# AveyTense Project Change Log

This document provides changeovers in the AveyTense project written in Python. About unknown or unseen on a large scale declarations please contact me either via email or Discord (*aveyzan*).

Declarations aren't tested under angle of optimization, rather of predicted results during compilation. For code changes, you can also ask me on the aforementioned ways to contact.

This document will show only versions, which were uploaded to PyPi since 6th August 2024, those are versions 0.3.26rc1 and above. Older versions may no longer be maintained as archives (to 0.3.24) due to improvements in code.

Versions, which have their upload date wrapped in circled brackets, but do not have an identifier, like: PyPi/Zip, PyPi or Zip, then this is the upload date to PyPi.

Documentation website is account-free and doesn't have any malicious code to inspect the website's viewers. The website is made to document AveyTense and its definitions, and for nothing bad. It even was verified in 2 different domains and both confirmed the website is safe to browse.


Document maintained since 7th December 2024.

# Versions and Changeovers

This section provides all existent versions since version 0.3.26rc1, and what changed after their release dates. * indicates versions that haven't been published yet; their upload date may vary too.

## Version 0.3.60* (30th Nov 2025)

## Version 0.3.59 (12th Nov 2025)

- Fixed method `aveytense.Tense.isMemoryView()` (invalid key type in `for` loop and corrected type hinting)
- Added class method `aveytense.Tense.isType()`.
- `aveytense.extensions`: Added file type aliases extracted from stub module `_typeshed`, type alias for special generic aliases in library `typing`, and callable type alias `EvaluateFunc` from `_typeshed`.

## Version 0.3.58 (30th Oct 2025)

- Added optional parameter `ituFormat` to constructor of `aveytense.Color` class to allow support for ITU T.416 Information technology
- New definitions:
    - `aveytense.Color.mix()` (static method)
    - `aveytense.Tense.tryOrPass()` (class method)
    - `aveytense.Tense.tryOrReturn()` (class method)
    - `aveytense.Tense.getAttr()` (class method)

## Version 0.3.57 (18th Oct 2025)

- Import optimizations for Python 3.14 (initiated since its upload date). To evade deprecation warning regarding `collections.abc.ByteString` (meant to be removed on 3.14, it was postponed to 3.17), this type alias has been re-declared
- Removed gimmick with `Ellipsis` and `None` in all sequence type checking class methods in class `aveytense.Tense` from 0.3.53. Instead consider using `aveytense.extensions.NoneType` and `aveytense.extensions.EllipsisType`
- DENOMINATION: `aveytense.types` → `aveytense.extensions` (refer to `typing_extensions` module on PyPi)
- `aveytense.constants`: Added unsigned and normal minimal and maximal integer values of the form borrowed from C: `[U]INT<bits>_(MIN|MAX)`. Bits values: 8, 16, 32, 64, 128, 256, 512, 1024, 2048, 4096, 8192. Restored JavaScript constants in older form before 0.3.53, and retracted them from `aveytense.Math`. `JS_MIN_VALUE`, however, received a little change - it now returns an instance of `decimal.Decimal` instead of `float`.

- `aveytense.util`: Added 2 missing properties for `CodeFlags` variable and `Flags.code` attribute: `method` and `hasDocString`. These only exist since Python 3.14
- Renamed following definitions:
    - Module `aveytense.extensions`
        - `IOReader` → `Reader`
        - `IOWriter` → `Writer`

## Version 0.3.56 (6th Oct 2025)

- Correctly backported ABCs from the `typing` library: `IO`, `TextIO` and `BinaryIO`, in module `aveytense.types` before Python 3.9 (while `typing_extensions` has these since 4.7.0 - meaning these exist for Python 3.7 and later). Adjusted imports to support Python 3.6 and 3.7 in the future
- Performance tests on `aveytense.RGB.__int__()` and `aveytense.RGB.__hex__()` methods. In the case of the first one it is advised to use `int(<object of class aveytense.RGB>)` for the fastest execution time (rather than what can be found in the internal code of `__int__()` method). In case of the `__hex__()` method, it should be used over the `hex` final property (although their execution times are very similar) and `hex(int(<object of class aveytense.RGB>))`.

## Version 0.3.55 (24th Sep 2025)

- Projected versions 0.3.55b2 (24th Sep) and 0.3.55rc1 (27th Sep) have been cancelled from uploading, and 0.3.55 (originally planned for 30th Sep) have been uploaded in emergency mode, because the owner of AveyTense is about to service their computer. That means next versions are likely to be postponed. Version 0.3.55 will be probably the last version uploaded to PyPi this month.
- Removed all class methods and attributes in `aveytense.Tense` class equal to `itertools` library equivalents; instead import `aveytense.types` to access these, including backported classes `batched` and `pairwise`.

## Version 0.3.55b1 (21st Sep 2025)

- The internal class for `aveytense.Tense.versionInfo` and `aveytense.aveytenseVersion()` is no longer inheritable
- Fixed `aveytense.types.TypingGenericType` to be a clean type alias of `typing._GenericAlias`
- Added remaining quasi-generic (in AveyTense these are called *secondary*) types from `builtins` module (`aveytense.types`; in brackets amount of type parameters):

    - ***AVT_Filter*** (1)
    - ***AVT_Map*** (1)
    - ***AVT_Reversed*** (1)
    - ***AVT_Slice*** (3)
- Added new class method ***aveytense.Tense.splitGeneric()***

## Version 0.3.55a2 (18th Sep 2025)

- Removed type alias in class ***aveytense.types.strict***: ***AVT_Tuple*** (argument: supports variable amount of type arguments, if no ellipsis given)
- Revoked the ***strict*** class in ***aveytense.types*** submodule due to incorrect type hinting
- Class method ***aveytense.Tense.extend()*** received support for asynchronous generators and mere generators
- Added support for ***enumerate*** objects in class method ***aveytense.Tense.getGeneric()***, including the ***AVT_Enumerate*** type alias in submodule ***aveytense.types***
- New generic types in ***aveytense.types*** submodule (in brackets amount of type parameters):
    - ***AVT_Accumulate*** (1)
    - ***AVT_Batched*** (1)
    - ***AVT_Chain*** (1)
    - ***AVT_Combinations*** (1)
    - ***AVT_CombinationsReplacement*** (1)
    - ***AVT_Compress*** (1)
    - ***AVT_Count*** (1)
    - ***AVT_Cycle*** (1)
    - ***AVT_DropWhile*** (1)
    - ***AVT_FilterFalse*** (1)
    - ***AVT_GroupBy*** (2)
    - ***AVT_Islice*** (1)
    - ***AVT_Pairwise*** (1)
    - ***AVT_Permutations*** (1)
    - ***AVT_Product*** (1)
    - ***AVT_Repeat*** (1)
    - ***AVT_StarMap*** (1)
    - ***AVT_TakeWhile*** (1)
    - ***AVT_Zip*** (1)
    - ***AVT_ZipLongest*** (1)
- Replaced ***TypeError*** catching statement with ***Exception*** catching statement in class method ***aveytense.Tense.probability2()***

- Added support for ***zip*** class objects in class method ***aveytense.Tense.getGeneric()***

## Version 0.3.55a1 (15th Sep 2025)

- Patched issue in ***aveytense.Tense.shuffle()*** (lack of statement for objects of classes extending ABC ***collections.abc.Sequence***)
- Missed in ***aveytense.Tense.fib()***: the ***n*** parameter is now optional
- Every type with the ***AVT_*** prefix are now completely generic; consider using their ***__origin__*** property to retrieve their origin; for type checking consider using equivalents without the ***AVT_*** prefix
- Added ***AVT_*** prefixed types ***AVT_UnionType*** and ***AVT_GenericAlias*** to backport ***types.GenericAlias*** and ***types.UnionType*** before - respectively - Python 3.9 and 3.10 (none of these are generic though). Submodule ***aveytense.types***
- Added a little more strict versions of ***AVT_*** prefixed types (inside class ***strict*** in submodule ***aveytense.types***). These types, however, may be incorrectly type hinted
- Slowly starting to support Python 3.15
- Class method ***aveytense.Tense.append()*** received support for asynchronous generators and mere generators. Next version the same feature ***aveytense.Tense.extend()*** will have
- Class method ***aveytense.Tense.toList()*** now supports asynchronous iterable objects

## Version 0.3.54 (12th Sep 2025)

- Created minimally defined ***AVT_*** prefixed generic type aliases for ***memoryview*** and ***array.array*** classes in submodule ***aveytense.types*** - ***AVT_Array*** and ***AVT_MemoryView***. These are returned from the ***aveytense.Tense.getGeneric()*** class method, respectively: before Python 3.12 and 3.14. This allows the use of ***memoryview*** and ***array.array*** class instances to the ***aveytense.Tense.getGeneric()*** class method. Preparing every declaration to be ready for Python 3.14, this will be done for version 0.3.57.
- Class method ***aveytense.Math.fib()*** now returns a generator instead of an integer.
- Re-created ***os.PathLike*** generic ABC for Python 3.8 as ***AVT_PathLike***
- First commands: ***aveytense-upgrade*** to upgrade the AveyTense project (alias to command ***pip install --upgrade aveytense***) and ***aveytense-version*** to view the current AveyTense version

- `aveytense.Tense.random()`: Added support for mappings (no overlap with `aveytense.Tense.pick()` found)
- Added version info class `aveytense.types.TypingExtensionsVersionInfo` for checking version of `typing_extensions`

## Version 0.3.53 (30th Aug 2025) - Restructure Typing For Older Python Versions

- `None` and `Ellipsis` can be passed to sequence type checking class methods in class `~ .Tense`; means wrapping both in `type` class constructor is no longer required. However, there may be malfunctions when it comes to type hinting, this behavior may change in the future. If correct type hint is top-priority, consider using `~.types.NoneType` or `~.types.EllipsisType` classes instead
- Added unrealized concept from 0.3.41: the decorator `@aveytense.util.all`
- Added type alias `aveytense.types.CoroutineWrapperType`, and imported class `SimpleNamespace` from Python library `types`
- Fought errors occurring between Python versions 3.8 and 3.11; many errors were about subscripting with non-subscriptable types; replaced with their equivalents with the `AVT_` prefix. Patched errors when verifying backward-compatibility:

    - `aveytense.types.TypingConcatenateType`: received `TypeError` for Python 3.10 and least. Replace last type argument from ellipsis to instance of `typing.ParamSpec` to support Python versions before 3.11

    - `aveytense.types.Hashxof`: `_hashlib.HASHXOF` was not defined before Python 3.9. Receive type from invocation of function `hashlib.shake_128()` before Python 3.9

    - `aveytense.types.EllipsisType`: reference to undefined class `EllipsisType` before Python 3.10. Receive type of `Ellipsis` before 3.10 and retain import of Python `types.EllipsisType` since 3.10

    - `aveytense.Tense.getGeneric()`: invalid syntax in `typing.Union[*v]` before Python 3.11. Replace with `eval()` result instead

    - *Internal definitions in submodule `aveytense.util` for class `ParamVar`*: use of PEP 616 string methods `str.removesuffix()` and `str.removeprefix()`; before Python 3.9 they weren't defined. Re-implement these for Python 3.8

    - *Definitions in class **aveytense.Tense***: reference to ***types.UnionType*** before Python 3.10 and lack of statement, what made union types tuple internal variable unbound. Fix by restructuring the statements

    - *Final properties in class **aveytense.util.ParamVar** for annotations and one for function signature*: the ***globals*** parameter in function ***eval()*** was normally passed by keyword. Use position instead of keyword for parameter ***globals*** for versions before Python 3.13

- Asynchronous abstract base classes and classes for mapping views can now be normally returned from ***~.Tense.getGeneric()*** class method instead of ***None***. *Footnote: I have been struggling with finding a solution, especially retrieving a mere generator from an asynchronous iterable object, but it was worth it!*
- When generator or asynchronous generator is passed to ***~.Tense.reverse()*** and ***~.Tense.shuffle()*** class method, their modified copy is returned
- Class methods of class ***~.Tense*** that have the ***condition*** parameter no longer force its value being lambda expressions (one-parameter callable object instead)
- Added abstract base classes from module ***contextlib*** (***AsyncContextManager*** and ***ContextManager***) as well as provided their subscripted versions with the ***AVT_*** prefix (***AVT_AsyncContextManager*** and ***AVT_ContextManager***) in submodule ***~.types***
- Class method ***~.Tense.generator()*** no longer uses generator expression to return the generator; now code also overrides the ***__qualname__*** attribute so that the string returned from generator object won't have more complex notation as ***<generator object Tense.generator.<locals>._gen_ at 0x000001E6396BE4D0>***
- Added new class methods
    - ***~.Tense.removeSuffix()***
    - ***~.Tense.removePrefix()***
    - ***~.Tense.getFlags()***
    - ***~.Tense.asyncGenerator()***
    - ***~.Tense.isAsyncGenerator()***
- Name changes
    - ***~.constants.JS_MIN_VALUE*** → ***~.Math.JS_MIN_VALUE***
    - ***~.constants.JS_MAX_VALUE*** → ***~.Math.JS_MAX_VALUE***
    - ***~.constants.JS_MIN_SAFE_INTEGER*** → ***~.Math.JS_MIN_SAFE_INTEGER***
    - ***~.constants.JS_MAX_SAFE_INTEGER*** → ***~.Math.JS_MAX_INTEGER***
    - ***~.Tense.startswith()*** → ***~.Tense.startsWith()***
    - ***~.Tense.endswith()*** → ***~.Tense.endsWith()***

# Version 0.3.52 (18th Aug 2025) - Restructure Typing

- Type annotations from `~.util.ParamVar` that use the `__annotations__` function attribute are now officially un-stringized. Parameters from final properties that return annotated parameters which weren't sorted as they appear in the function's signature are now in correct order.
- Final property `~.util.ParamVar.keywordWithDefaults` now returns tuple instead of `None` (there was return statement missing)
- Renamed submodule `~.types_collection` to `types`. This will be probably last module denomination since removed submodule `~.tcs`
- Added type alias `~ .AbroadType` and class `~ .types.GenExprType` to use with `isinstance()` inbuilt function. The same change occurred on type alias `~ .types.LambdaType` that is no longer the same type as Pythonic `types.LambdaType`. New types that are prefixed by `AVT_` are meant to warrant type subscription backward-compatibility for Python 3.8, and should be only used for type hinting. For type checking, consider using classes/types *not* prefixed by `AVT_`. List of all type aliases with the `AVT_` prefix featured in this AveyTense version:
    - *Inbuilt classes*
        - `~.types.AVT_Dict`
        - `~.types.AVT_FrozenSet`
        - `~.types.AVT_List`
        - `~.types.AVT_Set`
        - `~.types.AVT_Tuple`
        - `~.types.AVT_Type`
    - *Classes from Python module* `collections`
        - `~.types.AVT_ChainMap`
        - `~.types.AVT_Counter`
        - `~.types.AVT_DefaultDict`
        - `~.types.AVT_Deque`
        - `~.types.AVT_OrderedDict`
    - *Abstract base classes from Python module* `collections.abc`
        - `~.types.AVT_AsyncGenerator`
        - `~.types.AVT_AsyncIterable`
        - `~.types.AVT_AsyncIterator`
        - `~.types.AVT_Awaitable`
        - `~.types.AVT_Callable`
        - `~.types.AVT_Collection`
        - `~.types.AVT_Container`

        - *~.types.AVT_Coroutine*
        - *~.types.AVT_Generator*
        - *~.types.AVT_ItemsView*
        - *~.types.AVT_Iterable*
        - *~.types.AVT_Iterator*
        - *~.types.AVT_KeysView*
        - *~.types.AVT_Mapping*
        - *~.types.AVT_MappingView*
        - *~.types.AVT_MutableMapping*
        - *~.types.AVT_MutableSequence*
        - *~.types.AVT_MutableUniqual* (refer to *MutableSet*)
        - *~.types.AVT_Reversible*
        - *~.types.AVT_Sequence*
        - *~.types.AVT_Uniqual* (refer to *AbstractSet*)
        - *~.types.AVT_ValuesView*

    - *Classes from Python module re*
        - *~.types.AVT_Match*
        - *~.types.AVT_Pattern*

- Removed decorator *~.util.ClassLike* and its alias *~.util.classlike*
- Mended type aliases *~.types.Hash* and *~.types.Hashxof*; instead of using the *type* class constructor to denote these types from functions *hashlib.sha1()* and *hashlib.shake_128()*, code was revamped to import classes *HASH* and *HASHXOF* from module *_hashlib*
- Added new type aliases
    - *~.types.AnyMeta*
    - *~.types.ProtocolMeta*
    - *~.types.TypedDictMeta*
    - *~.types.TypingGenericType*
    - *~.types.TypingConcatenateType*
    - *~.types.TypingAnnotatedType*
    - *~.types.TypingCallableType*
    - *~.types.TypingUnpackType*
    - *~.types.TypingUnionType*
    - *~.types.TypingLiteralType*
    - *~.types.TypeForm* (*typing_extensions* >= 4.13.0 exclusive, may occur on Python 3.14 in *typing* module)
- Added new class methods
    - *~.Tense.extract()*
    - *~.Tense.getGeneric()*
    - *~.Tense.isLambda()*

    - `~.Tense.isIterable()`
    - `~.Tense.isIterator()`
    - `~.Tense.isGenerator()`
    - `~.Tense.isGenExpr()`
    - `~.Tense.isAwaitable()`
    - `~.Tense.isCoroutine()`

## Version 0.3.51 (3rd Aug 2025) - More Figurate Number Functions!

- Added initial support for types (with static method `~.util.ParamVar.fromType()`) and built-in functions in class `~.util.ParamVar`. Not all built-in functions and methods can be introspected efficiently though. Added final properties with suffix `-withDefaults`
- Big change in class `~.Math`. Added following class methods (noteworthy: if `n` was equal 0, then 0 in all of these will be returned):
    - `~.Math.triangularCentered()`
    - `~.Math.square()`
    - `~.Math.squareCentered()`
    - `~.Math.pentagonalCentered()`
    - `~.Math.hexagonalCentered()`
    - `~.Math.heptagonalCentered()`
    - `~.Math.octagonalCentered()`
    - `~.Math.nonagonalCentered()`
    - `~.Math.decagonalCentered()`
    - `~.Math.hendecagonal()`
    - `~.Math.hendecagonalCentered()`
    - `~.Math.dodecagonal()`
    - `~.Math.dodecagonalCentered()`
    - `~.Math.tridecagonal()`
    - `~.Math.tridecagonalCentered()`
    - `~.Math.star()`
    - `~.Math.pyramidal()`
    - `~.Math.polygonalCentered()`
    - `~.Math.pentatope()`
    - `~.Math.tetrahedral()`
    - `~.Math.tetrahedralCentered()`
    - `~.Math.cube()`
    - `~.Math.cubeCentered()`
    - `~.Math.octahedral()`
    - `~.Math.octahedralCentered()`
    - `~.Math.dodecahedral()`

    - ***~.Math.dodecahedralCentered()***
    - ***~.Math.icosahedral()***
    - ***~.Math.icosahedralCentered()***
    - ***~.Math.pronic()***
    - ***~.Math.stellaOctangula()***
- Fixed indexes for ***~.Math.fib()*** class method, and the same class method received support for zero value
- In class ***~.Tense*** added new class method ***~.Tense.getAllItemsTypes()***

## Version 0.3.50 (12th July 2025)

- Classes ***~.util.uniquelist*** and ***~ .util.uniquetuple*** now have ellipsis as the default value to comply with constructors of inbuilt classes ***list*** and ***tuple***. Ellipsis as the parameter value returns, respectively, empty list and empty tuple.
- Corrected final properties in class ***~.util.ParamVar*** to support instance methods passed via class reference (those are de facto functions).
- Corrected traceback entries
- Migrated page from https://aveyzan.glitch.me/ to https://aveyzan.xyz/ due to announcement from Glitch concerning project hosting from 8th July. That means the AveyTense documentation is under new link https://aveyzan.xyz/aveytense/
- Added class method ***~.Tense.isAbroad()*** to inspect abroad objects

## Version 0.3.49 (3rd July 2025)

- Added class method ***~.Tense.flatten()***
- Removed following methods (meant to be removed in version 0.3.48):
    - ***~.RGB.__pos__()*** (deprecated since 0.3.41)
    - ***~.RGB.__neg__()*** (deprecated since 0.3.41)
    - ***~.RGB.__invert__()*** (deprecated since 0.3.41)

## Version 0.3.48 (24th June 2025)

- Reduced distribution size by almost 2 kilobytes
- Fixed ***~.util.ParamVar.signature*** final property in case of type annotations. In the previous version, annotations were included only in parameters with default value (excluding variable argument and variable keyword argument). Added return type annotation to the returned signature
- Removed following methods:
    - ***~.Tense.upgrade()*** (deprecated since 0.3.46)

    - ***~.Color.__pos__()*** (deprecated since 0.3.47)
    - ***~.Color.__neg__()*** (deprecated since 0.3.47)
    - ***~.Color.__invert__()*** (deprecated since 0.3.47)
- Removed following classes:
    - ***~.types_collection.String*** (deprecated since 0.3.41)
    - ***~.types_collection.Integer*** (deprecated since 0.3.41)
    - ***~.types_collection.Float*** (deprecated since 0.3.41)
    - ***~.types_collection.Complex*** (deprecated since 0.3.41)
    - ***~.types_collection.Boolean*** (deprecated since 0.3.41)
    - ***~.types_collection.UnicodeRepresentable*** (deprecated since 0.3.41)
    - ***~.protogen*** (deprecated since 0.3.47)
- Added class ***~.util.uniquelist*** to return a version of an iterable without duplicate items and changing their order as a list, and ***~.util.uniquetuple*** class being the same class as ***~.util.uniquelist***, however, a tuple is returned instead

## Version 0.3.47 (12th June 2025) - Pre-PEP 570 Positional Parameters Support & Traceback Shortening

- Revamped final properties in class ***~.util.ParamVar*** to support doubly underlined parameters before PEP 570. Before Python 3.8 (version when PEP 570 document was affirmed), to create positional-only parameters, it was required to put 2 underlines before the parameter names
- Revamped final property ***~.util.ParamVar.annotations*** to return parameters as they appear in the function's signature and their types, plus function's return type behind the key ***"return"***
- Final property ***~.util.ParamVar.annotated*** no longer include the ***"return"*** item in returned string tuple; instead check it via ***dict(~.util.ParamVar.annotations).get("return")*** statement
- 2nd item from tuple returned from ***~.util.ParamVar.annotatedDefaults*** final property is now another 2-item tuple with content: annotated type and default value (earlier it was only a default value)
- ***~.util.ParamVar.signature*** final property now displays type annotations in parameters where they were used
- All constants with prefix ***MATH_*** in submodule ***~.constants*** have been moved to class ***~.Math***, and lost the ***MATH_*** prefix in their names
- Changed traceback (reduced entries to 1)
- Up to be removed in version 0.3.48, since now deprecated:
    - class ***~.protogen***
    - ***~.Color.__pos__()***

    - ***~.Color.__neg__()***
    - ***~.Color.__invert__()***
- Removed submodule ***~.games*** (will be used for separate project ***aveytense_games***; for now consider manually asking to send this missing file if some declarations from it are needed)
- Removed constant ***~.constants.SMASH_HIT_CHECKPOINTS***

## Version 0.3.46 (30th May 2025) - Probability Decade & Backporting to Python 3.8

- Backported the project to Python 3.8, so support for Python 3.8 is now warranted! If there are errors in Python versions preceding the present one (3.13), consider contacting me! Support for Python 3.8 is just initialized, and may require further inspection. During installation process, PyPi project's ***typing_extensions*** version 4.10.0 or greater is now required
- Class method ***~.Tense.probability2()*** no longer requires same-type arguments, types can vary now. Removed restriction of ***sys.maxsize***
- Added class method ***~.Tense.probability3()***, which is circa 3 times faster than ***~.Tense.probability()*** class method
- Class methods ***~.Tense.append()***, ***~.Tense.extend()*** and ***~.Tense.exclude()*** now accept mappings (in this case added overloads)
- Re-implemented class method ***~.Tense.probability()***; shortened its code and removed restriction of ***sys.maxsize***
- ***~.Tense.probability()*** now supports a wider gamut of types. Except lists, tuples, sets, deques, frozensets, dictionaries, their ABCs equivalents (***Sequence***, ***AbstractSet*** and ***Mapping***) are now supported (deques themselves inherit ***MutableSequence*** ABC, which extends ***Sequence*** ABC; refer to inbuilt submodule ***collections.abc***). Returned type is no longer required to be an integer
- Removed option ***~.TenseOptions.disableProbability2LengthLimit*** due to change in class methods ***~.Tense.probability()*** and ***~.Tense.probability2()***
- ***~.Tense.pick()*** class method no longer has overloads. Not really colossal meaning since it concerned the mapping signature overload (type parameter for key wasn't used, and has been replaced to ***typing.Any***). Sequences and sets are now coerced to list so that the picked element may be returned without throwing unnecessary errors
- Mended ***~.Math.exsec()*** and ***~.Math.excosec()*** class methods (error thrown when sine or cosine wasn't equal 0)
- Patch (may be final) in final properties in ***~.util.ParamVar*** responsible to retrieve variable arguments (and signature is now displayed correctly)
- Class method ***~.Tense.upgrade()*** is now deprecated (will be removed on 0.3.48)

- Added class methods:
  - `~.Tense.invert()`
  - `~.Math.fma()` (alias to *fused multiply-add* operation)
  - `~.Math.coth()`
  - `~.Math.sech()`
  - `~.Math.cosech()`
  - `~.Math.acot()`
  - `~.Math.acoth()`
  - `~.Math.asech()`
  - `~.Math.acosech()`

## Version 0.3.45 (16th May 2025)

- Revamped class methods `~.Tense.reverse()` and `~.Tense.shuffle()`: after conclusion, sets cannot be reversed nor shuffled, sets passed to both methods now return a list. Every time, returned set was sorted ascendingly, despite its documentation said “unordered collection of unique elements”
- Appended feature for `~.Tense.expect()`, which was meant to be provided in version 0.3.48: parameter `m` may now also be a `set`, `range` and abroad (from `abroad()` function) objects.
- `~.util.MutableString` class constructor now accepts instances of itself, same class received instance methods: `join()` and `reverse()`.
- Static method `~.RGB.fromValue()` now supports shortened CSS syntax for hex colors (applies to strings only)
- Fixed final properties `~.util.ParamVar.variable` and `~.util.ParamVar.all`

## Version 0.3.44 (3rd May 2025)

- Re-implemented final property `~.util.ParamVar.signature`; it no longer uses any inbuilt module in its code
- Revamped method `~.util.ParamVar.__str__()` - it now also returns count of annotated parameters
- Variable argument and variable keyword argument are now correctly assembled in final property `~.util.ParamVar.all`
- Submodule `~ .extensions` is now public (renamed from `~._extensions`)
- Added new final properties in class `~.util.ParamVar`:
  - `~.util.ParamVar.positionalNoDefaults`
  - `~.util.ParamVar.universalNoDefaults`
  - `~.util.ParamVar.keywordNoDefaults`
  - `~.util.ParamVar.annotatedDefaults`
  - `~.util.ParamVar.annotatedNoDefaults`
  - `~.util.ParamVar.positionalCount`

- ***~.util.ParamVar.positionalDefaultsCount***
- ***~.util.ParamVar.positionalNoDefaultsCount***
- ***~.util.ParamVar.universalCount***
- ***~.util.ParamVar.universalDefaultsCount***
- ***~.util.ParamVar.universalNoDefaultsCount***
- ***~.util.ParamVar.keywordCount***
- ***~.util.ParamVar.keywordDefaultsCount***
- ***~.util.ParamVar.keywordNoDefaultsCount***
- ***~.util.ParamVar.annotatedCount***
- ***~.util.ParamVar.annotatedDefaultsCount***
- ***~.util.ParamVar.annotatedNoDefaultsCount***
- ***~.util.ParamVar.allCount***
- ***~.util.ParamVar.allDefaultsCount***
- ***~.util.ParamVar.allNoDefaultsCount***
- ***~.util.ParamVar.variableCount***

## Version 0.3.43 (18th Apr 2025)

- Added class method ***~.Tense.isTuple2()*** being extension of inbuilt function ***isinstance()*** and ***~.Tense.isTuple()***
- Added class decorator ***~.util.finalpropertycontainer***
- Fixed class decorator ***~.util.finalproperty***, and ***~.util.MutableString.__str__()*** (earlier, it was returning a string list)
- Patch in ***~.util.ParamVar.universal*** final property (one item too much, and that item wasn't in the universal category, but positional)
- Added decorator ***~.util.classproperty*** for backward-compatibility before Python 3.13

## Version 0.3.42 (2nd Apr 2025)

- Class methods ***~.Tense.reverse()*** and ***~.Tense.shuffle()*** provided support for generators; ***~.Tense.reverse()*** also received support for classes extending ABC ***collections.abc.Reversible***
- Due to change in class ***~.util.finalproperty***, class method ***~.Tense.isFinalProperty()*** has been updated. In this case it is now impossible to return ***True*** if there is no concrete parent class of the member. It is now mandatory to wrap both class and its member name as a string into a 2-item tuple. Behavior for globally accessible final properties is kept.
- Removed submodule ***aveytense.fencord***
- Caught anomaly in static method ***~ .RGB.fromValue()*** - strings weren't implicitly converted to RGB tuple correctly. It is discouraged to provide less than 6

hexadecimal characters in a string, and sooner or later this technique will be excised
- Removed constants `~.constants.MC_ENCHANTS` and `~.constants.MC_DURABILITY` (replacing it with `~.games.Minecraft.durability` final property; first one now occurs as final property `~.games.Minecraft.enchantments`)
- Added new classes:
  - `~.util.ParamVar`
  - `~.util.MutableString` (do not misinterpret it as an abstract base class)
- Added following class methods:
  - `~.Tense.startswith()`
  - `~.Tense.endswith()`
  - `~.Tense.test()`

## Version 0.3.41 (14th Mar 2025)

Happy irrational constant π day!

- It is no longer permitted to pass ANSI escape code to text in constructor of class `~.Color`
- Regulated checking type in constructor of class `~.Color` and `~.Tense.probability()` to comply with older Python versions before 3.10
- More proper representations of several class objects scattered around the project
- Removed submodule `aveytense.databases`
- Added new classes:
  - `~.Colors` and its alias `~.Colours`
  - `~.games.Minesweeper`
  - `~.games.Sudoku`
  - `~.games.Minecraft`
  - `~.games.Cards`
  - `~.games.TicTacToe`
- Changed following methods:
  - `~.games.Games.emptyField()` → `~.games.TicTacToe.EMPTY` (no longer class method, it is a constant now)
  - `~.games.Games.ttBoardDisplay()` → `~.games.TicTacToe.displayBoard()` (no longer class method)
  - `~.games.Games.ttBoardGenerate()` → `~.games.TicTacToe.clear()` (no longer class method; returns self instead of new duo-dimensional string list)
  - `~.games.Games.ttBoardLocationSet()` → `~.games.TicTacToe.set()` (no longer class method)

    - `~.games.Games.ttBoardSyntax()` → `~.games.TicTacToe.syntax()` (no longer class method)
    - `~.games.Games.ttChangeChars()` → `~.games.TicTacToe.__init__()` (no longer class method)
    - `~.games.Games.ttLineMatch()` → `~.games.TicTacToe.match()` (no longer class method)
    - `~.games.Games.ttNextPlayer()` → `~.games.TicTacToe.nextPlayer()` (no longer class method)
    - `~.games.Games.ttIndexCheck()` → `~.games.TicTacToe.check()` (no longer class method)
    - `~.games.Games.ttFirstPlayer()` → `~.games.TicTacToe.__init__()` (no longer class method)
    - `~.Tense.upgrade()` now allows to properly upgrade the `pip` module, thanks to this project
    - `~.Tense.print()` lost `invokedAs` parameter, instead received `reprFirst` parameter, which allows to invoke `repr()` function first before `str()`
    - `~.Tense.isDecimal()` → `~.Tense.isDecimal2()`
    - `~.games.Games.mcEnchBook()` → `~.games.Minecraft.enchBook()`
- Added following class methods:
    - `~.Math.diagonals()`
    - `~.Math.fib()`
    - `~.Math.stdev()`
    - `~.Math.toRadians()`
    - `~.Math.toDegrees()`
    - `~.Math.toGradians()`
    - `~.Math.isFraction()`
    - `~.Math.isDecimal()`
    - `~.Math.isNumber()`
    - `~.Tense.isSlice()`
    - `~.Tense.isProperty()`
    - `~.Tense.isFinalProperty()`
    - `~.Tense.isFinalClass()`
    - `~.Tense.equal()`
    - `~.Tense.inequal()`

## Version 0.3.40 (27th Feb 2025)

- In the document, symbol `~` will be now referred to as the main module. Moreover, the main module has been renamed from `tense` to `aveytense`; please **switch** to

this name! This change has to prevent importing conflict with PyPi library called *tense*
- More imports from *typing_extensions* module than from *typing* module to comply with changes occurred before (like *TypeVarTuple* before Python 3.13 is imported via *typing_extensions*, not via *typing* module)
- Added support for *types.GenericAlias* before Python 3.9 (unconfirmed yet since AveyTense project is normally available since Python 3.9), project support since Python 3.8 will be rethought
- Fixed *~.Tense.isNone()*. Earlier, it meant errors for class methods which use this method.
- *~.Tense.any()* and *~.Tense.all()* now hold default value ellipsis in parameter condition, support for *None* has been removed, use ellipsis instead for default value
- Patch in methods *~.Math.perm()* (returned result is now correct), *~.Math.gcd()* and *~.Math.lcm()* (both allowed only negative integers thanks to missing *not* keyword)
- Updated string representation for *~.constants.VERSION_INFO* and when initializing class *~.Tense*
- Removed setting *~.TenseOptions.initializationMessage* and class *~.FencordOptions*
- Added missing modes in method *~.Tense.group()*
- Added following class methods:
    - *Tense.expect()*
    - *Math.mean()*

## Version 0.3.39 (14th Feb 2025)

Happy Valentine's Day! Apparently it turns out that this day is the 45th day of the year, and the upload date of this version refers to number 45, which, squared, returns the year of upload date of this version. Many updates have occurred so far!

- Finished numeric string checking on *tense.RGB.fromValue()* static method
- Submodule *tense.fencord* may be no longer continued and may be removed - contact Aveyzan to access the submodule in case it was no longer available
- Constants from *tense.constants* are no longer wildcard-imported via *tense* module, instead added aliased import via new variable *tense.Tense.constants*
- All class methods in class *tense.Math* now have type checking (and trigonometric functions received domain checking)
- Class method *tense.Tense.upgrade()* received optional parameter *aveytense*, which allows to upgrade AveyTense project only or not; defaults to *True*
- Updated *tense.Math.abs()* class method, see here for note

- Class method `tense.Math.isInRange()` received a `mode` parameter to control intervals (4 modes are possible: closed, closed-open, open-closed and open, aliased, respectively, to: `c`, `co`, `oc` and `o`; case is insensitive)
- `tense.Math.lcm()` and `tense.Math.gcd()` class methods received support before Python 3.9
- `tense.Math.log()` class method now has parameter `x` as positional-only
- Added following class methods:
    - `Math.lwdp()` (alias to *least with digit product*)
    - `Math.maxDigit()`
    - `Math.minDigit()`
    - `Math.perm()`
    - `Math.toDigits()`

## Version 0.3.38 (11th Feb 2025)

- Fixed constructor of `tense.Color` class for `tense.RGB` instances
- Added domain checking for class methods:
    - `Math.acosec()`
    - `Math.asec()`
    - `Math.asin()`
    - `Math.acos()`
- Finished class `tense.RGBA`
- Corrected checking numeric strings for static method `tense.RGB.fromValue()` and `tense.Color`
- Added new class methods:
    - `Math.isIncreasing()`
    - `Math.isDecreasing()`
    - `Math.isMonotonous()`
    - `Math.isNonIncreasing()`
    - `Math.isNonDecreasing()`
    - `Math.isConstant()`
    - `Tense.isBinary()`
    - `Tense.isOctal()`
    - `Tense.isDecimal()`
    - `Tense.isHexadecimal()`
    - `Tense.isFinalVar()`

## Version 0.3.37 (9th Feb 2025)

- Removed all Tkinter declarations (in this case classes)
- Class method `tense.Tense.clear()` now accepts mutable mappings (such as dictionaries) and instances of `tense.Color`

- Added type alias `tense.Colour` for `tense.Color` class
- Added experimental class `tense.RGBA`
- Fixed constructor of `tense.Color` class and static method `tense.RGB.fromValue()`

## Version 0.3.37a1 (3rd Feb 2025)

- Removed all Tkinter declarations (in this case classes), which were accessible via submodule `tense.types_collection`
- added `@tense.util.finalproperty` class decorator to create final properties accessible via class instance (warning: will not work with `@staticmethod` inbuilt decorator; with `@classmethod` since Python 3.13 it is not possible). For final fields consider using `tense.util.FinalVar`

## Version 0.3.36 (2nd Feb 2025)

- Removed Tkinter imports (its declarations will be, forever, removed in further version - 0.3.37a1)
- Removed `tense.File` class
- Added static method `tense.RGB.fromValue()`
- Added many values checking at once in the following class methods:
    - `Tense.isInt() / Tense.isInteger()`
    - `Tense.isFloat()`
    - `Tense.isComplex()`
    - `Tense.isBool() / Tense.isBoolean()`
    - `Tense.isStr() / Tense.isString()`
    - `Tense.isNone()`
    - `Tense.isEllipsis()`

- Added initial support for generic types in the following class methods:
    - `Tense.isList()`
    - `Tense.isTuple()`
    - `Tense.isDict()`
    - `Tense.isSet()`
    - `Tense.isFrozenSet()`

- Planning update on class method `tense.Tense.probability()`; it will concern any-type support and many sequences/mappings support
- Added string constants and constants `MODE_AND` and `MODE_OR`
- Added class method `tense.Math.isInRange()`

## Version 0.3.35 (23rd Jan 2025)

- Another patch in method `Tense.probability()` (disqualified `ZeroDivisionError` when more than 2 values are passed, in modulo operation)
- Finished class `FinalVar`, and moved it to the `tense.util` submodule, simultaneously removing it from the `tense.types_collection` submodule. Class `FinalVar` may be also subclassed. Value getting concerns usage of `+` unary plus sign and the `x` attribute (would explain class itself is a subclass of `typing.NamedTuple`)
- Moved class `ClassLike` and its lowercase alias from `tense.types_collection` to the `tense.util` submodule
- Removed option `TenseOption.enableFurryDeclarations` , and moved all methods from class Tense related to this option (`Tense.owoify()` and its alias), `Tense.aeify()` and `Tense.oeify()` have been moved to submodule `tense.games` as global functions
- Added methods `Tense.starmap()`, `Tense.isMemoryView()`, `Tense.isBytes()`, `Tense.isByteArray()`, `Tense.isSet()`, `Tense.isFrozenSet()`,
- Method `Tense.abroadEach()` may now return any-typed list, parameter `each` is now keyword-only
- Methods checking sequences (list, tuple, set, frozenset, dict) now allow union types, see updates in this page
- All constants from submodule `tense.constants` representing integer/float values are now preceded with `MATH_`, to prevent importing conflicts especially with `E` math constant
- In-code change: since Python 3.11 digit limit is disabled
- Added several enum members to submodule `tense.constants`
- Deprecated `tense.constants.VERSION_ID` and `tense.constants.VERSION_LIST`

## Version 0.3.34 (15th Jan 2025)

Happy (Late) New Year 2025!!

- removed class `NennaiStrings` (might have been removed on 0.3.28)
- removed class `NennaiAbroads` due to incorporation with class `Tense`
- removed class `NennaiRandomize`; methods in the class before removal:
    - `randomize()` (instead invoke `Tense.pick()` or `Tense.random()` with one parameter)
    - `randomizeInt()` (instead invoke `Tense.random()` with 2 integer parameters)
    - `randomizeStr()` (instead invoke new method `Tense.randstr()`)
    - `randomizeStr2()` (instead invoke `Tense.shuffle()`)

    - ***`randomizeUuid()`*** (instead invoke ***`Tense.uuidRandom()`***)
- class object returned from function ***`abroad()`*** now features shallow and deep copy operations
- new submodule ***`tense.util`***, moved special classes ***`Abstract`***, ***`Final`*** and ***`Frozen`*** there, along with their decorator equivalents and ***`AbstractFinal`***, ***`FinalFrozen`*** and ***`AbstractFrozen`***
- added new methods: ***`Tense.intersection()`***, ***`Tense.union()`*** (both are same as, respectively, ***`set.intersection()`*** and ***`set.union()`***, only difference is that first sequence doesn't have to be a set necessarily)
- due to fact that any mutable sequences passed to methods ***`Tense.append()`*** and ***`Tense.extend()`*** are coerced to list, any iterables are now allowed in first parameter
- added a new option ***`TenseOption.enableFurryDeclarations`*** (defaults to ***`False`***), which toggles on accessibility to methods ***`Tense.owoify()`*** and ***`Tense.uwuify()`*** (when it is set to ***`True`***)
- class ***`Tense`*** lost *informational* string conversion and support for unary ***`+`***, unary ***`-`***, and ***`~`*** operators (out-of-date)
- code update; least internal imports
- method ***`Tense.pick()`*** received ***`secure`*** parameter
- first overloaded methods: ***`Tense.bisect(), Tense.insort(), Tense.eval(), Tense.intersection(), Tense.union(), Tense.difference(), Tense.occurrences(), Tense.reverse(), Tense.shuffle(), Tense.all(), Tense.any(), Tense.first(), Tense.last(), Tense.random(), Tense.pick()`***
- method ***`Tense.isList()`*** received an optional parameter ***`type`*** allowing to select a list type as additional step before return of boolean value (e.g. ***`Tense.isList([""], int)`*** would return ***`False`***, since it is a string list and we await an integer list)
- method ***`Tense.isTuple()`*** received optional parameters ***`type`***; use as ***`Tense.isTuple(v, type)`***, second overload with parameter ***`types`*** is under experiments (e.g. ***`Tense.isTuple((87,), int)`*** returns ***`True`***, because passed is integer tuple, and expected is integer-typed tuple), tests to be done presumably on 0.3.36
- method ***`Tense.isDict()`*** received optional parameters ***`ktype`*** and ***`vtype`*** (both, for providing demanded types for, respectively, key and value in a dictionary)
- method ***`Tense.pick()`*** now, after sequences, accepts mappings (second overload)
- added method ***`Tense.hasattr()`*** being extension of inbuilt function ***`hasattr()`*** (difference is many attribute checking at once via string tuple, and mode selection - logical AND or OR)
- aspirations to add method ***`Games.blackjack()`***

## Version 0.3.33 (25th Dec 2024)

- amended setting `TenseOptions.disableProbability2LengthLimit` via code in method `Tense.probability2()` (inner code issue fixed)
- edit in code `Tense.probability()`

## Version 0.3.32 (16th Dec 2024)

- fixes on several ABCs from `tense.types_collection`: `Sequence`, `MutableSequence`, `Uniqual`, `MutableUniqual`, `Mapping`, `MutableMapping` to correct type checking, whichever is used in inspections (patched error with valid parameters actually being instances of these classes)
- sequence returned from `abroad()` function (`AbroadInitializer`) received more features: support for `+` operator (excluding unary notation, which is already provided), for `*` operator, which will act the same as in case of lists, and for `in` operator, allowing to check whether an item belongs to sequence, reversing sequence
- Added property `Tense.none` (console-specific/IDLE-specific since it returns `None`)
- added new method `Tense.occurrences()` counting amount of times item(s) appearing in a sequence, and `Tense.difference()` returning list of items of the first sequence that don't belong to the second sequence (can be also used vice versa with parameter `invert`)

## Version 0.3.31 (8th Dec 2024)

- class `TenseTk` is now deprecated and will be removed on 0.3.36, as well as Tkinter variables from `tense.types_collection` submodule. Cause of that is receding from Tkinter technology
- several methods in class `Tense` lost support for Tkinter variables, same as in `Games.mcEnchBook()` method
- suspended tests on `Frozen` class (couldn't find solution which may be like code in class `enum.Enum`)
- shortened code of `Games.mcEnchBook()` method
- moved class `Games` to new submodule `tense.games`
- added new option in class `TenseOptions`: `disableProbability2LengthLimit`, which applies on `Tense.probability2()` to prevent using internal list to generate result
- in method `Tense.disassemble()` added lacking parameter `showOffsets`
- method `Tense.repeat()` now factually returns an instance of `itertools.repeat` class; parameter `times` may now also have value `None` (earlier it supported only integers and default value of that parameter was 2, and now it is `None`)

- removed method ***`Tense.error()`***
- added method ***`Tense.isDict()`*** for checking whether a value is a dictionary
- added math methods, all in class ***`tense.Math`***: ***`isPositive()`***, ***`isNegative()`***, ***`isPrime()`***, ***`isComposite()`***, ***`lcm()`***, ***`gcd()`***

## Version 0.3.30 (29th Nov 2024)

Begin of tests on class ***`Frozen`*** from ***`tense.types_collection`***.
This version was planned to be uploaded on 13th December 2024.

## Version 0.3.29 (PyPi/Zip: 27th Nov 2024)

Fixed the ***`Final`*** class from submodule ***`tense.types_collection`*** and better efficiency of class ***`AbroadInitializer`*** (now features item getting technique).
This version was planned to be uploaded on 3rd December 2024.

## Version 0.3.28 (PyPi/Zip: 23rd Nov 2024)

- module ***`typing_extensions`*** is now required during installation of AveyTense via PyPi, gimmick via inbuilt Python library ***`subprocess`*** will be still kept anyway
- added ***`tense.RGB`*** and ***`tense.CMYK`*** classes as helpline for ***`tense.Color`*** class
- removed static methods ***`Final.method`*** and ***`Abstract.method`***, since they are inherited by their subclasses, use ***`finalmethod`*** and ***`abstractmethod`*** functions instead (submodule ***`tense.types_collection`***)
- reorganization of type ***`_AbroadInitializer`*** - it is now a class and offers easier cast to list, tuple and also set (respectively, unary operators: +, - and ~). To be more explanatory: this type is returned from ***`abroad()`*** function

This version was uploaded on date as though (23rd November 2024).

## Version 0.3.27 (PyPi: 22nd Nov 2024 / Zip: 11th Nov 2024)

- class ***`Tense`*** no longer extends ***`NennaiStrings`*** class (argument: it will only inherit only a few useless methods), class itself is marked as deprecated
- constructor of class ***`tense.Color`*** no longer has parameter ***`underlineColor`*** (argument: no changes made in the underline when displaying text on the console)
- renamed class ***`tense.YamiTk`*** to ***`TenseTk`***

- several internal enumerator classes changed base class from ***IntFlag*** (***types_collection.IntegerFlag***) to ***Enum*** to prevent math operations on their members
- method ***tense.fencord.Fencord.fixedEmbed()*** is now marked as experimental and not available at runtime
- class ***tense.fencord.Servers*** is now marked as experimental and not available at runtime (argument: error after recent tests)
- cancelled text styling methods from ***tense.fencord.Fencord*** class
- revamp in class ***tense.fencord.FontStyles***: added new method ***addText()***, constructor now features entry text parameter along with styling option (2 parameters)
- renamed class ***tense.fencord.FontStyles*** to ***FontStyler***
- removed properties from class ***tense.fencord.Fencord***: ***getTree*** and ***getClient*** (instead use ***tree*** and ***client***)

## Version 0.3.27rc2 (PyPi/Zip: 27th Oct 2024)

Not many things were logged. In this case only change was adding ***finalmethod*** and ***abstractmethod*** function into submodule ***tense.types_collection***.
This version was planned to be uploaded on 3rd November 2024.

## Version 0.3.27rc1 (PyPi: 20th Oct 2024 / Zip: 19th Oct 2024)

When going to release this version, it was ensured that every written declaration will accord with Python 3.13 (while being on Python 3.12.7 in the meantime). More changes (and most concern submodule ***tense.types_collection***):

- handover of class ***Frozen*** and decorator ***frozen*** (first via class inheritance; both defined in submodule ***tense.types_collection***)
- classes ***Abstract*** and ***Final*** (submodule ***tense.types_collection***) no longer require keywords in subclass section, respectively: ***abstract = True*** and ***final = True*** (these may be now omitted)
- cancelled functions ***classvar*** and ***finalvar*** (submodule ***tense.types_collection***)
- several bug fixes

This version was planned to be uploaded on 23rd October 2024.

## Version 0.3.27b3 (PyPi/Zip: 13th Oct 2024)

Both changes concern declarations on the ***tense.types_collection*** submodule.

- `Final` is now class replacing `FinalClass` class, formally `Final` type referring to `typing.Final` has been renamed to `Final2` due to constancy failure
- a new class `Allocator` based on the use of `bytearray.__alloc__`

## Version 0.3.27b2 (PyPi/Zip: 26th Sep 2024)

Changes mainly concern `tense.types_collection` submodule, one repair: class `tense.types_collection.Buffer`: method `__buffer__` type `BufferFlags` with support before Python 3.12.

## Version 0.3.27b1 (PyPi/Zip: 23rd Sep 2024)

- added 'tense.types_collection.AbstractMeta' type alias being equivalent to 'abc.ABCMeta' (use it as 'metaclass = AbstractMeta')
- more patches on 'tense.types_collection.Abstract' class, since now you can use 'class AbstractClass(Abstract, abstract = True)', and even `define()` static method to create your keyword instead of 'abstract' (idea with that
- method has been actually retracted)
- implemented 'Tense.probability()' class method in Java (as static method); access since version 0.4.0 (if not, then on either separate project or Tense 1.2.0)
- project on increasing possible max 'length' parameter value (powering by 2) via 2d list; tests will be done on 0.3.28rc1 (earlier: 0.3.27rc1). To disable this feature, there will be option: 'TenseOptions.probabilityExtendedLength'

## Version 0.3.27a5 (PyPi/Zip: 13th Sep 2024)

- general note: Tense will be prepared to have Python 3.13 supported, if final release of that version is out (predicted: 1st Oct 2024)
- patched 'tense.types_collection.Abstract' class (use '_init_sentinel' subclass parameter to success putting an abstract class), used on new class 'TenseOptions'. For uncertainty you can still use 'abc.ABC' or 'metaclass = abc.ABCMeta' instead of this solution. There is no general note you should avoid using 'tense.types_collection.Abstract' class anyway. Class 'TenseOptions' has to feature attributes, which may be checked in various classes inside 'tense' module, without initializing this class in overall
- decreased number of supported types in 'reckon()' function based on ABCs extended by these types. Nevertheless, it still should work as intended. Change also concerns 'abroad()' function, and its variations
- little change in inner code of 'Tense.probability()' to assemble inner variable's '_length' correct type ('int', not 'Any')

- added >> and << operators support for 'tense.Tense' class; these work as these in C/C++, however, they work vice versa (>> - output, << - input), unless setting 'Tense.streamLikeC' is set to True. To avoid getting many class 'tense.Tense' initialization messages, there is setting 'TenseOptions.initializationMessages'. If set to False (it has False by default), these messages won't take place, True otherwise.

## Version 0.3.27a4 (PyPi/Zip: 9th Sep 2024)

- Tense.any() and Tense.all() lost 'default' parameter
- added experimental submodule 'tense.databases' (do not use it yet)
- added math methods (class 'Math'): 'triangular()', 'pentagonal()', 'hexagonal()', 'heptagonal()', 'octagonal()', 'nonagonal()', 'decagonal()', 'polygonal()'

## Version 0.3.27a3 (PyPi: 3rd Sep 2024 / Zip: 2nd Sep 2024)

- added class decorator for functions: 'tense.types_collection.ClassLike'
- same module: 'classvar()' and 'finalvar()' functions are now deprecated (for first just use 'typing.ClassVar', for second - 'tense.FinalVar')

## Version 0.3.27a2 (PyPi: 29th Aug 2024 / Zip: 28th Aug 2024)

- added variables 'Tense.versionId' and 'Tense.versionTuple' (see also 'tense.constants')
- versioning now on experimental local module 'tense._versioning' (accessible via 'tense.constants')
- 'Fencord.getClient' and 'Fencord.getTree' properties are now deprecated, use 'client' and tree' properties instead (Discord)
- Fencord class received 'id' and 'display' properties (Discord)
- added class method 'Tense.socket()'

## Version 0.3.27a1 (PyPi/Zip: 27th Aug 2024)

- moved all Fencord styling methods to new class 'tense.fencord.FontStyles', these in Fencord class are since now deprecated and will be removed on 0.3.27 (Discord)
- experimental method 'Fencord.send()' (Discord)
- migrated all exception classes to local module 'tense._exceptions' (accessible via 'tense.types_collection')

## Version 0.3.26 (PyPi: 26th Aug 2024 / Zip: 24th Aug 2024)

- despite slight change in 'Tense.error()', this class method remains deprecated
- added methods 'Tense.isEllipsis()' (alias to 'value is ...'), 'Tense.toList()' (with additional ABC ListConvertible), 'Tense.toString()', 'Tense.toStr()'
- added new auxiliary static method 'Fencord.fixedEmbed()' for preventing 25-field limit error by creating several Embed class instances (Discord)
- planning for 0.3.27 or 0.3.28 new instance method 'Fencord.send()', being equivalent to 'Interaction.response.send_message()' (Discord)
- retracted support for 'tense_eight' module on PyPi (ensure you have 3.9 or greater instead)

## Version 0.3.26rc3 (PyPi/Zip: 21st Aug 2024)

- for 'tense.tcs' module included many variables, including __constants__, which returns list of constants defined in that module
- renamed 'tense.tcs' to 'tense.types_collection', constants extracted from that module migrated to a new module 'tense.constants'
- added 'parent' parameter for 'tense.fencord.Fencord.slashCommand()', 'servers' parameter now support any iterable object filled with instances of 'discord.Object' and
- single 'discord.Object' class object (patched issue with 'tense.types_collection.Iterable' ABC; it wasn't returning an iterator, as it should)
- 'tense.types_collection' module received many new types and classes
- 'Tense.pick()' now supports any sequences, including sets
- added new methods: 'Tense.toString()', 'Tense.toStr()', 'Tense.toList()', 'Math.fact()', 'Tense.timeit()'
- 'tense_eight' isn't developed anymore, switch to 3.9 if your still stuck in 3.8 or least

## Version 0.3.26rc2 (PyPi: 16th Aug 2024 / Zip: 15th Aug 2024)

- decorator 'tense.fencord.Fencord.slashCommand()' has been restored after experiments. Perhaps this decorator by my eye doesn't return the same type as discord.app_commands.command() decorator, it actually works.
- function 'tense.FinalVar' became a constructible class (earlier: function). Reference to 'tense.tcs.FinalVar' class.
- another patches in method 'Tense.owoify()'

- several classes received __all__ and __dir__ attributes, tense.Color class received also __constants__ and other constants for so-called advanced styling
- cancelled classes tense.primary.TenseType and tense.primary.ModernString (last one under experiments)
- added tkinter.IntVar class support for tense.Tense.random()
- several rewritten classes from 'enum' and 'tkinter' module became original classes subclasses
- added class methods 'Tense.reverse()' (use on lists and strings, experimented on sets, dicts) and 'Tense.architecture()' (your system's architecture)

## Version 0.3.26rc1 (PyPi/Zip: 7th Aug 2024)

- now officially on PyPi repository! https://pypi.org/project/AveyTense/ Install via 'pip install AveyTense'. Supported versions since 3.9
- extended list of ABCs in module 'tense.tcs'
- added classes 'tense.ChangeVar' and 'tense.Color', class-like function 'tense.FinalVar'
- cancelled class 'tense.extensions.ANSIColor'
- 'tense.Tense.probability()' and 'probability2()' received support for 'tkinter.IntVar'