Python Data Analytics Cheat Sheet

Hamed Ahmadinia, Ph.D.
Hamed.Ahmadinia@metropolia.fi

Core Operations

Arithmetic:
- `+` → `df['age'] + 5` → Add 5 to all ages
- `-` → `df['capital_gain'] - 1000` → Deduct 1000
- `*` → `df['hours'] * 2` → Double hours
- `/` → `df['age'] / 2` → Float division
- `//` → `df['qty'] // 10` → Integer division
- `**` → `df['score'] ** 2` → Square values
- `%` → `df['index'] % 7` → Modulo operation

Comparisons:
- `==` → `df['income'] == '>50K'` → High earners
- `!=` → `df['edu'] != 'Bachelors'` → Non-Bachelors
- `>` → `df['age'] > 40` → Senior filter
- `<` → `df['hours'] < 30` → Part-time
- `>=` → `df['exp'] >= 5` → 5+ yrs experience
- `<=` → `df['score'] <= 100` → Valid scores

Assignment:
- `+=` → `age += 5` → Increment age
- `-=` → `balance -= 100` → Deduct balance
- `*=` → `factor *= 1.1` → 10
- `/=` → `total /= 2` → Halve total
- `//=` → `items //= 5` → Batch grouping
- `**=` → `base **= 3` → Cube base
- `% =` → `index % = 10` → Cycle index

Strings

Key Methods:
- `.strip()` → `" Data ".strip()` → Clean edges
- `.upper()/.lower()` → `"text".upper()` → "TEXT"
- `.split()/.join()` → `"a,b,c".split(",")` → List
- `.replace()` → `"01/02".replace("/","-")` → Date format

Lists

Operations:
- `.append()` → `lst.append(5)`
- `.insert()` → `lst.insert(0,5)`
- `.remove()` → `lst.remove('a')`
- `sorted()` → `sorted(lst)` new list

Comprehension:

```
squares = [x**2 for x in nums
           if x > 0]
```

Sets

Operations:
- `add()` → `unique.add("new")`
- `remove()` → `unique.discard("old")`
- `union()` → `set1 | set2`
- `intersection()` → `set1 & set2`

Example:

```
categories = set(df['category'])
if "Tech" in categories:
    process_tech()
```

Dictionaries

Methods:
- `.keys()` → List all keys
- `.values()` → List all values
- `.get()` → `dict.get('key', default)`
- `.update()` → Merge dicts

Example:

```
employee = {
    'name': 'Alice',
    'dept': 'Data Science',
    'projects': 5
}
```

Control Flow

Conditionals:

```
if temp > 30:
    category = "Hot"
elif temp > 20:
    category = "Mild"
else:
    category = "Cold"
```

Loops:

```
for idx, row in df.iterrows():
    if row['age'] > 65:
        mark_retired(idx)
```

File Handling

Pandas I/O:

```
# Read CSV with limited cols
df = pd.read_csv('data.csv',
                 usecols=['age','income'])

# Save filtered
df[df['score']>80].to_excel('high.xlsx')
```

Debugging

try/except:

```
try:
    ratio = a / b
except ZeroDivisionError:
    ratio = 0
finally:
    log_calc()
```

Type Checking:

```
if not isinstance(val, (int,float)):
    raise ValueError("Numeric input needed")
```

Best Practices

- **Type Hints:** `def process(df:pd.DataFrame)->None:`
- **Docstrings:** `"""Perform stats"""`
- **Error Handling:** Use specific exceptions
- **Vectorization:** Prefer pandas ops over loops