Автономная некоммерческая организация высшего образования
"Северо-Западный университет"

ПРОГРАММА УЧЕБНОЙ ДИСЦИПЛИНЫ

ОП 06 Основы алгоритмизации и программирования

основной профессиональной образовательной программы
08.02.15 Информационное моделирование в строительстве

Санкт-Петербург
2025

СОДЕРЖАНИЕ

1. ОБЩАЯ ХАРАКТЕРИСТИКА ПРОГРАММЫ УЧЕБНОЙ ДИСЦИПЛИНЫ

«ОП.06 ОСНОВЫ АЛГОРИТМИЗАЦИИ И ПРОГРАММИРОВАНИЯ»

1.1. Место дисциплины в структуре основной образовательной программы:

Учебная дисциплина «**ОП.06 Основы алгоритмизации и программирования**» является обязательной частью общепрофессионального цикла образовательной программы в соответствии с ФГОС СПО по специальности 08.02.15 Информационное моделирование в строительстве.

Особое значение дисциплина имеет при формировании и развитии ОК 01, ОК 02, ОК 09, ПК 1.1., ПК 1.5., ПК 2.2., ПК 2.3., ПК 3.1., ПК 3.2., ПК 3.3.

1.2. Цель и планируемые результаты освоения дисциплины:

В рамках программы дисциплины обучающимися осваиваются умения и знания

Код ПК, ОК	Умения	Знания
ОК 01 ОК 02 ОК 09 ПК 1.1. ПК 1.5. ПК 2.2. ПК 2.3. ПК 3.1. ПК 3.2. ПК 3.3.	– работать в среде программирования; – использовать языки программирования	– типы данных; – базовые конструкции изучаемых языков программирования; – интегрированные среды программирования на изучаемых языках.

2. СТРУКТУРА И СОДЕРЖАНИЕ УЧЕБНОЙ ДИСЦИПЛИНЫ

2.1. Объем учебной дисциплины и виды учебной работы

Вид учебной работы	**Объем в часах**
Объем образовательной программы учебной дисциплины	**51**
в т.ч. в форме практической подготовки	**22**
в т. ч.:	
практические занятия	22
Самостоятельная работа	17
Промежуточная аттестация (дифференцированный зачет) (за счет часов, отведенных на освоение дисциплины)	2

2.2. Тематический план и содержание учебной дисциплины

Наименование разделов и тем	Содержание учебного материала и формы организации деятельности обучающихся	Объем, акад. ч. / в том числе в форме практической подготовки, акад. ч.	Коды компетенций, формированию которых способствует элемент программы
1	*2*	*3*	*4*
Раздел 1. Основные принципы алгоритмизации и программирования			
Тема 1.1 Основные понятия алгоритмизации	**Лекция/ урок**		ОК 01, ОК 02 ОК 09, ПК 1.1. ПК 1.5., ПК 2.2. ПК 2.3., ПК 3.1. ПК 3.2., ПК 3.3.
	Понятие алгоритма и его свойства. Типы алгоритмов. Способы описания алгоритмов. Базовые алгоритмические структуры: линейные, разветвляющиеся, циклические. Основные базовые типы данных и их характеристика. Основы алгебры логики. Логические операции и логические функции.	**2**	
Тема 1.2 Принципы разработки алгоритмов	**Лекция/ урок**		ОК 01, ОК 02 ОК 09, ПК 1.1. ПК 1.5., ПК 2.2. ПК 2.3., ПК 3.1. ПК 3.2., ПК 3.3.
	Принципы построения алгоритмов: использование базовых структур, метод последовательной детализации, сборочный метод. Разработка алгоритмов сложной структуры.		
	Практические занятия		
	Разработка линейных алгоритмов и алгоритмов ветвления. **Разработка циклических алгоритмов.**	2/2	
Тема 1.3 Языки и системы программирования	**Лекция/ урок**		ОК 01, ОК 02 ОК 09, ПК 1.1. ПК 1.5., ПК 2.2. ПК 2.3., ПК 3.1. ПК 3.2., ПК 3.3.
	Классификация языков программирования. Понятие интегрированной среды программирования. Способы классификации систем программирования. Перечень и назначение модулей системы программирования.	**2**	
Тема 1.4 Парадигмы программирования	**Лекция/ урок**		ОК 01, ОК 02 ОК 09, ПК 1.1. ПК 1.5., ПК 2.2. ПК 2.3., ПК 3.1. ПК 3.2., ПК 3.3.
	Этапы разработки программ: системный анализ, алгоритмизация, программирование, отладка, сопровождение. Характеристика и задачи каждого этапа. Принципы структурного программирования: использование базовых структур, декомпозиция базовых структур. Понятия основных элементов		

	ООП: объекты, классы, методы. Свойства ООП: наследование, инкапсуляция, полиморфизм. Принципы модульного программирования.		
Тема 1.5 Принципы отладки и тестового контроля	**Лекция/ урок**		ОК 01, ОК 02 ОК 09, ПК 1.1. ПК 1.5., ПК 2.2. ПК 2.3., ПК 3.1. ПК 3.2., ПК 3.3.
	Понятие отладки. Понятие тестового контроля и набора тестов. Проверка граничных условий, ветвей алгоритма, ошибочных исходных данных. Функциональное и структурное тестирование.	**1**	
Раздел 2. Язык программирования			
Тема 2.1 Характеристика языка	**Лекция/ урок**		ОК 01, ОК 02 ОК 09, ПК 1.1. ПК 1.5., ПК 2.2. ПК 2.3., ПК 3.1. ПК 3.2., ПК 3.3.
	История и особенности языка. Области применения. Характеристика системы программирования. Процесс трансляции и выполнения программы.	**1**	
Тема 2.2 Элементы языка. Простые типы данных	**Лекция/ урок**		ОК 01, ОК 02 ОК 09, ПК 1.1. ПК 1.5., ПК 2.2. ПК 2.3., ПК 3.1. ПК 3.2., ПК 3.3.
	Алфавит и лексика языка. Структура программы. Типы данных языка программирования. Переменные и их описания. Операции с переменными и константами. Правила записи выражений и операций. Организация ввода/вывода данных.		
	Практические занятия		
	Знакомство с инструментальной средой программирования	2/2	
Тема 2.3 Базовые конструкции структурного программирования	**Лекция/ урок**		ОК 01, ОК 02 ОК 09, ПК 1.1. ПК 1.5., ПК 2.2. ПК 2.3., ПК 3.1. ПК 3.2., ПК 3.3.
	Организация ветвлений. Операторы циклов (с предусловием, с постусловием, с параметром). Операторы передачи управления.	**1**	
	Практические занятия		
	Разработка программ разветвляющейся структуры. Разработка программ с использованием цикла с предусловием. Разработка программ с использованием цикла с параметром.	2/2	
Тема 2.4 Работа с массивами и указателями. Структурные типы данных	**Лекция/ урок**		ОК 01, ОК 02 ОК 09, ПК 1.1. ПК 1.5., ПК 2.2. ПК 2.3., ПК 3.1. ПК 3.2., ПК 3.3.
	Одномерные и многомерные массивы, их формирование, сортировка, обработка. Указатели и операции над ними. Работа со строками. Структуры и объединения.	**1**	
	Практические занятия		
	Разработка программ с использованием одномерных массивов и указателей. Сортировка одномерных массивов.	2/2	

	Разработка программ с использованием двумерных массивов. Сортировка двумерных массивов. Разработка программ с использованием структур. Разработка программ с использованием строк.		
Тема 2.5 Процедуры и функции. Работа с файлами	**Лекция/ урок**		ОК 01, ОК 02 ОК 09, ПК 1.1. ПК 1.5., ПК 2.2. ПК 2.3., ПК 3.1. ПК 3.2., ПК 3.3.
	Определение процедур и функций. Области видимости. Глобальные и локальные переменные. Обращение к процедурам и функциям. Использование библиотечных функций. Рекурсивное определение функций. Шаблоны функций. Файловый ввод/вывод. Организация обмена данными между программой и внешними устройствами компьютера. Ввод и вывод текстовой информации. Неформатированный ввод/вывод данных. Дополнительные операции с файлами.	**1**	
	Практические занятия		
	Разработка программ с использованием функций. Разработка программ с использованием рекурсивных функций. Разработка программ работы со структурированными файлами. Разработка программ работы с текстовыми файлами. Разработка программ работы с неструктурированными файлами.	2/2 2/2 2/2	
Раздел 3. Основы объектно-ориентированного программирования			
Тема 3.1 Класс - как механизм создания объектов	**Лекция/ урок**		ОК 01, ОК 02 ОК 09, ПК 1.1. ПК 1.5., ПК 2.2. ПК 2.3., ПК 3.1. ПК 3.2., ПК 3.3.
	Понятия: класс, объект, свойства объекта, методы. Синтаксис объявления класса. Описание объектов. Спецификаторы доступа (private, public, protected). Описание функций-членов класса. Принцип инкапсуляции.	**1**	
	Практические занятия		
	Организация классов и принцип инкапсуляции. Разработка приложений с использованием классов.	2/2 2/2	
Тема 3.2 Принципы наследования и полиморфизма Понятия деструктора и конструктора	**Лекция/ урок**		ОК 01, ОК 02 ОК 09, ПК 1.1. ПК 1.5., ПК 2.2. ПК 2.3., ПК 3.1. ПК 3.2., ПК 3.3.
	Механизм наследования для формирования иерархии классов. Формат объявления класса потомка. Режим доступа. Примеры организации классов-наследников. Назначение и свойства конструкторов, деструкторов. Их описание. Вызов в программе конструкторов, деструкторов. Примеры программ с конструкторами и деструкторами.	**1**	

	Практические занятия		
	Программная реализация принципов наследования. Программная реализация принципов полиморфизма. Разработка конструкторов и деструкторов.	2/2	
Раздел 4. Модульное программирование			
Тема 4.1 Понятие модульного программирования	**Лекция/ урок**		ОК 01, ОК 02 ОК 09, ПК 1.1. ПК 1.5., ПК 2.2. ПК 2.3., ПК 3.1. ПК 3.2., ПК 3.3.
	Модульное программирование как метод разработки программ. Программный модуль и его основные характеристики. Типовая структура программного модуля. Инкапсуляция в модулях. Порядок разработки программного модуля. Связность модулей. Ошибки периода исполнения и логические ошибки в программах. Обработка ошибок. Исключительные ситуации. Организация обработки исключительных ситуаций.	**1**	
Тема 4.2 Разработка приложений	**Лекция/ урок**		ОК 01, ОК 02 ОК 09, ПК 1.1. ПК 1.5., ПК 2.2. ПК 2.3., ПК 3.1. ПК 3.2., ПК 3.3.
	Среда разработки приложений. Архитектура оконных приложений. Конфигурации для создания консольных и оконных приложений. Среда разработки приложений. Архитектура оконных приложений. Конфигурации для создания консольных и оконных приложений. Разработка приложений как многомодульного проекта.		
	Практические занятия		
	Разработка многомодульных приложений.	1/1	
Промежуточная аттестации в форме дифференцированного зачета		2	
Всего:		**34**	

3. УСЛОВИЯ РЕАЛИЗАЦИИ УЧЕБНОЙ ДИСЦИПЛИНЫ

3.1. Для реализации программы учебной дисциплины должны быть предусмотрены следующие специальные помещения:

Для реализации программы учебной дисциплины должны быть предусмотрены следующие специальные помещения:

Наименование помещений для проведения всех видов учебной деятельности, предусмотренной учебным планом, в том числе помещения для самостоятельной работы, с указанием перечня основного оборудования, учебно-наглядных пособий и используемого программного обеспечения	Адрес (местоположение) помещений для проведения всех видов учебной деятельности, предусмотренной учебным планом (в случае реализации образовательной программы в сетевой форме дополнительно указывается наименование организации, с которой заключен договор)
Специализированная многофункциональная учебная аудитория для проведения учебных занятий лекционного типа, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, в том числе, для организации практической подготовки обучающийся, с перечнем основного оборудования (аудитория № 304): Столы для обучающихся; Стулья для обучающихся; Стол педагогического работника; Стул педагогического работника; Компьютер с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду лицензиата; Интерактивная доска; Проектор	191015, г. Санкт-Петербург, Кавалергардская улица, дом 7, литера А (73,9 кв.м.; этаж 3, пом. 9-Н (ч.п. №№ 1-18))
Специализированная многофункциональная учебная аудитория для проведения учебных занятий семинарского типа, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, в том числе, для организации практической подготовки обучающийся, с перечнем основного оборудования (аудитория № 401): Столы для обучающихся; Стулья для обучающихся; Стол педагогического работника; Стул педагогического работника; Компьютеры с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду лицензиата; Интерактивная доска; Проектор Сканер; Принтер	191015, г. Санкт-Петербург, Кавалергардская улица, дом 7, литера А (44,5 кв.м.; этаж 4, пом. 10-Н (ч.п. №№ 1-19))
Специализированная многофункциональная учебная аудитория для проведения учебных занятий семинарского типа, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, в том числе, для организации практической подготовки обучающийся, с перечнем основного оборудования (аудитория № 402): Столы для обучающихся; Стулья для обучающихся; Стол педагогического работника; Стул педагогического работника; Компьютеры с возможностью подключения к сети	191015, г. Санкт-Петербург, Кавалергардская улица, дом 7, литера А (44,1 кв.м.; этаж 4, пом. 10-Н (ч.п. №№ 1-19))

«Интернет» и обеспечением доступа в электронную информационно-образовательную среду лицензиата; Интерактивная доска; Проектор; Сканер; Принтер	
Специализированная многофункциональная учебная аудитория для проведения учебных занятий семинарского типа, групповых и индивидуальных консультаций, текущего контроля и промежуточной аттестации, в том числе, для организации практической подготовки обучающийся, с перечнем основного оборудования (Информационно-аналитическая лаборатория (аудитория № 311)): Столы для обучающихся; Стулья для обучающихся; Стол педагогического работника; Стул педагогического работника; Компьютеры с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду лицензиата Телевизор Устройства вывода звуковой информации: звуковые колонки и наушники	191015, г. Санкт-Петербург, Кавалергардская улица, дом 7, литера А (45,4 кв.м.; этаж 3, пом. 9-Н (ч.п. №№ 1-18))
Помещение для самостоятельной работы обучающихся с перечнем основного оборудования (аудитория № 305): Столы для обучающихся; Стулья для обучающихся; Компьютеры с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду лицензиата; Ноутбуки с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду лицензиата; Принтер; Сканер	191015, г. Санкт-Петербург, Кавалергардская улица, дом 7, литера А (16,2 кв.м.; этаж 3, пом. 9-Н (ч.п. №№ 1-18))
Помещение для самостоятельной работы обучающихся с перечнем основного оборудования (аудитория № 306): Столы для обучающихся; Стулья для обучающихся; Компьютеры с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду лицензиата; Ноутбуки с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду лицензиата; Принтер; Сканер	191015, г. Санкт-Петербург, Кавалергардская улица, дом 7, литера А (15,4 кв.м.; этаж 3, пом. 9-Н (ч.п. №№ 1-18))
Помещение для самостоятельной работы обучающихся с перечнем основного оборудования (аудитория № 307): Столы для обучающихся; Стулья для обучающихся; Компьютеры с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду лицензиата; Ноутбуки с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду лицензиата; Принтер; Сканер	191015, г. Санкт-Петербург, Кавалергардская улица, дом 7, литера А (15,5 кв.м.; этаж 3, пом. 9-Н (ч.п. №№ 1-18))
Помещение для воспитательной работы обучающихся с перечнем основного оборудования (аудитория № 303): Стол педагогического работника;	191015, г. Санкт-Петербург, Кавалергардская улица, дом 7, литера А

Стул педагогического работника; Столы для обучающихся; Стулья для обучающихся; Компьютеры с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду лицензиата; Ноутбуки с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду лицензиата; Принтер; Сканер; Ударная установка; Устройства вывода звуковой информации: звуковые колонки и наушники	(16,2 кв.м.; этаж 3, пом. 9-Н (ч.п. №№ 1-18))
Помещение для воспитательной работы обучающихся с перечнем основного оборудования (аудитория № 403): Стол педагогического работника; Стул педагогического работника; Столы для обучающихся; Стулья для обучающихся; Компьютеры с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду лицензиата; Ноутбуки с возможностью подключения к сети «Интернет» и обеспечением доступа в электронную информационно-образовательную среду лицензиата; Принтер; Сканер; Электрическое фортепиано; Устройства вывода звуковой информации: звуковые колонки и наушники	191015, г. Санкт-Петербург, Кавалергардская улица, дом 7, литера А (16,2 кв.м.; этаж 4, пом. 1--Н (ч.п. №№ 1-19))

3.2. Информационное обеспечение реализации программы

Для реализации программы библиотечный фонд образовательной организации должен иметь печатные и/или электронные образовательные и информационные ресурсы, для использования в образовательном процессе. При формировании библиотечного фонда образовательной организацией выбирается не менее одного издания из перечисленных ниже печатных изданий и (или) электронных изданий в качестве основного, при этом список, может быть дополнен новыми изданиями.

3.2.1. Основные печатные издания

3.2.2. Основные электронные издания

1. Трофимов, В. В. Основы алгоритмизации и программирования : учебник для среднего профессионального образования / В. В. Трофимов, Т. А. Павловская ; под редакцией В. В. Трофимова. — Москва : Издательство Юрайт, 2022. — 137 с. — (Профессиональное образование). — ISBN 978-5-534-07321-8. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/493261 (дата обращения: 01.11.2022).
2. Черпаков, И. В. Основы программирования : учебник и практикум для среднего профессионального образования / И. В. Черпаков. — Москва : Издательство Юрайт, 2022. — 219 с. — (Профессиональное образование). — ISBN 978-5-9916-9984-6. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/491068 (дата обращения: 01.11.2022).

3.2.3. Дополнительные источники

1. Голицына, О. Л. Основы алгоритмизации и программирования : учебное пособие / О.Л. Голицына, И.И. Попов. — 4-е изд., испр. и доп. — Москва : ФОРУМ : ИНФРА-М, 2021. — 431 с. — (Среднее профессиональное образование). - ISBN 978-5-00091-570-7. - Текст : электронный. - URL: https://znanium.com/catalog/product/1150328 (дата обращения: 01.11.2022). – Режим доступа: по подписке.

2. Колдаев, В. Д. Основы алгоритмизации и программирования : учебное пособие / В.Д. Колдаев ; под ред. проф. Л.Г. Гагариной. — Москва : ФОРУМ : ИНФРА-М, 2022. — 414 с. — (Среднее профессиональное образование). - ISBN 978-5-8199-0733-7. - Текст : электронный. - URL: https://znanium.com/catalog/product/1735805 (дата обращения: 01.11.2022). – Режим доступа: по подписке.

3. Трофимов, В. В. Алгоритмизация и программирование : учебник для вузов / В. В. Трофимов, Т. А. Павловская ; под редакцией В. В. Трофимова. — Москва : Издательство Юрайт, 2022. — 137 с. — (Высшее образование). — ISBN 978-5-534-07834-3. — Текст : электронный // Образовательная платформа Юрайт [сайт]. — URL: https://urait.ru/bcode/491215 (дата обращения: 01.11.2022).

4. КОНТРОЛЬ И ОЦЕНКА РЕЗУЛЬТАТОВ ОСВОЕНИЯ УЧЕБНОЙ ДИСЦИПЛИНЫ

Результаты обучения	Критерии оценки	Методы оценки
Перечень знаний, осваиваемых в рамках дисциплины		
Уметь: – работать в среде программирования; – использовать языки программирования	Демонстрирует сформированность элементов общих и профессиональных компетенций при выполнении заданий. Планирует последовательность действий. Самостоятельно выполняет необходимые действия. Осуществляет самоконтроль действий и при необходимости их корректировку	**При текущем контроле успеваемости:** Оценка результатов устного опроса Оценка результатов письменного опроса или заданий в тестовой форме Оценка результатов выполнения работ (заданий) при проведении практических занятий и др. **При промежуточной аттестации:** Дифференцированный зачет
Перечень умений, осваиваемых в рамках дисциплины		
Знать: – типы данных; – базовые конструкции изучаемых языков программирования;	Излагает (перечисляет, называет) существенное содержание вопроса Приводит примеры	**При текущем контроле успеваемости:** Оценка результатов устного опроса

– интегрированные среды программирования на изучаемых языках.	Использует в речи основные понятия, термины Правильность. Самостоятельность Соответствие времени, отведенного на выполнение задания. Проявление активности.	Оценка результатов письменного опроса или заданий в тестовой форме Оценка результатов выполнения работ (заданий) при проведении практических занятий и др. **При промежуточной аттестации:** Дифференцированный зачет

КОМПЛЕКТ ОЦЕНОЧНЫХ МАТЕРИАЛОВ

для проведения текущей и промежуточной аттестации
по учебной дисциплине

ОП.06 Основы алгоритмизации и программирования

основной профессиональной образовательной программы

08.02.15 Информационное моделирование в строительстве

Санкт-Петербург
2025

ОБЩИЕ ПОЛОЖЕНИЯ

Комплект оценочных материалов предназначен для контроля и оценки образовательных достижений обучающихся, освоивших программу учебной дисциплины ОП.06 Основы алгоритмизации и программирования

В результате контроля и оценки по дисциплине осуществляется комплексная проверка следующих общих и профессиональных компетенций: (для дисциплин по ФГОС СПО)

В результате контроля и оценки по дисциплине осуществляется комплексная проверка следующих общих и профессиональных компетенций:

Таблица 1

Общие компетенции	Показатели оценки результата
ОК 01. Выбирать способы решения задач профессиональной деятельности применительно к различным контекстам;	Планирование информационного поиска из широкого набора источников, необходимого для выполнения профессиональных задач;
ОК 02. Использовать современные средства поиска, анализа и интерпретации информации, и информационные технологии для выполнения задач профессиональной деятельности;	Определять задачи для поиска информации; определять необходимые источники информации; планировать процесс поиска; структурировать получаемую информацию; выделять наиболее значимое в перечне информации; оценивать практическую значимость результатов поиска; оформлять результаты поиска
ОК 09. Пользоваться профессиональной документацией на государственном и иностранном языках.	Применение в профессиональной деятельности инструкций на государственном и иностранном языке.

Таблица 2

Профессиональные компетенции	Показатели оценки результата
ПК 1.1 Адаптировать программные средства в соответствии со стандартами применения технологий информационного моделирования зданий.	Использование программные средства в соответствие с применением технологий информационного моделирования
ПК 1.5 Автоматизировать решение задач формирования, анализа и передачи данных о здании средствами программ информационного моделирования.	Выполнять диагностику и расчеты с использованием программ для автоматизированных расчетов;
ПК 2.2 Проектировать строительные конструкции с использование технологии информационного моделирования.	Выполнять проектирование строительных конструкций в соответствие с чертежами, схемами, требованиями нормативных документов и техники безопасности;
ПК 2.3 Проектировать инженерные сети и оборудование с использованием технологии информационного моделирования.	Выполнять проектирование инженерных сетей в соответствие с чертежами, схемами, требованиями нормативных документов и техники безопасности;
ПК 3.1 Формировать данные структурных элементов информационной модели при решении профильных задач на этапе разработки архитектурной,	Умение формировать данные структурны элементов информационной модели при решение прикладных профессиональных задач;

конструктивной частей, инженерных систем и оборудования проекта.	
ПК 3.2 Обрабатывать данные структурных элементов информационной модели при решении профильных задач на этапе разработки архитектурной, конструктивной частей, инженерных систем и оборудования проекта.	Выполнять обработку данных структурных элементов информационной модели, с использованием базовых функций прикладных программ программирования;
ПК 3.3 Актуализировать данные структурных элементов информационной модели при решении профильных задач на этапе разработки архитектурной, конструктивной частей, инженерных систем и оборудования проекта.	Выполнять поиск актуальных данных структурных элементов информационной модели при решение профильных задач;

«уметь – знать»

Уметь:	
У-1	работать в среде программирования
У-2	использовать языки программирования
Знать:	
З-1	типы данных
З-2	базовые конструкции изучаемых языков программирования
З-3	интегрированные среды программирования на изучаемых языках

1.1. Типовые задания для оценки освоения дисциплины.

Задание 1: Тема: «Логический тип данных. Логические выражения, операции отношения и логические операции на языке Python»

Проверяемые результаты обучения: У-1, 2; З-1, 2 Текст задания:

Приступая к решению задач этого раздела, следует вспомнить, что:

- программы с линейной структурой являются простейшими и используются, как правило, для реализации обычных вычислений по формулам;
- в программах с линейной структурой инструкции выполняются последовательно, одна за другой;

- алгоритм программы с линейной структурой может быть представлен следующим образом

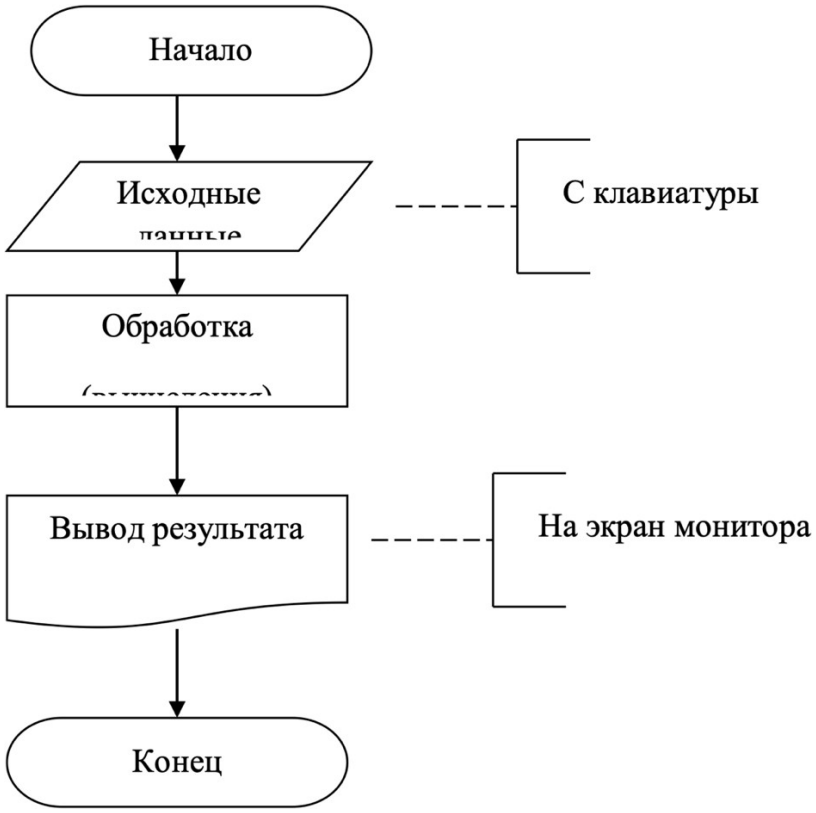
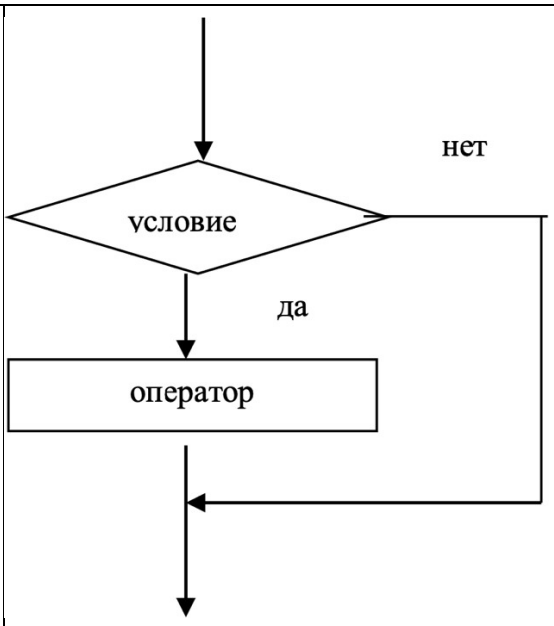
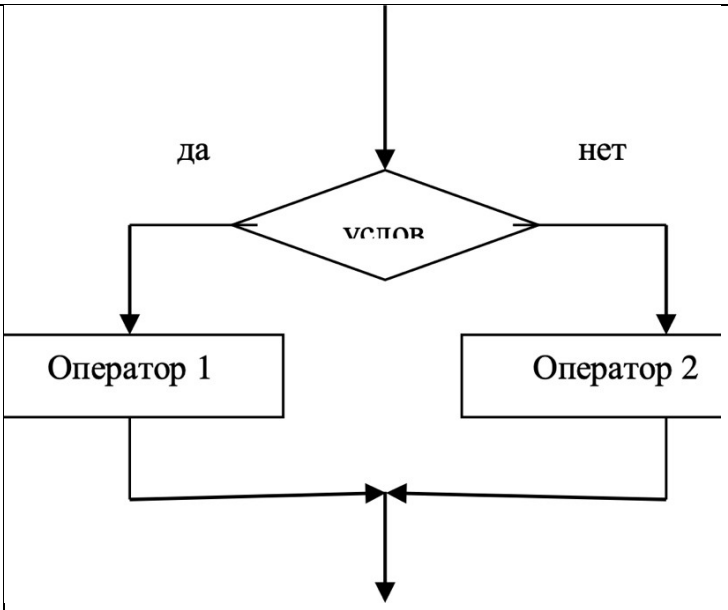


Рис. 1 – Блок-схема линейной конструкции

Ветвление — алгоритм, в котором предусмотрены разветвления, указанные в последовательности действий на два направления в зависимости от итогов проверки заданного условия. То есть такой алгоритм, обязательно содержит условие и в зависимости от результата выполнения условия происходит выбор действия.

Например: Если рабочему дали задание на разработку конструкторской документации, то выполняем это задание, иначе будем заниматься своими делами. Таких примеров можем привести много из обычной жизни и наук. К примеру, физика: **Если** удар упругий, **то** масса тела сохраняется, **иначе** масса изменяется.

Алгоритмическая конструкция ветвление программируется с помощью **условного оператора If**, который может быть представлен двумя вариантами (Таблица 1).Конструкция	Графическое представление (блок-схема)
1 Вариант — неполное ветвление	

If <условие> **Then** <оператор> Неполное ветвление — в зависимости от результата проверки условия либо выполняются действия одной ветви «да» (оператор), либо эти действия не выполняются.	
2 Вариант — полное ветвление	
If <условие> **Then** <оператор 1> **Else** <оператор 2> Полное ветвление — в зависимости от результата проверки условия выполняются только оператор 1 ветви «да» или только оператор 2 ветви «нет»	

Условие — это логическое выражение, которое может принимать одно из двух значений: true (истина — условие выполняется) и false (ложь — условие не выполняется).

В условии используются операции отношения (=, <> , >,<, >=, <=) и логические операции (and (И), or (ИЛИ), xor (исключающее ИЛИ), not (отрицание)). Если требуется проверить несколько условий, их объединяют с помощью логических операций.

Примеры логических выражений:

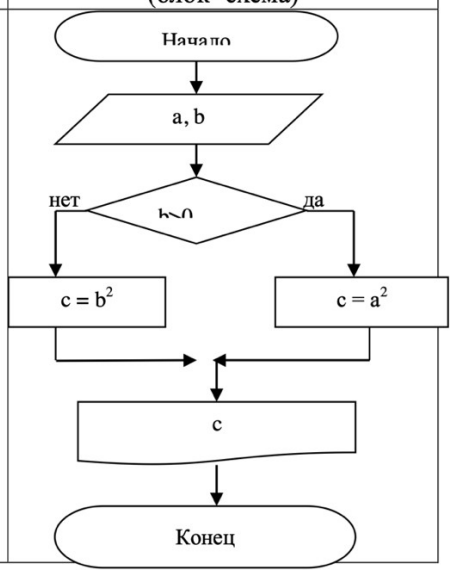
A<2

(x<>0) and (y<>0)

Если между служебными словами стоят несколько операторов, то они заключаются в операторные скобки Begin...End

Рассмотрим пример:

Даны 2 вещественных числа. Если числа положительные, то возвести в квадрат первое число, иначе возвести в квадрат второе число.

Листинг программы	Графическое представление (блок -схема)
Program Primer_1; Uses crt; {Обозначим 1- ое число через переменную a, 2-ое через переменную b, результат — c} Var a, b, c: real; Begin Clrscr; Writeln('Введите первое число '); Readln(a); Writeln('Введите второе число '); Readln(b); **If** (a>0) and (b>0) **Then** c:=sqr(a) **Else** c:=sqr(b); Writeln('Результат = ', c:5:2); End.	

Задание:

1. Написать программу вычисления объема цилиндра. Ниже представлен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

Вычисление объема цилиндра Введите

исходные данные:

Радиус основания (см) —> 5 Высота

цилиндра (см) —> 10 Объем цилиндра

1570.80 куб. см.

Для завершения работы программы нажмите <Enter>.

2. Написать программу вычисления объема цилиндра. Ниже представлен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

Вычисление объема цилиндра

Введите исходные данные: Радиус основания (см) —> 5

Высота цилиндра (см) —> 10 Объем

цилиндра 1570.80 куб. см.

Для завершения работы программы нажмите <Enter>.

3. Написать программу вычисления двух выражений:

$$z_1 = \frac{\sqrt{2b + 2\sqrt{b^2 - 4}}}{\sqrt{b^2 - 4 + b + 2}}$$

$$z_2 = \sqrt{\frac{a + b}{a - 3}}$$

4. Для каждой программы составить блок-схему

5. Выполните задание по варианту, назначенному преподавателем.

Вариант 1

Задание 1

Даны три действительные числа. Возвести в квадрат те из них, значения которых положительны, и в четвертую степень — отрицательные.

Задание 2

Написать программу, которая вычисляет частное от деления двух чисел. Программа должна проверять правильность введенных пользователем данных и, если они неверные (делитель равен нулю), выдавать сообщение об ошибке. Ниже представлен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

Вычисление частного.
Введите в одной строке делимое и делитель,затем нажмите <Enter> -> ***12 О***
Вы ошиблись. Делитель не должен быть равен нулю.

Вариант 2

Задание 1

Даны два действительные числа. Если числа положительны найти их сумму, если отрицательны — произведение

Задание 2

Написать программу вычисления стоимости покупки с учетом скидки. Скидка в 10% предоставляется, если сумма покупки больше 1000 руб. Ниже представлен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

Вычисление стоимости покупки с учетом скидки. Введите сумму покупки и нажмите <Enter>

*-> **1200***

Вам предоставляется скидка 10%

Сумма покупки с учетом скидки: 1080.00 руб.

Задание 3

Оформить отчет. Отчет должен содержать коды программ и блок-схемы

Вариант 3

Задание 1

Даны действительные числа x и y, не равные друг другу. Меньшее из этих чисел заменить половиной их суммы, а большее — их удвоенным произведением

Задание 2
Написать программу проверки знания даты основания Санкт-Петербурга. В случае неверного ответа пользователя программа должна выводить правильный ответ. Ниже представлен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).
В каком году был основан Санкт-Петербург? Введите число и

нажмите <Enter>

*-> **1705***

Вы ошиблись, Санкт-Петербург был основан в 1703 году.

Задание 3

Оформить отчет. Отчет должен содержать коды программ и блок-схемы

Вариант 4

Задание 1

Данные два вещественных числа. Если первое число больше второго, то возвести его в третью степень, если равно второму — прибавить к нему второе число

Задание 2

Написать программу определения стоимости разговора по телефону с учетом скидки 20%, предоставляемой по воскресеньям. Ниже представлен рекомендуемый вид экрана программы во время ее работы (данные, введенные пользователем, выделены полужирным шрифтом).

Вычисление стоимости разговора по телефону. Введите

исходные данные:

*Длительность разговора (целое количество минут) —> **3** День недели (1 - понедельник, ... 7 — воскресенье) —> **6** Предоставляется скидка 20%. Стоимость*

разговора: 5.52 руб.

Задание 3

Оформить отчет. Отчет должен содержать коды программ и блок-схемы

Вариант 5

Задание 1

Даны три действительные числа. Если первое число больше второго, умножить данное число на 5, если первое число больше третьего — разделить на два.

Задание 2

Написать программу— модель анализа пожарного датчика в помещении, которая выводит сообщение «Пожароопасная ситуация», если температура в комнате превысила 60 ºС

Задание 3

Оформить отчет. Отчет должен содержать коды программ и блок-схемы

Вариант 6

Задание 1

Даны действительные числа a, b, c. Удвоить эти числа, если a □ b □ c , иначе оставить без изменения.

Задание 2

Написать программу, которая анализирует данные о возрасте и относит человека к одной из четырех групп: дошкольник, ученик, работник, пенсионер. Возраст вводится с клавиатуры.

Задание 3

Оформить отчет. Отчет должен содержать коды программ и блок-схемы

Вариант 7

Задание 1

Даны три числа a, b, c. Определить какое из них равно d. Если ни одно не равно d, то найти сумму чисел a, b, c.

Задание 2

Написать программу вычисления стоимости покупки с учетом скидки. Скидка в 3% предоставляется в том случае, если сумма покупки больше 500 руб., в 5% — если сумма больше 1000 руб. Ниже представлен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

Вычисление стоимости покупки с учетом скидки. Введите сумму покупки и нажмите <Enter>
*-> **640***
Вам предоставляется скидка 3%
Сумма покупки с учетом скидки: 620.80 руб.

Задание 3

Оформить отчет. Отчет должен содержать коды программ и блок-схемы

Вариант 8

Задание 1

Даны три действительные числа. Найти минимальное и максимальное число. Задание 2
Написать программу проверки знания истории архитектуры. Программа должна вывести вопрос и три варианта ответа. Пользователь должен выбрать правильный ответ и ввести его номер. Ниже представлен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).
Архитектор Исаакиевского собора: 1. Доменико Трезини
2. Огюст Монферран
3. Карл Росси
Введите номер правильного ответа и нажмите <Enter>
*-> **3***
Вы ошиблись.
Архитектор Исаакиевского собора — Огюст Монферран.
Задание 3
Оформить отчет. Отчет должен содержать коды программ и блок-схемы

Вариант 9

Задание 1

Даны три действительные числа. Если все числа положительны, найти среднее арифметическое, иначе произведение.

Задание 2

Написать программу проверки знания истории архитектуры. Программа должна вывести вопрос и три варианта ответа, а пользователь — выбрать правильный ответ и ввести его номер. Ниже представлен рекомендуемый вид экрана во время работы программы (данные, введенные пользователем, выделены полужирным шрифтом).

Невский проспект получил свое название:

1. По имени реки, на берегах которой расположен Санкт-Петербург

2. По имени близко расположенного монастыря Александро-Невской лавры 3. В память о знаменитом полководце Александре Невском

Введите номер правильного ответа и нажмите <Enter>

*-> **3***

Вы ошиблись. Правильный ответ:

2.

Задание 3

Оформить отчет. Отчет должен содержать коды программ и блок-схемы

Вариант 10

Задание 1

Даны два вещественных числа, если числа не равны нулю, возвести из в третью степень, иначе во вторую степень.

Задание 2

Написать программу, которая сравнивает два числа, введенных с клавиатуры. Программа должна указать, какое число больше, или, если числа равны, вывести соответствующее сообщение. Ниже представлен рекомендуемый вид экрана во время работы программы.

Введите в одной строке два целых числа

*-> **34 67***

34 меньше 67.

Задание 3

Оформить отчет. Отчет должен содержать коды программ и блок-схемы

Контрольные вопросы

1. Какие типы данные вы знаете?
2. Что такое алгоритм?
3. Приведите пример алгоритма из реальной жизни.
4. Какими свойствами обладает алгоритм?
5. Что такое линейная конструкция?
6. Какие операторы используются для реализации линейной конструкции в программе?
7. Назовите процедуры ввода/вывода данных.
8. Что такое формат вывода данных?
9. Перечислите основные разделы программы.

Критерии оценки:

Оценка «**отлично**» выставляется, если все задания выполнено полностью самостоятельно и полностью соответствует поставленной задаче. Студент изобразил блок-схему выполнения алгоритма, программа написано верно, студент ответил на все контрольные вопросы.

Оценка «**хорошо**» выставляется, если задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче, но при этом допущены несущественные неточности, устраненные без помощи преподавателя. Студент изобразил блок-схему выполнения алгоритма, программа написано верно, студент ответил на все контрольные вопросы.

Оценка «**удовлетворительно**» выставляется, если задание выполнено не в полном объеме или не полностью соответствует поставленной задаче, при этом могут быть допущены несущественные неточности, устраненные с помощью преподавателя. Студент изобразил блок-схему выполнения алгоритма, но не совсем верную, программа написано с частичными ошибками, студент ответил частично на 5 контрольных вопроса.

Оценка «**неудовлетворительно**» выставляется, если задание не выполнено и полностью не соответствует поставленной задаче, допущены существенные неточности, которые обучающийся не может устранить.

Задание 2: Тема: «Создание презентации. Графические объекты, текст, таблицы. Диаграммы как элементы презентаций. Выбор дизайна, эффекты, анимации. Настройка показа (MS Power Point)»

Проверяемые результаты обучения: У-1, 2, 3, 4, 5; З – 1, 2, 3, 4, 5 Текст задания:

Операции сравнения в Python

Ниже приведена таблица, в которой указаны основные операции сравнения, которые можно применять на языке Python (см. табл. 1).

Таблица 1. Операции сравнения в языке Python

Операция сравнен ия	**Функция**
==	Возвращает **True**, если оба операнда равны. Иначе возвращает **Fa lse**
!=	Возвращает **True**, если оба операнда НЕ равны. Иначе возвращае т **False**
>	Возвращает **True**, если первый операнд больше второго
<	Возвращает **True**, если первый операнд меньше второго
>=	Возвращает **True**, если первый операнд больше или равен втором у
<=	Возвращает **True**, если первый операнд меньше или равен втором у

Рассмотрим примеры использования каждого оператора сравнения (рис. 1, рис. 2, рис. 3, рис. 4, рис. 5, рис. 6):

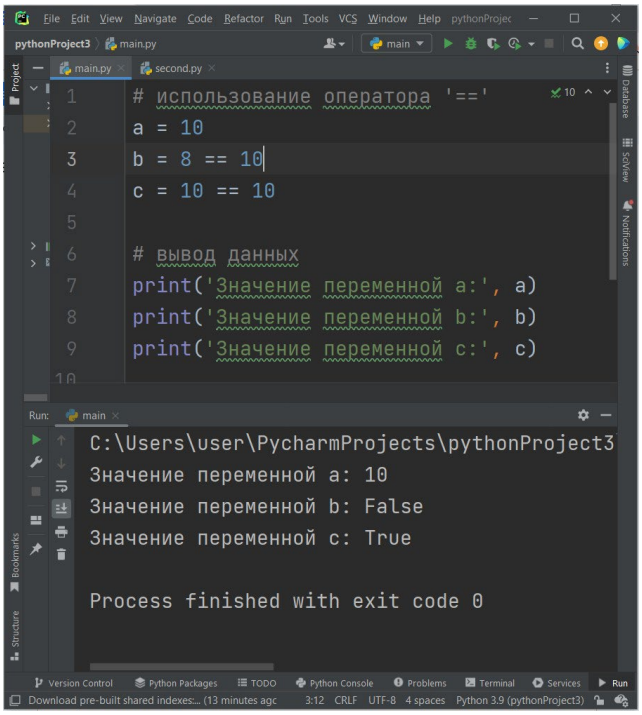


Рисунок 1 - Пример работы программы с использованием оператора сравнения ‘==’

Произведем **анализ работы** данной программы: в первой строке указан однострочный комментарий. В строке 2 вводится переменная с именем **‘a’**, в которую записывается значе ние**10**. В строке 3 вводится переменная с именем **‘b’**, в которую записывается значение **8**, после чего применяется оператор **‘==’**, справа от которого пишется значение **10**. Значение данной переменной, в таком случае, будет **False**. Если значение после знака ‘=’ будет точн о таким же, как и значение после оператора присваивания ‘==’, то значение переменнойбу дет **True** (как можно заметить при выводе переменной с именем **‘c’**). В строке 4 вводится п еременная с именем **‘c’**.

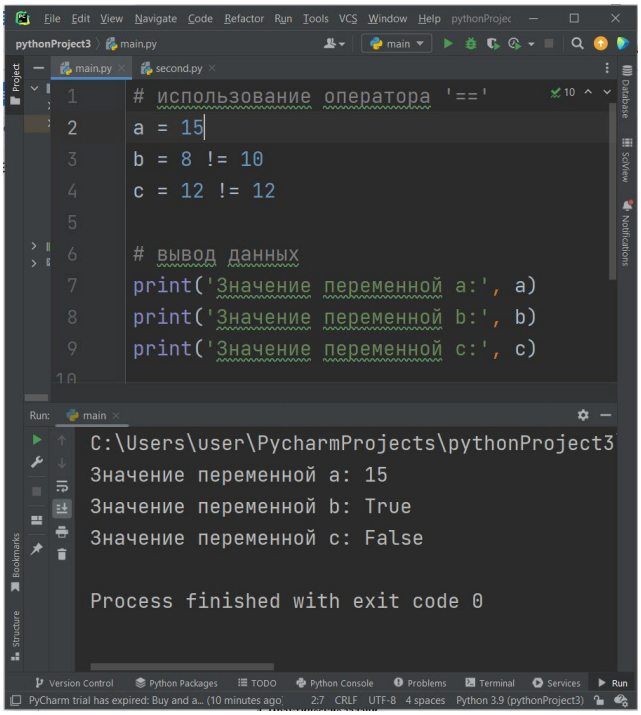


Рисунок 2 - Пример работы программы с использованием оператора сравнения ‘!=’

В последнем примере наоборот - если значение после знака ‘=’ будет точно таким же, к ак и значение после оператора присваивания ‘!=’, то значение переменной будет **False** (как можно заметить при выводе переменной с именем **‘c’**). Принцип работы такой же, у нас ес ть три переменных со значениями **15**, **8** и **12**.

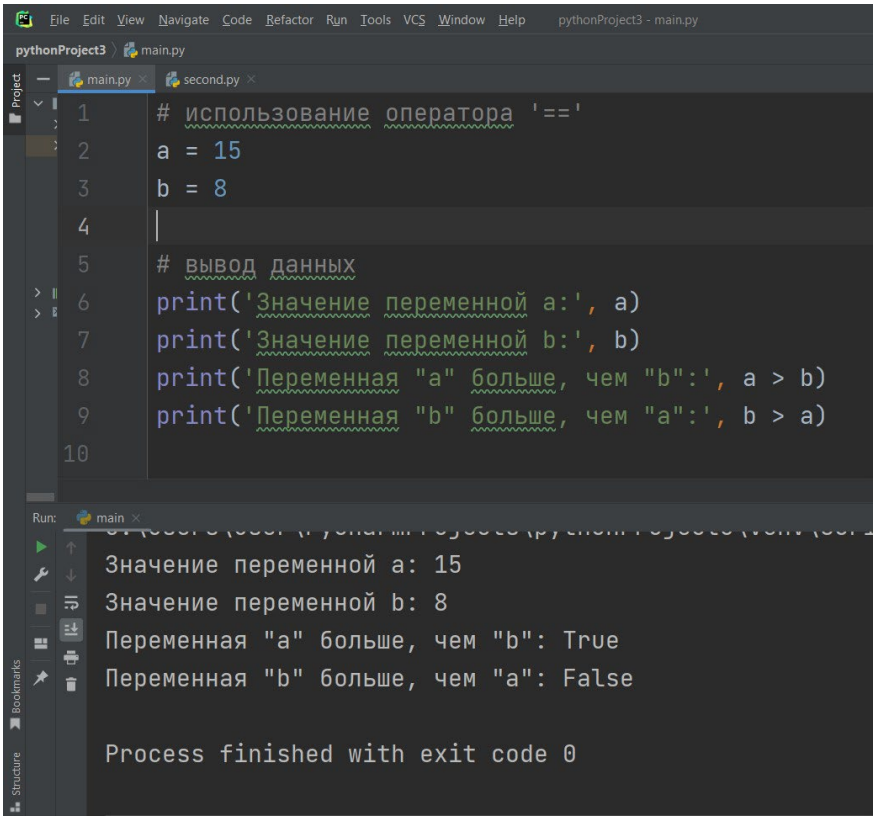


Рисунок 3 - Пример работы программы с использованием оператора сравнения ‘>’

В данном примере используются две переменные: **‘a’** со значением **15** и **‘b’** со значение м **8**. В строках 6 и 7 выводятся значения переменных **‘a’** и **‘b’**. В строке 8 мы задаем вопро сконсоли - значение переменной **‘a’** больше, чем значение переменной **‘b’**. Это правда, поэ тому в выводе данной строки можно заметить значение **True**. Обратный вопрос задается в строке 9, результатом является, соответственно, значение **False**.

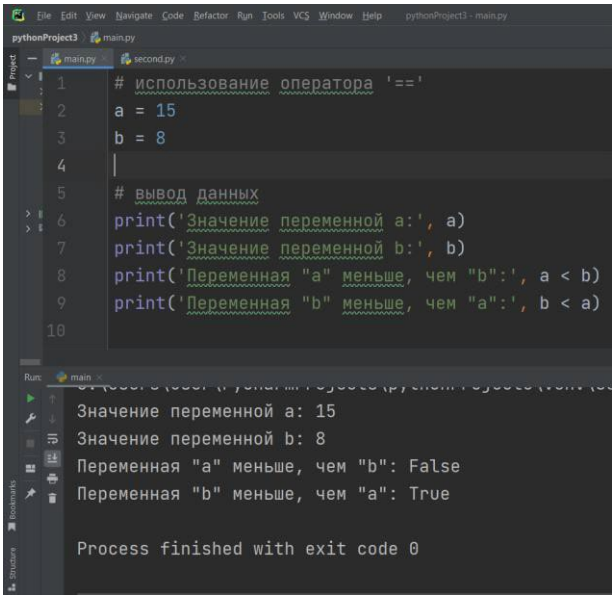


Рисунок 4 - Пример работы программы с использованием оператора сравнения ‘<’

В данном примере обратная ситуация: в строках 8 и 9 применяется оператор сравнения **‘<’ (меньше)**. Соответственно, когда результат операции является правдой (истиной), как встроке 9, в консоли можно заметить значение **True**. Соответственно, в строке 8 можно за метить значение **False**.

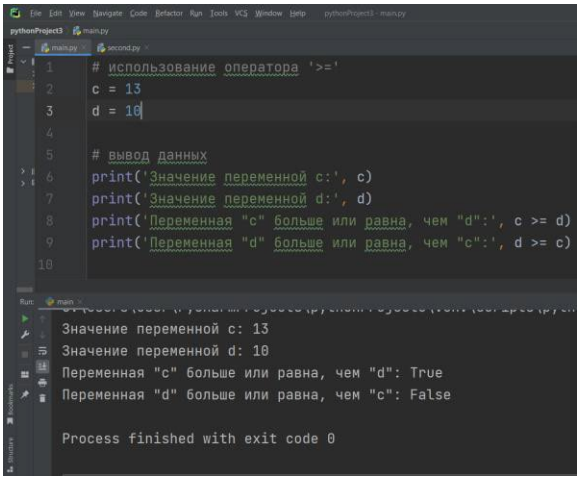


Рисунок 5 - Пример работы программы с использованием оператора сравнения ‘>=’

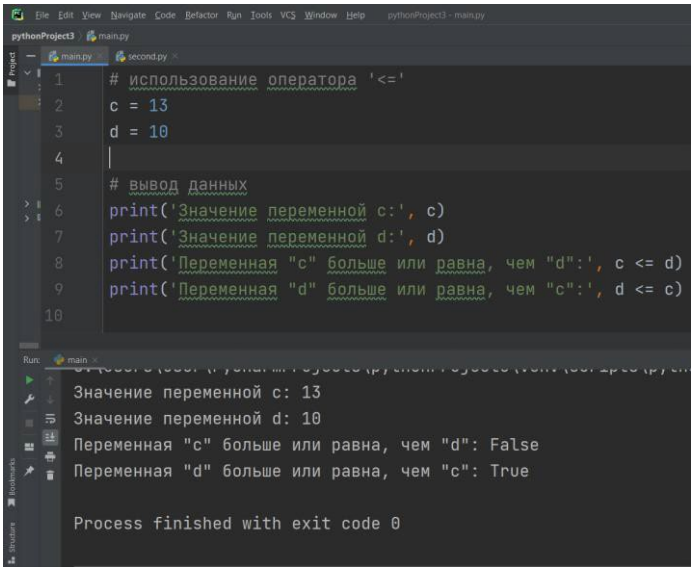


Рисунок 6 - Пример работы программы с использованием оператора сравнения ‘<=’

В последних четырех примерах алгоритм примерно одинаковый: вводится две переменн ые; выводятся их значения, после чего применяются определенные операторы присваиван ия. Врезультате применения операторов присваивания, на консоли можно заметить либо з начение **True**, либо **False**. Операции сравнения могут сравнивать самые различные объект ы - строки, числа, логические значения, однако, **оба операнда операции должны предста влять один и тот же тип данных.**

Приведем примеры использования операторов сравнения, которые применяются к с трокам (рис. 7, рис. 8):

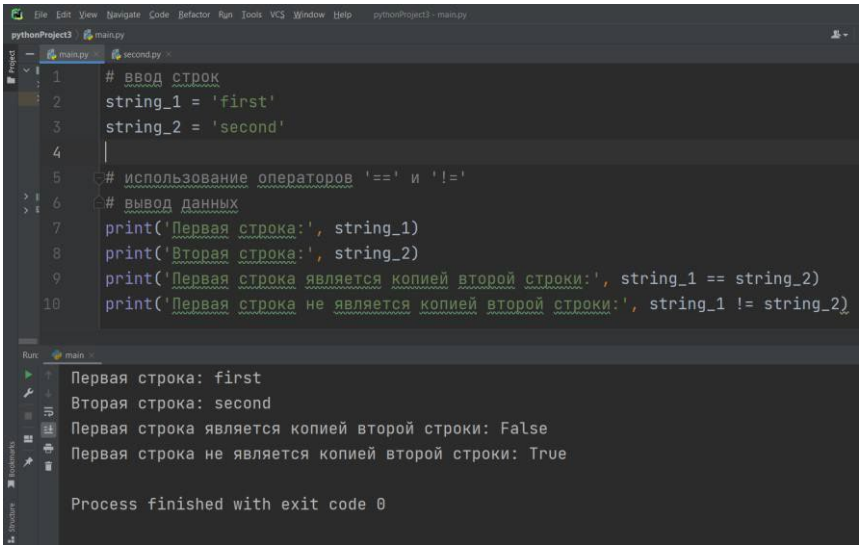
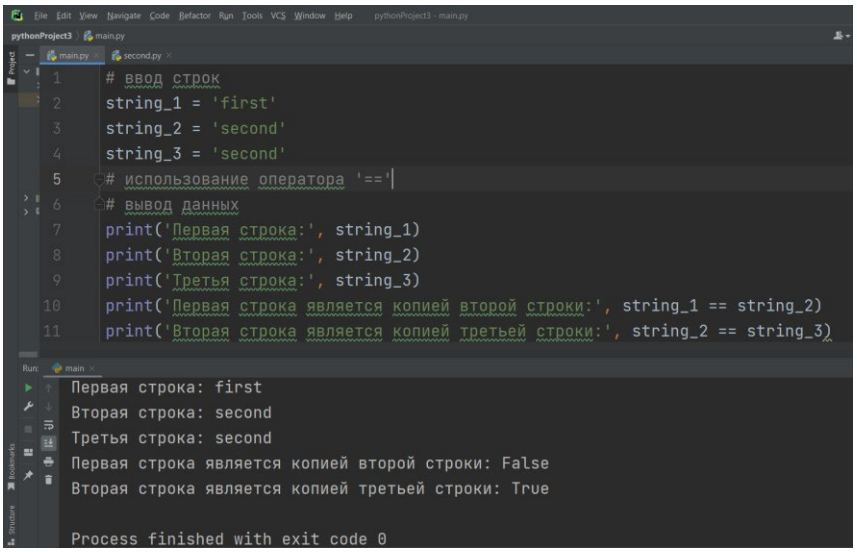


Рисунок 7 - Пример работы программы

с использованием оператора сравнения и строк

Как можно заметить, в строках 2 и 3 вводятся две строки с именами **‘string_1’** и **‘string_ 2’**. Затем, в строках 7 и 8 выводятся содержимое обеих строк. Кроме того, в строках 9 и 10 происходит

следующее: в строке 9 задается вопрос - содержимое строки **‘string_2’** является копией содержимого строки **‘string_1’**? Это ложь, поэтому на консоли можно заметитьзн ачение **False**. В строке 10 задается обратный вопрос, поэтому, соответственно, на консоли можно заметить



значение **True**.

Рисунок 8 - Пример работы программы
с использованием оператора сравнения и строк

В данном примере вводится еще одна строка с именем **‘string_3’** и содержимым, иденти чным содержимому строки с именем **‘string_2’**. Соответственно, когда консоли задаетсяво прос, одинаковы ли содержимые переменных **‘string_2’** и **‘string_3’**, в последней строке ко нсоли можно заметить значение **True**. Соответственно, когда спрашивается ободнозначнос ти содержимого строк **‘string_1’** и **‘string_2’**, на консоли можно заметить значение **False**.

Язык Python обладает всеми возможностями, которых следует ожидать от современ ного языка программирования. В тоже время, язык Python является объектно- ориентированным языком программирования (ООП). ООП является современным подходо м к решению задач с помощью вычислительных машин. В рамках ООП собственнаяинфор мация программы и команды, которые она передает компьютеру, записываются интуитивн о понятным образом. Это не является единственным способом разработки программ, но в больших проектах, как правило, предпочтительный.

Логические операции в Python

Для создания составных условных выражений любой сложности применяются спец иальные операции, которые называются *логическими операциями*. Рассмотрим основныело гические операции, которые можно использовать на языке программирования Python:

- оператор **and** (логическое умножение) применяется, как правило, исключительно д ля двух операндов (рис. 9):

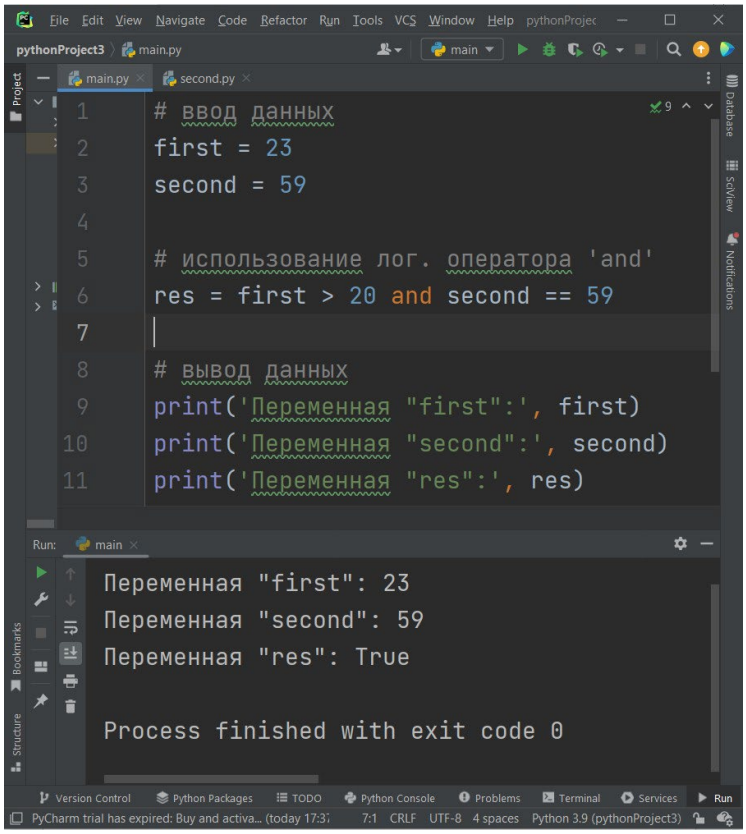


Рисунок 9 - Пример работы программы с использованием оператора *and*

В данном случае, оператор **and** сравнивает результаты двух выражений: **first > 20** и **second == 59.** Если оба этих выражений возвращают значение **True**, то оператор **and** возвращаетзнач ение **True** (как это можно заметить в третьей строке консоли).

Рассмотрим еще один прим ер, где в консоли можно заметить значение **False** (рис. 10):

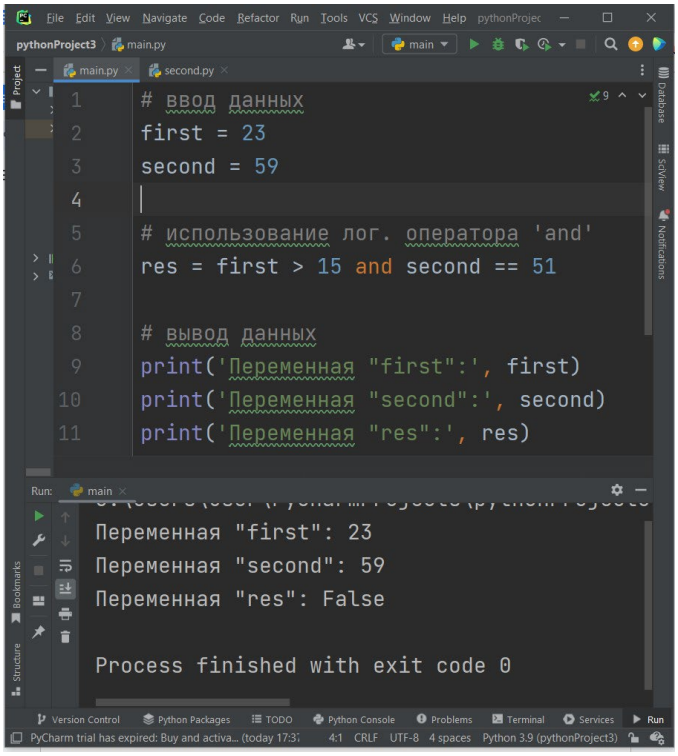


Рисунок 10 - Пример работы программы с использованием оператора *and*

- оператор **or** (логическое сложение), также может применяться к двум операндам с о динаковым типом (рис. 11):

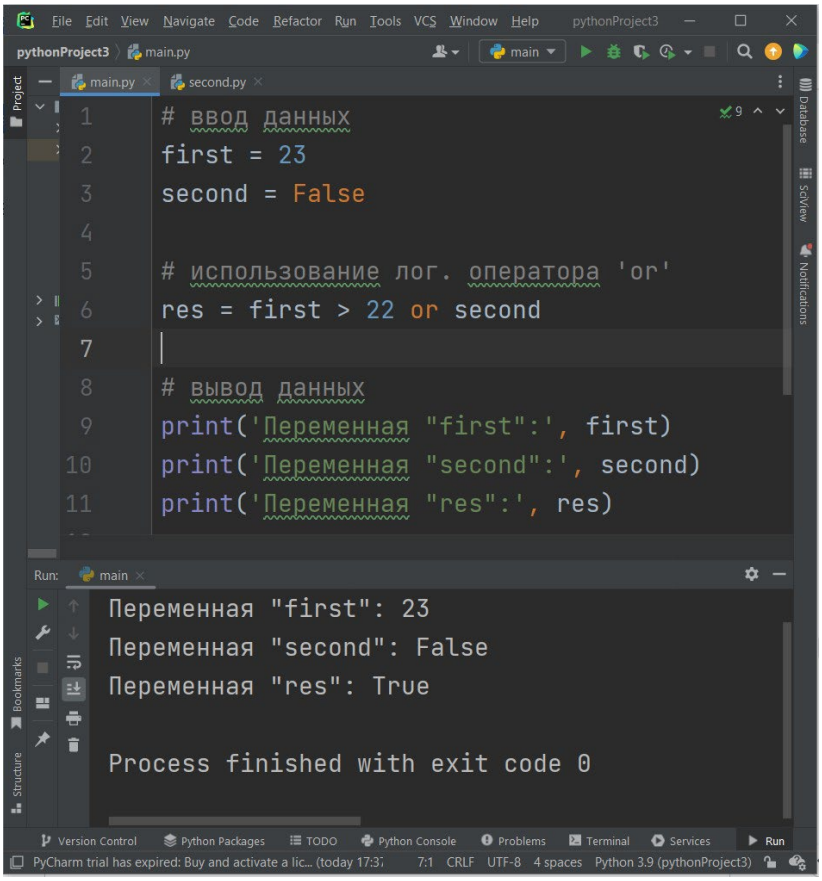


Рисунок 11 - Пример работы программы с использованием оператора *or* Проведем анализ работы данной программы: в строке 2 вводится переменная с именем **‘ first’** и значением **23**. В строке 3 вводится переменная с именем **‘second’** со значением **Fals**
e. Встроке 6 используется логический оператор **or**. Если значение переменной **‘first’** боль ше, чем **22**, выводится значение **True** (как в данном примере), в противном случае выводит сязначение **False**.

Рассмотрим еще один пример использования оператора **or**, когда на консоли выводится значение **False** (рис. 12):

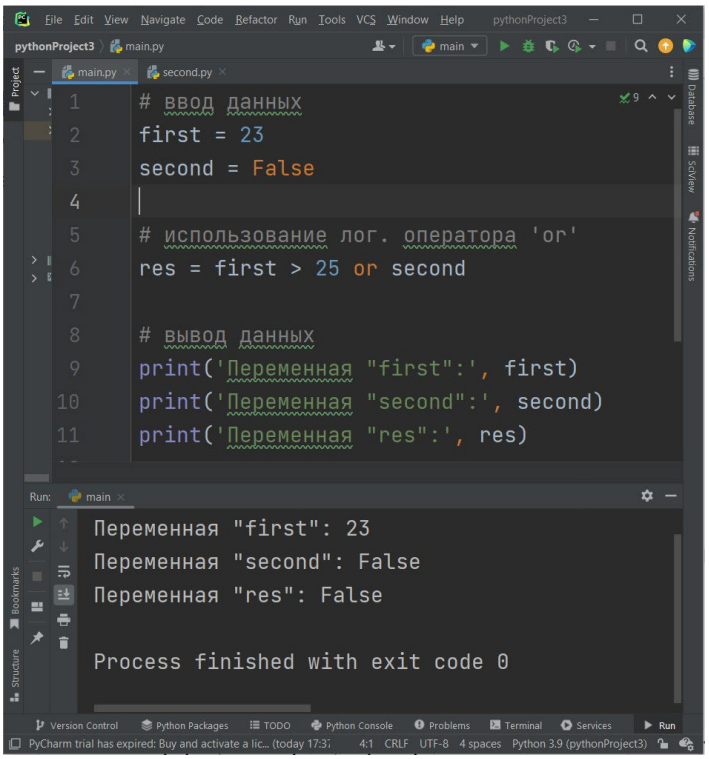


Рисунок 12 - Пример работы программы с использованием оператора *or*

В данном случае, наоборот, значение переменной **‘first’** меньше, чем **25**, поэтому в трет ьей строке консоли выводится значение **False**. В таком случае, если выражение, котороена ходится слева от оператора **or** истинно, тогда на консоли можно заметить значение **True**. В противном случае, можно заметить значение **False**.

- оператор **not** (логическое отрицание), который возвращает значение **True**, если выр ажение равно **False** (рис. 13

):

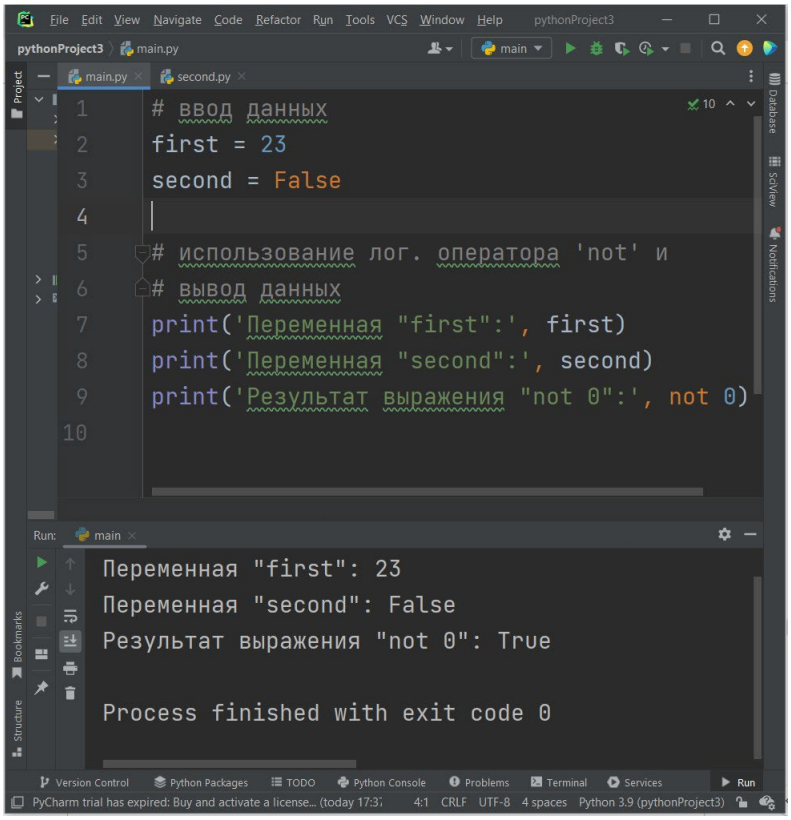


Рисунок 13 - Пример работы программы с использованием оператора *not*

Рассмотрим еще один пример использования оператора **or**, когда на консоли выводится значение **False** (рис. 14):

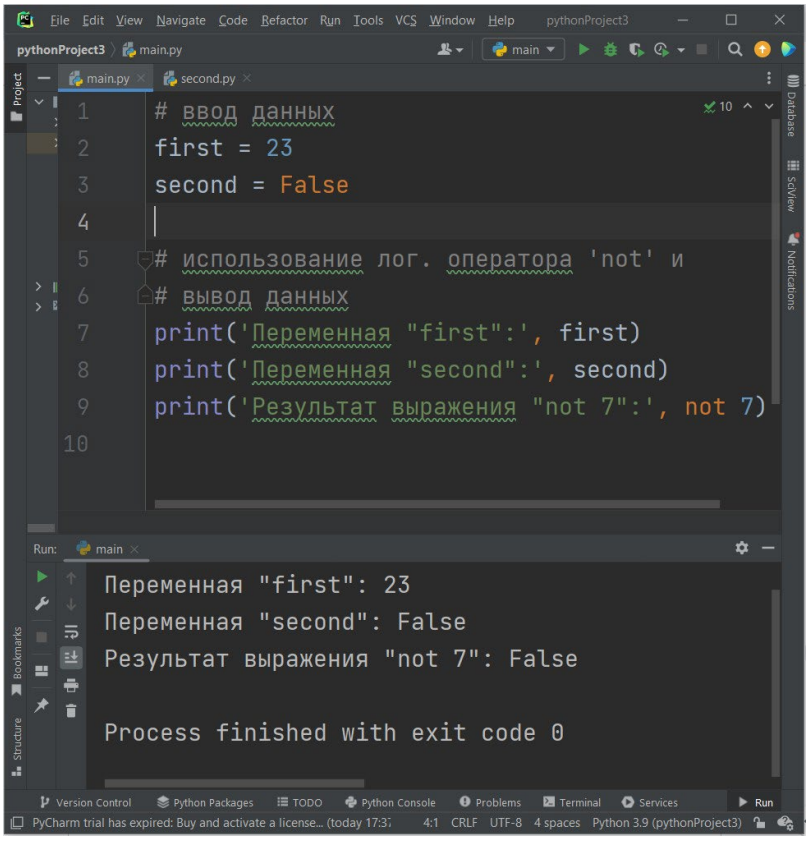


Рисунок 14 - Пример работы программы с использованием оператора *not***Применение оператора in**

Данный оператор возвращает значение **True**, если в некотором наборе значений ест ь определенное значение. Рассмотрим синтаксис применения оператора **in** (рис. 12):

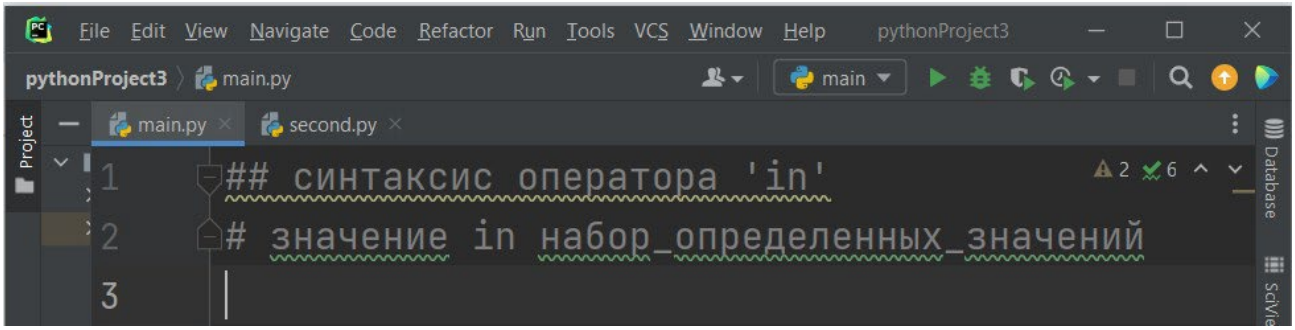


Рисунок 12 - Синтаксис применения оператора *in*

Можно, например, задать строку при помощи набора символов. С помощью оператора **i n** возможно проверить - есть ли в строке подстрока (набор из меньшего числа символов, че мвся строка) (рис. 13, рис. 14):

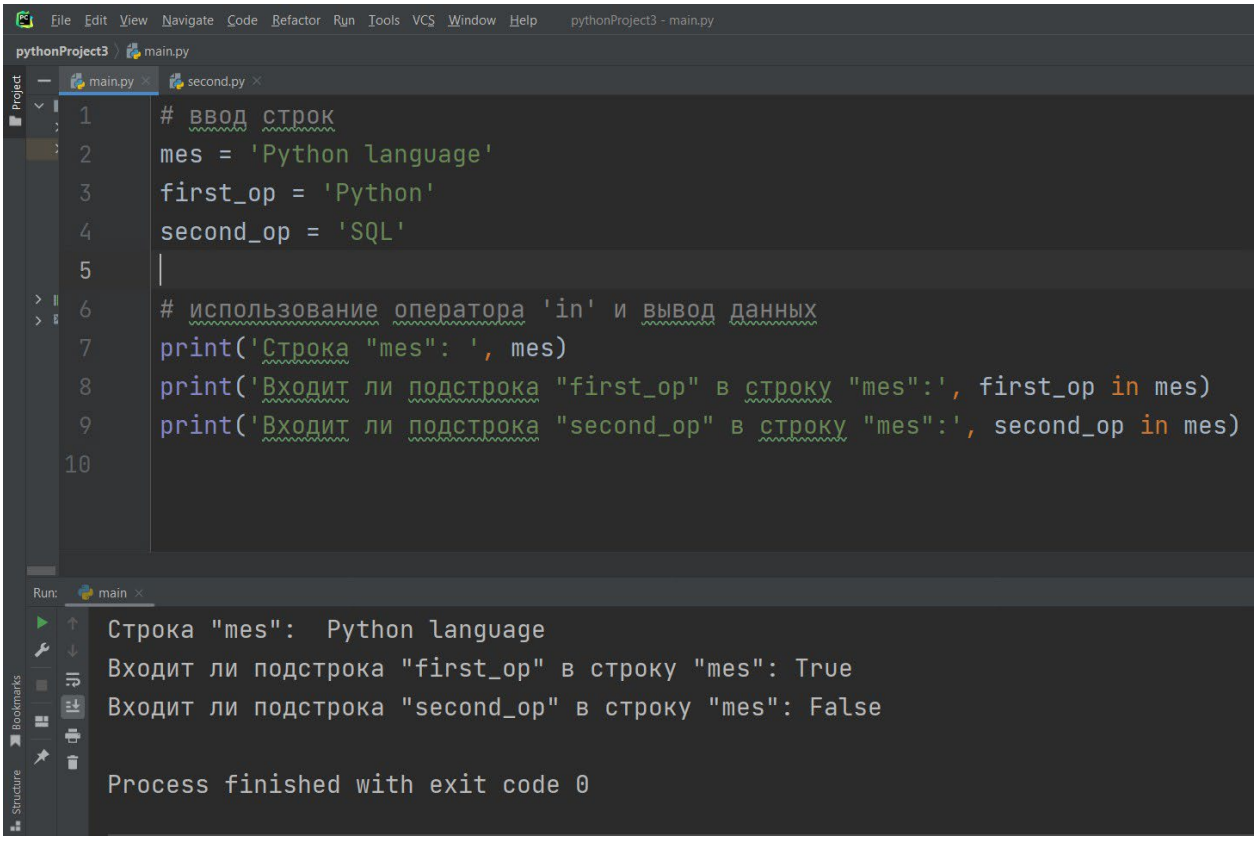


Рисунок 13 - Пример работы программы с использованием оператора *in*

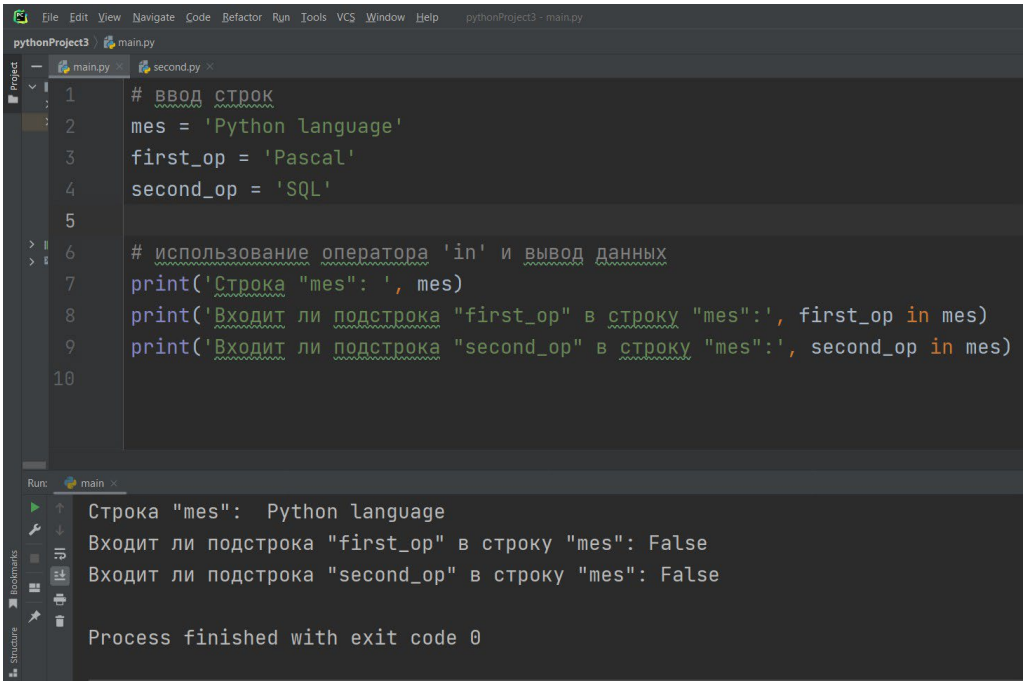


Рисунок 14 - Пример работы программы с использованием оператора *in*

Цикл while

Цикл **while** проверяет истинность некоторого условия, и если условие истинно, то выполняются инструкции цикла. Он имеет следующее формальное стандартное определение (рис. 15):

Рисунок 15 - Конструкция цикла while

```
while условное_выражение:
    инструкции
```

После ключевого слова **while** указывается условное выражение, и пока это выражение возвращает значение True, будет выполняться блок инструкций, который идет далее.

Все инструкции, которые относятся к циклу **while**, располагаются на последующих строках и должны иметь отступ от начала ключевого слова **while** (рис. 16):

```
1  number = 1
2
3  while number < 5:
4      print(f"number = {number}")
5      number += 1
6  print("Работа программы завершена")
```

Рисунок 16 - Пример работы цикла while

В данном случае цикл while будет выполняться, пока переменная number меньше 5. Сам блок цикла состоит из двух инструкций (рис. 17):

```
1  print(f"number = {number}")
2  number += 1
```

Рисунок 17 - Конструкции цикла

Здесь стоит обратить внимание на то, что они имеют отступы от начала оператора **while** - в данном случае от начала строки. Благодаря этому Python может определить, что они принадлежат циклу. В самом цикле сначала выводится значение переменной number, а потом ей присваивается новое значение. Также стоит не забывать о том, что последняя инструкция **print("Работа программы завершена")** не имеет отступов от начала строки, поэтому она не входит в цикл while.

Весь процесс цикла можно представить следующим образом:

1. Сначала проверяется значение переменной **number** - больше ли оно 5. И поскольку вначале переменная равна 1, то это условие возвращает True, и поэтому выполняются инструкции цикла;
2. Инструкции цикла выводят на консоль строку number = 1. И далее значение переменной number увеличивается на единицу - теперь она равна 2. Однократное выполнение блока инструкций цикла называется **итерацией**. То есть таким образом, в цикле выполняется первая **итерация**;
3. Снова проверяется условие number < 5. Оно по прежнему равно True, так как number = 2, поэтому выполняются инструкции цикла;

4. Инструкции цикла выводят на консоль строку number = 2. И далее значение переменной number опять увеличивается на единицу - теперь она равна 3. Таким образом, выполняется вторая итерация;
5. Опять проверяется условие number < 5. Оно по прежнему равно True, так как number = 3, поэтому выполняются инструкции цикла;
6. Инструкции цикла выводят на консоль строку number = 3. И далее значение переменной number опять увеличивается на единицу - теперь она равна 4. То есть выполняется третья итерация;
7. Снова проверяется условие number < 5. Оно по прежнему равно True, так как number = 4, поэтому выполняются инструкции цикла;
8. Инструкции цикла выводят на консоль строку number = 4. И далее значение переменной number опять увеличивается на единицу - теперь она равна 5. То есть выполняется четвертая итерация;
9. И вновь проверяется условие number < 5. Но теперь оно равно False, так как number = 5, поэтому выполняются выход из цикла. Все цикл - завершился. Дальше уже выполняются действия, которые определены после цикла. Таким образом, данный цикл произведет четыре прохода или четыре итерации.

В итоге при выполнении кода мы получим следующий консольный вывод (рис. 18):

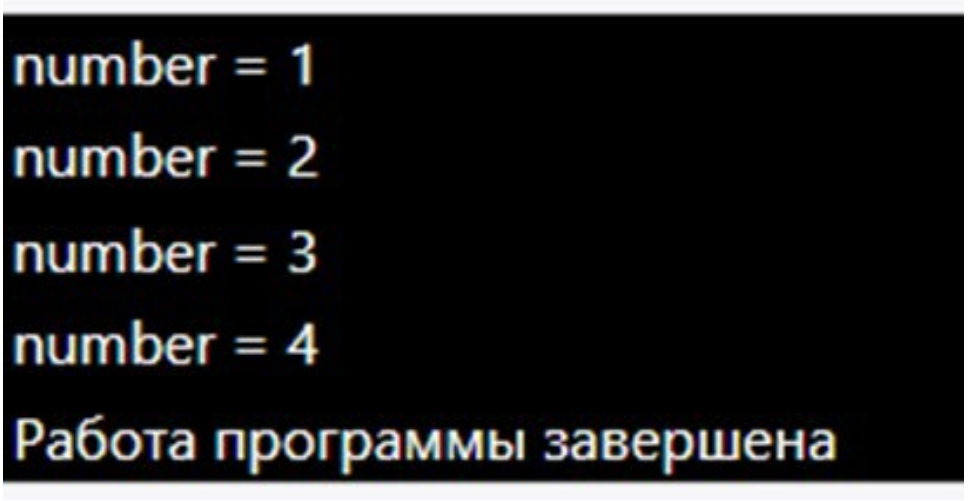


Рисунок 18 - Консольный вывод при использовании цикла while

Для цикла **while** также можно определить дополнительный блок **else**, инструкции которого выполняются, когда условие равно False (рис. 19):

```
number = 1

while number < 5:
    print(f"number = {number}")
    number += 1
else:
    print(f"number = {number}. Работа цикла завершена")
print("Работа программы завершена")
```

Рисунок 19 - Пример программы с циклом while

Задание Вариант 1

Создать программу, в которой используется следующая строка: **‘mes’ = ‘Python’**. Кроме того, в программе используется подстрока: **‘first_op’**. В данной программе необходимо реализовать оператор **in** и проверить - входит ли подстрока с именем **‘first_op’** в строку **‘mes’**. Результаты продемонстрировать на консоли.

Вариант

Создать программу, в которой используются переменные с именами **‘a’**, **‘c’** и **‘e’**. Необходимо использовать специальный оператор **‘!=’**, относительно двух пар переменных

- **‘a’** и **‘c’**, а затем **‘a’** и **‘e’**. Результаты продемонстрировать на консоли.

Вариант 3

Создать программу, в которой пользователь вводит три строки из символов. Необходимо вывести данные строки отдельно на консоль.

Вариант 4

Создать программу с использованием цикла **while** и оператора **continue**. Цикл **while** работает с переменной **‘z’**, значение которой изначально равно 3. Данный цикл выводит значение переменной **‘z’** во второй степени. При каждой итерации значение переменной **‘z’** увеличивается на единицу. Когда значение переменной становится равным 6, оператор **continue**приостанавливает работу цикла **while**.

Вариант 5

Создать программу, в которой используются переменные с именами **‘c’** и **‘d’**. Необходимо использовать специальный оператор **‘>=’**, относительно данных переменных. Результаты продемонстрировать на консоли.

Вариант 6.

Создать программу, в которой используются: переменная **‘q’** с типом **str**, переменная **‘w’** с типом **float** и переменная **‘e’** с типом **list.** Необходимо при помощи функции **type()** вывести на консоль типы данных переменных.

Вариант 7

Создать программу с использованием цикла **for**. Данный цикл перебирает значения переменной **‘x’** от 1 до 5 и переменной **‘y’** от 2 до 5 и выводит соответствующие значения на консоль.

Вариант 8

Создать программу в среде PyCharm, в которой вводятся с клавиатуры три переменные с типом данных **int**: **‘a’**, **‘b’** и **‘c’**. Затем в переменную **‘z’** записывается сумма значений данных переменных. Вывести на консоль значения введенных переменных и сумму.

Вариант 9

Создать программу, в которой пользователь вводит 5 строк из символов. Необходимо вывести данные строки отдельно на консоль.

Вариант 10

Создать программу, в которой используются переменные с именами **‘x’** и **‘y’**. Необходимо использовать специальный оператор **‘>=’**, относительно данных переменных. Результаты продемонстрировать на консоли.

Контрольные вопросы

1. Какую роль выполняет команда float в языке Python?
2. Что такое логическая операция?
3. Как можно передавать аргументы программе?
4. Какую роль выполняет функция boolean() в языке Python?
5. Какое минимальное количество аргументов можно передать программе на языке Python за один раз?

Критерии оценки:

Оценка «**отлично**» выставляется, если задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче. Студент изобразил блок-схему выполнения алгоритма, программа написано верно, студент ответил на все 5 контрольных вопросов.

Оценка «**хорошо**» выставляется, если задание выполнено полностью самостоятельно и полностью соответствует поставленной задаче, но при этом допущены несущественные неточности, устраненные без помощи преподавателя. Студент изобразил блок-схему выполнения алгоритма, программа написано верно, студент ответил на все 4 контрольных вопроса.

Оценка «**удовлетворительно**» выставляется, если задание выполнено не в полном объеме или не полностью соответствует поставленной задаче, при этом могут быть допущены

несущественные неточности, устраненные с помощью преподавателя. Студент изобразил блок-схему выполнения алгоритма, но не совсем верную, программа написано с частичными ошибками, студент ответил на все 3 контрольных вопроса.

Оценка «**неудовлетворительно**» выставляется, если задание не выполнено и полностью не соответствует поставленной задаче, допущены существенные неточности, которые обучающийся не может устранить

Вопросы для подготовки к дифференцированному зачету

ОП.02 Прикладные компьютерные программы в профессиональной деятельности

Специальность 08.02.15

1. Типы данных в языке программирования
2. Структура программы
3. Оператор чтения (ввода)
4. Оператор записи (вывода)
5. Понятие алгоритма и его свойства. Разработка алгоритмов сложной структуры
6. Ветвления в программах
7. Использование составного оператора
8. Оператор выбора
9. Цикл с предусловием
10. Цикл с постусловием
11. Цикл с параметром
12. Подпрограммы–функции
13. Подпрограммы-процедуры
14. Классификация языков программирования. Понятие интегрированной среды программирования.
15. Характеристика системы программирования. Процесс трансляции и выполнения программы.
16. Организация ветвлений. Операторы циклов (с предусловием, с постусловием, с параметром)
17. Операторы передачи управления. Разработка алгоритмов сложной структуры.
18. Строковый тип данных
19. Функции со строками
20. Процедуры со строками
21. Стандартные функции ord(x) и chr(x)
22. Одномерные и многомерные массивы, их формирование, сортировка, обработка
23. Работа со строками. Структуры и объединения.

24. Двумерные массивы
25. Множественный тип данных
26. Операции над множествами
27. Файлы и файловые переменные
28. Правило создания и заполнения файлов
29. Правило чтения из файлов
30. Текстовые файлы
31. Комбинированный тип данных
32. Работа с файлами записей
33. Указатели и динамические структуры
34. Связь между динамическими величинами и их указателями
35. Логические типы данных. Итеративные конструкции

Практические задания к экзамену

1. Найдите сумму элементов одномерного массива, больших 10. Элементы одномерного массива заданы с помощью датчика случайных чисел.
2. Разработать программу распечатки на экране двумерного массива размером 8 х 8.
3. Первый элемент геометрической прогрессии равен 3. Шаг прогрессии равен 2. Найти 10-ый элемент заданной прогрессии
4. Написать программу, позволяющую по данному числу (1 – 12) выводить название соответствующего ему месяца
5. Составить программу-генератор простых чисел. В основу положить формулу 2 х 2 + 29 при $0 \leq x \leq 28$.
6. В одномерном массиве из 15 элементов подсчитать количество чисел кратных заданному К.
7. Составить программу для нахождения 20-го элемента арифметической прогрессии. Первый элемент равен 4. Приращение прогрессии (шаг) равен 3.
8. Составить программу для нахождения наибольшего общего делителя (НОД) с применением процедуры.
9. В одномерном массиве из 13 элементов подсчитать количество отрицательных чисел. Элементы задаются с помощью датчика случайных чисел.
10. Вычислить значение функции
11. Распечатать фамилии рабочих бригады, начинающиеся с букв «А», с указанием их месячной зарплаты.
12. Распечатать фамилии детей детского сада, которые родились в определенном месяце. Указать их возраст и группу.

13. Записать в файл последовательного доступа N действительных чисел. Вычислить произведение компонентов файла и вывести на печать

14. Заполнить файл последовательного доступа целыми числами, полученными с помощью датчика случайных чисел.

15. Вычислить значение выражения по формуле 2-x – cos x + sin (2xy).

16. Составить программу для нахождения наибольшего общего делителя (НОД) с применением подпрограммы-функции.

17. Составить программу вычисления объема цилиндра, имеющего высоту H и радиус R.

18. Написать программу, которая по номеру дня недели (целому числу от 1 до 7) выдает в качестве результата количество уроков в вашем классе в этот день.

19. Вычислить значение функции x = 0,5 sin(x)

20. В одномерном массиве из 13 элементов подсчитать количество отрицательных чисел. Элементы задаются с помощью датчика случайных чисел.

21. Даны два действительных числа x и y, не равные друг другу. Меньшее из этих двух чисел заменить половиной их суммы, а большее – их удвоенным произведением.

22. Даны три действительных числа. Возвести в квадрат те из них, значения которых неотрицательны, и в четвертую степень – отрицательные.

23. Дана длина ребра куба. Найти площадь грани, площадь полной поверхности и объем этого куба.

24. В одномерном массиве b[n], заданном с помощью датчика случайных чисел, найти максимальный элемент

25. Составить программу-генератор простых чисел. В основу положить формулу при 1<x<36

26. Одномерный массив целых чисел b[n] задан с помощью ввода данных с клавиатуры. Найти произведение элементов массива

27. Одномерный массив b[n] задан с помощью датчика случайных чисел. Найти сумму четных элементов массива

28. В целочисленной последовательности есть нулевые элементы. Вывести на экран номера этих элементов

29. Двумерный массив 6x8 задан с помощью датчика случайных чисел. Распечатать на экране матрицу

30. Одномерный массив задан с помощью датчика случайных чисел на числовом отрезке от 0 до 9. Найти произведение всех компонент массива.

31. Одномерный массив из 21 элемента задан с помощью датчика случайных чисел. Подсчитать сумму отрицательных элементов

32. Одномерный массив из 25 элементов вводится с клавиатуры. Найти количество нечетных элементов.

33. Дано натуральное n. Вычислить n+1*(n-23*4)

34. Заполнить файл последовательного доступа f целыми числами, полученными с помощью датчика случайных чисел. Подсчитать количество четных чисел в последовательности.

35. Дан двумерный массив. Найти строку с максимальной суммой элементов. Дополнительный массив не использовать.

36. Функция print(), помимо вывода результатов работы программы, допускает проведение разнообразных операций с данными. Произведите 5 операций с различными типами данных.

37. Создайте 5 переменных и узнайте их тип с помощью type().

38. Используя функции словарей, создать два словаря: Cities = [Moscow, St.Petersburg, Vladimir] и Population = [17201245, 4520196, 3124075] и вывести их на экран.

39. Создать словарь Cities = [Moscow, St.Petersburg, Vladimir, Rostov], вывести его. Затем добавить в данный словарь два города: Omsk и Novosibirsk.

40. В программе вводятся три переменные с типом int. Необходимо вывести значения данных переменных на консоль.

41. Создать программу в среде PyCharm, в которой вводятся с клавиатуры три переменные с типом данных int: ‘a’, ‘b’ и ‘c’. Затем в переменную ‘z’ записывается сумма значений данных переменных. Вывести на консоль значения введенных переменных и сумму.

42. Создать программу в среде PyCharm, в которой вводятся с клавиатуры три переменные с типом данных float: ‘x’, ‘y’ и ‘z’. Затем в переменную ‘s’ записывается произведений значений данных переменных. Вывести на консоль значения введенных переменных и произведение.

43. В программе вводятся три переменные с типом int. Необходимо вывести значения данных переменных на консоль.

44. Создать программу, в которой пользователь вводит три строки из символов. Необходимо вывести данные строки отдельно на консоль.

45. Написать программу, в которой переменная ‘a’ имеет тип данных str, ‘b’ - int и ‘c’ - float. Ввести данные по этим переменным и вывести данные переменные на консоль.

46. Создайте 5 переменных и узнайте их тип с помощью type().

47. Создать программу, в которой используются: переменная ‘q’ с типом str, переменная ‘w’ с типом float и переменная ‘e’ с типом list. Необходимо при помощи

функции type() вывести на консоль типы данных переменных.

48. Создать программу в среде PyCharm, в которой вводятся с клавиатуры три переменные с типом данных int: ‘a’, ‘b’ и ‘c’. Затем в переменную ‘z’ записывается сумма значений данных переменных. Вывести на консоль значения введенных переменных и сумму.

49. Создать программу, в которой используется следующая строка: ‘mes’ = ‘Python’. Кроме того, в программе используется подстрока: ‘first_op’. В данной программе необходимо реализовать оператор in и проверить - входит ли подстрока с именем ‘first_op’ в строку ‘mes’. Результаты продемонстрировать на консоли.

50. Создать программу, в которой используются переменные с именами ‘a’, ‘c’ и ‘e’. Необходимо использовать специальный оператор ‘!=’, относительно двух пар переменных - ‘a’ и ‘c’, а затем ‘a’ и ‘e’. Результаты продемонстрировать на консоли.

51. Создать программу, в которой пользователь вводит три строки из символов. Необходимо вывести данные строки отдельно на консоль.

52. Создать программу с использованием цикла while и оператора continue. Цикл while работает с переменной ‘z’, значение которой изначально равно 3. Данный цикл выводит значение переменной ‘z’ во второй степени. При каждой итерации значение переменной ‘z’ увеличивается на единицу. Когда значение переменной становится равным 6, оператор continue приостанавливает работу цикла while.

53. Создать программу, в которой используются переменные с именами ‘c’ и ‘d’. Необходимо использовать специальный оператор ‘>=’, относительно данных переменных. Результаты продемонстрировать на консоли.

54. Создать программу, в которой используются: переменная ‘q’ с типом str, переменная ‘w’ с типом float и переменная ‘e’ с типом list. Необходимо при помощи функции type() вывести на консоль типы данных переменных.

55. Создать программу с использованием цикла for. Данный цикл перебирает значения переменной ‘x’ от 1 до 5 и переменной ‘y’ от 2 до 5 и выводит соответствующие значения на консоль.

56. Создать программу в среде PyCharm, в которой вводятся с клавиатуры три переменные с типом данных int: ‘a’, ‘b’ и ‘c’. Затем в переменную ‘z’ записывается сумма значений данных переменных. Вывести на консоль значения введенных переменных и сумму.

57. Создать программу с использованием цикла for. Данный цикл перебирает значения переменной ‘a’ от 3 до 7 и переменной ‘b’ от 10 до 13 и выводит соответствующие значения на консоль.

58. Создать программу, в которой происходит перебор элементов списка с именем ‘words’ при помощи цикла while.

59. Создать программу, в которой реализован оператор выхода из цикла break. Изначально вводится переменная ‘num’ со значением 0. При каждой итерации цикла значение переменной ‘num’ увеличивается на единицу. Когда значение переменной ‘num’ становится равным 3, оператор break завершает работу цикла (на консоли должны быть выведены значения 1 и 2).

60. Создать программу с использованием цикла while и оператора continue. Цикл while работает с переменной ‘z’, значение которой изначально равно 3. Данный цикл выводит значение переменной ‘z’ во второй степени. При каждой итерации значение переменной ‘z’ увеличивается на единицу. Когда значение переменной становится равным 6, оператор continue приостанавливает работу цикла while.

61. Создать программу, в которой реализован оператор выхода из цикла break. Изначально вводится переменная ‘s’ со значением 0. При каждой итерации цикла значение переменной ‘s’ увеличивается на единицу. Когда значение переменной ‘s’ становится равным 5, оператор break завершает работу цикла (на консоли должны быть выведены значения с 1 по 4 включительно).

62. Создать программу, в которой используются: переменная ‘q’ с типом str, переменная ‘w’ с типом float и переменная ‘e’ с типом list. Необходимо при помощи функции type() вывести на консоль типы данных переменных.

63. Создать программу с использованием цикла for. Данный цикл перебирает значения переменной ‘x’ от 1 до 5 и переменной ‘y’ от 2 до 5 и выводит соответствующие значения на консоль.

64. Создать программу с использованием цикла while и переменной ‘y’, значение которой, изначально, равно 2. Данный цикл выводит значения переменной ‘y’ в третьей степени до тех пор, пока значение переменной ‘y’ не станет равным 10. При каждой итерации значение переменной ‘y’ увеличивается на 2.

65. Создать программу, в которой вводятся списки ‘m’ = [1, 3, 5, 7, 9] и ‘x’ = [11, 23, 35]. Необходимо рассчитать количество элементов в данных списках и вывести данные значения на консоль.

66. Создать программу, в которой происходит перебор элементов списка с именем ‘words’ при помощи цикла while.

67. Создать программу, в которой вводится список ‘x’ = [1, 2, 3, 4, 5, 6]. Необходимо найти минимальный элемент в данном списке и вывести его на консоль.

68. Создать программу, в которой используется строка ‘z’ = ‘abc’. Необходимо применить оператор “*” и вывести данную строку 4 раза подряд в строчку.

69. Создать программу, в которой реализован оператор выхода из цикла break. Изначально

вводится переменная ‘s’ со значением 0. При каждой итерации цикла значение переменной ‘s’ увеличивается на единицу. Когда значение переменной ‘s’ становится равным 5, оператор break завершает работу цикла (на консоли должны быть выведены значения с 1 по 4 включительно).

70. Создать программу, в которой используются переменные ‘a’ и ‘b’. Реализовать в данной программе оператор ‘if’ со следующим условием: если значение переменной ‘a’ больше, чем значение переменной ‘b’, то выводится значение переменной ‘b’; если значения переменных равны, то выводится сообщение ‘Значения переменных равны’; в противном случае, выводится значение переменной ‘b’. Реализовать все три итерации (в каждой итерации выполняется свое условие). Результаты продемонстрировать на консоли.

71. Создать программу, в которой происходит перебор элементов списка с именем ‘words’ = [‘one’, ‘two’, ‘three’, ‘four’] при помощи цикла while.

72. Создать программу с использованием цикла while и переменной ‘y’, значение которой, изначально, равно 2. Данный цикл выводит значения переменной ‘y’ в третьей степени до тех пор, пока значение переменной ‘y’ не станет равным 10. При каждой итерации значение переменной ‘y’ увеличивается на 2.

73. Создать программу, в которой осуществлена триангуляция. Задать точечные изменения по осям x и y от 0 до 4.

74. Создать программу в среде PyCharm, в которой вводятся с клавиатуры три переменные с типом данных float: ‘x’, ‘y’ и ‘z’. Затем в переменную ‘s’ записывается произведений значений данных переменных. Вывести на консоль значения введенных переменных и произведение.

75. Используя библиотеки matplotlib и scipy, произвести построение 2D-графика, где значения переменной *x*(горизонтальная ось) варьируются от 0 до 10, а значения переменной *y* - от 10 до 20. Добавить подписи к графику, вывести его на экран.

76. Создать программу, в которой пользователь вводит три строки из символов. Необходимо вывести данные строки отдельно на консоль.

77. Создать класс с именем SecondClass() и реализовать в данном классе метод *_len_()* и операцию *len()*. При первой итерации посчитать количество элементов в объекте a = [‘one’, ‘two’, ‘three’, ‘four’], а при второй итерации - посчитать количество элементов в объекте b = [1, 3, 5, 7, 9].

78. Используя библиотеки matplotlib и scipy, произвести построение 2D-графика, где значения переменной *x*и *y* варьируются от 0 до 50. Добавить подписи к графику, вывести его на экран.

79. Создать программу, в которой осуществлена триангуляция. Задать точечные изменения по осям x и y от 0 до 6.

80. Создать программу с использованием цикла for. Данный цикл перебирает значения переменной ‘a’ от 3 до 7 и переменной ‘b’ от 10 до 13 и выводит соответствующие значения на консоль.

81. Создать программу, в которой вводятся два числа с типом данных float. После ввода данных чисел, переменные складываются. Реализовать исключение, при котором, при вводе строки, пользователь получает сообщение: “Вы используете неверный тип данных”.

82. Написать программу, в которой используются две переменные - ‘a’ с типом данных float и ‘b’ с типом данных int. Реализовать конвертирование переменной ‘a’ в тип данных int. После этого, произвести умножение данных переменных. Реализовать исключение, при котором, при вводе строки, пользователь получает сообщение: “Аварийное завершение программы!”.

83. Используя библиотеки matplotlib и scipy, произвести построение 2D-графика, где значения переменной *x*(горизонтальная ось) варьируются от 0 до 10, а значения переменной *y* - от 10 до 20. Добавить подписи к графику, вывести его на экран.

84. Создать программу, в которой вводятся три числа с типом данных int. После ввода данных чисел организовать умножение данных переменных. Реализовать исключение, при котором, при вводе строки, пользователь получает сообщение: “Вы используете неверный тип данных”.

85. Создать класс с именем SecondClass() и реализовать в данном классе метод *len_()* и операцию *len()*. При первой итерации посчитать количество элементов в объекте
a = [‘one’, ‘two’, ‘three’, ‘four’], а при второй итерации - посчитать количество элементов в объекте b = [1, 3, 5, 7, 9].

86. Написать программу, в которой используются три переменные - ‘a’ с типом данных float, ‘b’ с типом данных int и ‘c’ - с типом данных float. Реализовать конвертирование переменных ‘a’ и ‘c’ в тип данных int. После этого, произвести умножение данных переменных. Реализовать исключение, при котором, при вводе строки, пользователь получает сообщение: “Аварийное завершение программы!”.

87. Создать программу, в которой осуществлена триангуляция. Задать точечные изменения по осям x и y от 0 до 6.

88. Создать программу, в которой вводятся три числа с типом данных float. После ввода данных чисел, из первой введенной вычитаются две остальные. Реализовать исключение, при котором, при вводе строки, пользователь получает сообщение: “Вы используете неверный тип данных”.