Группа: ИУ5-31Б
Студент: До Тхе Фу

Постановка задачи

1. Разработайте бота для Telegram. Бот позволяет пользователю выбирать категории продуктов, а затем получать информацию о выбранных продуктах.

Текст программы

```
from telegram import Update, InlineKeyboardButton, InlineKeyboardMarkup
from telegram.ext import ApplicationBuilder, CommandHandler, CallbackQueryHandler, ContextTypes

# Данные о продуктах
PRODUCTS = {
    "Мясо": {
        "Курица": {"Калории": 165, "Белки": 31, "Жиры": 3.6, "Углеводы": 0},
        "Свинина": {"Калории": 242, "Белки": 27, "Жиры": 14, "Углеводы": 0},
        "Говядина": {"Калории": 250, "Белки": 26, "Жиры": 15, "Углеводы": 0},
    },
    "Овощи": {
        "Морковь": {"Калории": 41, "Белки": 0.9, "Жиры": 0.2, "Углеводы": 9.6},
        "Картофель": {"Калории": 77, "Белки": 2, "Жиры": 0.1, "Углеводы": 17},
        "Брокколи": {"Калории": 34, "Белки": 2.8, "Жиры": 0.4, "Углеводы": 6.6},
    },
    "Молочные продукты": {
        "Молоко 2%": {"Калории": 50, "Белки": 3.4, "Жиры": 2, "Углеводы": 5},
        "Йогурт": {"Калории": 59, "Белки": 10, "Жиры": 0.4, "Углеводы": 3.6},
        "Сыр": {"Калории": 402, "Белки": 25, "Жиры": 33, "Углеводы": 1.3},
    },
    "Фрукты": {
        "Яблоко": {"Калории": 52, "Белки": 0.3, "Жиры": 0.2, "Углеводы": 14},
        "Банан": {"Калории": 89, "Белки": 1.1, "Жиры": 0.3, "Углеводы": 23},
        "Апельсин": {"Калории": 47, "Белки": 0.9, "Жиры": 0.1, "Углеводы": 12},
    },
}

# Команда /start
async def start(update: Update, context: ContextTypes.DEFAULT_TYPE):
    # Кнопки для категорий
```

```
    keyboard = [
        [InlineKeyboardButton(category, callback_data=category)] for category in PRODUCTS
    ]
    reply_markup = InlineKeyboardMarkup(keyboard)
    await update.message.reply_text("Привет! Выберите категорию:", reply_markup=reply_markup)

# Обработчик выбора категории
async def category_callback(update: Update, context: ContextTypes.DEFAULT_TYPE):
    query = update.callback_query
    await query.answer()

    category = query.data  # Получаем выбранную категорию
    if category in PRODUCTS:
        # Сохраняем категорию в контексте
        context.user_data['current_category'] = category

        # Создаем кнопки для продуктов + кнопку "Назад"
        keyboard = [
            [InlineKeyboardButton(product, callback_data=f"product_{product}") for product in PRODUCTS[category]]
        ]
        keyboard.append([InlineKeyboardButton("← Назад к категориям", callback_data="BACK_TO_CATEGORIES")])
        reply_markup = InlineKeyboardMarkup(keyboard)

        await query.edit_message_text(
            text=f"Вы выбрали категорию: {category}\nВыберите продукт:",
            reply_markup=reply_markup
        )

# Обработчик выбора продукта
async def product_callback(update: Update, context: ContextTypes.DEFAULT_TYPE):
    query = update.callback_query
    await query.answer()

    product_name = query.data.replace("product_", "")  # Получаем название продукта
    for category, items in PRODUCTS.items():
        if product_name in items:
            product = items[product_name]
            details = "\n".join(f"{key}: {value}" for key, value in product.items())
            # Добавляем кнопку "Назад"
            keyboard = [[InlineKeyboardButton("← Назад к категории", callback_data="BACK_TO_CATEGORY")]]
            reply_markup = InlineKeyboardMarkup(keyboard)
```

```
            await query.edit_message_text(
                text=f"Информация о продукте '{product_name}':\n{details}",
                reply_markup=reply_markup
            )
            return

# Обработчик кнопки "Назад к категориям"
async def back_to_categories(update: Update, context: ContextTypes.DEFAULT_TYPE):
    query = update.callback_query
    await query.answer()

    # Получаем список категорий
    keyboard = [
        [InlineKeyboardButton(category, callback_data=category)] for category in PRODUCTS
    ]
    reply_markup = InlineKeyboardMarkup(keyboard)

    # Отправляем сообщение с кнопками категорий
    await query.edit_message_text(
        text="Выберите категорию:",
        reply_markup=reply_markup
    )

# Обработчик кнопки "Назад к категории"
async def back_to_category(update: Update, context: ContextTypes.DEFAULT_TYPE):
    query = update.callback_query
    await query.answer()

    category = context.user_data.get("current_category", None)
    if category:
        # Кнопки для продуктов в выбранной категории
        keyboard = [
            [InlineKeyboardButton(product, callback_data=f"product_{product}") for product in PRODUCTS[category]]
        ]
        keyboard.append([InlineKeyboardButton("← Назад к категориям", callback_data="BACK_TO_CATEGORIES")])
        reply_markup = InlineKeyboardMarkup(keyboard)

        await query.edit_message_text(
            text=f"Вы выбрали категорию: {category}\nВыберите продукт:",
            reply_markup=reply_markup
```

```
    )

# Основной код запуска бота
def main():
    app = ApplicationBuilder().token("7731281669:AAHitFDk7X5fS0nhkWVkSTuqyRC-YmK3bRI").build()
    app.add_handler(CommandHandler("start", start))
    app.add_handler(CallbackQueryHandler(category_callback, pattern="^(Мясо|Овощи|Молочные
продукты|Фрукты)$"))
    app.add_handler(CallbackQueryHandler(product_callback, pattern="^product_"))
    app.add_handler(CallbackQueryHandler(back_to_categories, pattern="BACK_TO_CATEGORIES"))
    app.add_handler(CallbackQueryHandler(back_to_category, pattern="BACK_TO_CATEGORY"))

    print("Бот запущен!")
    app.run_polling()

if __name__ == "__main__":
    main()
```

Вывод