PYTHON FOR COMPUTATIONAL PROBLEM SOLVING

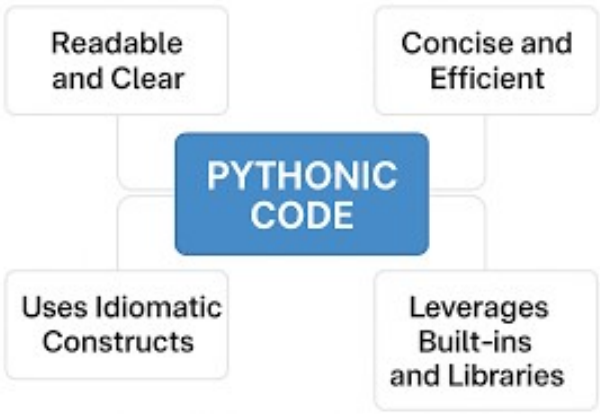
UE25CS151A

Department of Computer Science and Engineering

Standards and Guidelines for Pythonic solution

What does Pythonic mean?

- Writing code in a way that embraces the principles and idioms of the python programming language.
- Writing code that feels “natural” in python.
- Not just working code but clean and elegant code.
- Prioritizes readability, simplicity and maintainability.
- Follows community guidelines (PEPs)covered in upcoming slides.



Python Enhancement Proposal (PEP)

What is a PEP?

- A PEP is a design document providing information to the Python community.
- Used to describe a new feature for python or it's processes or it's environment.
- Primary mechanism for proposing major new features.

What PEPs matter for Pythonic solutions?

- PEP 8 – Style Guide for Python Code
- PEP 20 – The Zen of Python
- PEP 257 – Docstring Conventions.

Standards and Guidelines for Pythonic solution

"Code is read much more often than it is written"

- Guido Van Rossum

PEP 8 highlights.

- Indentation – Use 4 spaces for indentation.
- Max line length 79 chars.
- Naming Conventions
  - snake_case – Functions and Variables
  - PascalCase – Classes
  - ALL_CAPS – Constants

PEP 20 – The Zen of python.

- Long time Pythoneer Tim Peters succinctly channels the BDFL's guiding principles for Python's design into 20 aphorisms, only 19 of which have been written down.

Examples given below

- Beautiful is better than ugly.
- Explicit is better than implicit.
- Simple is better than complex.
- Complex is better than complicated.
- Flat is better than nested.
- Sparse is better than dense.

(Run import this in python to see all 19 aphorisms)

PES UNIVERSITY
CELEBRATING 50 YEARS

PEP 20 – The Zen of python.

```
>>> import this
The Zen of Python, by Tim Peters

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.
Flat is better than nested.
Sparse is better than dense.
Readability counts.
Special cases aren't special enough to break the rules.
Although practicality beats purity.
Errors should never pass silently.
Unless explicitly silenced.
In the face of ambiguity, refuse the temptation to guess.
There should be one-- and preferably only one --obvious way to do it.
Although that way may not be obvious at first unless you're Dutch.
Now is better than never.
Although never is often better than *right* now.
If the implementation is hard to explain, it's a bad idea.
If the implementation is easy to explain, it may be a good idea.
Namespaces are one honking great idea -- let's do more of those!
>>>
```

Standards and Guidelines for Pythonic solution

PES UNIVERSITY
CELEBRATING 50 YEARS

Readability First

- Descriptive variable names.
- Keep functions short and focused.
- Avoid clever but confusing one liners.
- Modularity and Function decomposition.
- Documentation and Comments.

```
1  #Bad
2  a = [90, 85, 70]
3  b = sum(a)/len(a)
4  print(b)
5
6  #Good
7  student_scores = [90, 85, 70]
8  average_score = sum(student_scores) / len(student_scores)
9  print(average_score)
```

Use Built-ins

- Prefer python's built-in functions.

```
# Not Pythonic
for i in range(len(items)):
    print(i, items[i])

# Pythonic
for i, item in enumerate(items):
    print(i, item)
```

Comprehensions

- Comprehensions are more concise and readable than loops.

```
# Not Pythonic
squares = []
for n in range(10):
    squares.append(n*n)

# Pythonic
squares = [n*n for n in range(10)]
```

What is a docstring?

A docstring is a string literal that occurs as the first statement in a module, function, class, or method definition.

Docstrings PEP 257

- Explain what your function/class does.
- Helps with documentation and ide hints.
- Makes your code more approachable and helps in collaboration.

```
1  def add(a: int, b: int) -> int:
2      """Return the sum of two integers."""
3      return a + b
```

THANK YOU

Department of Computer Science and Engineering

Prof. Kundhavai K R, CSE Department

Ack: Teaching Assistant – Adithya Jeyaramsankar
PES2UG22CS029