## 3.3. Листинги

|  |
| --- |
| Доработаем функцию деления из проекта “калькулятор” добавив обработку исключения |
| def divide(x, y):  if y == 0:  return "Ошибка. Деление на ноль"  else:  return x / y |
| def divide(x, y):  try:  return x / y  except ZeroDivisionError:  return "Ошибка. Деление на ноль" |
| В проекте про телефонный справочник также можно изменить функцию удаления контакта. |
| def delete\_contact():  name = input("Введите имя для удаления")  if name in contacts:  del contacts[name]  print(contacts)  print("Контакт успешно удален")  else:  print("Контакт не найден") |
| def delete\_contact():  name = input("Введите имя для удаления: ")  try:  del contacts[name]  print("Контакт успешно удален")  print(contacts)  except KeyError:  print("Контакт не найден") |
| t = {  "кошка": "cat",  "собака": "dog",  "книга": "book",  "яблоко": "apple",  "солнце": "sun",  "вода": "water",  "дерево": "tree",  "цветок": "flower",  "машина": "car",  "дом": "house",  # Английские слова  "cat": "кошка",  "dog": "собака",  "book": "книга",  "apple": "яблоко",  "sun": "солнце",  "water": "вода",  "tree": "дерево",  "flower": "цветок",  "car": "машина",  "house": "дом"  }  # Ввод слова пользователем  w = input("Введите слово: ").lower()  # Поиск перевода и вывод  if w in t:  print(f"Перевод слова '{w}': {t[w]}")  else:  print("Слово не найдено в словаре.") |
| try:  # Ввод слова пользователем  w = input("Введите слово: ").lower()  # Поиск перевода и вывод  print(f"Перевод слова '{w}': {t[w]}")  except KeyError:  print("Слово не найдено в словаре.") |
| Проект “Бронирование мест”. Список мест будет в виде списка кортежей.  [('Б1', 'свободно'), ('Б2', 'свободно')] |
| # Инициализация списка мест с использованием спискового включения  seats = [(f"Б{i}", 'свободно') for i in range(1, 10)]  print(seats) |
| # Отображение итогового состояния бронирования  print("Итоговое состояние бронирования мест:")  for seat in seats:  print(f"{seat[0]}: {seat[1]}") |
| def book\_seat(seats):  seat\_name = input("Введите место для бронирования (от Б1 до Б9): ")  try:  # Находим индекс места, которое пользователь хочет забронировать  i = seats.index((seat\_name, 'свободно'))  # Отмечаем место как забронированное  seats[i] = (seat\_name, 'забронировано')  print(f"Место '{seat\_name}' успешно забронировано.")  except ValueError:  print(f"Место '{seat\_name}' уже забронировано или не существует.") |
| # Запуск функции бронирования с возможностью повторного ввода  while True:  book\_seat(seats)  booking = input("Хотите забронировать еще одно место? (да/нет): ")  if booking.lower() != 'да':  break |
| Добавление еще одного except для непредвиденной ошибки  except ValueError:  print(f"Место '{seat\_name}' уже забронировано или не существует.")  except Exception as e:  print(f"Произошла неожиданная ошибка: {e}") |
| # Полный код проекта  def book\_seat(seats):  seat\_name = input("Введите место для бронирования (от Б1 до Б9): ")  try:  # Находим индекс места, которое пользователь хочет забронировать  i = seats.index((seat\_name, 'свободно'))  # Отмечаем место как забронированное  seats[i] = (seat\_name, 'забронировано')  print(f"Место '{seat\_name}' успешно забронировано.")  except ValueError:  print(f"Место '{seat\_name}' уже забронировано или не существует.")  # Инициализация списка мест с использованием спискового включения  seats = [(f"Б{i}", 'свободно') for i in range(1, 10)]  print(seats)  # Запуск функции бронирования с возможностью повторного ввода  while True:  book\_seat(seats)  booking = input("Хотите забронировать еще одно место? (да/нет): ")  if booking.lower() != 'да':  break  # Отображение итогового состояния бронирования  print("Итоговое состояние бронирования мест:")  for seat in seats:  print(f"{seat[0]}: {seat[1]}") |
| Работа с файлами |
| file = open('proba.txt')  content = file.read()  print(content)  file.close() |
| Traceback (most recent call last):  File "C:/Py/Scripts5/File\_\_01.py", line 1, in <module>  file = open('proba.txt', 'r')  FileNotFoundError: [Errno 2] No such file or directory: 'proba.txt' |
| Создаем в папке со скриптами файл proba.txt и ошибка пропадет,  но программа ничего не выведет, так как файл пустой. |
| Добавим в файл proba.txt фразу “This is the content of the file.” |
| Теперь программа выведет содержимое файла на экран.  “This is the content of the file.” |
| Изменим содержимое файла на “Это содержимое файла”. Программа выведет абракадабру. Р­С‚Рѕ СЃРѕРґРµСЂР¶РёРјРѕРµ С„Р°Р№Р»Р°. |
| Чтобы это не происходило надо указать кодировку файла. |
| # Открываем файл с указанием кодировки UTF-8  file = open('proba.txt', encoding='utf-8')  content = file.read()  print(content)  file.close()  # Видим нормальный текст |
| # Изменяем кодировку  file = open('proba2.txt', encoding='windows-1251')  content = file.read()  print(content)  file.close()  # Снова видим абракадабру |
|  |
| Изменим имя файла и опять получим ошибку.  Давайте использовать try except для обработки этой ошибки. |
| try:  file = open('proba3.txt', encoding='UTF-8')  content = file.read()  print(content)  file.close()  except FileNotFoundError:  print("Файл не найден.")  # Теперь ошибки нет - пишет что файл не найден. |
| Более правильный способ - это использование with open |
| try:  with open('proba2.txt', encoding='UTF-8') as file:  content = file.read()  print(content)  except FileNotFoundError:  print("Файл не найден.") |
| Попробуем открыть файл и записать в него что-нибудь. |
| file = open('proba.txt')  file.write("Некий текст")  file.close() |
| Traceback (most recent call last):  File "C:/Py/Scripts5/File\_\_03.py", line 2, in <module>  file.write("Некий текст")  io.UnsupportedOperation: not writable |
| Чтобы исправить ошибку надо открыть файл для записи.  file = open('proba.txt', 'w')  file.write("Некий текст")  file.close() |
| Запрещаем запись в файл и тестируем. |
| Traceback (most recent call last):  File "C:/Py/Scripts5/File\_\_03.py", line 1, in <module>  file = open('proba.txt', 'w')  PermissionError: [Errno 13] Permission denied: 'proba.txt' |
| Более правильный способ - это использование with open |
| try:  with open('proba.txt', 'w') as file:  file.write("Некий текст.")  except IOError:  print("Ошибка ввода-вывода.") |
| Попробуем открыть файл указав неправильную кодировку |
| with open('proba2.txt', encoding='ascii') as file:  content = file.read()  print(content) |
| UnicodeDecodeError: 'ascii' codec can't decode byte 0xcd in position 0: ordinal not in range(128) |
| try:  with open('proba2.txt', encoding='ascii') as file:  content = file.read()  print(content)  except UnicodeDecodeError:  print("Ошибка декодирования.") |
| try:  with open(proba2.txt', 'w') as file:  file.write("Некий текст.")  except IOError:  print("Ошибка ввода-вывода.")  except OSError:  print("Ошибка операционной системы.")  except UnicodeEncodeError:  print("Ошибка кодирования текста.")  except Exception as e:  print(f"Неизвестная ошибка: {e}") |
| Подсчет количества слов в файле |
| def count(file\_name):  try:  with open(file\_name, encoding='utf-8') as file:  content = file.read()  words = content.split()  return len(words)  except FileNotFoundError:  print("Файл не найден")  except Exception as e:  print(f"Произошла ошибка: {e}")  words = count("proba2.txt")  print(f"Количество слов в файле: {words}") |
| Подсчет количества событий в логе |
| def analyze\_log(file\_name):  counter = {"INFO": 0, "ERROR": 0, "WARNING": 0}  try:  with open(file\_name) as file:  for line in file:  if "[INFO]" in line:  counter["INFO"] += 1  elif "[ERROR]" in line:  counter["ERROR"] += 1  elif "[WARNING]" in line:  counter["WARNING"] += 1  for event, count in counter.items():  print(f"{event}: {count}")  except FileNotFoundError:  print(f"Файл '{file\_name}' не найден.")  except Exception as e:  print(f"Произошла ошибка при чтении файла: {e}")  analyze\_log('log.txt') |