Untitled-1

```
# %% [markdown]
# MUHAMMAD ABDULLAH
#
# B22F0577AI054
#
# SUBMITTED TO: MAM ANILA HABIB
#
# PROGRAMMING FOR AI LAB
#
# AI LAB:12

# %% [markdown]
# #Python Lists

# %% [markdown]
# Lists are ordered collection of data items. They store multiple items in a single variable. List items are separated by commas and enclosed within square brackets []. Lists are changeable meaning we can alter them after creation.

# %%
lst1 = [1,2,2,3,5,4,6]
lst2 = ["Red", "Green", "Blue"]
print(lst1)
print(lst2)

# %%
details = ["Abdullah", 18, "FYBScIT", 9.8]
print(details)

# %% [markdown]
# List Indexes
# Each item/element in a list has its own unique index. This index can be used to access any particular item from the list. The first item has index [0], second item has index [1], third item has index [2] and so on.

# %%
colors = ["Red", "Green", "Blue", "Yellow", "Green"]
     #      [0]      [1]     [2]      [3]      [4]

# %% [markdown]
# Accessing list items:

# %% [markdown]
# Positive Indexing:
# As we have seen that list items have index, as such we can access items using these indexes.

# %%
colors = ["Red", "Green", "Blue", "Yellow", "Green"]
#          [0]      [1]     [2]      [3]      [4]
print(colors[2])
print(colors[4])
print(colors[0])

# %% [markdown]
# Negative Indexing:
```

```

# %% [markdown]
# Similar to positive indexing, negative indexing is also used to access items, but from the end of the list. The last item has index [-1], second last item has index [-2], third last item has index [-3] and so on

# %%
colors = ["Red", "Green", "Blue", "Yellow", "Green"]

#[-5] [-4] [-3] [-2] [-1]
print(colors[-1])
print(colors[-3])
print(colors[-4])

# %% [markdown]
# Check for item:
# We can check if a given item is present in the list. This is done using the in keyword.

# %%
colors = ["Red", "Green", "Blue", "Yellow", "Green"]
if "Yellow" in colors:
    print("Yellow is present.")
else:
    print("Yellow is absent.")

# %% [markdown]
# Add List Items

# %% [markdown]
# There are three methods to add items to list: append(), insert() and extend()
#
# append(): This method appends items to the end of the existing list.

# %%
colors = ["Red", "Green", "Blue", "Yellow", "Green"]
if "Orange" in colors:
    print("Orange is present.")
else:
    print("Orange is absent.")

# %% [markdown]
# What if you want to insert an item in the middle of the list? At a specific index?

# %% [markdown]
# insert():
# This method inserts an item at the given index. User has to specify index and the item to be inserted within the insert() method

# %%
animals = ["cat", "dog", "bat", "mouse", "pig", "horse", "donkey", "goat", "cow"]
print(animals[3:7]) #using positive indexes
print(animals[-7:-2])   #using negative indexes

# %% [markdown]
# What if you want to append an entire list or any other collection (set, tuple, dictionary) to the existing list?

```

```
# %% [markdown]
# extend():
# This method adds an entire list or any other collection datatype (set, tuple, dictionary) to the existing list.

# %%
colors = ["voilet", "indigo", "blue"]
colors.append("green")
print(colors)

# %%
colors = ["voilet", "indigo", "blue"]
#           [0]        [1]      [2]

colors.insert(1, "green")   #inserts item at index 1
# updated list: colors = ["voilet", "green", "indigo", "blue"]
#       indexs              [0]       [1]       [2]      [3]

print(colors)

# %% [markdown]
# concatenate two lists:

# %% [markdown]
# you can simply concatenate two list to join two lists.

# %%
#add a list to a list
colors = ["voilet", "indigo", "blue"]
rainbow = ["green", "yellow", "orange", "red"]
colors.extend(rainbow)
print(colors)

# %% [markdown]
# Remove List Items
#
# There are various methods to remove items from the list: pop(), remove(), del(), clear()
#
# pop(): This method removes the last item of the list if no index is provided. If an index is provided, then it removes item at that specified index.

# %%
#add a tuple to a list
cars = ["Hyundai", "Tata", "Mahindra"]
cars2 = ("Mercedes", "Volkswagen", "BMW")
cars.extend(cars2)
print(cars)

# %%
colors = ["voilet", "indigo", "blue", "green"]
colors2 = ["yellow", "orange", "red"]
print(colors + colors2)

# %% [markdown]
# remove():
# This method removes specific item from the list.

```

```
# %%
colors = ["Red", "Green", "Blue", "Yellow", "Green"]
colors.pop()        #removes the last item of the list
print(colors)

# %% [markdown]
# del:
# del is not a method, rather it is a keyword which deletes item at specific from the list, or deletes the list entirely.

# %%
colors = ["Red", "Green", "Blue", "Yellow", "Green"]
colors.pop(1)       #removes item at index 1
print(colors)

# %% [markdown]
# What if we don’t want to delete the entire list, we just want to delete all items within that list?

# %% [markdown]
# clear():
# This method clears all items in the list and prints an empty list.

# %%
colors = ["voilet", "indigo", "blue", "green", "yellow"]
colors.remove("blue")
print(colors)

# %% [markdown]
# Change List Items

# %% [markdown]
# Changing items from list is easier, you just have to specify the index where you want to replace the item with existing item.

# %%
colors = ["voilet", "indigo", "blue", "green", "yellow"]
del colors[3]
print(colors)

# %%
colors = ["voilet", "indigo", "blue", "green", "yellow"]
colors.clear()
print(colors)

# %% [markdown]
# List Comprehension

# %% [markdown]
# List comprehensions are used for creating new lists from other iterables like lists, tuples, dictionaries, sets, and even in arrays and strings.
#
# Syntax:
#
# List = [expression(item) for item in iterable if condition]
#
# expression: it is the item which is being iterated.
```

```
#
# iterable: it can be list, tuples, dictionaries, sets, and even in arrays and strings.
#
# condition: condition checks if the item should be added to the new list or not.

# %% [markdown]
# Example 1: accepts items with the small letter “o” in the new list

# %%
names = ["Harry", "Sarah", "Nadia", "Oleg", "Steve"]
names[2] = "Millie"
print(names)

# %%
names = ["Harry", "Sarah", "Nadia", "Oleg", "Steve"]
names[2:4] = ["juan", "Anastasia"]
print(names)

# %% [markdown]
# List Methods

# %% [markdown]
# We have discussed methods like append(), clear(), extend(), insert(), pop(), remove() before. Now we will learn about some more list methods:

# %% [markdown]
# sort(): This method sorts the list in ascending order

# %%
colors = ["voilet", "indigo", "blue", "green"]
colors.sort()
print(colors)

num = [4,2,5,3,6,1,2,1,2,8,9,7]
num.sort()
print(num)

# %%
colors = ["voilet", "indigo", "blue", "green"]
colors.sort(reverse=True)
print(colors)

num = [4,2,5,3,6,1,2,1,2,8,9,7]
num.sort(reverse=True)
print(num)

# %% [markdown]
# reverse(): This method reverses the order of the list.

# %%
colors = ["voilet", "indigo", "blue", "green"]
colors.reverse()
print(colors)

num = [4,2,5,3,6,1,2,1,2,8,9,7]
num.reverse()
```

```
print(num)

# %% [markdown]
# copy(): Returns copy of the list. This can be done to perform operations on the list without modifying the original list.

# %%
colors = ["voilet", "green", "indigo", "blue"]
newlist = colors.copy()
print(colors)
print(newlist)
```