|  |  |  |
| --- | --- | --- |
| Защищено:  Нардид А.Н.    "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |  | Демонстрация:  Нардид А.Н.  "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |

**Отчет по ДЗ по курсу**

**Парадигмы и конструкции языков программирования**

**ГУИМЦ**

#### Тема работы: "Лист задач в консольном окне. "

4

(количество листов)

|  |  |
| --- | --- |
| ИСПОЛНИТЕЛЬ: |  |
| студент группы ИУ5Ц-53Б | \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ |
|  | (подпись) |
| Рабинович А.Д. | "\_\_"\_\_\_\_\_\_\_\_\_\_\_\_\_2024 г. |

Москва, МГТУ - 2024 г.

# Тема и задание для выполнения лабораторной работы.

Тема работы: «Лист задач в консольном окне.»

**Задание:**

1. Разработать код, реализующий программу для управления списком задач.
2. Разобрать с основными функциями программы:
3. Запустить программу и выполните следующие действия:

Добавьте три задачи с описанием.

Просмотрите список задач.

Удалите одну из задач.

Сохраните оставшиеся задачи в файл.

Перезапустите программу и загрузите задачи из файла.

Убедитесь, что данные загрузились корректно.

# Листинг программы

# import json

# class Task:

# def \_\_init\_\_(self, description):

# self.description = description

# def \_\_str\_\_(self):

# return self.description

# class ToDoList:

# def \_\_init\_\_(self):

# self.tasks = []

# def add\_task(self, description):

# if not description.strip():

# print("Ошибка: описание задачи не может быть пустым.")

# return

# new\_task = Task(description.strip())

# self.tasks.append(new\_task)

# print(f"Задача '{description.strip()}' успешно добавлена.")

# def view\_tasks(self):

# if not self.tasks:

# print("Нет доступных задач.")

# else:

# print("Ваши задачи:")

# for idx, task in enumerate(self.tasks, start=1):

# print(f"{idx}. {task}")

# def delete\_task(self, task\_number):

# if 1 <= task\_number <= len(self.tasks):

# removed\_task = self.tasks.pop(task\_number - 1)

# print(f"Задача '{removed\_task}' успешно удалена.")

# else:

# print("Неверный номер задачи.")

# def save\_to\_file(self, filename):

# with open(filename, 'w') as file:

# json.dump([task.description for task in self.tasks], file)

# print(f"Задачи сохранены в файл {filename}.")

# def load\_from\_file(self, filename):

# try:

# with open(filename, 'r') as file:

# tasks = json.load(file)

# self.tasks = [Task(description) for description in tasks]

# print(f"Задачи загружены из файла {filename}.")

# except FileNotFoundError:

# print(f"Файл {filename} не найден.")

# # Main program

# def main():

# todo\_list = ToDoList()

# filename = "tasks.json"

# 

# while True:

# print("\nМеню:")

# print("1. Добавить задачу")

# print("2. Показать задачи")

# print("3. Удалить задачу")

# print("4. Сохранить задачи")

# print("5. Загрузить задачи")

# print("6. Выход")

# choice = input("Введите ваш выбор: ")

# 

# if choice == "1":

# description = input("Введите описание задачи: ")

# todo\_list.add\_task(description)

# elif choice == "2":

# todo\_list.view\_tasks()

# elif choice == "3":

# try:

# task\_number = int(input("Введите номер задачи для удаления: "))

# todo\_list.delete\_task(task\_number)

# except ValueError:

# print("Ошибка: пожалуйста, введите корректный номер.")

# elif choice == "4":

# todo\_list.save\_to\_file(filename)

# elif choice == "5":

# todo\_list.load\_from\_file(filename)

# elif choice == "6":

# print("Выход из программы. До свидания!")

# break

# else:

# print("Неверный выбор. Пожалуйста, попробуйте снова.")

# if \_\_name\_\_ == "\_\_main\_\_":

# main()

# Результаты работы программы

Меню:

1. Добавить задачу

2. Показать задачи

3. Удалить задачу

4. Сохранить задачи

5. Загрузить задачи

6. Выход

Введите ваш выбор: