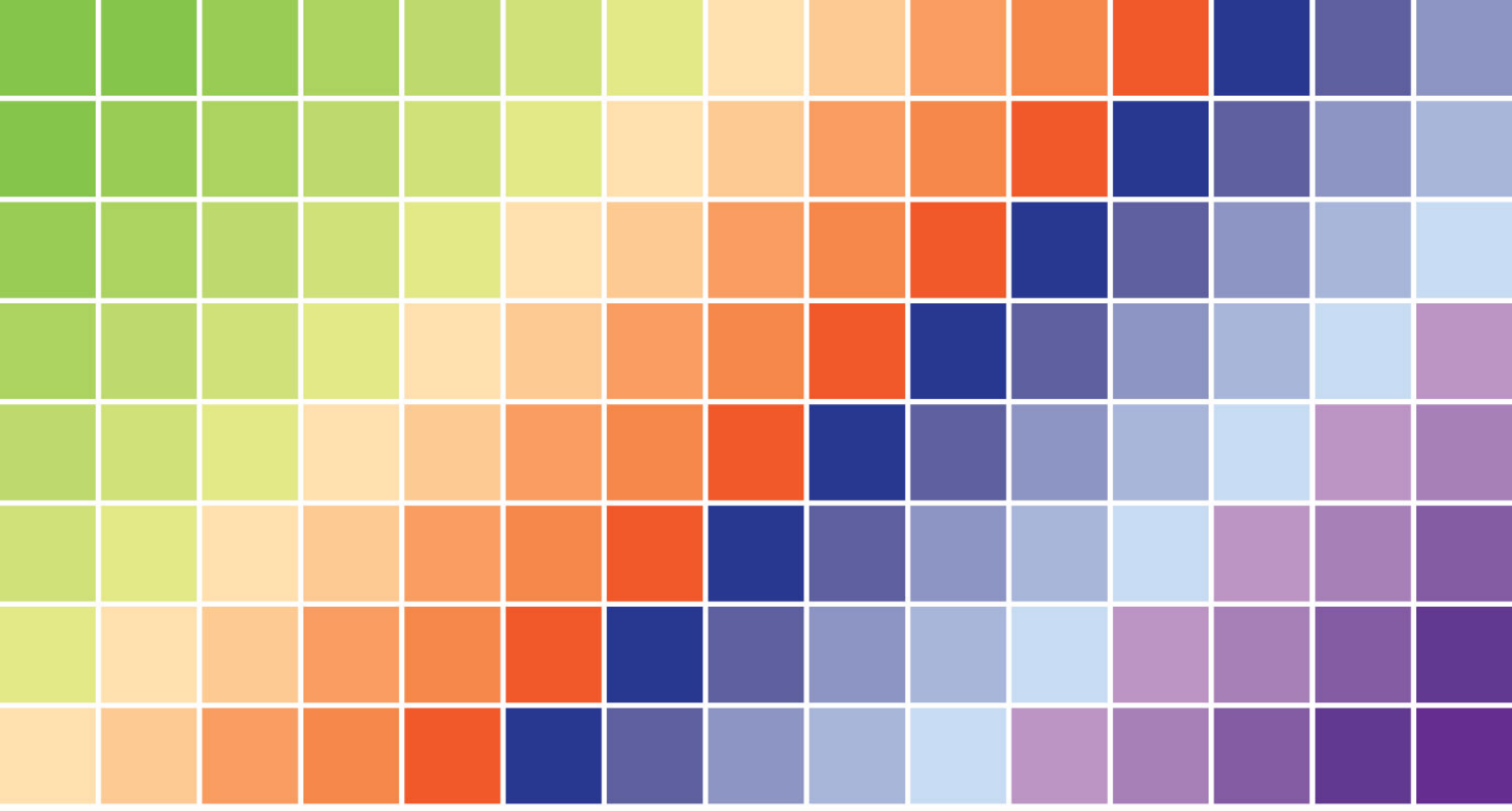


School of Computer Science

СТАНДАРТНЫЕ БИБЛИОТЕКИ PYTHON

ПРОГРАММИРОВАНИЕ НА PYTHON

Лекции для IT-школы

ВОПРОСЫ
СОЗДАНИЕ МНОЖЕСТВА

Какой способ создать множество является недопустимым:

1. a_set = {}
2. a_set = {1}
3. a_set = {1, 2}
4. a_set = {(1,),(2,)}
5. a_set = set(list(tuple([1]*10)))

???

ВОПРОСЫ
УДАЛЕНИЕ ИЗ МНОЖЕСТВА

a_set = {1, 2, 3, 4, 5}

Какой вызов не породит исключение:

1. a_set.remove(6)
2. a_set.delete(6)
3. a_set.erase(6)
4. a_set.discard(6)

???

ВОПРОСЫ
ДОБАВЛЕНИЕ В МНОЖЕСТВО

Как добавить значения из списка upd_list в множество a_set:

1. a_set.insert(upd_list)
2. a_set.add(upd_list)
3. a_set.update(upd_list)
4. a_set.append(upd_list)

???

ВОПРОСЫ
ИТЕРАЦИЯ ПО МНОЖЕСТВУ

Сколько букв напечатает этот код?

```
>>> for letter in set('apple'):
        print(letter)
```

В каком порядке?

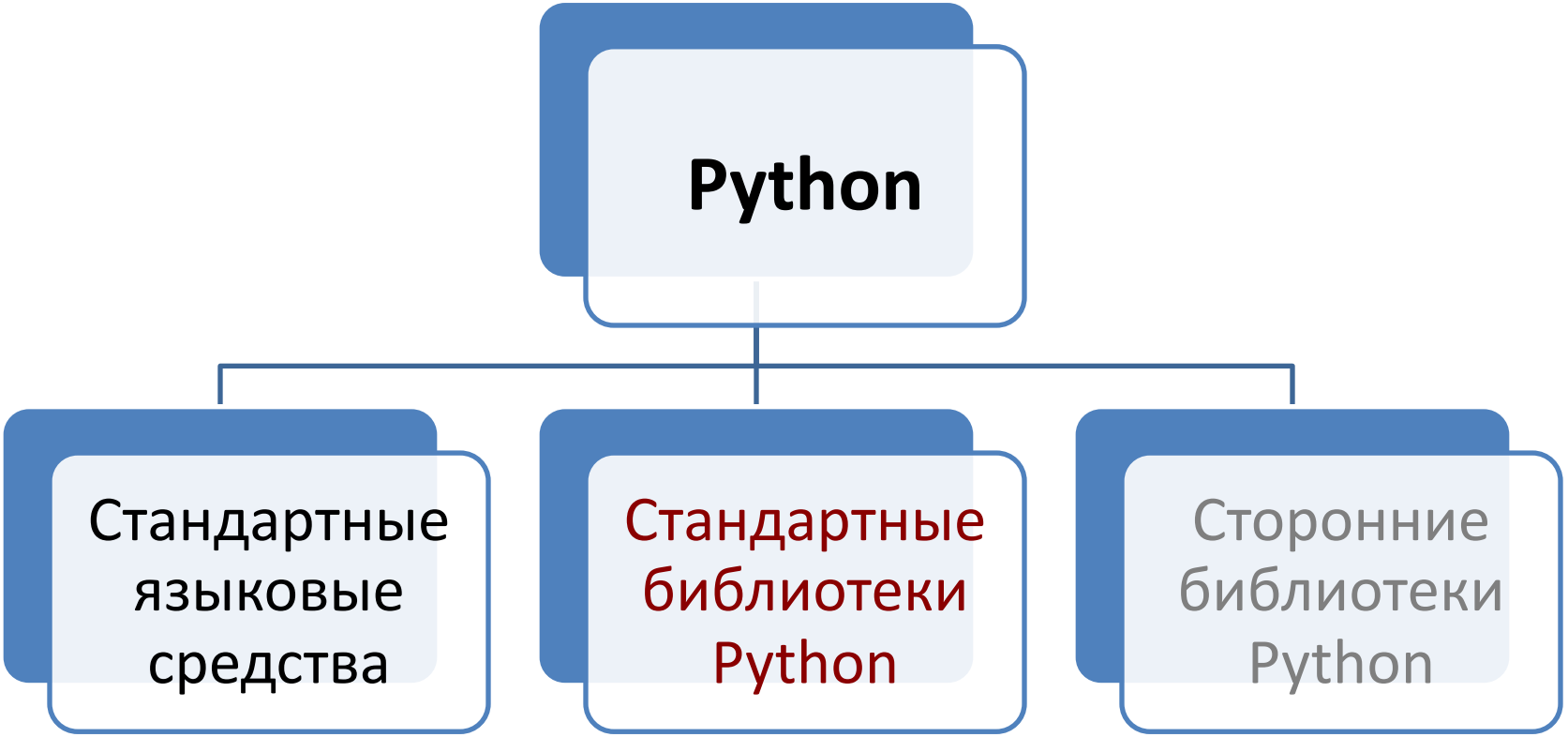
ДЛИНА МНОЖЕСТВА

Какова длина множества number_set?

```
>>> number_set = {1, 2, 3, 3, 4, 5}
>>> len(number_set)
```

???

ДОСТУПНЫЕ СРЕДСТВА PYTHON. КЛАССИФИКАЦИЯ



СТАНДАРТНЫЕ БИБЛИОТЕКИ PYTHON. СВОЙСТВА

- Стандартные библиотеки Python:
  - Предоставляют широкий спектр функций без необходимости установки пакетов
  - Включают средства для работы с разными типами данных, файлами, взаимодействия с ОС, поддержки сетевых протоколов и т.д.
- Базовая документация: https://docs.python.org/3/library
- Подключаются с помощью import <имя модуля>

ДОПОЛНИТЕЛЬНЫЕ ТИПЫ ДАННЫХ
РАБОТА СО СТРОКАМИ

Модуль	Описание и возможное использование
string	Сравнение со строковыми константами, например, digits, whitespace, punctuation
re	Поиск и замена текста с использованием регулярных выражений
struct	Работа с двоичными данными. Чтение и запись структурированных данных в файлы
difflib	Вычисление расхождений, поиск различий между строками и последовательностями, создание файлов различий и исправлений
textwrap	Перенос и заполнение текста, форматирование текста посредством разбиения строк или добавления пробелов

ДОПОЛНИТЕЛЬНЫЕ ТИПЫ ДАННЫХ
РАБОТА С ДАТОЙ, ВРЕМЕНЕМ. КОНСТАНТЫ

Модуль	Описание и возможное использование
datetime	Работа с датой и временем
time	Доступ к системному времени и преобразования времени
calendar	Работа с календарем
enum	Создание перечислений, связывающих символические имена с константами
pprint	Структурированная печать данных

РАБОТА С ДАТАМИ И ВРЕМЕНЕМ. МОДУЛЬ DATETIME

- Используемые типы данных:
  - date, datetime и time – для хранения даты, даты и времени, только времени
  - timedelta – для выполнения арифметических действий с датами и временем
- Используемые методы:
  - today(), now(), date(), time() – получение дат или времени
  - strptime(), strftime() – форматирование дат по шаблону (см. http://strftime.org)
- Базовая документация: https://docs.python.org/3/library/datetime.html
- Пример: см. функцию get_date() в console.py
- Еще примеры: https://pythonworld.ru/moduli/modul-datetime.html, https://metanit.com/python/tutorial/8.1.php

ДОПОЛНИТЕЛЬНЫЕ ТИПЫ ДАННЫХ
РАБОТА С КОЛЛЕКЦИЯМИ

Модуль	Описание и возможное использование
collections	Дополнительные контейнерные типы DefaultDict, OrderedDict, namedtuple, Counter, deque, …
array	Эффективная реализация типизированных массивов с числовыми элементами
copy	Операции глубокого и поверхностного копирования данных
typing	Поддержка разметки кода рекомендациями относительно типа объектов, особенно параметров функций и возвращаемых значений
types	Словари «только для чтения» (MappingProxyType) и атрибутный доступ по ключу в словаре (SimpleNamespace)

КОЛЛЕКЦИОННЫЕ ТИПЫ ДАННЫХ. КАКИЕ ТИПЫ ДАННЫХ ИСПОЛЬЗОВАТЬ?

- Всего несколько полей – 2 или 3?
  - Порядок следования полей легко запоминается
  - Имена полей излишни
  - ➢ **Используем обыкновенный кортеж**

- Нужны неизменяемые поля?
- Нужно избавиться от индексов полей, т.к. их сложно запоминать?
  - ➢ **Используем collections.namedtuple или typing.NamedTuple**

КОЛЛЕКЦИОННЫЕ ТИПЫ ДАННЫХ. КАКИЕ ТИПЫ ДАННЫХ ИСПОЛЬЗОВАТЬ?

- Не хотим усложнять?
  - Нужен удобный, «массиво-подобный» синтаксис
  - Нужна сериализация в JSON
  - ➢ **Используем обыкновенный словарь**

- Нужен полный контроль над собственной структурой данных?
  - ➢ **Делаем собственный класс с методами доступа и @property**

КОЛЛЕКЦИОННЫЕ ТИПЫ ДАННЫХ. КАКИЕ ТИПЫ ДАННЫХ ИСПОЛЬЗОВАТЬ?

- Нужно добавить в объект поведение (методы)?
  - **Написать собственный класс с методами** или
  - **Расширить collections.namedtuple / typing.NamedTuple**

- Нужно плотно упаковать данные?
  - Важна передача данные по сети или компактное хранение на диске
  - **Используем struct.Struct**

ДОПОЛНИТЕЛЬНЫЕ ТИПЫ ДАННЫХ
ЧИСЛОВЫЕ И МАТЕМАТИЧЕСКИЕ МОДУЛИ

Модуль	Описание и возможное использование
math, cmath	Математические операции для работы с вещественными и комплексными числами
decimal	Точные операции для вещественных десятичных чисел
statistics	Функции для вычисления величин из области математической статистики
fractions	Рациональные числа
random	Генерация псевдослучайных чисел и вариантов
functools	Функции более высокого порядка и операции с вызываемыми объектами
operator	Стандартные операторы как функции

РАБОТА С ФАЙЛАМИ В СТИЛЕ ОС

Модуль	Описание и возможное использование
os.path	Стандартные манипуляции с полными именами файлов
pathlib	Объектно-ориентированная работа с полными именами файлов
fileinput	Перебор строк из разных входных потоков
filecmp	Сравнение файлов и каталогов
tempfile	Генерирование временных файлов и каталогов
glob, fnmatch	Работа с путями и именами файлов с использованием шаблонов в стиле UNIX
shutil	Высокоуровневые операции с файлами

РАБОТА С ФАЙЛАМИ
ХРАНЕНИЕ, ОБРАБОТКА И РАЗНЫЕ ФОРМАТЫ

Модуль	Описание / Использование
linecache	Произвольный доступ к строкам из текстового файла с кэшированием
pickle, shelve	Сериализация и хранение объектов Python в бинарных файлах собственного формата
sqlite3	Работа с локальными базами данных SQLite
zlib, gzip, bz2, zipfile, tarfile	Работа с архивными файлами и сжатием данных
csv	Чтение и запись файлов в формате CSV
configparser	Разбор конфигурационных файлов; чтение/запись ini-файлов в стиле Windows

РАБОТА С ОС
СЕРВИСНЫЕ ФУНКЦИИ ОС

Модуль	Описание и возможное использование
os	Различные интерфейсы операционной системы
io	Основные средства для работы с потоками
otpparse	Мощный парсер параметров командой строки
logging	Средства журналирования событий
platform	Работа с идентификационными данными платформы
ctypes	Библиотека для работы с внешними функциями
threading	Высокоуровневый интерфейс программных потоков
subprocess	Управление подпроцессами

ИНТЕРНЕТ ПРОТОКОЛЫ И ФОРМАТЫ
СЕРВИСНЫЕ МОДУЛИ

Модуль	Описание и возможное использование
socket, ssl	Низкоуровневые сетевые интерфейсы и обертка SSL для объектов сокетов
email	Пакет для работы с электронной почтой и MIME
json	Кодирование и декодирование формата JSON
mailbox	Работа с почтовыми ящиками в разных форматах
html.parser	Разбор HTML
base64, binhex, binascii, uu	Кодирование / декодирование файлов или потоков с различными кодировками
htpp.server	Серверы HTTP

СРЕДСТВА РАЗРАБОТКИ И ОТЛАДКИ
ВСПОМОГАТЕЛЬНЫЕ МОДУЛИ ПРОГРАММИСТА

Модуль	Описание и возможное использование
pydoc	Генератор документации и электронной справки
doctest	Тестирование интерактивных примеров Python
unittest	Инфраструктура модульного тестирования
pdb	Консольный отладчик Python
cProfile	Профилировщик Python
timeit	Измерение времени выполнения небольших фрагментов кода
sys	Системные параметры и функции
__future__	Определения будущих команд – возможностей, которые будут добавлены в Python

ПРОЕКТ «РОССИЙСКАЯ ИСТОРИЯ»

DEADLINE = 30.01.2021

- Рассмотрите файл russian_history.json с описанием событий из истории России
- Загрузите данные из этого файла в Python и определите структуру полученных данных
- Напишите программу для проверки знаний Российской истории у школьников. Требования:
  - Вопросы задаются в случайном порядке – при каждом запуске разные и неповторяющиеся для сеанса работы
  - Количество задаваемых вопросов для каждого сеанса работы программы должно быть ограничено
  - Используйте свой механизм расчета результатов тестирования в баллах, исходя из структуры данных
- Для запроса даты исторических событий по шаблону необходимо использовать функцию get_date() из модуля console.py

СПАСИБО ЗА ВНИМАНИЕ !

ВОПРОСЫ ?

School of Computer Science