```
import pandas as pd

from extractors.direct_downloads import extract_with_request
from models.registry import Registry


class AUT(Registry):
    def __init__(self, **kwargs):
        """
         Child container for extracting, transforming, and loading registry data for Austria

        :param kwargs: see models.registry for details
        """

        [illegible]

        # assign constant values for final registry
        [illegible]
        [illegible]
        [illegible]
        [illegible]

        # assign registry input data source and filename
        [illegible]
[illegible]
        [illegible]

        # assign registry input data extraction method
        [illegible]
        # assign registry input data reading method
        [illegible]
        # assign registry data wrangling method
        [illegible]

        # extract, transform, and load registry data
        [illegible]

    def extract_registry(self):
        """
        Extracts registry input data from remote source

        :return: effect - creates [DATA_DIR|TEST_DIR]/[REGISTRY_INPUT_DIR]/[self.registry_code]/[self.registry_filename] file
        """

        # extract registry input data
        [illegible]
                                      [illegible]

        # read registry input data
        [illegible]

    def read_registry(self, registry=None):
        """
        Reads registry input data

        :param (io.StringIO) registry: text stream of extracted registry input data
        :return: effect - modifies self.registry
        """

        [illegible]
            # handle when registry input data has been extracted but not in current session
            [illegible]

        # read registry input data
        [illegible]
        # prep registry input data
        [illegible]

        # assign registry input data
        [illegible]

    def wrangle_registry(self):
        """
        Transforms registry data

        :return: effect - modifies self.wrangled_records
        """

        # loop through raw registrations
        [illegible]
            # initialize storage for wrangled representation of registration in focus
            [illegible]
                [illegible]
                [illegible]
                [illegible]
                [illegible]
                [illegible]
                [illegible]
                [illegible]
            [illegible]

            # update list of transformed values to record registrant and accommodate multiple registrants
            [illegible]

    def __add_registrant_copies(self, row, record):
        """
        Adds copies of a wrangled registration with updated registrant information to reflect all registrants

        :param (namedtuple) row: raw observation of a registration to extract registrants from
        :param (dict) record: wrangled representation of 'row' argument to duplicate
        :return: effect - modifies self.wrangled_records
        """

        # parse entity names and addresses for registration
        [illegible]

        # add copies of wrangled registration with updated registrant info to reflect all registrants
        [illegible]
            # avoid modifying original wrangled representation of registration
            [illegible]

            # update registrant-related values
            [illegible]
            [illegible]

            # update list of transformed values to populate final registry
            [illegible]

    @staticmethod
    def __format_colname(colname):
        """
        Formats column name for namedtuple compatibility

        :param (str) colname: column name from registry input data to format
        :return: formatted copy of 'colname' argument suitable for namedtuple indexing
        """

        [illegible]
            # handle corner-case
            [illegible]
        [illegible]
            [illegible]

        [illegible]

    @staticmethod
    def __get_names_and_addresses(operator):
        """
        Parses and formats registrant name(s) and address(es)

        :param (str) operator: registrant information to extract values from
        :return: formatted registrant name(s); formatted registrant address(es)
        """

        [illegible]
            # handle corner-case where extra new-line character exists
            [illegible]

        # isolate registrant components
        [illegible]

        # initialize storage for parsed registrant information
        [illegible]
        [illegible]
        [illegible]

        # loop through registrant components
        [illegible]
            # extract and assign values
            [illegible]
            [illegible]
            [illegible]

        [illegible]
```