

- 
- A blue parallelogram and a light green parallelogram are positioned in the upper-left corner of the slide. The blue shape is partially behind the green one. Both shapes have a black outline and are set against a dark blue background with diagonal stripes.
- **Discussion On Assignments**
 - **Practice Problems**



map() in Python

Is a built-in function that allows you to process and transform all the items in an iterable without using an explicit `for loop`, a technique commonly known as `mapping`. `map()`

Is useful when you need to apply a transformation function to each item in an iterable and transform them into a new iterable. `map()` is one of the tools that support a `functional programming style` in Python.



Syntax of map()

Python

```
map(function, iterable[, iterable1, iterable2,..., iterableN])
```

You can use any built-in function with `map()`, provided that the function takes an argument and returns a value.



Filter() in Python

The `filter()` function returns an iterator where the items are filtered through a function to test if the item is accepted or not.

Syntax

```
filter(function, iterable)
```


Example

```
numbers = [1, 2, 3, 4, 5, 6, 7]
```

```
# the lambda function returns True for even numbers  
even_numbers_iterator = filter(lambda x: (x%2 == 0),  
numbers)
```

```
# converting to list  
even_numbers = list(even_numbers_iterator)
```

```
print(even_numbers)
```

- 
- convert all the items in a list from a string to an integer number. use `map()` along with `int()`
 - **Processing Multiple Input Iterables With `map()`**



List comprehension is more concise and easier to read as compared to map.

List comprehension are used when a list of results is required as map only returns a map object and does not return any list. Map is faster in case of calling an already defined function (as no lambda is required)

List comprehension has a simpler configuration than the map function.

List comprehension can be used together with if condition as replacement of filter method. Map function has no such functionality.

List comprehension returns a list, whereas the map function returns an object of Iterable.

List comprehension execution is faster than that of map function when the formula expression is huge and complex.

Map function is faster than list comprehension when the formula is already defined as a function earlier. So, that map function is used without lambda expression.