестации.</p> <p>Компетенции, закреплённые за дисциплиной, сформированы на уровне – «высокий».</p>
82-68/ C	«хорошо»/ «зачтено (хорошо)»/ «зачтено»	<p>Выставляется обучающемуся, если он знает теоретический и практический материал, грамотно и по существу излагает его на занятиях и в ходе промежуточной аттестации, не допуская существенных неточностей.</p> <p>Обучающийся правильно применяет теоретические положения при решении практических задач профессиональной направленности разного уровня сложности, владеет необходимыми для этого навыками и приёмами.</p> <p>Достаточно хорошо ориентируется в учебной и профессиональной литературе.</p> <p>Оценка по дисциплине выставляется обучающемуся с учётом результатов текущей и промежуточной аттестации.</p> <p>Компетенции, закреплённые за дисциплиной, сформированы на уровне – «хороший».</p>
67-50/ D,E	«удовлетворительно»/ «зачтено (удовлетворительно)»/ «зачтено»	<p>Выставляется обучающемуся, если он знает на базовом уровне теоретический и практический материал, допускает отдельные ошибки при его изложении на занятиях и в ходе промежуточной аттестации.</p> <p>Обучающийся испытывает определённые затруднения в применении теоретических положений при решении практических задач профессиональной направленности стандартного уровня сложности, владеет необходимыми для этого базовыми навыками и приёмами.</p> <p>Демонстрирует достаточный уровень знания учебной литературы по дисциплине.</p> <p>Оценка по дисциплине выставляется обучающемуся с учётом результатов текущей и промежуточной</p>

Баллы/ Шкала ECTS	Оценка по дисциплине	Критерии оценки результатов обучения по дисциплине
		аттестации. Компетенции, закреплённые за дисциплиной, сформированы на уровне – «достаточный».
49-0/ F,FX	«неудовлетворительно»/ не зачтено	Выставляется обучающемуся, если он не знает на базовом уровне теоретический и практический материал, допускает грубые ошибки при его изложении на занятиях и в ходе промежуточной аттестации. Обучающийся испытывает серьёзные затруднения в применении теоретических положений при решении практических задач профессиональной направленности стандартного уровня сложности, не владеет необходимыми для этого навыками и приёмами. Демонстрирует фрагментарные знания учебной литературы по дисциплине. Оценка по дисциплине выставляется обучающемуся с учётом результатов текущей и промежуточной аттестации. Компетенции на уровне «достаточный», закреплённые за дисциплиной, не сформированы.

5.3. Оценочные средства (материалы) для текущего контроля успеваемости, промежуточной аттестации обучающихся по дисциплине

Типовые контрольные задания или иные материалы, необходимые для оценки знаний, умений, навыков и (или) опыта деятельности - проверка сформированности компетенций УК-2, ОПК-7

Контрольные вопросы	Реализуемые компетенции
1. Введение . Переменные, константы. Типы данных. Операторы, выражения.	ОПК-7.1; ОПК-7.2; ОПК-7.3; УК-2.1; УК-2.2
2. Ввод-вывод данных.	ОПК-7.1; ОПК-7.2; ОПК-7.3; УК-2.1; УК-2.2
2. Библиотека Scikit-Learn.	ОПК-7.1; ОПК-7.2; ОПК-7.3; УК-2.1; УК-2.2
3. Исключения C/C++/Python.	ОПК-7.1; ОПК-7.2; ОПК-7.3; УК-2.1; УК-2.2
4. Ветвления и циклы в C/C++/Python.	ОПК-7.1; ОПК-7.2; ОПК-7.3; УК-2.1; УК-2.2
5. Строки. Базовые операции. Функции и методы строк.	ОПК-7.1; ОПК-7.2; ОПК-7.3; УК-2.1; УК-2.2
6. Списки. Базовые операции. Функции и методы списков. Генераторы списков. Вложенные списки.	ОПК-7.1; ОПК-7.2; ОПК-7.3; УК-2.1; УК-2.2
7. Кортежи. Базовые операции.	ОПК-7.1; ОПК-7.2; ОПК-7.3; УК-2.1; УК-2.2
8. Множества. Базовые операции. Генераторы множеств.	ОПК-7.1; ОПК-7.2; ОПК-7.3; УК-2.1; УК-2.2
9. Словари. Создание, базовые операции. Методы словарей.	ОПК-7.1; ОПК-7.2; ОПК-

	7.3; УК-2.1; УК-2.2
10. Функции. Передача параметров в функцию. Области видимости переменных.	ОПК-7.1; ОПК-7.2; ОПК-7.3; УК-2.1; УК-2.2
11. Дополнительные возможности при работе с функциями.	ОПК-7.1; ОПК-7.2; ОПК-7.3; УК-2.1; УК-2.2
12. MNIST.	ОПК-7.1; ОПК-7.2; ОПК-7.3; УК-2.1; УК-2.2
13. Показатели производительности алгоритмов.	ОПК-7.1; ОПК-7.2; ОПК-7.3; УК-2.1; УК-2.2
14. Классификация на множестве классов с использованием Python.	ОПК-7.1; ОПК-7.2; ОПК-7.3; УК-2.1; УК-2.2
15. Принципы фон-Неймановской архитектуры.	ОПК-7.1; ОПК-7.2; ОПК-7.3; УК-2.1; УК-2.2
16. Оптимизация на Python: пакетный градиентный спуск, стохастический градиентный спуск.	ОПК-7.1; ОПК-7.2; ОПК-7.3; УК-2.1; УК-2.2
17. Обучение и визуализация дерева принятия решений.	ОПК-7.1; ОПК-7.2; ОПК-7.3; УК-2.1; УК-2.2
18. Алгоритмы обучения . Понятие об энтропии.	ОПК-7.1; ОПК-7.2; ОПК-7.3; УК-2.1; УК-2.2
19. Ансамблевое обучение. Бэггинг и вставка в Scikit-Learn.	ОПК-7.1; ОПК-7.2; ОПК-7.3; УК-2.1; УК-2.2
20. Случайные леса на Python.	ОПК-7.1; ОПК-7.2; ОПК-7.3; УК-2.1; УК-2.2
21. Framework TensorFlow. Создание графа.	ОПК-7.1; ОПК-7.2; ОПК-7.3; УК-2.1; УК-2.2
22. Линейная регрессия с помощью TensorFlow.	ОПК-7.1; ОПК-7.2; ОПК-7.3; УК-2.1; УК-2.2
23. Визуализация графа и кривых обучения с использованием TensorBoard.	ОПК-7.1; ОПК-7.2; ОПК-7.3; УК-2.1; УК-2.2
24. Искусственные нейронные сети.	ОПК-7.1; ОПК-7.2; ОПК-7.3; УК-2.1; УК-2.2
25. Персептрон и его многослойная модификация.	ОПК-7.1; ОПК-7.2; ОПК-7.3; УК-2.1; УК-2.2

Примерные задания для тестирования- проверка сформированности компетенций УК-2,ОПК-7

1. Что такое препроцессор?

- Составная системного блока, предназначенная для обработки данных
- Составная процессора, предназначенной для вычислений с плавающей запятой
- + Составляющая компиляции, которая обрабатывает директивы или команды

2. Что такое заголовочные файлы?

- Название программы, указывается при сохранении
- Название главной функции или функции пользователя
- + Модули, сохраняют заголовки функций
- + Стандартные библиотеки, расположенные в папке include

6. Учебно-методическое и информационное обеспечение дисциплины

6.1. Список источников и литератур

Литература основная

1. Алымова Елена Владимировна. Конечные автоматы и формальные языки: Учебник; ВО - Бакалавриат. - Ростов-на-Дону: Издательство Южного федерального университета (ЮФУ), 2018. - 292 с. - ВО - Бакалавриат. - ISBN 9785927523979. Ссылка на ресурс: <http://new.znaniy.com/go.php?id=1020503>
2. Бильфельд Николай Валентинович. Современные средства реализации автоматизированных систем. Работа с Google таблицами : Учебное пособие. - 1. - Москва ; Москва : Издательский Центр РИОР: ООО "Научно-издательский центр ИНФРА-М", 2018. - 171 с. - ISBN 9785369017210.
1. Трусов А. Н. Редактирование и внесение информации в XML-документы автоматизированных информационных систем[Текст] = Editing and entering information into XML-documents of automated information systems / А. Н. Трусов, П. Ю. Иванченко, Д. А. Кацура // Программные продукты и системы. - 2017. - Т. 30, № 1. - С. 81-84. - Библиогр.: с. 84 (13 назв.). - ил.: 1 рис.
2. Белов А. В. (руководитель). Построение системы атрибутивного поиска в системах управления учетными или идентификационными доменами при внедрении интеграционных решений[Текст] / А. В. Белов, М. И. Нежурина, А. Д. Шестова // Научно-техническая информация. Сер. 2, Информационные процессы и системы. - 2017. - № 11. - С. 5-9. - Примеч. в сносках. - Библиогр.: с. 9 (5 назв.). - Ил.: 1 рис.
3. Колдаев Виктор Дмитриевич. Численные методы и программирование : Учебное пособие. - 1. - Москва; Москва: Издательский Дом "ФОРУМ" : ООО "Научно-издательский центр ИНФРА-М", 2017. - 336 с. - ВО - Бакалавриат. - ISBN 978-5-8199-0333-9.
4. Карпенков Степан Харланович. Современные средства информационных технологий: учебное пособие для студентов высших учебных заведений, обучающихся по направлениям подготовки дипломированных специалистов "Информатика и вычислительная техника" и "Информационные системы" / С. Х. Карпенков. - 2-е изд., испр. и доп. - Москва : КноРус, 2017. - 399, [1] с. : рис., табл. ; 22 см. - Библиогр.: с. 399-400. - ISBN 978-5-390-00393-0 : 318.50.

Дополнительная

1. Ступина А.А. Технология надежного программирования задач автоматизации управления в технических системах: ВО - Магистратура. - 1. - Красноярск: Сибирский федеральный университет, 2011. - 164 с. - ВО - Магистратура. - ISBN 9785763823547. Ссылка на ресурс: <http://new.znaniy.com/go.php?id=442655>

Веселовский А. М. Построение слабосвязанного программного обеспечения для оценивания качества проектных решений[Текст] / А. М. Веселовский // Программные продукты и системы. - 2011. - N 1. - С. 73-76. - Библиогр.: с. 76 (6 назв.). - ил.: 3 рис.

6.2. Перечень ресурсов информационно-телекоммуникационной сети «Интернет», необходимый для освоения дисциплины

Адреса ресурсов Интернет

1. Qt Documentation. [Электронный ресурс]. URL <http://doc.qt.io/>
2. Русское Qt-сообщество. Программирование Qt. [Электронный ресурс]. URL: <http://qt-doc.ru/>

7. Материально-техническое обеспечение дисциплины

- 1) для лекционных занятий – лекционный класс с видеопроектором и компьютером, на котором должны быть установлены следующее ПО:

№ п/п	Наименование ПО	Производитель	Способ распространения
1	Microsoft Office 2010	Microsoft	лицензионное
2	Windows 10 Pro	Microsoft	лицензионное
3	Kaspersky Endpoint Security	Kaspersky	лицензионное

- 2) для практических занятий – компьютерный класс, оборудованный современными персональными компьютерами для каждого студента. На компьютере должны быть установлено следующее ПО:

3)

№п /п	Наименование ПО	Производитель	Способ распространения (лицензионное или свободно распространяемое)
1	Microsoft Office 2010	Microsoft	лицензионное
2	Windows 7 Pro	Microsoft	лицензионное
3	Microsoft Share Point 2010	Microsoft	лицензионное
4	Microsoft Office 2013	Microsoft	лицензионное
5	Windows 10 Pro	Microsoft	лицензионное
6	Kaspersky Endpoint Security	Kaspersky	Лицензионное
7	Vmware Player 15.5	VMWare	Режим доступа: https://www.vmware.com/products/ Демо-версия
8	Phyton	Phyton Software foundation	Режим доступа: https://www.python.org/downloads/ Открытое ПО
9	MS Visual Studio	Microsoft	Режим доступа: https://visualstudio.microsoft.com/ru/downloads/ Демо-версия

Для проведения занятий лекционного типа предлагаются тематические иллюстрации в формате презентаций PowerPoint.

Перечень БД и ИСС

№ п / п	Наименование
1	Международные реферативные наукометрические БД, доступные в рамках национальной подписки в 2020 г. Web of Science Scopus
2	Профессиональные полнотекстовые БД, доступные в рамках национальной подписки в 2020 г. Журналы Cambridge University Press ProQuest Dissertation & Theses Global SAGE Journals Журналы Taylor and Francis
3	Профессиональные полнотекстовые БД JSTOR

	Издания по общественным и гуманитарным наукам Электронная библиотека Grebennikon.ru
4	Компьютерные справочные правовые системы Консультант Плюс, Гарант

8. Обеспечение образовательного процесса для лиц с ограниченными возможностями здоровья

В ходе реализации дисциплины используются следующие дополнительные методы обучения, текущего контроля успеваемости и промежуточной аттестации обучающихся в зависимости от их индивидуальных особенностей:

- для слепых и слабовидящих:
 - лекции оформляются в виде электронного документа, доступного с помощью компьютера со специализированным программным обеспечением;
 - письменные задания выполняются на компьютере со специализированным программным обеспечением, или могут быть заменены устным ответом;
 - обеспечивается индивидуальное равномерное освещение не менее 300 люкс;
 - для выполнения задания при необходимости предоставляется увеличивающее устройство; возможно также использование собственных увеличивающих устройств;
 - письменные задания оформляются увеличенным шрифтом;
 - экзамен и зачёт проводятся в устной форме или выполняются в письменной форме на компьютере.
- для глухих и слабослышащих:
 - лекции оформляются в виде электронного документа, либо предоставляется звукоусиливающая аппаратура индивидуального пользования;
 - письменные задания выполняются на компьютере в письменной форме;
 - экзамен и зачёт проводятся в письменной форме на компьютере; возможно проведение в форме тестирования.
- для лиц с нарушениями опорно-двигательного аппарата:
 - лекции оформляются в виде электронного документа, доступного с помощью компьютера со специализированным программным обеспечением;
 - письменные задания выполняются на компьютере со специализированным программным обеспечением;
 - экзамен и зачёт проводятся в устной форме или выполняются в письменной форме на компьютере.

При необходимости предусматривается увеличение времени для подготовки ответа.

Процедура проведения промежуточной аттестации для обучающихся устанавливается с учётом их индивидуальных психофизических особенностей. Промежуточная аттестация может проводиться в несколько этапов.

При проведении процедуры оценивания результатов обучения предусматривается использование технических средств, необходимых в связи с индивидуальными особенностями обучающихся. Эти средства могут быть предоставлены университетом, или могут использоваться собственные технические средства.

Проведение процедуры оценивания результатов обучения допускается с использованием дистанционных образовательных технологий.

Обеспечивается доступ к информационным и библиографическим ресурсам в сети Интернет для каждого обучающегося в формах, адаптированных к ограничениям их здоровья и восприятия информации:

- для слепых и слабовидящих:
 - в печатной форме увеличенным шрифтом;
 - в форме электронного документа;

- в форме аудиофайла.
- для глухих и слабослышащих:
 - в печатной форме;
 - в форме электронного документа.
- для обучающихся с нарушениями опорно-двигательного аппарата:
 - в печатной форме;
 - в форме электронного документа;
 - в форме аудиофайла.

Учебные аудитории для всех видов контактной и самостоятельной работы, научная библиотека и иные помещения для обучения оснащены специальным оборудованием и учебными местами с техническими средствами обучения:

- для слепых и слабовидящих:
 - устройством для сканирования и чтения с камерой SARA CE;
 - дисплеем Брайля PAC Mate 20;
 - принтером Брайля EmBraille ViewPlus;
- для глухих и слабослышащих:
 - автоматизированным рабочим местом для людей с нарушением слуха и слабослышащих;
 - акустический усилитель и колонки;
- для обучающихся с нарушениями опорно-двигательного аппарата:
 - передвижными, регулируемые эргономическими партами СИ-1;
 - компьютерной техникой со специальным программным обеспечением.

9. Методические материалы

9.1. Планы практических занятий. Методические указания по организации и проведению - *проверка сформированности компетенций УК-2, ОПК-7*

Практическое занятие 1 (6 часов). Первоначальная настройка Microsoft Visual Studio Express - проверка сформированности компетенций УК-2, ОПК-7

Выполнение задания :

1. Чтобы приступить к установке Microsoft Visual Studio Express, надо скачать дистрибутив с сайта Microsoft. Прокрутите страницу и найдите там блок с Express for Desktop.
2. Перед тем как нажать кнопку Загрузка, просмотрите пункты System Requirements (системные требования к вашему компьютеру).
3. Если они не отвечают вашим, например, у вас более старая версия Windows, то найдите в Google Microsoft Visual Studio 2010 Express или Microsoft Visual Studio 2013 Express и выберите ту версию, которая сможет нормально работать на вашем компьютере.
4. Если все требования выполняются – жмите Загрузка
5. После завершения установки настраиваем среду разработки. Студенты создают новый проект, внести в него код и запустить. В открывшейся MVS нажимаем – Создать проект.
6. В этом окне нажмите на Visual C++, Win32, Консольное приложение Win32, введите любое имя для вашего нового проекта и снимите галочку “Создать каталог для решения”. Жмем ОК.
7. Создаем файл с расширением .cpp. В него мы будем писать код программы. Делается это так: в окошке Обозреватель решений нажмите правой кнопкой мыши на имя вашего проекта (у меня это ConsoleApplication2)
8. После написания тестового кода обучающимися нажмите сочетание клавиш Ctrl + Shift + B – начнется компиляция программы. Вы должны увидеть в нижней строке

окна Сборка: успешно: 1, с ошибками: 0 и т.д. После этого нажимаем Ctrl + F5 и видим в открывшемся окне сообщение: “Мы создали первый проект в MVS”

Замечания по выполнению задания:

1. переменным можно присвоить не только определённое значение, а и результат какого-то вычисления: `amount_of_apples1 = amount_of_apples2 + 33;`
2. объявлять переменные желательно в начале main-функции. А при необходимости еще и оставить комментарий, что они будут хранить.
3. регистр букв имени имеет значение. Имена Apple и apple обозначают разные переменные.

Список литературы:

Приведён в п. 6 данной РПД

Материально-техническое обеспечение практического занятия: аудитория, оснащенная презентационной техникой (проектор, экран, компьютер/ноутбук). Компьютеры по количеству обучающихся с развёрнутой ОС MS Windows, виртуальной машиной VMPlayer

Практическое занятие 2 (8 часов). Работа со строками и константами - проверка сформированности компетенций УК-2, ОПК-7

Выполнение задания:

1. Определить константу, которая будет хранить количество дней в неделе: `const int daysInWeek = 7;`
2. Чтобы дать понять компилятору, что это константа, а не обычная переменная, перед типом данных обязательно использовать ключевое слово `const`.
3. Чтобы показать значение переменной на экран, достаточно обратиться к ней по имени: `cout << inBox;` Обратите внимание, как работает `cout`.
4. С помощью оператора `<<`, мы можем чередовать показ текста и показ значения переменной. Так мы делали в строках 17, 19, 20 и 27. В строке 24 в переменную `inBox` записывается сумма переменных: `amount_of_apples1 + amount_of_apples2`. То есть, компилятор, сначала сложит значения этих переменных, а потом запишет сумму в `inBox`.
5. Выводим значение `inBox` на экран – строка 27. Запускаем программу (сначала Ctrl + Shift + B, если ошибок нет то далее Ctrl + F5)
6. Выполнить тестовые задания на C/C++ и Python.
7. Подготовить Отчет.

Замечания по выполнению задания:

- 1 тип данных указывается перед именем переменной и определяет какие данные в ней будут храниться (число, символ...) и сколько памяти необходимо под них выделить.
- 2 имя переменной дает программист, соблюдая определённые правила (указаны выше).
- 3 чтобы создать (объявить) переменную надо указать её тип и дать имя. Переменную желательно сразу инициализировать (присвоить значение при создании): `тип имя = значение;`
- 4 чтобы объявить константу необходимо использовать ключевое слово `const` и обязательно сразу присвоить значение: `const тип имя = значение.`

Список литературы:

Приведён в п. 6 данной РПД

Материально-техническое обеспечение практического занятия: аудитория, оснащенная презентационной техникой (проектор, экран, компьютер/ноутбук). Компьютеры по количеству обучающихся с развёрнутой ОС MS Windows, виртуальной машиной VMPlayer

Практическое занятие 3 (8 часов). Вывод данных на экран и ввод данных с клавиатуры

- проверка сформированности компетенций УК-2, ОПК-7

Выполнение задания:

1. Изучить несколько специальных символьных последовательностей, которые помогут нам манипулировать выводом данных на экран.
2. Вывести на экран цитату из фильма. Сразу пусть прозвучит сигнал, который привлечет внимание пользователя на экран.
3. По центру разместим заголовок и название фильма, из которого цитируем, а ниже – цитату.
4. Организовать ввод данных используя операторы `cin` и `>>`. Синтаксис следующий: `cin >> имя Переменной`.
5. Запустить программу и обратите внимание – программа выполнит команду 11-й строки и, дойдя до оператора `cin >>`, остановится и начнется ожидание действия от пользователя.
6. Надо ввести значение и нажать Enter. Как только переменная получит значение, введенное с клавиатуры, программа продолжит выполнение.
7. Научиться применять унарные операторы, для которых необходим один операнд. Называются они инкремент (`++`) и декремент (`--`). Роль этих операторов в том, чтобы изменить (увеличить или уменьшить соответственно) значение переменной на единицу, при этом значение будет перезаписано.
8. Рассмотреть пример, в котором будем изменять значение переменной `variable` на единицу тремя различными способами, как указано в тестовом примере.
9. После каждой операции в строках 11, 13 и 15. к значению переменной `variable` прибавляется единица. Как вы видите, самая короткая запись – это запись с использованием инкремента. Ниже, в строках 17 – 19, мы трижды применили декремент и в итоге получим значение `variable` уменьшенное на 3.
10. Выполнить тестовые задания на C/C++ и Python.
11. Подготовить Отчет.

Контрольные вопросы:

1. Математические функции стандартной библиотеки Си (`<math.h>`).
2. Форматированный ввод-вывод (`<stdio.h>`): параметры функций `printf()`, `scanf()`.
3. Файловый ввод-вывод (`<stdio.h>`): форматированный и бесформатный.
4. Массивы. Передача массивов в функции.
5. Определение функции. Прототип функции. Рекурсия.

Список литературы:

Приведён в п. 6 данной РПД

Материально-техническое обеспечение практического занятия: аудитория, оснащенная презентационной техникой (проектор, экран, компьютер/ноутбук). Компьютеры по количеству обучающихся с развёрнутой ОС MS Windows, виртуальной машиной VMPlayer

Практическое занятие 4 (8 часов). Генератор случайных чисел - проверка сформированности компетенций УК-2,ОПК-7

Выполнение задания:

1. Если воспользоваться только функцией `rand()` – получите одинаковые “случайные числа” от запуска к запуску.
2. Набрать следующий код и откомпилируйте программу несколько раз. Обратите внимание, что “случайные числа” всегда будут одинаковы.
3. Случайное число генерируется в строке 11 и записывается в *i*-й элемент массива `randomDigits`.
4. Числа генерируются не совсем случайные. Чтобы добиться “настоящей” случайности чисел при повторных запусках программы, необходимо применить функцию `srand()` до функции `rand()`. При этом надо передать ей в виде параметра функцию `time()` с параметром `NULL`: `srand(time(NULL));` (параметр или аргумент функции – это то, что прописывается в круглых скобках после имени функции)
5. `srand()` получает в виде параметра текущее системное время, которое при каждом запуске программы будет разным. Это позволит функции `rand()` каждый раз генерировать именно случайные числа.
6. В первом цикле `for` происходит генерация случайных чисел определённых диапазонов и их запись в соответствующие массивы. В каждом шаге цикла будут генерироваться новые случайные числа. Возможно кому-то сложно разобраться как это происходит. Рассмотрим детально:
7. `rand() % 7` – `rand()` генерирует число и далее вычисляется остаток от деления на 7 от этого числа. Понятно, что это могут быть числа только от 0 до 6. Например генерируется 50 – остаток от деления на 7 будет равен 1, генерируется 49 – остаток от деления на 7 будет равен 0.
8. Выполнить тестовые задания на C/C++ и Python.
9. Подготовить Отчет.

Замечание по выполнению задания:

- 1 `1 + rand() % 7` – очень похоже на предыдущий случай, только 0 мы уже не увидим, а вот 7 появится в диапазоне. Например генерируется 49 – остаток от деления на 7 равен 0 и к нему добавляется единица, генерируется 6 – остаток от деления на 7 равен 6 и опять же добавляется единица.
- 2 `200 + rand() % 101` – даст нам число от 200 до 300. Например генерируется 100 – остаток от деления на 101 равен 100 и добавляется 200. Получаем число 300. Генерируется 202: $200 + (202 \% 101) = 200 + 0 = 200$.
- 3 `rand() % 41 - 20` – от – 20 до 20. Например генерируется 1: $(1 \% 40) - 20 = 1 - 20 = -19$; генерируется 30: $30 - 20 = 10$.
- 4 `0,01 * (rand() % 101)` – от 0.01 до 1. Например генерируется 55: $0.01 * 55 = 0.55$.
- 5 Для использования `time()` необходимо подключить библиотечный файл `ctime` (`time.h` для более старых компиляторов)

Список литературы:

Приведён в п. 6 данной РПД

Материально-техническое обеспечение практического занятия: аудитория, оснащенная презентационной техникой (проектор, экран, компьютер/ноутбук). Компьютеры по количеству обучающихся с развёрнутой ОС MS Windows, виртуальной машиной VMPlayer.

Практическое занятие 5 (8 часов). Работа с таблицами. Одномерные и многомерные массивы - проверка сформированности компетенций УК-2, ОПК-7

Выполнение задания:

1. Массив – это совокупность определенного количества однотипных переменных, имеющих одно имя. Например, `int array [3];`. Эта запись означает, что мы объявили массив с именем `array`, который содержит в себе 3 переменные типа `int`.
2. Переменные массива называют элементами
3. Каждый элемент имеет свой уникальный индекс – свой порядковый номер. Используя индекс, Вы можете обращаться к конкретному элементу.
4. В строке 12 определить целочисленную константу `SIZE`, которая будет хранить размер массива (определённое нами, количество его элементов).
5. В строке 13 объявить массив: указываем тип данных, которые будут храниться в ячейках массива, даем имя и указываем размер в квадратных скобках.
6. попробовать в нашем примере внести любую другую цифру в константу `SIZE`. И вы увидите, что программа будет прекрасно работать – создаст массив на столько элементов, на сколько вы укажете, внесет данные и отобразит их на экране.
7. В строках 15 – 19 определить цикл `for`. Его счетчик `i` будет служить индексом элементов массива. В самом начале, он равен 0 и с каждым шагом будет увеличиваться на единицу до тех пор, пока не станет равным `SIZE` – количеству элементов массива.
8. Изучить многомерные массивы данных на примере с паркингом. Показать пользователю схему паркинга: этажи и места для парковки. Чтобы забронировать место он должен выбрать номер этажа и номер места. После бронирования – записать значение 0 в соответствующую ячейку, что будет означать “место занято”.
9. Выполнить тестовые задания на C/C++ и Python..
10. Оформить Отчет.

Контрольные вопросы:

1. Массивы. Передача массивов в функции.
2. Массивы: одномерные и двумерные.
3. Модульный подход в программировании. Использование *.h файлов. Раздельная компиляция.
4. Оператор `typedef`. Приведение типов.
5. Операторы в выражениях языка Си. Приоритет операторов. Оператор `sizeof()`.
6. Операторы инкремента и декремента.

Список литературы:

Приведён в п. 6 данной РПД

Материально-техническое обеспечение практического занятия: аудитория, оснащенная презентационной техникой (проектор, экран, компьютер/ноутбук). Компьютеры по количеству обучающихся с развёрнутой ОС MS Windows, виртуальной машиной VMPlayer

Практическое занятие 6 (8 часов). Циклы и операторы ветвления – проверка сформированности компетенций УК-2, ОПК-7

Выполнение задания:

1. Рассмотреть операторы, которые применяются ниже для логических операций:

Операторы сравнения (операторы отношения)		
Оператор	как бы задаем вопрос	результат при сравнении значений 3 и 8
> (больше)	значение слева больше чем справа?	cout << (3 > 8); на экране 0 (false)
< (меньше)	значение слева меньше чем справа?	cout << (3 < 8); на экране 1 (true)
>= (больше или равно)	значение слева больше или равно значению справа?	cout << (3 >= 8); на экране 0 (false)
<= (меньше или равно)	значение слева меньше или равно значению справа?	cout << (3 <= 8); на экране 1 (true)
Операторы равенства		
== (равно)	значение слева равно значению справа?	cout << (3 == 8); на экране 0 (false)
!= (не равно)	значение слева не равно значению справа?	cout << (3 != 8); на экране 1 (true)

- Изучить синтаксис операторов if / if else. Понять принцип работы оператора выбора if – если условие в круглых скобках истина (true), то код блока выполнится. Если ложь (false) – игнорируется и выполнение программы начинается со строки кода, следующей под блоком if.
- Разобраться с тем, что означает цикл в программировании. Цикл – это специальный оператор, с помощью которого происходит повторение определённого участка кода определённое количество раз (какой участок кода повторять и когда цикл должен прерваться – определяет программист)
- Не обязательно использовать постфиксный инкремент или декремент. Изменяем управляющую переменную так, как того требует задача. Это может быть ++i, i += 2, i += 20, i -= 15...
- Рассмотреть 4 тестовых примера, варианты которых раздает студентам преподаватель..
- Управляющая переменная i изменяется от 8 до 88 включительно, при этом шаг изменения равен 8. То есть сначала i = 8, на второй итерации 16 и так далее до 88.
- Управляющая переменная i изменяется от 3000 до 300 включительно, с уменьшением при каждой итерации на 300 (3000, 2700, 2400...)
- Управляющая переменная i изменяется от 0 до 100 включительно, с увеличением при каждой итерации на 10. (0, 10, 20...100)
- Управляющая переменная i изменяется от 3000 до 3 включительно, с делением при каждой итерации на 10. (3000, 300, 30, 3).
- Оформить итоговый отчет о проделанной работе.

Контрольные вопросы:

- Блоки и правила видимости переменных.
- Виды операторов присваивания в языке Си.
- Глобальные и внешние переменные.

4. Директива `#define` препроцессора и ее использование. Макроопределения с параметром.
5. Директива `#include` препроцессора и ее использование.
6. Директивы условной компиляции препроцессора и их использование.
7. Использование функций: заголовок, тело и вызов функции.
8. Логические (булевские) операторы и операторы сравнения.

Список литературы:

Приведён в п. 6 данной РПД

Материально-техническое обеспечение практического занятия: аудитория, оснащенная презентационной техникой (проектор, экран, компьютер/ноутбук). Компьютеры по количеству обучающихся с развёрнутой ОС MS Windows, виртуальной машиной VMPlayer

Приложение 1

АННОТАЦИЯ ДИСЦИПЛИНЫ

Дисциплина « Языки программирования» реализуется на факультете Информационных систем и безопасности для студентов 1-го курса, обучающихся по программе бакалавриата по направлению подготовки 10.03.01 Информационная безопасность (профили подготовки – Организация и технология защиты информации и Безопасность автоматизированных систем) кафедрой комплексной защиты информации.

Цель дисциплины: приобретение знаний, навыков и умений в области высокоуровневых языков программирования, а также освоение современных алгоритмов анализа больших данных.

Задачи дисциплины: изучение базовых принципов программирования; изучение специализированных технологий и методов программирования на языках C/C++ и Python для анализа и хранения данных; изучение главных управляющих структур языков при использовании функций Win API; приобретение навыков и умений по разработке алгоритмов в задачах анализа данных с использованием библиотек графической и потоковой обработки;

Дисциплина направлена на формирование следующих компетенций:

- УК-2 - Способен определять круг задач в рамках поставленной цели и выбирать оптимальные способы их решения, исходя из действующих правовых норм, имеющихся ресурсов и ограничений
- УК-2.1 - Способен применять соответствующий математический аппарат для решения профессиональных задач
- УК-2.2 - Способен использовать знания о важнейших нормах, институтах и отраслях действующего российского права для определения круга задач и оптимальных способов их решения
- ОПК-7 -Способен использовать языки программирования и технологии разработки программных средств для решения задач профессиональной деятельности

- ОПК-7.1 -Знает основные принципы построения компьютера, формы и способы представления данных; области и особенности применения языков программирования высокого уровня
- ОПК-7.2 - Умеет работать с интегрированной средой разработки программного обеспечения; работать с интегрированной средой разработки программного обеспечения; разрабатывать и реализовывать на языке высокого уровня алгоритмы решения типовых профессиональных задач
- ОПК-7.3- Владеет навыками разработки, документирования, тестирования и отладки программ; разработки алгоритмов решения типовых профессиональных задач

В результате освоения дисциплины обучающийся должен демонстрировать следующие результаты образования:

Знать: основные принципы и способы представления данных; построения вычислительных блоков компьютерных систем; области и особенности применения языков программирования высокого уровня (C/C++/Phyton)

Уметь: решать типовые программно-математические задачи защиты информации;работать с интегрированной средой разработки программного обеспечения; работать с интегрированной средой разработки и реализовывать алгоритмы на примере MS Visual Studio и Phyton Shell

Владеть: навыками использования положений стандартов при разработке, настройке и оптимизации программных модулей на алгоритмических языках программирования; навыками разработки, документирования, тестирования и отладки программ; разработки алгоритмов на примере MS Visual Studio и Phyton Shell

Рабочей программой предусмотрены следующие виды контроля: текущий контроль успеваемости в форме опроса, контрольной работы,тестирования, промежуточная аттестация в форме экзамена.

Общая трудоемкость освоения дисциплины составляет 4 зачетные единицы.

УТВЕРЖДЕНО
Протокол заседания кафедры
№ _____ от _____

ЛИСТ ИЗМЕНЕНИЙ

в рабочей программе (*модуле*) дисциплины «Языки программирования» по направлению
подготовки 10.03.01 «Информационная безопасность»

на 20__/20__ учебный год

1. В _____ вносятся следующие изменения:

(элемент рабочей программы)

1.1.;

1.2.;

...

1.9.

2. В _____ вносятся следующие изменения:

(элемент рабочей программы)

2.1.;

2.2.;

...

2.9.

3. В _____ вносятся следующие изменения:

(элемент рабочей программы)

3.1.;

3.2.;

...

3.9.

Составитель

подпись

расшифровка подписи

дата