```
!pip install tgt
```

```
import tgt
```

```
!wget https://pkholyavin.github.io/mastersprogramming/cta0001.TextGrid
```

```
grid = tgt.io.read_textgrid("cta0001.TextGrid")
```

Задание для выполнения в классе: напишите цикл, который перебирает все интервалы из уровня `"phonetic real"` и выводит на экран название каждого интервала и его серединную точку.

```
tier = grid.get_tier_by_name("phonetic real")
for interval in tier:
    mid = (interval.start_time + interval.end_time) / 2
    print(interval.text, round(mid, 3)) # округлим до 3 знаков после запятой дл
```

Задание для выполнения в классе: напишите функцию, которая принимает на вход имя файла .seg, читает из него метки, создаёт новый TextGrid и уровень IntervalTier, добавляет в него все интервалы, ограниченные парами соседних меток, и записывает это всё в файл .TextGrid

Не забудем, что в файлах .TextGrid время хранится в секундах! Чтобы перевести время из отсчётов в секунды, нужно разделить его на частоту дискретизации.

```
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 read_seg(filename: str, encoding: str = "utf-8-sig") -> tuple[dict, list[di
    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_CHAN
                "level": code_to_level[int(level)],
                "name": name
            }
            labels.append(label)
        return params, labels


    import os.path


    def seg_to_textgrid(filename: str, res_filename: str|None = None, max_time: float|None = Nor
        params, labels = read_seg(filename)
        new_grid = tgt.core.TextGrid()
        new_tier = tgt.core.IntervalTier()
        new_grid.add_tier(new_tier)
        for left, right in zip(labels, labels[1:]):
            start_time = left["position"] / params["SAMPLING_FREQ"]
            end_time = right["position"] / params["SAMPLING_FREQ"]
            text = left["name"]
            new_tier.add_interval(tgt.core.Interval(start_time, end_time, text))

        # если пользователь не задал имя для нового файла,
        # возьмём старое, но заменим в нём расширение
        if res_filename is None:
            res_filename = os.path.splitext(filename)[0] + ".TextGrid"

        # общее время длительности файла может не совпадать со временем последней метки
        # чтобы не читать файл .wav ради его длительности, пусть пользователь задаст
        # длительность файла самостоятельно
        if max_time is not None:
            if max_time < new_tier[-1].end_time:
                raise ValueError("specified max_time is earlier than last interval")
            new_tier.end_time = max_time
        tgt.io.write_to_file(new_grid, res_filename, format="long")


    !wget https://pkholyavin.github.io/mastersprogramming/cta0001.seg_B2


    seg_to_textgrid("cta0001.seg_B2", "out.TextGrid", max_time=1.747)
```