1. Подключение библиотеки Otus.ToDoList.ConsoleBot
   * Добавить к себе в решение и в зависимости к своему проекту с ботом проект Otus.ToDoList.ConsoleBot [GitHub](https://github.com/OTUS-NET/C-Sharp-Basic/tree/main/Homeworks/5%20%D0%9E%D0%9E%D0%9F%20%D0%BA%D0%BB%D0%B0%D1%81%D1%81%D1%8B%20%D0%B8%20%D0%B8%D0%BD%D1%82%D0%B5%D1%80%D1%84%D0%B5%D0%B9%D1%81%D1%8B/Otus.ToDoList.ConsoleBot" \o "GitHub" \t "_blank)
   * Ознакомиться с классами в папке Types и с README.md
   * Создать класс UpdateHandler, который реализует интерфейс IUpdateHandler, и перенести в метод HandleUpdateAsync обработку всех команд. Вместо Console.WriteLine использовать SendMessage у ITelegramBotClient
   * Перенести try/catch в HandleUpdateAsync. В Main оставить catch(Exception)
   * Для вывода в коноль сообщений использовать метод ITelegramBotClient.SendMessage
   * Код библиотеки Otus.ToDoList.ConsoleBot не нужно изменять
2. Удалить команду /echo
3. Изменение логики команды /start
   * Не нужно запрашивать имя
   * Добавить класс ToDoUser
     + Свойства
       - Guid UserId
       - long TelegramUserId
       - string TelegramUserName
       - DateTime RegisteredAt
4. Добавление класса сервиса UserService
   * Добавить интерфейс IUserService
   * interface IUserService
   * {
   * ToDoUser RegisterUser(long telegramUserId, string telegramUserName);
   * ToDoUser? GetUser(long telegramUserId);
   * }
   * Создать класс UserService, который реализует интерфейс IUserService. Заполнять telegramUserId и telegramUserName нужно из значений Update.Message.From
   * Добавить использование IUserService в UpdateHandler. Получать IUserService нужно через конструктор
   * При команде /start нужно вызвать метод IUserService.RegisterUser.
   * Если пользователь не зарегистрирован, то ему доступны только команды /help /info
5. Добавление класса ToDoItem
   * Добавить enum ToDoItemState с двумя значениями
     + Active
     + Completed
   * Добавить класс ToDoItem
     + Свойства
       - Guid Id
       - ToDoUser User
       - string Name
       - DateTime CreatedAt
       - ToDoItemState State
       - DateTime? StateChangedAt
   * Добавить использование класса ToDoItem вместо хранения только имени задачи
6. Изменение логики /showtasks
   * Выводить только задачи с ToDoItemState.Active
   * Добавить вывод CreatedAt и Id. Пример: Имя задачи - 01.01.2025 00:00:00 - 17056344-0e03-4a21-b0dd-f0d30a5abf49
7. Добавление класса сервиса ToDoService
   * Добавить интерфейс IToDoService
   * public interface IToDoService
   * {
   * IReadOnlyList<ToDoItem> GetAllByUserId(Guid userId);
   * //Возвращает ToDoItem для UserId со статусом Active
   * IReadOnlyList<ToDoItem> GetActiveByUserId(Guid userId);
   * ToDoItem Add(ToDoUser user, string name);
   * void MarkCompleted(Guid id);
   * void Delete(Guid id);
   * }
   * Создать класс ToDoService, который реализует интерфейс IToDoService. Перенести в него логику обработки команд. Проверки на максимальное количество задач, на максимальную длину задачи и на дубликаты тоже нужно перенести в ToDoService.
   * Добавить использование IToDoService в UpdateHandler. Получать IToDoService нужно через конструктор
   * Изменить формат обработки команды /addtask. Нужно сразу передавать имя задачи в команде. Пример: /addtask Новая задача
   * Изменить формат обработки команды /removetask. Нужно сразу передавать номер задачи в команде. Пример: /removetask 2
8. Добавление команды /completetask
   * Добавить обработку новой команды /completetask. При обработке команды использовать метод IToDoService.MarkAsCompleted
   * Пример: /completetask 73c7940a-ca8c-4327-8a15-9119bffd1d5e
9. Добавление команды /showalltasks
   * Добавить обработку новой команды /showalltasks. По ней выводить команды с любым State и добавить State в вывод
   * Пример: (Active) Имя задачи - 01.01.2025 00:00:00 - ffbfe448-4b39-4778-98aa-1aed98f7eed8
10. Обновить /help

Примечание: Можно заменить catch с разными типами исключений, если в них нет кастомной логики, на один catch(Exception ex). Так как в предыдущем задание сatch с разными типами исключений добавлялись в учебных целям и в реальных проектах не нужно делать catch на каждый тип исключения, если в них нет специальной логики.