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

**Отчет по лабораторной работе № 6 по курсу**

**Парадигмы и конструкции языков программирования**

**ГУИМЦ**

#### Тема работы: " Разработка простого бота для Telegram с использованием языка Python."

2

(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: "Разработка простого бота для Telegram с использованием языка Python.".

Разработайте простого бота для Telegram. Бот должен использовать функциональность создания кнопок.

# Листинг программы

# import logging

# from aiogram import Bot, Dispatcher, types

# from aiogram.dispatcher.filters import Command, Text

# from aiogram.utils import executor

# from aiogram.contrib.fsm\_storage.memory import MemoryStorage

# from aiogram.types import InlineKeyboardButton, InlineKeyboardMarkup, CallbackQuery

# # Set up logging

# logging.basicConfig(level=logging.INFO)

# # Initialize bot and dispatcher

# bot = Bot(token='')

# storage = MemoryStorage()

# dp = Dispatcher(bot, storage)

# masters={

# "Парикмахер": "Запись на услугу парикмахера",

# "Мастер маникюра": "Запись на услугу мастра маникюра",

# "Бровист": "Запись на услугу бровиста",

# "Косметолог": "Запись на услугу косметолога"

# }

# @dp.message\_handler(Command("start"))

# async def start(message: types.Message):

# await message.answer('Привет! Я бот салона красоты. Какие услуги вас интересуют?')

# keyboard = InlineKeyboardMarkup()

# for master in masters.keys():

# keyboard.add(InlineKeyboardButton(master, callback\_data=master))

# await message.answer("Выберите мастера:", reply\_markup=keyboard)

# @dp.callback\_query\_handler(lambda c: c.data in masters.keys())

# async def process\_callback(callback\_query: CallbackQuery):

# master\_name = callback\_query.data

# await bot.send\_message(callback\_query.from\_user.id, masters[master\_name])

# await callback\_query.answer()

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

# executor.start\_polling(dp, skip\_updates=True)

# Результаты работы программы

