

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ   
РОССИЙСКОЙ ФЕДЕРАЦИИ

**ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ**

**ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«ДОНСКОЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ» (ДГТУ)**

Факультет: Информатика и вычислительная техника

Кафедра: Кибербезопасность информационных систем

**Лабораторная работ №7**

Выполнил обучающийся гр.ВКБ21

Юровского Никиты Владимировича

(Фамилия, Имя, Отчество)

Проверила:

Ст.Пр. Жданова Марина Михайловна

(должность, Фамилия, Имя, Отчество)

Ростов-на-Дону

2024

**Лабораторная работа №7**

**Тема: «СПИСКИ И СЛОВАРИ»**

Вариант 2

1. Пусть список студентов представлен в виде структуры [[No, ФИО, Возраст, Группа],[No, ФИО, Возраст, Группа],[No, ФИО, Возраст, Группа]]. Преобразуйте список в словарь вида: {No: [ФИО, Возраст, Группа], No: [....], No: [....]}

Решение:

students = [  
 [1, "Иванов Иван Иванович", 20, "Группа A"],  
 [2, "Петров Петр Петрович", 21, "Группа B"],  
 [3, "Сидоров Сидор Сидорович", 22, "Группа C"]  
]  
  
students\_dict = {student[0]: student[1:] for student in students}  
  
print(students\_dict)

1. Добавьте к задаче №1 для словаря возможность (без преобразования словаря обратно в список):
   1. Увеличить возраст конкретного студента на 1. Поиск по «ФИО» («ФИО» студента необходимо ввести с клавиатуры).
   2. Изменить «ФИО» студента. Поиск по «ФИО» (старое и новое «ФИО» студента необходимо ввести с клавиатуры).
   3. Увеличить возраст конкретного студента на 1. Поиск по «No» («No» студента необходимо ввести с клавиатуры).
   4. Изменить группу студента. Поиск по «ФИО» («ФИО» студента и новый номер группы необходимо ввести с клавиатуры).
   5. Удалить запись о студенте. Поиск по «No» («No» студента, которого нужно удалить из списка, задается с клавиатуры)
   6. Если возраст студента больше 22 уменьшить его на 1.
   7. Если возраст студента равен 23, удалить его из списка.
   8. У всех студентов с фамилией «Иванов» увеличить возраст на 1.
   9. У студентов с фамилией «Иванов» изменить фамилию на «Сидоров».
   10. 10.Поменять «ФИО» и «Группа» местами.
2. Добавьте к пользовательскому интерфейсу из задачи №2 возможность вывода из словаря (без его преобразования в список):
   1. Списка студентов (а также информацию о них) группы 'БО-111111'.
   2. Списка студентов (а также информацию о них) с номерами 1-10.
   3. Списка студентов (а также информацию о них) в возрасте 22 лет.
   4. Список студентов (а также информацию о них) с фамилией 'Иванов'.
   5. Списка студентов (а также информацию о них), чьи фамилии заканчиваются на «а».
   6. Списка студентов (а также информацию о них), чей возраст – четное число.
   7. Списка студентов (а также информацию о них), если в возрасте студента встречается число 5.

Списка студентов (а также информацию о них), если их номера группы длиннее 7 символов.

from tkinter import \*  
import tkinter as tk  
from tkinter import messagebox, simpledialog  
  
# Исходный список студентов  
students\_list = [  
 [1, "Иванов Иван Иванович", 20, "БО-111111"],  
 [2, "Петров Петр Петрович", 21, "БО-111111"],  
 [3, "Сидоров Сидор Сидорович", 22, "БО-222222"],  
 [4, "Иванова Мария Ивановна", 23, "БО-222222"],  
 [5, "Иванов Алексей Иванович", 22, "БО-33333"]  
]  
  
# Преобразование списка в словарь для удобного доступа к данным студентов  
students\_dict = {student[0]: student[1:] for student in students\_list}  
  
  
def update\_student\_info():  
 # Обновляем информацию о студентах в метке  
 info = "Студенты:\n"  
 for key, value in students\_dict.items():  
 info += f"No: {key}, ФИО: {value[0]}, Возраст: {value[1]}, Группа: {value[2]}\n"  
 students\_info\_label.config(text=info)  
  
  
def increase\_age\_by\_name():  
 # Запрашиваем ФИО студента для увеличения возраста  
 full\_name = simpledialog.askstring("Увеличить возраст", "Введите ФИО студента:")  
 if full\_name:  
 found = False  
 for key, value in students\_dict.items():  
 if value[0] == full\_name:  
 value[1] += 1 # Увеличиваем возраст на 1 год  
 found = True  
 break  
 if found:  
 messagebox.showinfo("Успех", f"Возраст студента {full\_name} увеличен на 1 год.")  
 update\_student\_info()  
 else:  
 messagebox.showerror("Ошибка", f"Студент с ФИО '{full\_name}' не найден.")  
  
  
def change\_name():  
 # Изменяем ФИО студента  
 old\_name = simpledialog.askstring("Изменить ФИО", "Введите старое ФИО студента:")  
 new\_name = simpledialog.askstring("Изменить ФИО", "Введите новое ФИО студента:")  
 if old\_name and new\_name:  
 found = False  
 for key, value in students\_dict.items():  
 if value[0] == old\_name:  
 value[0] = new\_name # Меняем старое ФИО на новое  
 found = True  
 break  
 if found:  
 messagebox.showinfo("Успех", f"ФИО студента изменено с '{old\_name}' на '{new\_name}'.")  
 update\_student\_info()  
 else:  
 messagebox.showerror("Ошибка", f"Студент с ФИО '{old\_name}' не найден.")  
  
  
def increase\_age\_by\_no():  
 # Увеличиваем возраст студента по его номеру  
 no = simpledialog.askinteger("Увеличить возраст", "Введите No студента:")  
 if no is not None and no in students\_dict:  
 students\_dict[no][1] += 1 # Увеличиваем возраст на 1 год  
 messagebox.showinfo("Успех", f"Возраст студента с No {no} увеличен на 1 год.")  
 update\_student\_info()  
 else:  
 messagebox.showerror("Ошибка", f"Студент с No '{no}' не найден.")  
  
  
def change\_group():  
 # Изменяем группу студента по его ФИО  
 full\_name = simpledialog.askstring("Изменить группу", "Введите ФИО студента:")  
 new\_group = simpledialog.askstring("Изменить группу", "Введите новую группу:")  
 if full\_name and new\_group:  
 found = False  
 for key, value in students\_dict.items():  
 if value[0] == full\_name:  
 value[2] = new\_group # Меняем группу на новую  
 found = True  
 break  
 if found:  
 messagebox.showinfo("Успех", f"Группа студента {full\_name} изменена на '{new\_group}'.")  
 update\_student\_info()  
 else:  
 messagebox.showerror("Ошибка", f"Студент с ФИО '{full\_name}' не найден.")  
  
  
def delete\_student\_by\_no():  
 # Удаляем запись о студенте по его номеру  
 no = simpledialog.askinteger("Удалить студента", "Введите No студента для удаления:")  
 if no is not None and no in students\_dict:  
 del students\_dict[no] # Удаляем студента из словаря  
 messagebox.showinfo("Успех", f"Студент с No {no} удален.")  
 update\_student\_info()  
 else:  
 messagebox.showerror("Ошибка", f"Студент с No '{no}' не найден.")  
  
  
def adjust\_age\_conditions():  
 # Корректируем возраст студентов по заданным условиям  
 for key in list(students\_dict.keys()):  
 age = students\_dict[key][1]  
 if age > 22:  
 students\_dict[key][1] -= 1 # Уменьшаем возраст на 1 год для студентов старше 22 лет  
 elif age == 23:  
 del students\_dict[key] # Удаляем студентов в возрасте 23 лет  
 messagebox.showinfo("Успех", "Возраст студентов скорректирован.")  
 update\_student\_info()  
  
  
def increase\_age\_ivanov():  
 # Увеличиваем возраст всех студентов с фамилией 'Иванов'  
 for key, value in students\_dict.items():  
 if value[0].startswith("Иванов"):  
 value[1] += 1 # Увеличиваем возраст на 1 год  
 messagebox.showinfo("Успех", "Возраст студентов с фамилией 'Иванов' увеличен на 1 год.")  
 update\_student\_info()  
  
  
def change\_surname\_ivanov():  
 # Изменяем фамилию всех студентов 'Иванов' на 'Сидоров'  
 for key, value in students\_dict.items():  
 if value[0].startswith("Иванов"):  
 value[0] = value[0].replace("Иванов", "Сидоров") # Заменяем фамилию  
 messagebox.showinfo("Успех", "Фамилия студентов 'Иванов' изменена на 'Сидоров'.")  
 update\_student\_info()  
  
  
def swap\_fio\_group():  
 # Меняем местами ФИО и группу у всех студентов  
 for key, value in students\_dict.items():  
 value[0], value[2] = value[2], value[0] # Меняем местами значения ФИО и группы  
 messagebox.showinfo("Успех", "ФИО и группа студентов поменяны местами.")  
 update\_student\_info()  
  
  
def filter\_students(condition):  
 # Фильтруем студентов по заданному условию и возвращаем отфильтрованный словарь  
 filtered\_students = {}  
 for key, value in students\_dict.items():  
 if condition(value):  
 filtered\_students[key] = value  
 return filtered\_students  
  
  
def show\_filtered\_students(condition):  
 # Отображаем отфильтрованных студентов согласно заданному условию  
 filtered\_students = filter\_students(condition)  
 info = "Отфильтрованные студенты:\n"  
  
 if not filtered\_students:  
 info += "Нет студентов по заданному критерию."  
  
 for key, value in filtered\_students.items():  
 info += f"No: {key}, ФИО: {value[0]}, Возраст: {value[1]}, Группа: {value[2]}\n"  
  
 filtered\_students\_label.config(text=info)  
  
  
# Создаем главное окно приложения  
root = tk.Tk()  
root.title("Управление студентами")  
root.attributes('-fullscreen', True)  
  
# Создаем фреймы для организации интерфейса приложения  
frame = Frame(borderwidth=1, relief=SOLID)  
frame.grid(column=0, row=0, padx=20, pady=20, ipady=30)  
frame1 = Frame(borderwidth=1, relief=SOLID)  
frame1.grid(column=1, row=0, padx=20, pady=20, sticky=N)  
frame2 = Frame(borderwidth=1, relief=SOLID)  
frame2.grid(sticky=NE, column=2, row=0, padx=20, pady=20)  
  
# Метка для отображения информации о студентах  
students\_info\_label = tk.Label(frame, text="", justify=tk.LEFT, font=('Arial', 14))  
students\_info\_label.grid(column=0, row=0, padx=20, pady=15)  
  
# Метка для отображения отфильтрованных студентов  
filtered\_students\_label = tk.Label(frame, text="", justify=tk.LEFT, font=('Arial', 14))  
filtered\_students\_label.grid(column=0, row=1, padx=20, pady=15)  
  
# Кнопки для различных операций над данными о студентах  
tk.Button(root, text="Закрыть окно", command=root.destroy, font=('Arial', 14), width=15).grid(column=0, row=2)  
tk.Button(frame1, text="Увеличить возраст по ФИО", width=23, command=increase\_age\_by\_name).grid(pady=5)  
tk.Button(frame1, text="Изменить ФИО студента", width=23, command=change\_name).grid(pady=5)  
tk.Button(frame1, text="Увеличить возраст по No", command=increase\_age\_by\_no, width=29).grid(pady=5)  
tk.Button(frame1, text="Изменить группу студента", command=change\_group, width=29).grid(pady=5)  
tk.Button(frame1, text="Удалить запись о студенте по No", command=delete\_student\_by\_no, width=29).grid(pady=5)  
tk.Button(frame1, text="Скорректировать возраст студентов", command=adjust\_age\_conditions, width=29).grid(pady=5)  
tk.Button(frame1, text="Увеличить возраст 'Иванов'", command=increase\_age\_ivanov, width=29).grid(pady=5)  
tk.Button(frame1, text="Изменить фамилию 'Иванов' на 'Сидоров'", command=change\_surname\_ivanov, width=36).grid(pady=5)  
tk.Button(frame1, text="Поменять ФИО и Группу местами", command=swap\_fio\_group, width=36).grid(pady=5, padx=5)  
  
# Кнопки для фильтрации студентов по различным критериям  
tk.Button(frame2, text="Студенты группы 'Группа A'",  
 command=lambda: show\_filtered\_students(lambda x: x[2] == 'Группа A'), width=31).grid(pady=5)  
tk.Button(frame2, text="Студенты с номерами 1-10", command=lambda: show\_filtered\_students(lambda x: x in range(1, 11)),  
 width=31).grid(pady=5)  
tk.Button(frame2, text="Студенты в возрасте 22 лет", command=lambda: show\_filtered\_students(lambda x: x[1] == 22),  
 width=31).grid(pady=5)  
tk.Button(frame2, text="Студенты с фамилией 'Иванов'",  
 command=lambda: show\_filtered\_students(lambda x: x[0].startswith('Иванов')), width=31).grid(pady=5)  
tk.Button(frame2, text="Студенты с фамилиями на 'а'",  
 command=lambda: show\_filtered\_students(lambda x: x[0].endswith('а')), width=31).grid(pady=5)  
tk.Button(frame2, text="Студенты с четным возрастом", command=lambda: show\_filtered\_students(lambda x: x[1] % 2 == 0),  
 width=31).grid(pady=5)  
tk.Button(frame2, text="Студенты с возрастом содержащим '5'",  
 command=lambda: show\_filtered\_students(lambda x: '5' in str(x[1])), width=31).grid(pady=5)  
tk.Button(frame2, text="Студенты с четным номером", command=lambda: show\_filtered\_students(lambda x: x % 2 == 0),  
 width=31).grid(pady=5)  
tk.Button(frame2, text="Студенты с длинной группой больше 7 символов",  
 command=lambda: show\_filtered\_students(lambda x: len(x[2]) > 7), width=43).grid(pady=5)  
tk.Button(frame2, text="Студенты с номером группы заканчивающимся на '1'",  
 command=lambda: show\_filtered\_students(lambda x: x[2].endswith('1')), width=43).grid(padx=5, pady=5)  
  
# Инициализируем отображение информации о студентах  
update\_student\_info()  
  
# Запускаем главный цикл приложения  
root.mainloop()