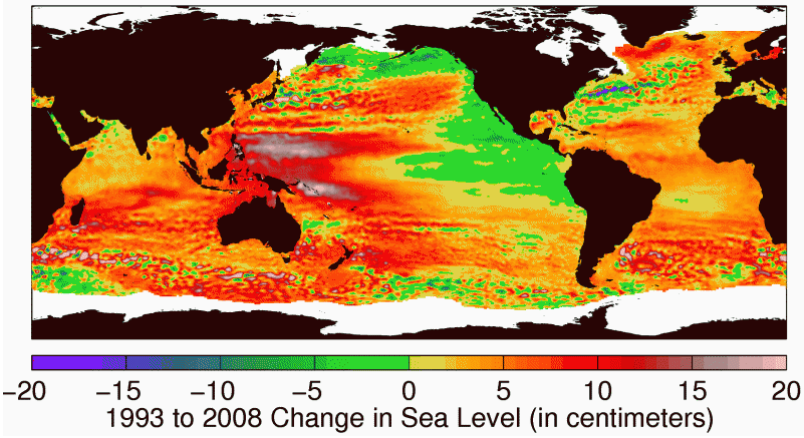
Lesson 3 - Tuples & Lists

Storing collections of items

Melting Glaciers

As global average temperatures increase, glaciers, ice sheets and ice caps melt and contribute towards rising sea levels. Studies show that between the years 1900 and 1990, the sea level rose at a rate of 1.7 mm/year. By 2000, the rate had almost doubled to 3.2 mm/year. The Greenland Ice Sheet is in particular melting at an unprecedented rate, and as it melts, poses a serious risk of flooding many coastal cities, like London, New York, and Sydney.

However, as glaciers and ice sheets melt, the melted ice is not evenly distributed across the globe. This leads to three processes that influence the extent to which a place is impacted, referred to as their "sea-level fingerprint". Firstly, when ice shrinks, the force of gravity on the land underneath decreases, causing water to be pushed away from that land mass. This then causes the land underneath that melting ice to expand since it is not as compressed by the ice. Finally, as the Earth spins and wobbles, otherwise known as precession, water becomes redistributed across the globe. While many countries have invested in stabilizing natural barriers along coastlines such as trees and vegetation and maintaining artificial barriers like seawalls (which now cover up to 60% of China's mainland coast), not all countries have the funds to implement measures such as creating or stabilizing barriers. Hence, there is a growing need to develop computational systems that can forecast sea-level rise.



Source:
The Ocean Portal Team Reviewed by Dr. Joshua K. Willis. (2019, August 05). Sea Level Rise. Retrieved December 03, 2020, from https://ocean.si.edu/through-time/ancient-seas/sea-level-rise

Tuples and **lists** are both collections of items that can be accessed through an **index** (a position in an ordered list). The difference between tuples and lists include:

1. **Syntax**: lists use square brackets [ ] and tuples use parentheses ( )
2. **Mutable/Immutable**: values can be changed/modified in lists, but not tuples.
3. **Space/processing speed:** since tuples are immutable, they take up less space and operations executed in a tuple are faster.

Part 1: Create a tuple

1. Use parentheses ( ) to create a tuple for the coastal cities New York, London, and Sydney.

   Coastal_cities = (“New York”, “London”, “Sydney”)

2. Print the tuple and use the type function to classify it.

Example:

main.py

```
1  coastal_cities = ("New York", "London", "Sydney")
2  print(coastal_cities)
3  print(type(coastal_cities))
```

```
('New York', 'London', 'Sydney')
<class 'tuple'>
```

Part 2: Index the tuple

1. Using the tuple from part 1, an index will be used to call one string individually. Each string correlates to an index number, beginning with 0.

String:	New York	London	Sydney
Index:	0	1	2

2. Call the string, “New York”, by referencing its position using square brackets [ ] and use the print function.

   print (coastal_cities [0])

Example:

main.py

```
1  coastal_cities = ("New York", "London", "Sydney")
2  print(coastal_cities[0])
```

```
New York
```

✅***Task 1: create a tuple for the sea level rising rates (1.7 mm/year, 3.2 mm/year) in the article and include your prediction for the future as the 3rd value in the tuple. Print the value indexed at 1.***

Part 3: Create a list and append it

1. Using the same data as the tuple in the last task, create a list using square brackets: [ ].
2. To **append** the list (add a value) place a period at the end of the name of the variable, followed by append and the value "12.0".
3. Print the list and identify the type of data.

Example:

main.py

```
1  sea_level_rising_rate = [1.7, 3.2, 6.4]
2  sea_level_rising_rate.append(12.0)
3  print(sea_level_rising_rate)
4  print(type(sea_level_rising_rate))
```

```
[1.7, 3.2, 6.4, 12.0]
<class 'list'>
```

Part 4: Use a negative index for the list

1. Using the same list as in task 1, a **negative index** will be used to refer to the position of a value, starting from the last value in the list as -1.

Float:	1.7	3.2	6.4	12.0
Negative Index:	-4	-3	-2	-1

✅***Task 2: find the value 3.2 using the print and negative index functions.***

Part 5: Slice a list

1. To slice items out of the list (select a range of index numbers), use a colon : and square brackets: [ ].

   sea_level_rising_rate = [1.7, 3.2, 6.4, 12.0]
2. Select all the values up until **(but not including)** position 3 by placing the colon before the 3.

   print (sea_level_rising_rate [:3])
3. Do the reverse function, by placing the colon after the 3, to include the value at position 3 and after position 3 **(inclusive)**.

   print (sea_level_rising_rate [3:])

✅*Task 3: create a slice that includes the values 3.2 and 6.4 only.*