

Practical No-20

Title- Sets in Python

What is sets ?

A Set is an unordered collection data type that is iterable, mutable

Creating sets in python

```
var = {"danish", "is", "good"}  
print(type(var))  
print(var)
```

OUTPUT



```
"C:\Users\Danish\PycharmProjects\p program\venv\Scripts\python.exe" "C:/Users/  
<class 'set'>  
{'good', 'is', 'danish'}  
  
Process finished with exit code 0
```

Adding element to python set

CODE

```
myset = {"danish", "is", "good"}  
print(myset)  
myset.add("d")  
print(myset)
```

OUTPUT



```
"C:\Users\Danish\PycharmProjects\p_programm\venv\Scripts\python.exe" "C:/
{'good', 'is', 'danish'}
{'good', 'is', 'danish', 'd'}

Process finished with exit code 0
```

Different operations on Python sets

1) **Union** - Two sets can be merged using union() function or | operator.

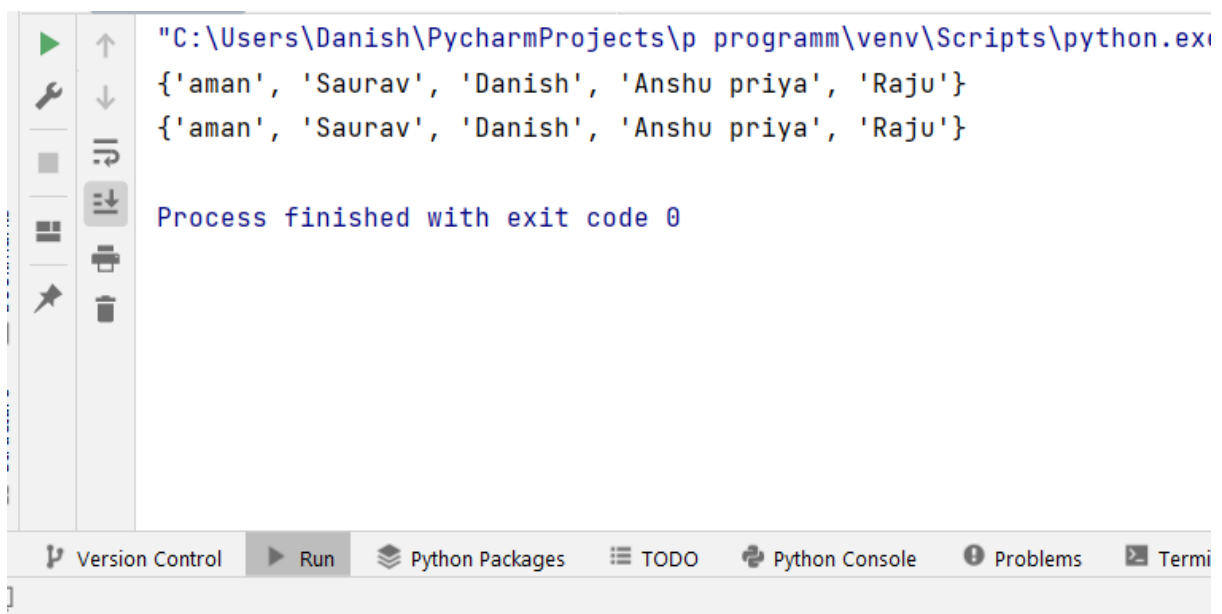
CODE

```
people = {"Danish", "aman", "Saurav"}
vampires = {"simran", "sachin"}
dracula = {"Anshu priya", "Raju"}
population = people.union(vampires)
population = people | dracula
```

```
print(population)
```

```
print(population)
```

OUTPUT



```
"C:\Users\Danish\PycharmProjects\p_programm\venv\Scripts\python.exe"
{'aman', 'Saurav', 'Danish', 'Anshu priya', 'Raju'}
{'aman', 'Saurav', 'Danish', 'Anshu priya', 'Raju'}

Process finished with exit code 0
```

2) Intersection- This can be done through `intersection()` or `&` operator. Common Elements are selected.

CODE

```
x = {"apple", "banana", "cherry"}
y = {"google", "microsoft", "apple"}

z = x.intersection(y)
print(z)
```

OUTPUT



The screenshot shows a terminal window with the following output:

```
"C:\Users\Danish\PycharmProjects\p_programm\venv\Scripts\python.exe" "C:
{'apple'}
```

Below the output, it says "Process finished with exit code 0". The IDE interface includes a toolbar on the left with icons for running, debugging, and other actions, and a bottom toolbar with tabs for "Version Control", "Run", "Python Packages", "TODO", "Python Console", "Problems", and "Terminal".

3) Difference- The difference between the two sets in Python is equal to the difference between the number of elements in two sets.

Example- set A = {10, 20, 30, 40, 80} set B = {100, 30, 80, 40, 60}

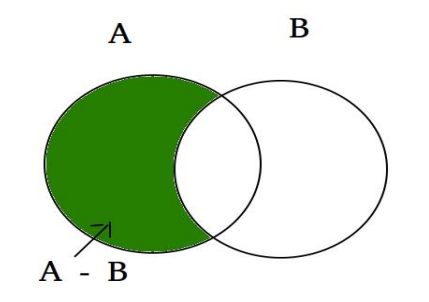
set A - set B = {10, 20}

set B - set A = {100, 60}

Explanation: A - B is equal to the elements present in A but not in B

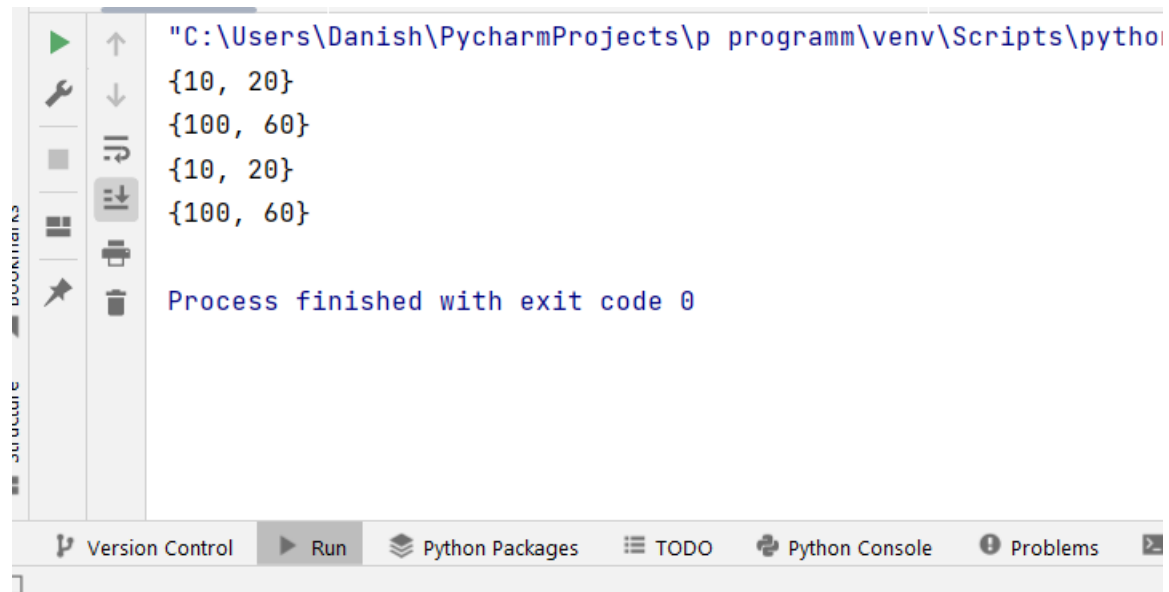
B - A is equal to the elements present in B but not in A

Ven Diagram



CODE

```
A = {10, 20, 30, 40, 80}
B = {100, 30, 80, 40, 60}
print (A.difference(B))
print (B.difference(A))
print (A - B)
print (B - A)
```

OUTPUT

```
"C:\Users\Danish\PycharmProjects\p programm\venv\Scripts\python.exe"
{10, 20}
{100, 60}
{10, 20}
{100, 60}

Process finished with exit code 0
```

The screenshot shows the PyCharm IDE interface. On the left is the 'Toolbox' panel with icons for Run, Debug, Run with Coverage, Test, Run with Profiler, and Run with Coverage and Profiler. The main area displays the output of a Python script. The output shows the results of set difference operations: {10, 20} for A.difference(B), {100, 60} for B.difference(A), {10, 20} for A - B, and {100, 60} for B - A. At the bottom, a message states 'Process finished with exit code 0'. The bottom status bar includes tabs for Version Control, Run, Python Packages, TODO, Python Console, and Problems.

CODE

