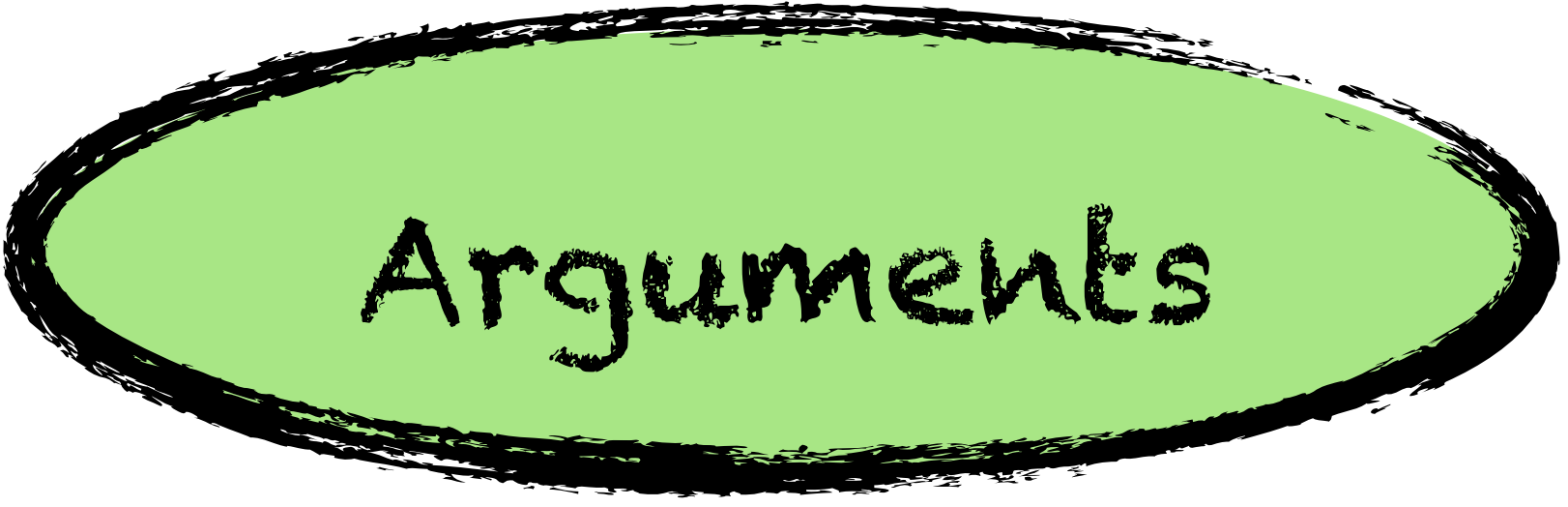
scratch — (179 x 99)

```python
#!/usr/bin/env python3
# -*- coding: utf-8 -*-


from utils import time_me, print_args, print_retval
from utils import get_gtf_value
from utils.rna import RNATranslationTable
import re
#from os.path import splitext

import logging
logger = logging.getLogger() # root logger
logging.basicConfig(level=logging.INFO,format='%(message)s')

@time_me
@print_args
def get_all_transcripts(filename="Homo_sapiens.GRCh38.87.gtf", chromosome='7', gene='ENSG00000001626'):

    transcripts = {}

    # First pass: Fetch all transcripts for the given gene and chromosome
    # ====================================================================
    logger.debug('First pass on file %s' % filename)
    logger.debug('Chr %s | Gene %s' % (chromosome,gene))
    with open(filename, mode="rt") as gtf:
        #gene_id = 'gene_id "%s"' % gene
        gene_re = re.compile(r'gene_id\s+"?{}"?'.format(gene))
        for line in gtf:
            blocks = line.split("\t")
            # Only that chromosome and
            if (
                len(blocks) < 9 or              # no comments, please
                blocks[0] != chromosome or      # only that chromosome. Careful: not comparing integers!
                blocks[2] != 'transcript' or    # the line should be a transcript
                not gene_re.search(blocks[8])  # Is that the right gene?
            ):
                continue # skip to the next line

            # Otherwise, it is a transcript for the given gene and chromosome
            attributes = blocks[8]
            transcript_id = get_gtf_value('transcript_id',attributes)

            assert( transcript_id ) # is not None
            assert (transcript_id not in transcripts), ("How come I see transcript %s already? \n\nLine:\n\n%s" % (transcript_id,line))

            start  = int(blocks[3])
            end    = int(blocks[4])
            strand = 1 if blocks[6] == '+' else -1

            # Adding it to the table
            transcripts[transcript_id] = {
                'start':start,
                'end':end,
                'strand':strand,
                'exons':{}, # exons will be added in the second pass. Empty so far.
                'start_codon': None,
                'stop_codon': None
            }
            logger.debug('Added record: {} => {}'.format(transcript_id,transcripts[transcript_id]))

    logger.debug('Transcripts after first pass')
    logger.debug(transcripts)

    # Second pass, fetching the exons for those transcripts
    # Must rescan, can't reuse the gtf iterator: it's at the end already.
    # ====================================================================
    logger.debug('Second pass')
    with open(filename, mode="rt") as gtf:
        for line in gtf:
            blocks = line.split("\t")

            if (
                len(blocks) < 9 or          # no comments, please
                blocks[0] != chromosome or  # only that chromosome
                not (blocks[2] == "exon" or blocks[2] == "start_codon" or blocks[2] == "stop_codon")
            ):
                continue # Skip that line

            feature = blocks[2]
            attributes = blocks[8]

            transcript_id = get_gtf_value('transcript_id',attributes)

            if transcript_id not in transcripts: # checking the keys
                continue # Skip cuz not a transcript for that given gene

            if not gene_re.search(attributes):
                print("Weird! I should have a gene_id {gene} in {attr}".format(gene=gene,attr=attributes))


            if feature == "exon":
                logger.debug('Found an exon')
                exon_id = get_gtf_value('exon_id',attributes)
                exons = transcripts[transcript_id].get('exons',None)
                assert( exons is not None )
                if exon_id in exons:
                    print("Weird! Have I seen that exon %s before?" % exon_id)
```

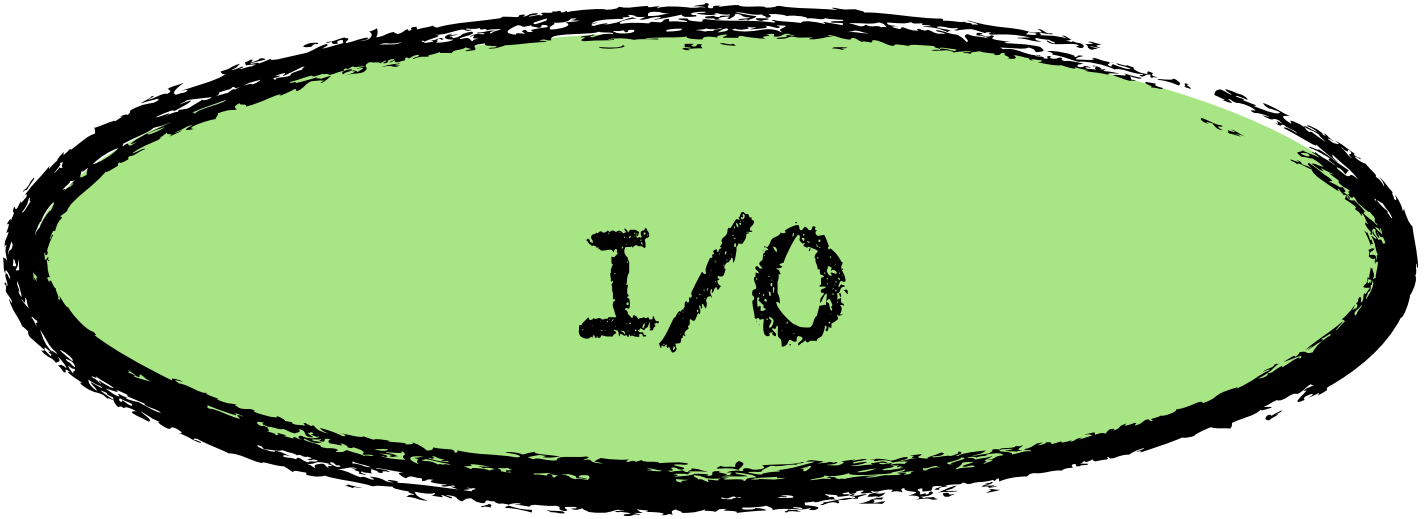
U:--- main.py Top L26 Git-vt17 (Python MMM) 02:37 1.21


Variables

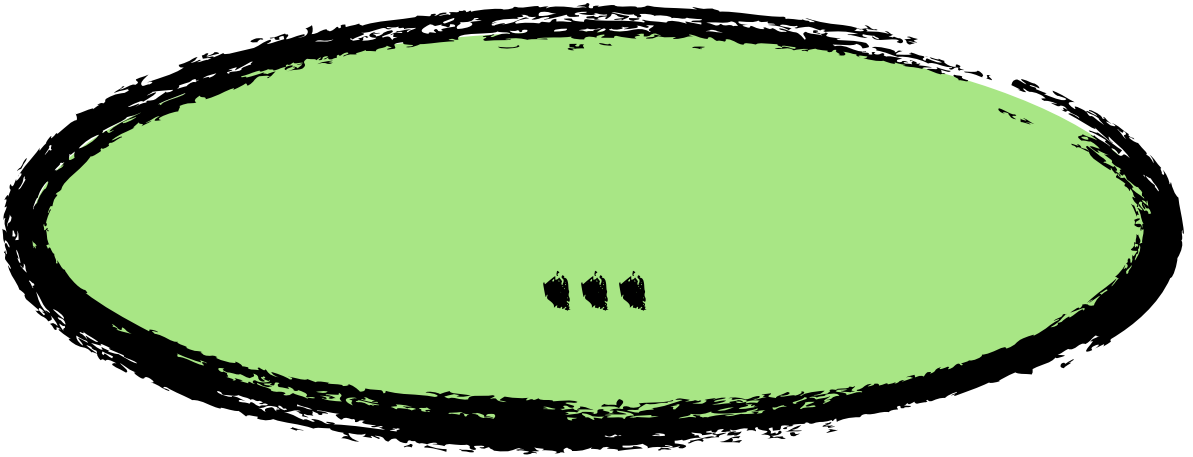
Functions


Arguments

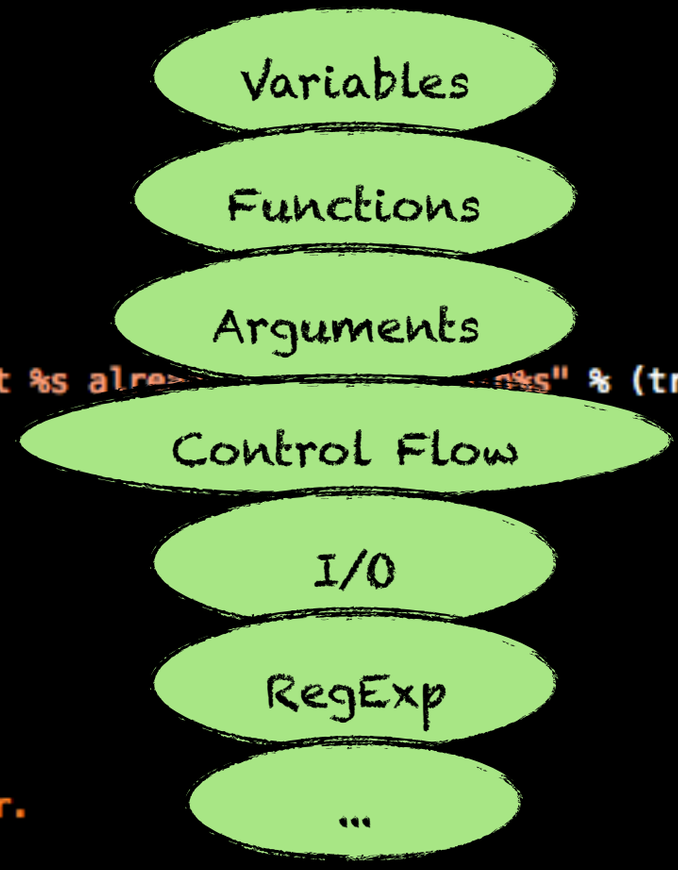
Control Flow


I/O


RegExp



```python
@time_me
@print_args
def get_all_transcripts(filename="Homo_sapiens.GRCh38.87.gtf", chromosome='7', gene='ENSG00000001626'):

    transcripts = {}

    # First pass: Fetch all transcripts for the given gene and chromosome
    # ====================================================================
    logger.debug('First pass on file %s' % filename)
    logger.debug('Chr %s | Gene %s' % (chromosome,gene))
    with open(filename, mode="rt") as gtf:
        #gene_id = 'gene_id "%s"' % gene
        gene_re = re.compile(r'gene_id\s+"?{}"?'.format(gene))
        for line in gtf:
            blocks = line.split("\t")
            # Only that chromosome and
            if (
                len(blocks) < 9 or                  # no comments, please
                    blocks[0] != chromosome or      # only that chromosome. Careful: not comparing integers!
                    blocks[2] != 'transcript' or    # the line should be a transcript
                    not gene_re.search(blocks[8])  # Is that the right gene?
                ):
                    continue # skip to the next line

            # Otherwise, it is a transcript for the given gene and chromosome
            attributes = blocks[8]
            transcript_id = get_gtf_value('transcript_id',attributes)

            assert( transcript_id ) # is not None
            assert (transcript_id not in transcripts), ("How come I see transcript %s alre[illegible]%s" % (transc

            start  = int(blocks[3])
            end    = int(blocks[4])
            strand = 1 if blocks[6] == '+' else -1

            # Adding it to the table
            transcripts[transcript_id] = {
                'start':start,
                'end':end,
                'strand':strand,
                'exons':{}, # exons will be added in the second pass. Empty so far.
                'start_codon': None,
                'stop_codon': None
            }
            logger.debug('Added record: {} => {}'.format(transcript_id,transcripts[transcript_id]))

    logger.debug('Transcripts after first pass')
    logger.debug(transcripts)

    # Second pass: fetching the exons for these transcripts
```



Why Python?

Repetitive task: automate!

* Write code
* transform it to executable (compile)
* run
* check/test
* start again...

Too slow... => Python is for you