

Контрольное задание к зачету

Включает два задания: 1 – задание практической по созданию программного кода на Python; 2 – вопросы по лекциям

1 Задание. Написать программный код на Python по решению задачи и ответить на вопросы преподавателя

Требования к результатам выполнения работы и условия успешной сдачи 1-го задания контрольной работы:

1. Подготовлено пошаговое описание решений представленных задач.
 2. Разработана программная реализация представленных задач.
 3. Подготовлено краткое описание разработанного программного кода.
 4. Программная реализация представленных примеров выложены в личный репозиторий на GitHub.
 5. Ответить на вопросы преподавателя
1. Написать функцию, имеющую 3 параметра: первые 2 - числа, третий - операция, которая должна быть произведена над ними. Если третий параметр «+», то нужно сложить числа, если «-» — вычесть, «*» — умножить, «/» — разделить (первое на второе). Функция возвращает результат выполнения операции над числами. Если операция не совпадает с указанными выше, то выводится сообщение "Неизвестная операция", и возвращается значение None.
 2. Напишите функцию, которая для заданного радиуса r вычисляет площадь круга и длину окружности. Функция возвращает кортеж из 2 значений.
 3. Напишите функцию, которая находит наибольший общий делитель двух чисел, используя модифицированный алгоритм Евклида: нужно заменять большее число на остаток от деления большего на меньшее до тех пор, пока этот остаток не станет равен нулю; тогда второе и есть НОД. Функция должна возвращать найденное значение.
 4. Дана дробь nm , n и m - натуральные числа. Напишите 2 функции, которые сокращают эту дробь, то есть находят числа p и q такие, что $nm=pq$, и дробь pq — несократимая:
 - аргументами функции являются числа n , m , функция возвращает кортеж (p, q) ;
 - аргументом функции является список $[n, m]$, функция не возвращает значения, а изменяет этот список на $[p, q]$.
- Для поиска НОД воспользуйтесь функцией из предыдущего задания.
5. Напишите функцию для решения уравнений степени не выше второй (квадратные и линейные):
 - если у функции три аргумента, их надо трактовать как a , b и c в уравнении $ax^2+bx+c=0$;
 - если два — как коэффициенты b и c в уравнении $bx+c=0$;
 - если один — как коэффициент c в уравнении $c = 0$;

- если список коэффициентов пуст или коэффициентов больше трёх, то функция должна вернуть None.

Функция возвращает список, содержащий все корни уравнения (два, один или ни одного). Если корнем является любое значение x, функция возвращает список, содержащий символ «*» (["*"]).

6. Напишите функцию, параметрами которой являются слово и любое количество произвольных строк. Функция возвращает список строк, в которых встречается указанное в первом параметре слово. Строки могут состоять из букв и пробелов. Регистр букв не имеет значения. Например, при вызове f('Два', 'двадцать пять', 'сорок два') функция вернет список ['сорок два']. Вызовите функцию для случая, когда строки хранятся в заранее созданном списке.

7. Напишите функцию, которая имеет произвольное количество именованных параметров. Функция возвращает список всех имен параметров и сумму всех значений. Например, при вызове f(a=2, b=3,

8. Создайте собственный модуль, поместив в него ваши функции, созданные в предыдущих заданиях. Напишите любую программу с использованием функций этого модуля.

9. Ниже приведены класс Point (точка), у которого имеются 2 атрибута x и y (координаты) и методы __init__() и __str__(), и класс Rect (прямоугольник), у которого есть:

- два атрибута (верхний левый угол и правый нижний угол прямоугольника). Значениями атрибутов являются объекты класса Point;
- методы __init__() и __str__();
- метод sides(), возвращающий длины сторон прямоугольника;
- метод perim(), вычисляющий периметр прямоугольника.

```
class Point:
    def __init__(self, x, y):
        self.x = x
        self.y = y

    def __str__(self):
        return "Point(" + str(self.x) + ", " + str(self.y) + ")"

class Rect:
    def __init__(self, top_left, bottom_right):
        self.A = top_left
        self.C = bottom_right

    def __str__(self):
        r = "Rect(" + str(self.A) + ", "
        r = r + str(self.C) + ")"
        return r

    def sides(self):
        return abs(self.C.x - self.A.x), abs(self.A.y - self.C.y)

    def perim(self):
        a, b = self.sides()
        return 2 * (a + b)
```

Задание:

- продемонстрируйте работу с классами, создав необходимые объекты и вызвав все их методы;

- создайте аналогичный класс для треугольника с такими же методами.

10. Создайте класс Length (Длина), имеющий свойства:

- value (значение),
- unit (единица измерения).

При изменении единицы измерения значение должно соответственно меняться. Например, при переходе от сантиметров к метрам значение должно уменьшаться в 100 раз.

Допустимые значения свойства unit: 'см', 'м', 'км'. Организуйте эту проверку.

Продемонстрируйте работу с классом.

11. Создайте класс Ведомость, имеющий

атрибут класса:

- список_дисциплин (значением является список с названиями дисциплин);

свойства:

- дисциплина (при задании значения проверять наличие дисциплины в атрибуте список_дисциплин),

- группа;

методы:

- put – добавляет в ведомость информацию об оценке студента (фамилия, оценка – параметры метода). Для хранения данных внутри класса используйте словарь, в котором ключом является фамилия студента. Возможные оценки – «отлично», «хорошо», «удовл.», «неудовл.», «н/я»;
- get – возвращает оценку, полученную студентом (фамилия студента – параметр метода);
- change – изменяет оценку, полученную студентом (фамилия студента и новая оценка – параметры метода);
- del – удаляет информацию о студенте из ведомости (фамилия студента – параметр метода);
- result – возвращает кортеж из 5 чисел (количество каждого вида оценок в ведомости);
- __init__ – конструктор;
- __str__ – возвращает строку, содержащую заголовки (название экзамена, группа) и результаты экзамена в виде таблицы;
- count – возвращает количество студентов в ведомости;
- names – возвращает список фамилий, имеющих в ведомости.

2 Задание. – Дать полный ответ на вопросы по лекциям:

2.1. Современные компьютерные технологии. Эволюция автоматизированных систем управления компанией: MRP, MRP II, ERP, CRM, SRM, SCM, MES, PLM, BPM - системы. Эволюция подходов к интеграции ИС.

2.2. Технологии BPM. Технология автоматизации бизнес-процессов, основанная на использовании программных роботов и искусственного интеллекта RPA (Robotic Process Automation). Инструменты RPA. Системы класса BPMS (Business Process Management Suite). Внутренняя архитектура системы класса BPMS. Требования к системе класса BPMS.

2.3. Компьютерные технологии бизнес-аналитики (платформы Data Discovery). Технологии лидеров рынка BI – платформы Tableau, Qlik Sense, Power BI. Внедрение и управление данными в масштабах организации. BI-платформа для интерактивного анализа данных Tableau. Расширенная (предиктивная) аналитика в Tableau, Qlik Sense, Power BI.

2.4. Современная ИТ-инфраструктура. Преимущества дата-центричной архитектуры ИТ для современного предприятия.

2.5. Информационно-аналитические системы. Методы и задачи интеллектуального анализа данных, машинного обучения и обработки больших данных. Обзор рынка

информационно-аналитических систем: BI-системы, Data Science and Machine Learning Platforms.

2.6. Области применения методов и технологий интеллектуального анализа данных, машинного обучения и обработки больших данных. Понятие «Интернет Вещей». Примеры и основные области применения «Интернета Вещей». Интеллектуальное управление и анализ данных в «Интернете Вещей». Диагностика в интеллектуальных системах «Интернета Вещей».

2.7. Инструментальные средства управления проектами. Рынок специализированного программного обеспечения для управления проектами (решения класса Project Management, PM).

2.8. Этапы проекта разработки систем искусственного интеллекта и процессы жизненного цикла систем с учетом требований информационной безопасности.

2.9. Методологии управления ЖЦ ИС. Гибкие методология управления проектами разработки и внедрения систем искусственного интеллекта.

Компьютерные технологии управления жизненным циклом информационной системы.

2.10. Модели жизненного цикла информационной системы. Основные этапы жизненного цикла информационной системы: планированием, анализом требований (ТЭО, ТЗ), проектированием, реализацией, внедрением и эксплуатацией. Состав и содержание проектных работ на различных этапах жизненного цикла.

2.11. Содержание стадий внедрения ИС: анализа требований, уточнения требований и проектирования ИС. Подходы к сбору требований в различных методологиях разработки ИС. Подходы к проектированию: структурный (функциональный подход) SADT, IDEF0, DFD, IDEF3, ER; объектно-ориентированный подход UML, методология ARIS. Управление проектами (PMBok). Обзор CASE-средств.

2.12. Современные управленческие концепции проектного менеджмента, основанные на принципах гибких методологий по управлению проектами.. Достоинства и недостатки гибкого управления проектами. Требования к информационному обеспечению адаптивных процессов проектного управления. On-line сервисы для поддержки гибких методологий.

2.13. Компьютерные технологии интеллектуального анализа данных и обработки больших данных. Цикл обработки данных: поиск данных, сбор данных, очистка данных, трансформация данных, интеллектуальный анализ данных, интерпретация и практическое применение результатов.

2.14. Компьютерные технологии программирования. Обзор современных языков программирования, классификация и сравнительный анализ наиболее распространенных языков программирования. Основные парадигмы программирования (императивное программирование, декларативное программирование, объектно-ориентированное программирование, функциональное программирование) и их сравнительный анализ. Классификация и специфика применения языка программирования Python.