অধ্যায়-১১

ফাইলের ইনপুট আউটপুট অপারেশন
(Files Input Output Operation)

১১.০ ভূমিকা (Introduction) ঃ

কোনো সমস্যা সমাধানের জন্যই প্রোগ্রাম তৈরি করা হয়। প্রোগ্রামিং এ ইনপুট আউটপুট অপারেশন খুবই গুরুত্বপূর্ণ বিষয়। কারণ প্রোগ্রামে যদি সঠিক পদ্ধতিতে ডাটা ইনপুট ও আউটপুট না করা যায় তাহলে প্রোগ্রামিং-এর মূল উদ্দেশ্য ব্যাহত হবে। পাইথনে ইনপুট আউটপুট অপারেশনের জন্য বিভিন্ন ধরনের বিল্ট ইন ফাংশন রয়েছে। আবার স্থায়ীভাবে তথ্য সংরক্ষণ করা এবং প্রয়োজনে তথ্যসমূহ রিড (read) করার জন্য ফাইল অপারেশন প্রয়োজন। ফাইল অপারেশন বলতে কোনো ফাইলে ইনপুট-আউটপুট কার্যাদি (ডাটা পড়া, লেখা, পরিবর্তন করা, পরিমার্জন করা ইত্যাদি) সম্পাদনকে বুঝায়। উক্ত অধ্যায়ে স্ট্যান্ডার্ড ইনপুট আউটপুট অপারেশন, ফাইল অপারেশন ও তাদের অন্যান্য বিষয়াদি নিয়ে বিশদ আলোচনা করা হয়েছে।

১১.০.১ পাইথনের বেসিক ইনপুট আউটপুট ফাংশনসমূহ (Basic input output functions) ঃ

যে-কোনো ইনপুট আউটপুট অপারেশনের জন্য প্রোগ্রাম এবং কম্পিউটারের বিভিন্ন ইনপুট আউটপুট ডিভাইসের মধ্যে নিবিড় সংযোগ থাকা আবশ্যক। প্রোগ্রাম এবং কম্পিউটারের বিভিন্ন ইনপুট আউটপুট ডিভাইসের মধ্যে ইন্টারফেসের জন্য প্রোগ্রামিং ল্যাংগুয়েজে ইনপুট আউটপুট ফাংশন ব্যবহার করা হয়। প্রোগ্রামে কী-বোর্ড, মাউস কিংবা অন্য কোনো ইনপুট ডিভাইস হতে ডাটা সরবরাহের জন্য ইনপুট ফাংশন ব্যবহার করা হয় আর প্রয়োজনীয় প্রক্রিয়াকরণ শেষে মনিটর, প্রিন্টার বা অন্য কোনো আউটপুট ডিভাইসে আউটপুট পাঠানোর জন্য আউটপুট ফাংশন ব্যবহার করা হয়। পাইথনে ডাটা ইনপুট এবং আউটপুট এর জন্য কিছু স্পেশাল ফাংশন ব্যবহার করা হয়। ইনপুট এবং আউটপুট এর বেসিক ফাংশনগুলো হলো ঃ

- input() ফাংশন ও
- print() ফাংশন।

১১.১ স্ক্রিনে আউটপুট প্রদর্শন (Printing to the screen) ঃ

স্ক্রিনের মধ্যে ডাটা আউটপুটের জন্য পাইথনে print() ফাংশন ব্যবহার করা হয়। print() ফাংশন কেবলমাত্র তাই আউটপুট দেয়, যা এর আর্গুমেন্ট হিসেবে দেয়া হয়।

উদাহরণ-১ ঃ

```
>>> print('Hello, How are You? ')
```

আউটপুট ঃ

```
Hello, How are You?
```

উদাহরণ-২ ঃ

```
>>> print(12+5)
```

আউটপুট ঃ

```
17
```

উদাহরণ-৩ ঃ

```
>>> print('Country Name: Bangladeash\nTotal Population: 160Million')
```

আউটপুট ঃ

```
Country Name: Bangladeash
Total Population: 160Million
```

print() ফাংশন এর সিনট্যাক্স (Syntax) ঃ print() ফাংশনের সিনট্যাক্স বা গঠন বা ফরম্যাট হলো নিম্নরূপ ঃ
print(*objects, sep=' ', end='\n', file=sys.stdout, flush=False)
যেখানে-

- 'objects' হলো সেই ভ্যালু, যাকে প্রিন্ট করতে হবে।
- 'sep' হলো একাধিক ভ্যালুর মধ্যকার সেপারেটর। সেপারেটর হিসেবে বাই ডিফল্ট স্পেস ক্যারেক্টার ব্যবহৃত হয়।
- সকল ভ্যালু প্রিন্ট হওয়ার পর 'end' প্রিন্ট হয় এবং কার্সরকে নতুন লাইনে নিয়ে যায়।
- 'file' হলো সেই ভ্যালু যেখানে ভ্যালুসমূহ প্রিন্ট হবে। ফাইলের ডিফল্ট ভ্যালু হলো 'sys.stdout' বা স্ক্রিন।

উদাহরণ-১ ঃ

```
print(1,2,3,4)
```

আউটপুট ঃ

```
1 2 3 4
```

উদাহরণ-২ ঃ

```
print(1,2,3,4,sep='*')
```

আউটপুট ঃ

```
1*2*3*4
```

উদাহরণ-৩ ঃ

```
print(1,2,3,4,sep='#',end='&')
```

আউটপুট ঃ

```
1#2#3#4&
```

আউটপুট ফরম্যাটিং (Output formatting) ঃ কোনো পার্টিকুলার ফরম্যাটে আউটপুট প্রদর্শনের জন্য কিংবা আউপুটকে অধিকতর দৃষ্টিনন্দন করার প্রক্রিয়াকে আউটপুট ফরম্যাটিং বলে। যে সমস্ত স্টেটমেন্ট বা ফাংশন ব্যবহার করে আউটপুট ফরম্যাটিং করা হয়, তাদেরকে ফরম্যাটেড আউটপুট স্টেটমেন্ট বা ফরম্যাটেড আউটপুট ফাংশন বলে। পাইথনে আউটপুট ফরম্যাটিং-এর জন্য str.format() ব্যবহার করা হয়। format() মেথডের মাধ্যমে একটি স্ট্রিং এর মধ্যে থাকা কিছু আরগুমেন্টকে রিপ্লেস বা সাবস্টিটিউট করা যায়। format() মেথডের মধ্যকার প্রত্যেকটি আরগুমেন্ট দিয়ে এর সামনে থাকা স্ট্রিং এর মধ্যকার প্লেস হোল্ডারগুলোকে রিপ্লেস করা হয়। প্লেস হোল্ডারগুলো { } এর সাথে ইনডেক্স বা নাম ব্যবহার করে ডিফাইন করা হয়।

উদাহরণ-১ ঃ

```
x = 5; y = 10
print('The value of x is {} and y is {}'.format(x,y))
```

আউটপুট ঃ

```
The value of x is 5 and y is 10
```

উদাহরণ-২ ঃ

```
print('I love {0} and {1}'.format('baba','maa'))
```

আউটপুট ঃ

```
I love baba and maa
```

উদাহরণ-৩ ঃ

```
print('I love {1} and {0}'.format('baba','maa'))
```

আউটপুট ঃ

```
I love maa and baba
```

উপরোক্ত উদাহরণগুলো লক্ষ্য করলে দেখা যাবে যে, ফরম্যাটিংয়ের সময় ইন্ডেক্সগুলো ০, ১, ২.... এভাবে সিরিয়ালিই দিতে হবে ব্যাপারটি এমন নয়, এগুলো আগে পরে কিংবা একাধিকবার ও ব্যবহার করা যায়।

format() মেথডের মধ্যে নাম ওয়ালা আর্গুমেন্ট পাঠিয়ে স্ট্রিং এর মধ্যকার প্লেস হোল্ডার গুলোতে সেই নামে সেগুলোকে ব্যবহার করেও কাজ করা যায়।

উদাহরণ-৪ ঃ

```
message = "If x = {x} and y = {y}, then x+y = {z}".format(x =
20, y = 300, z = 20+300)
print(message)
```

আউটপুট ঃ

```
If x = 20 and y = 300, then x+y = 320
```

format() মেথডের মধ্যে সি প্রোগ্রামিং-এর মতো % অপারেটর ব্যবহার করেও কাজ করা যায়।

উদাহরণ-৫ ঃ

```
x = 12.3456789
print('The value of x is %3.2f' %x)
```

আউটপুট ঃ

```
The value of x is 12.35
```

উদাহরণ-৬ ঃ

```
x = 12.3456789
print('The value of x is %3.4f' %x)
```

আউটপুট ঃ

```
The value of x is 12.3457
```

১১.২ কী-বোর্ড থেকে ইনপুট গ্রহণ (Reading keyboard input) ঃ

পাইথন প্রোগ্রামিং-এর ক্ষেত্রে প্রোগ্রামে দুইভাবে মান ইনপুট নেয়া যায়, যথা–

- সরাসরি (Directly)
- ইউজার হতে (From keyboard)

সরাসরি (Directly) ঃ অন্যান্য প্রোগ্রামিং ল্যাংগুয়েজের মতো পাইথনেও সরাসরি মান ইনপুট নেয়া যায়। এজন্য ভেরিয়েবল ডিক্লারেশনের সময় অ্যাসাইনমেন্ট অপারেটর ব্যবহার করে মান ইনপুট নেয়া হয়।

উদাহরণ ঃ

```
x=5
y=4
print (x)
print (y)
```

আউটপুট ঃ

```
5
4
```

ইউজার হতে মান গ্রহণ (From keyboard) ঃ পাইথনে ইউজারের মাধ্যমে কী-বোর্ড থেকে ইনপুট গ্রহণের জন্য দুই ধরনের ফাংশন ব্যবহার করা হয়, যথা–

- raw_ input() ফাংশন এবং
- input() ফাংশন

raw_ input() ফাংশন ঃ raw_ input() ফাংশন ব্যবহার করে পাইথন প্রোগ্রামে ডাটা ইনপুট নেয়া হয়। তবে পাইথন ভার্সন-৩ কিংবা এর পরবর্তী ভার্সনসমূহে এই ফাংশনটি কাজ করে না। raw_ input() ফাংশনের গঠন নিম্নরূপ ঃ

raw_input([prompt])

যেখানে– 'prompt' হলো সেই ভ্যালু, যার মান ইনপুট হিসেবে গ্রহণ করা হবে। এটি একটি optional অংশ।

উদাহরণ ঃ

```
str = raw_input("Enter your Name: ");
print ("Your Name is : ", str)
```

আউটপুট ঃ

```
Enter your Name:  Mahbub Alam
Your Name is :   Mahbub Alam
```

input() ফাংশন ঃ input() ফাংশনের মাধ্যমে কী-বোর্ড থেকে কোনো মান, ক্যারেক্টার, শব্দ (word) ইত্যাদি ইনপুট নিয়ে কোনো ভেরিয়েবলে সংরক্ষণ করা যায় এবং পরে এগুলোকে নিয়ে প্রসেস করে আউটপুটে প্রদর্শন করা যায়।

input() ফাংশন এর সিনট্যাক্স (Syntax) ঃ input() ফাংশনের সিনট্যাক্স বা গঠন বা ফরম্যাট হলো নিম্নরূপ ঃ

input([prompt])

যেখানে-

'prompt' হলো সেই ভ্যালু, যার মান ইনপুট হিসেবে গ্রহণ করা হবে। এটি একটি optional অংশ।

উদাহরণ-১ ঃ

```
>>> input("What is your Name:")
```

আউটপুট ঃ

```
What is your Name:
```

এক্ষেত্রে পাইথন ইন্টারপ্রিটার যখন উপরোক্ত কোডটি এক্সিকিউট করবে তখন ইন্টারপ্রিটার ইউজার কর্তৃক প্রদত্ত ইনপুট এর জন্য অপেক্ষা করবে। ইন্টারপ্রিটারটি ততক্ষণ পর্যন্ত অপেক্ষা করবে ইউজার যতক্ষণ না Enter বাটন প্রেস করবে কিংবা ইনপুটে আরেকটি নিউলাইন ক্যারেক্টার না আসবে। উপরোক্ত উদাহরণের ক্ষেত্রে বলা যায়, ইউজার যখন 'What is your Name:' এর পর Mahbub লিখে এন্টার কী প্রেস করবে তখন সে আউটপুটে Mahbub নামটি প্রদর্শন করবে।

```
>>> input("What is your Name:")
What is your Name: Mahbub
'Mahbub'
```

উদাহরণ-২ ঃ

```
a=input("Enter the Value of a:")
b=input("Enter the Value of b:")
if(a>b):
    print("a is greater than b")
else:
    print("b is greater than a")
```

আউটপুট ঃ

```
Enter the Value of a:15
Enter the Value of b:25
b is greater than a
```

১১.৩ ফাইল ও ফাইল অপারেশন (Describe file and file operation) :

ফাইল (File) : ফাইল বা ডাটা ফাইল হচ্ছে স্টোরেজ ডিভাইসের (Hard disk, Floppy disk, Compact disk, Digital video disk ইত্যাদি) এমন একটি স্পেস (space) যেখানে স্থায়ীভাবে তথ্য সংরক্ষণ করা যায় এবং প্রয়োজনে তথ্যসমূহ রিড (read) করা যায়।

ফাইলের প্রকারভেদ (Classification of data files) : ডাটা ফাইল প্রধানত দুই প্রকার, যথা–

- স্ট্রিম অরিয়েন্টেড বা স্ট্যান্ডার্ড ডাটা ফাইল (Stream oriented or standard data file)
- সিস্টেম অরিয়েন্টেড বা লো লেভেল ডাটা ফাইল (System oriented or low level data file)

স্ট্রিম অরিয়েন্টেড বা স্ট্যান্ডার্ড ডাটা ফাইল (Stream oriented or standard data file) : স্ট্রিম অরিয়েন্টেড বা স্ট্যান্ডার্ড ডাটা ফাইল হচ্ছে বহুল প্রচলিত ডাটা ফাইল এবং এসব ডাটা ফাইল নিয়ে কাজ করা অনেক সহজ।

স্ট্রিম অরিয়েন্টেড ডাটা ফাইল আবার দু'প্রকার, যথা–

- টেক্সট ফাইল (Text file) ও
- আনফরম্যাটেড ডাটা ফাইল (Unformated data file)

টেক্সট ফাইলসমূহ ডাটা আইটেম, স্ট্রিং ও সংখ্যা ইত্যাদি নিয়ে গঠিত। আনফরম্যাটেড ফাইলসমূহ সাধারণত জটিল ধরনের ডাটা স্ট্রাকচার। যেমন– অ্যারে (Array) ও স্ট্রাকচার (Structure)-কে রিপ্রেজেন্ট করে।

সিস্টেম অরিয়েন্টেড বা লো লেভেল ডাটা ফাইল (System oriented or low-level data file) : সিস্টেম অরিয়েন্টেড ডাটা ফাইলসমূহ স্ট্রিম অরিয়েন্টেড ডাটা ফাইলের তুলনায় একটু কঠিন এবং কম্পিউটার অপারেটিং সিস্টেমের সাথে অধিকতর সম্পর্কিত (related)।

ফাইল অপারেশন (File operation) : কোনো ফাইলে ইনপুট-আউটপুট কার্যাদি (ডাটা পড়া, লিখা, পরিবর্তন করা, পরিমার্জন করা ইত্যাদি) সম্পাদনের জন্য যেসব অপারেশন সম্পন্ন করা হয় তাদেরকে ফাইল অপারেশন (File operation) বলে। যেমন–

- ফাইলের নামকরণ করা
- ফাইল খোলা
- ফাইল থেকে ডাটা পড়া
- ফাইলে ডাটা লেখা
- ফাইল বন্ধ করা ইত্যাদি।

ফাইল অপারেশনে ব্যবহৃত ফাংশনসমূহ (Functions used in file operation) : ফাইল অপারেশনে (ফাইল খোলা, বন্ধ করা, ফাইলে ডাটা লেখা, ফাইল থেকে ডাটা পড়া) বিভিন্ন ধরনের লাইব্রেরি ফাংশন ব্যবহার করা হয়। নিম্নে এসব লাইব্রেরি ফাংশন ও তাদের কাজ উল্লেখ করা হলো :

পাইথন ফাইল অপারেশনে ব্যবহৃত মেথডসমূহ

ফাংশনের নাম	কাজ
close()	ওপেন করা ফাইল বন্ধ করে।
detach()	TextIOBase হতে বাইনারি বাফারকে পৃথক করার জন্য এই মেথডটি ব্যবহার করা হয়।
fileno()	ফাইল ডেসক্রিপ্টর এর মান রিটার্ন
flush()	ফাইলের ইন্টারনাল বাফার ফ্লাশ করার জন্য এই মেথডটি ব্যবহার হয়।
isatty()	যদি ফাইলটি ইন্টার‍্যাক্টিভ হয় তাহলে True ভ্যালু রিটার্ন করে।
read(*n*)	থেকে সর্বশেষ ক্যারেক্টার পর্যন্ত পড়ার জন্য এই মেথডটি ব্যবহার হয়।

পাইথন ফাইল অপারেশনে ব্যবহৃত মেথডসমূহ	
ফাংশনের নাম	কাজ
readable()	যদি ফাইলটি রিডেবল হয় তাহলে True ভ্যালু রিটার্ন করে।
readline(*n*=-1)	ফাইলকে রিড করে এবং ফাইল হতে একটি লাইন রিটার্ন করে।
readlines(*n*=-1)	ফাইলকে রিড করে এবং ফাইল হতে লাইনের একটি লিস্ট রিটার্ন করে।
seek(*offset,from*=SEEK_SET)	ফাইল পয়েন্টারকে ফাইলের নির্দিষ্ট স্থানে নিয়ে যায়।
seekable()	যদি ফাইলটি র‍্যান্ডম অ্যাক্সেস সাপোর্ট করে তবে True ভ্যালু রিটার্ন করে।
tell()	ফাইলের বর্তমান অবস্থান (location) নির্দেশ করে।
truncate(*size*=None)	ফাইল সাইজকে ট্রাংকেইট করে অর্থাৎ ফাইল সাইজ স্পেসিফাই করা না থাকলে এই মেথডের মাধ্যমে ফাইল সাইজ পুনরায় স্পেসিফাই করা হয়।
writable()	যদি ফাইলটি রাইটেবল হয় তাহলে True ভ্যালু রিটার্ন করে।
write(*s*)	ফাইলে কোনো কিছু (string) লেখার জন্য এবং লিখিত ক্যারেক্টারসমূহ রিটার্ন করার জন্য মেথডটি ব্যবহার করা হয়।
writelines(*lines*)	ফাইলে এক সেট লাইন রাইট করে।

১১.৪ ফাইল খোলা ও ফাইল বন্ধ করা (Opening & closing files) ঃ

ফাইল খোলা (Open a file) ঃ সংরক্ষিত ডিস্কের ফাইলে কিছু লিখতে বা ফাইল থেকে কিছু পড়তে হলে তার পূর্বে ফাইল Open করতে হয়। ফাইল খোলার জন্য পাইথনে open() ফাংশন ব্যবহার করা হয়। প্রোগ্রামের open () ফাংশন অপারেটিং সিস্টেমকে ফাইলের নাম এবং ফাইল খোলার উদ্দেশ্য (পড়া/লেখা) জানিয়ে দেয়। ফাংশন অপারেটিং সিস্টেম যদি ফাইল খুলতে পারে তবে ঐ ফাইলের প্রথম Byte এর Address-কে Structure File এর Pointer আকারে পাঠিয়ে দেয়।

open () ফাংশন ঃ পাইথনে ফাইল ওপেন করার জন্য open() ফাংশন ব্যবহার করা হয়। open () ফাংশনের গঠন বা সিনট্যাক্স বা ফরম্যাট নিম্নরূপ ঃ

```
file object = open(file_name [, access_mode][, buffering])
```

open () ফাংশনে তিনটি প্যারামিটার আছে। যথা–

- ফাইলের নাম
- অ্যাক্সেস মোড এবং
- বাফারিং

ফাইলের নাম ঃ প্রথম প্যারামিটার হলো ফাইলের নাম। যদি পাইথন স্ক্রিপ্ট ও ফাইলটি যদি কম্পিউটারের একই ডিরেক্টরিতে হয়, তবে শুধু ফাইলের নামটা স্ট্রিং হিসাবে দিতে হয়। যেমন–

```
file object = open ("file_name.txt")
```

আর ফাইলটি যদি আলাদা ডিরেক্টরিতে থাকে তবে পুরো পাথটি স্ট্রিং হিসাবে দিতে হবে। যেমন–

```
file object = open ("C:\Users\Mahbub CPI\Desktop\66631\Python Link..txt")
```

বাফারিং ঃ পাইথনে open () ফাংশনের অন্যতম একটি প্যারামিটার হলো বাফারিং। ফাইল অ্যাক্সেসিং এর সময় যদি বাফারিং ভ্যালু 0 হয় তবে পাইথনে কোনো বাফারিং সংঘটিত হয় না। তবে যদি বাফারিং ভ্যালু 1 হয় তবে পাইথনে বাফারিং সংঘটিত হয়। আবার বাফারিং ভ্যালু যদি 1 এর চেয়ে বেশি হয় তবে নির্ধারিত বাফার সাইজের মধ্যে বাফারিং সংঘটিত হয়। আর যদি বাফারিং ভ্যালু নেগেটিভ হয় তাহলে বাফার সাইজ হবে ডিফল্ট সাইজ।

অ্যাক্সেস মোড ঃ দ্বিতীয় প্যারামিটার হলো অ্যাক্সেস মোড। পাইথন প্রোগ্রামে ফাইল নতুন করে তৈরি করা হবে, না পূর্বে তৈরি করা কোনো ফাইল পরিবর্তন বা পরিবর্ধনের জন্য খোলা হবে, না শুধু পড়ার জন্য ফাইল খোলা হবে তা ফাইল মোডের উপর নির্ভর করে। নিম্নের টেবিলে open () ফাংশনে ব্যবহৃত বিভিন্ন মোড এবং তাদের ব্যবহার উল্লেখ করা হয়েছে। পাইথন প্রোগ্রামে ব্যবহৃত ফাইলে উল্লেখিত যে-কোনো একটি মোড ব্যবহার করা যায়। তবে একসাথে একটির বেশি মোড ব্যবহার করা যায় না।

ফাইল মোড	মোডের নাম	ব্যবহার
"r"	রিড মোড	এই মোডের কাজ হল পূর্বে তৈরিকৃত কোন ফাইলকে শুধুমাত্র রিড করার জন্য ওপেন করা। এটি ডিফল্ট মোড এবং এতে ফাইল পয়েন্টার ফাইলের শুরুতেই অবস্থান করে।
"r+"	রিড বিঝাস মোড	এই মোডের মাধ্যমে পূর্বে তৈরিকৃত কোন ফাইলকে রিডিং ও রাইটিং উভয় কাজের জন্য ওপেন করা হয়। এতে ও ফাইল পয়েন্টার ফাইলের শুরুতেই অবস্থান করে।
"rb"	বাইনারী রিড মোড	বাইনারী ফাইলকে রিড মোডে ওপেন করার জন্য এই মোড ব্যবহার করা হয়। এটি ডিফল্ট মোড এবং এতে ফাইল পয়েন্টার ফাইলের শুরুতেই অবস্থান করে।
"rb+"	বাইনারী রিড বিঝাস মোড	বাইনারী ফাইলকে রিড ও রাইট উভয় মোডে ওপেন করার জন্য এই মোড ব্যবহার করা হয়। এতে ও ফাইল পয়েন্টার ফাইলের শুরুতেই অবস্থান করে।
"w"	রাইট মোড	শুধুমাত্র কোন ফাইলে রাইট করার জন্য এই মোডে ফাইল ওপেন করা হয়। এই নামে পুরাতন কোন ফাইল থাকলে তা ওভাররাইট হয়। আর না থাকলে নতুন ফাইল তৈরি হয়।
"w+"	রাইট প্লাস মোড	কোন ফাইলে রিডিং ও রাইটিং উভয় কাজের জন্য এই মোডে ফাইল ওপেন করা হয়। এই নামে পুরাতন কোন ফাইল থাকলে তা ওভাররাইট হয়। আর না থাকলে নতুন ফাইল তৈরি হয়।
"wb"	বাইনারী রাইট মোড	কোন বাইনারী ফাইলকে রাইট মোডে খোলার জন্য এই মোড ব্যবহার করা হয়। এই নামে পুরাতন কোন ফাইল থাকলে তা ওভাররাইট হয়। আর না থাকলে নতুন ফাইল তৈরি হয়।
"wb+"	বাইনারী রাইট প্লাস মোড	কোন বাইনানরী ফাইলে রিডিং ও রাইটিং উভয় কাজের জন্য এই মোডে ফাইল ওপেন করা হয়। এই নামে পুরাতন কোন ফাইল থাকলে তা ওভাররাইট হয়। আর না থাকলে নতুন ফাইল তৈরি হয়।
"a"	অ্যাপেন্ড মোড	অ্যাপেন্ডিং এর জন্য ফাইলকে এই মোডে ওপেন করা হয়। এই নামে পুরাতন কোন ফাইল থাকলে এই মোডে ফাইল পয়েন্টার ফাইলের একেবারে শেষে থাকে। আর না থাকলে রাইটিং এর জন্য নতুন ফাইল তৈরি হয়।
"a+"	অ্যাপেন্ড প্লাস মোড	অ্যাপেন্ডিং ও রিডিং এর জন্য ফাইলকে এই মোডে ওপেন করা হয়। এই নামে পুরাতন কোন ফাইল থাকলে এই মোডে ফাইল পয়েন্টার ফাইলের একেবারে শেষে থাকে। আর না থাকলে রিডিং ও রাইটিং উভয় কাজের জন্য নতুন ফাইল তৈরি হয়।
"ab"	বাইনারী অ্যাপেন্ড মোড	অ্যাপেন্ডিং এর জন্য বাইনারী ফাইলকে এই মোডে ওপেন করা হয়। এই নামে পুরাতন কোন ফাইল থাকলে এই মোডে ফাইল পয়েন্টার ফাইলের একেবারে শেষে থাকে। আর না থাকলে রাইটিং এর জন্য নতুন ফাইল তৈরি হয়।
"ab+"	বাইনারী অ্যাপেন্ড প্লাস মোড	অ্যাপেন্ডিং ও রিডিং এর জন্য বাইনারী ফাইলকে এই মোডে ওপেন করা হয়। এই নামে পুরাতন কোন ফাইল থাকলে এই মোডে ফাইল পয়েন্টার ফাইলের একেবারে শেষে থাকে। আর না থাকলে রিডিং ও রাইটিং উভয় কাজের জন্য নতুন ফাইল তৈরি হয়।

উদাহরণঃ

```
f = open('file.txt', 'w')
f.write('Welcome to Python Programming Language')
f.close()
```

উপরোক্ত প্রোগ্রামে 'file.txt' ফাইলটি ডিফল্ট ভাবে রিডিং ওপেন হয়েছে। কারণঃ ওপেন মোড ব্যবহার করলে ফাইল ডিফল্ট ভাবে রিডিং মোডে ওপেন হয়। আবার উক্ত ফাইলে রাইটিং মোড ওপেন করার জন্য 'w' ব্যবহার করা হয়েছে। পরবর্তীতে write() ফাংশন ব্যবহার করে 'Welcome to Python Programming Language' স্ট্রিংটি ফাইলে রাইট করা হয়েছে।

এবার নিচের প্রোগ্রামটি লিখে রান করলে ফাইলে পূর্বে রাইটকৃত 'Welcome to Python Programming Language' স্ট্রিংটি রিড মোডের কারণে আউটপুটে প্রদর্শিত হবে।

```
f = open('file.txt', 'r')
x= f.read()
print(x)
f.close()
```

আউটপুট ঃ

```
Welcome to Python Programming Language
```

ফাইল বন্ধ করা (Closing a file) ঃ ফাইল open করে ফাইলে read/write করার পর ফাইল অবশ্যই close (বন্ধ) করতে হবে। নাহলে অকারণেই পাইথনের কাছে ফাইলটি ওপেন অবস্থাতে থাকবে, যা বস্তুত মেমরি দখল করে রাখবে এবং প্রোগ্রামের পারফরমেন্সে খারাপ ভূমিকা রাখবে। ফাইল close করার জন্য পাইথনে close () ফাংশন ব্যবহৃত হয়। close () ফাংশনের ফরম্যাট হলো ঃ

FileName .close ()

close () ফাংশন ব্যবহার করার বেশ কয়েকটি সুবিধা রয়েছে, যথা–

- File-এ কিছু লিখলে তা মূলত disk -এ লেখা হয়। তবে ফাইলে কিছু লিখলেই তা সাথে সাথে disk-এ লেখা হয় না, বরং তা buffer- এ সংরক্ষিত থাকে এবং buffer পূর্ণ হলে বা close () ফাংশন ব্যবহার করলে ঐ data disk এ লেখা হয়, ফলে ফাইলের যথাযথ ব্যবহার নিশ্চিত হয়।
- close () ব্যবহার করলে প্রোগ্রামের readability বৃদ্ধি পায়।

উদাহরণ ঃ

```
my_file = open('test.txt', 'w')
my_file.write('Welcome to Python Programming Language')
my_file.close()
```

১১.৫ ফাইল থেকে পড়া ও ফাইলে লেখা (Reading & writing files) ঃ

ফাইলে লেখা (Writing to a file) ঃ কোনো ফাইলে লেখার জন্য ফাইলটি প্রথমে অবশ্যই ওপেন করতে হবে। ওপেন করার জন্য রিড মোড ('r') বা রাইট মোড ('w') , বা অ্যাপেন্ড মোড ('a'), কিংবা এক্সক্লুসিভ ক্রিয়েশন মোড ('x') ব্যবহার করা হয়। তবে রাইট ('w') মোডে খোলার সময় অবশ্যই সতর্ক থাকতে হবে। কেননা রাইট মোডে ফাইলটি ওপেন করলে ফাইলে বিদ্যমান সকল ডাটা মুছে যাবে। পাইথনে কোনো ফাইলে কোনো কিছু লেখার জন্য **write ()** ফাংশন ব্যবহার করা হয়। নিম্নে **write ()** ফাংশন এর সিনট্যাক্স উল্লেখ করা হলো ঃ

FileName .write (str)

এখানে 'str' হলো সেই স্ট্রিং ভ্যালু, যা মান আমরা ফাইলে রাইট করতে চাই। যেমন–

File.write("Welcome to the Python Language, By-Mohammed Mahbub Alam")

উদাহরণ-১ ঃ

```
File = open("Test.txt", "w")
File.write("Welcome to the Python Language, By-Mohammed Mahbub Alam")
File.close()
File = open("Test.txt", "r")
print(File.read())
File.close()
```

আউটপুট ঃ

```
Welcome to the Python Language, By-Mohammed Mahbub Alam
```

উপরোক্ত প্রোগ্রামের দুটি অংশ। প্রথম অংশে ফাইলকে ওপেন করে সেখানে একটি লাইন লেখা হয়েছে। আমাদের চলতি উদাহরণ মোতাবেক এই নামের ফাইলটি আগে থেকেই ছিল। কিন্তু 'w' মোডে খোলার কারণে এবং এখানে নতুন করে লেখার কারণে ওই ফাইলের আগের সব কন্টেন্ট মুছে যাবে এবং নতুন write করা কন্টেন্ট লেখা হবে। যদি ওই নামের ফাইল না থাকতো, তাহলে পাইথন নতুন করে ওই নামে একটি ফাইল তৈরি করে সেখানে লিখতো। লেখা শেষে ফাইলটিকে ক্লোজ করা হয়েছে।

দ্বিতীয় অংশে আবার সেই ফাইলকে পড়ার জন্য 'r' মোডে খোলা হয়েছে এবং সব কন্টেন্ট রিড করার পর স্ক্রিনে প্রিন্ট করা হয়েছে।

উদাহরণ-২ ঃ

```
with open('app.log', 'w', encoding = 'utf-8') as f:
    #first line
    f.write('Welcome to Python Programming\n')
    #second line
    f.write('This Book is written by Mohammed Mahbub Alam\n')
    #third line
    f.write('This book is published from Haque Publications\n')
with open('app.log', 'r', encoding = 'utf-8') as f:
    content = f.readlines()
for line in content:
    print(line)
```

আউটপুট ঃ

```
Welcome to Python Programming
This Book is written by Mohammed Mahbub Alam
This book is published from Haque Publications
```

ফাইল থেকে পড়া (Reading from a file) ঃ

ফাইল থেকে কোনো স্ট্রিং রিড করার জন্য সর্বপ্রথম ফাইলটি ওপেন করতে হবে। তারপর ফাইল থেকে কোনো স্ট্রিং কিংবা বাইনারি ডাটা রিড করার জন্য read() ফাংশন ব্যবহার করা হয়। read() ফাংশনের সিনট্যাক্স বা গঠন বা ফরম্যাট হলো নিম্নরূপ ঃ

```
FileName. read ([count])
```

এখানে count হলো স্ট্রিং-এর ক্যারেক্টার সংখ্যা। অর্থাৎ count-এর যে মান দেয়া হবে সে পর্যন্ত রিড করবে। যেমন–

```
fo = open("foo.txt", "r+")
str = fo.read(10);
print "Read String is : ", str
# Close opend file
fo.close()
```

পাইথনে কোনো ফাইল ওপেন করে তাকে বিভিন্নভাবে রিড করা যায়, যেমন–

ফাইলের সকল কন্টেন্ট পড়ার জন্য ঃ ফাইলের সকল কন্টেন্ট পড়ার জন্য কোনো আরগুমেন্ট ছাড়া read() ফাংশন ব্যবহার করা হয়।

উদাহরণ ঃ

```
File = open("Test.txt", "r")
content = File.read()
print(content)
File.close()
```

আউটপুট ঃ

```
Welcome to the Python Language, By-Mohammed Mahbub Alam
```

উপরোক্ত উদাহরণে ওপেন ফাংশন ব্যবহার করে এবং ফাইলের পাথ ডিফাইন করে দিয়ে File নামের একটি ফাইল ওপেন করেছি। এরপর এই অবেজক্টের মেথড Read ব্যবহার করে পুরো ফাইলে থাকা কন্টেন্ট পড়ে Content ভেরিয়েবলে জমা করেছি। অতঃপর, একটি প্রিন্ট স্টেটমেন্ট ব্যবহার করে সেই কন্টেন্ট স্ক্রিনে প্রিন্ট করেছি। আর কাজ শেষে ফাইল অবজেক্ট-এর Close মেথড ব্যবহার করে ফাইলকে ক্লোজ করেছি।

ফাইলের কন্টেন্ট বাইট আকারে পড়ার জন্য ঃ ফাইলের কন্টেন্ট বাইট আকারে পড়ার জন্য আরগুমেন্ট হিসেবে বাইট নির্ধারণ করে দেয়া যায়।

উদাহরণ ঃ

```
File = open("Test.txt", "r")
just_one_character = File.read(1)
print(just_one_character)
remaining_four_characters = File.read(4)
print(remaining_four_characters)
rest_of_the_file = File.read()
print(rest_of_the_file)
File.close()
```

আউটপুট ঃ

```
W
elco
me to the Python Language, By-Mohammed Mahbub Alam
```

উপরোক্ত প্রোগ্রামে তিন বার ফাইল থেকে কন্টেন্ট পড়া হয়েছে, কিন্তু তিনভাবে। প্রথমবার মাত্র একটি বাইট পড়া হয়েছে। এক বাইট মানে একটি ক্যারেক্টার। তাই সেটি প্রিন্ট করেছে শুধু W. এর পরে আবার পড়া হয়েছে ৪টি বাইট। তাই Elco এই চার ক্যারেক্টার পড়া হয়েছে। যেহেতু আমরা একই ফাইল অবজেক্ট (File) নিয়ে দ্বিতীয় বারও কাজ করেছি তাই এবার যে ৪বাইট পড়তে চেয়েছি সেটা আসলে H এর পর থেকে ৪বাইট। তৃতীয় বার কোনো আর্গুমেন্ট ছাড়া Read মেথড ব্যবহার করা হয়েছে এবং ফাইলের বাকি সব কন্টেন্ট পড়ে প্রিন্ট করা হয়েছে। এবারও যেহেতু একই ফাইল অবজেক্ট-এর উপরেই কাজ করা হয়েছে তাই rest_of_the_file ভ্যারিয়েবলে কিন্তু H, Elco-এর পর থেকে অর্থাৎ Me ... থেকে শেষ পর্যন্ত সব কন্টেন্ট জমা হয়েছে।

ফাইলের কন্টেন্ট লাইন বাই লাইন আকারে পড়ার জন্য ঃ ফাইলের কন্টেন্ট লাইন বাই লাইন আকারে পড়ার জন্য readline() ফাংশন ব্যবহার করা হয়।

উদাহরণ ঃ

```
with open('app.log', 'w', encoding = 'utf-8') as f:
    #first line
    f.write('Welcome to Python Programming\n')
    #second line
    f.write('This Book is written by Mohammed Mahbub Alam\n')
    #third line
    f.write('This book is published from Haque Publications\n')

with open('app.log', 'r', encoding = 'utf-8') as f:
    content = f.readline()
print(content)
```

আউটপুট ঃ

```
Welcome to Python Programming
```

ফাইলের কন্টেন্ট লাইন বাই লাইন আকারে পড়ার জন্য readline() ফাংশন ছাড়া ফর লুপ ও ব্যবহার করা যায়।

উদাহরণ ঃ

```
file_to_work = open("Test.txt", "r")
for my_line in file_to_work:
    print(my_line)
file_to_work.close()
```

আউটপুট ঃ

```
Welcome to the Python Language, By-Mohammed Mahbub Alam
```

১১.৬ এরর হ্যান্ডলিং (Error handling in python) ঃ

ফাইল অপারেশনের সময় পাইথন প্রোগ্রামিং ল্যাংগুয়েজে লিখিত প্রোগ্রামে নানা সময়ে নানা কারণে নানান ধরনের error message বা error দেখায়। বিভিন্ন কারণে এই এররগুলো সংঘটিত হতে পারে। যেমন–

- ফাইলের 'end of file' সঠিকভাবে detect না করতে পারা।
- ওপেন করা হয় নেই এমন কোনো ফাইল ব্যবহারের চেষ্টা করা।
- অন্য কোনো Operation এর জন্য Open কৃত File এ Operation চালানোর চেষ্টা করা।
- Write protected file-এ কোনো কিছু Write করার চেষ্টা করা।
- Invalid file name যুক্ত File open করার চেষ্টা করা।
- ফাইলে ডাটা সংরক্ষণের জন্য পর্যাপ্ত জায়গা বরাদ্দ না থাকা।
- ফাইলের এক্সটেনশন নেইম ব্যবহারে ভুল করা।
- রিড (r) মোডের ফাইল রাইট (w) মোডে রিড করার চেষ্টা করা।
- ফাইলে কোনো অবৈধ অপারেশনের চেষ্টা করা।

নিম্নে পাইথনে সংঘটিত এররসমূহের একটি তালিকা উল্লেখ করা হলো ঃ

পাইথনে সংঘটিত এরর বা ত্রুটিসমূহ	
এরর	বর্ণনা
ArithmeticError	বিভিন্ন ধরনের গাণিতিক সমস্যার কারণে এ ধরনের ত্রুটি সংঘটিত হয়।
RuntimeError	নির্দিষ্ট কোন এক্সেপশন ক্যাটাগরির মধ্যে না পড়লে এ ধরনের ত্রুটি সংঘটিত হয়।
BufferError	বাফার রিলেটেড অপারেশন সম্পন্ন না হলে এ ধরনের ত্রুটি সংঘটিত হয়।
FileNotFoundError	নির্দিষ্ট ফাইলটি খুঁজে না পেলে এ ধরনের ত্রুটি সংঘটিত হয়।
ImportError	import স্টেটমেন্ট import করতে ব্যর্থ হলে এ ধরনের এরর সংঘটিত হয়।
IndentationError	প্রোগ্রাম কোড ব্লকের ইনডেন্টেশন সঠিক জায়গায় ব্যবহার না করলে এ ধরনের এরর সংঘটিত হয়।
IndexError	নির্দিষ্ট সিকুয়েন্সের চাহিদা মোতাবেক নির্দিষ্ট ইনডেক্স খুঁজে না পেলে এ ধরনের এরর সংঘটিত হয়।
KeyboardInterrupt	Ctrl + C চেপে কোন প্রোগ্রামের এক্সিকিউশন থামিয়ে দেয়া হলে এ ধরনের এরর সংঘটিত হয়।
SystemError	পাইথন প্রোগ্রামের মধ্যে ইন্টারনাল এরর থাকলে এ ধরনের এরর সংঘটিত হয়।
SyntaxError	পাইথন প্রোগ্রামে ভুল কোন কী-ওয়ার্ড বা স্টেটমেন্ট থাকলে এ ধরনের এরর সংঘটিত হয়।
KeyError	ডিকশনারিতে নির্দিষ্ট কোন কী খুঁজে পাওয়া না গেলে এ ধরণের এরর সংঘটিত হয়।
NameError	প্রোগ্রামে নির্দিষ্ট নামের কোন লোকাল বা গ্লোবাল ভেরিয়েবল খুঁজে পাওয়া না গেলে এ ধরণের এরর সংঘটিত হয়।
OSError	অপারেটিং সিস্টেম রিলেটেড কোন এরর রিটার্ন করলে এ ধরণের এরর সংঘটিত হয়।
AssertionError	assert স্টেটমেন্ট ব্যর্থ হলে এ ধরণের ত্রুটি সংঘটিত হয়।
StopIteration	next() মেথডটি কোন অবজেক্টকে পয়েন্ট করতে না পারলে এ ধরণের এরর সংঘটিত হয়।
TabError	প্রোগ্রাম কোডিং এর সময় একই সাথে ট্যাব আর স্পেস দিয়ে ইন্ডেন্টেশন করা হলে এ ধরণের এরর সংঘটিত হয়।
TypeError	কোন ফাংশন বা অপারেশনে ডাটা টাইপ জনিত সমস্যা থাকলে এ ধরণের এরর সংঘটিত হয়।
UnboundLocalError	পাইথন প্রোগ্রামের কোন ফাংশনে কোন ভ্যালু ছাড়াই লোকাল ভেরিয়েবল ব্যবহার করা হলে এ ধরণের এরর সংঘটিত হয়।
ValueError	পাইথন প্রোগ্রামের বিল্ট-ইন ফাংশন যদি এমন কোন আর্গুমেন্ট গ্রহণ করে যার ভ্যালুতে সমস্যা রয়েছে এবং এই সমস্যার সমাধান অসম্ভব তখন এ ধরণের এরর সংঘটিত হয়।
WindowsError	উইন্ডোজ অপারেটিং সিস্টেম জনিত সমস্যার কারণে এ ধরণের এরর সংঘটিত হয়।
ZeroDivisionError	কোন ভ্যালুকে শুন্য দিয়ে ভাগ করার চেষ্টা করা হলে এ ধরণের ত্রুটি সংঘটিত হয়।
AttributeError	কোন অ্যাট্রিবিউট রেফারেন্স বা অ্যাসাইনমেন্ট ফেইল করলে এ ধরণের ত্রুটি সংঘটিত হয়।
OverflowError	কোন নিউমেরিক টাইপের ডাটা ম্যাক্সিমাম লিমিট করলে এ ধরণের ত্রুটি সংঘটিত হয়।

এরর হ্যান্ডলিং (Error handling in python) ঃ পাইথনে সংঘটিত উপরোক্ত ভুলসমূহ যাতে প্রোগ্রামে রেইজ করতে না পারে তার জন্য ব্যবস্থা গ্রহণ করার প্রক্রিয়াকে এরর হ্যান্ডলিং বলে। পাইথনে এরর হ্যান্ডলিং-এর জন্য তিনটি টেকনিক ব্যবহার করা হয়। যথা—

- try ... except টেকনিক
- try ... except ... else টেকনিক এবং
- try ... except ... finally টেকনিক

নিম্নে এ সকল টেকনিক ব্যবহার করে এরর হ্যান্ডলিং প্রক্রিয়া ব্যাখ্যা করা হলো ঃ

উদাহরণ ঃ

```
try:
    f = open('test.txt')
    s = f. readline ( )
    i = int(s.strip())
except IOError as e:
    print("I/O error({0}): {1}".format(e.errno,e.strerror))
except ValueError:
    print("No valid integer in line.")
except:
    print("Unexpected error!", sys.exc_info () [0])
```

আউটপুট ঃ

```
No valid integer in line.
```

উপরোক্ত উদাহরণে দেখা যায় যে, একটি try ব্লকের জন্য যত খুশি তত except ব্লক লেখা যায়। তবে প্রতিটি except ব্লকে এক্সেপশনের নাম উল্লেখ করতে হবে। একটা নির্দিষ্ট এক্সেপশন রেইজ হলেই কেবল ঐ নির্দিষ্ট except ব্লক এক্সিকিউট হবে। এখানে আমরা একটি try ব্লকের জন্য তিনটি except ব্লক লিখেছি। প্রথমটি IOError এর জন্য, দ্বিতীয়টি ValueError-এর জন্য আর শেষেরটি ঐ দুইটা বাদে যে-কোনো এররের জন্য। এ ধরনের except-কে পাইথনে Bare Except বলা হয়। পারতপক্ষে বেয়ার এক্সেপ্ট এভয়েড করা উচিত। কারণ বেয়ার এক্সেপ্ট সব ধরনের এররকে হাইড করে দেয় ফলে আমরা জানতেই পারব না ঠিক কোনো এক্সেপশনটাকে আমরা ক্যাচ করছি। তাই সবসময় এক্সেপশনের নাম উল্লেখ করে এরর হ্যান্ডেল করা উচিত।

try ... except ... else টেকনিক ঃ try ... except ... else টেকনিকের ক্ষেত্রে else ব্লক except ব্লকের শেষে বসে। try ব্লকে কোনো এক্সেপশন রেইজ না হলেই কেবল else ব্লকের কোড এক্সিকিউট হবে।

উদাহরণ ঃ

```
try:
    a = 15
    b = 3
    print(a * b)
except ValueError as e:
    print(e)
else:
    print('There is no exception.')
```

আউটপুট ঃ

```
45
There is no exception.
```

try ... except টেকনিকঃ try ... except টেকনিকের ক্ষেত্রে নরমাল কোডগুলো try বক্সকের ভিতর থাকবে। এক্ষেত্রে কোন এক্সেপশন সংঘটিত হলে except বক্সকের কোড এক্সিকিউট হবে। এক্সেপশন হ্যান্ডল করার জন্য except এর পরে স্পেস দিয়ে এক্সেপশনের নাম উলেস্নখ করতে হয়।

উদাহরণ ঃ

```
try:
    with open('file.txt', 'r') as f:
        x = f.read()
        print(x)
except FileNotFoundError:
    print('Python Programming Language')
    print('Written by Mohammed Mahbub Alam')
try:
    list=[ ]
    printlist[0]
except IndexError as y:
        print(y)
finally:
    print('Thanks a Lot')
```

আউটপুট ঃ

```
Welcome to Python Programming Language
Thanks a Lot
Traceback (most recent call last):
  File "C:/Users/Mahbub CPI/AppData/Local/Programs/Python/Python36-32/UcPWI.py", line 10, in <module>
    printlist[0]
NameError: name 'printlist' is not defined
```

try ... except ... finally টেকনিক ঃ try ... except ... finally টেকনিকের ক্ষেত্রে finally বয়ক একেবারে শেষে বসে। আর কোন এক্সেপশন সংঘটিত হোক বা না হোক finally ব্লকের কোড ঠিকই এক্সিকিউট হবে।

উদাহরণঃ

```
try:
    with open('file.txt', 'r') as f:
        x = f.read()
        print(x)
except FileNotFoundError:
    print('The file does not exist.')
finally:
    print('Thanks a Lot')
```

আউটপুটঃ

```
Welcome to Python Programming Language
Thanks a Lot
```

অনুশীলনী-১১

অতি সংক্ষিপ্ত প্রশ্নোত্তর ঃ

১। **ফাইল অপারেশন বলতে কী বুঝায়?**

উত্তর ঃ ফাইল অপারেশন বলতে কোনো ফাইলে ইনপুট-আউটপুট কার্যাদি (ডাটা পড়া, লেখা, পরিবর্তন করা, পরিমার্জন করা ইত্যাদি) সম্পাদনকে বুঝায়।

২। **পাইথনের বেসিক ইনপুট আউটপুট ফাংশনগুলো কী কী?**

উত্তর ঃ পাইথনের বেসিক ইনপুট আউটপুট ফাংশনগুলো হলো ঃ

- input() ফাংশন ও
- print() ফাংশন।

৩। **print() ফাংশন এর ব্যবহার লেখ।**

উত্তর ঃ স্ক্রিনের মধ্যে ডাটা আউটপুটের জন্য পাইথনে print() ফাংশন ব্যবহার করা হয়।

৪। **print() ফাংশন এর সিনট্যাক্স লেখ।**

উত্তর ঃ **print() ফাংশন এর সিনট্যাক্স (Syntax) ঃ** print() ফাংশনের সিনট্যাক্স বা গঠন বা ফরম্যাট হলো নিম্নরূপ ঃ
print(*objects, sep=' ', end='\n', file=sys.stdout, flush=False)
যেখানে–

- 'objects' হলো সেই ভ্যালু, যাকে প্রিন্ট করতে হবে।
- 'sep' হলো একাধিক ভ্যালুর মধ্যকার সেপারেটর। সেপারেটর হিসেবে বাই ডিফল্ট স্পেস ক্যারেক্টার ব্যবহৃত হয়।
- সকল ভ্যালু প্রিন্ট হওয়ার পর 'end' প্রিন্ট হয় এবং কার্সরকে নতুন লাইনে নিয়ে যায়।
- 'file' হলো সেই ভ্যালু হলো যেখানে ভ্যালুসমূহ প্রিন্ট হবে। ফাইলের ডিফল্ট ভ্যালু হলো 'sys.stdout' বা স্ক্রিন।

৫। **আউটপুট ফরম্যাটিং বলতে কী বুঝায়?**

উত্তর ঃ কোনো পার্টিকুলার ফরম্যাটে আউটপুট প্রদর্শনের জন্য কিংবা আউপুটকে অধিকতর দৃষ্টিনন্দন করার প্রক্রিয়াকে আউটপুট ফরম্যাটিং বলে।

৬। **ফরম্যাটেড আউটপুট ফাংশন কাকে বলে?**

উত্তর ঃ যে-সমস্ত স্টেটমেন্ট বা ফাংশন ব্যবহার করে আউটপুট ফরম্যাটিং করা হয়, তাদেরকে ফরম্যাটেড আউটপুট স্টেটমেন্ট বা ফরম্যাটেড আউটপুট ফাংশন বলে।

৭। **পাইথনে আউটপুট ফরম্যাটিং এর জন্য কোনো ফাংশন ব্যবহার করা হয়?**

উত্তর ঃ পাইথনে আউটপুট ফরম্যাটিং এর জন্য str.format() ব্যবহার করা হয়।

৮। **পাইথন প্রোগ্রামে কী কী উপায়ে মান ইনপুট নেয়া হয়?**

উত্তর ঃ পাইথন প্রোগ্রামিং-এর ক্ষেত্রে প্রোগ্রামে দুইভাবে মান ইনপুট নেয়া যায়, যথা–

- সরাসরি (Directly)
- ইউজার হতে (From Keyboard)।

৯। **সরাসরি মান ইনপুট নেয়ার উপায় কী?**

উত্তর ঃ সরাসরি মান ইনপুট নেয়ার ক্ষেত্রে ভেরিয়েবল ডিক্লারেশনের সময় অ্যাসাইনমেন্ট অপারেটর ব্যবহার করা হয়। যেমন–
x = 5
y = 4

১০। **কী-বোর্ড থেকে মান ইনপুটের জন্য কী কী ফাংশন ব্যবহার করা হয়?**

উত্তর ঃ পাইথনে ইউজারের মাধ্যমে কী-বোর্ড থেকে ইনপুট গ্রহণের জন্য দুই ধরনের ফাংশন ব্যবহার করা হয়, যথা–

- raw_ input() ফাংশন এবং
- input() ফাংশন।

১১। **input() ফাংশনের কাজ কী?**

উত্তর ঃ input() ফাংশনের মাধ্যমে কী-বোর্ড থেকে কোনো মান, ক্যারেক্টার, শব্দ (word) ইত্যাদি ইনপুট নিয়ে কোনো ভেরিয়েবলে সংরক্ষণ করা যায় এবং পরে এগুলোকে নিয়ে প্রসেস করে আউটপুটে প্রদর্শন করা যায়।

১২। **input() ফাংশনের সিনট্যাক্স লেখ।**

উত্তর ঃ input() ফাংশনের সিনট্যাক্স বা গঠন বা ফরম্যাট হলো নিম্নরূপ ঃ

```
input([prompt])
```

যেখানে–

'prompt' হলো সেই ভ্যালু, যার মান ইনপুট হিসেবে গ্রহণ করা হবে। এটি একটি optional অংশ।

উদাহরণ ঃ

```
>>> input("What is your Name:")
```

১৩। **ফাইল কী?**

উত্তর ঃ ফাইল বা ডাটা ফাইল হচ্ছে স্টোরেজ ডিভাইসের (Hard disk, Floppy disk, Compact disk, Digital video disk ইত্যাদি) এমন একটি স্পেস (space), যেখানে স্থায়ীভাবে তথ্য সংরক্ষণ করা যায় এবং প্রয়োজনে তথ্যসমূহ রিড (read) করা যায়।

১৪। **ফাইল কত প্রকার ও কী কী?**

উত্তর ঃ ফাইল প্রধানত দুই প্রকার, যথা–

- স্ট্রিম অরিয়েন্টেড বা স্ট্যান্ডার্ড ডাটা ফাইল (Stream oriented or Standard data file)
- সিস্টেম অরিয়েন্টেড বা লো লেভেল ডাটা ফাইল (System oriented or Low level data file)।

১৫। **ফাইল অপারেশন বলতে কী বুঝায়?**

উত্তর ঃ কোনো ফাইলে ইনপুট-আউটপুট কার্যাদি (ডাটা পড়া, লেখা, পরিবর্তন করা, পরিমার্জন করা ইত্যাদি) সম্পাদনের জন্য যে-সব অপারেশন সম্পন্ন করা হয়, তাদেরকে ফাইল অপারেশন (File operation) বলে। যেমন–

- ফাইলের নামকরণ করা
- ফাইল খোলা
- ফাইল থেকে ডাটা পড়া
- ফাইলে ডাটা লেখা
- ফাইল বন্ধ করা ইত্যাদি।

১৬। **open() ফাংশনের কাজ কী?**

উত্তর ঃ পাইথনে ফাইল ওপেন করার জন্য open() ফাংশন ব্যবহার করা হয়।

১৭। **open() ফাংশনের সিনট্যাক্স লেখ।**

উত্তর ঃ open () ফাংশনের গঠন বা সিনট্যাক্স বা ফরম্যাট নিম্নরূপ ঃ

```
file object = open(file_name [, access_mode][, buffering])
```

open () ফাংশনে তিনটি প্যারামিটার আছে, যথা–

- ফাইলের নাম
- অ্যাক্সেস মোড এবং
- বাফারিং।

১৮। **close() ফাংশনের কাজ কী?**

উত্তর ঃ ফাইল close করার জন্য পাইথনে close () ফাংশন ব্যবহৃত হয়।

১৯। **close() ফাংশনের সিনট্যাক্স লেখ।**

উত্তর ঃ close () ফাংশনের ফরম্যাট হলো ঃ

```
FileName .close ()
```

২০। **write() ফাংশনের কাজ কী?**

উত্তর ঃ পাইথনে কোনো ফাইলে কোনো কিছু লেখার জন্য write () ফাংশন ব্যবহার করা হয়।

২১। **write() ফাংশনের সিনট্যাক্স লেখ।**

উত্তর ঃ write () ফাংশন এর সিনট্যাক্স উল্লেখ করা হলো ঃ

FileName .write (str)

এখানে str হলো সেই স্ট্রিং ভ্যালু, যার মান আমরা ফাইলে রাইট করতে চাই। যেমন–

File.write("Welcome to the Python Language, By-Mohammed Mahbub Alam")

২২। **ফাইল থেকে পড়ার জন্য কোনো ফাংশন ব্যবহার করা হয়?**

উত্তর ঃ ফাইল থেকে কোনো স্ট্রিং কিংবা বাইনারি ডাটা পড়ার জন্য read() ফাংশন ব্যবহার করা হয়।

২৩। **read() ফাংশনের সিনট্যাক্স উল্লেখ কর।**

উত্তর ঃ read() ফাংশনের সিনট্যাক্স বা গঠন বা ফরম্যাট হলো নিম্নরূপ ঃ

FileName. read ([count])

এখানে count হলো স্ট্রিং এর ক্যারেক্টার সংখ্যা। অর্থাৎ count এর যে মান দেয়া হবে সে পর্যন্ত রিড করবে। যেমন–

```
fo = open("foo.txt", "r+")
str = fo.read(10);
```

২৪। **readline() ফাংশনের কাজ কী?**

উত্তর ঃ ফাইলের কন্টেন্ট লাইন বাই লাইন আকারে পড়ার জন্য readline() ফাংশন ব্যবহার করা হয়।

২৫। **Run time error কখন সংঘটিত হয়?**

উত্তর ঃ যখন কোনো এরর হ্যান্ডেল করার জন্য জেনারেল কোনো স্ট্যান্ডার্ড এক্সেপশন পাওয়া না যায়, তখন Run time error সংঘটিত হয়।

২৬। **Type error কী?**

উত্তর ঃ যখন কোনো ফাংশন বা অপারেশন কোনো অসঙ্গত টাইপের অবজেক্টে অ্যাপ্লাই করা হয় তখন যে ধরনের এরর সংঘটিত হয়, সে সকল এররকে Type error বলে।

২৭। **এরর হ্যান্ডলিং এর জন্য কয়টি ও কী কী ব্যবহার করা হয় ?**

উত্তর ঃ পাইথনে এরর হ্যান্ডলিং এর জন্য তিনটি টেকনিক ব্যবহার করা হয়, যথা–

- try ... except টেকনিক
- try ... except ... else টেকনিক এবং
- try ... except ... finally টেকনিক।

সংক্ষিপ্ত প্রশ্নোত্তর ঃ

১। **উদাহরণসহ আউটপুট ফরম্যাটিং প্রক্রিয়া বর্ণনা কর।**

উত্তর সংকেত ঃ ১১.১ নং অনুচ্ছেদ দ্রষ্টব্য।

২। **input()এবং raw_input() ফাংশনের সিনট্যাক্সসহ উদাহরণ দাও।**

উত্তর সংকেত ঃ ১১.২ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। **ফাইল অপারেশনে ব্যবহৃত যে-কোনো তিনটি ফাংশনের কাজ উল্লেখ কর।**

উত্তর সংকেত ঃ ১১.৩ নং অনুচ্ছেদ দ্রষ্টব্য।

৪। **open() ফাংশনের প্যারামিটারসমূহ বর্ণনা কর।**

উত্তর সংকেত ঃ ১১.৪ নং অনুচ্ছেদ দ্রষ্টব্য।

৫। **বিভিন্ন প্রকার ফাইল অ্যাক্সেস মোডের ব্যবহার লেখ।**

উত্তর সংকেত ঃ ১১.৪ নং অনুচ্ছেদ দ্রষ্টব্য।

৬। **close() ফাংশন ব্যবহারের সুবিধাগুলো উল্লেখ কর।**

উত্তর ঃ close () ফাংশন ব্যবহারের সুবিধা ঃ

- File-এ কিছু লেখলে তা মূলত disk-এ লেখা হয়। তবে ফাইলে কিছু লেখলেই তা সাথে সাথে disk-এ লেখা হয় না, বরং তা buffer-এ সংরক্ষিত থাকে এবং buffer পূর্ণ হলে বা close () ফাংশন ব্যবহার করলে ঐ data disk-এ লেখা হয়। ফলে ফাইলের যথাযথ ব্যবহার নিশ্চিত হয়।
- close () ব্যবহার করলে প্রোগ্রামের readability বৃদ্ধি পায়।

৭। **ফাইলে কী কী কারণে এরর সংঘটিত হতে পারে তার তালিকা উল্লেখ কর।**

উত্তর ঃ যে-সকল কারণে ফাইলে এরর সংঘটিত হতে পারে তার তালিকা নিম্নরূপ ঃ

- ফাইলের 'end of file' সঠিকভাবে detect না করতে পারা।
- ওপেন করা হয় নেই এমন কোনো ফাইল ব্যবহারের চেষ্টা করা।
- অন্য কোনো Operation এর জন্য Open-কৃত File-এ Operation চালানোর চেষ্টা করা।
- Write protected file-এ কোনো কিছু Write করার চেষ্টা করা।
- Invalid file name-যুক্ত file open করার চেষ্টা করা।
- ফাইলে ডাটা সংরক্ষণের জন্য পর্যাপ্ত জায়গা বরাদ্দ না থাকা।
- ফাইলের এক্সটেনশন নেইম ব্যবহারে ভুল করা।
- রিড (r) মোডের ফাইল রাইট (w) মোডে রিড করার চেষ্টা করা।
- ফাইলে কোনো অবৈধ অপারেশনের চেষ্টা করা।

৮। **ফাইলে সংঘটিত যে-কোনো তিনটি এরর বর্ণনা কর।**

উত্তর ঃ নিম্নে পাইথনের ফাইল অপারেশনে সংঘটিত তিনটি এরর বর্ণনা নিম্নে দেয়া হলো ঃ

পাইথনে সংঘটিত এরর বা ত্রুটিসমূহ	
এরর	**বর্ণনা**
FileNotFoundError	যখন কোনো নির্দিষ্ট ফাইলের অস্তিত্ব খুঁজে না পাওয়া যায় তখন এ ধরনের এরর সংঘটিত হয়।
ImportError	যখন import স্টেটমেন্ট ফেল করে তখন এ ধরনের এরর সংঘটিত হয়।
IndentationError	যদি কোড ব্লকের ইনডেন্টেশন ঠিকমত মেইনটেইন না করা হয় তখন এ ধরনের এরর সংঘটিত হয়।

রচনামূলক প্রশ্নোত্তর ঃ

১। **একটি ফাইলে ডাটা স্টোর করার প্রোগ্রাম তৈরি কর।** [বাকাশিবো-২০১৭]

উত্তর সংকেত ঃ ১১.৫ নং অনুচ্ছেদ দ্রষ্টব্য।

২। **উদাহরণসহ try.....except এরর হ্যান্ডলিং টেকনিক বর্ণনা কর।**

উত্তর সংকেত ঃ ১১.৬ নং অনুচ্ছেদ দ্রষ্টব্য।

৩। **উদাহরণসহ try.....except finally এরর হ্যান্ডলিং টেকনিক বর্ণনা কর।**

উত্তর সংকেত ঃ ১১.৬ নং অনুচ্ছেদ দ্রষ্টব্য।