Бауманский учебный центр «Специалист»

Программа курса Программирование на Python. Уровень 2. Объектно - ориентированное программирование

Код курса: **ПИТОН2**

Тема	Ак. часов
Модуль 1. Введение • Знакомство. Цели и задачи курса. Расписание занятий. • Установка PyCharm, знакомство со средой разработки и настройка. Проверка Python 3.15 и выбранного интерпретатора.	**1.5**
Модуль 2. Основы ООП • Понятия ООП: полиморфизм, инкапсуляция, абстракция и наследование. • Понятия класса, объекта и экземпляра. Атрибуты и методы класса. • Ссылочная модель Python: класс и экземпляр как объекты. • **Практикум:** реализация собственных типов данных с помощью классов.	**6.5**
Модуль 3. Инкапсуляция и доступ к содержимому класса • Создание собственных классов как новых типов данных. • Создание атрибутов и методов новых типов данных. • Сокрытие и доступ к атрибутам и методам. Приватные атрибуты и name mangling. • **Практикум:** начало работы над сквозным проектом. • Проектирование и реализация консольных игр.	**8**
Модуль 4. Полиморфизм • Суть полиморфизма. Специальные методы. • Методы отображения __str__ и __repr__. • Поведение контейнера и последовательности. Итерируемость. • **Практическая работа:** создание итератора.	**4**
Модуль 5. Функциональное программирование • Функция как объект. • Функциональное программирование: map, filter, functools.reduce. • Замыкание. Лямбда-функция. • **Практикум:** продолжение работы над сквозным проектом.	**4**
Модуль 6. Функции декораторы • Понятие декоратора. Создание функций-декораторов. • Вызываемые экземпляры. Создание классов-декораторов. • **Практикум:** создание собственной функции-декоратора.	**4**
Модуль 7. Отношения между классами • Виды отношений между классами. • Агрегация. Композиция. • **Практическая работа:** продолжение работы над сквозным проектом.	**4**

Модуль 8. Наследование • Композиция классов. Наследование и иерархия классов. • Функция super(). Множественное наследование на примере mixin-классов. • Порядок разрешения методов (MRO), функции isinstance() и issubclass(). • Краткое введение в аннотации типов и роль статического анализатора как подготовка к новой теме. • Ограничения множественного наследования в системе типов Python 3.15. Декоратор @typing.disjoint_base. • Несвязанные disjoint-базы не должны иметь общего потомка. Совместимый анализатор типов выявляет конфликт. Сам декоратор не запрещает наследование во время выполнения. • **Практикум:** реализация иерархии классов. **Короткое задание:** разобрать конфликт двух несвязанных disjoint-баз и диагностику совместимого анализатора; сравнить с допустимой иерархией.	**6**
Модуль 9. Абстракция • Понятие интерфейса. Абстрактные классы. • Декораторы модуля abc. Примеры реализации требований к определённому интерфейсу. • Подведение итогов по сквозному проекту. Демонстрация работы и объяснение архитектуры.	**2**
Аудиторная нагрузка в классе с преподавателем	**40 ак.ч. + 40 ак.ч. самостоятельно**

По окончании обучения на курсе проводится итоговая аттестация. Аттестация проводится в виде теста на последнем занятии или на основании оценок практических работ, выполняемых во время обучения на курсе.

Контакты центра

Для частных лиц
+7 (495) 232-32-16

Для организаций
+7 (495) 780-48-44

E-mail
info@specialist.ru