অধ্যায়-৯

টাপল
(Tuple)

৯.০ ভূমিকা (Introduction) ঃ

লিস্টের মতোই পাইথনের আরেকটি ডাটা স্ট্রাকচার হচ্ছে Tuple. টাপলের ভ্যালু পরিবর্তন করা যায় না। লিস্ট তৈরি করতে যেমন দুটো [ ] ব্র্যাকেট ব্যবহার করতে হয় আর টাপল তৈরি করতে দুটো প্যারেনথেসিস () ব্যবহার করা হয়। তবে প্যারেনথেসিস ছাড়াও শুধু কমা চিহ্ন দিয়ে ভ্যালুগুলোকে আলাদা করেও টাপল তৈরি করা যায়। উক্ত অধ্যায়ে আমরা টাপল ও তার ব্যবহার সম্পর্কিত বিষয়াদি নিয়ে বিশদ আলোচনা করব।

৯.০.১ টাপল ও লিস্টের মধ্যে পার্থক্য (Diifference between tuple & list) ঃ

টাপল ও লিস্টের মধ্যে মূল পার্থক্য হলো টাপল হল ইমিউটেবল আর লিস্ট হলো মিউটেবল। অর্থাৎ কারণ লিস্টকে যেমন ইচ্ছেমতো পরিবর্তন, পরিবর্ধন করা যায় কিন্তু টাপলকে ইচ্ছেমতো পরিবর্তন, পরিবর্ধন করা যায় না। সাধারণত লিস্টে একই ধরনের ডাটা টাইপ ব্যবহৃত হয় কিন্তু টাপলে ভিন্ন ভিন্ন ধরনের ডাটা টাইপ ব্যবহৃত হয়।

৯.০.২ লিস্টের তুলনায় টাপলের সুবিধাসমূহ (Advantages of tuple over list) ঃ

লিস্টের তুলনায় টাপলের বেশ কিছু সুবিধা রয়েছে, যার কারণে পাইথনে টাপলকে ব্যবহার করা হয়। যেমনঃ

- টাপল লিস্টের তুলনায় faster।
- লিস্টের তুলনায় টাপলের কোডিং অধিকতর নিরাপদ।
- লিস্ট যেহেতু এক ধরনের মিউটেবল ডাটা টাইপ তাই তাকে কোনো ডিকশনারির কী হিসেবে ব্যবহার করা যায়না। কিন্তু টাপলকে ডিকশনারির কী হিসেবে ব্যবহার করা যায়।

৯.১ টাপল ও টাপলে ভ্যালু অ্যাসাইনমেন্ট (Define tuple & assigning values in tuples) ঃ

টাপল (Tuple) ঃ টাপল হচ্ছে পাইথনের অন্যতম বৈচিত্রপূর্ণ ডাটা টাইপ, যা প্যারেনথেসিস () এর ভেতর কমার সাহায্যে উপাদানসহ প্রকাশ করা হয়। টাপলের আইটেমগুলো বিভিন্ন ধরনের ডাটা টাইপের হতে পারে। ইন্টিজার, ফ্লোট, স্ট্রিং, লিস্ট, ডিকশনারি এমনকি টাপলও হতে পারে।

উদাহরণ ঃ

```
Tuple1 = ('physics', 'chemistry', 1997, 2000);
Tuple2 = (1, 2, 3, 4, 5);
Tuple 3 = ("a", "b", "c", "d");
Tuple4 = ();
Tuple5 = (50,);
```

টাপলের এলিমেন্টেগুলোকে কমা দিয়ে দিয়ে যুক্ত করা হয় এর এলিমেন্টগুলো ইনডেক্স অনুযায়ী (0,1,2,3.....) সাজানো থাকে। ইনডেক্সের মানসবসময় 0 থেকে শুরু হয়। কোনো আইটেম ছাড়াও টাপল হতে পারে। একে ফাঁকা টাপল (Empty tuple) বলে। আবার শুধুমাত্র একটি মাত্র আইটেম নিয়েও টাপল হয়। এক্ষেত্রে একমাত্র টাপল আইটেমের শেষে ও কমা বসবে। টাপলকে ও স্ট্রিং এর মতো sliced, concatenated ইত্যাদি করা যায়।

টাপলে ভ্যালু অ্যাসাইনমেন্ট বা টাপল আনপ্যাকিং (Assigning values in tuples/ Tuple unpacking) ঃ

টাপলে সাধারণত দুই ভাবে ভ্যালু অ্যাসাইন করা যায়। যথা ঃ

১। টাপলের মধ্যকার সকল ভ্যালুকে এক লাইনেই আলাদা আলাদা নতুন ভ্যারিয়েবল হিসেবে অ্যাসাইন করা যায়। যেমন–

```
Roll_Numbers = (96001, 96002, 96003)
Mahi, Mahdee, Tahsin = Roll_Numbers
print(Mahi)
print(Mahdee)
print(Tahsin)
```

আউটপুটঃ

```
96001
96002
96003
```

২। টাপলের মধ্যে কোন ভেরিয়েবলের জন্য একাধিক ভ্যালু অ্যাসাইন করতে হলে ঐ ভেরিয়েবলের পূর্বে * চিহ্ন যুক্ত করে যে কোন সংখ্যক ভ্যালুকে এর মধ্যে অ্যাসাইন করা যায়। অধিক সংখ্যক ভ্যালু থাকলে সেগুলোকে আলাদা আলাদা ভেরিয়েবল হিসেবে অ্যাসাইন করা যায়।

উদাহরণ ঃ

```
x, *y, z = (10, 20, 30, 44, 55, 66, 77, 88, 99)
print(x)
print(y)
print(z)
```

আউটপুট ঃ

```
10
(20, 30, 44, 55, 66, 77, 88)
99
```

উপরোক্ত উদাহরণের ক্ষেত্রে লক্ষ্য করলে দেখা যাবে যে, ভেরিয়েবল x এর মধ্যে 10 কে, ভেরিয়েবল y এর মধ্যে (20, 30, 44, 55, 66, 77, 88) এবং ভেরিয়েবল z এর মধ্যে 99 কে অ্যাসাইন করে দেখানো হয়েছে।

৯.১.১ টাপলের উপাদান অ্যাক্সেসিং (Define tuple & accessing values from tuple) ঃ

টাপলের উপাদান অ্যাক্সেসিং (Accessing values from tuple)

টাপলের 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 ঃ পাইথনের কোনো আইটেমকে অ্যাক্সেস করার জন্য ইনডেক্স নম্বর ব্যবহার করা হলে, তাকে ইনডেক্সিং পদ্ধতি বলে। লিস্টের ইনডেক্স নম্বর শুরু হয় 0 থেকে।

	P	Y	T	H	O	N
Indexing	0	1	2	3	4	5

উদাহরণ-১ ঃ

```
Tuple = ('P','y','t','h','o','n')
print(Tuple[0])
print(Tuple[5])
print(Tuple[2])
```

আউটপুট ঃ

```
P
n
t
```

উদাহরণ-২ ঃ

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

আউটপুট ঃ

```
Mahi
Manbub
('Mahbub', 'Mahfuz', 'Samsul', 96002)
('Mahi', 'Mahbub', 'Mahfuz', 'Samsul', 96002)
('Mahfuz', 'Samsul', 96002, 88.87)
```

নেগেটিভ ইনডেক্সিং (Negative indexing) পদ্ধতি ঃ নেগেটিভ ইনডেক্সিং এর ক্ষেত্রে সর্বশেষ আইটেম ইনডেক্স -1 দ্বারা, দ্বিতীয় সর্বশেষ আইটেমকে ইনডেক্স -2 দ্বারা বুঝায়। এভাবেই নেগেটিভ ইনডেক্সিং এর ধারা চলতে থাকে।

	P	Y	T	H	O	N
Negative Indexing	-6	-5	-4	-3	-2	-1

উদাহরণ ঃ

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

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

আউটপুট ঃ

```
('Mahbub', 'Mahfuz', 'Samsul', 96002)
('Mahi', 'Mahbub', 'Mahfuz', 'Samsul', 96002)
('Mahfuz', 'Samsul', 96002, 88.87)
```

উদাহরণ-২ ঃ

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

আউটপুট ঃ

```
('Mahfuz', 'Samsul', 96002)
('Mahi', 'Mahbub', 'Mahfuz')
(96002, 88.87)
```

৯.২ টাপলের উপাদান আপডেটিং ও ডিলেটিং (Updating & deleting tuple elements) ঃ

টাপলের উপাদান আপডেটিং (Updating tuple elements) ঃ আপডেটিং বলতে সাধারণত পরিবর্তন, পরিবর্ধন, সংযোজন বা বিয়োজন ইত্যাদি অপারেশনকে বুঝায়। পাইথনে লিস্টের মতো টাপলে ইচ্ছেমতো পরিবর্তন, পরিবর্ধন ইত্যাদি করা যায় না।

উদাহরণ-১ ঃ

```
tup1 = (12, 34.56);
tup2 = ('abc', 'xyz');
tup1[0] = 100;
print(tup1)
```

আউটপুট ঃ

```
Traceback (most recent call last):
    File "C:/Users/Mahbub CPI/Desktop/11/ToupleUpdate.py", line 3, in <module>
        tup1[0] = 100;
TypeError: 'tuple' object does not support item assignment
```

উপরোক্ত প্রোগ্রামে পাইথন এরর প্রদর্শন করছে। কারণ আমরা জানি, টাপলের কোনো মান পরিবর্তন করা যায় না কিন্তু আমরা টাপল-1 এর আইটেম 12 এর মান পরিবর্তন করে 100 করতে চেয়েছিলাম, যা পাইথনের টাপলে কখনো সম্ভব নয়। তবে পুরাতন টাপলে নতুন আইটেম যোগ করে নতুন টাপল বানাতে পারি। যেমন–

উদাহরণ ঃ

```
tup1 = (12, 34.56);
tup2 = ('abc', 'xyz');
tup3=tup1+tup2;
print(tup3)
```

আউটপুট ঃ

```
(12, 34.56, 'abc', 'xyz')
```

টাপলের উপাদান বিয়োজন (Deleting list elements) ঃ পাইথনে টাপল থেকে এর কোনো উপাদানকে রিমুভ করা যায়না। তবে সম্পূর্ন টাপলকে ডিলিট করা যায়। এজন্য পাইথনে del() ফাংশন ব্যবহার করা হয়। নিম্নে del() ফাংশন ব্যবহার করে সম্পূর্ন টাপলকে বা এর সবকটি উপাদানকে ডিলিট করা যায়।

উদাহরণ ঃ

```
tup = ('physics', 'chemistry', 1997, 2000);
print (tup)
del tup;
print ("After deleting tup : ")
print (tup)
```

আউটপুট ঃ

```
('physics', 'chemistry', 1997, 2000)
After deleting tup :
Traceback (most recent call last):
 File "C:/Users/Mahbub CPI/Desktop/11/ToupleUpdate.py", line 5, in <module>
        print (tup)
NameError: name 'tup' is not defined
```

উপরোক্ত উদাহরণে del() ফাংশন ব্যবহার করার পূর্বে একটি টাপল ('physics', 'chemistry', 1997, 2000) আউটপুট হিসেবে প্রদর্শিত হয়েছে। কিন্তু del() ফাংশন ব্যবহার করার পর আউটপুট দেখতে চাইলে এরর দেখানো হয়েছে। কারণ del() ফাংশনটি কার্যকর হওয়ার পর টাপলটি সম্পূর্ন ডিলিট হয়ে গেছে।

৯.৩ টাপলের বিভিন্ন অপারেশন (Basic tuples opreation) ঃ

পাইথনে মোট তিন ধরনের টাপল অপারেশন রয়েছে, যথা–

- কনক্যাটেশন অপারেটর
- রিপিটিশন অপারেটর
- মেম্বারশিপ অপারেটর

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

উদাহরণ-১ ঃ

```
Tuple1=("Mahfuza",96002,"Mahi",23022015)
Tuple2=(19072017,"Mahdee","Sumaiya")
Tuple3=(125,354,"Tonny", "Munny","Sumaiya")
Tuple4=Tuple1+Tuple2+Tuple3
print(Tuple4)
```

আউটপুট ঃ

```
('Mahfuza', 96002, 'Mahi', 23022015, 19072017, 'Mahdee', 'Sumaiya', 125, 354, 'Tonny', 'Munny', 'Sumaiya')
```

উদাহরণ-২ ঃ

```
Tuple1=("Mahbub",960021,"Mahi",23022015)
Tuple2=Tuple1+(19072017,"Mahdee","Tahsin")
print(Tuple2)
```

আউটপুট ঃ

```
('Mahbub', 960021, 'Mahi', 23022015, 19072017, 'Mahdee', 'Tahsin')
```

উদাহরণ-৩ ঃ

```
Tuple1=("Mahbub",960021,"Mahi",23022015)
print(Tuple1+(19072017,"Mahdee","Tahsin"))
```

আউটপুটঃ

```
('Mahbub', 960021, 'Mahi', 23022015, 19072017, 'Mahdee', 'Tahsin')
```

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

উদাহরণ ঃ

```
Tuple1=("Mahbub",960021,"Mahi",23022015)
print(Tuple1*2)
```

আউটপুট ঃ

```
('Mahbub', 960021, 'Mahi', 23022015, 'Mahbub', 960021, 'Mahi', 23022015)
```

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

উদাহরণ-১ ঃ

```
Tuple1=("Mahbub",960021,"Mahi",23022015)
Tuple2=(19072017,"Mahdee","Tahsin")
print("Samsul" in Tuple1)
print("Mahbub" in Tuple1)
print("Mahdee" in Tuple2)
print("Mahbub" in Tuple2)
```

আউটপুট ঃ

```
False
True
True
False
```

উদাহরণ-২ ঃ

```
Tuple1=("Mahbub",960021,"Mahi",23022015)
Tuple2=(19072017,"Mahdee","Tahsin")
print("Samsul"  not in Tuple1)
print("Mahdee"  not in Tuple1)
print("Mahdee" not in Tuple2)
print("Mahbub" in Tuple1)
```

আউটপুট ঃ

```
True
True
False
True
```

৯.৪ নো এনক্লোজিং ডেলিমিটার (No enclosing delimiters) ঃ

কোনো আইডেন্টিফাইং সিম্বল যেমন– লিস্টের জন্য ব্র্যাকেট [ ], টাপলের জন্য প্যারেনথেসিস () ইত্যাদি ছাড়াই শুধুমাত্র কমা (,) ব্যবহার করে একাধিক অবজেক্টের কোনো সেটকে লেখা যায় এবং আউটপুট পাওয়া যায়। এটাই হচ্ছে No Enclosing Delimiters।

উদাহরণ ঃ

```
>>> 'abc', -4.24e93, 18+6.6j, 'xyz'
('abc', -4.24e+93, (18+6.6j), 'xyz')
>>> x, y = 1, 2;
>>> "Value of x , y : ", x,y
('Value of x , y : ', 1, 2)
>>>
```

আউটপুট ঃ

```
>>> 'abc', -4.24e93, 18+6.6j, 'xyz'
('abc', -4.24e+93, (18+6.6j), 'xyz')
>>> x, y = 1, 2;
>>> "Value of x , y : ", x,y
('Value of x , y : ', 1, 2)
>>>
```

৯.৫ টাপলে ব্যবহৃত বিল্ট ইন ফাংশন ও মেথডসমূহ (Built in tuple functions and methods) ঃ

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

পাইথনের টাপলে ব্যবহৃত বিল্ট ইন ফাংশনসমূহ

বিল্ট ইন ফাংশন	বর্ণনা
sum()	টাপলে বিদ্যমান উপাদানসমূহের যোগফল নির্ণয়ের জন্য এই ফাংশনটি ব্যবহৃত হয়।
any()	টাপলের যে-কোনো একটি উপাদানের মান সত্য হলে ও আউটপুট হিসেবে এই ফাংশনটি True রিটার্ন করে। টাপলটি empty হলেই কেবল আউটপুট হিসেবে False রিটার্ন করে।
max()	টাপলে বিদ্যমান সকল উপাদানের মধ্যে সর্ববৃহৎ আইটেমটি জানার জন্য এই ফাংশনটি ব্যবহৃত হয়।
enumerate()	এই ফাংশনটি টাপলের সকল উপাদানের ইনডেক্স নম্বর ও তাদের মান Pair হিসেবে ধারণ করে।
len()	টাপলে বিদ্যমান সকল উপাদানের সংখ্যা জানার জন্য এই ফাংশনটি ব্যবহৃত হয়।
all()	টাপলের সকল উপাদানের মান সত্য হলে আউটপুট হিসেবে এই ফাংশনটি True রিটার্ন করে।
list()	পাইথনের ইটারেবল ডাটা স্ট্রাকচারসমূহকে (list, string, set, dictionary) টাপলে রূপান্তরের জন্য এই ফাংশনটি ব্যবহৃত হয়।
min()	টাপলে বিদ্যমান সকল উপাদানের মধ্যে সবচেয়ে ছোট আইটেমটি জানার জন্য এই ফাংশনটি ব্যবহৃত হয়।
sorted()	সর্টেড টাপল রিটার্নের 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 ঃ

```
tuple = (1, 3, 4, 5)
print(all(tuple))

# all values false
tuple = (0, False)
print(all(tuple))

# one false value
tuple = (1, 3, 4, 0)
print(all(tuple))

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

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 ঃ

```
tuple = (1, 3, 4, 5)
print(any(tuple))
tuple = (0, False)
print(any(tuple))
tuple = (1, 3, 4, 0)
print(any(tuple))
tuple = (0, False, 5)
print(any(tuple))
tuple = ()
print(any(tuple))
```

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 tuple
print(tuple(enumerateGrocery))
# changing the default counter
enumerateGrocery = enumerate(grocery, 10)
print(tuple(enumerateGrocery))
```

আউটপুট ঃ

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

len()ফাংশন ঃ

সিনট্যাক্স ঃ

len(testTuple))

উদাহরণ ঃ

```
testTuple = ()
print(testTuple, 'length is', len(testTuple))
testTuple = (1, 2, 3)
print(testTuple, 'length is', len(testTuple))
```

আউটপুট ঃ

```
() length is 0
(1, 2, 3) length is 3
```

tuple()ফাংশন ঃ

সিনট্যাক্স ঃ

tuple((iterable))

উদাহরণ ঃ

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

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 ঃ

```
Tuple1 = ('e', 'a', 'u', 'o', 'i')
print(sorted(Tuple1))
Tuple2 = ('e', 'a', 'u', 'o', 'i')
print(sorted(Tuple2, 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)
```

আউটপুট ঃ

```
4.5
14.5
```

টাপলে ব্যবহৃত মেথডসমূহ (Tuple methods) ঃ টাপল ম্যানিপুলেশনের জন্য পাইথনে মেথডসমূহ ব্যবহার করা হয়। পাইথনে মাত্র দুটি বিল্ট ইন মেথড ব্যবহার করা হয়। যেমন–

পাইথনের লিস্টে ব্যবহৃত মেথডসমূহ	
মেথড	বর্ণনা
index()	কোনো একটি এলিমেন্ট টাপলের কোন ইনডেক্সে অবস্থান করছে সেটা চেক করার জন্য এই মেথড ব্যবহার করা হয়।
count()	একটি টাপলে বিদ্যমান উপাদান সংখ্যা গণনা করার জন্য এই মেথড ব্যবহার করা হয়।

index() মেথড ঃ

সিনট্যাক্স ঃ

tuple.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 ঃ

tuple.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
```

৯.৫.১ পাইথনের সেট (Set in python) ঃ

পাইথনের সেট (Set in python) ঃ পাইথনে সেট হলো লিস্টের মতোই এক ধরনের ডাটা টাইপ। তবে লিস্টের সাথে সেটের পার্থক্য হলো ঃ লিস্টে একই আইটেম একাধিকবার থাকতে পারে কিন্তু সেটে একটি আইটেম কেবলমাত্র একবারই থাকতে পারে।

উদাহরণ ঃ

```
my_set = {1,2,3,4,3,2}
print(my_set)
my_set = set([1,2,3,2])
print(my_set)
```

আউটপুট ঃ

```
{1, 2, 3, 4}
{1, 2, 3}
```

পাইথনে সেট তৈরি করার জন্য আমরা { } চিহ্ন বা set() ফাংশন ব্যবহার করতে পারি। set() ফাংশন দিয়ে ফাঁকা সেট তৈরি করা যায়। কিন্তু { } চিহ্ন দিয়ে ফাঁকা সেট তৈরি করা যায় না। এই চিহ্ন ব্যবহার করে ফাঁকা ডিকশনারি তৈরি করা হয়।

উদাহরণ ঃ

```
a = { }
print(type(a))
a = set()
print(type(a))
```

আউটপুট ঃ

```
<class 'dict'>
<class 'set'>
```

সেটের উপাদান আপডেটিং (Updating set elements) ঃ পাইথন সেটে কোনো একটি এলিমেন্ট সংযোজনের জন্য add() মেথড এবং একাধিক এলিমেন্ট সংযোজনের জন্য update() মেথড ব্যবহার করা হয়।

উদাহরণ ঃ

```
my_set = {1,3}
print(my_set)
my_set.add(2)
print(my_set)
my_set.update([2,3,4])
print(my_set)
my_set.update([4,5], {1,6,8})
print(my_set)
```

আউটপুট ঃ

```
{1, 3}
{1, 2, 3}
{1, 2, 3, 4}
{1, 2, 3, 4, 5, 6, 8}
```

সেটের উপাদান বিয়োজন (Deleting set elements) ঃ পাইথনে সেটের সুনির্দিষ্ট কোনো উপাদানকে রিমুভ করার জন্য চারটি মেথড ব্যবহার করা হয়। যথা– discard () মেথড, remove() মেথড, pop() মেথড ও clear() মেথড। discard () মেথড ব্যবহার করলে আইটেমটি ঐ সেটের মেম্বার না হলে ও সেটটি অপরিবর্তিত থাকে। remove() মেথড ব্যবহার করলে এবং আইটেমটি ঐ সেটের মেম্বার না হলে প্রোগ্রামে এরর প্রদর্শন করে। pop() মেথড ব্যবহার করে সেটের প্রথম উপাদানটিকে রিমুভ করা যায় এবং সেটের সবকটি এলিমেন্টকে ডিলিট করার জন্য clear() মেথডটি ব্যবহার করা হয়।

উদাহরণ-১ ঃ

```
my_set = {1, 3, 4, 5, 6}
print(my_set)
my_set.discard(4)
print(my_set)
my_set.remove(6)
print(my_set)
my_set.discard(2)
print(my_set)
my_set.remove(6)
print(my_set)
```

আউটপুট ঃ

```
{1, 3, 4, 5, 6}
{1, 3, 5, 6}
{1, 3, 5}
{1, 3, 5}
Traceback (most recent call last):
    File "C:/Users/Mahbub CPI/Desktop/11/set.py", line 9, in <module>
        my_set.remove(6)
KeyError: 6
```

উপরোক্ত উদাহরণে সর্বশেষ কমান্ডটি ব্যাতীত প্রতিটি কমান্ডের জন্য আউটপুট পাওয়া গেছে কিন্তু সর্বশেষ কমান্ড remove(6) এর জন্য আউটপুটে এরর দেখানো হয়েছে। কারণ 6 আইটেমটি ঐ সেটের মেম্বার নয়।

উদাহরণ-২ ঃ

```
my_set = {1, 3, 4, 5, 6}
my_set.pop()
print(my_set)
my_set.remove(6)
print(my_set)
my_set.discard(2)
print(my_set)
my_set.clear()
print(my_set)
```

আউটপুট ঃ

```
{3, 4, 5, 6}
{3, 4, 5}
{3, 4, 5}
set()
```

পাইথন সেটের বিভিন্ন অপারেশন (Basic set opreation in python) ঃ

পাইথনে মোট পাঁচ ধরনের সেট অপারেশন রয়েছে, যথা–

- ইউনিয়ন অপারেশন
- ইন্টারসেকশন অপারেশন
- ডিফারেন্স অপারেশন
- সিমেট্রিক ডিফারেন্স অপারেশন ও
- মেম্বারশিপ অপারেশন

ইউনিয়ন অপারেশন (Set union) ঃ পাইথনে Union () ফাংশন ব্যবহার করে দুইটি সেটের উপাদানগুলোকে একত্রিত করে নতুন কোনো ইউনিয়ন সেট গঠন করা হয়। ইউনিয়ন অপারেশনে '|' চিহ্ন ও ব্যবহার করা হয়।

ইন্টারসেকশন অপারেশন (Intersection) ঃ পাইথনে intersection () ফাংশন ব্যবহার করে দুইটি সেটের কমন উপাদানগুলোকে নিয়ে নতুন কোনো সেট গঠন করা হয়। ইন্টারসেকশন অপারেশনে '&' চিহ্ন ও ব্যবহার করা হয়।

ডিফারেন্স অপারেশন (Difference) ঃ পাইথনে Differene () ফাংশন ব্যবহার করে দুইটি সেটের আন কমন উপাদানগুলোকে নিয়ে নতুন কোনো সেট গঠন করা হয়। তা ছাড়া ডিফারেন্স অপারেশনে '-' চিহ্ন ও ব্যবহার করা হয়।

সিমেট্রিক ডিফারেন্স অপারেশন (Symmetric difference) ঃ পাইথনে সিমেট্রিক ডিফারেন্স অপারেশনের মাধ্যমে দুইটি সেটের উপাদানগুলোর মধ্যে প্রতিটি সেটের অনুপস্থিত উপাদানগুলো নিয়ে নতুন কোনো সেট গঠন করা হয়। সিমেট্রিক ডিফারেন্স অপারেশনে '^' চিহ্ন ব্যবহার করা হয়।

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

উদাহরণ-১ ঃ

```
A = {1, 2, 3, 4, 5}
B = {4, 5, 6, 7, 8}
# Union Operation
print (A.union(B))
print(A | B)
# Intersection Operation
print (A.intersection(B))
print (B.intersection(A))
print(A & B)
# Difference Operation
print(A.difference(B))
print(B.difference(A))
print(A - B)
# Symmetric Difference Operation
print(A ^ B)
```

আউটপুট ঃ

```
{1, 2, 3, 4, 5, 6, 7, 8}
{1, 2, 3, 4, 5, 6, 7, 8}
{4, 5}
{4, 5}
{4, 5}
{1, 2, 3}
{8, 6, 7}
{1, 2, 3}
{1, 2, 3, 6, 7, 8}
```

উদাহরণ-২ ঃ

```
my_set = set("apple")
print('a' in my_set)
print('p' not in my_set)
```

আউটপুট ঃ

```
True
False
```

সেটে ব্যবহৃত বিল্ট ইন ফাংশন (Built in Functions) ঃ পাইথন প্রোগ্রামে সেট নিয়ে দ্রুততার সঙ্গে এবং সুনির্দিষ্ট কোনো কাজ সম্পাদনের জন্য এতে বেশ কিছু বিল্ট ইন ফাংশন রয়েছে। নিম্নে সেটে ব্যবহৃত এ সকল বিল্ট ইন ফাংশনসমূহ ও তাদের ব্যবহার উল্লেখ করা হলো ঃ

পাইথনের সেটে ব্যবহৃত বিল্ট ইন ফাংশনসমূহ	
বিল্ট ইন ফাংশন	**বর্ণনা**
all()	সেটের সকল উপাদানের মান সত্য হলে আউটপুট হিসেবে এই ফাংশনটি True রিটার্ন করে।
any()	সেটের যে-কোনো একটি উপাদানের মান সত্য হলেও আউটপুট হিসেবে এ ফাংশনটি True রিটার্ন করে। সেটটি empty হলেই কেবল আউটপুট হিসেবে False রিটার্ন করে।
enumerate()	এ ফাংশনটি সেটের সকল উপাদানের ইনডেক্স নম্বর ও তাদের মান Pair হিসেবে ধারণ করে।
len()	সেটে বিদ্যমান সকল উপাদানের সংখ্যা জানার জন্য এ ফাংশনটি ব্যবহৃত হয়।
max()	সেটে বিদ্যমান সকল উপাদানের মধ্যে সর্ববৃহৎ আইটেমটি জানার জন্য এ ফাংশনটি ব্যবহৃত হয়।
min()	সেটে বিদ্যমান সকল উপাদানের মধ্যে সবচেয়ে ছোট আইটেমটি জানার জন্য এ ফাংশনটি ব্যবহৃত হয়।
sorted()	সর্টেড সেট রিটার্নের জন্য এই ফাংশনটি ব্যবহৃত হয়।
sum()	সেটে বিদ্যমান উপাদানসমূহের যোগফল নির্ণয়ের জন্য এ ফাংশনটি ব্যবহৃত হয়।

সেটে ব্যবহৃত বিল্ট ইন মেথডসমূহ (Set methods) ঃ সেট 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 সেটে ব্যবহৃত মেথডসমূহ	
মেথড	**বর্ণনা**
update()	দুই বা ততোধিক সেটের উপাদানগুলোকে সংযুক্ত করে একটি আপডেট সেট তৈরির জন্য এ মেথড ব্যবহার করা হয়।
clear()	সেটে বিদ্যমান সকল উপাদানকে রিমুভ করার জন্য এ মেথড ব্যবহার করা হয়।
copy()	সেটের একটি কপি তৈরির জন্য এ মেথড ব্যবহার করা হয়।
difference()	দুই বা ততোধিক সেটের আন কমন উপাদানগুলো নিয়ে নতুন একটি সেট তৈরির জন্য এ মেথড ব্যবহার করা হয়।
add()	সেটে নতুন কোনো এলিমেন্ট যুক্ত করার জন্য এ মেথড ব্যবহার করা হয়।
difference_update()	দুই বা ততোধিক সেটের আন কমন উপাদানগুলো নিয়ে সেট আপডেটের জন্য এ মেথড ব্যবহার করা হয়।
discard()	একটি সেট হতে কোনো এলিমেন্ট রিমুভ করার জন্য এই মেথড ব্যবহার করা হয়।
intersection()	দুই বা ততোধিক সেটের কমন উপাদানগুলো নিয়ে নতুন একটি সেট তৈরির জন্য এ মেথড ব্যবহার করা হয়।
intersection_update()	দুই বা ততোধিক সেটের কমন উপাদানগুলো নিয়ে সেট আপডেটের জন্য এ মেথড ব্যবহার করা হয়।
issubset()	অন্য কোনো সেটের মধ্যে যদি এ সেট বিদ্যমান থাকে তবে True ভ্যালু রিটার্ন করার জন্য এ মেথড ব্যবহার করা হয়।
issuperset()	এই সেট যদি অন্য কোনো সেটের মধ্যে বিদ্যমান থাকে তবে True ভ্যালু রিটার্ন করার জন্য এ মেথড ব্যবহার করা হয়।
pop()	সেটে বিদ্যমান প্রথম উপাদানকে রিমুভ করার জন্য এ মেথড ব্যবহার করা হয়।
remove()	সেটে বিদ্যমান উপাদানগুলো থেকে কোনো এলিমেন্ট রিমুভ করার জন্য এ মেথড ব্যবহার করা হয়। এলিমেন্টটি এ সেটে না থাকলে এরর দেখাবে।

পাইথনের সেটে ব্যবহৃত মেথডসমূহ	
মেথড	বর্ণনা
symmetric_difference()	দুই বা ততোধিক সেটের সিমেট্রিক ডিফারেন্সকে একটি নতুন সেট হিসেবে রিটার্ন করার জন্য এ মেথড ব্যবহার করা হয়।
symmetric_difference _update()	দুই বা ততোধিক সেটের সিমেট্রিক ডিফারেন্সকে নিয়ে একটি আপডেট সেট তৈরির জন্য এ মেথড ব্যবহার করা হয়।
isdisjoint()	দুই বা ততোধিক সেটের মধ্যে কোনো কমন উপাদান না থাকলে True ভ্যালু রিটার্ন করার জন্য এ মেথড ব্যবহার করা হয়।
union()	দুই বা ততোধিক সেটের উপাদানগুলোকে সংযুক্ত করে নতুন কোনো সেট তৈরির জন্য এ মেথড ব্যবহার করা হয়।

৯.৫.২ পাইথনের ডিকশনারি (Dictionary in python) ঃ

ডিকশনারি এমন এক ধরণের ডাটা স্ট্রাকচার যাতে লিস্টের মতই বিভিন্ন ধরনের এলিমেন্ট বা অবজেক্ট স্টোর করা যায়। তবে ডিকশনারীতে এলিমেন্টগুলোকে ম্যানুয়ালি ইনডেক্সিং করতে হয়। অর্থাৎ প্রোগ্রামারকেই প্রতিটি এলিমেন্টের জন্য আলাদা আলাদা ইনডেক্স নির্ধারণ করে দিতে হয়। পাইথন্ ডিকশনারীতে দুটি কার্লী ব্র্যাকেট { } এর মধ্যে কোলন চিহ্ন (ঃ) দিয়ে কী- ভ্যালু pair তৈরি করা হয় এবং প্রতিটি pair কে কমা (,) দিয়ে আলাদা করা হয়। যেমনঃ

```
my_dictionary = {"Mahi": 96001, "Mahdee": 96002, "Tahsin": 96003}
```

ডিকশনারীর মধ্যে কোন কী-ভ্যালু pair না থাকলে তাকে ফাঁকা ডিকশনারি বলে। যেমন– my_dictionary = { }

ডিকশনারীর উপাদান অ্যাক্সেসিং (Accessing Values from Dictionary) ঃ

ডিকশনারির উপাদানসমূহকে অ্যাক্সেস করার নিয়ম অনেকটা লিস্টের মতই। লিস্টের বেলায় থার্ড ব্র্যাকেট এর মধ্যে ইনডেক্স দিয়ে যেমন ইনডেক্সের ভ্যালু পাওয়া যায়, ঠিক তেমনি ডিকশনারির বেলায় ও ইনডেক্সের যায়গায় কী ব্যবহার করে, এর সাথে pair হিসেবে থাকা ভ্যালুকে অ্যাক্সেস করা যায়।

উদাহরণঃ

```
my_dictionary={'Name':'Mahbub','Roll':960021,'Department':'Computer'}
print(my_dictionary['Name'])
print(my_dictionary['Department'])
print(my_dictionary['Roll'])
```

আউটপুটঃ

```
Mahbub
Computer
960021
```

ডিকশনারির উপাদান আপডেটিং (Updating dictionary elements) ঃ পাইথন ডিকশনারিতে কোনো এলিমেন্ট সংযোজনের জন্য কিংবা কোনো এলিমেন্টের মান পরিবর্তনের জন্য অ্যাসাইনমেন্ট অপারেটর (=) ব্যবহার করা হয়।

উদাহরণ ঃ

```
my_dict = {'Name':'Mahbub', 'Roll': 96038}
# update value
my_dict['Roll'] = 96027
print(my_dict)
# add item
my_dict['Address'] = 'Comilla'
print(my_dict)
```

আউটপুট ঃ

```
{'Name': 'Mahbub', 'Roll': 96027}
{'Name': 'Mahbub', 'Roll': 96027, 'Address': 'Comilla'}
```

ডিকশনারির উপাদান বিয়োজন (Deleting dictionary elements) ঃ পাইথনে ডিকশনারি থেকে এর উপাদানগুলোকে রিমুভ করার জন্য চারটি মেথড ব্যবহার করা হয়। যথা— pop() মেথড, popitem() মেথড, clear() মেথড ও del() মেথড। ডিকশনারির সুনির্দিষ্ট কোনো আইটেমকে রিমুভ করার জন্য pop() মেথড, ডিকশনারির arbitrary আইটেমকে রিমুভ করার জন্য popitem() মেথড, ডিকশনারির সকল আইটেম রিমুভ করার জন্য clear() মেথড ও পুরো ডিকশনারিকে মুছে দেয়ার জন্য del() মেথডটি ব্যবহার করা হয়।

উদাহরণ ঃ

```
squares = {1:1, 2:4, 3:9, 4:16, 5:25}
# remove a particular item
print(squares.pop(4))
print(squares)
# remove an arbitrary item
print(squares.popitem())
print(squares)
# delete a particular item
del squares[5]
print(squares)
# remove all items
squares.clear()
print(squares)
# delete the dictionary itself
del squares
print(squares)
```

আউটপুট ঃ

```
16
{1 : 1, 2 : 4, 3 : 9, 5 : 25}
(1, 1)
{2 : 4, 3 : 9, 5 : 25}
{2 : 4, 3 : 9}
{ }
Traceback (most recent call last):
    File "C:/Users/Mahbub CPI/Desktop/11/dictionary.py", line 9, in <module>
        del squares[5]
KeyError: 5
```

উপরোক্ত উদাহরণে সর্বশেষ কমান্ডটি ব্যাতীত প্রতিটি কমান্ডের জন্য আউটপুট পাওয়া গেছে কিন্তু সর্বশেষ কমান্ড print(squares) এর জন্য আউটপুটে এরর দেখানো হয়েছে। কারণ ডিকশনারির সকল আইটেম ডিলিট হওয়ার কারণে কোনো আইটেম দেখাতে না পেরে কম্পাইলার এক্ষেত্রে আউটপুটে এরর দেখিয়েছে।

ডিকশনারিতে ব্যবহৃত বিল্ট ইন ফাংশন (Built in functions) ঃ পাইথন প্রোগ্রামের ডিকশনারিতে দ্রুততার সঙ্গে এবং সুনির্দিষ্ট কোনো কাজ সম্পাদনের জন্য এতে বেশ কিছু বিল্ট ইন ফাংশন রয়েছে। নিম্নে ডিকশনারিতে ব্যবহৃত এ সকল বিল্ট ইন ফাংশনসমূহ ও তাদের ব্যবহার উল্লেখ করা হলো ঃ

পাইথনের ডিকশনারিতে ব্যবহৃত বিল্ট ইন ফাংশনসমূহ	
বিল্ট ইন ফাংশন	বর্ণনা
all()	ডিকশনারির সকল উপাদানের মান সত্য হলে আউটপুট হিসেবে এ ফাংশনটি True রিটার্ন করে।
any()	ডিকশনারির যে-কোনো একটি উপাদানের মান সত্য হলে ও আউটপুট হিসেবে এ ফাংশনটি True রিটার্ন করে। ডিকশনারি empty হলেই কেবল আউটপুট হিসেবে False রিটার্ন করে।
len()	ডিকশনারিতে বিদ্যমান সকল উপাদানের সংখ্যা জানার জন্য এই ফাংশনটি ব্যবহৃত হয়।
comp()	দুটি ডিকশনারির আইটেমগুলোর মধ্যে তুলনা করার জন্য এই ফাংশনটি ব্যবহৃত হয়।
sorted()	ডিকশনারির জন্য একটি নতুন সর্টেড কী লিস্ট রিটার্ন করার জন্য এই ফাংশনটি ব্যবহৃত হয়।

উদাহরণ ঃ

```
squares = {1: 1, 3: 9, 5: 25, 7: 49, 9: 81}
print(len(squares))
print(sorted(squares))
```

আউটপুট ঃ

```
5
[1, 3, 5, 7, 9]
```

ডিকশনারিতে ব্যবহৃত বিল্ট ইন মেথডসমূহ (Set methods) ঃ ডিকশনারি 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 ডিকশনারিতে ব্যবহৃত মেথডসমূহ

মেথড	বর্ণনা
popitem()	আইটেমকে রিমুভ করা ও কোনো অরবিটারি আইটেম (কী, ভ্যালু) রিটার্ন করার জন্য এই মেথড ব্যবহার করা হয়। আর ডিকশনারিটি যদি ফাঁকা হয় তাহলে কী এরর দেখাবে।
copy()	ডিকশনারির একটি কপি তৈরির জন্য এই মেথড ব্যবহার করা হয়।
values()	ডিকশনারি ভ্যালুগুলোর একটি নতুন ভিউ রিটার্ন করার জন্য এই মেথড ব্যবহার করা হয়।
fromkeys(seq[,v])	**seq** হতে **v** এর সমান কী ভ্যালুসহ নতুন একটি ডিকশনারি রিটার্ন করার জন্য এ মেথড ব্যবহার করা হয়।
get(key[,d])	কী ভ্যালু রিটার্ন করার জন্য এই মেথড ব্যবহার করা হয়। আর যদি কী অনুপস্থিত থাকে তবে **d**-এর মান (defaults to none) প্রদর্শন করে।
items()	ডিকশনারি আইটেমগুলোর একটি নতুন ভিউ রিটার্ন করার জন্য এই মেথড ব্যবহার করা হয়।
keys()	ডিকশনারি কী-গুলোর একটি নতুন ভিউ রিটার্ন করার জন্য এই মেথড ব্যবহার করা হয়।
pop(key[,d])	কী ভ্যালুসহ কোনো আইটেম রিমুভ করা এবং এর ভ্যালু রিটার্ন করার জন্য এই মেথড ব্যবহার করা হয়। আর যদি কী অনুপস্থিত থাকে ও **d** এর মান সরবরাহ না করা হয় তবে কী এরর প্রদর্শন করে।
setdefault(key[,d])	কী' টি যদি ডিকশনারিতে বিদ্যমান থাকে তবে এই ফাংশনের মাধ্যমে এর মান প্রদর্শন করা যায়। আর যদি কী' টি ডিকশনারিতে অনুপস্থিত থাকে তবে **d** এর মানসহ কী ইনসার্ট করাএবং **d** এর মান রিটার্ন করা যায়।
update([other])	অন্য কোনো ডিকশনারির (কী/ভ্যালু) pair সহ ডিকশনারি আপডেট করার জন্য এই মেথড ব্যবহার করা হয়।
clear()	ডিকশনারিতে বিদ্যমান সকল উপাদানকে রিমুভ করার জন্য এই মেথড ব্যবহার করা হয়।

উদাহরণ ঃ

```
marks = {}.fromkeys(['Math','English','Science'], 0)
# Output: {'English': 0, 'Math': 0, 'Science': 0}
print(marks)
for item in marks.items():
    print(item)
# Output: ['English', 'Math', 'Science']
list(sorted(marks.keys()))
```

আউটপুট ঃ

```
{'Math': 0, 'English': 0, 'Science': 0}
('Math', 0)
('English', 0)
('Science', 0)
```

৯.৬ টাপল ব্যবহার করে একটি প্রোগ্রাম (Simple program using tuples) ঃ

```
my_tuple = ( )                # empty tuple
print (my_tuple)              # output: ( )

my_tuple = (1, 2, 3)          # tuple having integers
print (my_tuple)              # Output: (1, 2, 3)

my_tuple = (1, "Hello", 3.4) # tuple with mixed data type.
print (my_tuple)              # output: (1, "Hello". 3.4)

my_tupel = ("Rahim", [8, 4, 6], (1, 2, 3)) # nested tuple.
print (my_tuple)              # output: ('Rahim', [8, 4, 6], (1, 2, 3))
```

অনুশীলনী-৯

অতি সংক্ষিপ্ত প্রশ্নোত্তর ঃ

১। **টাপল কী ?**

উত্তর ঃ টাপল হচ্ছে পাইথনের অন্যতম বৈচিত্র্যপূর্ণ ডাটা টাইপ, যা প্যারেনথেসিস () এর ভেতর কমার সাহায্যে উপাদানসহ প্রকাশ করা হয়।

২। **টাপল ও লিস্টের মধ্যে পার্থক্য কী?**

উত্তর ঃ টাপল ও লিস্টের মধ্যে পার্থক্য হলো– লিস্টকে যেমন ইচ্ছেমতো পরিবর্তন, পরিবর্ধন করা যায় কিন্তু টাপলকে ইচ্ছেমতো পরিবর্তন, পরিবর্ধন করা যায় না। সাধারণত লিস্টে একই ধরনের ডাটা টাইপ ব্যবহৃত হয় কিন্তু টাপলে ভিন্ন ভিন্ন ধরনের ডাটা টাইপ ব্যবহৃত হয়।

৩। **টাপলের উপাদান অ্যাক্সেস করার জন্য পাইথনে কয় ধরনের ও কী কী পদ্ধতি রয়েছে?**

উত্তর ঃ টাপলের উপাদানসমূহকে অ্যাক্সেস করার জন্য পাইথনে তিনটি পদ্ধতি প্রচলিত আছে, যথা–

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

৪। **টাপল অপারেশন কত প্রকার ও কী কী?**

উত্তর ঃ পাইথনে মোট তিন ধরনের টাপল অপারেশন রয়েছে, যথা ঃ

- কনক্যাটেশন অপারেটর
- রিপিটিশন অপারেটর
- মেম্বারশিপ অপারেটর

৫। **টাপলে কনক্যাটেনেশন অপারেটর ব্যবহার করা কেন?**

উত্তর ঃ দুটি টাপল এর উপাদানসমূহকে সংযুক্ত করে নতুন কোনো টাপল তৈরি করার জন্য কনক্যাটেশন অপারেটর (+) ব্যবহার করা হয়। এক্ষেত্রে টাপলের উপাদানসমূহ বিভিন্ন টাইপের হলেও কোনো সমস্যা নেই।

৬। **রিপিটিশন অপারেটরের ব্যবহার উল্লেখ কর।**

উত্তর ঃ টাপলে বিদ্যমান উপাদানসমূহকে একাধিক বার রিপিট করার জন্য পাইথনে রিপিটেশন অপারেটর (*) ব্যবহার করা হয়। এক্ষেত্রেও টাপলের উপাদানসমূহ বিভিন্ন টাইপের হলেও কোনো সমস্যা নেই।

৭। **No enclosing delimeter বলতে কী বুঝায়?**

উত্তর ঃ কোনো আইডেন্টিফাইং সিম্বল যেমন– লিস্টের জন্য ব্র্যাকেট [ ], টাপলের জন্য প্যারেনথেসিস () ইত্যাদি ছাড়াই শুধুমাত্র কমা (,) ব্যবহার করে একাধিক অবজেক্টের কোনো সেটকে লেখা যায় এবং আউটপুট পাওয়া যায়। এটাই হচ্ছে No Enclosing Delimiters।

৮। **টাপলে বিল্ট ইন ফাংশন ব্যবহারের সুবিধা কী?**

উত্তর ঃ টাপলে বিল্ট ইন ফাংশন ব্যবহার করে দ্রুততার সঙ্গে সুনির্দিষ্ট কোনো কাজ সম্পাদন করা যায়।

৯। **টাপলে list() ফাংশনের কাজ কী?**

উত্তর ঃ পাইথনের ইটারেবল ডাটা স্ট্রাকচারসমূহকে (list, string, set, dictionary) টাপলে রূপান্তরের জন্য এই ফাংশনটি ব্যবহৃত হয়।

১০। **টাপলে sorted() ফাংশনের কাজ কী?**

উত্তর ঃ সর্টেড টাপল রিটার্নের জন্য sorted() ফাংশনটি ব্যবহৃত হয়।

১১। **টাপলে enumerate() ফাংশনের কাজ কী?**

উত্তর ঃ এই ফাংশনটি টাপলের সকল উপাদানের ইনডেক্স নম্বর ও তাদের মান Pair হিসেবে ধারণ করে।

১২। **টাপলে max() ফাংশনের কাজ কী?**

উত্তর ঃ টাপলে বিদ্যমান সকল উপাদানের মধ্যে সর্ববৃহৎ আইটেমটি জানার জন্য max() ফাংশন ব্যবহৃত হয়।

১৩। **টাপলে min() ফাংশনের কাজ কী?**

উত্তর ঃ টাপলে বিদ্যমান সকল উপাদানের মধ্যে সবচেয়ে ছোট আইটেমটি জানার জন্য min() ফাংশনটি ব্যবহৃত হয়।

১৪। **টাপলে ব্যবহৃত বিল্ট ইন মেথড কয়টি কী কী?**

উত্তর ঃ পাইথনের টাপলে মাত্র দুটি বিল্ট ইন মেথড ব্যবহার করা হয়, যথা– index() ও count()।

১৫। **টাপলে ব্যবহৃত বিল্ট ইন মেথডদ্বয়ের কাজ কী?**

উত্তর ঃ টাপলে ব্যবহৃত বিল্ট ইন মেথডদ্বয়ের কাজ নিম্নরূপ ঃ

মেথড	কাজ
index()	কোনো একটি এলিমেন্ট টাপলের কোনো ইনডেক্সে অবস্থান করছে সেটা চেক করার জন্য এই মেথড ব্যবহার করা হয়।
count()	একটি টাপলে বিদ্যমান উপাদান সংখ্যা গণনা করার জন্য এই মেথড ব্যবহার করা হয়।

১৬। **লিস্ট ও সেটের মধ্যে পার্থক্য কী?**

উত্তর ঃ লিস্টের সাথে সেটের পার্থক্য হলো ঃ লিস্টে একই আইটেম একাধিকবার থাকতে পারে কিন্তু সেটে একটি আইটেম কেবলমাত্র একবারই থাকতে পারে।

১৭। **সেটে উপাদান সংযোজনের জন্য ব্যবহৃত মেথডসমূহের নাম লেখ।**

উত্তর ঃ পাইথন সেটে কোনো এলিমেন্ট সংযোজনের জন্য দুটি মেথড ব্যবহৃত হয়। যথা– add() মেথড এবং update() মেথড। একটি এলিমেন্ট সংযোজনের জন্য add() মেথড এবং একাধিক এলিমেন্ট সংযোজনের জন্য update() মেথড ব্যবহার করা হয়।

১৮। **পাইথনে কয় ধরনের ও কী কী সেট অপারেশন আছে?**

উত্তর পাইথনে মোট পাঁচ ধরনের সেট অপারেশন রয়েছে, যথা–

- ইউনিয়ন অপারেশন
- ইন্টারসেকশন অপারেশন
- ডিফারেন্স অপারেশন
- সিমেট্রিক ডিফারেন্স অপারেশন ও
- মেম্বারশিপ অপারেশন।

১৯। **পাইথনের সেট অপারেশনে ব্যবহৃত হয় এমন কয়েকটি বিল্ট ইন ফাংশনের নাম লেখ।**

উত্তর পাইথনের সেট অপারেশনে ব্যবহৃত হয় এমন কয়েকটি উল্লেখযোগ্য বিল্ট ইন ফাংশন হলো : all(), any(), enumearate(), len(), max(), min(), sorted()এবং sum()।

২০। **পাইথনের সেট অপারেশনে ব্যবহৃত হয় এমন কয়েকটি বিল্ট ইন মেথডের নাম লেখ।**

উত্তর পাইথনের সেট অপারেশনে ব্যবহৃত হয় এমন কয়েকটি উল্লেখযোগ্য বিল্ট ইন মেথড হলো : add(), clear(), copy(), difference(), discard(), update(), isdisjoint(), intersection(), issubset(), issuperset(), union ইত্যাদি।

২১। **সেটে difference() মেথডের কাজ কী?**

উত্তর দুই বা ততোধিক সেটের আনকমন উপাদানগুলো নিয়ে নতুন একটি সেট তৈরির জন্য **difference()** মেথড ব্যবহার করা হয়।

২২। **সেটে discard() মেথডের কাজ কী?**

উত্তর একটি সেট হতে কোনো এলিমেন্ট রিমুভ করার জন্য এই মেথড ব্যবহার করা হয়।

২৩। **সেটে intersection() মেথডের কাজ কী?**

উত্তর দুই বা ততোধিক সেটের কমন উপাদানগুলো নিয়ে নতুন একটি সেট তৈরির জন্য এই মেথড ব্যবহার করা হয়।

২৪। **সেটে union() মেথডের কাজ কী?**

উত্তর দুই বা ততোধিক সেটের উপাদানগুলোকে সংযুক্ত করে নতুন কোনো সেট তৈরির জন্য union() মেথড ব্যবহার করা হয়।

২৫। **সেটে difference_update() মেথডের কাজ কী?**

উত্তর দুই বা ততোধিক সেটের আনকমন উপাদানগুলো নিয়ে সেট আপডেটের জন্য difference_update() মেথড ব্যবহার করা হয়।

২৬। **সেটে isdisjoint() মেথডের কাজ কী?**

উত্তর দুই বা ততোধিক সেটের মধ্যে কোনো কমন উপাদান না থাকলে True ভ্যালু রিটার্ন করার জন্য isdisjoint() মেথড ব্যবহার করা হয়।

২৭। **সেটে issubset() মেথডের কাজ কী?**

উত্তর অন্য কোনো সেটের মধ্যে যদি এই সেট বিদ্যমান থাকে তবে True ভ্যালু রিটার্ন করার জন্য issubset() মেথড ব্যবহার করা হয়।

২৮। **সেটে issuperset() মেথডের কাজ কী?**

উত্তর এই সেট যদি অন্য কোনো সেটের মধ্যে বিদ্যমান থাকে তবে True ভ্যালু রিটার্ন করার জন্য issuperset() মেথড ব্যবহার করা হয়।

২৯। **ডিকশনারি বলতে কী বুঝায়?**

উত্তর ঃ ডিকশনারি এমন এক ধরনের ডাটা স্ট্রাকচার, যার মধ্যেও লিস্টের মতো বিভিন্ন রকম এলিমেন্ট বা অবজেক্ট স্টোর করা যায়। কিন্তু এক্ষেত্রে এর এলিমেন্টগুলোকে ম্যানুয়ালি ইনডেক্সিং করতে হয়। অর্থাৎ প্রোগ্রামারকেই প্রতিটি এলিমেন্টের জন্য একটি key বা ইনডেক্স নির্ধারণ করে দিতে হয়।

৩০। **পাইথন প্রোগ্রামে ডিকশনারি কেন ব্যবহার করা হয়?** [বাকাশিবো-২০১৭]

উত্তর ঃ ডিকশনারি এমন এক ধরনের ডাটা স্ট্রাকচার যার মধ্যে ডাটাগুলো জোড়ায় জোড়ায় থাকে। জোড়ার প্রথম অংশকে বলা হয় কি (key) আর দ্বিতীয় অংশকে বলা হয় ভ্যালু (Value) এ ভ্যালুগুলোকে অ্যাক্সেস করা হয় নির্দিষ্ট কি (key) ভ্যালু দিয়ে। অর্থাৎ পাইথন প্রোগ্রামে কোনো ডাটা ম্যানুয়ালি সার্চ করার প্রয়োজন হলে ডিকশনারি ব্যবহার করা হয়।

৩১। **ডিকশনারিতে ব্যবহৃত হয় এমন যে-কোনো তিনটি বিল্ট ইন মেথডের নাম লেখ।**

উত্তর ঃ ডিকশনারিতে ব্যবহৃত হয় এমন যে-কোনো তিনটি বিল্ট ইন মেথডের নাম হলো ঃ clear(), copy() এবং popitem()

সংক্ষিপ্ত প্রশ্নোত্তর ঃ

১। **লিস্টের তুলনায় টাপলের সুবিধাসমূহ কী?**

উত্তর ঃ লিস্টের তুলনায় টাপলের বেশ কিছু সুবিধা রয়েছে, যার কারণে পাইথনে টাপলকে ব্যবহার করা হয়। যেমন–

- টাপল লিস্টের তুলনায় faster।
- লিস্টের তুলনায় টাপলের কোডিং অধিকতর নিরাপদ।
- লিস্ট যেহেতু এক ধরনের মিউটেবল ডাটা টাইপ, তাই তাকে কোনো ডিকশনারির কী হিসেবে ব্যবহার করা যায় না। কিন্তু টাপলকে ডিকশনারির কী হিসেবে ব্যবহার করা যায়।

২। **টাপলে ভ্যালু অ্যাসাইনমেন্ট বা টাপল আনপ্যাকিং পদ্ধতি বর্ণনা কর।**

উত্তর সংকেত ঃ ৯.১ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। **টাপলের ইনডেক্স পদ্ধতির উদাহরণ দাও।**

উত্তর ঃ পাইথনের কোনো আইটেমকে অ্যাক্সেস করার জন্য ইনডেক্স নম্বর ব্যবহার করা হয়। লিস্টের ইনডেক্স নম্বর শুরু হয় 0 থেকে।

	P	Y	T	H	O	N
Indexing	0	1	2	3	4	5

উদাহরণ ঃ

```
Tuple = ('P','y','t','h','o','n')
print(Tuple[0])
print(Tuple[5])
```

৪। **টাপলের নেগেটিভ ইনডেক্স পদ্ধতির উদাহরণ দাও।**

উত্তর ঃ নেগেটিভ ইনডেক্সিং এর ক্ষেত্রে সর্বশেষ আইটেম ইনডেক্স –1 দ্বারা, দ্বিতীয় সর্বশেষ আইটেমকে ইনডেক্স –2 দ্বারা বুঝায়। এভাবেই নেগেটিভ ইনডেক্সিং এর ধারা চলতে থাকে।

	P	Y	T	H	O	N
Negative Indexing	-6	-5	-4	-3	-2	-1

উদাহরণ ঃ

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

৫। টাপলে del() ফাংশনের প্রয়োগ দেখাও।

উত্তর ঃ সম্পূর্ণ টাপলকে বা এর সবকটি উপাদানকে ডিলিট করার জন্য পাইথনে del() ফাংশন ব্যবহার করা হয়। যেমন—

```
tup = ('physics', 'chemistry', 1997, 2000);
del tup;
```

৬। টাপলের মেম্বারশিপ অপারেটরের ব্যাখ্যা কর।

উত্তর সংকেত ঃ ৯.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

৭। সিনট্যাক্স ও উদাহরণসহ index() মেথড ব্যাখ্যা কর।

উত্তর সংকেত ঃ ৯.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

৮। সিনট্যাক্স ও উদাহরণসহ count() মেথড ব্যাখ্যা কর।

উত্তর সংকেত ঃ ৯.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

৯। সেটের উপাদানসমূহ রিমুভ করার জন্য ব্যবহৃত মেথডসমূহের কাজ উল্লেখ কর।

উত্তর ঃ পাইথনে টাপল থেকে এর সুনির্দিষ্ট কোনো উপাদানকে রিমুভ করার জন্য চারটি মেথড ব্যবহার করা হয়। যথা— discard () মেথড, remove() মেথড, pop() মেথড ও clear() মেথড। discard () মেথড ব্যবহার করলে আইটেমটি ঐ সেটের মেম্বার না হলে ও সেটটি অপরিবর্তিত থাকে। remove() মেথড ব্যবহার করলে এবং আইটেমটি ঐ সেটের মেম্বার না হলে প্রোগ্রামে এরর প্রদর্শন করে। pop() মেথড ব্যবহার করে সেটের প্রথম উপাদানটিকে রিমুভ করা যায় এবং সেটের সবকটি এলিমেন্টকে ডিলিট করার জন্য clear() মেথডটি ব্যবহার করা হয়।

১০। বিভিন্ন প্রকার সেট অপারেশন ব্যাখ্যা কর।

উত্তর ঃ পাইথনে মোট পাঁচ ধরনের সেট অপারেশন রয়েছে। যথা—

- ইউনিয়ন অপারেশন
- ইন্টারসেকশন অপারেশন
- ডিফারেন্স অপারেশন
- সিমেট্রিক ডিফারেন্স অপারেশন ও
- মেম্বারশিপ অপারেশন

ইউনিয়ন অপারেশন (Set union) ঃ পাইথনে Union () ফাংশন ব্যবহার করে দুইটি সেটের উপাদানগুলোকে একত্রিত করে নতুন কোনো ইউনিয়ন সেট গঠন করা হয়। ইউনিয়ন অপারেশনে '|' চিহ্নও ব্যবহার করা হয়।

ইন্টারসেকশন অপারেশন (Intersection) ঃ পাইথনে intersection () ফাংশন ব্যবহার করে দুইটি সেটের কমন উপাদানগুলোকে নিয়ে নতুন কোনো সেট গঠন করা হয়। ইন্টারসেকশন অপারেশনে '&' চিহ্ন ও ব্যবহার করা হয়।

ডিফারেন্স অপারেশন (Difference) ঃ পাইথনে Differene () ফাংশন ব্যবহার করে দুইটি সেটের আনকমন উপাদানগুলোকে নিয়ে নতুন কোনো সেট গঠন করা হয়। তা ছাড়া ডিফারেন্স অপারেশনে '-' চিহ্ন ও ব্যবহার করা হয়।

সিমেট্রিক ডিফারেন্স অপারেশন (Symmetric Difference) ঃ পাইথনে সিমেট্রিক ডিফারেন্স অপারেশনের মাধ্যমে দুইটি সেটের উপাদানগুলোর মধ্যে প্রতিটি সেটের অনুপস্থিত উপাদানগুলো নিয়ে নতুন কোনো সেট গঠন করা হয়। সিমেট্রিক ডিফারেন্স অপারেশনে '^' চিহ্ন ব্যবহার করা হয়।

১১। **পাইথন সেট অপারেশনে ব্যবহৃত যে-কোনো তিনটি বিল্ট ইন ফাংশন বর্ণনা কর।**

উত্তর ঃ পাইথন সেট অপারেশনে ব্যবহৃত তিনটি বিল্ট ইন ফাংশনের বর্ণনা নিম্নরূপ ঃ

পাইথনের সেটে ব্যবহৃত বিল্ট ইন ফাংশনসমূহ	
বিল্ট ইন ফাংশন	বর্ণনা
len()	সেটে বিদ্যমান সকল উপাদানের সংখ্যা জানার জন্য এই ফাংশনটি ব্যবহৃত হয়।
max()	সেটে বিদ্যমান সকল উপাদানের মধ্যে সর্ববৃহৎ আইটেমটি জানার জন্য এ ফাংশনটি ব্যবহৃত হয়।
min()	সেটে বিদ্যমান সকল উপাদানের মধ্যে সবচেয়ে ছোট আইটেমটি জানার জন্য এ ফাংশনটি ব্যবহৃত হয়।

১২। **পাইথন সেট অপারেশনে ব্যবহৃত যে-কোনো তিনটি বিল্ট ইন মেথড বর্ণনা কর।**

উত্তর ঃ পাইথন সেট অপারেশনে ব্যবহৃত তিনটি বিল্ট ইন মেথডের বর্ণনা নিম্নরূপ ঃ

পাইথনের সেটে ব্যবহৃত মেথডসমূহ	
মেথড	বর্ণনা
difference()	দুই বা ততোধিক সেটের আনকমন উপাদানগুলো নিয়ে নতুন একটি সেট তৈরির জন্য এই মেথড ব্যবহার করা হয়।
intersection()	দুই বা ততোধিক সেটের কমন উপাদানগুলো নিয়ে নতুন একটি সেট তৈরির জন্য এই মেথড ব্যবহার করা হয়।
union()	দুই বা ততোধিক সেটের উপাদানগুলোকে সংযুক্ত করে নতুন কোনো সেট তৈরির জন্য এই মেথড ব্যবহার করা হয়।

১৩। **ডিকশনারির উপাদান অ্যাক্সেসিং পদ্ধতি উল্লেখ কর।**

উত্তর সংকেত ঃ ৯.৫.২ নং অনুচ্ছেদ দ্রষ্টব্য।

১৪। **ডিকশনারির উপাদান আপডেটিং প্রক্রিয়া উদাহরণসহ বর্ণনা কর।**

উত্তর সংকেত ঃ ৯.৫.২ নং অনুচ্ছেদ দ্রষ্টব্য।

১৫। **ডিকশনারিতে ব্যবহৃত হয় এমন যে-কোনো তিনটি বিল্ট ইন ফাংশন বর্ণনা কর।**

উত্তর সংকেত ঃ ৯.৫.২ নং অনুচ্ছেদ দ্রষ্টব্য।

১৬। **ডিকশনারিতে ব্যবহৃত হয় এমন যে-কোনো তিনটি বিল্ট ইন মেথড বর্ণনা কর।**

উত্তর সংকেত ঃ ৯.৫.২ নং অনুচ্ছেদ দ্রষ্টব্য।

রচনামূলক প্রশ্নাবলি ঃ

১। **উদাহরণসহ টাপলের উপাদান অ্যাক্সেসিং পদ্ধতিসমূহ আলোচনা কর।**

উত্তর সংকেত ঃ ৯.১ নং অনুচ্ছেদ দ্রষ্টব্য।

২। **টাপলের বিভিন্ন অপারেশন ব্যাখ্যা কর।**

উত্তর সংকেত ঃ ৯.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। **টাপলে ব্যবহৃত বিল্ট ইন ফাংশনসমূহ বর্ণনা কর।**

উত্তর সংকেত ঃ ৯.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

৪। **সেটে উপাদান সংযোজন ও বিয়োজন প্রক্রিয়া উদাহরণসহ বর্ণনা কর।**

উত্তর সংকেত ঃ ৯.২ নং অনুচ্ছেদ দ্রষ্টব্য।

৫। **ডিকশনারির উপাদান অ্যাক্সেসিং, আপডেটিং ও বিয়োজন প্রক্রিয়া উদাহরণসহ বর্ণনা কর।**

উত্তর সংকেত ঃ ৯.৫.২ নং অনুচ্ছেদ দ্রষ্টব্য।