অধ্যায়-৮

লিস্ট
(List)

৮.০ ভূমিকা (Introduction) ঃ

পাইথনে ছয় ধরনের বিল্ট ইন টাইপ ডাটা স্ট্রাকচার আছে। সেগুলো হচ্ছে– numeric, sequence, mapping, class, instance এবং exception. এর মধ্যে বহুল ব্যবহৃত বেসিক ডাটা স্ট্রাকচার হচ্ছে sequence। এর প্রত্যেকটি এলিমেন্টের জন্য একটি নাম্বার অ্যাসাইন করা হয়, যাকে ইনডেক্স বা পজিশন বলা যায়। প্রথম ইনডেক্স শূন্য, তারপর ১ এবং এরপর ক্রমানুযায়ী মান বাড়তে থাকে। পাইথনে আবার তিন ধরনের বেসিক sequence টাইপ ডাটা স্ট্রাকচার আছে। যেমন– list, tuple, এবং xrange object। পাইথনে লিস্ট হলো একটা স্মার্ট ডাটা টাইপ। কারণ, প্রোগ্রামিং এ লিস্টের ব্যবহারিক প্রয়োগ অনেক বেশি। সি, সি++ বা জাভা ল্যাংগুয়েজে যাকে অ্যারে বলা হয় পাইথনে তাকেই লিস্ট বলে। তবে অ্যারের চেয়ে লিস্টের প্রায়োগিক সক্ষমতা অনেক বেশি। উক্ত অধ্যায়ে আমরা লিস্ট ও তার ব্যবহার সম্পর্কিত বিষয়াদি নিয়ে বিশদ আলোচনা করব।

৮.১ লিস্ট ও লিস্টের উপাদান (Define list and its elements type) ঃ

লিস্ট (List) ঃ List হচ্ছে পাইথনের সবচেয়ে বৈচিত্রপূর্ণ ডাটা টাইপ, যা স্কয়ার ব্র্যাকেট [ ] এর ভেতর কমার সাহায্যে উপাদানসহ প্রকাশ করা হয়। অন্যান্য ল্যাংগুয়েজে যাকে অ্যারে বলা হয় পাইথনে তাকেই লিস্ট বলে। তবে List এর উপাদানগুলো অ্যারের মতো একই রকম ডাটা টাইপ হবার প্রয়োজন নেই। যেমন–

```
list1 = ['physics', 'chemistry', 1997, 2000];
list2 = [1, 2, 3, 4, 5 ];
list3 = ["a", "b", "c", "d"];
```

লিস্টের এলিমেন্টেগুলোকে কমা দিয়ে দিয়ে যুক্ত করা হয় এর এলিমেন্টগুলো ইনডেক্স অনুযায়ী (0,1,2,3.....) সাজানো থাকে। ইনডেক্সের মান সবসময় 0 থেকে শুরু হয়। লিস্টকে sliced, concatenated ইত্যাদি করা যায়।

লিস্টের এলিমেন্ট টাইপ (Elements types of list) ঃ পাইথনে লিস্টের মধ্যে বিভিন্ন টাইপের ডাটা বা এলিমেন্ট রাখা যেতে পারে। যেমন–

- ইন্টিজার টাইপ ডাটা
- ফ্লোটিং পয়েন্ট টাইপ ডাটা
- স্ট্রিং
- লিস্ট
- টাপল
- ডিকশনারি ও
- মিক্সড ডাটা টাইপ

লিস্ট ও ইন্টিজার টাইপ ডাটা (List & Integer type data) ঃ শুধুমাত্র ইন্টিজার সংখ্যা ব্যবহার করে যে লিস্ট গঠিত হয় তাকে ইন্টিজার টাইপের লিস্ট বলে।

উদাহরণ ঃ

```
>>> Numbers=[5,7,4,6,3,18,1,9]
>>> print (Numbers)
[5, 7, 4, 6, 3, 18, 1, 9]
>>> print (Numbers[5])
18
>>> print (Numbers[3])
6
>>> Numbers.sort()
>>> print(Numbers)
[1, 3, 4, 5, 6, 7, 9, 18]
>>>
```

লিস্ট ও ফ্লোটিং পয়েন্ট টাইপ ডাটা (List & floating point type data) ঃ শুধুমাত্র ফ্লোটিং পয়েন্ট টাইপ সংখ্যা ব্যবহার করেও লিস্ট তৈরি করা যায়। যেমন–

```
>>> Numbers=[5.4,3.7,10.4,25.6,33.7,7.18,172.34,97.99]
>>> print (Numbers)
[5.4, 3.7, 10.4, 25.6, 33.7, 7.18, 172.34, 97.99]
>>> print (Numbers[3])
25.6
>>> Numbers.sort()
>>> print(Numbers)
[3.7, 5.4, 7.18, 10.4, 25.6, 33.7, 97.99, 172.34]
>>>
```

লিস্ট ও স্ট্রিং (List & String) ঃ একাধিক স্ট্রিং এর সমন্বয়ে পাইথনে লিস্ট তৈরি করা যায়। যেমন–

```
>>> Names=["Fatima","Rahima","Sumaiya","Tahsin","Tanveer"]
>>> print(Names[1],Names[0],Names[3])
Rahima Fatima Tahsin
>>>
```

লিস্টের মধ্যে লিস্ট (List within list) ঃ লিস্টের মধ্যেও লিস্ট তৈরি করা যায়। যেমন–

```
list1 = ['physics', 'chemistry', 1997, 2000];
list2 = [1, 2, 3, 4, 5 ];
list3 = ["a", "b", "c", "d"];
```

মিক্সড ডাটা টাইপ ও লিস্ট (List & Mixed type data) ঃ একাধিক টাইপের ডাটাকে (যেমন– ইন্টিজার, ফ্লোট, স্ট্রিং) একত্র করে ও লিস্ট তৈরি করা যায়।

উদাহরণ ঃ

```
>>> MyList=["Tanveer",2017,99.5]
>>> print(MyList)
['Tanveer', 2017, 99.5]
>>> print(MyList[2])
99.5
```

৮.২ লিস্টের উপাদান অ্যাক্সেসিং (Accessing values from list) ঃ

লিস্টের উপাদানসমূহকে অ্যাক্সেস করার জন্য পাইথনে বিভিন্ন পদ্ধতি প্রচলিত আছে। পদ্ধতিসমূহ হচ্ছে ঃ

- ইনডেক্সিং (Indexing) পদ্ধতি
- নেগেটিভ ইনডেক্সিং (Negative Indexing) পদ্ধতি এবং
- স্লাইসিং (Slicing) পদ্ধতি

নিম্নে পদ্ধতিসমূহ আলোচনা করা হলো ঃ

ইনডেক্সিং (Indexing) পদ্ধতি ঃ যে পদ্ধতিতে লিস্টের কোনো আইটেমকে অ্যাক্সেস করার জন্য ইনডেক্স নম্বর ব্যবহার করা হয় তাকে ইনডেক্সিং পদ্ধতি বলে। লিস্টের ইনডেক্স নম্বর শুরু হয় 0 থেকে।

উদাহরণ-১ ঃ

```
my_list = ['a','p','p','L','e']
print(my_list[0])
print(my_list[2])
print(my_list[4])
n_list = ["Happy", [2,0,1,5]]
print(n_list[0][1])
print(n_list[1][3])
```

আউটপুট ঃ

```
a
p
e
a
5
>>>
```

উদাহরণ-২ ঃ

```
b = ['Mahi', 'Mahbub', 'Mahfuz', 'Samsul', 96002, 88.87]
print(b[0])
print(b[1])
print(b[1:5])
print(b[:5])
print(b[2:])
```

আউটপুট ঃ

```
Mahi
Mahbub
['Mahbub', 'Mahfuz', 'Samsul', 96002]
['Mahi', 'Mahbub', 'Mahfuz', 'Samsul', 96002]
['Mahfuz', 'Samsul', 96002, 88.87]
>>>
```

নেগেটিভ ইনডেক্সিং (Negative Indexing) পদ্ধতি ঃ পাইথনে নেগেটিভ ইনডেক্সিং এর ব্যবহারও রয়েছে। এক্ষেত্রে ইনডেক্স –1 দ্বারা সর্বশেষ আইটেম এবং ইনডেক্স-2 দ্বারা দ্বিতীয় সর্বশেষ আইটেমকে বুঝায়। এভাবেই নেগেটিভ ইনডেক্সিং-এর ধারা চলতে থাকে।

উদাহরণ ঃ

```
b = ['Mahi', 'Mahbub', 'Mahfuz', 'Samsul', 96002, 88.87]
print(b[0])
print(b[-1])
print(b[-4:-1])
print(b[:-3])
print(b[-2:])
```

আউটপুট ঃ

```
Mahi
88.87
['Mahfuz', 'Samsul', 96002]
['Mahi', 'Mahbub', 'Mahfuz']
[96002, 88.87]
```

স্লাইসিং (Slicing) পদ্ধতি ঃ স্লাইসিং অপারেটর (ঃ) বা কোলন ব্যবহার করে লিস্টের আইটেমকে অ্যাক্সেস পদ্ধতিকে স্লাইসিং বলে।

উদাহরণ-১ ঃ

```
b = ['Mahi', 'Mahbub', 'Mahfuz', 'Samsul', 96002, 88.87]
print(b[1:5])
print(b[:5])
print(b[2:])
```

আউটপুট ঃ

```
['Mahbub', 'Mahfuz', 'Samsul', 96002]
['Mahi', 'Mahbub', 'Mahfuz', 'Samsul', 96002]
['Mahfuz', 'Samsul', 96002, 88.87]
```

উদাহরণ-২ ঃ

```
b = ['Mahi', 'Mahbub', 'Mahfuz', 'Samsul', 96002, 88.87]
print(b[-4:-1])
print(b[:-3])
print(b[-2:])
```

আউটপুট ঃ

```
['Mahfuz', 'Samsul', 96002]
['Mahi', 'Mahbub', 'Mahfuz']
[96002, 88.87]
```

৮.৩ লিস্টের উপাদান আপডেটিং ও ডিলেটিং (Updating & deleting list elements) ঃ

লিস্টের উপাদান আপডেটিং (Updating list elements) ঃ লিস্টের উপাদান আপডেটিং বলতে লিস্টের ইনডেক্সে যে আইটেমগুলো আছে সেগুলোর মধ্যে কোনো পরিবর্তন, পরিবর্ধন, সংযোজন, বিয়োজন ইত্যাদিকে অপারেশনকে বুঝায়। লিস্টের কোনো একটি আইটেম যদি 'Mahi' হয় তাহলে তা পরিবর্তন করে 'Mahdee'-তে রূপান্তর করার প্রক্রিয়াই হচ্ছে আপডেটিং। লিস্ট আপডেটিং এর ক্ষেত্রে স্ট্রিংয়ের পরিবর্তে স্ট্রিং, ইন্টিজারের পরিবর্তে ইন্টিজার আর ফ্লোটের পরিবর্তে ফ্লোট টাইপের ডাটাই ব্যবহার করতে হবে এরকম কোনো বিধি-নিষেধ নেই। যে-কোনো ডাটা টাইপের পরিবর্তে অন্য যে-কোনো ডাটা টাইপ ব্যবহার করতে পারি। লিস্ট এলিমেন্টকে আপডেট করার জন্য অ্যাসাইনমেন্ট অপরেটর ('=') ব্যবহার করা হয়।

উদাহরণ ঃ

```
b = ['Mahi', 587, 'COBOL', 3.1416, 'Samsul', 96002]
b[0] = 'Mahdee'
print (b)
b[4] = 900
print (b)
b[5] = 6814.25
print (b)
```

আউটপুট ঃ

```
['Mahdee', 587, 'COBOL', 3.1416, 'Samsul', 96002]
['Mahdee', 587, 'COBOL', 3.1416, 900, 96002]
['Mahdee', 587, 'COBOL', 3.1416, 900, 6814.25]
```

লিস্টে নতুন কোনো উপাদান সংযোজন (Inserting new item in the list) ঃ লিস্টে নতুন কোনো আইটেম সংযোজন করতে চাইলে পাইথনে তিন ধরনের ফাংশন ব্যবহার করা হয়। যথা– append() ফাংশন, insert() ফাংশন ও extend() ফাংশন । append() ফাংশন দিয়ে কোনো আইটেম যোগ করা হলে সেটা সবসময় লিস্টের শেষে যোগ হবে। কিন্তু insert() ফাংশন ব্যবহার করে নতুন আইটেমটিকে যে-কোনো ইনডেক্সে সংযোজন করা যায়। সেক্ষেত্রে নতুন আইটেমটিকে যে ইনডেক্সে সংযোজন করতে চাই তার ইনডেক্স নম্বর আর নতুন আইটেমটি বলে দিতে হবে। কিন্তু একাধিক আইটেম এক সঙ্গে সংযোজন করার জন্য extend() ফাংশন ব্যবহার করা হয়। তবে এক্ষেত্রে আইটেমগুলি লিস্ট হিসেবে নং বরং সিঙ্গেল আইটেম হিসেবে সংযোজিত হয়।

উদাহরণ ঃ

```
b = ['Mahi', 587, 'COBOL', 3.1416, 'Samsul', 96002]
b.append('Mahdee')
print (b)
b.insert(3,'Tonny')
print (b)
b.extend('Munny''Sumaiya''Tahsin')
print(b)
```

আউটপুট ঃ

```
['Mahi', 587, 'COBOL', 3.1416, 'Samsul', 96002, 'Mahdee']
['Mahi', 587, 'COBOL', 'Tonny', 3.1416, 'Samsul', 96002, 'Mahdee']
['Mahi', 587, 'COBOL', 'Tonny', 3.1416, 'Samsul', 96002,
'Mahdee', 'M', 'u', 'n', 'n', 'y', 'S', 'u', 'm', 'a', 'i', 'y',
'a', 'T', 'a', 'h', 's', 'i', 'n']
```

লিস্টের উপাদান বিয়োজন (Deleting list elements) ঃ লিস্ট থেকে এর কোনো উপাদান রিমুভ করার জন্য পাইথনে দুই ধরনের ফাংশন রয়েছে। যথা– del() ফাংশন ও remove() ফাংশন। যে আইটেমটিকে বাদ দিতে চাই তার ইনডেক্স জানা থাকলে del() ফাংশন ও ইনডেক্স নম্বর জানা না থাকলে remove() ফাংশন ব্যবহার করা হয়।

উদাহরণ ঃ

```
b = ['Mahi', 587, 'COBOL', 3.1416, 'Samsul', 96002]
del b[2]
print (b)
b.remove('Samsul')
print (b)
b.remove(96002)
print (b)
del b[-1]
print (b)
```

আউটপুটঃ

```
['Mahi', 587, 3.1416, 'Samsul', 96002]
['Mahi', 587, 3.1416, 96002]
['Mahi', 587, 3.1416]
['Mahi', 587]
```

৮.৪ লিস্টের বিভিন্ন অপারেশন (Basic list opreation) ঃ

পাইথনে মোট তিন ধরনের লিস্ট অপারেশন আছে। যথা–

- কনক্যাটেশন অপারেটর
- রিপিটিশন অপারেটর
- মেম্বারশিপ অপারেটর

কনক্যাটেশন অপারেটর (Concatenation operator) ঃ দুটি লিস্ট এর উপাদানসমূহকে সংযুক্ত করে নতুন কোনো লিস্ট তৈরি করার জন্য কনক্যাটেশন অপারেটর (+) ব্যবহার করা হয়। লিস্ট এর উপাদানসমূহ বিভিন্ন টাইপের হলেও কোনো সমস্যা নেই।

উদাহরণ-১ ঃ

```
List1=["Mahbub",960021,"Mahi",23022015]
List2=[19072017,"Mahdee","Tahsin"]
List3=List1+List2
print(List3)
```

আউটপুট ঃ

```
['Mahbub', 960021, 'Mahi', 23022015, 19072017, 'Mahdee', 'Tahsin']
```

```
List1=["Mahbub",960021,"Mahi",23022015]
List2=List1+[19072017,"Mahdee","Tahsin"]
print(List2)
```

আউটপুট ঃ

```
['Mahbub', 960021, 'Mahi', 23022015, 19072017, 'Mahdee', 'Tahsin']
```

উদাহরণ-৩ ঃ

```
List1=["Mahbub",960021,"Mahi",23022015]
print(List1+[19072017,"Mahdee","Tahsin"])
```

আউটপুট ঃ

```
['Mahbub', 960021, 'Mahi', 23022015, 19072017, 'Mahdee', 'Tahsin']
```

রিপিটিশন অপারেটর (Repetition operator) ঃ কোনো লিস্ট এ বিদ্যমান উপাদানসমূহকে একাধিক বার রিপিট করার জন্য পাইথনে রিপিটেশন অপারেটর (*) ব্যবহার করা হয়। এক্ষেত্রেও লিস্ট এর উপাদানসমূহ বিভিন্ন টাইপের হলেও কোনো সমস্যা নেই।

উদাহরণ ঃ

```
List1=["Mahbub",960021,"Mahi",23022015]
print(List1*2)
```

আউটপুট ঃ

```
['Mahbub', 960021, 'Mahi', 23022015, 'Mahbub', 960021, 'Mahi', 23022015]
```

মেম্বারশিপ অপারেটর (Membership operator) ঃ কোনো লিস্টের মধ্যে নির্দিষ্ট কোনো এলিমেন্টের উপস্থিতি বা অনুপস্থিতি চেক করার জন্য মেম্বারশিপ অপারেটর ব্যবহার করা হয়। পাইথনের লিস্টে দুই ধরনের মেম্বারশিপ অপারেটর ব্যবহৃত হয়। যেমন– in অপারেটর ও not in অপারেটর। কোনো লিস্টের মধ্যে নির্দিষ্ট কোনো এলিমেন্ট উপস্থিত আছে কি না সেটা চেক করার জন্য in অপারেটর ব্যবহার করা হয়। যদি এলিমেন্টটি লিস্টের মধ্যে এক বা একাধিকবার থাকে তাহলে এটি True রিটার্ন করে অন্যথায় False রিটার্ন করে। একইভাবে লিস্টের সাথে not in অপারেটর ব্যবহার করে কোনো এলিমেন্টের অনুপস্থিতি চেক করা যায়।

উদাহরণ-১ ঃ

```
List1=["Mahbub",960021,"Mahi",23022015]
List2=[19072017,"Mahdee","Tahsin"]
print("Samsul" in List1)
print("Mahdee" in List1)
print("Mahdee" in List2)
print("Mahbub" in List2)
```

আউটপুট ঃ

```
False
False
True
False
```

উদাহরণ-২ ঃ

```
List1=["Mahbub",960021,"Mahi",73022015]
List2=[19072017,"Mahdee","Tahsin"]
print("Samsul"  not in List1)
print("Mahdee"  not in List1)
print("Mahdee" not in List2)
print("Mahbub" in List1)
```

আউটপুট ঃ

```
True
True
False
True
```

নিম্নে পাইথনে ব্যবহৃত বিভিন্ন মেম্বারশীপ অপারেটরের উদাহরণ ও তাদের ফলাফল দেখানো হল।

Python Expression	Results
[21, 27, 23] + [34, 52, 63]	[21, 27, 23, 34, 52, 63]
['Mahdee'] * 3	['Mahdee', 'Mahdee', 'Mahdee']
8 in [1, 2, 3,5,7]	False

৮.৫ লিস্টে ব্যবহৃত বিল্ট ইন ফাংশন ও মেথডসমূহ (Built in functions and methods) ঃ

লিস্টে ব্যবহৃত বিল্ট ইন ফাংশন (Built in functions) ঃ পাইথন প্রোগ্রামে লিস্ট নিয়ে দ্রুততার সঙ্গে এবং সুনির্দিষ্ট কোনো কাজ সম্পাদনের জন্য এতে বেশ কিছু বিল্ট ইন ফাংশন রয়েছে। নিম্নে লিস্টে ব্যবহৃত এসকল বিল্ট ইন ফাংশনসমূহ ও তাদের ব্যবহার উল্লেখ করা হলো ঃ

পাইথনের লিস্টে ব্যবহৃত বিল্ট ইন ফাংশনসমূহ	
বিল্ট ইন ফাংশন	**বর্ণনা**
min()	লিস্টে বিদ্যমান সকল উপাদানের মধ্যে সবচেয়ে ছোট আইটেমটি জানার জন্য এই ফাংশনটি ব্যবহৃত হয়।
any()	লিস্টের যে-কোনো একটি উপাদানের মান সত্য হলে ও আউটপুট হিসেবে এই ফাংশনটি True রিটার্ন করে। লিস্টটি empty হলেই কেবল আউটপুট হিসেবে False রিটার্ন করে।
enumerate()	এই ফাংশনটি টাপল হিসেবে লিস্টের সকল উপাদানের ইনডেক্স নম্বর ও তাদের মান ধারণ করে।
all()	লিস্টের সকল উপাদানের মান সত্য হলে আউটপুট হিসেবে এই ফাংশনটি True রিটার্ন করে।
cmp(list1, list2)	একাধিক লিস্টের উপাদানসমূহের মধ্যে তুলনা করার জন্য এই ফাংশনটি ব্যবহৃত হয়।
len()	লিস্টে বিদ্যমান সকল উপাদানের সংখ্যা জানার জন্য এই ফাংশনটি ব্যবহৃত হয়।
list()	পাইথনের ইটারেবল ডাটা স্ট্রাকচারসমূহকে (tuple, string, set, dictionary) লিস্টে রূপান্তরের জন্য এই ফাংশনটি ব্যবহৃত হয়।
max()	লিস্টে বিদ্যমান সকল উপাদানের মধ্যে সর্ববৃহৎ আইটেমটি জানার জন্য এই ফাংশনটি ব্যবহৃত হয়।
sorted()	সর্টেড লিস্ট রিটার্নের জন্য এই ফাংশনটি ব্যবহৃত হয়।
sum()	লিস্টে বিদ্যমান উপাদানসমূহের যোগফল নির্ণয়ের জন্য এই ফাংশনটি ব্যবহৃত হয়।

all()ফাংশন ঃ

সিনট্যাক্স ঃ

all(iterable)

ট্রুথ টেবিল ঃ

When	Return value
All values are true	True
All values are false	False
One value is true (others are false)	False
One value is false (others are true)	False
Empty Iterable	True

উদাহরণ ঃ

```
list = [1, 3, 4, 5]
print(all(list))

# all values false
list = [0, False]
print(all(list))

# one false value
list = [1, 3, 4, 0]
print(all(list))

# one true value
list = [0, False, 5]
print(all(list))
# empty iterable
list = [ ]
print(all(list))
```

আউটপুট ঃ

```
True
False
False
False
True
```

any()ফাংশন ঃ

সিনট্যাক্স ঃ

```
any(iterable)
```

ট্রুথ টেবিল ঃ

When	Return value
All values are true	True
All values are false	False
One value is true (others are false)	True
One value is false (others are true)	True
Empty Iterable	False

উদাহরণ ঃ

```
list = [1, 3, 4, 5]
print(any(list))
list = [0, False]
print(any(list))
list = [1, 3, 4, 0]
print(any(list))
list = [0, False, 5]
print(any(list))
list = [ ]
print(any(list))
```

আউটপুট ঃ

```
True
False
True
True
False
```

enumearte()ফাংশন ঃ

সিনট্যাক্স ঃ

enumerate(iterable, start=0)

উদাহরণ ঃ

```
grocery = ['bread', 'milk', 'butter']
enumerateGrocery = enumerate(grocery)
print(type(enumerateGrocery))
# converting to list
print(list(enumerateGrocery))
# changing the default counter
enumerateGrocery = enumerate(grocery, 10)
print(list(enumerateGrocery))
```

আউটপুট ঃ

```
<class 'enumerate'>
[(0, 'bread'), (1, 'milk'), (2, 'butter')]
[(10, 'bread'), (11, 'milk'), (12, 'butter')]
```

len()ফাংশন ঃ

সিনট্যাক্স ঃ

len(testList))

উদাহরণ ঃ

```
testList = []
print(testList, 'length is', len(testList))
testList = [1, 2, 3]
print(testList, 'length is', len(testList))
```

আউটপুট ঃ

```
[] length is 0
[1, 2, 3] length is 3
```

list()ফাংশন ঃ

সিনট্যাক্স ঃ

list([iterable])

উদাহরণ ঃ

```
# empty list
print(list())
# vowel string
vowelString = 'aeiou'
print(list(vowelString))
# vowel tuple
vowelTuple = ('a', 'e', 'i', 'o', 'u')
print(list(vowelTuple))
# vowel list
vowelList = ['a', 'e', 'i', 'o', 'u']
print(list(vowelList))
```

আউটপুট ঃ

```
[]
['a', 'e', 'i', 'o', 'u']
['a', 'e', 'i', 'o', 'u']
['a', 'e', 'i', 'o', 'u']
```

max()ফাংশন ঃ

সিনট্যাক্স ঃ

max(iterable, *iterables[,key, default])

max(arg1, arg2, *args[, key])

উদাহরণ ঃ

```
# using max(arg1, arg2, *args)
print('Maximum is:', max(1, 3, 2, 5, 4))
# using max(iterable)
num = [1, 3, 2, 8, 5, 10, 6]
print('Maximum is:', max(num))
```

আউটপুট ঃ

```
Maximum is: 5
Maximum is: 10
```

min()ফাংশন ঃ

সিনট্যাক্স ঃ

min(iterable, *iterables[,key, default])
min(arg1, arg2, *args[, key])

উদাহরণ ঃ

```
# using min(arg1, arg2, *args)
print('minimum is:', min(17, 3, 22, 5, 4))
# using min(iterable)
num = [11, 3, 2, 8, 5, 10, 6]
print('minimum is:', min(num))
```

আউটপুট ঃ

```
minimum is: 3
minimum is: 2
```

sorted()ফাংশন ঃ

সিনট্যাক্স ঃ

sorted(iterable[, key][, reverse])

উদাহরণ ঃ

```
List1 = ['e', 'a', 'u', 'o', 'i']
print(sorted(List1))
List2 = ['e', 'a', 'u', 'o', 'i']
print(sorted(List2, reverse=True))
```

আউটপুট ঃ

```
['a', 'e', 'i', 'o', 'u']
['u', 'o', 'i', 'e', 'a']
```

sum()ফাংশন ঃ

সিনট্যাক্স ঃ

sum(iterable, start)

উদাহরণ ঃ

```
Numbers = [2.5, 3, 4, -5]
# start parameter is not provided
Summation = sum(Numbers)
print(Summation)
# start = 10
Summation = sum(Numbers, 10)
print(Summation)
```

আউটপুটঃ

```
4.5
14.5
```

লিস্টে ব্যবহৃত মেথডসমূহ (List methods) ঃ লিস্ট ম্যানিপুলেশনের জন্য পাইথনে মেথডসমূহ ব্যবহার করা হয়। পাইথনে যে সকল বিল্ট ইন মেথডসমূহ ব্যবহার করা হয় তার একটি তালিকা ও তাদের বর্ণনা নিম্নে উল্লেখ করা হলো ঃ

পাইথনের লিস্টে ব্যবহৃত মেথডসমূহ	
মেথড	**বর্ণনা**
extend()	একটি লিস্টে বিদ্যমান সকল উপাদান অন্য লিস্টে সংযোজন করার জন্য এই মেথড ব্যবহার করা হয়।
copy()	লিস্টের নির্দিষ্ট কোনো আইটেমকে কপি করার জন্য এই মেথড ব্যবহার করা হয়।
append()	লিস্টের শেষে একটি এলিমেন্ট সংযুক্ত করার জন্য এই মেথড ব্যবহার করা হয়।
remove()	লিস্টের নির্দিষ্ট কোনো আইটেমকে বিয়োজন বা রিমুভ করার জন্য এই মেথড ব্যবহার করা হয়।
insert()	লিস্টের নির্দিষ্ট কোনো পজিশন বা ইনডেক্সে কোনো আইটেম সংযোজনের জন্য এই মেথড ব্যবহার করা হয়।
index()	কোনো একটি এলিমেন্ট লিস্টের কোন ইনডেক্সে অবস্থান করছে সেটা চেক করার জন্য এই মেথড ব্যবহার করা হয়।
pop()	লিস্টের একেবারে শেষ আইটেমটিকে রিমুভ করার জন্য এই মেথড ব্যবহার করা হয়।
clear()	একটি লিস্টে বিদ্যমান সকল উপাদানকে রিমুভ করার জন্য এই মেথড ব্যবহার করা হয়।
count()	একটি লিস্টে বিদ্যমান উপাদান সংখ্যা গণনা করার জন্য এই মেথড ব্যবহার করা হয়।
sort()	একটি লিস্টে বিদ্যমান উপাদানগুলোকে অ্যাসেনডিং অর্ডারে সাজানোর জন্য এই মেথড ব্যবহার করা হয়।
reverse()	লিস্টে বিদ্যমান উপাদানগুলোকে রিভার্স অর্ডারে সাজানোর জন্য এই মেথড ব্যবহার করা হয়।

append() মেথড ঃ

সিনট্যাক্স ঃ

```
list.append(item)
```

উদাহরণ ঃ

```
List1 = ['Mahi', 'Mahdee', 'Suhaima']
List1.append('Tahsin')
print('Updated list is: ', List1)
List2=['Tonny','Munny','Sumaiya']
List1.append(List2)
print('Updated list is: ', List1)
```

আউটপুট ঃ

```
Updated list is:  ['Mahi', 'Mahdee', 'Suhaima', 'Tahsin']
Updated list is:  ['Mahi', Mahdee', Suhaima', 'Tahsin', [Tonny', 'Momy', 'Sumaiya']]
```

extend() মেথড ঃ

সিনট্যাক্স ঃ

list1.extend(list2)

উদাহরণ ঃ

```
List1=['Mahi','Mahdee','Suhaima']
List2=['Tonny','Munny','Sumaiya']
List1.extend(List2)
print(List1)
```

আউটপুট ঃ

```
['Mahi', 'Mahdee', 'Suhaima', 'Tonny', 'Munny', 'Sumaiya']
```

insert() মেথড ঃ

সিনট্যাক্স ঃ

list.insert(index, element)

উদাহরণ ঃ

```
vowel = ['a', 'e', 'i', 'u']
# inserting element to list at 4th position
vowel.insert(3, 'o')
print('Updated List: ', vowel)
list1 = [{1, 2}, [5, 6, 7]]
x = (3, 4)
# inserting x to the list
list1.insert(1, x)
print('Updated List: ', list1)
```

আউটপুট ঃ

```
Updated List:  ['a', 'e', 'i', 'o', 'u']
Updated List:  [{1, 2}, (3, 4), [5, 6, 7]]
```

remove() মেথড ঃ

সিনট্যাক্স ঃ

list.remove(element)

উদাহরণ ঃ

```
vowel = ['a', 'e', 'i', 'o','u']
vowel.remove('o')
print('Updated List: ', vowel)
animal = ['cat', 'dog', 'dog', 'guinea pig', 'dog']
animal.remove('dog')
print('Updated animal list: ', animal)
```

আউটপুট ঃ

```
Updated List:  ['a', 'e', 'i', 'u']
Updated animal list:  ['cat', 'dog', 'guinea pig', 'dog']
```

pop() মেথড ঃ

সিনট্যাক্স ঃ

list.pop(index)

উদাহরণ ঃ

```
language = ['Python', 'Java', 'C++', 'Cobol', 'C']
return_value = language.pop(3)
print('Return value: ', return_value)
print('Updated List: ', language)
```

আউটপুট ঃ

```
Return Value:  Cobol
Updated List:  ['Python', 'Java', 'C++', 'C']
```

clear() মেথড ঃ

সিনট্যাক্স ঃ

list.clear()

উদাহরণ ঃ

```
list = ['Python', 'Java', 'C++', 'Cobol', 'C']
list.clear()
print('Updated List: ', list)
```

আউটপুট ঃ

```
Updated List:  [ ]
```

index() মেথড ঃ

সিনট্যাক্স ঃ

list.index(element)

উদাহরণ ঃ

```
vowels = ['a', 'e', 'i', 'o', 'i', 'u']
# element 'e' is searched
index = vowels.index('e')
# index is printed
print('The index of e:', index)
# element 'i' is searched
index = vowels.index('i')
# only the first index of the element is printed
print('The index of i:', index)
```

আউটপুট ঃ

```
The index of e: 1
The index of i: 2
```

count() মেথড ঃ

সিনট্যাক্স ঃ

list.count(element)

উদাহরণ ঃ

```
vowels = ['a', 'e', 'i', 'o', 'i', 'u']
# count element 'i'
count = vowels.count('i')
# print count
print('The count of i is:', count)
# count element 'p'
count = vowels.count('p')
# print count
print('The count of p is:', count)
```

আউটপুট ঃ

```
The count of i is: 2
The count of p is: 0
```

sort() মেথড ঃ

সিনট্যাক্স ঃ

list.sort()

উদাহরণ ঃ

```
# vowels list
vowels = ['e', 'a', 'u', 'o', 'i']
# sort the vowels
vowels.sort()
# print vowels
print('Sorted list:', vowels)
```

আউটপুট ঃ

```
Sorted list: ['a', 'e', 'i', 'o', 'u']
```

reverse() মেথড ঃ

সিনট্যাক্স ঃ

list.reverse()

উদাহরণ ঃ

```
os = ['Windows', 'macOS', 'Linux']
print('Original List:', os)
# List Reverse
os.reverse()
# updated list
print('Updated List:', os)
reversed_list = os[::-1]
print('Updated List:', reversed_list)
for o in reversed(os):
    print(o)
```

আউটপুট ঃ

```
Original List: ['Windows', 'macOS', 'Linux']
Updated List: ['Linux', 'macOS', 'Windows']
Updated List: ['Windows', 'macOS', 'Linux']
Windows
macOS
Linux
```

৮.৬ লিস্ট ব্যবহার করে কতিপয় প্রোগ্রাম (Simple programs using list) ঃ

প্রোগ্রাম-১ ঃ একটি লিস্টে কতগুলো সংখ্যা ইনপুট নিয়ে তাদের যোগফল নির্ণয় করার জন্য প্রোগ্রাম।

```
Numbers = [1, 4, 9, 16]
sum = 0
for num in Numbers:
    sum += num
print ("summation is=", sum )
z = [ 10, 20, 30 ]
print(all ( [ x>10 for x in z ] ))
print(all( [ "", "0", [0], "None" ] ))
print(any ( [ x > 10 for x in z ] ))
print(any ( [ x > 50 for x in z ] ))
print(any ( [ [ ], ( ), { }, None, 0 ] ))
print(any ( [ [0], ( ), { }, None, 0 ] ))
print(all ( [ ] ))
print(any ( [ ] ))
```

প্রোগ্রাম-২ ঃ লিস্টের কতকগুলো সংখ্যা বড় হতে ছোট ক্রমানুসারে সাজানোর একটি প্রোগ্রাম তৈরি কর।

```
my_list = [1, 5, 2, 6, 3, 7, 4]
my_list.sort (reverse = true)
print ("sorted list in descending:", my_list)
```

অনুশীলনী-৮

অতি সংক্ষিপ্ত প্রশ্নোত্তর ঃ

১। **লিস্ট বলতে কী বুঝায়?**

উত্তর ঃ লিস্ট হচ্ছে পাইথনের সবচেয়ে বৈচিত্র্যপূর্ণ ডাটা টাইপ, যা স্কয়ার ব্র্যাকেট [ ] এর ভেতর কমার সাহায্যে উপাদানসহ প্রকাশ করা হয়। অন্যান্য ল্যাংগুয়েজে যাকে অ্যারে বলা হয় পাইথনে তাকেই লিস্ট বলে।

২। **লিস্ট এলিমেন্টগুলোকে পাইথনে কীভাবে সংযুক্ত করা হয়?**

উত্তর ঃ পাইথনে লিস্ট এলিমেন্টেগুলোকে কমা দিয়ে যুক্ত করা হয়। এর এলিমেন্টগুলো ইনডেক্স অনুযায়ী (0,1,2,3.....) সাজানো থাকে। ইনডেক্সের মান সবসময় 0 থেকে শুরু হয়।

৩। **লিস্ট এলিমেন্ট কত প্রকার ও কী কী ?**

উত্তর ঃ পাইথনের লিস্ট এলিমেন্ট বিভিন্ন টাইপের হতে পারে, যেমন–

- ইন্টিজার টাইপ ডাটা
- ফ্লোটিং পয়েন্ট টাইপ ডাটা
- স্ট্রিং
- লিস্ট
- টাপল
- ডিকশনারি ও
- মিক্সড ডাটা টাইপ ইত্যাদি।

৪। **লিস্ট ও অ্যারের মধ্যে পার্থক্য লেখ।**

উত্তর ঃ সি, সি++ বা জাভা ল্যাংগুয়েজে যাকে অ্যারে বলা হয় পাইথনে তাকেই লিস্ট বলে। তবে লিস্ট অ্যারের চেয়েও স্মার্ট। কারণ প্রোগ্রামিং এ লিস্টের প্রায়োগিক সক্ষমতা অনেক বেশি।

৫। **লিস্টের উপাদানে অ্যাক্সেসিং পদ্ধতিসমূহের নাম লেখ।**

উত্তর ঃ লিস্টের উপাদানসমূহকে অ্যাক্সেস করার জন্য পাইথনে বিভিন্ন পদ্ধতি প্রচলিত আছে। পদ্ধতিসমূহ হচ্ছে–

- ইনডেক্সিং (Indexing) পদ্ধতি
- নেগেটিভ ইনডেক্সিং (Negative Indexing) পদ্ধতি এবং
- স্লাইসিং (Slicing) পদ্ধতি।

৬। **ইনডেক্সিং বলতে কী বুঝায়?**

উত্তর ঃ যে পদ্ধতিতে লিস্টের কোনো আইটেমকে অ্যাক্সেস করার জন্য ইনডেক্স নম্বর ব্যবহার করা হয়, তাকে ইনডেক্সিং পদ্ধতি বলে। লিস্টের ইনডেক্স নম্বর শুরু হয় 0 থেকে।

৭। **নেগেটিভ ইনডেক্সিং বলতে কী বুঝায়?**

উত্তর ঃ যে পদ্ধতিতে লিস্টের কোনো আইটেমকে অ্যাক্সেস করার জন্য ইনডেক্স হিসেবে নেগেটিভ নম্বর ব্যবহার করা হয়, তাকে নেগেটিভ ইনডেক্সিং বলে। নেগেটিভ ইনডেক্সিং-এর ক্ষেত্রে লিস্টের ইনডেক্স নম্বর শুরু হয় –1 দ্বারা। –1 দ্বারা সর্বশেষ আইটেম, ইনডেক্স -2 দ্বারা দ্বিতীয় সর্বশেষ আইটেম এবং এভাবেই এটি চলতে থাকে।

৮। **স্লাইসিং বলতে কী বুঝায়?**

উত্তর ঃ স্লাইসিং অপারেটর (ঃ) বা কোলন ব্যবহার করে লিস্টের আইটেমকে অ্যাক্সেস পদ্ধতিকে স্লাইসিং বলে।

৯। **লিস্টের উপাদান আপডেটিং বলতে কী বুঝায়?**

উত্তর ঃ লিস্টের উপাদান আপডেটিং বলতে লিস্টের ইনডেক্সে যে আইটেমগুলো আছে সেগুলোর মধ্যে কোনো পরিবর্তন, পরিবর্ধন, সংযোজন, বিয়োজন ইত্যাদির অপারেশনকে বুঝায়। লিস্টের কোনো একটি আইটেম যদি 'Mahi' হয় তাহলে তা পরিবর্তন করে 'Mahdee'-তে রূপান্তর করার প্রক্রিয়াই হচ্ছে আপডেটিং।

১০। **লিস্টের উপাদান আপডেটিং-এর কোন অপারেটর ব্যবহার করা হয়?**

উত্তর ঃ লিস্ট এলিমেন্টকে আপডেট করার জন্য অ্যাসাইনমেন্ট অপরেটর ('=') ব্যবহার করা হয়।

১১। **লিস্টে নতুন কোনো আইটেম সংযোজনের জন্য কী কী ফাংশন ব্যবহার করা হয়?**

উত্তর ঃ লিস্টে নতুন কোনো আইটেম সংযোজন করতে চাইলে পাইথনে তিন ধরনের ফাংশন ব্যবহার করা হয়, যথা- append() ফাংশন, insert() ফাংশন ও extend() ফাংশন ।

১২। **লিস্ট থেকে কোনো আইটেম বিয়োজনের জন্য কী কী ফাংশন ব্যবহার করা হয়?**

উত্তর ঃ লিস্ট থেকে এর কোনো উপাদান রিমুভ করার জন্য পাইথনে দুই ধরনের ফাংশন রয়েছে। যথা– del() ফাংশন ও remove() ফাংশন।

১৩। **পাইথনের লিস্ট অপারেশন কত প্রকার ও কী কী?**

উত্তর ঃ পাইথনে মোট তিন ধরনের লিস্ট অপারেশন আছে। যথা–

➢ কনক্যাটেশন অপারেটর

➢ রিপিটিশন অপারেটর এবং

➢ মেম্বারশিপ অপারেটর।

১৪। **লিস্টের রিপিটিশন অপারেশন কীভাবে সংঘটিত হয়?**

উত্তর ঃ রিপিটেশন অপারেটর (*) ব্যবহার কোনো লিস্ট-এ বিদ্যমান উপাদানসমূহকে একাধিক বার রিপিট করা হয়। এক্ষেত্রে লিস্ট-এর উপাদানসমূহ বিভিন্ন টাইপের হলেও কোনো সমস্যা নেই।

১৫। লিস্ট অপারেশনে ব্যবহৃত হয় এমন যে-কোনো তিনটি বিল্ট ইন ফাংশনের নাম লেখ।

উত্তর ঃ লিস্ট অপারেশনে ব্যবহৃত হয় এমন যে-কোনো তিনটি বিল্ট ইন ফাংশনের নাম হলো ঃ all(), any() এবং len() ইত্যাদি।

১৬। len() ফাংশনের কাজ কী?

উত্তর ঃ লিস্টে বিদ্যমান সকল উপাদানের সংখ্যা জানার জন্য len() ফাংশন ব্যবহৃত হয়।

১৭। sum() ফাংশনের কাজ কী?

উত্তর ঃ লিস্টে বিদ্যমান উপাদানসমূহের যোগফল নির্ণয়ের জন্য এই ফাংশনটি ব্যবহৃত হয়।

১৮। cmp() ফাংশনের কাজ কী?

উত্তর ঃ একাধিক লিস্টের উপাদানসমূহের মধ্যে তুলনা করার জন্য এই ফাংশনটি ব্যবহৃত হয়।

১৯। all() ও any() ফাংশনের মধ্যে পার্থক্য কী?

উত্তর ঃ লিস্টের সকল উপাদানের মান সত্য হলে আউটপুট হিসেবে all() ফাংশনটি True রিটার্ন করে। অপরদিকে লিস্টের যে-কোনো একটি উপাদানের মান সত্য হলেও আউটপুট হিসেবে any() ফাংশনটি True রিটার্ন করে। লিস্টটি empty হলেই কেবল আউটপুট হিসেবে False রিটার্ন করে।

২০। append() ও insert() মেথডের মধ্যে পার্থক্য কী?

উত্তর ঃ লিস্টের শেষে একটি এলিমেন্ট সংযুক্ত করার জন্য append() মেথড ব্যবহার করা হয়। অপরদিকে লিস্টের নির্দিষ্ট কোনো পজিশন বা ইনডেক্সে কোনো আইটেম সংযোজনের জন্য insert() মেথড ব্যবহার করা হয়।

২১। pop() মেথড ব্যবহার করা হয় কেন?

উত্তর ঃ লিস্টের একেবারে শেষ আইটেমটিকে রিমুভ করার জন্য **pop()** মেথড ব্যবহার করা হয়।

২২। count() মেথডের কাজ কী?

উত্তর ঃ একটি লিস্টে বিদ্যমান উপাদান সংখ্যা গণনা করার জন্য **count()** মেথড ব্যবহার করা হয়।

২৩। copy() মেথডের কাজ কী?

উত্তর ঃ লিস্টের নির্দিষ্ট কোনো আইটেমকে কপি করার জন্য **copy()** মেথড ব্যবহার করা হয়।

২৪। index() মেথডের কাজ কী?

উত্তর ঃ কোনো একটি এলিমেন্ট লিস্টের কোনো ইনডেক্সে অবস্থান করছে সেটা চেক করার জন্য **index()** মেথড ব্যবহার করা হয়।

২৫। extend() মেথড ব্যবহার করা হয় কেন?

উত্তর ঃ একটি লিস্টে বিদ্যমান সকল উপাদান অন্য লিস্টে সংযোজন করার জন্য **extend()** মেথড ব্যবহার করা হয়।

সংক্ষিপ্ত প্রশ্নোত্তর ঃ

১। **ইন্টিজার টাইপ ডাটা ব্যবহার করে লিস্ট তৈরি করে দেখাও।**

উত্তর সংকেত ঃ ৮.১ নং অনুচ্ছেদ দ্রষ্টব্য।

২। **স্ট্রিং টাইপ ডাটা ব্যবহার করে লিস্ট তৈরি করে দেখাও।**

উত্তর সংকেত ঃ ৮.১ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। **লিস্টের মধ্যে লিস্ট তৈরি করে দেখাও।**

উত্তর সংকেত ঃ ৮.১ নং অনুচ্ছেদ দ্রষ্টব্য।

৪। **উদাহরণসহ লিস্ট ইন্ডেক্সিং পদ্ধতি সংক্ষেপে বর্ণনা কর।**

উত্তর সংকেত ঃ ৮.২ নং অনুচ্ছেদ দ্রষ্টব্য।

৫। **লিস্টে নেগেটিভ ইনডেক্সিং পদ্ধতি কীভাবে কাজ করে?**

উত্তর সংকেত ঃ ৮.২ নং অনুচ্ছেদ দ্রষ্টব্য।

৬। **লিস্টে স্লাইসিং পদ্ধতির ব্যবহার দেখাও।**

উত্তর সংকেত ঃ ৮.২ নং অনুচ্ছেদ দ্রষ্টব্য।

৭। **উদাহরণসহ লিস্টের উপাদান আপডেটিং পদ্ধতি সংক্ষেপে বর্ণনা কর।**

উত্তর সংকেত ঃ ৮.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

৮। **লিস্টে উপাদান সংযোজনের জন্য ব্যবহৃত ফাংশনসমূহ ব্যাখ্যা কর।**

উত্তর ঃ লিস্টে নতুন কোনো উপাদান সংযোজন করতে চাইলে পাইথনে তিন ধরনের ফাংশন ব্যবহার করা হয়। যথা– append() ফাংশন, insert() ফাংশন ও extend() ফাংশন। append() ফাংশন দিয়ে কোনো আইটেম যোগ করা হলে সেটা সবসময় লিস্টের শেষে যোগ হবে। কিন্তু insert() ফাংশন ব্যবহার করে নতুন আইটেমটিকে যে-কোনো ইনডেক্সে সংযোজন করা যায়। সেক্ষেত্রে নতুন আইটেমটিকে যে ইনডেক্সে সংযোজন করতে চাই তার ইনডেক্স নম্বর আর নতুন আইটেমটি বলে দিতে হবে। কিন্তু একাধিক আইটেম এক সঙ্গে সংযোজন করার জন্য extend() ফাংশন ব্যবহার করা হয়। তবে এক্ষেত্রে আইটেমগুলো লিস্ট হিসেবে নং বরং সিঙ্গেল আইটেম হিসেবে সংযোজিত হয়।

৯। **পাইথন প্রোগ্রামে বিভিন্ন প্রকার মৌলিক লিস্ট অপারেশনগুলোর সংক্ষিপ্ত বর্ণনা দাও।** [বাকাশিবো-২০১৭]

উত্তর সংকেত ঃ ৮.৪ নং অনুচ্ছেদ দ্রষ্টব্য।

১০। **লিস্টের কনক্যাটেনেশন অপারেশন সংক্ষেপে বর্ণনা কর।**

উত্তর ঃ দুটি লিস্ট-এর উপাদানসমূহকে সংযুক্ত করে নতুন কোনো লিস্ট তৈরি করার জন্য কনক্যাটেশন অপারেটর (+) ব্যবহার করা হয়। লিস্ট-এর উপাদানসমূহ বিভিন্ন টাইপের হলেও কোনো সমস্যা নেই। যেমন–

```
List1=["Mahbub",960021,"Mahi",23022015]
List2=[19072017,"Mahdee","Tahsin"]
List3=List1+List2
print(List3)
```

১১। **লিস্টের মেম্বারশিপ অপারেশন সংক্ষেপে ব্যাখ্যা কর।**

উত্তর ঃ কোনো লিস্টের মধ্যে নির্দিষ্ট কোনো এলিমেন্টের উপস্থিতি বা অনুপস্থিতি চেক করার জন্য মেম্বারশিপ অপারেটর ব্যবহার করা হয়। পাইথনের লিস্টে দুই ধরনের মেম্বারশিপ অপারেটর ব্যবহৃত হয়। যেমন– in অপারেটর ও not in অপারেটর। কোনো লিস্টের মধ্যে নির্দিষ্ট কোনো এলিমেন্ট উপস্থিত আছে কি না সেটা চেক করার জন্য in অপারেটর ব্যবহার করা হয়। যদি এলিমেন্টটি লিস্টের মধ্যে এক বা একাধিকবার থাকে তাহলে এটি True রিটার্ন করে, অন্যথায় False রিটার্ন করে। একইভাবে লিস্টের সাথে not in অপারেটর ব্যবহার করে কোনো এলিমেন্টের অনুপস্থিতি চেক করা যায়। যেমন–

```
List1=["Mahbub",960021,"Mahi",23022015]
print("Samsul" in List1)
print("Mahdee" in List1)
```

১২। **লিস্টে ব্যবহৃত যে-কোনো তিনটি বিল্ট ইন ফাংশনের কাজ লেখ।**

উত্তর ঃ পাইথন প্রোগ্রামে লিস্ট নিয়ে দ্রুততার সঙ্গে এবং সুনির্দিষ্ট কোনো কাজ সম্পাদনের জন্য এতে বেশ কিছু বিল্ট ইন ফাংশন রয়েছে। নিম্নে লিস্টে ব্যবহৃত এরূপ তিনটি বিল্ট ইন ফাংশন ও তাদের ব্যবহার উল্লেখ করা হলো ঃ

পাইথনের লিস্টে ব্যবহৃত বিল্ট ইন ফাংশনসমূহ	
বিল্ট ইন ফাংশন	**বর্ণনা**
all()	লিস্টের সকল উপাদানের মান সত্য হলে আউটপুট হিসেবে এই ফাংশনটি True রিটার্ন করে।
any()	লিস্টের যে-কোনো একটি উপাদানের মান সত্য হলেও আউটপুট হিসেবে এই ফাংশনটি True রিটার্ন করে। লিস্টটি empty হলেই কেবল আউটপুট হিসেবে False রিটার্ন করে।
len()	লিস্টে বিদ্যমান সকল উপাদানের সংখ্যা জানার জন্য এই ফাংশনটি ব্যবহৃত হয়।

১৩। **লিস্টে ব্যবহৃত যে-কোনো তিনটি বিল্ট ইন মেথডের কাজ লেখ।**

উত্তর ঃ পাইথন লিস্টে ব্যবহৃত তিনটি বিল্ট ইন মেথডের কাজ উল্লেখ করা হলো ঃ

পাইথনের লিস্টে ব্যবহৃত মেথডসমূহ	
মেথড	**বর্ণনা**
insert()	লিস্টের নির্দিষ্ট কোনো পজিশন বা ইনডেক্সে কোনো আইটেম সংযোজনের জন্য এই মেথড ব্যবহার করা হয়।
remove()	লিস্টের নির্দিষ্ট কোনো আইটেমকে বিয়োজন বা রিমুভ করার জন্য এই মেথড ব্যবহার করা হয়।
clear()	একটি লিস্টে বিদ্যমান সকল উপাদানকে রিমুভ করার জন্য এই মেথড ব্যবহার করা হয়।

১৪। সিনট্যাক্স ও উদাহরণসহ clear() মেথডের ব্যবহার দেখাও।

উত্তর সংকেত ঃ ৮.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

১৫। সিনট্যাক্স ও উদাহরণসহ append() মেথডের ব্যবহার দেখাও।

উত্তর সংকেত ঃ ৮.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

১৬। সিনট্যাক্স ও উদাহরণসহ extend() মেথডের প্রয়োগ দেখাও।

উত্তর সংকেত ঃ ৮.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

১৭। সিনট্যাক্স ও উদাহরণসহ pop() মেথডের ব্যবহার দেখাও।

উত্তর সংকেত ঃ ৮.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

১৮। সিনট্যাক্স ও উদাহরণসহ sort() মেথডের প্রয়োগ দেখাও।

উত্তর সংকেত ঃ ৮.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

১৯। সিনট্যাক্স ও উদাহরণসহ reverse() মেথডের ব্যবহার দেখাও।

উত্তর সংকেত ঃ ৮.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

রচনামূলক প্রশ্নাবলি ঃ

১। উদাহরণসহ বিভিন্ন প্রকার ডাটা টাইপ লিস্ট অপারেশন ব্যাখ্যা কর।

উত্তর সংকেত ঃ ৮.১ নং অনুচ্ছেদ দ্রষ্টব্য।

২। উদাহরণসহ লিস্টের উপাদান অ্যাক্সেসিং পদ্ধতিসমূহ বর্ণনা কর।

উত্তর সংকেত ঃ ৮.২ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। লিস্টে কোনো উপাদান সংযোজন, বিয়োজন ও আপডেটিং পদ্ধতি বর্ণনা কর।

উত্তর সংকেত ঃ ৮.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

৪। উদাহরণসহ বিভিন্ন প্রকার লিস্ট অপারেশন ব্যাখ্যা কর।

উত্তর সংকেত ঃ ৮.৪ নং অনুচ্ছেদ দ্রষ্টব্য।

৫। লিস্টে ব্যবহৃত বিল্ট ইন ফাংশনসমূহ ব্যাখ্যা কর।

উত্তর সংকেত ঃ ৮.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

৬। লিস্টে ব্যবহৃত বিল্ট ইন মেথডসমূহ ব্যাখ্যা কর।

উত্তর সংকেত ঃ ৮.৫ নং অনুচ্ছেদ দ্রষ্টব্য।