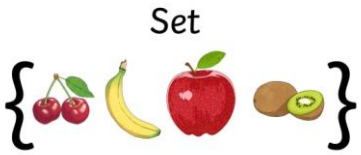
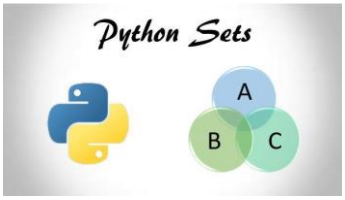
Python Sets





In python a **set** is:

A collection of **unordered** data that **does NOT contain duplicates.**

Sets are normally used when you are concern only with: classifying data, or **identifying if a particular piece of data contained in the collection.**

Python's built-in set type has the following characteristics:

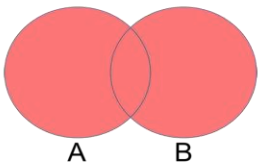
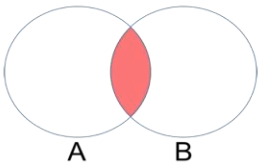
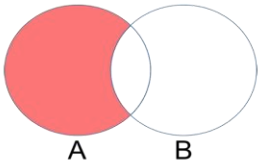
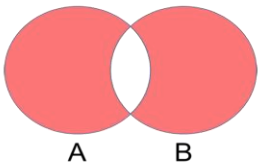
1. Sets are **unordered**.
2. Set elements are **unique**. Duplicate elements are **not** allowed.
3. A set, itself, may be modified, but the elements contained in the set must be of an **immutable** type.

We can use **sets** in Python to compute standard math operations such as:

- **Union**,
- **Intersection**,
- **Difference**,
- **Symmetric difference**.

The image to the right shows a couple standard math operations on two **sets: A** and **B**. The red part of each Venn diagram is the resulting set of a given set operation.

Please make sure you have watch all **three** videos on sets available on the course page!

Set Operation	Venn Diagram	Interpretation
Union		$A \cup B$, is the set of all values that are a member of A, or B, or both.
Intersection		$A \cap B$, is the set of all values that are members of both A and B.
Difference		$A \setminus B$, is the set of all values of A that are not members of B
Symmetric Difference		$A \triangle B$, is the set of all values which are in one of the sets, but not both.

Again, please make sure you have watch all **three** videos on sets available on the course page!

Useful functions/methods you can use on sets in python:

FUNCTION	DESCRIPTION
`add()`	Adds an element to a set
`remove()`	Removes an element from a set. If the element is not present in the set, raise a KeyError
`clear()`	Removes all elements form a set
`copy()`	Returns a shallow copy of a set
`pop()`	Removes and returns an arbitary set element. Raise KeyError if the set is empty
`update()`	Updates a set with the union of itself and others
`union()`	Returns the union of sets in a new set
`difference()`	Returns the difference of two or more sets as a new set
`difference_update()`	Removes all elements of another set from this set
`discard()`	Removes an element from set if it is a member. (Do nothing if the element is not in set)
`intersection()`	Returns the intersection of two sets as a new set
`intersection_update()`	Updates the set with the intersection of itself and another
`isdisjoint()`	Returns True if two sets have a null intersection
`issubset()`	Returns True if another set contains this set
`issuperset()`	Returns True if this set contains another set
`symmetric_difference()`	Returns the symmetric difference of two sets as a new set
`symmetric_difference_update()`	Updates a set with the symmetric difference of itself and another

Exercises:

Note: You can NOT create an **empty** set using **`set1={}.`**
This will create a **dictionary** (another data type).
You must do something like `set1=set()`
For some of the following exercises **you can use the code below to help you generate a random** set **of numbers**. The code below generates a set of 20 numbers (each randomly generated between 1 and 30).

Exercise#1 Try it out!

```
import random
x=0
set1=set()
while x < 20:
  new=random.randint(1,30)
  set1.add(new)
  x=x+1
print(set1)
```

2. Create a **list** of numbers called `list1`. Make sure the list has at least a few numbers that are repeated. Example: `list1 = [18,1,14,15,1,18,2,3,4,4,5].` Now use the `set()` method to turn the ***list*** into a ***set*** and print it out on the screen.

3. Use the code from **exercise 1** to generate a ***set*** of numbers. Write a program that will then let the ***user*** **add** elements to the set.

4. Write a program that will ***allow a user*** to **remove** elements from a set.

5. Write a program that will remove all the duplicates from a ***list*** by turning it into a ***set.***

6. Find out the number of unique items in a ***list*** by turning it into a set and then using the `len()` function.

7. Given a following two sets, write a program that will find the **intersection** between them and remove those elements found in the intersection from the first set.

```
First Set  {65, 42, 78, 83, 23, 57, 29}
Second Set  {67, 73, 43, 48, 83, 57, 29}
```

8. Use the code at the beginning of the assignment to create two sets randomly generated sets of number: Set1 and Set2. then determine:

a) if Set1 a **Subset** of Set2
b) if Set1 a **Superset** Set2
c) if {1,5} a **subset** of Set1
d) if {1,5} a **subset** of Set2
e) if Set2 a **superset** of {3,9,20}
f) what the **common elements** are between the two sets
g) A set of all numbers that are found in either Set1 or Set2 but **not both** (symmetric difference). Look it up. Know what it means!
h) A set that is the **union** of Set1 and Set2

9. Write a Python program to find the common elements in **5** sets generated with the code at the beginning of the assignment.

10. Find whether a given string has unique characters (using sets)

11. Create a new set that is the **difference** between the following two sets. You can copy and paste the sets into Replit if you wish.

A = {1, 2, 3, 4, 5}
B = {4, 5, 6, 7, 8}

12. Create a **union** of the follow 3 sets: *copy and paste the sets in to Replit if you like.*

Math class = {'ryan', 'beth', 'tony', 'rick', 'karen'}
Science class ={'cory,'denis','beth', 'rick', 'scott', 'steve','karen}
English class = {'beth','karen','steve','dave','tina','skinner','william'}

13. You are responsible for programming a scanner that is used to **create a list of all the different colors of paint that are needed repaint a house**. As you scan the walls and fixtures around the house, the scanner takes input in the form of different colors, for example: red, yellow, blue, brown, yellow, yellow, yellow, green, blue, etc.

As you scan around the house **the scanner may scan the same color twice** (window frames, baseboards, etc. may happen to have the same color)

- Create a program that generates **list** of only 8 colors (selected **randomly** from the **list** of the 12 different colors shown below).

(black, blue, brown, green, grey, orange, pink, purple, red, white, yellow, emerald)

This list of 8 colors **will be the input for your scanner.**

- Now create a **set** of *only* the **unique** colors you will *need* to paint the house.
- Now "scan" 2 more houses (by generating two more ***lists*** of 8 colors) and create a new **set** that contains only the colors needed to **paint all three houses.**
- Are there any "unpopular" colors that where **not used in *any* of the houses**? Create a program that will generate a **set** of these "unpopular" colors once all three houses have been scanned.

14. A school decides to partner students together with "study buddies" from the other school based on exam scores. Create two **sets** (each with 30 elements) of randomly generated numbers from 80 to 100.

    Then find all the students grades that **do NOT** have a partner (create a **set** of the grades **who do not have a matching grade in their partner school**).

Complete the following University of Waterloo contest problems. (Please attempt each using sets)

Exercise#15

Problem J3: From 1987 to 2013

Problem Description
You might be surprised to know that 2013 is the first year since 1987 with distinct digits. The years 2014, 2015, 2016, 2017, 2018, 2019 each have distinct digits. 2012 does not have distinct digits, since the digit 2 is repeated.

Given a year, what is the next year with distinct digits?

Input Specification
The input consists of one integer Y ($0 \leq Y \leq 10000$), representing the starting year.

Output Specification
The output will be the single integer D, which is the next year after Y with distinct digits.

Sample Input 1
```
1987
```

Output for Sample Input 1
```
2013
```

Sample Input 2
```
999
```

Output for Sample Input 2
```
1023
```

Exercise#16

Problem J2: Rotating letters

Problem Description
An artist wants to construct a sign whose letters will rotate freely in the breeze. In order to do this, she must only use letters that are not changed by rotation of 180 degrees: I, O, S, H, Z, X, and N.

Write a program that reads a word and determines whether the word can be used on the sign.

Input Specification
The input will consist of one word, all in uppercase letters, with no spaces. The maximum length of the word will be 30 letters, and the word will have at least one letter in it.

Output Specification
Output `YES` if the input word can be used on the sign; otherwise, output `NO`.

Sample Input 1
```
SHINS
```

Output for Sample Input 1
```
YES
```

Sample Input 2
```
NOISE
```

Output for Sample Input 2
```
NO
```