



# Домашнее задание

## Цель

Сделать workflow в n8n для автоматического анализа лабораторных результатов пациента с определением критичности показателя и формированием итогового ответа.

Закрепить навыки:

- построения логики обработки данных в n8n;
- валидации входных параметров;
- использования узла Code;
- работы с ветвлением (IF);
- формирования структурированного выходного объекта.

## Задание

### Шаг 1. Подготовка workflow

1. Создать новый workflow в n8n.
2. Добавить узел Manual Trigger.
3. Настроить структуру workflow, включающую:
  - узел обработки и валидации данных;
  - узел логического ветвления;
  - две отдельные ветки формирования ответа.
4. Приложить скриншот схемы workflow (видны названия узлов и связи).

### Шаг 2. Валидация входных данных

Создать узел Code (например, `Validate & Triage`), который:

1. Принимает объект с данными:
  - `patientId`
  - `test`
  - `value`
  - `unit`

- collectedAt
  - lab (опционально)
2. Выполняет:
    - проверку наличия обязательных полей;
    - проверку числового типа значения value;
    - нормализацию названия теста (приведение к нижнему регистру).
  3. Если обязательные поля отсутствуют — выполнение должно завершаться ошибкой.
  4. Рассчитывает параметр severity:
    - "critical"
    - "normal"
  5. Определяет triageReason — строку с объяснением принятого решения.
  6. Возвращает структурированный объект, содержащий:
    - исходные данные;
    - severity;
    - triageReason;
    - receivedAt (текущее время).

## Шаг 3. Настройка логического ветвления

1. Добавить узел IF.
2. Настроить условие:
  - если severity равен "critical" → переход в ветку True;
  - иначе → ветка False.
3. Приложить скриншот настроек условия.

## Шаг 4. Формирование ответа для критического случая

В ветке True создать узел Code (например, Build Critical Response ), который формирует итоговый объект:

- ok
- isCritical = true
- patientId
- test
- value
- unit

- severity
- triageReason
- recommendedAction (эскалация, срочное уведомление врача)
- receivedAt

Приложить результат выполнения.

## Шаг 5. Формирование ответа для нормального случая

В ветке False создать узел Code (например, `Build Normal Response`), который формирует итоговый объект:

- ok
- isCritical = false
- patientId
- test
- value
- unit
- severity
- triageReason
- recommendedAction (рутинная обработка)
- receivedAt

Приложить результат выполнения.

## Шаг 6. Тестирование workflow

Сделать минимум два запуска:

1. Сценарий с критическим значением (например, повышенный уровень калия).
2. Сценарий с нормальным значением.

Приложить:

- скриншоты выполнения обоих сценариев;
- краткое пояснение, почему один случай классифицирован как `critical`, а другой как `normal`.

## Подсказки по ключевым частям

- Для обработки логики используйте узел Code.
- Для ветвления используйте IF.
- Используйте явные проверки обязательных полей.
- Добавляйте комментарии внутри кода для пояснения логики.
- Убедитесь, что workflow выполняется без ошибок.

## Что проверить перед отправкой (чек-лист)

- Создана корректная структура workflow.
- Реализована валидация входных данных.
- Настроено логическое ветвление.
- Реализованы две разные ветки ответа.
- Выполнено тестирование двух сценариев.
- Результаты приложены.
- Workflow выполняется без ошибок.

## Советы по улучшению работы

- Добавьте проверку соответствия единиц измерения ожидаемым.
- Вынесите пороги критичности в отдельный объект внутри кода.
- Добавьте обработку случая, если тест не поддерживается.
- Сделайте имена узлов понятными и структурированными.

Ответ

Ссылка на решение:

<https://disk.yandex.ru/d/yyWyLOYg6BPBnw>