|  |  |  |
| --- | --- | --- |
| МИНОБРНАУКИ РОССИИ | | |
| Федеральное государственное автономное  образовательное учреждение высшего образования  «Пермский государственный национальный  исследовательский университет» | | |
|  | Институт компьютерных наук и технологий | |
| **ОТЧЁТ**  по индивидуальной работе №1  по дисциплине «Языки программирования»  Вариант 2 | | |
|  | | Работу выполнил  студент группы ИТ-3,4-2024 1 курса  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Цыгвинцев О.С.  «\_\_\_» \_\_\_\_\_\_\_\_ 2024 г. |
| Работу проверил  \_\_\_\_\_\_\_\_\_\_\_\_\_\_ Лабутин И.А.  «\_\_\_» \_\_\_\_\_\_\_\_\_ 2024 г. |
| Пермь 2025 | | |

СОДЕРЖАНИЕ

[Постановка задачи 3](#_Toc188306489)

[Алгоритм решения 3](#_Toc188306490)

[Тестирование 4](#_Toc188306491)

[Код программы 5](#_Toc188306492)

# Постановка задачи

Создать файл записей, в котором хранится информация о каждом из воспитанников детского сада: фамилия, имя, дата рождения (год, месяц, число), название группы (младшая средняя старшая), заключение о состоянии здоровья ("здоров" или "нуждается в лечении") каждого из четырёх специалистов - невропатолога, отоларинголога, ортопеда и окулиста.

Разработать и реализовать программу "Учёт результатов диспансеризации", которая считывает исходную информацию и позволяет на основе неё создавать следующие отчёты:

Полный список всех детей, который будет отсортирован по убыванию количества заключений со значением «здоров» (то есть, сначала идут дети, для которых все четыре специалиста дали данное заключение, затем – три и т.д.). Для детей, входящих в одну группу по состоянию здоровья, использовать второй ключ сортировки – фамилия ребёнка (по возрастанию).

Список всех детей заданной группы, отсортированный по следующему ключу: год рождения (по возрастанию), месяц рождения (по возрастанию), число (по возрастанию).

Список всех детей, нуждающихся в лечении по заключении хотя бы одного специалиста, отсортированный по следующему ключу: группа (по возрастанию) + фамилия (по возрастанию).

Создать базу детей, включающую не менее 25 записей и на основе неё сформировать все указанные списки. База должна содержать такие записи, чтобы во всех списках явно прослеживался заданный вид сортировки по всем ключам. Для сортировки записей использовать сортировку выбором.

# Алгоритм решения

Для решения поставленной задачи было выбрано решение использовать структуру данных в виде списка словарей, где каждый словарь представляет собой запись о ребёнке. Такой подход позволяет получать доступ к данным по ключу и хорошо подходит для хранения записей с множеством полей. Для хранения самих данных используется файл CSV (текстовый формат, предназначенный для представления табличных данных), так как задача не требует развёртывания полноценного сервера с использованием СУБД, а формат CSV обеспечивает простоту работы с данными и их лёгкую читаемость.

Алгоритм сортировки был поставлен в условии задачи и является сортировкой выбором. Несмотря на долгое время выполнения данного алгоритма (сложность O(n^2)), он хорошо подходит для небольших наборов данных. Его принцип действия заключается в выборе минимального элемента в списке и его обмен со значением первой неотсортированной позиции. Для каждого отчёта реализована данная сортировка по одному из критерия: по количеству заключений "здоров", по принадлежности к группе и по наличию необходимости лечения.

Интерфейс пользователя реализован в виде меню с выбором следующего действия. После выполнения действия пользователь возвращается обратно в меню с возможностью выйти и закончить выполнение программы. Выбранная архитектура решения обеспечивает модульность, понятность кода и удобство его поддержки. Каждая функция отвечает за конкретную операцию (добавление, удаление, редактирование, формирование отчетов), что делает код структурированным и легким для понимания.

# Тестирование

Тест вывода отчётов. При выборе действия с 1 по 3 (рис. 1) пользователю отображается отчёт с указанной в задании сортировкой.



Рисунок 1 – Варианты действия 1-3

Тест добавления записи. При выборе действия 4 (рис. 2) можно добавить новую запись.

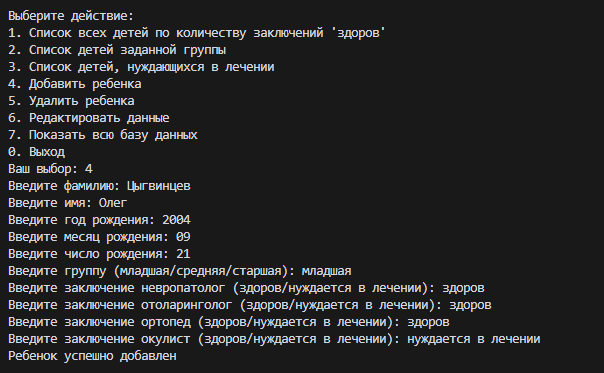


Рисунок 2 – Добавлении новой записи

После добавления записи, она создаётся в файле CSV (рис. 3).



Рисунок 3 – Добавленная запись

Если неправильно ввести значение группы или статус здоровья, то выйдет ошибка и запись не будет создана (рис. 4).



Рисунок 4 – Ошибка при неверном создании записи

Тест удаления записи. Удаление работает стабильно. Если ввести фамилию и имя несуществующего ребёнка, то программа выдаст ошибку (рис. 5).

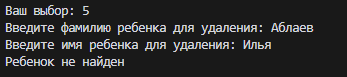


Рисунок 5 – Ошибка при удалении несуществующего ребёнка

Тест выбора несуществующего пункта меню. Если ввести действие, которого нет в списке, то программа выдаст ошибку «Неверный выбор» (рис. 6).

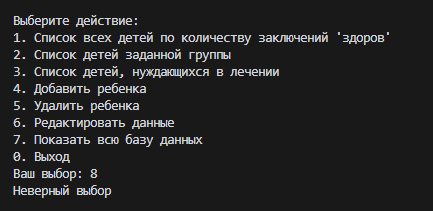


Рисунок 6 – Ошибка при выборе несуществующего пункта меню

# Код программы

import csv

from datetime import datetime

def read\_csv\_file(filename):

children = []

with open(filename, 'r', encoding='utf-8') as file:

reader = csv.DictReader(file)

for row in reader:

children.append(row)

return children

def count\_healthy(child):

doctors = ['невропатолог', 'отоларинголог', 'ортопед', 'окулист']

return sum(1 for doctor in doctors if child[doctor] == 'здоров')

def selection\_sort(arr, key\_func):

"""

Сортировка выбором с использованием функции ключа

"""

n = len(arr)

for i in range(n):

min\_idx = i

for j in range(i + 1, n):

if key\_func(arr[j]) < key\_func(arr[min\_idx]):

min\_idx = j

arr[i], arr[min\_idx] = arr[min\_idx], arr[i]

return arr

def report\_1(children):

"""Список детей по количеству заключений 'здоров'"""

# Добавляем количество "здоров" к каждому ребёнку

result = children.copy()

for child in result:

child['healthy\_count'] = count\_healthy(child)

# Сортировка по количеству "здоров" (убывание) и фамилии (возрастание)

return selection\_sort(result,

lambda x: (-x['healthy\_count'], x['фамилия']))

def report\_2(children, group):

"""Список детей заданной группы по дате рождения"""

# Фильтрация по группе

group\_children = [c for c in children if c['группа'] == group]

# Сортировка по дате рождения

return selection\_sort(group\_children,

lambda x: datetime(int(x['год']),

int(x['месяц']),

int(x['число'])))

def report\_3(children):

"""Список детей, нуждающихся в лечении"""

doctors = ['невропатолог', 'отоларинголог', 'ортопед', 'окулист']

needs\_treatment = []

for child in children:

if any(child[doctor] == 'нуждается в лечении' for doctor in doctors):

needs\_treatment.append(child)

# Сортировка по группе и фамилии

return selection\_sort(needs\_treatment,

lambda x: (x['группа'], x['фамилия']))

def save\_to\_csv(children, filename='childs.csv'):

"""Сохранение записи в CSV файл"""

if not children:

return

fieldnames = ['фамилия', 'имя', 'год', 'месяц', 'число', 'группа',

'невропатолог', 'отоларинголог', 'ортопед', 'окулист']

with open(filename, 'w', newline='', encoding='utf-8') as file:

writer = csv.DictWriter(file, fieldnames=fieldnames)

writer.writeheader()

for child in children:

# Создаем новый словарь

row = {field: child[field] for field in fieldnames}

writer.writerow(row)

def add\_child(children):

"""Добавление нового ребенка в базу"""

child = {

'фамилия': input("Введите фамилию: "),

'имя': input("Введите имя: "),

'год': input("Введите год рождения: "),

'месяц': input("Введите месяц рождения: "),

'число': input("Введите число рождения: "),

'группа': input("Введите группу (младшая/средняя/старшая): ")

}

if child['группа'] not in ['младшая', 'средняя', 'старшая']:

print("Неверная группа.")

return

doctors = ['невропатолог', 'отоларинголог', 'ортопед', 'окулист']

for doctor in doctors:

while True:

status = input(f"Введите заключение {doctor} (здоров/нуждается в лечении): ")

if status in ['здоров', 'нуждается в лечении']:

child[doctor] = status

break

print("Неверное значение! Попробуйте снова.")

children.append(child)

save\_to\_csv(children)

print("Ребенок успешно добавлен")

def remove\_child(children):

"""Удаление ребенка из базы"""

surname = input("Введите фамилию ребенка для удаления: ")

name = input("Введите имя ребенка для удаления: ")

for child in children[:]:

if child['фамилия'] == surname and child['имя'] == name:

children.remove(child)

save\_to\_csv(children)

print("Ребенок успешно удален.")

return

print("Ребенок не найден")

def edit\_child(children):

"""Редактирование данных ребенка"""

surname = input("Введите фамилию ребенка для редактирования: ")

name = input("Введите имя ребенка для редактирования: ")

for child in children:

if child['фамилия'] == surname and child['имя'] == name:

print("\nВыберите поле для редактирования:")

fields = ['фамилия', 'имя', 'год', 'месяц', 'число', 'группа',

'невропатолог', 'отоларинголог', 'ортопед', 'окулист']

for i, field in enumerate(fields, 1):

print(f"{i}. {field}")

choice = input("Ваш выбор: ")

if choice.isdigit():

idx = int(choice)

if 0 < idx <= 7:

field = fields[idx-1]

if field in ['невропатолог', 'отоларинголог', 'ортопед', 'окулист']:

while True:

value = input("Введите новое значение (здоров/нуждается в лечении): ")

if value in ['здоров', 'нуждается в лечении']:

child[field] = value

break

print("Неверное значение!")

else:

child[field] = input("Введите новое значение: ")

save\_to\_csv(children)

print("Данные успешно обновлены.")

return

print("Неверный выбор.")

return

print("Ребенок не найден.")

def print\_all\_children(children):

"""Вывод всей базы данных"""

if not children:

print("База данных пуста.")

return

for child in children:

print("\nИнформация о ребенке:")

for key, value in child.items():

print(f"{key}: {value}")

while True:

children = read\_csv\_file('childs.csv')

print("\nВыберите действие:")

print("1. Список всех детей по количеству заключений 'здоров'")

print("2. Список детей заданной группы")

print("3. Список детей, нуждающихся в лечении")

print("4. Добавить ребенка")

print("5. Удалить ребенка")

print("6. Редактировать данные")

print("7. Показать все записи о детях")

print("0. Выход")

choice = input("Ваш выбор: ")

if choice == '4':

add\_child(children)

elif choice == '5':

remove\_child(children)

elif choice == '6':

edit\_child(children)

elif choice == '7':

print\_all\_children(children)

elif choice == '1':

result = report\_1(children)

for child in result:

print(f"{child['фамилия']}: {child['healthy\_count']} специалистов 'здоров'")

elif choice == '2':

group = input("Введите группу (младшая/средняя/старшая): ")

if group not in ['младшая', 'средняя', 'старшая']:

print("Неверная группа.")

continue

result = report\_2(children, group)

for child in result:

date\_str = f"{child['число']}.{child['месяц']}.{child['год']}"

print(f"{child['фамилия']}, {date\_str}")

elif choice == '3':

result = report\_3(children)

for child in result:

print(f"Группа {child['группа']}, {child['фамилия']}")

elif choice == '0':

break

else:

print("Неверный выбор.")