Sets

GITAA
Transforming careers

In this lecture

- Sets
- Create sets
- Modify components
- Set operations
    - Union
    - Intersection
    - Difference
    - Symmetric difference

Sets

Sets

- Set is a collection of distinct objects
    - Do not hold duplicate items
- Stores elements in no particular order
- Created using curly braces { }

Create a set

- Create set ***age*** consisting of age of employees

```
age={56,52,41,63,41}
```

- Print the set

```
In [6]: print(age)
{56, 41, 52, 63}
```

- Removes the duplicate item(s) from the set and returns a set of unique elements as output

Create a set

- Create a set ***employee_name*** consisting of names of employees

```
employee_name={'Ram', 'Preethi','Sathish','John','Nirmal'}
```

- Print the set

```
In [4]: print(employee_name)
{'John', 'Sathish', 'Ram', 'Preethi', 'Nirmal'}
```

- In the above output, elements are ordered differently

Modify set using add( )

- add()- adds element(s) to the existing set at any position
- Adding the name ***Ganesh*** to the existing set ***employee_name***
- Syntax: `set_name.add(object)`

```
employee_name.add('Ganesh')
```

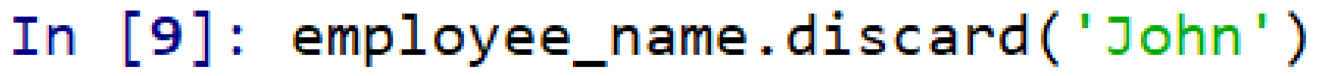
- Print the updated set

```
In [8]: print(employee_name)
{'Ganesh', 'John', 'Sathish', 'Ram', 'Preethi', 'Nirmal'}
```

- In this case Ganesh is added in the 1st position

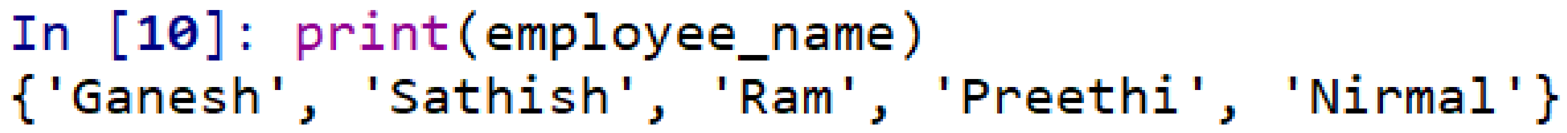
Modify set using discard( )

- **discard()**- removes matching object from an existing set
- Syntax: **set_name.discard(object)**
- Dropping the name ***John*** from ***employee_name***



```
In [9]: employee_name.discard('John')
```

- Print the updated set



```
In [10]: print(employee_name)
{'Ganesh', 'Sathish', 'Ram', 'Preethi', 'Nirmal'}
```

Modify set using clear( )

- `clear()`- removes all the elements from an existing set
- Syntax: `set_name.clear()`

```
In [11]: employee_name.clear()
```

- Print the updated set

```
In [12]: print(employee_name)
set()
```

Set operations

- Create two different sets with different programming languages

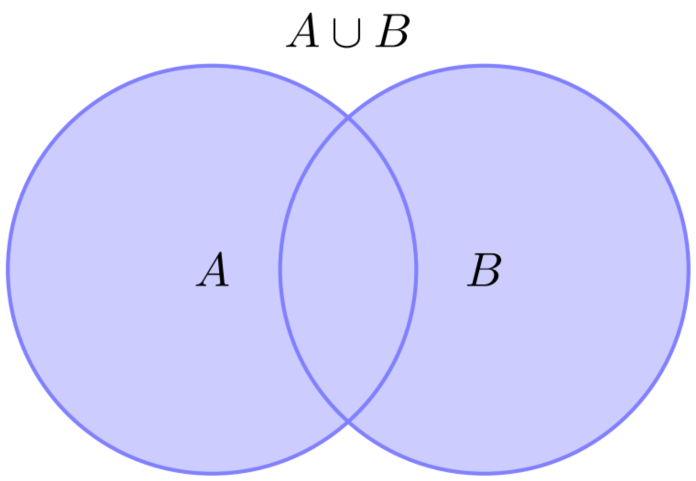
```
Junior_datascientist={'R','Python','Tableau'}
Datascientist={'R','Python', 'scala','Java','Tableau'}
```

- Print the set

```
In [14]: print(Junior_datascientist)
{'Tableau', 'Python', 'R'}

In [16]: print(Datascientist)
{'Tableau', 'R', 'Java', 'scala', 'Python'}
```

Set union

- **union()**- returns all elements belonging to both set A and B
- Syntax: **set_A.union(set_B)**

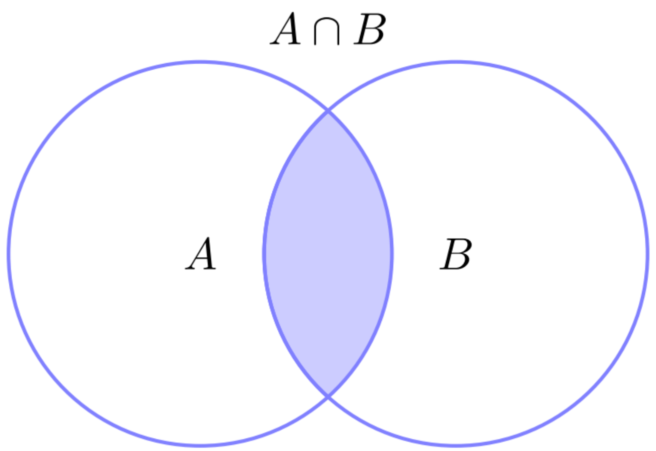


Source: Texample.net

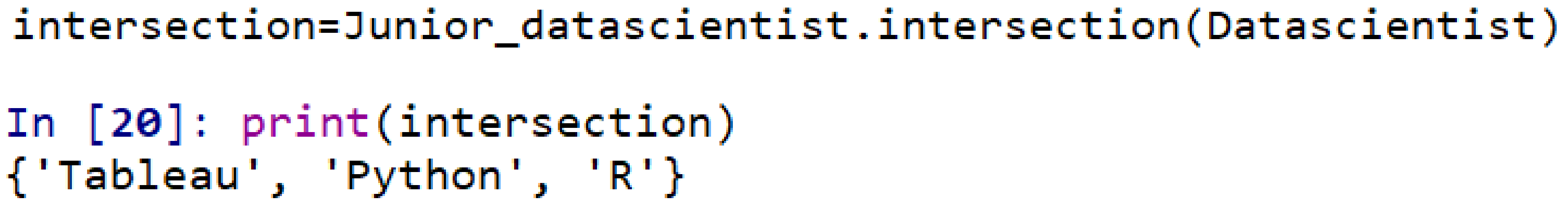
```
union=Junior_datascientist.union(Datascientist)

In [18]: print(union)
{'Tableau', 'R', 'Java', 'scala', 'Python'}
```

Set intersection

- `intersection()`- returns elements common to set A and B
- Syntax: `set_A.intersection(set_B)`



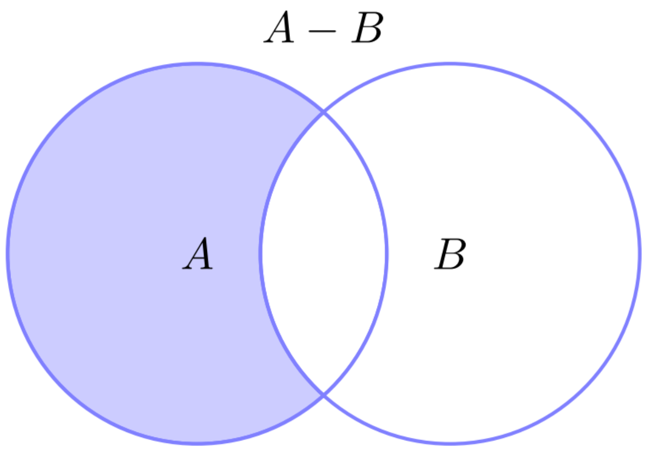
Source: Texample.net



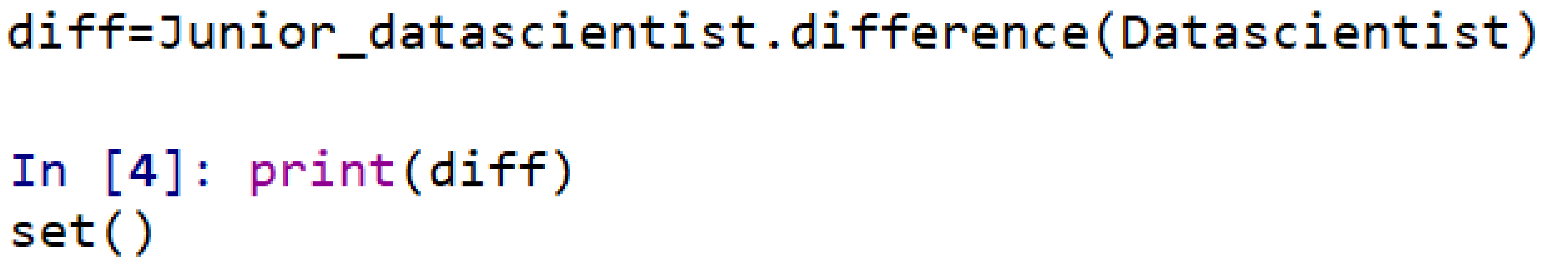
```
intersection=Junior_datascientist.intersection(Datascientist)

In [20]: print(intersection)
{'Tableau', 'Python', 'R'}
```

Set difference

- `difference()`- returns elements belonging to A but not B
- Syntax: `set_A.difference(set_B)`



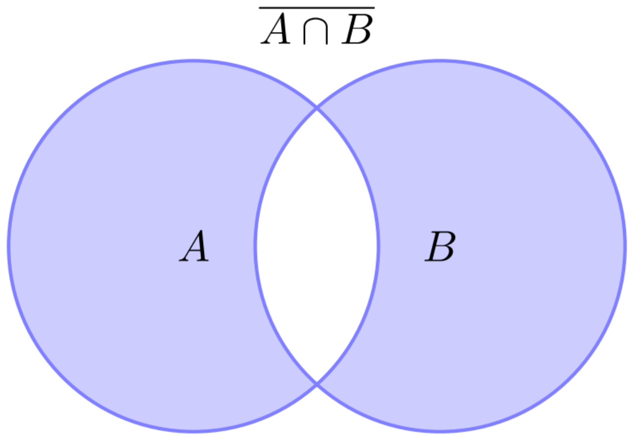
Source: Texample.net



```
diff=Junior_datascientist.difference(Datascientist)

In [4]: print(diff)
set()
```

Symmetric difference

- `symmetric_difference()`- returns elements not common to both sets
- Syntax: `set_A.symmetric_difference(set_B)`



Source: Texample.net

```
symm_diff=Junior_datascientist.symmetric_difference(Datascientist)
```

```
In [6]: print(symm_diff)
{'Java', 'scala'}
```

Summary

- Create sets
- Manipulate set using functions:
  - add - adds element to the existing set
  - discard - removes the specified object from an existing set
  - clear - removes all the elements from the set
- Set operations

```
rror_object
operation == "MIRROR_X":
mirror_mod.use_x = True
mirror_mod.use_y = False
mirror_mod.use_z = False
 _operation == "MIRROR_Y":
mirror_mod.use_x = False
mirror_mod.use_y = True
mirror_mod.use_z = False
 _operation == "MIRROR_Z":
mirror_mod.use_x = False
mirror_mod.use_y = False
mirror_mod.use_z = True

#selection at the end -add
r_ob.select= 1
ier_ob.select=1
ntext.scene.objects.active
("Selected" + str(modifier
mirror_ob.select = 0
 = bpy.context.selected_ob
data.objects[one.name].se

print("please select exactly
```

THANK YOU