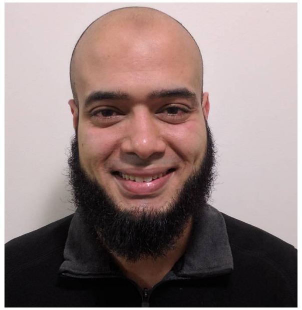
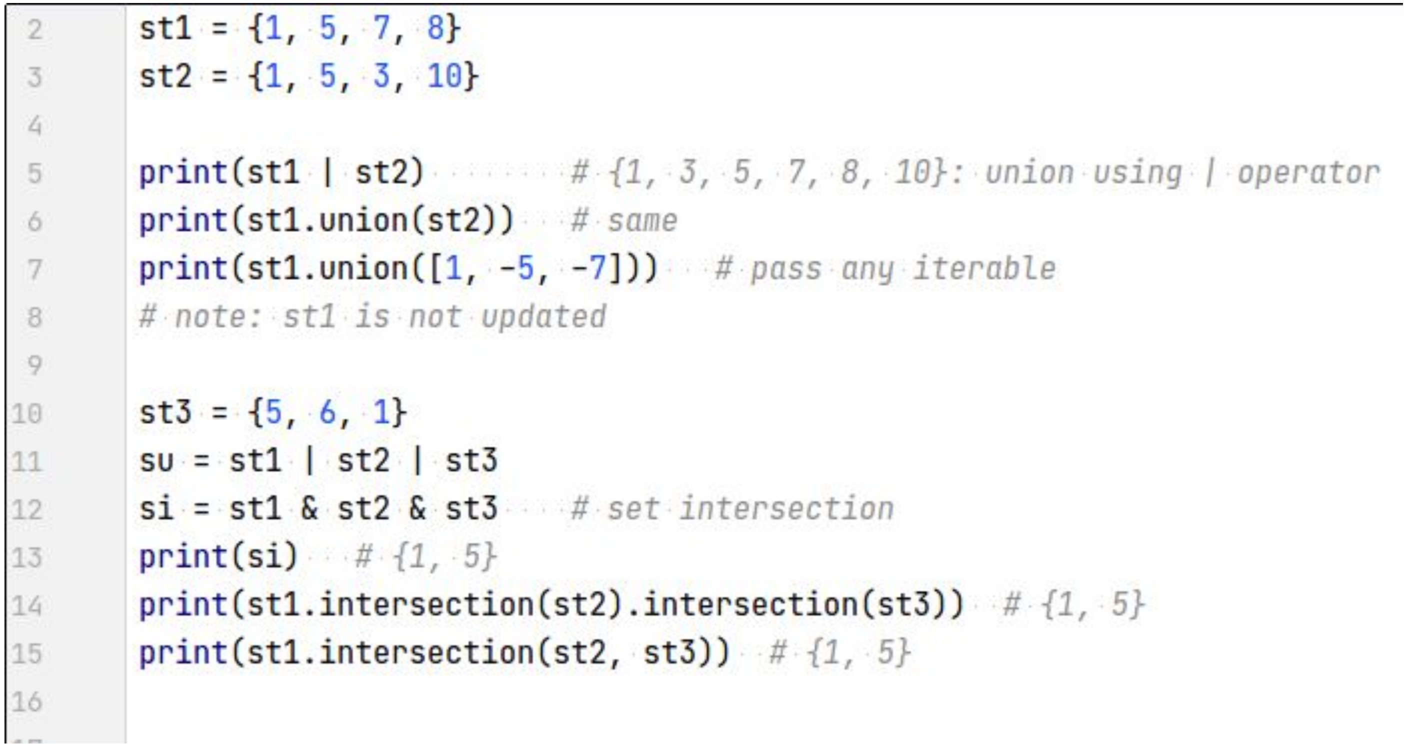
Python Programming
Set 2

Mostafa S. Ibrahim
Teaching, Training and Coaching since more than a decade!

Artificial Intelligence & Computer Vision Researcher
PhD from Simon Fraser University - Canada
Bachelor / Msc from Cairo University - Egypt
Ex-(Software Engineer / ICPC World Finalist)

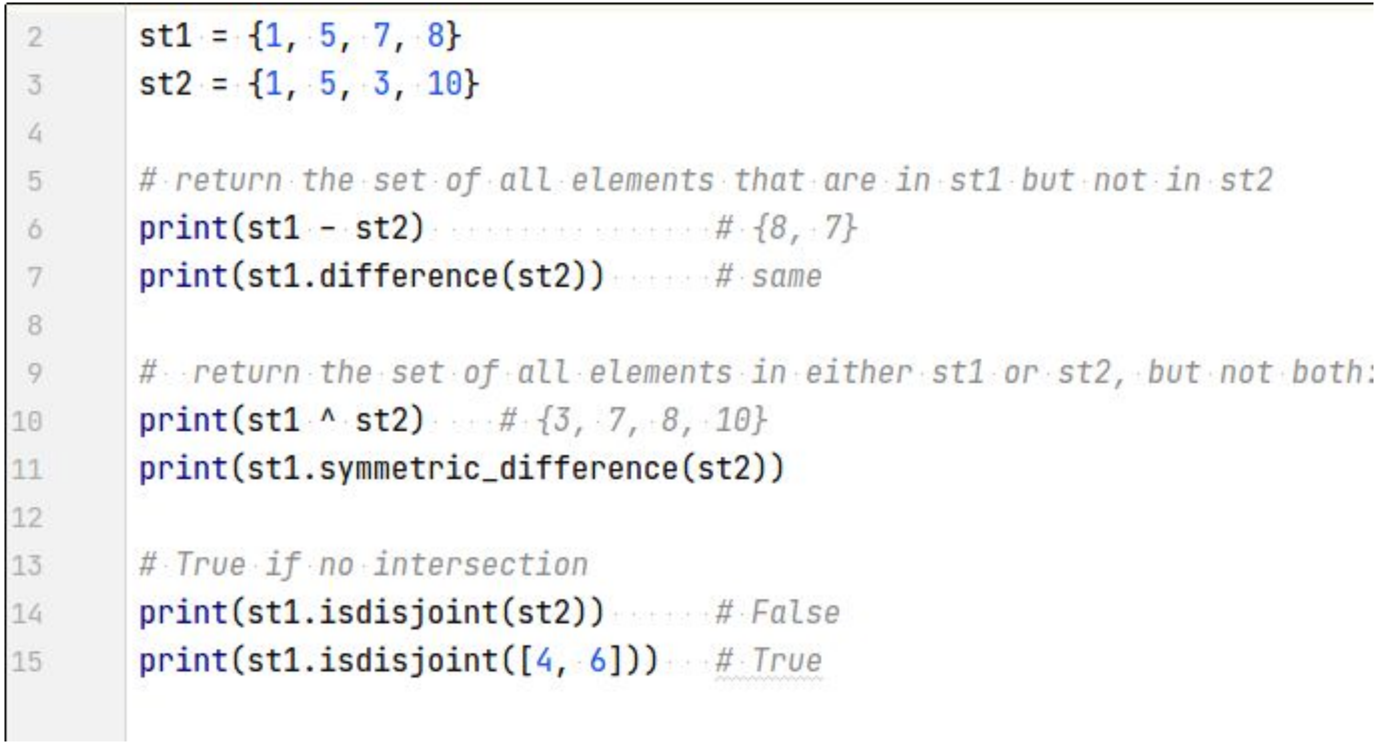


Union and Intersection



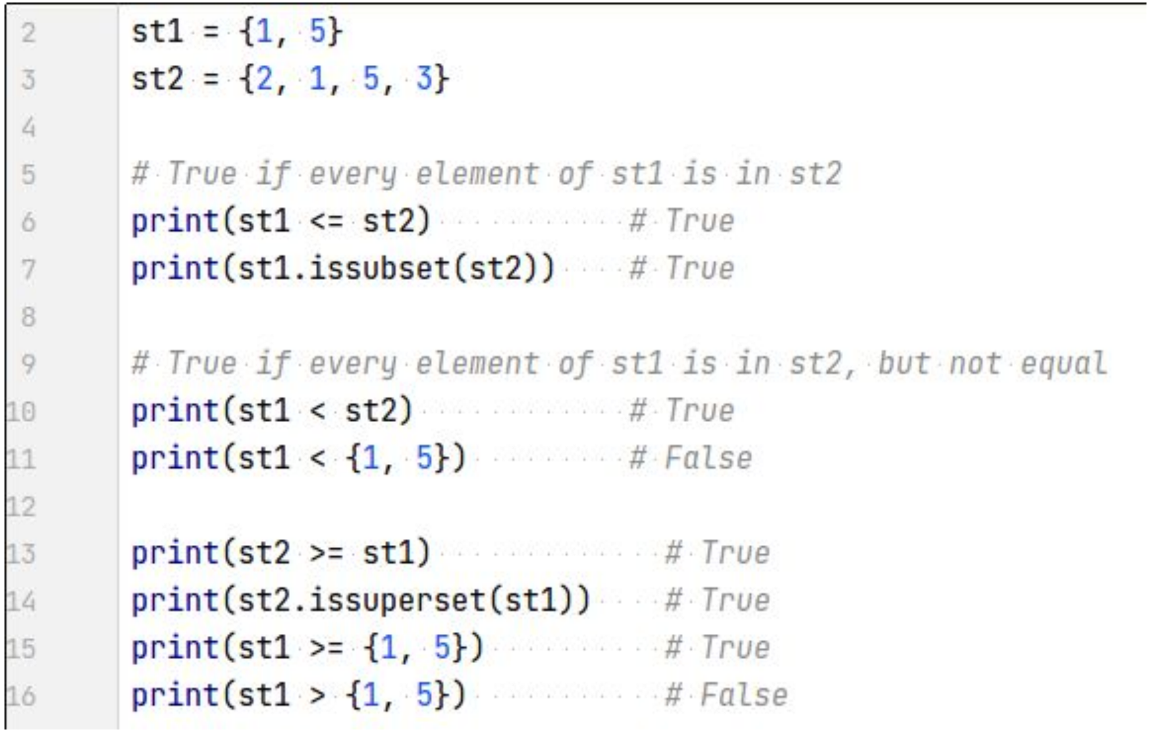
```python
st1 = {1, 5, 7, 8}
st2 = {1, 5, 3, 10}

print(st1 | st2)        # {1, 3, 5, 7, 8, 10}: union using | operator
print(st1.union(st2))   # same
print(st1.union([1, -5, -7]))   # pass any iterable
# note: st1 is not updated

st3 = {5, 6, 1}
su = st1 | st2 | st3
si = st1 & st2 & st3    # set intersection
print(si)   # {1, 5}
print(st1.intersection(st2).intersection(st3))  # {1, 5}
print(st1.intersection(st2, st3))  # {1, 5}

```

Difference



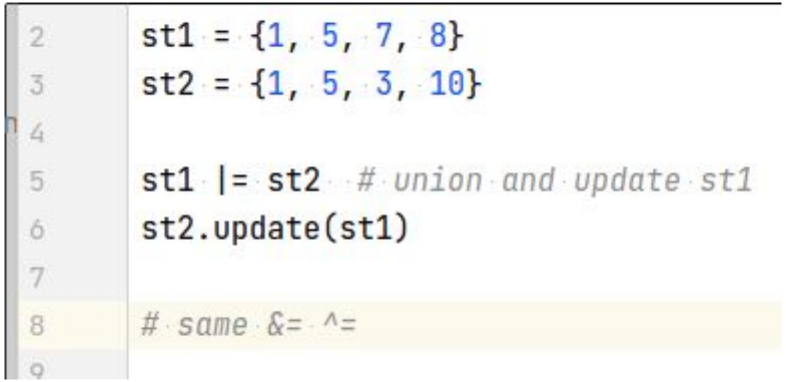
```
st1 = {1, 5, 7, 8}
st2 = {1, 5, 3, 10}

# return the set of all elements that are in st1 but not in st2
print(st1 - st2)                # {8, 7}
print(st1.difference(st2))      # same

#  return the set of all elements in either st1 or st2, but not both:
print(st1 ^ st2)    # {3, 7, 8, 10}
print(st1.symmetric_difference(st2))

# True if no intersection
print(st1.isdisjoint(st2))      # False
print(st1.isdisjoint([4, 6]))   # True
```

Is subset? superset?



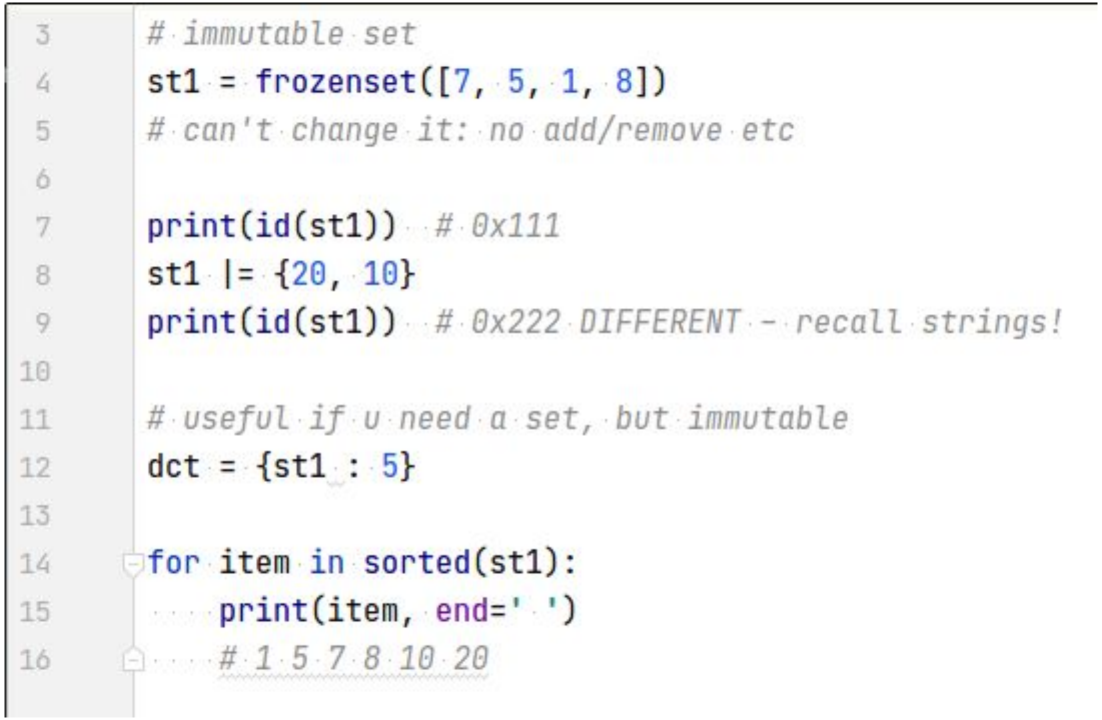
```python
st1 = {1, 5}
st2 = {2, 1, 5, 3}

# True if every element of st1 is in st2
print(st1 <= st2)           # True
print(st1.issubset(st2))    # True

# True if every element of st1 is in st2, but not equal
print(st1 < st2)            # True
print(st1 < {1, 5})         # False

print(st2 >= st1)             # True
print(st2.issuperset(st1))    # True
print(st1 >= {1, 5})          # True
print(st1 > {1, 5})           # False
```

Updates



```
st1 = {1, 5, 7, 8}
st2 = {1, 5, 3, 10}

st1 |= st2  # union and update st1
st2.update(st1)

# same &= ^=

```

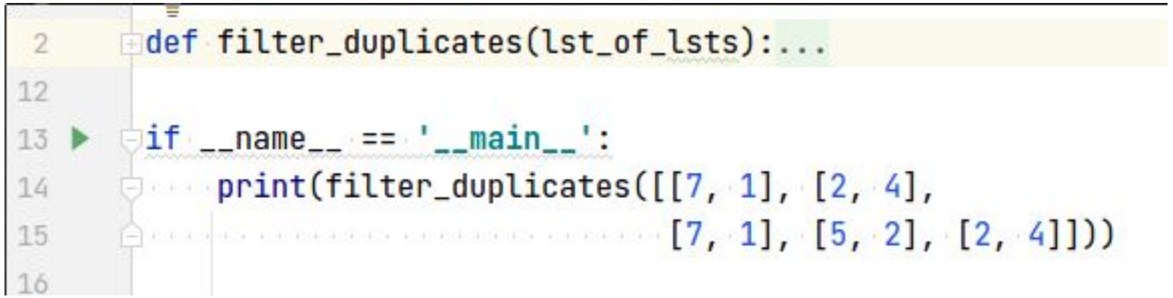
frozenset



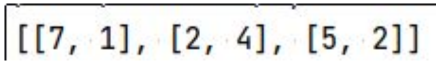
```python
# immutable set
st1 = frozenset([7, 5, 1, 8])
# can't change it: no add/remove etc

print(id(st1))  # 0x111
st1 |= {20, 10}
print(id(st1))  # 0x222 DIFFERENT - recall strings!

# useful if u need a set, but immutable
dct = {st1 : 5}

for item in sorted(st1):
    print(item, end=' ')
    # 1 5 7 8 10 20
```

Practice: Filter Duplicates v2!

- Write function: `def filter_duplicates(lst):`
    - Input is list of list of integers
    - Output: A new list after removing all duplicate lists
    - You don’t need to preserve order!

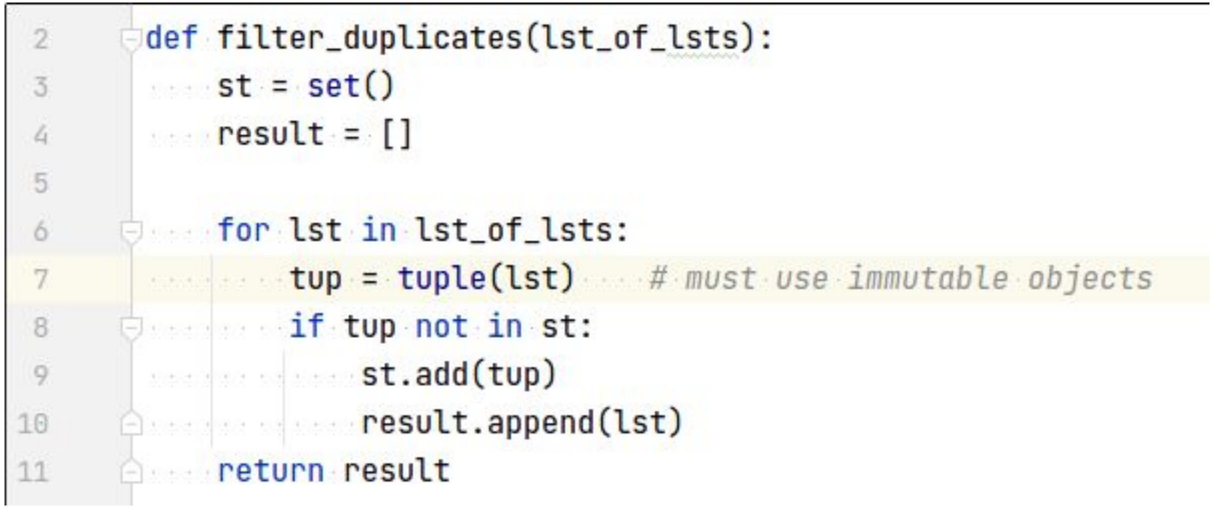


```python
def filter_duplicates(lst_of_lsts):...

if __name__ == '__main__':
    print(filter_duplicates([[7, 1], [2, 4],
                             [7, 1], [5, 2], [2, 4]]))
```



```
[[7, 1], [2, 4], [5, 2]]
```

Practice: Filter Duplicates v2!



```python
def filter_duplicates(lst_of_lsts):
    st = set()
    result = []

    for lst in lst_of_lsts:
        tup = tuple(lst)    # must use immutable objects
        if tup not in st:
            st.add(tup)
            result.append(lst)
    return result
```

"Acquire knowledge and impart it to the people."

"Seek knowledge from the Cradle to the Grave."