Задание для выполнения в классе:

1. Напишите функцию, которая принимает на вход списки меток звуков, слов и синтагм и возвращает список словарей, отражающих иерархическую структуру высказывания.

   Каждый словарь в списке – это синтагма. У него должны быть ключи:

   - `"model"` : интонационная модель;
   - `"start"` : время начала (в отсчётах);
   - `"end"` : время конца (в отсчётах);
   - `"nucleus"` : индекс слова, содержащего интонационный центр;
   - `"words"` : список слов.

     Каждое слово – это тоже словарь. У него должны быть ключи:

   - `"word"` : слово в орфографической записи;
   - `"start"` : время начала (в отсчётах);
   - `"end"` : время конца (в отсчётах);
   - `"is_stressed"` : является ли слово ударным (`True` или `False`);
   - `"is_nucleus"` : является ли слово интонационным центром (`True` или `False`);
   - `"is_prominent"` : несёт ли слово дополнительную интонационную выделенность (`True` или `False`);
   - `"stressed_vowel"` : индекс ударного гласного;
   - `"sounds"` : список звуков.

     Каждый звук – это тоже словарь. У него должны быть ключи:

   - `"sound"` : обозначение аллофона;
   - `"start"` : время начала (в отсчётах);
   - `"end"` : время конца (в отсчётах).

     Условные обозначения:

   - метки звуков находятся на уровне B1;
   - метки слов находятся на уровне Y1;
   - метки синтгам находятся на уровне R2;
   - на уровне синтагм метки пауз имеют названия вида pX, где X от 1 до 6;
   - по умолчанию интонационный центр находится на последнем слове синтагмы;

- если это не так, то перед центром стоит символ [-];
- просодически выделеные слова отмечены знаком [+];
- ударные гласные заканчиваются на символ 0;

```
!wget https://pkholyavin.github.io/mastersprogramming/cta0001-0010.zip
```

```
!unzip -q cta0001-0010.zip
```

```
from itertools import product
letters = "GBRY"
nums = "1234"
levels = [ch + num for num, ch in product(nums, letters)]
level_codes = [2 ** i for i in range(len(levels))]
code_to_level = {i: j for i, j in zip(level_codes, levels)}
level_to_code = {j: i for i, j in zip(level_codes, levels)}
```

```
def detect_encoding(file_path):
    encoding = "utf-8"
    try:
        l = open(file_path, 'r', encoding="utf-8").read()
        if l.startswith("\ufeff"):  # т.н. byte order mark
            encoding = "utf-8-sig"
    except UnicodeDecodeError:
        try:
            open(file_path, 'r', encoding="utf-16").read()
            encoding = "utf-16"
        except UnicodeError:
            encoding = "cp1251"
    return encoding
```

```
def read_seg(filename: str, encoding: str = "utf-8-sig") -> tuple[dict, list[dict]]:
    with open(filename, encoding=encoding) as f:
        lines = [line.strip() for line in f.readlines()]

    # найдём границы секций в списке строк:
    header_start = lines.index("[PARAMETERS]") + 1
    data_start = lines.index("[LABELS]") + 1

    # прочитаем параметры
    params = {}
    for line in lines[header_start:data_start - 1]:
        key, value = line.split("=")
        params[key] = int(value)

    # прочитаем метки
```

```
    labels = []
    for line in lines[data_start:]:
        # если в строке нет запятых, значит, это не метка и метки закончились
        if line.count(",") < 2:
            break
        pos, level, name = line.split(",", maxsplit=2)
        label = {
            "position": int(pos) // params["BYTE_PER_SAMPLE"] // params["N_CHANNEL"],
            "level": code_to_level[int(level)],
            "name": name
        }
        labels.append(label)
    return params, labels
```

2. Напишите функцию, которая принимает на вход имя звукового файла и сопутствующих меток и изображает график мелодической деклинации, т.е. максимумы ЧОТ в ударном гласном каждого слова, для каждой синтагмы. ЧОТ вычислите при помощи `parselmouth`.

```
!pip install praat-parselmouth
```

3. Напишите программу, которая обрабатывает все файлы в архиве и рисует N графиков (по количеству разных интонационных моделей, встретившихся в материале), на каждом из которых изображены все интонационные кривые внутри ядерного гласного из всех синтагм, оформленных этой моделью.

Домашнее задание: проделайте всё то же самое, но используйте метки G1 как источник информации о ЧОТ. Сравните полученные данные.