МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ
РОССИЙСКОЙ ФЕДЕРАЦИИ
федеральное государственное автономное образовательное учреждение
высшего образования
«МОСКОВСКИЙ ПОЛИТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ»
Факультет «Информационные технологии»

УТВЕРЖДАЮ
Декан факультета
«Информационные технологии»
_____ / Д.Г.Демидов /
«15» февраля 2024г.

РАБОЧАЯ ПРОГРАММА ДИСЦИПЛИНЫ

«Основы программирования»

Направление подготовки
09.03.03 Прикладная информатика

Профиль
«Информационные технологии управления бизнесом»

Квалификация
Бакалавр

Формы обучения
очная

Москва, 2024 г.

Разработчик(и):
ст.преподаватель /М.Н.Армаш/

Согласовано:
Заведующий кафедрой «Инфокогнитивные технологии»,

доцент, к.т.н. /Е.А.Пухова/

Содержание

1 Цели, задачи и планируемые результаты обучения по дисциплине

Цель освоения дисциплины	Сформировать у студентов базовые знания по теории программирования и технологииразработки программ для ЭВМ. Дать инструмент, позволяющий разрабатывать программное обеспечение для ЭВМ в выбранной предметной области, соответствующей объекту профессиональной деятельности по направлению подготовки и моделировать процессы и компоненты вычислительных систем и сетей.
Задачи дисциплины	Теоретически освоить методологию и основные принципы процедурного и объектно-ориентированного подходов к программированию. • Приобрести навыки разработки программ в интегрированных инструментальныхсредах.

самостоятельная работа над тематикой дисциплины для формирования компетенций основной образовательной программы (далее, ООП).

Обучение по дисциплине «Основы программирования» направлено на формирование у обучающихся следующих компетенций:

Код и наименование компетенций	Индикаторы достижения компетенции
УК-1. Способен осуществлять поиск, критический анализ и синтез информации, применять системный подход для решения поставленных задач	ИУК-1.1. Анализирует задачу, выделяя ее базовые составляющие ИУК-1.2. Осуществляет поиск, критически оценивает, обобщает, систематизирует и ранжирует информацию, требуемую для решения поставленной задачи ИУК-1.3. Рассматривает и предлагает рациональные варианты решения поставленной задачи, используя системный подход, критически оценивает их достоинства и недостатки
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	ИУК-2.1.Формулирует совокупность задач в рамках поставленной цели проекта, решение которых обеспечивает ее достижение ИУК-2.2. Определяет связи между поставленными задачами, основными компонентами проекта и ожидаемыми результатами его реализации ИУК-2.3. Выбирает оптимальные способы планирования, распределения зон ответственности, решения задач, анализа результатов с учетом действующих правовых норм, имеющихся условий, ресурсов и ограничений, возможностей использования
ОПК-7. Способен разрабатывать алгоритмы и программы, пригодные для практического применения	ИОПК-7.1. Знает основные языки программирования, операционные системы и оболочки, современные среды разработки программного обеспечения. ИОПК-7.2. Умеет составлять алгоритмы, писать и отлаживать коды на языке

	программирования, тестировать работоспособность программы, интегрировать программные модули. ИОПК-7.3. Владеет языком программирования, методами отладки и тестирования работоспособности программы

2 Место дисциплины в структуре образовательной программы

Дисциплина «Основы программирования» относится к обязательной части (части, формируемой участниками образовательных отношений) блока Б1 «Дисциплины (модули)».

Блок 1 «Дисциплины (модули)»	
Дисциплины и практики, знания и умения по которым необходимы как "входные" при изучении данной дисциплины	Информатика
Дисциплины, практики, ГИА, для которых изучение данной дисциплины необходимо какпредшествующее	Разработка корпоративных информационных систем Проектирование пользовательских интерфейсов Алгоритмы и структуры данных Основы тестирования Программная инженерия Инженерная коммуникация в области информационных технологий Методы машинного обучения Государственная итоговая аттестация

.

3 Структура и содержание дисциплины

Общая трудоемкость дисциплины составляет __4__ зачетных(е) единиц(ы) (_144_ часов).

3.1 Виды учебной работы и трудоемкость (по формам обучения)

3.1.1 Очная форма обучения

№ п/п	Вид учебной работы	Количество часов	Семестры	
			1	
1	**Аудиторные занятия**	**48**	48	
	В том числе:			
1.1	Лекции	8	8	
1.2	Семинарские/практические занятия	-		
1.3	Лабораторные занятия	40	40	
2	**Самостоятельная работа**	**96**	96	
	В том числе:			
2.1	для формирования «ОПК-7.1»	26		
2.2	для формирования «ОПК-7.2»	35		
2.3	для формирования «ОПК-7.3»	35		

3	**Промежуточная аттестация**			
	Зачет/диф.зачет/экзамен	**экзамен**	экзамен	
	Итого:	**144**	144	

3.1.2 Очно-заочная форма обучения
Не проводится

3.1.3 Заочная форма обучения
Не проводится

3.2 Тематический план изучения дисциплины

3.2.1 Очная форма обучения

№ п/п	**Разделы/темы дисциплины**	**Трудоемкость, час**					
		Всего	**Аудиторная работа**				**Самостоятельная работа**
			Лекции	Семинарские/ практические занятия	Лабораторные занятия	Практическая подготовка	
1	Раздел 1. Базовые конструкции в Python						
1.1	Тема 1.Интерпретируемые и компилируемые языки. Отличительные особенности языка Python. Среды разработки. Исполнение кода и отладка	4	1		1		2
1.2	Тема 2. Библиотека Qt: создание графического интерфейса	12			2		10
1.3	Тема 3. Переменные. Типы данных. Основные операторы. Приоритет и ассоциативность операторов. Целочисленная арифметика	4	1		1		2
1.4	Тема 4. Конструкция ветвление. Условный оператор. Каскадный условный оператор. Вложенные условия	9	1		2		6
1.5	Тема 5. Цикл while, for. Организация циклов. Вычисление суммы ряда. Получение таблицы значений функции. Организация разветвлений в цикле. Цикл в цикле.	9	1		2		6
1.6	Тема 6. Итераторы и генераторы	10			4		6
	Раздел 2. Знакомство с коллекциями	0					
2.1	Тема 1. Строки. Срезы. Методы строк. Типовые алгоритмы обработки строковых данных	9	1		2		6
2.2	Тема 2. Списки. Методы списков. Списочные выражения. Алгоритмы сортировки	8			2		6

2.3	Тема 3. Кортежи. Словари и множества. Хэш-таблицы. Модуль Collections.	10			4		6
2.4	Тема 4. Функции. Области видимости переменных. Возвращение значений из функций. Функции с переменным числом аргументов. Значения по умолчанию. Именованные аргументы.	13	1		4		8
2.4	Тема 5. Лямбда функции. Сортировка с параметром key. Рекурсия.	12			4		8
	Раздел 3. Разработка приложений	0					
3.1	Тема 1. Автоматизированное тестирование в Python. Обработка исключений	8			2		6
3.2	Тема 2. Принципы устройства и механика создания модулей и пакетов	8			2		6
3.3	Тема 3. Работа с файлами и форматированный вывод	9	1		4		4
3.4	Тема 4. ООП: инкапсуляция, наследование, полиморфизм	19	1		4		14
Итого		**144**	**8**		**40**		**96**

3.3 Содержание дисциплины

Раздел 1. Базовые конструкции в Python

Интерпретируемые и компилируемые языки. Отличительные особенности языка Python. Среды разработки. Исполнение кода и отладка.

Основные понятия. Интерпретируемые и компилируемые языки. Отличительные особенности языка python. Интегрированные среды, исполнение кода. Основные понятия программирования: исполнитель, система команд, алгоритм, программа, среда разработки.

Переменные. Типы данных. Основные операторы. Приоритет и ассоциативность операторов. Целочисленная арифметика.

Ключевые слова. Переменные. Идентификаторы. Типы. Оператор присваивания. Инициализация переменных. Локальные переменные. Встроенные числовые типы. Ввод-вывод. Составное форматирование. Операции. Выражения. Операнды и операторы. Унарные операторы. Бинарные операторы. Тернарный оператор. Приоритет и ассоциативность операторов. Арифметические операторы.

Конструкция ветвление. Условный оператор. Каскадный условный оператор. Вложенные условия.

Оператор выбора if (условный оператор). Простейшие программы с использованием условного оператора if и операторов ввода-вывода. Технология разработки программы.

Цикл while, for. Организация циклов. Вычисление суммы ряда. Получение таблицы значений функции.

Оператор итераций (цикл) for. Оператор итераций while. Организация циклов. Организация разветвлений. Разветвления в цикле. Устройство циклов for. Основные управляющие конструкции циклического алгоритма в Python. Простейшие циклы и циклы с переменными.

Раздел 2. Знакомство с коллекциями

Строки. Срезы. Методы строк. Типовые алгоритмы обработки строковых данных.

Работа с символами и строками. Методы строк. Поиск подстроки в строке. Удаление подстроки. Максимальный полиндром. Генерация перестановок.

Функции. Области видимости переменных. Возвращение значений из функций. Функции с переменным числом аргументов. Значения по умолчанию. Именованные аргументы

Функции. Аргументы. Объявление функции. Вызов функции. Области объявления и области видимости переменных Функции с переменным числом аргументов. Значения по умолчанию. Именованные аргументы.

Раздел 3. Решение прикладных задач

Работа с файлами и форматированный вывод

Чтение данных из файла, запись в файл. Оператор with. Способы форматирования строк. Организация форматированного вывода на консоль и в файл

ООП: инкапсуляция, наследование, полиморфизм

Причины появления, принципы и основные сущности объектно-ориентированного подхода к разработке ПО. Инкапсуляция, полиморфизм, наследование, композиция.

3.4 Тематика семинарских/практических и лабораторных занятий

3.4.1 Семинарские/практические занятия

Не запланировано учебным планом

.

3.4.2 Лабораторные занятия

Интерпретируемые и компилируемые языки. Отличительные особенности языка Python. Среды разработки. Исполнение кода и отладка. Знакомство со средой разработки, изучение основных команд, виджетов, упаковщиков и приемов работы.

- ***Библиотека Qt: создание графического.*** Изучить основы создания пользовательских интерфейсов с помощью PyQtDesigner. Изучение приложения QtDesigner.
- ***Переменные. Типы данных. Основные операторы. Приоритет и ассоциативность операторов. Целочисленная арифметика.*** Изучить создание простой программы, с линейной записью математических формул на языке программирования Python.
- ***Конструкция ветвление. Условный оператор. Каскадный условный оператор. Вложенные условия.*** Изучить способы использования условных операторов ветвления на языке программирования Python.
- ***Цикл while, for. Организация циклов.*** Изучить базовые конструкций структурного программирования - операторов цикла.
- ***Итераторы и генераторы.*** Изучить понятие и реализация итераторов и генераторов. Предназначение, особенности устройства и работы, типовые сферы применения.

- Раздел 2. Знакомство с коллекциями
  - ***Строки. Срезы. Методы строк. Типовые алгоритмы обработки строковых данных***. Изучить способы работы со структурированными данными: списки, кортежи, словари.
  - ***Списки. Методы списков. Списочные выражения. Алгоритмы сортировки.*** Изучить оптимальность выбранных методов решения использования списков: изменение элементов массива, поиск максимумов и минимумов, сортировка.
  - ***Кортежи. Словари и множества. Хэш-таблицы. Модуль Collections.*** Изучить оптимальность выбранных методов решения использования кортежей, словарей и множеств: изменение элементов словарей, поиск суммы, произведения, количества.

- ***Функции. Области видимости переменных. Возвращение значений из функций. Функции с переменным числом аргументов. Значения по умолчанию. Именованные аргументы.*** Изучить использование подпрограмм в создании проекта.
- ***Лямбда функции. Сортировка с параметром key***. ***Рекурсия.*** Понятие лямбда функции и области ее применения. Сортировка с параметром key. Рекурсивные функции.

• Раздел 3. Решение прикладных задач

- ***Автоматизированное тестирование в Python. Обработка исключений.*** Изучить наиболее распространенные системы и подходы автоматического тестирования. Понятие исключения, синтаксис их обработки. Применение исключений при разработке и отладке программ.
- ***Принципы устройства и механика создания модулей и пакетов.*** Изучить понятие модуля и пакета, размещение и импорт модуля.
- ***Работа с файлами и форматированный вывод.*** Изучить способы чтение/запись данных в файл. Изучить операции с папками и файлами.
- ***ООП: инкапсуляция, наследование, полиморфизм.*** Изучить методы написании программ с использованием Классов.

3.5. Тематика курсовых проектов (курсовых работ)

Темы курсовых проектов представляется студентам в зависимости от предложений по проектной деятельности или студент может предложить свою тему.

4 Учебно-методическое и информационное обеспечение

4.1 Нормативные документы и ГОСТы

1. Федеральный закон от 29 декабря 2012 года № 273-ФЗ «Об образовании в Российской Федерации» (с изменениями и дополнениями);

2. Федеральный государственный образовательный стандарт высшего образования - бакалавриат по направлению подготовки 09.03.03 Прикладная информатика, утвержденный приказом Министерства образования и науки Российской Федерации от 19.09.2017 № 922.

3. Приказ Министерства образования и науки РФ от 05 апреля 2017 г. № 301 «Об утверждении Порядка организации и осуществления образовательной деятельности по образовательным программам высшего образования – программам бакалавриата, программам специалитета, программам магистратуры;

4. Порядок проведения государственной итоговой аттестации по образовательным программам высшего образования – программам бакалавриата, программам специалитета и программам магистратуры, утвержденный приказом Минобрнауки России от 29 июня 2015 г. № 636;

5. Положение о практической подготовке обучающихся, утвержденное приказом Министерства науки и высшего образования Российской Федерации и Министерства просвещения Российской Федерации от 5 августа 2020 г. № 885/390;

6. Устав и локальные нормативные акты Московского политеха

4.2 Основная литература

1. Чернышев, С. А. Основы программирования на Python : учебное пособие для вузов / С. А. Чернышев. — 2-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2023. — 349 с. — (Высшее образование). — ISBN 978-5-534-17139-6. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://www.urait.ru/bcode/532446.

2. Федоров, Д. Ю. Программирование на языке высокого уровня Python : учебное пособие для вузов / Д. Ю. Федоров. — 5-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2023. — 227 с. — (Высшее образование). — ISBN 978-5-534-17323-9. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://www.urait.ru/bcode/532868

4.3 Дополнительная литература

1. Гниденко, И. Г. Технологии и методы программирования : учебное пособие для вузов / И. Г. Гниденко, Ф. Ф. Павлов, Д. Ю. Федоров. — 2-е изд., перераб. и доп. — Москва : Издательство Юрайт, 2023. — 248 с. — (Высшее образование). — ISBN 978-5-534-18130-2. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://www.urait.ru/bcode/534336 .

2. Тузовский, А. Ф. Объектно-ориентированное программирование : учебное пособие для вузов / А. Ф. Тузовский. — Москва : Издательство Юрайт, 2023. — 213 с. — (Высшее образование). — ISBN 978-5-534-16316-2. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://www.urait.ru/bcode/530800 (дата обращения: 02.12.2023).

4.4 Электронные образовательные ресурсы

1. Курс: Основы программирования Python https://online.mospolytech.ru/course/view.php?id=11744
2. http://window.edu.ru - Информационная система "Единое окно доступа кобразовательным ресурсам"
3. https://openedu.ru - «Национальная платформа открытого образования»(ресурсы открытого доступа)
4. http://www.ispras.ru/programming/
5. http://www.nbmgu.ru/
6. pythonworld.ru. Сайт «Python 3 для начинающих»
7. pythontutor.ru Сайт «Питонтьютор»

4.5 Лицензионное и свободно распространяемое программное обеспечение

1. Visual Studio Code (свободно распространяемое программное обеспечение)
2. PyCharm Community Edition (свободно распространяемое программное обеспечение)
3. QtDesigner (свободно распространяемое программное обеспечение)
4. Google Chrome (свободно распространяемое программное обеспечение)

4.6 Современные профессиональные базы данных и информационные справочные системы

1. https://elibrary.ru - Научная электронная библиотека eLIBRARY.RU (ресурсы открытого доступа)
2. https://www.rsl.ru - Российская Государственная Библиотека (ресурсыоткрытого доступа)
3. https://link.springer.com - Международная реферативная база данных научныхизданий Springerlink (ресурсы открытого доступа)
4. https://zbmath.org - Международная реферативная база данных научныхизданий zbMATH (ресурсы открытого доступа)

5 Материально-техническое обеспечение

Для проведения лабораторных работ и самостоятельной работы студентов подходят аудитории, оснащенные компьютерами с программным обеспечением в соответствии со списком в пункте 4.5 и подключенные к интернету.

Число рабочих мест в аудитории должно быть достаточным для обеспечения индивидуальной работы студентов.

Рабочее место преподавателя должно быть оснащено компьютером с подключенным к нему проектором или иным аналогичным по функциональному назначению оборудованием.

6 Методические рекомендации

6.1 Методические рекомендации для преподавателя по организации обучения

При подготовке к занятиям следует предварительно проработать материал занятия, предусмотрев его подачу точно в отведенное для этого время занятия. Следует подготовить необходимые материалы – теоретические сведения, задачи и др. При проведении занятия следует контролировать подачу материала и решение заданий с учетом учебного времени, отведенного для занятия.

При проверке работ и отчетов следует учитывать не только правильность выполнения заданий, но и оптимальность выбранных методов решения, правильность выполнения всех его шагов.

6.2 Методические указания для обучающихся по освоению дисциплины

Изучение дисциплины осуществляется в строгом соответствии с целевой установкой в тесной взаимосвязи учебным планом. Основой теоретической подготовки студентов являются аудиторные занятия и лекции, материалы лабораторных работ.

В процессе самостоятельной работы студенты закрепляют и углубляют знания, полученные во время аудиторных занятий, дорабатывают конспекты и записи, готовятся к проведению и обрабатывают результаты лабораторных работ, готовятся к промежуточной аттестации, а также самостоятельно изучают отдельные темы учебной программы.

На занятиях студентов, в том числе предполагающих практическую деятельность, осуществляется закрепление полученных, в том числе и в процессе самостоятельной работы, знаний. Особое внимание обращается на развитие умений и навыков установления связи положений теории с профессиональной деятельностью будущего специалиста в области Веб-технологий.

Самостоятельная работа осуществляется индивидуально. Контроль самостоятельной работы организуется в двух формах:

- самоконтроль и самооценка студента;
- контроль со стороны преподавателей (текущий и промежуточный).

Текущий контроль осуществляется на аудиторных занятиях, промежуточный контроль осуществляется в письменной (устной) форме.

Критериями оценки результатов самостоятельной работы студента являются:

- уровень освоения студентом учебного материала;
- умения студента использовать теоретические знания при выполнении лабораторных работ;
- сформированность компетенций;
- оформление материала в соответствии с требованиями.

7 Фонд оценочных средств

7.1 Методы контроля и оценивания результатов обучения

Приведенные ниже правила выставления оценок и опозданий могут быть изменены, если преподаватель сочтет это необходимым. В процессе обучения используются следующие оценочные формы самостоятельной работы студентов, оценочные средства текущего контроля успеваемости и промежуточных аттестаций:

- выполнение и защита лабораторных работ;
- разработка и защита мини проектов.

В соответствии с планом на дисциплины студентам выдаются задания на лабораторные работы. Помимо требований и описания функционала в работе указан крайний срок сдачи. Работа считается сданной если в ней реализовано 80% и более требований и функционала, описанного в задании.

В случае опоздания студент имеет право сдать работу, со штрафным -1 баллом за каждый урок просрочки. Опоздания предназначены для решения особых ситуаций, таких как болезнь или чрезвычайные семейные обстоятельства.

Задания, сданное позже, чем трех уроков просрочки, не будут оцениваться. В связи с зависимостью между работами студентам может потребоваться все равно выполнить предыдущие работы, даже если они не оцениваются.

После сдачи лабораторной работы студент должен ее защитить. Во время защиты лабораторной работы преподаватель проверяет выполнение критериев и требований задания, а студент отвечает на вопросы преподавателя по его коду, а также теоретических вопросов. Если студент отказывается отвечать на вопросы, или дает полностью неверные ответы, или ответы не по теме, то работа может считаться сданной, но при этом она не оценивается.

Работа должна быть выполнена студентом самостоятельно: студент должен объяснить свой код и ход выполнения работы, если эти правила не соблюдаются, то работа не считается сданной и не оценивается.

Рубежный контроль заключается в разработка и защита мини проектов.

Студенты должны заранее сообщать о том, что у них могут возникнуть трудности со своевременной сдачей задания или проекта. При наличии реальных причин задержки студентам следует как можно скорее связаться с преподавателем и обсудить возможные условия.

7.2 Шкала и критерии оценивания результатов обучения

Лабораторная работа оценивается в процентах степени выполнения следующих критериев и для выставления оценки суммируются проценты за каждый из четырех критериев:

1. Полнота выполнения практического задания (1 балла): соответствует ли функциональность заданным требованиям и целям, насколько точно и без ошибок код выполняет поставленные задачи, насколько эффективно задание отвечает требованиям целевой аудитории и обеспечивает приятное восприятие.
2. Качество и структура кода (1 балл): качество, читаемость и организация кода, рациональность выполнения задания, последовательность именования и соблюдение лучших практик.
3. Творчество и инновации (0,5 балла): творческий подход студентов к выполнению заданий, насколько студенты вышли за рамки основных требований и реализовали дополнительные возможности или использовали уникальные решения.
4. Ответы на вопросы по коду студента и теории (2,5 балла):

Дает краткий ответ, содержащий ошибки или неточности. На наводящие вопросы отвечает неправильно (0.5 балла из 2,5 балла)

Дает развернутый ответ, содержащий ошибки или неточности. На наводящие вопросы отвечает неверно (1 из 2,5 балла)

Дает развернутый ответ, содержащий ошибки или неточности. На наводящие вопросы отвечает правильно (1 из 2,5 балла)

Дает правильные и развернутые ответы на вопросы (2.5 балла из 2,5 балла)

Мини проект оценивается по следующим критериям:

Студент может предложить свою тему, но не позднее чем за два недели до сдачи мини проекта. В противном случае тема назначается преподавателем.

Примерные темы мини проектов.

1. Фитнес браслет.
2. Разработать симулятор кадрового агента по приему на работу
3. Калькулятор стоимости путешествия
4. Создание программы для вычисления площади произвольной геометрической фигур.
5. Сервис путешествия с последующей рекомендацией туров, на основе предыдущего выбора.
6. Кривые 2 порядка.
7. Курс по линейная алгебра на Python.
8. Сапер.
9. Проверка слов на верное написание.
10. Тренажер времен английский язык.
11. Редактор сетей Петри.
12. Курс по оптимизации на Python
13. Поиск онлайн школы по подготовке к поступлению
14. Приложение преобразования вещественных значений в разные системы счисления
15. Приложение упрощения логических элементов
16. Игра пять в ряд
17. Клуб игры в бридж
18. Крестики-нолики
19. Курс по классам тестирование знаний на Python
20. Создание интерактивного приложения для выстраивания пользователем последовательностей из однотипных объектов в виде кристаллов.
21. Разработка игрового приложения "Мэмори" со звуком" с сохранением статистики лидеров.
22. Игровая программа "Шахматы".
23. Путешествие подбор авиабилетов
24. Ведение списка дел. Позволяет пользователю добавлять, удалять и отмечать задачи как выполненные
25. Создание интерактивной развивающей игры для детей "Угадай мелодию".
26. График функции (с сохранение и подгрузкой предыдущих результатов)
27. Разработка игрового приложения “Мозаика” с сохранением статистики лидеров.
28. Пользовательская игра “морской бой”
29. Разработка игрового приложения “Домино” с сохранением статистики лидеров.
30. игра судоку
31. Кроссплатформенный текстовый редактор
32. Кроссплатформенный матричный калькулятор.
33. Реализация класса для работы с комплексными числами.
34. WMS для домашнего использования
35. Разработка логической игры "2048” с сохранением статистики лидеров.
36. Разработка игрового приложения “Пазлы” с сохранением статистики лидеров.

37. Тестировщик знаний по истории с занесением результатов тестирования в текстовый документ
38. Прогноз погоды (парсинг сайтов)
39. Игра Математико
40. Математический тренажер (с отслеживанием статистики)

ПОРЯДОК ВЫПОЛНЕНИЯ РАБОТЫ

1. Согласовать с руководителем техническое задание мини проекта, разработанного студентом, после исследования предметной области выбранной темы, не позднее чем за две неделе до сдачи мини проекта.
2. Осуществлять разработку кода программного продукта.
3. Выполнять тестирование информационных ресурсов.
4. Анализ и систематизация полученных результатов.
5. Руководство пользователя.
6. Руководство системного программиста.
7. Загрузить в репозиторий не позднее чем за 3 дня до сдачи – 5 баллов.

КРИТЕРИИ ОЦЕНКИ ВО ВРЕМЯ ЗАЩИТЫ ЗАДАНИЯ

№	Наименование критерия	Балл
•	**Разработка программного обеспечения**	**20**
1.	Реализация основного функционала	15
2.	Дизайн программного обеспечения (креативность)	5
3.	Соответствие руководству по стилю	5
	Документирование	**25**
4.	Анализ и систематизация полученных результатов	5
5.	Руководство пользователя	10
6.	Руководство системного программиста	10

Результат работы оценивается согласно приведенным выше критериям, выполнение каждого из которых увеличивает результирующий баллом на указанное значение. Результатом выполнения лабораторных работ и мини проекта становится
Соответствие баллов в 100 балльной рейтинговой системе оценке по 4-бальной шкале:

Оценка	Критерии оценивания
Неудовлетворительно	от 0 до 54 баллов от суммарных баллов, полученных при защите мини проекта.
Удовлетворительно	от 55 до 69 баллов от суммарных баллов, полученных при защите мини проекта.
Хорошо	от 70 до 84 баллов от суммарных баллов, полученных при защите мини проекта.
Отлично	от 85 до 100 баллов от суммарных баллов, полученных при защите мини проекта.

Форма промежуточной аттестации: экзамен.
Промежуточная аттестация обучающихся в форме экзамена проводится по результатам выполнения всех видов учебной работы, предусмотренных учебным планом по данной дисциплине (модулю), при этом учитываются результаты текущего контроля успеваемости в течение семестра. Оценка степени достижения обучающимися планируемых результатов обучения по дисциплине (модулю) проводится преподавателем, ведущим занятия по дисциплине (модулю) методом экспертной оценки. По итогам промежуточной аттестации по дисциплине (модулю) выставляется оценка.

К промежуточной аттестации допускаются только студенты, выполнившие все виды учебной работы, предусмотренные рабочей программой по дисциплине – выполнение и защита лабораторных работ, в разработка и защита мини проектов согласно полученному заданию с достижением порогового значения оценки в 55 балл.

7.3 Оценочные средства

7.3.1 Текущий контроль

Текущий контроль осуществляется на аудиторных занятиях и промежуточный контроль осуществляется при защите мини проекта (устной) форме.

Примерный список вопросов

1. Алгоритмы. Свойство алгоритмов.
2. Исполнители способы записи алгоритмов.
3. Развитие языков программирования.
4. Парадигмы программирования. Типизация в языках программирования. Области применения языков программирования.
5. Жизненный цикл программы. Программа. Программный продукт и его характеристики.
6. Основные этапы решения задач на компьютере.
7. История языков программирования. Технология структурного программирования.
8. Программирование в Python. Основные характеристики языка. Структура программы.
9. Интегральная средаVC или PyCharm. Компиляторы трансляторы и интерпретаторы.
10. Типы данных. Простые типы данных. Производные типы данных. Структурированные типы данных.
11. Оператор присваивания. Операции и выражения. Правила формирования и вычисления выражений.
12. Виды алгоритмических конструкций.
13. Линейные алгоритмы. Примеры программ. Ввод и вывод данных.
14. Разветвляющие алгоритмы. Условный оператор. Оператор выбора.
15. Циклы с постусловием и с предусловием. Вложенные условные операторы.
16. Цикл с параметром. Вложенные условные операторы.
17. Одномерные массивы. Определение и инициализация числовых массивов.
18. Массивы. Различные алгоритмы сортировок.
19. Двумерные массивы. Определение и инициализация числовых массивов.

20. Символьные массивы в Python. Задание символьных массивов. Инициализация символьных массивов.

21. Структурированный типы данных.

22. Общие сведения о подпрограммах. Определение и вызов подпрограмм.

23. Объявление и определение функций. Формальные и фактические параметры функции.

24. Организация функций. Область видимости и время жизни переменной. Механизм передачи параметров.

25. Рекурсия. Программирование рекурсивных алгоритмов.

26. Исключение. Обработка исключений и RTTI.

27. Модульное программирование. Понятие модуля. Структура модуля. Компиляция и компоновка программы.

28. Модульное программирование. Стандартные модули.

29. Файловый ввод/вывод в языке Python. Базовые функции файловой системы.

30. История развития ООП. Базовые понятия ООП: объект, его свойства и методы, класс, интерфейс.

31. Основные принципы ООП: инкапсуляция, наследования, полиморфизм.

32. Основные понятия объектно-ориентированного программирования.

33. Основные принципы объектно-ориентированного программирования. Классы.

34. Организация методов в Python. Перегрузка методов.

35. Компонентно-ориентированный подход.

36. Классы: основные понятия. Операции класса. Иерархия классов.

37. Коллекции. Списки.

38. Коллекции. Массивы

39. Файлы. Последовательного доступа. Особенности использования.

40. Файлы. Произвольного доступа. Особенности использования.

41. Синтаксис интерфейсов. Интерфейсы и наследование.

42. Основные принципы объектно-ориентированного программирования. Указатели.

43. Классы: основные понятия. Параметризованные классы.

44. Назначение и виды паттернов. Наследование и ассоциации.

45. Событийно-управляемое программирование. Элементы управления. Диалоговые окна. Обработчики событий.

46. Правила разработки интерфейсов пользователя.

47. Разработка функционального интерфейса приложения.

48. Создание интерфейса приложения. Разработка функциональной схемы работы приложения.

7.3.2 Промежуточная аттестация

Оценочные средства для промежуточной аттестации не требуется, так как оценка за промежуточную аттестацию выставляется по балльно-рейтинговой системе, описанной в пункте 7.2.