# প্রধান সূচিপত্র


| অধ্যায় | অধ্যায়ের নাম | পৃষ্ঠা নং |
|---|---|---|
| অধ্যায়-০১ | পাইথন ফাংশনস | ০২ |
| অধ্যায়-০২ | পাইথন-এর ফাইল অপারেশন | ১৩ |
| অধ্যায়-০৩ | মডিউল,প্যাকেজ এবং অ্যাপ্লিকেশন সফটওয়্যার | ১৯ |
| অধ্যায়-০৪ | অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং- এর মৌলিক ধারণা | ২৬ |
| অধ্যায়-০৫ | অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং- এর ফোর পিলার | ৩৫ |
| অধ্যায়-০৬ | পাইথন ইটারেটর, জেনারেটর এবং ডেকোরেটর | ৪৪ |
| অধ্যায়-০৭ | পাইথন- এর এক্সসেপশন এবং এরর হ্যান্ডেলিং | ৫০ |
| অধ্যায়-০৮ | পাইথন এর লগিং | ৫৫ |
| অধ্যায়-০৯ | ইউনিট টেস্টিং | ৬১ |
| অধ্যায়-১০ | পাইথন রেগুলার এক্সপ্রেশন | ৬৬ |
| অধ্যায়-১১ | অ্যাপ্লিকেশন সফটওয়্যার | ৭৩ |

# অধ্যায় ০১ পাইথন ফাংশনস

এই অধ্যায়ের প্রধান সূচিপত্র (contants)


| বিষয় | পৃষ্ঠা নং |
|---|---|
| ❖ সূত্রাবলী ও শর্টকার্ট টেকনিক | - |
| ❖ অতি সংক্ষিপ্ত প্রশ্নাবলি | ০৩ |
| ❖ সংক্ষিপ্ত প্রশ্নাবলি | ০৪ |
| ❖ রচনামূলক প্রশ্নাবলি | ০৮ |


## বোর্ড প্রশ্নাবলির বিশ্লেষন (Board Questions Analysis)

এই অধ্যায় থেকে বিগত তিন বছরের ডিপ্লোমা ইন ইঞ্জিনিয়ারিং, ডিপ্লোমা ইন টেক্সটাইল, ডিপ্লোমা ইন এগ্রিকালচার, ডিপ্লোমা ইন মেরিন পরীক্ষায় মোট প্রশ্ন....নিচে 'Board Questions Analysis' অংশে এই অধ্যায় থেকে কোন সালে কোন বোর্ডে কতটি প্রশ্ন হয়েছে তা দেখানো হলো।

| সাল / বোর্ড | অতিসংক্ষিপ্ত | সংক্ষিপ্ত | রচনামূলক |
|---|---|---|---|
| ২০২৪ | ১টি | ২টি | ১টি |
| ২০২৩ | ২টি | ১টি | ২টি |
| ২০২২ | ২টি | ২টি | ১টি |

"সবচেয়ে শক্তিশালী বা বুদ্ধিমান নয়, পরিবর্তনের সঙ্গে যে সবচেয়ে বেশি মানিয়ে নিতে পারে, সেই টিকে থাকে"

চার্লস ডারউইন

## অতি সংক্ষিপ্ত প্রশ্ন

**১. ফাংশন বলতে কী বুঝ?** [বাকাশিবো-২০২২ (অনুষ্ঠিত-২৩), ২৩ (পরি), ২৪ (অনুষ্ঠিত-২৫)]

**উত্তরঃ** ফাংশন (Function) হলো একটি বিশেষ ধরনের নিয়ম বা প্রক্রিয়া, যেখানে একটি নির্দিষ্ট ইনপুট (Input) দিলে তার বিপরীতে একটি সুনির্দিষ্ট আউটপুট (Output) পাওয়া যায়।

**২. ফাংশন কত প্রকার ও কী কী ?**

**উত্তরঃ** ফাংশন ২ প্রকার। যথাঃ লাইব্রেরি ফাংশন, ইউজার ডিফাইন্ড ফাংশন।

**৩. User Defined Function কাকে বলে?**

**উত্তরঃ** User Defined Function (UDF) হলো এমন এক ধরনের ফাংশন যা প্রোগ্রামার বা ব্যবহারকারী নিজের প্রয়োজন অনুযায়ী নিজে তৈরি করেন।

**৪. Recursive Function কী?**

**উত্তরঃ** Recursive Function (রিকার্সিভ ফাংশন) হলো এমন একটি ফাংশন যা নিজেই নিজেকে কল (Call) করে।

**৫. পাইথনে ফাংশন তৈরির Syntex লেখ?**

**উত্তরঃ** def function_name(parameters):

```
    """Docstring: ফাংশনটি কী কাজ করে তার বর্ণনা (ঐচ্ছিক)"""

    # ফাংশনের ভেতরের কাজ (Code block)

    return result # ফলাফল ফেরত দেওয়া (ঐচ্ছিক)
```

**৬. ফাংশন রিটার্ন বলতে কী বুঝ?**

**উত্তরঃ** ফাংশন রিটার্ন (Return) হলো একটি ফাংশনের কাজের শেষ ফলাফল যা সে ব্যবহারকারী বা প্রোগ্রামের কাছে ফেরত পাঠায়।

**৭. লোকাল ভেরিয়েবল ও গ্লোবাল ভেরিয়েবল বলতে কী বুঝ?** [বাকাশিবো-২০২৩ (পরি)]

**উত্তরঃ** ১. লোকাল ভেরিয়েবল (Local Variable): যখন কোনো ভেরিয়েবল একটি ফাংশনের ভেতরে ডিক্লেয়ার বা তৈরি করা হয়, তখন তাকে লোকাল ভেরিয়েবল বলে।

২. গ্লোবাল ভেরিয়েবল (Global Variable)যখন কোনো ভেরিয়েবল সব ফাংশনের বাইরে অর্থাৎ মেইন প্রোগ্রামের শুরুতে ডিক্লেয়ার করা হয়, তখন তাকে গ্লোবাল ভেরিয়েবল বলে।

**৮. ল্যাম্বডা কী?**

**উত্তরঃ** ল্যাম্বডা (Lambda) হলো পাইথনের একটি বিশেষ ধরনের ফাংশন, যার কোনো নির্দিষ্ট নাম থাকে না। একে অনেক সময় 'Anonymous Function' বা নামহীন ফাংশনও বলা হয়।

**৯. ফাংশন কলিং কী?**

**উত্তরঃ** ফাংশন কলিং (Function Calling) হলো একটি তৈরি করে রাখা ফাংশনকে তার কাজ শুরু করার জন্য নির্দেশ দেওয়া বা তাকে "ডাকা"।

**১০. ৫ টি লাইব্রেরি ফাংশনের নাম লেখ।**

**উত্তরঃ** পাইথনে সাধারণত ব্যবহৃত ৫টি গুরুত্বপূর্ণ লাইব্রেরি ফাংশন (Built-in Functions) নিচে দেওয়া হলো।

print(). len(). input(), type(). sum()

**১১. ফাংশন Prototype – এর কাজ কী?**

**উত্তরঃ** ফাংশন প্রোটোটাইপ (Function Prototype) মূলত একটি ফাংশনের অগ্রিম ঘোষণা (Declaration)। এটি কম্পাইলারকে জানায় যে, প্রোগ্রামে ভবিষ্যতে একটি নির্দিষ্ট নামের ফাংশন ব্যবহার করা হবে, যার প্যারামিটার এবং রিটার্ন টাইপ কেমন হবে।

**১২. ফাংশন আরগুমেন্ট বলতে কী বুঝায়?** [বাকাশিবো-২০২২ (অনুষ্ঠিত-২৩)]

**উত্তর:** যখন একটি ফাংশন কল করা হয় তখন ঐ ফাংশনের প্যারামিটারের টাইপ, অর্ডার এবং নাম্বার অনুসারে ভ্যালু পাসকরা হয়। এই ভ্যালুকেই ফাংশনের আরগুমেন্ট বলা হয়।

## সংক্ষিপ্ত প্রশ্ন

**১. ফাংশন ব্যবহারের সুবিধাসমূহ লেখ।** [বাকাশিবো-২০২৩ (অনুষ্ঠিত-২৪)]

**উত্তরঃ** ফাংশন ব্যবহার করার ফলে অনেকগুলো সুবিধা পাওয়া যায়। যেমন-

- ফাংশন ব্যবহার করার ফলে প্রোগ্রামের কোনো অংশ মূল প্রোগ্রামে বার বার লেখার প্রয়োজন হয় না।
- প্রোগ্রাম লেখতে অপেক্ষাকৃত কম সময় লাগে।
- প্রোগ্রাম লেখতে অপেক্ষাকৃত কম সময় লাগে।

- প্রোগ্রামের ভুল নির্ণয় ও সংশোধন করা সহজ হয়।
- বড় প্রোগ্রাম অনেকগুলো মডিউলে ভাগ করে নিয়ে এক এক মডিউল আলাদাভাবে বিভিন্ন প্রোগ্রামের দিয়ে প্রোগ্রাম করা হলে এবং এগুলো ফাংশন হিসাবে ব্যবহার করা হলে স্বল্পতম সময়ে প্রোগ্রাম সম্পন্ন করা সম্ভব হয়।
- অত্যন্ত জরুরি প্রোগ্রামকে ফাংশন হিসাবে লাইব্রেরিতে জমা করে রাখলে অন্য ব্যবহারকারী প্রয়োজনে তা ব্যবহার করতে পারে।

**২. পাইথনে ফাংশন তৈরির ক্ষেত্রে কী কী নিয়ম মেনে চলতে হয়?**

**উত্তরঃ :** পাইথনে ফাংশন তৈরি বা ফাংশন ডিফাইন করার ক্ষেত্রে সুনির্দিষ্ট কিছু নিয়মকানুন মেনে চলতে হয়। যেমন-

- ফাংশন ব্লক অবশ্যই def কী-ওয়ার্ড দিয়ে শুরু হবে। def-এর পর একটা স্পেস দিয়ে ফাংশনের নাম থাকবে। ফাংশনের নামকরণের ক্ষেত্রে আইডেন্টিফায়ার নামকরণের নিয়মাবলি মানতে হবে। তবে ফাংশনের নাম আন্ডারস্কোর '-' দিয়ে শুরু করা যেতে পারে।
- ফাংশনের নামের পর প্যারেনথেসিস '()' থাকবে। প্যারেনথেসিসের ভিতর এক বা একাধিক প্যারামিটার (parameter) বা আরগুমেন্ট (argument) কমা দিয়ে ব্যবহার করা যাবে। প্যারামিটার বা আরগুমেন্ট হচ্ছে ফাংশন দিয়ে কী কী ডাটা পাস করবে। ফাংশনে এক বা একাধিক Parameter পাস করা যেতে পারে। আবার কোনো কোনো ফাংশনে কোনো Parameter নাও থাকতে পারে। এটা নির্ভর করে কী ধরনের ফাংশন লেখা হচ্ছে তার উপর। একের অধিক Parameter থাকলে তাদেরকে কমা দিয়ে দিয়ে লেখতে হয়।
- প্যারেনথেসিসের পর কোলন (:) চিহ্ন থাকবে।
- ফাংশনের সব স্টেটমেন্ট ইনডেন্টেড থাকবে। প্রথম স্টেটমেন্টে কমেন্ট ব্যবহার করলে ভালো হয়, যাতে ফাংশনের একটা সংক্ষিপ্ত পরিচয় থাকবে। তবে এটা বাধ্যতামূলক নয়।
- সাধারণত return কী-ওয়ার্ড ব্যবহার করে ফাংশনের এক্সিকিউশন শেষ করা হয়। কোনো ডাটা রিটার্ন করার ক্ষেত্রে তার কী-ওয়ার্ডের পর একটা স্পেস দিতে হবে। বস্তুত কাজ শেষে ফাংশনটি কী রিটার্ন করবে, তা-ই return দিয়ে পাস করা হয়। ফাংশন যদি কোনো কিছু রিটার্ন না করে তাহলে return 0 বা return না লেখলেও হবে।

**৩. লোকাল ভেরিয়েবল ও গ্লোবাল ভেরিয়েবলের মধ্যে পার্থক্য লেখ।** [বাকাশিবো-২০২৪ (অনুষ্ঠিত-২৫)]

**উত্তরঃ**

| বৈশিষ্ট্য | লোকাল ভেরিয়েবল (Local Variable) | গ্লোবাল ভেরিয়েবল (Global Variable) |
|---|---|---|
| সংজ্ঞা | এটি একটি ফাংশন বা ব্লকের ভেতরে ডিক্লেয়ার করা হয়। | এটি ফাংশনের বাইরে বা প্রোগ্রামের শুরুতে ডিক্লেয়ার করা হয়। |
| সুযোগ (Scope) | শুধুমাত্র ওই ফাংশন বা ব্লকের ভেতরেই এটি অ্যাক্সেস করা যায়। | প্রোগ্রামের যেকোনো অংশ বা যেকোনো ফাংশন থেকে এটি অ্যাক্সেস করা যায়। |
| জীবনকাল (Lifetime) | ফাংশনটির কাজ শেষ হওয়া মাত্রই এটি মেমোরি থেকে মুছে যায়। | প্রোগ্রামটি যতক্ষণ সচল থাকে, এটি মেমোরিতে ততক্ষণ স্থায়ী হয়। |
| ডেটা শেয়ারিং | এটি অন্যান্য ফাংশনের সাথে ডেটা শেয়ার করতে পারে না। | এটি সহজেই বিভিন্ন ফাংশনের মধ্যে ডেটা আদান-প্রদান করতে পারে। |
| মেমোরি অবস্থান | এটি সাধারণত স্ট্যাক (Stack) মেমোরিতে জমা থাকে। | এটি সাধারণত ফিক্সড বা গ্লোবাল মেমোরিতে জমা থাকে। |

8. Built in function & user define finction – **এর মাঝে পার্থক্য লেখ।** [বাকাশিবো-২০২৪ (অনুষ্ঠিত-২৫)]

**উত্তরঃ**

| বৈশিষ্ট্য | বিল্ট-ইন ফাংশন (Built-in Function) | ইউজার-ডিফাইন্ড ফাংশন (User-defined Function) |
|---|---|---|
| সংজ্ঞা | এই ফাংশনগুলো প্রোগ্রামিং ল্যাঙ্গুয়েজ বা কম্পাইলারের সাথে আগে থেকেই তৈরি থাকে। | প্রোগ্রামার নিজের বিশেষ কোনো কাজ সম্পন্ন করার জন্য এটি নিজে তৈরি করেন। |

| বৈশিষ্ট্য | বিল্ট-ইন ফাংশন (Built-in Function) | ইউজার-ডিফাইন্ড ফাংশন (User-defined Function) |
|---|---|---|
| লাইব্রেরি | এগুলো সাধারণত স্ট্যান্ডার্ড লাইব্রেরি বা হেডার ফাইলে থাকে। | এগুলো প্রোগ্রামের সোর্স কোডের ভেতরেই ডিফাইন করা হয়। |
| নামকরণ | এগুলোর নাম নির্দিষ্ট, যা পরিবর্তন করা সম্ভব নয়। (যেমন: print(), len()) | প্রোগ্রামার নিজের পছন্দমতো যেকোনো নাম দিতে পারেন। |
| ব্যবহারের উদ্দেশ্য | সাধারণ এবং বহুল ব্যবহৃত কাজগুলো (যেমন ইনপুট/আউটপুট) সহজ করার জন্য। | নির্দিষ্ট প্রজেক্ট বা জটিল কোনো লজিক সমাধানের জন্য। |
| উদাহরণ | Python-এ sum(), max(), input() ইত্যাদি। | def amar_function(): দিয়ে তৈরি করা ফাংশন। |

**৫. পাস বাই রেফারেন্স ও পাস বাই ভ্যালু কী, তা সংক্ষেপে লেখ।** [বাকাশিবো-২০২২ (অনুষ্ঠিত-২৩), ২৩ (পরি)]

**অথবা, Passing by reference কী?**

**উত্তরঃ** পাস বাই রেফারেন্স (Pass by reference): রেফারেন্স ভেরিয়েবলের আরগুমেন্টের মাধ্যমে এক ফাংশন থেকেঅন্য ফাংশনে ডাটা পাঠানোর পদ্ধতিকে পাস বাই রেফারেন্স বলা হয়। পাস বাই রেফারেন্স প্রক্রিয়ায় কোনো ফাংশন থেরে রেফারেন্স ভেরিয়েবলের আরগুমেন্ট বিশিষ্ট কোনো ফাংশন কল করার সময় কল্ড ফাংশনের প্রতিটি ফরমাল প্যারামিটারের বিপরীতে যে অ্যাকচুয়াল ডাটা ব্যবহার করা হয় সেগুলো সরাসরি মেমরি অ্যাড্রেস ব্যবহার করে। ফলে মূল ফাংশনে কর ফাংশনের আরগুমেন্ট বা লোকাল ভেরিয়েবলের অনুরূপ নাম এবং ডাটা টাইপে কোনো ভেরিয়েবল থাকলে কল্ড ফাংশনের আরগুমেন্ট বা লোকাল ভেরিয়েবলের মানের পরিবর্তে মূল ফাংশনের ভেরিয়েবলের মান পরিবর্তিত হয়।

পাস বাই ভ্যালু (Pass by value): ফাংশন কল করার সময় আরগুমেন্ট ভেরিয়েবলের মাধ্যমে এক ফাংশন থেকে অনা ফাংশনে ডাটা পাঠানোর পদ্ধতিকে পাস বাই ভ্যালু বলা হয়। পাস বাই ভ্যালু প্রক্রিয়ায় কম্পাইলার যখন কোনো ফাংশন কল করে তখন ব্যবহারকারী ফাংশনে ফাংশন কলের পরবর্তী স্টেটমেন্টের অ্যাড্রেস (যাকে রিটার্ন

অ্যাড্রেস বলা হয়), কষ্ঠ ফাংশনের আরগুমেন্টসমূহ (যদি থাকে) পর্যায়ক্রমে ডানদিক থেকে এবং সর্বশেষে কল্ড ফাংশনের লোকাল ভেরিয়েবলগুলো পর্যায়ক্রমে স্ট্যাকে (অস্থায়ী মেমরি) জমা রাখে।

## রচনামূলক প্রশ্ন

**১. ফাংশন কী? এর শ্রেণীবিভাগ আলোচনা কর।**

**উত্তরঃ** ফাংশন একটি যন্ত্রের মতো, যেখানে আপনি কিছু ইনপুট দেবেন এবং সেটি আপনাকে একটি নির্দিষ্ট আউটপুট বা ফলাফল দেবে।

ফাংশনকে মূলত দুটি প্রধান ভাগে ভাগ করা যায়। নিচে বিস্তারিত আলোচনা করা হলো:

১. লাইব্রেরি বা বিল্ট-ইন ফাংশন (Library/Built-in Functions)

এই ফাংশনগুলো প্রোগ্রামিং ল্যাঙ্গুয়েজের সাথেই আগে থেকে তৈরি করা থাকে। প্রোগ্রামারদের শুধু এগুলো ব্যবহার করার নিয়ম জানলেই চলে, এগুলো তৈরি করার প্রয়োজন হয় না।

- উদাহরণ: * C ল্যাঙ্গুয়েজে: printf(), scanf(), sqrt()
    - Python-এ: print(), len(), input()
- সুবিধা: এগুলো ব্যবহার করা সহজ এবং কোডিংয়ের সময় বাঁচায়।

২. ইউজার-ডিফাইন্ড ফাংশন (User-defined Functions)

প্রোগ্রামারের নিজের প্রয়োজন অনুযায়ী যখন কোনো ফাংশন তৈরি করা হয়, তখন তাকে ইউজার-ডিফাইন্ড ফাংশন বলে। এটি প্রোগ্রামের লজিক অনুযায়ী কাজ করে।

- গঠন: এই ফাংশনের তিনটি অংশ থাকে:
    1. Function Declaration: ফাংশনটি কী ধরণের হবে তা জানানো।
    2. Function Definition: ফাংশনটি আসলে কী কাজ করবে তা লেখা।
    3. Function Call: ফাংশনটিকে কাজ করার জন্য নির্দেশ দেওয়া।

কাজের ধরণের ওপর ভিত্তি করে আরও ৪টি ভাগ:

ফাংশন ডেটা গ্রহণ করছে কি না বা ফলাফল ফেরত দিচ্ছে কি না, তার ওপর ভিত্তি করে একে আরও চার ভাগে ভাগ করা যায়:

1. No Argument & No Return Value: কোনো ইনপুট নেয় না এবং কোনো ফলাফলও পাঠায় না।

2. With Argument & No Return Value: ইনপুট নেয় কিন্তু কোনো ফলাফল সরাসরি ফেরত দেয় না (শুধু প্রিন্ট করে দিতে পারে)।

3. No Argument & With Return Value: কোনো ইনপুট নেয় না কিন্তু প্রসেসিং শেষে একটি ফলাফল ফেরত পাঠায়।

4. With Argument & With Return Value: ইনপুটও নেয় এবং ফলাফলও ফেরত পাঠায়। (এটি সবচেয়ে বেশি ব্যবহৃত হয়)।

**২. উদাহারণসহ ফাংশন ডেফিনেশন ও ফাংশন কলিং প্রক্রিয়া ব্যাখ্যা কর।**

**উত্তরঃ** প্রোগ্রামিংয়ে একটি ফাংশন কীভাবে কাজ করবে তা লিখে রাখাকে বলা হয় ফাংশন ডেফিনেশন, আর সেই কাজটিকে কার্যকর করার জন্য ফাংশনটিকে ডাকা বা ব্যবহার করাকে বলা হয় ফাংশন কলিং।

১. ফাংশন ডেফিনেশন (Function Definition)

ফাংশন ডেফিনেশন হলো সেই অংশ যেখানে ফাংশনের নাম, ইনপুট (Parameters) এবং এর ভেতরে কী কাজ হবে তা নির্ধারণ করা হয়। এটি অনেকটা রান্নার রেসিপি লিখে রাখার মতো।

```
def ফাংশনের_নাম(প্যারামিটার):
    # কোড বা লজিক
    return ফলাফল
```

২. ফাংশন কলিং (Function Calling)

ফাংশন ডিফাইন করলেই সেটি নিজে নিজে কাজ শুরু করে না। একে কাজ করানোর জন্য এর নাম ধরে ডাকতে হয়, যাকে বলা হয় ফাংশন কলিং। এটি অনেকটা রেসিপি দেখে রান্না শুরু করার মতো।

**৩. ফাংশন ব্যবহারপূর্বক লিস্ট – এর ৫ টি সংখ্যাকে যোগ করার প্রগ্রাম লেখ।** [বাকাশিবো-২০২২ (অনুষ্ঠিত-২৩)]

**উত্তরঃ** # ১. ফাংশন ডেফিনেশন

```
def sum_list_numbers(numbers):
total = 0
for n in numbers:
total = total + n
```

```
return total

# ২. ৫টি সংখ্যার একটি লিস্ট তৈরি



my_list = [10, 20, 30, 40, 50]

# ৩. ফাংশন কল করা

result = sum_list_numbers(my_list)

# ফলাফল প্রদর্শন

print("লিস্টের সংখ্যাগুলোর যোগফল:", result)
```

**৪. লাইব্রেরি ফাংশন এবং ইউজার ডিফাইন্ড ফাংশনের মাঝে পার্থক্য লেখ।** [বাকাশিবো-২০২৩ (অনুষ্ঠিত-২৪)]

অথবা, **Library function ও User define function-এর মাঝে পার্থক্য লেখ।**

**উত্তরঃ**

| **বৈশিষ্ট্য** | **লাইব্রেরি ফাংশন** | **ইউজার ডিফাইন্ড ফাংশন** |
|---|---|---|
| **সংজ্ঞা** | এই ফাংশনগুলো কম্পাইলারে আগে থেকেই তৈরি করা থাকে। | প্রোগ্রামার নিজের প্রয়োজন অনুযায়ী এই ফাংশন তৈরি করেন। |
| **ঘোষণা** | এগুলো ব্যবহারের জন্য নির্দিষ্ট হেডার ফাইল (যেমন: stdio.h) যুক্ত করতে হয়। | এগুলো ব্যবহারের জন্য প্রোগ্রামারকে নিজেই ফাংশনটি ডিফাইন করতে হয়। |
| **নামকরণ** | এর নাম পরিবর্তন করা সম্ভব নয়। | প্রোগ্রামার নিজের ইচ্ছেমতো নাম দিতে পারেন। |
| **কোড লিখা** | ব্যবহারকারীকে এই ফাংশনের ভেতরে কী কোড আছে তা লিখতে হয় না। | ব্যবহারকারীকে ফাংশনের সম্পূর্ণ লজিক বা কোড লিখতে হয়। |
| **উদাহরণ** | printf(), scanf(), sqrt(), pow() ইত্যাদি। | sum(), calculateArea(), display() ইত্যাদি। |

**৫. Function ব্যবহার করে ধনাত্মক পূর্ণ সংখ্যার Factorial-এর মান নির্ণয় করার Program লেখ।**
[বাকাশিবো-২০২৩ (অনুষ্ঠিত-২৪)]

**উত্তরঃ** def find_factorial(n):

```
    """এটি একটি ইউজার ডিফাইন্ড ফাংশন যা ফ্যাক্টোরিয়াল নির্ণয় করে"""
    fact = 1
    for i in range(1, n + 1):


        fact *= i
    return fact
# ব্যবহারকারীর কাছ থেকে ইনপুট গ্রহণ
try:
    num = int(input("একটি ধনাত্মক পূর্ণ সংখ্যা লিখুন: "))
    if num < 0:


        print("দুঃখিত, ঋণাত্মক সংখ্যার ফ্যাক্টোরিয়াল হয় না।")
    else:
        result = find_factorial(num)
        print(f"{num} এর ফ্যাক্টোরিয়াল হলো: {result}")
except ValueError:
    print("অনুগ্রহ করে একটি সঠিক পূর্ণ সংখ্যা লিখুন।")
```

**৬. ফাংশন ব্যবহার করে ত্রিভুজ ক্ষেত্রের ক্ষেত্রফল নির্ণয়ের প্রোগ্রাম লেখ।** [বাকাশিবো-২০২৩ (পরি), ২০২৪ (অনুষ্ঠিত-২৫)]]

**উত্তরঃ**

```
def calculate_triangle_area(base, height):
    area = 0.5 * base * height
    return area
try:
```

```
    b = float(input("ত্রিভুজের ভূমির দৈর্ঘ্য লিখুন: "))
    h = float(input("ত্রিভুজের উচ্চতা লিখুন: "))
    if b <= 0 or h <= 0:
        print("ভূমি এবং উচ্চতা অবশ্যই ধনাত্মক সংখ্যা হতে হবে।")
    else:
        result = calculate_triangle_area(b, h)
        print(f"ত্রিভুজটির ক্ষেত্রফল হলো: {result} বর্গ একক")
except ValueError:
    print("অনুগ্রহ করে সঠিক সংখ্যা ইনপুট দিন।")
```

# অধ্যায় ০২ পাইথন-এর ফাইল অপারেশন

এই অধ্যায়ের প্রধান সূচিপত্র (contants)


| বিষয় | পৃষ্ঠা নং |
|---|---|
| ❖ সূত্রাবলী ও শর্টকার্ট টেকনিক | - |
| ❖ অতি সংক্ষিপ্ত প্রশ্নাবলি | ১৪ |
| ❖ সংক্ষিপ্ত প্রশ্নাবলি | ১৫ |
| ❖ রচনামূলক প্রশ্নাবলি | ১৭ |


## বোর্ড প্রশ্নাবলির বিশ্লেষন (Board Questions Analysis)

এই অধ্যায় থেকে বিগত তিন বছরের ডিপ্লোমা ইন ইঞ্জিনিয়ারিং, ডিপ্লোমা ইন টেক্সটাইল, ডিপ্লোমা ইন এগ্রিকালচার, ডিপ্লোমা ইন মেরিন পরীক্ষায় মোট প্রশ্ন....নিচে 'Board Questions Analysis' অংশে এই অধ্যায় থেকে কোন সালে কোন বোর্ডে কতটি প্রশ্ন হয়েছে তা দেখানো হলো।

| সাল / বোর্ড | অতিসংক্ষিপ্ত | সংক্ষিপ্ত | রচনামূলক |
|---|---|---|---|
| ২০২৪ | ১টি | -- | -- |
| ২০২৩ | ২টি | -- | -- |
| ২০২২ | ১টি | -- | -- |

"গণিত তার কাছেই প্রকৃত সৌন্দর্য সহকারে ধরা দেয়,
যে বিশুদ্ধ মন ও ভালোবাসা নিয়ে গণিতের দিকে অগ্রসর হয়"

আর্কিমিডিস

## অতি সংক্ষিপ্ত প্রশ্ন

**১. ফাইল কী?**

**উত্তরঃ** ফাইল হলো কম্পিউটারে স্থায়ীভাবে সংরক্ষিত তথ্যের বা ডেটার একটি সংগ্রহ। এটি হার্ডডিস্ক বা অন্য কোনো স্টোরেজ ডিভাইসে নির্দিষ্ট নামে জমা থাকে, যাতে পরবর্তীতে প্রয়োজন অনুযায়ী ডেটা পড়া বা ব্যবহার করা যায়।

**২. ফাইল কত প্রকার ও কী কী?**

**উত্তরঃ** ডেটার ধরন ও সংরক্ষণের পদ্ধতির ওপর ভিত্তি করে ফাইল প্রধানত ২ প্রকার:

- টেক্সট ফাইল (Text File): এতে সাধারণ লেখা (ASCII বা Unicode) থাকে যা মানুষ সহজে পড়তে পারে (যেমন: .txt, .py, .c)।
- বাইনারি ফাইল (Binary File): এতে ডেটা ০ এবং ১ আকারে থাকে, যা সরাসরি পড়া যায় না (যেমন: .exe, .jpg, .mp3)।

**৩. ফাইল অপারেশন কী?** [বাকাশিবো-২০২২ (অনুষ্ঠিত-২৩)]

**উত্তরঃ** একটি ফাইলের ওপর যে সকল কাজ করা যায়, সেগুলোকে ফাইল অপারেশন বলে। প্রধান ফাইল অপারেশনগুলো হলো:

- ফাইল খোলা (Open)
- ফাইল থেকে তথ্য পড়া (Read)
- ফাইলে তথ্য লেখা (Write)
- ফাইল বন্ধ করা (Close)

**৪. পাইথন প্রোগামে কী কী উপায়ে মান ইনপুট করা হয়?** [বাকাশিবো-২০২৩ (পরি)]

**উত্তরঃ** পাইথনে মূলত দুইভাবে ইনপুট নেওয়া যায়:

- স্ট্যান্ডার্ড ইনপুট: input() ফাংশন ব্যবহার করে সরাসরি ব্যবহারকারীর কাছ থেকে।
- ফাইল ইনপুট: কোনো ফাইল থেকে ডেটা পড়ার মাধ্যমে (যেমন: read(), readline())।

**৫. Open () ফাংশন Syntax লেখ।** [বাকাশিবো-২০২৪ (অনুষ্ঠিত-২৫)]

**উত্তরঃ** পাইথনে ফাইল খোলার সিনট্যাক্স হলো:

file_object = open("file_name", "mode")

*এখানে file_name হলো ফাইলের নাম এবং mode হলো আপনি ফাইলটি কী উদ্দেশ্যে খুলছেন (যেমন: 'r' পড়ার জন্য, 'w' লেখার জন্য)।*

**৬. Close () ফাংশনের কাজ কী?** [বাকাশিবো-২০২৩ (পরি)]

**উত্তরঃ** close() ফাংশনের কাজ হলো খোলা ফাইলটি সঠিকভাবে বন্ধ করা। ফাইল বন্ধ করলে মেমোরি খালি হয় এবং ফাইলে করা পরিবর্তনগুলো স্থায়ীভাবে সংরক্ষিত (Save) হওয়া নিশ্চিত হয়।

**৭. Read () ফাংশনের Syntax লেখ।**

**উত্তরঃ** ফাইল থেকে ডেটা পড়ার সিনট্যাক্স হলো:

file_object.read(size)

*এখানে size অপশনাল। যদি কোনো সংখ্যা দেওয়া হয় তবে ততগুলো ক্যারেক্টার পড়বে, আর কিছু না দিলে পুরো ফাইলটি পড়বে।*

**৮. Realine () ফাংশনের কাজ কী।**

**উত্তরঃ** পাইথনে readline() ফাংশনের প্রধান কাজ হলো ফাইল থেকে একটি নির্দিষ্ট লাইন পড়া। এটি কল করা হলে ফাইল পয়েন্টার যেখানে থাকে, সেখান থেকে শুরু করে ওই লাইনের শেষ (নিউলাইন ক্যারেক্টার \n সহ) পর্যন্ত ডেটা রিটার্ন করে।

## সংক্ষিপ্ত প্রশ্ন

**১. ফাইল অপারেশন ব্যাখ্যা কর।**

**উত্তরঃ** একটি প্রোগ্রামের মাধ্যমে কোনো ফাইলের ওপর যে সকল কাজ সম্পন্ন করা হয়, তাকে ফাইল অপারেশন বলে। পাইথনে মূলত চারটি প্রধান ধাপে ফাইল অপারেশন সম্পন্ন হয়:

- Open (খোলা): কোনো কাজ করার আগে ফাইলটিকে মেমোরিতে ওপেন করতে হয়।
- Read/Write (পড়া বা লেখা): ফাইল থেকে তথ্য সংগ্রহ করা অথবা নতুন কোনো তথ্য ফাইলে যোগ করা।

- Process (প্রক্রিয়াকরণ): ফাইল থেকে পাওয়া ডেটা নিয়ে প্রয়োজনীয় হিসাব-নিকাশ করা।
- Close (বন্ধ করা): কাজ শেষ হলে ফাইলটি বন্ধ করে মেমোরি মুক্ত করা।

**২. Open () ফাংশনের প্যারামিটারসমূহ বর্ণনা কর।**

**উত্তরঃ** Open () ফাংশনের প্যারামিটারসমূহ বর্ণনা কর।

উত্তর: পাইথনের open() ফাংশনে সাধারণত দুটি প্রধান প্যারামিটার থাকে। যেমন: open(file, mode)

- File (ফাইল পাথ): এটি প্রথম প্যারামিটার। এখানে ফাইলের নাম বা পূর্ণ লোকেশন (Path) স্ট্রিং আকারে দিতে হয়। উদাহরণ: "data.txt"।
- Mode (মোড): এটি দ্বিতীয় প্যারামিটার। ফাইলটি কী উদ্দেশ্যে খোলা হচ্ছে তা এখানে নির্দিষ্ট করা হয়। কিছু কমন মোড হলো:
    - 'r': শুধুমাত্র পড়ার জন্য (Read)।
    - 'w': নতুন করে লেখার জন্য (Write) - আগের তথ্য মুছে যায়।
    - 'a': ফাইলের শেষে নতুন তথ্য যোগ করার জন্য (Append)।
    - 'b': বাইনারি মোডে খোলার জন্য।

**৩. Close () ফাংশন ব্যবহারের সুবিধাগুলো লেখ।**

**উত্তরঃ** Close () ফাংশন ব্যবহারের সুবিধাগুলো লেখ।

উত্তর: ফাইল ব্যবহারের পর close() ফাংশন দিয়ে ফাইল বন্ধ করার প্রধান সুবিধাগুলো হলো:

- মেমোরি রক্ষা: ফাইল খোলা থাকলে কম্পিউটার মেমোরি (RAM) দখল করে রাখে। বন্ধ করলে মেমোরি খালি হয়।
- ডেটা সুরক্ষা: ফাইল সঠিকভাবে বন্ধ না করলে অনেক সময় বাফার (Buffer) এ থাকা ডেটা হার্ডডিস্কে সেভ হয় না।
- অন্য প্রোগ্রামের ব্যবহারের সুযোগ: একটি ফাইল এক জায়গায় খোলা থাকলে অন্য কোনো প্রোগ্রাম অনেক সময় সেটি ব্যবহার করতে পারে না।
- সিস্টেম রিসোর্স: একটি অপারেটিং সিস্টেম একসাথে সীমিত সংখ্যক ফাইল খুলে রাখতে পারে। close() করলে সেই সীমা বজায় থাকে।

**৪. ফাইল অপারেশনের সুবিধা লেখ।** [বাকাশিবো-২০২২ (অনুষ্ঠিত-২৩)]

**উত্তরঃ** প্রোগ্রামে ফাইল অপারেশনের প্রধান সুবিধাগুলো নিচে দেওয়া হলো:

- স্থায়ী সংরক্ষণ (Persistence): প্রোগ্রামের আউটপুট বা ডেটা কম্পিউটারের মেমোরিতে স্থায়ীভাবে জমা থাকে, যা প্রোগ্রাম বন্ধ করার পরও হারিয়ে যায় না।
- বিশাল ডেটা হ্যান্ডলিং: অনেক বড় আকারের তথ্য যা কিবোর্ড দিয়ে ইনপুট দেওয়া সম্ভব নয়, তা ফাইল থেকে সরাসরি পড়া যায়।
- ডেটা পুনরায় ব্যবহার: একবার ফাইলে ডেটা সেভ করে রাখলে পরবর্তীতে অন্য কোনো প্রোগ্রামেও সেই ডেটা ব্যবহার করা যায়।
- সহজ ব্যাকআপ: ফাইলের মাধ্যমে খুব সহজেই প্রয়োজনীয় তথ্যের ব্যাকআপ কপি তৈরি করে রাখা সম্ভব।

## রচনামূলক প্রশ্ন

**১. ফাইল ওপেনিং ও ক্লোজিং ফাংশন বর্ণনা কর।**

**উত্তরঃ** ফাইল ওপেনিং ফাংশন: open()

পাইথনে কোনো ফাইল নিয়ে কাজ করতে হলে প্রথমে open() ফাংশন ব্যবহার করে সেটি খুলতে হয়। এই ফাংশনটি একটি File Object রিটার্ন করে।

সিনট্যাক্স: file_object = open("filename", "mode")

প্যারামিটার বর্ণনা:

- filename: ফাইলের নাম বা লোকেশন (যেমন: "test.txt")।
- mode: এটি ফাইলটি কী উদ্দেশ্যে খোলা হচ্ছে তা নির্ধারণ করে। উল্লেখযোগ্য কিছু মোড হলো:
    - 'r' (Read): ফাইল পড়ার জন্য (ডিফল্ট মোড)।
    - 'w' (Write): নতুন কিছু লেখার জন্য (আগের তথ্য মুছে যায়)।
    - 'a' (Append): ফাইলের শেষে নতুন তথ্য যোগ করার জন্য।
    - 'x' (Create): নতুন ফাইল তৈরির জন্য।

ফাইল ক্লোজিং ফাংশন: close()

ফাইলের কাজ শেষ হওয়ার পর close() মেথড ব্যবহার করে ফাইলটি বন্ধ করা অত্যন্ত জরুরি। এটি সিস্টেম রিসোর্স মুক্ত করে এবং নিশ্চিত করে যে সমস্ত ডেটা হার্ডডিস্কে সঠিকভাবে সেভ হয়েছে।

সিনট্যাক্স:

```
file_object.close()
```

কেন ক্লোজ করা প্রয়োজন?

- ফাইল ক্লোজ না করলে মেমোরি (RAM) অপচয় হয়।
- অনেক সময় ফাইল ক্লোজ না করলে ফাইলে লেখা তথ্য পুরোপুরি সেভ হয় না (Buffer issues)।
- একই ফাইল অন্য কোনো প্রোগ্রামে ব্যবহার করতে সমস্যা হতে পারে।

উদাহরণের মাধ্যমে ব্যবহার:

```
# ১. ফাইলটি লেখার জন্য ওপেন করা
my_file = open("example.txt", "w")

# ২. ফাইলে কিছু লেখা
my_file.write("Hello, this is a test file.")

# ৩. ফাইলটি ক্লোজ করা
my_file.close()
```

# অধ্যায় ০৩ মডিউল,প্যাকেজ এবং অ্যাপ্লিকেশন সফটওয়্যার

এই অধ্যায়ের প্রধান সূচিপত্র (contants)


| বিষয় | পৃষ্ঠা নং |
|---|---|
| ❖ সূত্রাবলী ও শর্টকার্ট টেকনিক | - |
| ❖ অতি সংক্ষিপ্ত প্রশ্নাবলি | ২০ |
| ❖ সংক্ষিপ্ত প্রশ্নাবলি | ২১ |
| ❖ রচনামূলক প্রশ্নাবলি | ২৪ |


## বোর্ড প্রশ্নাবলির বিশ্লেষন (Board Questions Analysis)

এই অধ্যায় থেকে বিগত তিন বছরের ডিপ্লোমা ইন ইঞ্জিনিয়ারিং, ডিপ্লোমা ইন টেক্সটাইল, ডিপ্লোমা ইন এগ্রিকালচার, ডিপ্লোমা ইন মেরিন পরীক্ষায় মোট প্রশ্ন....নিচে 'Board Questions Analysis' অংশে এই অধ্যায় থেকে কোন সালে কোন বোর্ডে কতটি প্রশ্ন হয়েছে তা দেখানো হলো।

| সাল / বোর্ড | অতিসংক্ষিপ্ত | সংক্ষিপ্ত | রচনামূলক |
|---|---|---|---|
| ২০২৪ | ১টি | ১টি | ১টি |
| ২০২৩ | ২টি | ৩টি | -- |
| ২০২২ | -- | ১টি | -- |

"গণিত তার কাছেই প্রকৃত সৌন্দর্য সহকারে ধরা দেয়,
যে বিশুদ্ধ মন ও ভালোবাসা নিয়ে গণিতের দিকে অগ্রসর হয়"

আর্কিমিডিস

## অতি সংক্ষিপ্ত প্রশ্ন

**১. অ্যাপ্লিকেশন সফটওয়্যার কী?** [বাকাশিবো-২০২৪ (অনুষ্ঠিত-২৫)]

**উত্তরঃ** অ্যাপ্লিকেশন সফটওয়্যার হলো এমন এক ধরনের প্রোগ্রাম বা প্রোগ্রামের সমষ্টি যা ব্যবহারকারী কে কোনো নির্দিষ্ট কাজ (যেমন: লেখালেখি, ছবি এডিট করা বা হিসাব রাখা) সম্পন্ন করতে সাহায্য করে। উদাহরণ: MS Word, Google Chrome।

**২. অ্যাপ্লিকেশন সফটওয়্যার এর প্রকারভেদগুলো লেখ।**

**উত্তরঃ** অ্যাপ্লিকেশন সফটওয়্যার সাধারণত ২ প্রকার: সিস্টেম সফটওয়্যার এবং অ্যাপ্লিকেশন সফটওয়্যার।

**৩. মডিউল কী?**

**উত্তরঃ** পাইথনে মডিউল হলো একটি ফাইল যার মধ্যে বিভিন্ন ফাংশন, ক্লাস বা ভেরিয়েবল সংজ্ঞায়িত করা থাকে। এটি কোডকে পুনরায় ব্যবহারযোগ্য এবং সুসংগঠিত করতে সাহায্য করে। একটি .py ফাইলই মূলত একটি মডিউল।

**৪. প্যাকেজ কী?** [বাকাশিবো-২০২৩ (অনুষ্ঠিত-২৪)]

**উত্তরঃ** প্যাকেজ হলো অনেকগুলো মডিউলের একটি সংগ্রহ বা ডিরেক্টরি (Folder)। যখন অনেকগুলো মডিউলকে একটি নির্দিষ্ট ফোল্ডারে গুছিয়ে রাখা হয় এবং সেখানে একটি __init__.py ফাইল থাকে, তখন তাকে প্যাকেজ বলে।

**৫. মডিউল তৈরির Syntax লেখ।**

**উত্তরঃ** পাইথনে মডিউল তৈরির বিশেষ কোনো সিনট্যাক্স নেই। যেকোনো কোড লিখে ফাইলটি .py এক্সটেনশনে সেভ করলেই তা মডিউল হয়ে যায়। ব্যবহারের নিয়ম:

```
import module_name
```

**৬. প্যাকেজ তৈরির Syntax লেখ।**

**উত্তরঃ** প্যাকেজ তৈরির পদ্ধতি হলো: ১. একটি ফোল্ডার তৈরি করা (যেমন: my_package) ২. তার ভেতরে একটি খালি ফাইল __init__.py তৈরি করা। ৩. প্রয়োজনীয় মডিউলগুলো সেই ফোল্ডারে রাখা। ব্যবহারের নিয়ম:

```
from my_package import module_name
```

**৭. আজকের Datetime কোন ফাংশন ব্যবহার করে প্রিন্ট করা হয়?**

**উত্তরঃ** পাইথনের datetime মডিউলের now() ফাংশন ব্যবহার করে আজকের তারিখ ও সময় প্রিন্ট করা হয়। উদাহরণ: datetime.datetime.now()

**৮. Matplotaib কী?**

**উত্তরঃ** Matplotlib হলো পাইথনের একটি শক্তিশালী লাইব্রেরি যা ডেটা ভিজ্যুয়ালাইজেশন বা গ্রাফ তৈরির কাজে ব্যবহৃত হয়। এর মাধ্যমে ২ডি (2D) গ্রাফ, চার্ট, হিস্টোগ্রাম ইত্যাদি তৈরি করা যায়।

**৯. Pandas কী?** [বাকাশিবো-২০২৩ (পরি)]

**উত্তরঃ** Pandas হলো পাইথনের একটি জনপ্রিয় ওপেন সোর্স লাইব্রেরি যা ডেটা অ্যানালাইসিস (Data Analysis) এবং ডেটা ম্যানিপুলেশনের জন্য ব্যবহৃত হয়। এটি মূলত টেবিল আকারের ডেটা (যেমন: Excel বা SQL ডেটা) নিয়ে কাজ করতে অত্যন্ত কার্যকর।

## সংক্ষিপ্ত প্রশ্ন

**১. মডিউল ও প্যাকেজ এর পার্থক্য লেখ।** [বাকাশিবো-২০২২(অনুষ্ঠিত-২৩) ২০২৩ (পরি)]

**উত্তরঃ**

| **বৈশিষ্ট্য** | **মডিউল (Module)** | **প্যাকেজ (Package)** |
|---|---|---|
| **সংজ্ঞা** | মডিউল হলো একটি একক পাইথন ফাইল যা ফাংশন, ক্লাস বা ভেরিয়েবল ধারণ করে। | প্যাকেজ হলো অনেকগুলো মডিউলের একটি ডিরেক্টরি বা ফোল্ডার। |
| **গঠন** | এটি মূলত একটি .py এক্সটেনশনযুক্ত ফাইল। | এটি একটি ফোল্ডার যার ভেতরে একাধিক .py ফাইল থাকে। |
| **অস্তিত্ব** | মডিউল নিজেই একটি ফাইল হিসেবে কাজ করে। | প্যাকেজ হতে হলে ফোল্ডারের ভেতরে অবশ্যই একটি __init__.py ফাইল থাকতে হয়। |
| **জটিলতা** | এটি ছোট এবং সাধারণ কাজের জন্য ব্যবহৃত হয়। | এটি বড় এবং জটিল প্রজেক্টের জন্য ব্যবহৃত হয়। |

| | | |
|---|---|---|
| **আমদানি (Import)** | সাধারণত import module_name দিয়ে কল করা হয়। | এটি import package_name.module_name দিয়ে কল করা হয়। |

**২. মডিউল তৈরি করার Syntax ও উদাহারণ লেখ।** [বাকাশিবো-২০২৩ (পরি)]

**উত্তরঃ** পাইথনে মডিউল তৈরির জন্য বিশেষ কোনো আলাদা কমান্ডের প্রয়োজন হয় না। যেকোনো পাইথন ফাইলই (.py) একটি মডিউল।

Syntax: যেকোনো টেক্সট এডিটরে ফাংশন বা ভেরিয়েবল লিখে ফাইলটি filename.py নামে সেভ করতে হয়। ব্যবহারের সময় import filename লিখতে হয়।

উদাহরণ: ১. প্রথমে একটি ফাইল তৈরি করি যার নাম calculator.py:

```
# এটি একটি মডিউল
def add(a, b):
    return a + b
def multiply(a, b):
    return a * b
```

এবার অন্য একটি ফাইলে মডিউলটি ব্যবহার করি:

```
import calculator
result = calculator.add(10, 5)
print(result) # আউটপুট: 15
```

**৩. প্যাকেজ তৈরি করার Syntax ও উদাহারণ লেখ।**

**উত্তরঃ** প্যাকেজ হলো অনেকগুলো মডিউলের সমষ্টি যা একটি নির্দিষ্ট ডিরেক্টরি বা ফোল্ডারে থাকে।

Syntax/পদ্ধতি: ১. একটি ফোল্ডার তৈরি করতে হবে। ২. ফোল্ডারের ভেতর একটি __init__.py ফাইল থাকতে হবে (এটি খালি হতে পারে)। ৩. প্রয়োজনীয় মডিউলগুলো (.py ফাইল) ঐ ফোল্ডারে রাখতে হবে।

উদাহরণ: ধরা যাক আমাদের প্যাকেজের নাম MyMath:

- MyMath/ (ফোল্ডার)
    - __init__.py
    - basic.py (মডিউল)

ব্যবহারের নিয়ম:

```
from MyMath import basic
print(basic.add(5, 10))
```

**৪. Matplotlip – এর কাজ কী?** [বাকাশিবো-২০২৩ (পরি)]

**উত্তরঃ** Matplotlib হলো পাইথনের সবচেয়ে জনপ্রিয় ডেটা ভিজ্যুয়ালাইজেশন লাইব্রেরি। এর প্রধান কাজগুলো হলো:

- গ্রাফ তৈরি: তথ্যের ওপর ভিত্তি করে লাইন গ্রাফ, বার চার্ট, পাই চার্ট ইত্যাদি তৈরি করা।
- ডেটা বিশ্লেষণ: বিশাল পরিমাণ তথ্যকে ছবির মাধ্যমে উপস্থাপন করে সহজবোধ্য করা।
- বৈজ্ঞানিক চিত্র: গবেষণার কাজে বিভিন্ন গাণিতিক ও বৈজ্ঞানিক ডেটার চিত্রায়ন করা।
- কাস্টমাইজেশন: গ্রাফের রং, লেবেল, টাইটেল এবং স্টাইল পরিবর্তন করে আকর্ষণীয় করে তোলা।

**৫. Pandas – এর কাজ কী?**

**উত্তরঃ** Pandas হলো পাইথনের একটি শক্তিশালী লাইব্রেরি যা মূলত ডেটা ম্যানিপুলেশন ও অ্যানালাইসিসের জন্য ব্যবহৃত হয়। এর প্রধান কাজগুলো হলো:

- ডেটা লোড করা: CSV, Excel, SQL বা JSON ফাইল থেকে ডেটা পড়া।
- ডেটা ক্লিনিং: অগোছালো বা ভুল ডেটা ঠিক করা এবং খালি ঘর (Null values) পূরণ করা।
- ডেটা স্লাইসিং: বড় ডেটাসেট থেকে প্রয়োজনীয় কলাম বা রো আলাদা করা।
- ডেটা ফ্রেম (DataFrame): টেবিল আকারে ডেটা সাজিয়ে রাখা যা দিয়ে খুব সহজে হিসাব-নিকাশ করা যায়।
- পরিসংখ্যান: ডেটার গড় (Mean), মধ্যমা (Median), বা সর্বোচ্চ/সর্বনিম্ন মান দ্রুত নির্ণয় করা।

**৬. মডিউল ও প্যাকেজ এর গুরুত্ব লেখ।** [বাকাশিবো-২০২৪ (অনুষ্ঠিত-২৫)]

**উত্তরঃ** প্রোগ্রামিংয়ে বড় কোডকে ছোট ছোট অংশে ভাগ করার জন্য মডিউল এবং প্যাকেজ অত্যন্ত গুরুত্বপূর্ণ। এদের গুরুত্ব নিচে আলোচনা করা হলো:

- কোড পুনরায় ব্যবহারযোগ্যতা (Reusability): একবার একটি মডিউল বা প্যাকেজ তৈরি করলে তা বারবার বিভিন্ন প্রোগ্রামে ব্যবহার করা যায়। ফলে বারবার একই কোড লিখতে হয় না।
- কোড ব্যবস্থাপনা (Organization): বিশাল বড় কোডকে ছোট ছোট মডিউলে ভাগ করলে কোড পরিচালনা করা সহজ হয়। এটি প্রজেক্টের জটিলতা কমায়।
- সহজ ডিবাগিং (Easier Debugging): প্রোগ্রামে কোনো ভুল বা এরর থাকলে নির্দিষ্ট মডিউল পরীক্ষা করে তা দ্রুত ঠিক করা যায়।

- টিম ওয়ার্ক (Teamwork): বড় প্রজেক্টে কাজ করার সময় আলাদা আলাদা মডিউলে কাজ ভাগ করে দিলে অনেক প্রোগ্রামার একসাথে কাজ করতে পারেন।
- নামকরণ সংঘর্ষ এড়ানো (Namespace Management): একই নামের ফাংশন বা ভেরিয়েবল আলাদা মডিউলে থাকলে একে অপরের সাথে মিশে যায় না।

## রচনামূলক প্রশ্ন

**১. অ্যাপ্লিকেশন সফটওয়্যার বলতে কী বুঝ? এর শ্রেণিবিভাগ বর্ণনা কর।** [বাকাশিবো-২০২৪ (অনুষ্ঠিত-২৫)]

**উত্তরঃ** অ্যাপ্লিকেশন সফটওয়্যার (Application Software): অ্যাপ্লিকেশন সফটওয়্যার হলো এমন এক ধরনের প্রোগ্রাম বা প্রোগ্রামের সমষ্টি, যা ব্যবহারকারীকে কোনো নির্দিষ্ট কাজ সম্পন্ন করতে সরাসরি সহায়তা করে। একে সংক্ষেপে 'অ্যাপ' (App) বলা হয়। যেমন—লেখালেখির জন্য MS Word, ছবি সম্পাদনার জন্য Photoshop ইত্যাদি।

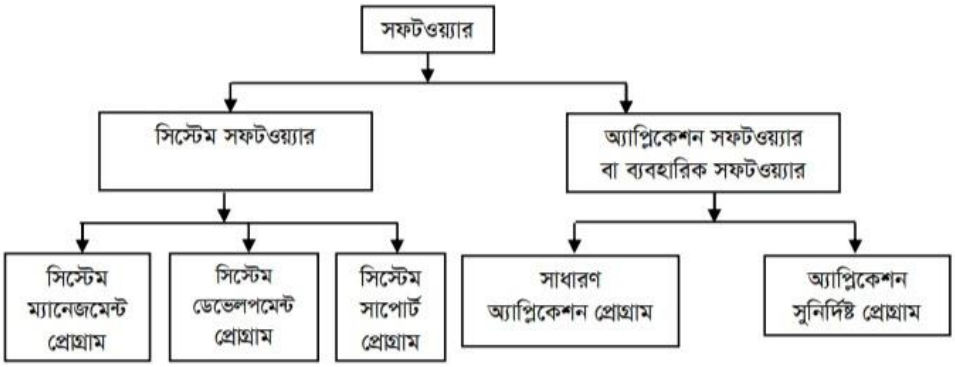
অ্যাপ্লিকেশন সফটওয়্যারের শ্রেণিবিভাগ: অ্যাপ্লিকেশন সফটওয়্যারকে প্রধানত দুই ভাগে ভাগ করা যায়:

- ১. প্যাকেজ সফটওয়্যার (Package Software): সাধারণ ব্যবহারকারীদের কথা মাথায় রেখে যখন কোনো সফটওয়্যার তৈরি করা হয়, যা সবার জন্য একই রকম কাজ করে, তাকে প্যাকেজ সফটওয়্যার বলে। এটি বাজারে বা অনলাইনে সবার জন্য উন্মুক্ত থাকে।
    - উদাহরণ: Microsoft Office, Google Chrome, Media Player।
- ২. কাস্টমাইজড বা টেইলর-মেড সফটওয়্যার (Customized Software): যখন কোনো নির্দিষ্ট প্রতিষ্ঠান বা ব্যক্তির চাহিদা অনুযায়ী বিশেষ কোনো কাজের জন্য সফটওয়্যার তৈরি করা হয়, তাকে কাস্টমাইজড সফটওয়্যার বলে।
    - উদাহরণ: ব্যাংকিং সফটওয়্যার, হাসপাতালের ম্যানেজমেন্ট সিস্টেম, সুপার শপের ইনভেন্টরি সফটওয়্যার।

কাজের ধরন অনুযায়ী আরও কিছু বিভাগ:

- ওয়ার্ড প্রসেসিং সফটওয়্যার: (যেমন: MS Word)
- স্প্রেডশিট সফটওয়্যার: (যেমন: MS Excel)
- ডেটাবেস সফটওয়্যার: (যেমন: Oracle, MS Access)

- গ্রাফিক্স সফটওয়্যার: (যেমন: Adobe Illustrator)




**২. মডিউল ও প্যাকেজ – এর গুরুত্ব বর্ণনা কর।**

**উত্তরঃ** পাইথনে বড় বড় প্রজেক্ট বা প্রোগ্রামকে ছোট ছোট অংশে ভাগ করে পরিচালনা করার জন্য মডিউল ও প্যাকেজ অপরিহার্য। এদের গুরুত্ব নিচে বিস্তারিত আলোচনা করা হলো:

- ১. কোড পুনরায় ব্যবহারযোগ্যতা (Reusability): একবার একটি মডিউল বা প্যাকেজ তৈরি করলে তা বারবার বিভিন্ন প্রোগ্রামে ব্যবহার করা যায়। ফলে একই কোড বারবার লিখতে হয় না, যা সময় ও পরিশ্রম সাশ্রয় করে।

- ২. সহজ ব্যবস্থাপনা (Better Organization): হাজার হাজার লাইনের কোডকে একটি ফাইলে না রেখে ছোট ছোট মডিউলে ভাগ করলে কোড পরিচালনা করা অনেক সহজ হয়। এটি প্রজেক্টের জটিলতা কমিয়ে দেয়।

- ৩. টিম-ওয়ার্ক বা দলগত কাজ: বড় প্রজেক্টে কাজ করার সময় আলাদা আলাদা মডিউল বা প্যাকেজ তৈরি করে বিভিন্ন প্রোগ্রামারের মধ্যে কাজ ভাগ করে দেওয়া যায়। এতে কাজের গতি বাড়ে।

- ৪. ত্রুটি সংশোধন বা ডিবাগিং (Easier Debugging): প্রোগ্রামে কোনো সমস্যা দেখা দিলে পুরো কোড না খুঁজে শুধুমাত্র নির্দিষ্ট মডিউলটি পরীক্ষা করে ত্রুটি সহজে শনাক্ত ও সংশোধন করা যায়।

- ৫. নেমস্পেস ম্যানেজমেন্ট (Namespace Management): একই নামের ফাংশন বা ভেরিয়েবল আলাদা মডিউলে থাকলে তাদের মধ্যে সংঘর্ষ (Conflict) হয় না। ফলে প্রোগ্রামাররা নিশ্চিন্তে তাদের পছন্দমতো নাম ব্যবহার করতে পারেন।

- ৬. মেমোরি সাশ্রয়: পুরো প্যাকেজ লোড না করে শুধুমাত্র প্রয়োজনীয় মডিউলটি import করে কাজ করা যায়, যা সিস্টেমের মেমোরি ও পারফরম্যান্স উন্নত করে।

# অধ্যায় ০৪ অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং- এর মৌলিক ধারণা

এই অধ্যায়ের প্রধান সূচিপত্র (contants)

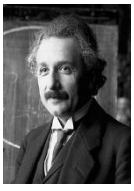

| বিষয় | পৃষ্ঠা নং |
|---|---|
| ❖ সূত্রাবলী ও শর্টকার্ট টেকনিক | - |
| ❖ অতি সংক্ষিপ্ত প্রশ্নাবলি | ২৭ |
| ❖ সংক্ষিপ্ত প্রশ্নাবলি | ২৯ |
| ❖ রচনামূলক প্রশ্নাবলি | ৩২ |


## বোর্ড প্রশ্নাবলির বিশ্লেষন (Board Questions Analysis)

এই অধ্যায় থেকে বিগত তিন বছরের ডিপ্লোমা ইন ইঞ্জিনিয়ারিং, ডিপ্লোমা ইন টেক্সটাইল, ডিপ্লোমা ইন এগ্রিকালচার, ডিপ্লোমা ইন মেরিন পরীক্ষায় মোট প্রশ্ন....নিচে 'Board Questions Analysis' অংশে এই অধ্যায় থেকে কোন সালে কোন বোর্ডে কতটি প্রশ্ন হয়েছে তা দেখানো হলো।

| সাল / বোর্ড | অতিসংক্ষিপ্ত | সংক্ষিপ্ত | রচনামূলক |
|---|---|---|---|
| ২০২৪ | ১টি | | ১টি |
| ২০২৩ | ২টি | ২টি | ২টি |
| ২০২২ | ৪টি | ১টি | -- |

" জিবন সাইকেল চালানোর মতো
ভারসাম্য রাখতে হলে চলতে থাকতেই হবে "



আলবার্ট আইনস্টাইন

## অতি সংক্ষিপ্ত প্রশ্ন

**১. OOP বলতে কী বুঝায়?**

**উত্তরঃ** OOP-এর পূর্ণরূপ হলো Object-Oriented Programming। এটি এমন একটি প্রোগ্রামিং পদ্ধতি যা ডেটা এবং ফাংশনকে একটি একক ইউনিটের (যাকে অবজেক্ট বলা হয়) মধ্যে সাজিয়ে প্রোগ্রাম তৈরি করে। এটি বাস্তব জগতের বস্তুর ধারণা ব্যবহার করে জটিল সমস্যার সমাধান সহজ করে।

**২. OOP – এর বৈশিষ্ট্যগুলো লেখ।**

**উত্তরঃ** OOP-এর প্রধান চারটি বৈশিষ্ট্য হলো:

- Encapsulation (এনক্যাপসুলেশন)
- Inheritance (ইনহেরিটেন্স)
- Polymorphism (পলিমরফিজম)
- Abstraction (অ্যাবস্ট্রাকশন)

**৩. ক্লাস কী?** [বাকাশিবো-২০২৩ (পরি), ২০২৪ (অনুষ্ঠিত-২৫)]

অথবা, **ক্লাস বলতে কি বুঝায়?** [বাকাশিবো-২০২৩(অনুষ্ঠিত-২৪)]

**উত্তরঃ** ক্লাস হলো অবজেক্ট তৈরির একটি ব্লু-প্রিন্ট (Blueprint) বা নকশা। একটি ক্লাসের মধ্যে অবজেক্টের বৈশিষ্ট্য (Variables) এবং কাজ করার নিয়ম (Methods) বর্ণিত থাকে।

**৪. পাইথন ল্যাংগুয়েজ এর ক্লাস তৈরির নিয়ম লেখ।** [বাকাশিবো-২০২২(অনুষ্ঠিত-২৩)]

**উত্তরঃ** পাইথনে ক্লাস তৈরির নিয়মগুলো হলো:

- class কীওয়ার্ড ব্যবহার করতে হয়।
- ক্লাসের নামের প্রথম অক্ষর সাধারণত বড় হাতের হয়।
- ক্লাসের নাম লিখে কোলন (:) চিহ্ন দিতে হয়।
- ক্লাসের ভেতরের কোডগুলো ইনডেন্টেশন (সঠিক দূরত্ব) বজায় রেখে লিখতে হয়।

**৫. ম্যাজিক মেথড বলতে কী বুঝায় ?**

**উত্তরঃ** পাইথনে যে সকল মেথডের নামের শুরুতে এবং শেষে দুটি করে আন্ডারস্কোর থাকে (যেমন: __init__, __str__), তাদের ম্যাজিক মেথড বলে। এগুলো বিশেষ পরিস্থিতিতে স্বয়ংক্রিয়ভাবে কাজ করে।

**৬. পাইথন প্রোগ্রামিং এ অবজেক্ট তৈরির নিয়ম লেখ।**

**উত্তরঃ** পাইথনে অবজেক্ট তৈরির নিয়ম হলো—প্রথমে একটি ভেরিয়েবল নাম লিখে সমান চিহ্নের পর ক্লাসের নাম এবং ছোট বন্ধনী দিতে হয়। যেমন: obj = ClassName()

**৭. কনস্ট্রাক্ট মেথড কী?**

**উত্তরঃ** কনস্ট্রাক্টর হলো এমন একটি মেথড যা ক্লাসের অবজেক্ট তৈরি হওয়ার সাথে সাথে নিজে থেকেই কাজ শুরু করে। পাইথনে __init__() মেথডকে কনস্ট্রাক্টর বলা হয়।

**৮. ডেস্ট্রাক্টর মেথড কাকে বলে?** [বাকাশিবো-২০২২(অনুষ্ঠিত-২৩)]

**উত্তরঃ** যখন কোনো অবজেক্টের কাজ শেষ হয়ে যায় এবং সেটি মেমোরি থেকে মুছে ফেলার প্রয়োজন হয়, তখন যে মেথড কাজ করে তাকে ডেস্ট্রাক্টর বলে। পাইথনে এটি __del__() মেথড।

**৯. অবজেক্ট কী?**

**উত্তরঃ** অবজেক্ট হলো ক্লাসের একটি বাস্তব রূপ বা ইনস্ট্যান্স (Instance)। যার নিজস্ব কিছু বৈশিষ্ট্য এবং কাজ করার ক্ষমতা থাকে।

**১০. অবজেক্ট ডিক্লারেশন ফরম্যাট লেখ।**

**উত্তরঃ** object_name = class_name()

**১১. ক্লাস ডিক্লারেশন ফরম্যাট লেখ।**

**উত্তরঃ** python class ClassName: # attributes and methods

**১২. মেথড কি?** [বাকাশিবো-২০২২(অনুষ্ঠিত-২৩)]

**উত্তরঃ** অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিংয়ে (OOP), কোনো ক্লাসের অভ্যন্তরে যে ফাংশনগুলো সংজ্ঞায়িত করা হয়, সেগুলোকে মেথড (Method) বলা হয়।

**১৩. পাইথন কোন লেভেল প্রোগ্রামিং ল্যাঙ্গুয়েজ?** [বাকাশিবো-২০২২(অনুষ্ঠিত-২৩)]

**উত্তরঃ** পাইথন একটি হাই-লেভেল (High-level) প্রোগ্রামিং ল্যাঙ্গুয়েজ।

## সংক্ষিপ্ত প্রশ্ন

**১. OOP এর গুরুত্ব লেখ।**

**উত্তরঃ** অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং (OOP) আধুনিক সফটওয়্যার ডেভেলপমেন্টে অত্যন্ত গুরুত্বপূর্ণ। এর গুরুত্ব নিচে আলোচনা করা হলো:

- কোড পুনরায় ব্যবহারযোগ্যতা (Reusability): ইনহেরিটেন্স ব্যবহারের মাধ্যমে একবার লেখা কোড বারবার ব্যবহার করা যায়, যা সময় ও শ্রম সাশ্রয় করে।
- সহজ রক্ষণাবেক্ষণ (Easy Maintenance): প্রোগ্রামের কোনো অংশে পরিবর্তন আনলে পুরো কোডে তার প্রভাব পড়ে না, ফলে সফটওয়্যার আপডেট বা রক্ষণাবেক্ষণ সহজ হয়।
- ডেটা নিরাপত্তা (Data Security): এনক্যাপসুলেশন বৈশিষ্ট্যের মাধ্যমে গুরুত্বপূর্ণ ডেটা গোপন রাখা যায়, যা অনাকাঙ্ক্ষিত পরিবর্তন থেকে রক্ষা করে।
- জটিলতা হ্রাস: বড় এবং জটিল প্রোগ্রামগুলোকে ছোট ছোট অবজেক্টে ভাগ করে কাজ করা হয় বলে প্রোগ্রামের জটিলতা কমে যায়।
- টিম ওয়ার্ক: আলাদা আলাদা অবজেক্ট নিয়ে বিভিন্ন প্রোগ্রামার একসাথে কাজ করতে পারেন, যা প্রজেক্টের গতি বাড়িয়ে দেয়।

**২. OOP – এর বৈশিষ্ট্যগুলো লেখ।**

**উত্তরঃ** OOP-এর প্রধান বৈশিষ্ট্যগুলো নিচে বর্ণনা করা হলো:

- অবজেক্ট (Object): যার নির্দিষ্ট বৈশিষ্ট্য (Attribute) এবং কাজ (Method) আছে।
- ক্লাস (Class): অবজেক্ট তৈরির ব্লু-প্রিন্ট বা নকশা।
- এনক্যাপসুলেশন (Encapsulation): ডেটা এবং মেথডকে একটি ইউনিটে আবদ্ধ করা এবং বাইরের হস্তক্ষেপ থেকে রক্ষা করা।
- ইনহেরিটেন্স (Inheritance): একটি ক্লাসের বৈশিষ্ট্য অন্য ক্লাসে স্থানান্তর করা।
- পলিমরফিজম (Polymorphism): একই মেথড বা ফাংশন বিভিন্ন পরিস্থিতিতে ভিন্ন ভিন্ন কাজ করার ক্ষমতা। অ্যাবস্ট্রাকশন (Abstraction): অপ্রয়োজনীয় জটিলতা লুকিয়ে শুধু প্রয়োজনীয় অংশ ব্যবহারকারীর সামনে উপস্থাপন করা।

**৩. পাইথনে ক্লাস তৈরির পদ্ধতি লেখ।**

**উত্তরঃ** বাকাশিবো (BTEB) পরীক্ষার রচনামূলক প্রশ্নের মান ও কাঠামো অনুযায়ী উত্তরগুলো নিচে বিস্তারিতভাবে দেওয়া হলো:

```
class ClassName:
    # ভেরিয়েবল বা এট্রিবিউট
    attribute_name = value
    # কনস্ট্রাক্টর মেথড
    def __init__(self, parameters):
        self.parameters = parameters
    # মেথড বা কাজ
    def method_name(self):
        # কোড ব্লক
```

**৪. অবজেক্ট কী? উদাহারণসহ লেখ।**

**উত্তরঃ** অবজেক্ট (Object): অবজেক্ট হলো ক্লাসের একটি বাস্তব রূপ বা উদাহরণ (Instance)। ক্লাস যদি হয় একটি নকশা, তবে সেই নকশা অনুযায়ী তৈরি করা বাস্তব জিনিসটিই হলো অবজেক্ট। প্রতিটি অবজেক্টের নিজস্ব ডেটা বা বৈশিষ্ট্য থাকে।

উদাহরণ: ধরা যাক, Car একটি ক্লাস। এই ক্লাসের নকশা ব্যবহার করে যখন আমরা একটি নির্দিষ্ট গাড়ি তৈরি করি, তখন সেটি হয় অবজেক্ট।

```
class Car:

    def __init__(self, color, model):

        self.color = color

        self.model = model

# এখানে 'my_car' হলো Car ক্লাসের একটি অবজেক্ট

my_car = Car("Red", "Toyota")

print(my_car.color) # আউটপুট: Red
```

এখানে my_car হলো একটি অবজেক্ট যার বৈশিষ্ট্য হলো রং লাল (Red) এবং মডেল টয়োটা (Toyota)।

**৫. অ্যাবস্ট্রাকশন কী? অ্যাবস্ট্রাকশন সম্পর্কে সংক্ষেপে লেখ।** [বাকাশিবো-২০২৩ (অনুষ্ঠিত-২৪)]

অথবা, **Abstraction কী? উদাহরণ দাও।** [বাকাশিবো-২০২৩ (পরি), ২০২৪ (অনুষ্ঠিত-২৫)]

**উত্তর:** অ্যাবস্ট্রাকশন হচ্ছে ক্লাসের অভ্যন্তরে ডাটা এবং ফাংশনকে ক্লাসের বাহির হতে অবাঞ্ছিত অ্যাক্সেস হতে সুরক্ষা প্রদান করা। নিচের প্রোগ্রামের Dog ক্লাসের মেম্বার নয় এরূপ অবজেক্টের মাধ্যমে Dog ক্লাসের ফাংশনসমূহ অ্যাক্সেস দিবে না।

প্রোগ্রাম

```
class Dog:
        def __init__(self, name):
          self.name = name
          print(self.name + " was adopted.")
 def bark(self):
     print("woof!")
spot = Dog("spot")
spot.bark()
```

**৬. if ... else-এর Syntax গুরুত্ব** লেখ। [বাকাশিবো-২০২২ (অনুষ্ঠিত-২৩)]

**উত্তর:**

```
if ... else-এর Syntax হলো:
if Condition:
     statements
else:
     statements
```

## রচনামূলক প্রশ্ন

**১. উদাহারণসহ ক্লাস ও অবজেক্ট বর্ণনা কর।**

**উত্তর:** ক্লাস (Class): ক্লাস হলো অবজেক্ট তৈরির একটি ব্লু-প্রিন্ট বা নকশা। এতে অবজেক্টের বৈশিষ্ট্য (Variables) এবং কার্যাবলি (Methods) সংজ্ঞায়িত থাকে। ক্লাস নিজে কোনো মেমোরি দখল করে না।

অবজেক্ট (Object): অবজেক্ট হলো ক্লাসের একটি বাস্তব রূপ বা ইনস্ট্যান্স (Instance)। ক্লাসের নকশা অনুযায়ী যখন কোনো ভেরিয়েবল তৈরি করা হয়, তাকে অবজেক্ট বলে।

উদাহরণ: ধরা যাক, Mobile একটি ক্লাস। এই ক্লাসের বৈশিষ্ট্য হতে পারে ব্র্যান্ড, মডেল এবং কাজ হতে পারে কল করা।

- ক্লাস: Mobile
- অবজেক্ট: Samsung S23, iPhone 15 (এগুলো বাস্তব মোবাইল)।

```
class Mobile:
    def __init__(self, brand, price):
        self.brand = brand
        self.price = price

# অবজেক্ট তৈরি
my_phone = Mobile("Samsung", 50000)
```

**২. ক্লাস এর মৌলিক স্ট্রাকচার বর্ণনা কর।**

**উত্তর:** পাইথনে একটি ক্লাসের মৌলিক কাঠামোতে নিচের অংশগুলো থাকে:

Class Definition: class কীওয়ার্ড দিয়ে ক্লাসের নাম ঘোষণা করা হয়। ২. Attributes (Variables): ক্লাসের ভেতরে ডেটা সংরক্ষণের জন্য ভেরিয়েবল থাকে। ৩. Constructor (init): এটি একটি বিশেষ মেথড যা অবজেক্ট তৈরির সময় স্বয়ংক্রিয়ভাবে রান করে এবং ভেরিয়েবলগুলোকে ইনিশিয়াল করে। ৪. Methods: ক্লাসের ভেতরে বিভিন্ন কাজ করার জন্য ফাংশন বা মেথড থাকে।

কাঠামো (Structure):

```
class ClassName:

    # ১. ক্লাস ভেরিয়েবল (ঐচ্ছিক)

    # ২. কনস্ট্রাক্টর

    def __init__(self, p1, p2):

        self.p1 = p1

        self.p2 = p2

    # ৩. মেথড

    def my_method(self):

        # কোড
```

**৩. ফাংশন ব্যবহার করে তিনটি সংখ্যার মধ্যে বৃহত্তম সংখ্যা নির্ণয়ের প্রোগ্রাম লেখ** [বাকাশিবো-২০২৩ (অনুষ্ঠিত-২৪)]

**উত্তর:**

```
def find_largest(a, b, c):
    if (a >= b) and (a >= c):
        return a
    elif (b >= a) and (b >= c):
        return b
    else:
        return c
# ইনপুট গ্রহণ
num1 = float(input("প্রথম সংখ্যা: "))
num2 = float(input("দ্বিতীয় সংখ্যা: "))
num3 = float(input("তৃতীয় সংখ্যা: "))
largest = find_largest(num1, num2, num3)
print(f"বৃহত্তম সংখ্যাটি হলো: {largest}")
```

**৪. ক্লাস ব্যবহার করে দ্বিঘাত সমীকরণের মূল নির্ণয়ের প্রোগ্রাম লেখ**

[বাকাশিবো-২০২৩ (পরি), ২০২৪ (অনুষ্ঠিত-২৫)]

**উত্তর:** দ্বিঘাত সমীকরণ $ax^2 + bx + c = 0$ এর মূল নির্ণয়ের জন্য $D = b^2 - 4ac$ (নিশ্চায়ক) ব্যবহার করা হয়।

```
import cmath # জটিল সংখ্যার হিসাবের জন্য
class QuadraticEquation:
    def __init__(self, a, b, c):
        self.a = a
        self.b = b
        self.c = c
    def find_roots(self):
        # নিশ্চায়ক (Discriminant) নির্ণয়
        d = (self.b**2) - (4*self.a*self.c)
        # মূলদ্বয় নির্ণয়
        root1 = (-self.b - cmath.sqrt(d)) / (2 * self.a)
        root2 = (-self.b + cmath.sqrt(d)) / (2 * self.a)
        return root1, root2
# ইনপুট ও অবজেক্ট তৈরি
a = float(input("a এর মান: "))
b = float(input("b এর মান: "))
c = float(input("c এর মান: "))
equation = QuadraticEquation(a, b, c)
r1, r2 = equation.find_roots()

print(f"মূল দুটি হলো: {r1} এবং {r2}")
```

# অধ্যায় ০৫ অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং- এর ফোর পিলার

এই অধ্যায়ের প্রধান সূচিপত্র (contants)

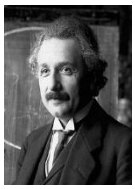

| বিষয় | পৃষ্ঠা নং |
|---|---|
| ❖ সূত্রাবলী ও শর্টকার্ট টেকনিক | - |
| ❖ অতি সংক্ষিপ্ত প্রশ্নাবলি | ৩৬ |
| ❖ সংক্ষিপ্ত প্রশ্নাবলি | ৩৭ |
| ❖ রচনামূলক প্রশ্নাবলি | ৩৯ |


## বোর্ড প্রশ্নাবলির বিশ্লেষন (Board Questions Analysis)

এই অধ্যায় থেকে বিগত তিন বছরের ডিপ্লোমা ইন ইঞ্জিনিয়ারিং, ডিপ্লোমা ইন টেক্সটাইল, ডিপ্লোমা ইন এগ্রিকালচার, ডিপ্লোমা ইন মেরিন পরীক্ষায় মোট প্রশ্ন....নিচে 'Board Questions Analysis' অংশে এই অধ্যায় থেকে কোন সালে কোন বোর্ডে কতটি প্রশ্ন হয়েছে তা দেখানো হলো।

| সাল / বোর্ড | অতিসংক্ষিপ্ত | সংক্ষিপ্ত | রচনামূলক |
|---|---|---|---|
| ২০২৪ | ১টি | ১টি | -- |
| ২০২৩ | ২টি | ২টি | ২টি |
| ২০২২ | ২টি | ২টি | ২টি |

" জিবন সাইকেল চালানোর মতো
ভারসাম্য রাখতে হলে চলতে থাকতেই হবে "



আলবার্ট আইনস্টাইন

## অতি সংক্ষিপ্ত প্রশ্ন

**১. ইনহেরিট্যান্স কত প্রকার ও কী কী?**

**উত্তর:** পাইথনে ইনহেরিট্যান্স প্রধানত ৫ প্রকার। যথা:

১. সিঙ্গেল ইনহেরিট্যান্স (Single Inheritance)

২. মাল্টিপল ইনহেরিট্যান্স (Multiple Inheritance)

৩. মাল্টিলেভেল ইনহেরিট্যান্স (Multilevel Inheritance)

৪. হায়ারার্কিক্যাল ইনহেরিট্যান্স (Hierarchical Inheritance)

৫. হাইব্রিড ইনহেরিট্যান্স (Hybrid Inheritance)

**২. অ্যাবট্রাকশন এর Syntax লেখ।**

**উত্তর:** পাইথনে অ্যাবস্ট্রাকশন ব্যবহারের জন্য abc মডিউল থেকে ABC এবং abstractmethod ইমপোর্ট করতে হয়। Syntax:

```
from abc import ABC, abstractmethod


class ClassName(ABC):
    @abstractmethod
    def method_name(self):
        pass
```

**৩. OOP – এর ৪ টি স্তম্ভের নাম লেখ।**

**উত্তর:** OOP-এর ৪টি স্তম্ভ বা মূল ভিত্তি হলো:

১. ইনহেরিট্যান্স (Inheritance)

২. পলিমরফিজম (Polymorphism)

৩. এনক্যাপসুলেশন (Encapsulation)

৪. অ্যাবস্ট্রাকশন (Abstraction)

**৪. ইনহেরিট্যান্স কী** [বাকাশিবো-২০২২ (অনুষ্ঠিত-২৩), [বাকাশিবো-২০২৩ (অনুষ্ঠিত-২৪) ২০২৪ (অনুষ্ঠিত-২৫)]

**উত্তর:** যে প্রক্রিয়ায় একটি ক্লাস (Child Class) অন্য একটি ক্লাসের (Parent Class) বৈশিষ্ট্য ও মেথডগুলো নিজের মধ্যে ধারণ বা গ্রহণ করতে পারে, তাকে ইনহেরিট্যান্স বলে। এর মাধ্যমে কোডের পুনঃব্যবহারযোগ্যতা বৃদ্ধি পায়।

**৫. পলিমারফিজম কী?** [বাকাশিবো-২০২২ (অনুষ্ঠিত-২৩), [বাকাশিবো-২০২৩ (অনুষ্ঠিত-২৪)]

**উত্তর:** পলিমরফিজম' শব্দের অর্থ 'বহুরূপতা'। প্রোগ্রামিংয়ের ভাষায়, একই নামের ফাংশন বা মেথড যখন ভিন্ন ভিন্ন পরিস্থিতিতে ভিন্ন ভিন্ন কাজ সম্পাদন করে, তখন তাকে পলিমরফিজম বলে।

**৬. এনক্যাপসুলেশন কী?**

**উত্তর:** ডেটা এবং সেই ডেটা নিয়ে কাজ করা মেথডগুলোকে একটি একক ইউনিটে (ক্লাস) আবদ্ধ করার প্রক্রিয়াকে এনক্যাপসুলেশন বলে। এটি ডেটাকে বাইরের অনাকাঙ্ক্ষিত হস্তক্ষেপ থেকে সুরক্ষা দেয়।

**৭. হায়ারিক্যাল ইনহেরিট্যান্স কী?**

**উত্তর:** যখন একটি মাত্র প্যারেন্ট ক্লাস (Parent Class) থেকে একাধিক চাইল্ড ক্লাস (Child Class) বৈশিষ্ট্য গ্রহণ করে, তখন তাকে হায়ারার্কিক্যাল ইনহেরিট্যান্স বলে।

**৮. সিঙ্গেল ইনহেরিট্যান্স কী?**

**উত্তর:** যখন একটি চাইল্ড ক্লাস শুধুমাত্র একটি প্যারেন্ট ক্লাস থেকে বৈশিষ্ট্য ইনহেরিট বা গ্রহণ করে, তখন তাকে সিঙ্গেল ইনহেরিট্যান্স বলে।

**৯. সেলফ কী?**

**উত্তর:** পাইথনে self হলো একটি রেফারেন্স ভেরিয়েবল যা ক্লাসের বর্তমান অবজেক্টকে নির্দেশ করে। মেথডের ভেতরে ক্লাসের ভেরিয়েবল বা মেথডগুলোকে এক্সেস করার জন্য এটি ব্যবহার করা হয়।

**১০. মাল্টিলেভেল ইনহেরিট্যান্স কী?**

**উত্তর:** যখন একটি ক্লাস অন্য একটি চাইল্ড ক্লাস থেকে বৈশিষ্ট্য গ্রহণ করে (অর্থাৎ একটি ক্লাস অন্য একটি ক্লাসের চাইল্ড এবং নিজেই আবার অন্য ক্লাসের প্যারেন্ট), তখন তাকে মাল্টিলেভেল ইনহেরিট্যান্স বলে। যেমন: দাদাজান -> বাবা -> সন্তান।

## সংক্ষিপ্ত প্রশ্ন

**১. মাল্টিলেভেল ইনহেরিট্যান্স ব্লক চিত্রে আঁক।**

**উত্তর:** মাল্টিলেভেল ইনহেরিট্যান্স: যখন একটি ক্লাস অন্য একটি চাইল্ড ক্লাস থেকে বৈশিষ্ট্য গ্রহণ করে (অর্থাৎ একটি ক্লাস অন্য একটি ক্লাসের চাইল্ড এবং নিজেই আবার অন্য ক্লাসের প্যারেন্ট), তখন তাকে মাল্টিলেভেল ইনহেরিট্যান্স বলে।

ব্লক চিত্র:

**২. OPP – এর ৪ পিলার লেখ।** [বাকাশিবো-২০২২ (অনুষ্ঠিত-২৩), ২০২৪ (অনুষ্ঠিত-২৫)]

**উত্তর:** অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং (OOP)-এর ৪টি প্রধান পিলার বা স্তম্ভ হলো:

১. ইনহেরিট্যান্স (Inheritance): এক ক্লাসের বৈশিষ্ট্য অন্য ক্লাসে স্থানান্তর।

২. এনক্যাপসুলেশন (Encapsulation): ডেটা এবং মেথডকে একটি ইউনিটে আবদ্ধ করা।

৩. পলিমরফিজম (Polymorphism): একই মেথডের ভিন্ন ভিন্ন রূপ বা কাজ।

৪. অ্যাবস্ট্রাকশন (Abstraction): জটিলতা লুকিয়ে কেবল প্রয়োজনীয় অংশ প্রদর্শন করা।

**৩. ইনহেরিট্যান্স বলতে কী বুঝায়? এর প্রকারভেদ লেখ**

**উত্তর:** ইনহেরিট্যান্স: যে প্রক্রিয়ায় একটি নতুন ক্লাস (Child/Derived Class) এক বা একাধিক বিদ্যমান ক্লাসের (Parent/Base Class) বৈশিষ্ট্য ও মেথডগুলো নিজের মধ্যে গ্রহণ করতে পারে, তাকে ইনহেরিট্যান্স বলে। এর মাধ্যমে কোডের পুনঃব্যবহারযোগ্যতা নিশ্চিত হয়।

ইনহেরিট্যান্সের প্রকারভেদ:

১. সিঙ্গেল ইনহেরিট্যান্স: একটি প্যারেন্ট ক্লাস থেকে একটি চাইল্ড ক্লাস তৈরি হয়।

২. মাল্টিপল ইনহেরিট্যান্স: একাধিক প্যারেন্ট ক্লাস থেকে একটি চাইল্ড ক্লাস তৈরি হয়।

৩. মাল্টিলেভেল ইনহেরিট্যান্স: একটি চাইল্ড ক্লাস থেকে অন্য একটি চাইল্ড ক্লাস তৈরি হয় (পর্যায়ক্রমিক)।

৪. হায়ারার্কিক্যাল ইনহেরিট্যান্স: একটি প্যারেন্ট ক্লাস থেকে একাধিক চাইল্ড ক্লাস তৈরি হয়।

৫. হাইব্রিড ইনহেরিট্যান্স: উপরের একাধিক ইনহেরিট্যান্সের সমন্বয়।

**৪. ৪ পিলারের গুরুত্ব ব্যাখ্যা কর।** [বাকাশিবো-২০২৩ (অনুষ্ঠিত-২৪), ২২, ২০২৩ (পরি)]

**উত্তর:** OOP-এর চারটি পিলারের গুরুত্ব নিচে আলোচনা করা হলো:

- ইনহেরিট্যান্স-এর গুরুত্ব: এর ফলে কোড বারবার লিখতে হয় না (Reusability)। একটি বেস ক্লাসে কোনো পরিবর্তন করলে তা স্বয়ংক্রিয়ভাবে তার সকল চাইল্ড ক্লাসে কার্যকর হয়, যা সময় বাঁচায়।

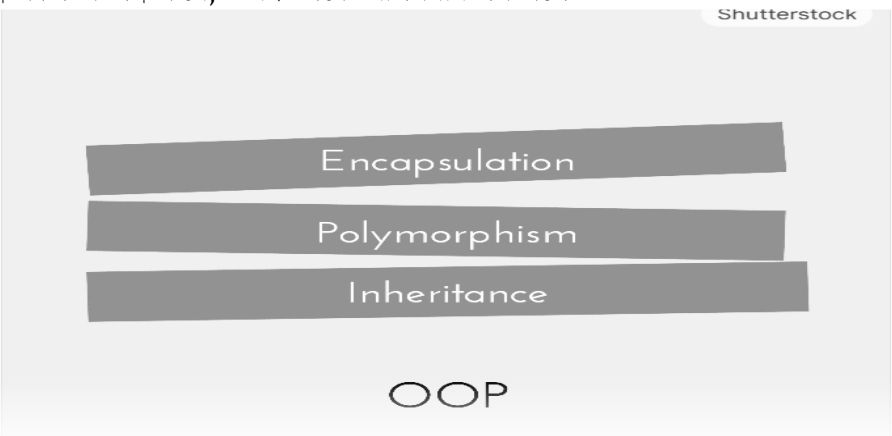
- এনক্যাপসুলেশন-এর গুরুত্ব: এটি ডেটাকে বাইরের হস্তক্ষেপ বা ভুল ব্যবহার থেকে রক্ষা করে। এটি প্রোগ্রামের নিরাপত্তা বাড়ায় এবং ডেটা হাইডিং (Data Hiding) নিশ্চিত করে।
- পলিমরফিজম-এর গুরুত্ব: এটি প্রোগ্রামকে আরও নমনীয় করে তোলে। একই নামের মেথড ভিন্ন ভিন্ন অবজেক্টের জন্য ভিন্নভাবে কাজ করতে পারে, ফলে কোড পড়া সহজ হয় এবং রক্ষণাবেক্ষণ সুবিধা বাড়ে।
- অ্যাবস্ট্রাকশন-এর গুরুত্ব: এটি ব্যবহারকারীর কাছ থেকে অভ্যন্তরীণ জটিল কাজের প্রক্রিয়া লুকিয়ে রাখে এবং শুধুমাত্র জরুরি ইন্টারফেস প্রদান করে। এটি প্রোগ্রামের জটিলতা কমায় এবং কাজের গতি বাড়ায়।

## রচনামূলক প্রশ্ন

**১. প্রোগ্রামসহ পলিমরফিজম পদ্ধতি বর্ণনা কর।** [বাকাশিবো-২০২৩ (পরি))

অথবা, **Polymorphism ব্যবহার করে Python Pragram লেখ।** [বাকাশিবো-২০২২(অনুষ্ঠিত-২৩)]

**উত্তর:** পলিমরফিজম (Polymorphism): 'Poly' শব্দের অর্থ বহু এবং 'Morph' শব্দের অর্থ রূপ। অর্থাৎ পলিমরফিজম মানে হলো "বহুরূপতা"। অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিংয়ে যখন একই নামের মেথড বা ফাংশন ভিন্ন ভিন্ন ক্লাসে ভিন্ন ভিন্ন কাজ সম্পন্ন করে, তখন তাকে পলিমরফিজম বলে।




পলিমরফিজম এর পাইথন প্রোগ্রাম (Method Overriding): নিচে একটি উদাহরণ দেওয়া হলো যেখানে make_sound() মেথডটি ভিন্ন ভিন্ন ক্লাসে (Dog এবং Cat) ভিন্ন ভিন্ন আউটপুট দিচ্ছে।

```
class Animal:
    def make_sound(self):
        print("প্রাণী শব্দ করে")
```

```
class Dog(Animal):
    def make_sound(self):
        print("ঘেউ ঘেউ (Bark)")

class Cat(Animal):
    def make_sound(self):
        print("মিউ মিউ (Meow)")

# অবজেক্ট তৈরি
animal = Animal()
dog = Dog()
cat = Cat()

# একই মেথড কল করা হচ্ছে কিন্তু ফলাফল ভিন্ন
animal.make_sound()
dog.make_sound()
cat.make_sound()
```

**আউটপুট:**

```
প্রাণী শব্দ করে
ঘেউ ঘেউ (Bark)
মিউ মিউ (Meow)
```

**২। ইনহেরিট্যান্স কী? বিস্তারিত বর্ণনা কর।** [বাকাশিবো-২০২৩ (পরি)]

অথবা, **Super Class ও Subclass ব্যবহার করে Python program লেখ।**

**উত্তর:** ইনহেরিট্যান্স (Inheritance): ইনহেরিট্যান্স হলো এমন একটি প্রক্রিয়া যার মাধ্যমে একটি ক্লাস (Child Class) অন্য একটি বিদ্যমান ক্লাসের (Parent Class) বৈশিষ্ট্য ও মেথডগুলো অর্জন করতে পারে। এটি কোডের পুনঃব্যবহারযোগ্যতা (Reusability) নিশ্চিত করে।

- Super Class (Parent Class): যে ক্লাস থেকে বৈশিষ্ট্য নেওয়া হয়।
- Sub Class (Child Class): যে ক্লাসটি বৈশিষ্ট্য গ্রহণ করে।

ইনহেরিট্যান্স এর পাইথন প্রোগ্রাম: এখানে Father হলো সুপার ক্লাস এবং Son হলো সাব ক্লাস।

```
# সুপার ক্লাস (Super Class)

class Father:

    def land(self):

        print("বাবার অনেক জমি আছে।")

# সাব ক্লাস (Sub Class)

class Son(Father):

    def money(self):

        print("ছেলের অনেক টাকা আছে।")

# সাব ক্লাসের অবজেক্ট তৈরি

myson = Son()

# সাব ক্লাস নিজের এবং বাবার মেথড এক্সেস করতে পারছে

myson.money()  # নিজের মেথড

myson.land()   # বাবার (Super Class) মেথড
```

**আউটপুট:**

```
ছেলের অনেক টাকা আছে।
```

বাবার অনেক জমি আছে।

**৩। "OOP-এর ৪ পিলার" সম্পর্কে বর্ণনা কর।** [বাকাশিবো-২০২২ (অনুষ্ঠিত-২৪)]

অথবা, **OOP-এর Feature-গুলো আলোচনা কর।**

অথবা, **OOP-এর বৈশিষ্ট্যগুলো আলোচনা কর।**

**উত্তর:** অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং (OOP) মূলত চারটি প্রধান স্তম্ভ বা পিলারের ওপর দাঁড়িয়ে আছে। এগুলো হলো:

- ১. ইনহেরিট্যান্স (Inheritance): এটি এমন একটি বৈশিষ্ট্য যার মাধ্যমে একটি ক্লাস অন্য ক্লাসের বৈশিষ্ট্য ও আচরণ উত্তরাধিকার সূত্রে পায়। এটি কোড ডুপ্লিকেশন কমায়।
- ২. এনক্যাপসুলেশন (Encapsulation): ডেটা (Variables) এবং মেথড (Methods) কে একটি সিঙ্গেল ইউনিটে বা ক্লাসে আবদ্ধ করে রাখার প্রক্রিয়া। এটি বাইরের ক্লাস থেকে ডেটার সরাসরি অ্যাক্সেস বন্ধ করে ডেটা সুরক্ষিত রাখে।
- ৩. পলিমরফিজম (Polymorphism): একই ইন্টারফেস বা মেথড ব্যবহার করে ভিন্ন ভিন্ন ধরনের কাজ করার ক্ষমতা। যেমন—একই add() ফাংশন দুটি সংখ্যা যোগ করতে পারে আবার দুটি স্ট্রিং জোড়া লাগাতে পারে।
- ৪. অ্যাবস্ট্রাকশন (Abstraction): প্রোগ্রামিংয়ের জটিল ইমপ্লিমেন্টেশন বা ভেতরের ক্রিয়াকলাপ ব্যবহারকারীর কাছ থেকে লুকিয়ে রাখা এবং শুধুমাত্র প্রয়োজনীয় অংশ প্রদর্শন করা। যেমন—গাড়ি চালানোর সময় আমরা শুধু স্টিয়ারিং ও ব্রেক দেখি, ইঞ্জিনের ভেতরের কাজ দেখি না।

**৪। প্রোগ্রামসহ এনক্যাপসুলেশন পদ্ধতি বর্ণনা কর।** [বাকাশিবো-২০২৩ (অনুষ্ঠিত-২৪)]

অথবা, **উদাহরণসহ এনক্যাপসুলেশন পদ্ধতি বর্ণন কর।**

**উত্তর:** এনক্যাপসুলেশন (Encapsulation): এনক্যাপসুলেশন হলো ডেটা হাইডিং বা ডেটা সুরক্ষার একটি পদ্ধতি। পাইথনে ক্লাসের ভেরিয়েবলগুলোকে private করার মাধ্যমে এনক্যাপসুলেশন বাস্তবায়ন করা হয়। ভেরিয়েবলের নামের আগে দুটি আন্ডারস্কোর (__) দিলে সেটি প্রাইভেট হয়ে যায় এবং ক্লাসের বাইরে থেকে সরাসরি এক্সেস করা যায় না।

**এনক্যাপসুলেশন এর পাইথন প্রোগ্রাম:**

```
class Student:
    def __init__(self):
        # প্রাইভেট ভেরিয়েবল (__ দিয়ে শুরু)
        self.__balance = 0
```

```
    # টাকা সেট করার জন্য পাবলিক মেথড (Setter)
    def set_balance(self, amount):

        if amount > 0:
            self.__balance = amount
            print(f"{amount} টাকা জমা হয়েছে।")
        else:
            print("টাকার পরিমাণ সঠিক নয়।")

    # টাকা দেখার জন্য পাবলিক মেথড (Getter)
    def get_balance(self):
        print(f"বর্তমান ব্যালেন্স: {self.__balance}")

# অবজেক্ট তৈরি
s = Student()

# মেথড ব্যবহার করে ব্যালেন্স সেট করা
s.set_balance(1000)

# মেথড ব্যবহার করে ব্যালেন্স দেখা
s.get_balance()

# সরাসরি প্রাইভেট ভেরিয়েবল এক্সেস করার চেষ্টা (এটি Error দিবে)
# print(s.__balance)  <-- এই লাইনটি কাজ করবে না
```

**আউটপুট:**

```
1000 টাকা জমা হয়েছে।

বর্তমান ব্যালেন্স: 1000
```

**ব্যাখ্যা:** ওপরের প্রোগ্রামে __balance ভেরিয়েবলটি সরাসরি পরিবর্তন করা যায় না। এটি পরিবর্তন করতে হলে অবশ্যই set_balance মেথড ব্যবহার করতে হবে, যা ডেটার সুরক্ষা নিশ্চিত করে।

# অধ্যায় ০৬ পাইথন ইটারেটর, জেনারেটর এবং ডেকোরেটর

এই অধ্যায়ের প্রধান সূচিপত্র (contants)


| বিষয় | পৃষ্ঠা নং |
|---|---|
| ❖ সূত্রাবলী ও শর্টকার্ট টেকনিক | - |
| ❖ অতি সংক্ষিপ্ত প্রশ্নাবলি | ৪৫ |
| ❖ সংক্ষিপ্ত প্রশ্নাবলি | ৪৬ |
| ❖ রচনামূলক প্রশ্নাবলি | ৪৬ |


## বোর্ড প্রশ্নাবলির বিশ্লেষন (Board Questions Analysis)

এই অধ্যায় থেকে বিগত তিন বছরের ডিপ্লোমা ইন ইঞ্জিনিয়ারিং, ডিপ্লোমা ইন টেক্সটাইল, ডিপ্লোমা ইন এগ্রিকালচার, ডিপ্লোমা ইন মেরিন পরীক্ষায় মোট প্রশ্ন....নিচে 'Board Questions Analysis' অংশে এই অধ্যায় থেকে কোন সালে কোন বোর্ডে কতটি প্রশ্ন হয়েছে তা দেখানো হলো।

| সাল / বোর্ড | অতিসংক্ষিপ্ত | সংক্ষিপ্ত | রচনামূলক |
|---|---|---|---|
| ২০২৪ | ২টি | -- | -- |
| ২০২৩ | ৩টি | -- | ১টি |
| ২০২২ | ১টি | -- | ১টি |

" প্রতিটি ক্রিয়ার বিপরীতে সমান
ও বিপরীত প্রতিক্রিয়া ঘটে "

আইজ্যাক নিউটন

## অতি সংক্ষিপ্ত প্রশ্ন

**১. Meta character কী?** [বাকাশিবো-২০২২(২৩)]

অথবা, **Meta character বলতে কী বুঝায়?** [বাকাশিবো-২০২৩(২৪)]

**উত্তর:** Meta character হলো এমন কতগুলো বিশেষ অক্ষর বা Character সেগুলোর পূর্বনির্ধারিত অর্থ আছে।

**২।ইটারেটর বলতে কী বুঝায়?** [বাকাশিবো-২০২৩ (২৪)]

**উত্তর:** Iterator এমন একটি অবজেক্ট, যা একটি গণনাযোগ্য সংখ্যক মান ধারণ করে এবং এটি লিস্ট, টাপল, সেট ইত্যাদির মতো ইটারেবল-এর উপর পুনরাবৃত্তি করতে ব্যবহৃত হয়। iter()-এর ভিতর যদি আমরা কোনো ইটারেবল অবজেক্টকে পাস করি তবে তা একটি ইটারেটর রিটার্ন করে। আর _next_() মেথড দিয়ে আমরা একটি ইটারেটরের পরবর্তী এলিমেন্ট বা আইটেমকে অ্যাক্সেস করতে পারি। কতগুলো ইলিমেন্টকে iter() ও next() মেথড ব্যবহার করে কাজ সম্পন্ন করাকে ইটারেটর বলে।

**৩। ডেকোরেটর বলতে কী বুঝায়?** [বাকাশিবো-২০২৩ (পরি), ২০২৪ (অনুষ্ঠিত-২৫)]

অথবা, **Decorator কী?** [বাকাশিবো-২০২২ (অনুষ্ঠিত-২৩)]

**উত্তর:** ডেকোরেটর হলো এমন একটি ফাংশন, যা অন্য কোনো ফাংশনের কার্যপরিধি কোনো প্রকার পরিবর্তন ছাড়াই এক্সটেন্ড করতে পারে। ডেকোরেটর লেখার সিনট্যাক্স হলো @decorator_name

ডেকোরেটর হচ্ছে সাধারণ রকমেরই একটা ফাংশন, যা অন্য আরেকটি ফাংশনকে মডিফাই করে তথা তার কাজকে বর্ধিত বা পরিবর্তিত করে। ডেকোরেটর পাইথনের একটি খুব শক্তিশালী এবং প্রয়োজনীয় টুল, কারণ এটি প্রোগ্রামারদের একটি ফাংশন বা ক্লাসের আচরণ পরিবর্তন করতে দেয়।

**৪। জেনারেটর কী?**

**উত্তর:** জেনারেটর একটি ফাংশন, এর কাজ হলো yield স্টেটমেন্ট ব্যবহার করে সিকুয়েন্স তৈরি করা। জেনারেটরও এক ধরনের ইটারেটর। যখন একটি জেনারেটর ফাংশন কল করা হয় তখন এটা ফাংশনের কোড এক্সিকিউট হবার আগেই একটি জেনারেটর অবজেক্ট রিটার্ন করে।

## সংক্ষিপ্ত প্রশ্ন

**১. ইটারেটর এর প্রিডিউলর কার্যনীতি সংক্ষেপে লেখ।**

**উত্তর:** ইটারেটর প্রিডিসেসর বা ব্যাকওয়ার্ড মুভমেন্টের মূল কাজ হলো ইটারেটরকে বর্তমান ডেটা এলিমেন্ট থেকে ঠিক আগের এলিমেন্টে সরিয়ে নেওয়া।

কার্যনীতি (Working Principle)

১. মেমোরি অ্যাড্রেস পরিবর্তন:

যখন ইটারেটরকে পেছনে নেওয়া হয় (Decrement করা হয়), তখন এটি অভ্যন্তরীণভাবে তার পয়েন্টার বা মেমোরি অ্যাড্রেস পরিবর্তন করে আগের নোড বা মেমোরি ব্লককে নির্দেশ করে।

২. বাইডাইরেকশনাল সক্ষমতা (Bidirectional Capability):

এই কাজটি করার জন্য ইটারেটরটিকে অবশ্যই Bidirectional Iterator বা Random Access Iterator হতে হয় (যেমন: Doubly Linked List বা Array)। 'Forward Iterator' (যেমন: Singly Linked List) পেছনে যেতে পারে না, তাই সেখানে এই নীতি কাজ করে না।

৩. অপারেটর ব্যবহার:

কোডিংয়ের ক্ষেত্রে সাধারণত ডিক্রিমেন্ট অপারেটর operator-- (যেমন: --it বা it--) অথবা std::prev() ফাংশন ব্যবহার করে এই কাজটি সম্পন্ন করা হয়।

গাণিতিক বা লজিক্যাল ধারণা

ধরি, একটি ইটারেটর $it$ বর্তমানে মেমোরি লোকেশন $X_n$-এ আছে। প্রিডিসেসর অপারেশন চালানোর পর:

$$it \leftarrow X_{n-1}$$

অর্থাৎ, এটি $n$ তম অবস্থান থেকে $(n-1)$ তম অবস্থানে সরে আসবে।

## রচনামূলক প্রশ্ন

**১. উদাহরণসহ ডেকোরেটর-এর প্রসিডিউর কার্যনীতি বর্ণনা কর।** [বাকাশিবো-২০২৩ (অনুষ্ঠিত-২৪)]

**উত্তর:** ডেকোরেটর (Decorator): ডেকোরেটর হলো এমন একটি ফাংশন যা অন্য একটি ফাংশনকে ইনপুট হিসেবে গ্রহণ করে এবং ওই ফাংশনের মূল কোড পরিবর্তন না করেই তার আচরণের সাথে নতুন কিছু বৈশিষ্ট্য যোগ করে। পাইথনে একে @ চিহ্ন দিয়ে প্রকাশ করা হয়।

কার্যনীতি (Procedure): ডেকোরেটরের কাজের ধাপগুলো নিচে দেওয়া হলো: ১. Higher Order Function: ডেকোরেটর একটি ফাংশনকে আর্গুমেন্ট হিসেবে নেয়। ২. Wrapper Function: ডেকোরেটরের ভেতরে একটি নতুন ফাংশন (Wrapper) তৈরি করা হয়। ৩. Behavior Extension: এই র্যাপার ফাংশনের ভেতরে মূল ফাংশনটি কল করার আগে বা পরে প্রয়োজনীয় কোড লেখা হয়। ৪. Return: সবশেষে ডেকোরেটরটি ওই র্যাপার ফাংশনটিকে রিটার্ন করে দেয়।

উদাহরণ (Python Code): ধরি, আমরা একটি ফাংশন তৈরি করব যা কোনো সংখ্যার ভাগফল বের করবে, কিন্তু ডেকোরেটর ব্যবহার করে চেক করব যেন ভাজক (divisor) শূন্য না হয়।

```
# ১. ডেকোরেটর ফাংশন
def smart_divide(func):
    def inner(a, b):
        print("ভাগ করার প্রস্তুতি চলছে...")
        if b == 0:
            print("দুঃখিত! ০ দিয়ে ভাগ করা যায় না।")
            return
        return func(a, b)
    return inner
# ২. মূল ফাংশনের উপর ডেকোরেটর প্রয়োগ (@ ব্যবহার করে)
@smart_divide
def divide(a, b):
    print(f"ভাগফল: {a / b}")
# ৩. ফাংশন কল
divide(10, 2)
divide(10, 0)
```

**ফলাফল:** প্রথম ক্ষেত্রে 10/2 ভাগ হবে। দ্বিতীয় ক্ষেত্রে 10/0 হওয়ার সময় ডেকোরেটরটি বাধা দেবে এবং এরর মেসেজ দেখাবে, কিন্তু মূল divide ফাংশনটি কল হবে না।

**২। Itegrator, Generator ও Decorator সম্পর্কে বিস্তারিত আলোচনা কর।**

[বাকাশিবো-২০২২ (অনুষ্ঠিত-২৩)]

**উত্তর:** ক) ইটারেটর (Iterator)

সংজ্ঞা: ইটারেটর হলো এমন একটি অবজেক্ট যা কোনো ডেটা কালেকশন (যেমন: List, Tuple) এর উপাদানগুলোকে এক এক করে ট্রাভার্স (Traverse) বা ভিজিট করতে পারে।

বৈশিষ্ট্য:

- এটি __iter__() এবং __next__() মেথড ব্যবহার করে কাজ করে।
- ইটারেটর মনে রাখে সে বর্তমানে কোন অবস্থানে আছে।
- মেমোরি ইফিশিয়েন্ট, কারণ এটি পুরো লিস্ট মেমোরিতে লোড না করে একবারে একটি ভ্যালু প্রসেস করে।

**উদাহরণ:**

```
list_data = [1, 2, 3]
it = iter(list_data) # ইটারেটর তৈরি
print(next(it))      # আউটপুট: 1
print(next(it))      # আউটপুট: 2
```

খ) জেনারেটর (Generator)

সংজ্ঞা: জেনারেটর হলো বিশেষ ধরণের ফাংশন যা সাধারণ ফাংশনের মতো return ব্যবহার না করে yield কি-ওয়ার্ড ব্যবহার করে ডেটা রিটার্ন করে। এটি নিজেই একটি ইটারেটর হিসেবে কাজ করে।

বৈশিষ্ট্য:

- এটি yield স্টেটমেন্ট ব্যবহার করে ভ্যালু রিটার্ন করে এবং ফাংশনের অবস্থা (state) সংরক্ষণ করে রাখে।
- পরবর্তী বার কল করলে ঠিক যেখানে শেষ হয়েছিল সেখান থেকে শুরু হয়।
- খুব বড় ডেটাসেট নিয়ে কাজ করার জন্য এটি সেরা, কারণ এটি অলসভাবে (Lazily) ডেটা জেনারেট করে।

**উদাহরণ:**

```
def my_generator():
    yield 1
    yield 2
    yield 3

gen = my_generator()
```

```
print(next(gen)) # আউটপুট: 1
```

**গ) ডেকোরেটর (Decorator)**

সংজ্ঞা: ১ নং প্রশ্নে বিস্তারিত বলা হয়েছে। সংক্ষেপে, এটি মেটা-প্রোগ্রামিংয়ের একটি অংশ যা অন্য ফাংশনের কার্যক্ষমতা বৃদ্ধি করে।

বৈশিষ্ট্য:

- এটি কোড রিইউজবিলিটি (Reusability) বাড়ায়।
- লগিং (Logging), অথেন্টিকেশন (Authentication) বা টাইমিং চেক করার জন্য এটি ব্যাপকভাবে ব্যবহৃত হয়।

# অধ্যায় ০৭ পাইথন- এর এক্সসেপশন এবং এরর হ্যান্ডেলিং

এই অধ্যায়ের প্রধান সূচিপত্র (contants)


| বিষয় | পৃষ্ঠা নং |
|---|---|
| ❖ সূত্রাবলী ও শর্টকার্ট টেকনিক | - |
| ❖ অতি সংক্ষিপ্ত প্রশ্নাবলি | ৫১ |
| ❖ সংক্ষিপ্ত প্রশ্নাবলি | ৫২ |
| ❖ রচনামূলক প্রশ্নাবলি | ৫৩ |


## বোর্ড প্রশ্নাবলির বিশ্লেষন (Board Questions Analysis)

এই অধ্যায় থেকে বিগত তিন বছরের ডিপ্লোমা ইন ইঞ্জিনিয়ারিং, ডিপ্লোমা ইন টেক্সটাইল, ডিপ্লোমা ইন এগ্রিকালচার, ডিপ্লোমা ইন মেরিন পরীক্ষায় মোট প্রশ্ন....নিচে 'Board Questions Analysis' অংশে এই অধ্যায় থেকে কোন সালে কোন বোর্ডে কতটি প্রশ্ন হয়েছে তা দেখানো হলো।

| সাল / বোর্ড | অতিসংক্ষিপ্ত | সংক্ষিপ্ত | রচনামূলক |
|---|---|---|---|
| ২০২৪ | ১টি | ১টি | -- |
| ২০২৩ | ৩টি | ২টি | -- |
| ২০২২ | ১টি | ১টি | ১টি |

" প্রতিটি ক্রিয়ার বিপরীতে সমান
ও বিপরীত প্রতিক্রিয়া ঘটে "

আইজ্যাক নিউটন

## অতি সংক্ষিপ্ত প্রশ্ন

**১. এক্সসেপশনস কী?**

**উত্তর:** প্রোগ্রাম চলাকালীন (Runtime) কোনো অস্বাভাবিক পরিস্থিতির কারণে যখন প্রোগ্রামের স্বাভাবিক প্রবাহ বাধাগ্রস্ত হয়, তখন তাকে এক্সেপশন বলে। এটি কোনো ভুল ইনপুট বা রিসোর্সের অভাবের কারণে হতে পারে। যেমন: শূন্য দিয়ে ভাগ করার চেষ্টা করা (ZeroDivisionError) বা এমন ফাইল খোলার চেষ্টা করা যা অস্তিত্বহীন।

**২. বাগ কী?** [বাকাশিবো-২০২২ (অনুষ্ঠিত-২৩), ২০২৩ (পরি)]

**উত্তর:** সফটওয়্যার বা প্রোগ্রামের মধ্যে থাকা ত্রুটি, ভুল বা খুতকে বাগ বলা হয়। বাগের কারণে প্রোগ্রাম কাঙ্ক্ষিত ফলাফল দেয় না, ক্রাশ করে অথবা ভুল আউটপুট দেখায়। এই বাগ খুঁজে বের করে সংশোধন করার প্রক্রিয়াকে বলা হয় ডিবাগিং (Debugging)।

**৩. সিনট্যাক্স এরর বলতে কী কী বুঝায়?**

**উত্তর:** প্রোগ্রামিং ভাষার ব্যাকরণ বা লেখার নিয়ম লঙ্ঘন করলে যে এরর দেখা দেয়, তাকে সিনট্যাক্স এরর বলে। একে 'কম্পাইল টাইম এরর'-ও বলা হয়।

- উদাহরণ: ব্র্যাকেট শেষ না করা, সেমিকোলন দিতে ভুলে যাওয়া, বা কী-ওয়ার্ড ভুল বানানে লেখা (যেমন: print এর বদলে prnt লেখা)।

**৪. বিল্ট – ইন এক্সসেপশনস কী? -** [বাকাশিবো-২০২৩ (অনুষ্ঠিত-২৪), ২০২৩ (পরি)]

**উত্তর:** প্রোগ্রামিং ভাষায় (যেমন পাইথনে) আগে থেকেই যে সকল এক্সেপশন বা ত্রুটির ধরন ডিফাইন বা তৈরি করা থাকে, সেগুলোকে বিল্ট-ইন এক্সেপশনস বলে। প্রোগ্রামার যখন কোড লেখেন, তখন এই স্ট্যান্ডার্ড এররগুলো স্বয়ংক্রিয়ভাবে শনাক্ত হয়।

- উদাহরণ: ValueError, TypeError, IndexError ইত্যাদি।

**৫. রেইসিং কী?**

**উত্তর:** প্রোগ্রামার যখন ইচ্ছা করে কোনো নির্দিষ্ট শর্তে একটি এরর বা এক্সেপশন তৈরি করেন, তখন তাকে রেইসিং (Raising an Exception) বলা হয়। পাইথনে এটি raise কী-ওয়ার্ডের মাধ্যমে করা হয়। এটি মূলত তখন ব্যবহৃত হয় যখন প্রোগ্রামার চান যে কোনো নির্দিষ্ট ভুল ডেটা ইনপুট দিলে প্রোগ্রামটি থেমে যাক এবং ইউজারকে সতর্কবার্তা দেখাক।

**৬. এরর কতপ্রকার ও কী কী?** [বাকাশিবো-২০২৪ (অনুষ্ঠিত-২৫)]

**উত্তর:** প্রোগ্রামিং এরর প্রধানত ৩ প্রকার। নিচে এগুলো আলোচনা করা হলো:

- ১. সিনট্যাক্স এরর (Syntax Error): প্রোগ্রামিং ভাষার নিয়ম মেনে কোড না লিখলে এই এরর হয়।
- ২. লজিক্যাল এরর (Logical Error): প্রোগ্রামের যুক্তি বা ফর্মুলা ভুল হলে এই এরর হয়। এতে প্রোগ্রাম রান করলেও ভুল ফলাফল দেয়। (যেমন: বেগের সূত্রের জায়গায় ত্বরণের সূত্র ব্যবহার করা)।
- ৩. রানটাইম এরর (Runtime Error): কোড ঠিক থাকা সত্ত্বেও প্রোগ্রাম চলাকালীন কোনো সমস্যার কারণে এটি ঘটে। যেমন: মেমোরি শেষ হয়ে যাওয়া বা ভুল টাইপের ইনপুট দেওয়া। এক্সেপশনগুলো মূলত রানটাইম এররের অন্তর্ভুক্ত।

## সংক্ষিপ্ত প্রশ্ন

**১. পাইথন এক্সসেপশনস কী? উদাহারণসহ লেখ।** [বাকাশিবো-২০২৩ (অনুষ্ঠিত-২৪), ২০২৪ (অনুষ্ঠিত-২৫)]

**উত্তর:** পাইথন প্রোগ্রামে কোড সিনট্যাক্স অনুযায়ী সঠিক থাকা সত্ত্বেও প্রোগ্রাম রান করার সময় (Execution time) যে ত্রুটি বা অস্বাভাবিক পরিস্থিতির উদ্ভব হয়, তাকে এক্সেপশন (Exception) বলে। যখন পাইথনে কোনো এক্সেপশন ঘটে, তখন এটি একটি অবজেক্ট তৈরি করে। যদি প্রোগ্রামার সেই এররটি হ্যান্ডেল না করেন, তবে প্রোগ্রামটি হঠাৎ বন্ধ (Crash) হয়ে যায়।

**উদাহরণ:** সবচেয়ে কমন উদাহরণ হলো কোনো সংখ্যাকে শূন্য (০) দিয়ে ভাগ করার চেষ্টা করা।

```
a = 10

b = 0

print(a / b)
```

**২. পাইথনের try & except এক্সসেপশনস সংক্ষেপে লেখ।** [বাকাশিবো-২০২৪ (অনুষ্ঠিত-২৫)]

**উত্তর:** পাইথনে এক্সেপশন হ্যান্ডেল করার জন্য মূল টুল হলো try এবং except ব্লক। এর মাধ্যমে প্রোগ্রাম ক্রাশ না করে বিকল্প উপায়ে সচল রাখা যায়।

কার্যপদ্ধতি: ১. try block: এখানে সেই কোডটুকু রাখা হয় যেটিতে এরর হওয়ার সম্ভাবনা থাকে। পাইথন প্রথমে এই ব্লকের কোড রান করার চেষ্টা করে। ২. except block: যদি try ব্লকে কোনো এরর ঘটে, তবে পাইথন সাথে সাথে সেই কোড থেকে বের হয়ে except ব্লকে চলে আসে। এখানে এরর মেসেজ বা সমাধানের কোড লেখা থাকে।

**কোড উদাহরণ:**

```
try:
    number = int(input("একটি সংখ্যা দিন: "))
    result = 10 / number
    print(result)
except ZeroDivisionError:
    print("ভুল: আপনি শূন্য দিয়ে ভাগ করতে পারবেন না।")
except ValueError:
    print("ভুল: দয়া করে একটি সঠিক পূর্ণসংখ্যা দিন।")
```

**৩, Error handaling কী?** [বাকাশিবো-২০২২ (অনুষ্ঠিত-২৩)]

**উত্তর:** এরর হ্যান্ডলিং (Error Handling): একটি প্রোগ্রাম চলার সময় সম্ভাব্য ত্রুটি বা এররগুলো আগে থেকে অনুমান করে সেগুলো এমনভাবে মোকাবিলা করার প্রক্রিয়াকে Error Handling বলা হয় যাতে প্রোগ্রামটি পুরোপুরি বন্ধ না হয়ে স্বাভাবিকভাবে চলতে পারে অথবা ইউজারকে সঠিক নির্দেশনা দিতে পারে।

এরর হ্যান্ডলিং-এর গুরুত্ব:

- প্রোগ্রামের স্থায়িত্ব (Stability): এটি প্রোগ্রামকে হুটহাট ক্রাশ করা থেকে রক্ষা করে।
- ইউজার এক্সপেরিয়েন্স: এররের বদলে ব্যবহারকারীকে সহজবোধ্য ত্রুটি বার্তা (Error Message) দেখানো যায়।
- ডিবাগিং সহজ করে: ডেভেলপার বুঝতে পারেন প্রোগ্রামের ঠিক কোন জায়গায় এবং কেন সমস্যা হচ্ছে।
- রিসোর্স ম্যানেজমেন্ট: কোনো এরর ঘটলে ফাইল ক্লোজ করা বা ডাটাবেজ কানেকশন ডিসকানেক্ট করার মতো জরুরি কাজগুলো নিশ্চিত করা যায়।

সফল এরর হ্যান্ডলিং-এর মূল উপাদান: ১. ত্রুটি শনাক্ত করা (Detecting). ২. ত্রুটি প্রতিরোধ করা (Preventing). ৩. ত্রুটি ঘটলে তার সঠিক উত্তর দেওয়া (Responding).

## রচনামূলক প্রশ্ন

**১. পাইথনের try & except এক্সসেপশনস বর্ণনা কর।**

**উত্তর:** পাইথনে কোনো কোড রান করার সময় যদি কোনো ত্রুটি (Error) ঘটে, তবে প্রোগ্রামটি বন্ধ বা ক্রাশ হয়ে যায়। এই পরিস্থিতি সামাল দেওয়ার জন্য try এবং except ব্লক ব্যবহার করা হয়। একেই Exception Handling বলা হয়।

এর বিভিন্ন ব্লকের কাজ:

- try ব্লক: এই ব্লকের ভেতরে সেই কোডগুলো লেখা হয় যেগুলোতে এরর হওয়ার সম্ভাবনা থাকে।
- except ব্লক: যদি try ব্লকে কোনো এরর ঘটে, তবে প্রোগ্রামটি ক্রাশ না করে সরাসরি except ব্লকে চলে আসে এবং সেখানে থাকা নির্দেশাবলী পালন করে।
- else ব্লক (ঐচ্ছিক): যদি try ব্লকে কোনো এরর না ঘটে, তবে এই ব্লকটি রান করে।
- finally ব্লক (ঐচ্ছিক): এরর ঘটুক বা না ঘটুক, এই ব্লকের কোড সবশেষে অবশ্যই রান করবে।

**একটি সাধারণ উদাহরণ:**

```
try:
    x = 10 / 0  # এখানে ZeroDivisionError ঘটবে
except:
    print("একটি ভুল হয়েছে! শূন্য দিয়ে ভাগ করা সম্ভব নয়।")
```

**২. Raising exception ব্যবহারপূর্বক python এ একটি প্রোগ্রাম লেখ।** [বাকাশিবো-২০২২ (অনুষ্ঠিত-২৩)]

**উত্তর:**
```
def check_age(age):
    if age < 0:
        raise ValueError("দুঃখিত, বয়স কখনো ঋণাত্মক হতে পারে না।")
    elif age < 18:
        raise Exception("আপনি এখনো ভোটার হওয়ার উপযুক্ত নন।")
    else:
        print("অভিনন্দন! আপনি ভোটার হওয়ার উপযুক্ত।")
try:
    user_age = int(input("আপনার বয়স লিখুন: "))
    check_age(user_age)
except ValueError as e:
    print(f"ইনপুট এরর: {e}")
except Exception as e:
    print(f"সতর্কবার্তা: {e}")
```

# অধ্যায় ০৮ পাইথন এর লগিং

এই অধ্যায়ের প্রধান সূচিপত্র (contants)


| বিষয় | পৃষ্ঠা নং |
|---|---|
| ❖ সূত্রাবলী ও শর্টকার্ট টেকনিক | - |
| ❖ অতি সংক্ষিপ্ত প্রশ্নাবলি | ৫৬ |
| ❖ সংক্ষিপ্ত প্রশ্নাবলি | ৫৬ |
| ❖ রচনামূলক প্রশ্নাবলি | ৫৮ |


## বোর্ড প্রশ্নাবলির বিশ্লেষন (Board Questions Analysis)

এই অধ্যায় থেকে বিগত তিন বছরের ডিপ্লোমা ইন ইঞ্জিনিয়ারিং, ডিপ্লোমা ইন টেক্সটাইল, ডিপ্লোমা ইন এগ্রিকালচার, ডিপ্লোমা ইন মেরিন পরীক্ষায় মোট প্রশ্ন....নিচে 'Board Questions Analysis' অংশে এই অধ্যায় থেকে কোন সালে কোন বোর্ডে কতটি প্রশ্ন হয়েছে তা দেখানো হলো।

| সাল / বোর্ড | অতিসংক্ষিপ্ত | সংক্ষিপ্ত | রচনামূলক |
|---|---|---|---|
| ২০২৪ | ১টি | | ২টি |
| ২০২৩ | -- | ২টি | -- |
| ২০২২ | -- | ১টি | ১টি |

" কখনো সপ্ন দেখা বন্ধ করো না,
পরিশ্রমই তোমাকে সাফল্যের দিকে নিয়ে যাবে "

মেরি কুরি

## অতি সংক্ষিপ্ত প্রশ্ন

**১. পাইথন লগিং কী?** [বাকাশিবো-২০২৪ (অনুষ্ঠিত-২৫)]

**উত্তর:** পাইথন লগিং (Logging) হলো একটি শক্তিশালী সিস্টেম যার মাধ্যমে একটি প্রোগ্রাম চলার সময় ব্যাকগ্রাউন্ডে কী কী ঘটছে, তার রেকর্ড রাখা যায়। এটি মূলত ডিবাগিং এবং সফটওয়্যার মনিটরিংয়ের জন্য ব্যবহৃত হয়। কোডে কোনো ভুল (Error) হলে বা কোনো নির্দিষ্ট ঘটনা ঘটলে তা ট্র্যাক করার জন্য প্রিন্ট স্টেটমেন্টের বদলে লগিং ব্যবহার করা অনেক বেশি পেশাদার এবং কার্যকর পদ্ধতি।

**২. লগিং এর বিভিন্ন লেভেলগুলো উল্লেখ কর।**

**উত্তর:**

| লেভেল | গুরুত্ব | ব্যবহার |
|---|---|---|
| DEBUG | সর্বনিম্ন | বিস্তারিত তথ্য, সাধারণত সমস্যা খুঁজে বের করার (Debugging) জন্য ব্যবহৃত হয়। |
| INFO | সাধারণ | প্রোগ্রাম ঠিকঠাক কাজ করছে কি না তার সাধারণ নিশ্চয়তা প্রদান করে। |
| WARNING | মাঝারি | কোনো অনাকাঙ্ক্ষিত বিষয় যা বর্তমানে সমস্যা নয়, কিন্তু ভবিষ্যতে হতে পারে। |
| ERROR | উচ্চ | কোনো গুরুতর সমস্যার কারণে প্রোগ্রামের একটি নির্দিষ্ট অংশ কাজ করতে পারছে না। |
| CRITICAL | সর্বোচ্চ | অত্যন্ত গুরুতর ভুল যার ফলে পুরো প্রোগ্রামটি বন্ধ হয়ে যেতে পারে। |

**৩. লগ ফাইল কী?**

**উত্তর:** লগ ফাইল হলো এমন একটি টেক্সট ফাইল যেখানে একটি সফটওয়্যার বা সিস্টেমের কার্যক্রমের ধারাবাহিক রেকর্ড বা ইতিহাস জমা থাকে। যখনই প্রোগ্রামে কোনো ঘটনা ঘটে (যেমন: ইউজার লগইন করা, কোনো ফাইল ডিলিট হওয়া বা কোনো এরর আসা), তখন সেই তথ্যগুলো সময়সহ এই ফাইলে লিখে রাখা হয়। এর ফলে পরবর্তীতে কোনো সমস্যা হলে ডেভেলপাররা লগ ফাইলটি চেক করে সহজেই বুঝতে পারেন ঠিক কখন এবং কেন সমস্যাটি তৈরি হয়েছিল।

## সংক্ষিপ্ত প্রশ্ন

**১. পাইথন লগিং কী? সংক্ষেপে লেখ।**

**উত্তর:** পাইথন লগিং (Logging) হলো একটি বিল্ট-ইন মডিউল যার মাধ্যমে প্রোগ্রাম চলার সময় ব্যাকগ্রাউন্ডে ঘটা

বিভিন্ন ইভেন্ট বা ঘটনা ট্র্যাক করা এবং রেকর্ড করে রাখা যায়। সফটওয়্যার ডেভেলপমেন্টের সময় ডিবাগিং (Debugging) করতে এবং পরবর্তীতে প্রোগ্রামের কোনো সমস্যা বিশ্লেষণ করতে এটি প্রিন্ট (print) ফাংশনের চেয়ে অনেক বেশি কার্যকর ভূমিকা পালন করে।

**২. লগিং এর বিভিন্ন লেভেলগুলো উল্লেখ কর।** [বাকাশিবো-২০২২ (অনুষ্ঠিত-২৩), ২৩ (অনুষ্ঠিত-২৪), ২০২৩ (পরি)]

**উত্তর:**

| **লেভেল** | **গুরুত্ব** | **ব্যবহার** |
|---|---|---|
| DEBUG | সর্বনিম্ন | বিস্তারিত তথ্য, সাধারণত সমস্যা খুঁজে বের করার (Debugging) জন্য ব্যবহৃত হয়। |
| INFO | সাধারণ | প্রোগ্রাম ঠিকঠাক কাজ করছে কি না তার সাধারণ নিশ্চয়তা প্রদান করে। |
| WARNING | মাঝারি | কোনো অনাকাঙ্ক্ষিত বিষয় যা বর্তমানে সমস্যা নয়, কিন্তু ভবিষ্যতে হতে পারে। |
| ERROR | উচ্চ | কোনো গুরুতর সমস্যার কারণে প্রোগ্রামের একটি নির্দিষ্ট অংশ কাজ করতে পারছে না। |
| CRITICAL | সর্বোচ্চ | অত্যন্ত গুরুতর ভুল যার ফলে পুরো প্রোগ্রামটি বন্ধ হয়ে যেতে পারে। |

**৩. লগিং এর আউটপুট ফরমেটিং এরং কনফিগারেশন সংক্ষেপে লেখ।**

**উত্তর:** লগিং ব্যবস্থাটি অত্যন্ত নমনীয়। আমরা চাইলেই লগের তথ্যগুলো কীভাবে দেখাবে (ফরমেটিং) এবং কোথায় সংরক্ষিত হবে (কনফিগারেশন), তা নিয়ন্ত্রণ করতে পারি।

ক) লগিং কনফিগারেশন (Configuration):

সাধারণত logging.basicConfig() ফাংশন ব্যবহার করে লগিং কনফিগার করা হয়। এর মাধ্যমে আমরা নির্ধারণ করতে পারি লগগুলো কনসোলে দেখাবে নাকি ফাইলে সেভ হবে। এর প্রধান প্যারামিটারগুলো হলো:

- filename: লগগুলো যে ফাইলে সেভ হবে তার নাম।
- filemode: ফাইলটি খোলার মোড (যেমন: 'w' বা 'a')।
- level: কোন লেভেল থেকে লগ রেকর্ড করা শুরু হবে (যেমন: INFO বা DEBUG)।

খ) আউটপুট ফরমেটিং (Formatting):

লগ মেসেজের সাথে সময়, তারিখ, বা লগ লেভেল দেখানোর জন্য format প্যারামিটার ব্যবহার করা হয়। এর মাধ্যমে লগের চেহারা পরিবর্তন করা যায়। কিছু সাধারণ অ্যাট্রিবিউট হলো:

- %(asctime)s : ঘটনার সময় ও তারিখ।
- %(levelname)s : লগের লেভেল (যেমন: INFO, ERROR)।
- %(message)s : মূল মেসেজ।

```
import logging
logging.basicConfig(
    filename='app.log',          # ফাইল এর নাম
    filemode='w',                # রাইট মোড
    level=logging.INFO,          # লগ লেভেল
    format='%(asctime)s - %(levelname)s - %(message)s' # ফরমেটিং
)
logging.info("This is a formatted log message.")
```

**ফলাফল (ফাইলে যা সেভ হবে):** 2023-10-27 10:30:00,123 - INFO - This is a formatted log message.

## রচনামূলক প্রশ্ন

**১. লগিং এর বিভিন্ন লেভেলগুলো বর্ণনা কর।** [বাকাশিবো-২০২৪ (অনুষ্ঠিত-২৫)]

**উত্তর:** পাইথন logging মডিউলে ঘটনার গুরুত্ব বা তীব্রতা বোঝানোর জন্য ৫টি স্ট্যান্ডার্ড লেভেল বা স্তর ব্যবহার করা হয়। এই লেভেলগুলো ডেভেলপারদের বুঝতে সাহায্য করে যে লগের তথ্যটি কতটা জরুরি। গুরুত্বের ক্রমানুসারে (কম থেকে বেশি) লেভেলগুলো নিচে বর্ণনা করা হলো:

| লেভেল (Level) | নিউমেরিক ভ্যালু | বর্ণনা ও ব্যবহার |
|---|---|---|

| | | |
|---|---|---|
| ১. DEBUG | ১০ (সর্বনিম্ন) | এটি মূলত সিস্টেমের বিস্তারিত তথ্যের জন্য ব্যবহৃত হয়। সাধারণত ডেভেলপাররা প্রোগ্রাম তৈরি বা ডিবাগ করার সময় ভেরিয়েবল চেক বা ফ্লো ট্রেস করার জন্য এটি ব্যবহার করেন। সাধারণ ব্যবহারের সময় এই তথ্যগুলো দেখার প্রয়োজন হয় না। |
| ২. INFO | 20 | এই লেভেলটি নিশ্চিত করে যে সিস্টেম বা প্রোগ্রাম প্রত্যাশা অনুযায়ী ঠিকঠাক কাজ করছে। যেমন: "সার্ভার চালু হয়েছে" বা "ডেটাবেস কানেক্ট হয়েছে" - এই ধরনের সাধারণ তথ্য এখানে থাকে। |
| ৩. WARNING | 30 | এটি একটি সতর্কবার্তা। প্রোগ্রামটি এখনও চলছে, কিন্তু এমন কিছু ঘটেছে যা ভবিষ্যতে সমস্যার সৃষ্টি করতে পারে। যেমন: "ডিস্ক স্পেস কমে আসছে" বা "পুরানো লাইব্রেরি ব্যবহার করা হচ্ছে"। ডিফল্টভাবে পাইথন এখান থেকেই লগ দেখানো শুরু করে। |
| ৪. ERROR | 40 | এটি গুরুতর সমস্যার ইঙ্গিত দেয়। যখন কোনো নির্দিষ্ট ফাংশন বা কাজ সম্পন্ন হতে ব্যর্থ হয় (যেমন: ফাইল নট ফাউন্ড বা কানেকশন ফেইল্ড), তখন এটি ব্যবহার করা হয়। তবে প্রোগ্রামটি ক্র্যাশ না করে বাকি কাজ চালিয়ে যেতে পারে। |
| ৫. CRITICAL | ৫০ (সর্বোচ্চ) | এটি সর্বোচ্চ স্তরের বা ভয়াবহ ত্রুটি। এর অর্থ হলো বড় ধরনের বিপর্যয় ঘটেছে যার ফলে পুরো প্রোগ্রাম বা অ্যাপ্লিকেশনটি বন্ধ হয়ে যেতে পারে। |

**২. লগিং এর আউটপুট ফরমেটিং এবং কনফিগারেশন বর্ণনা কর।**

[বাকাশিবো-২০২২ (অনুষ্ঠিত-২৩), ২০২৪ (অনুষ্ঠিত-২৫)]

**উত্তর:** পাইথনে লগিং ব্যবস্থাকে প্রয়োজন অনুযায়ী সাজানো বা কাস্টমাইজ করার প্রক্রিয়াকে কনফিগারেশন এবং লগের চেহারার পরিবর্তনকে ফরমেটিং বলা হয়। সাধারণত logging.basicConfig() ফাংশন ব্যবহার করে এই কাজগুলো করা হয়।

নিচে বিস্তারিত আলোচনা করা হলো:

ক) লগিং কনফিগারেশন (Configuration)

লগ মেসেজগুলো কোথায় জমা হবে, কোন মোডে ফাইল ওপেন হবে এবং কোন লেভেলের লগ রেকর্ড করা হবে—এসব ঠিক করা হয় কনফিগারেশনের মাধ্যমে। basicConfig ফাংশনের গুরুত্বপূর্ণ প্যারামিটারগুলো হলো:

১. filename: লগগুলো কনসোলে (স্ক্রিনে) না দেখিয়ে কোনো ফাইলে সেভ করতে চাইলে ফাইলের নাম এখানে দিতে হয় (যেমন: app.log)। ২. filemode: ফাইলটি কীভাবে খোলা হবে তা নির্দেশ করে।

- w (Write mode): প্রতিবার প্রোগ্রাম রান করলে আগের লগ মুছে নতুন করে লিখবে।

- a (Append mode): আগের লগের সাথে নতুন লগ যুক্ত করবে (এটি ডিফল্ট)। ৩. level: কোন লেভেল থেকে লগ রেকর্ড শুরু হবে তা নির্ধারণ করে। যেমন: logging.INFO সেট করলে INFO এবং তার ওপরের (WARNING, ERROR, CRITICAL) লগগুলো সেভ হবে, কিন্তু DEBUG বাদ যাবে।

খ) লগিং ফরমেটিং (Formatting)

লগ ফাইলে শুধুমাত্র মেসেজ সেভ না করে তার সাথে সময়, তারিখ, ফাইলের নাম বা লেভেলের নাম যুক্ত করার প্রক্রিয়াকে ফরমেটিং বলে। এটি format প্যারামিটারের মাধ্যমে করা হয়। কিছু বহুল ব্যবহৃত LogRecord অ্যাট্রিবিউট হলো:

- %(asctime)s : ঘটনাটি ঘটার তারিখ ও সময়।
- %(levelname)s : লগের লেভেল (যেমন: INFO, ERROR)।
- %(message)s : ব্যবহারকারীর দেওয়া মূল বার্তা।
- %(name)s : লগার-এর নাম।

**উদাহরণ (কোডসহ ব্যাখ্যা):**

```
import logging

# কনফিগারেশন এবং ফরমেটিং সেটআপ
logging.basicConfig(
    filename='system_log.txt',          # ১. লগ ফাইলের নাম
    filemode='a',                       # ২. অ্যাপেন্ড মোড
    level=logging.INFO,                 # ৩. শুধু INFO বা তার উপরের লেভেল দেখাবে
    format='%(asctime)s - %(levelname)s - %(message)s', # ৪. আউটপুট ফরমেট
    datefmt='%Y-%m-%d %H:%M:%S'         # ৫. সময়ের ফরমেট ঠিক করা
)

# লগ তৈরি করা
logging.warning("This is a warning message.")
logging.error("Database connection failed.")
```

**আউটপুট (system_log.txt ফাইলে যা দেখা যাবে):**

```
2023-10-25 14:30:10 - WARNING - This is a warning message.
2023-10-25 14:30:11 - ERROR - Database connection failed.
```

# অধ্যায় ০৯ ইউনিট টেস্টিং

এই অধ্যায়ের প্রধান সূচিপত্র (contants)


| বিষয় | পৃষ্ঠা নং |
|---|---|
| ❖ সূত্রাবলী ও শর্টকার্ট টেকনিক | - |
| ❖ অতি সংক্ষিপ্ত প্রশ্নাবলি | ৬২ |
| ❖ সংক্ষিপ্ত প্রশ্নাবলি | ৬৩ |
| ❖ রচনামূলক প্রশ্নাবলি | ৬৪ |


## বোর্ড প্রশ্নাবলির বিশ্লেষন (Board Questions Analysis)

এই অধ্যায় থেকে বিগত তিন বছরের ডিপ্লোমা ইন ইঞ্জিনিয়ারিং, ডিপ্লোমা ইন টেক্সটাইল, ডিপ্লোমা ইন এগ্রিকালচার, ডিপ্লোমা ইন মেরিন পরীক্ষায় মোট প্রশ্ন....নিচে 'Board Questions Analysis' অংশে এই অধ্যায় থেকে কোন সালে কোন বোর্ডে কতটি প্রশ্ন হয়েছে তা দেখানো হলো।

| সাল / বোর্ড | অতিসংক্ষিপ্ত | সংক্ষিপ্ত | রচনামূলক |
|---|---|---|---|
| ২০২৪ | -- | ১টি | -- |
| ২০২৩ | ২টি | -- | -- |
| ২০২২ | -- | -- | ১টি |

" কখনো সপ্ন দেখা বন্ধ করো না,
পরিশ্রমই তোমাকে সাফল্যের দিকে নিয়ে যাবে "

মেরি কুরি

## অতি সংক্ষিপ্ত প্রশ্ন

**১. ইউনিট টেস্টিং কী?** [বাকাশিবো-২০২৩ (অনুষ্ঠিত-২৪)]

**উত্তর:** ইউনিট টেস্টিং (Unit Testing) হলো সফটওয়্যার টেস্টিং-এর একটি প্রাথমিক ধাপ, যেখানে সফটওয়্যারের সোর্স কোডের সবচেয়ে ক্ষুদ্রতম অংশ বা 'ইউনিট' (যেমন: ফাংশন, মেথড বা ক্লাস) আলাদা আলাদাভাবে পরীক্ষা করা হয়। এর মূল লক্ষ্য হলো কোডের প্রতিটি একক অংশ বা ইউনিট স্বাধীনভাবে এবং সঠিকভাবে কাজ করছে কি না তা নিশ্চিত করা। সাধারণত ডেভেলপাররাই এই টেস্টিং সম্পন্ন করেন।

**২. ইউনিট টেস্টিং এর প্রয়জনীয়তা কী?** [বাকাশিবো-২০২৩ (পরি)]

**উত্তর:** সফটওয়্যার ডেভেলপমেন্টে ইউনিট টেস্টিং অত্যন্ত গুরুত্বপূর্ণ। নিচে এর প্রয়োজনীয়তাগুলো পয়েন্ট আকারে দেওয়া হলো:

1. ত্রুটি শনাক্তকরণ: ডেভেলপমেন্টের প্রাথমিক পর্যায়েই কোডের ভুল বা বাগ (Bug) খুঁজে বের করতে সাহায্য করে।
2. কোড পরিবর্তন সহজ করা: ইউনিট টেস্ট থাকলে ভবিষ্যতে কোডে কোনো পরিবর্তন বা রিফ্যাক্টরিং (Refactoring) করলে পুরনো কোড ভেঙে যাচ্ছে কি না, তা সহজেই বোঝা যায়।
3. ডিবাগিং সহজ করা: যেহেতু প্রতিটি ছোট অংশ আলাদাভাবে টেস্ট করা হয়, তাই কোনো সমস্যা হলে ঠিক কোথায় ভুল হয়েছে তা দ্রুত খুঁজে বের করা যায়।
4. খরচ ও সময় সাশ্রয়: সফটওয়্যার তৈরির শেষের দিকে ভুল ধরা পড়ার চেয়ে শুরুতে ধরা পড়লে তা ঠিক করতে সময় ও খরচ—উভয়ই কম লাগে।
5. ডকুমেন্টেশন হিসেবে কাজ করে: ইউনিট টেস্ট কেসগুলো দেখলে বোঝা যায় যে কোডটি আসলে কী কাজ করার কথা, তাই এটি এক ধরনের ডকুমেন্টেশন হিসেবেও কাজ করে।

**৩. পাইথন বিল্ট – ইন ইউনিট টেস্ট ইউনিটগুলোর নাম লেখ।**

**উত্তর:** পাইথনের বিল্ট-ইন টেস্টিং ফ্রেমওয়ার্কের নাম হলো unittest। এর প্রধান চারটি উপাদান বা কনসেপ্ট রয়েছে:

1. টেস্ট ফিক্সচার (Test Fixture): টেস্ট রান করার জন্য প্রয়োজনীয় পরিবেশ তৈরি করা। যেমন: টেস্টের শুরুতে ডাটাবেস কানেকশন দেওয়া (Setup) এবং শেষে তা বন্ধ করা (Teardown)।

2. টেস্ট কেস (Test Case): এটি টেস্টিংয়ের মূল একক। unittest.TestCase ক্লাস ইনহেরিট করে নতুন টেস্ট কেস তৈরি করা হয়, যেখানে নির্দিষ্ট ইনপুটের জন্য আউটপুট সঠিক কি না তা চেক করা হয়।

3. টেস্ট স্যুট (Test Suite): অনেকগুলো টেস্ট কেস বা টেস্ট স্যুটের সমষ্টি। যখন একসাথে অনেকগুলো টেস্ট রান করার প্রয়োজন হয়, তখন টেস্ট স্যুট ব্যবহার করা হয়।

4. টেস্ট রানার (Test Runner): এটি এমন একটি কম্পোনেন্ট যা টেস্টগুলোকে চালায় (Execute করে) এবং ফলাফল (Pass/Fail) প্রদর্শন করে।

## সংক্ষিপ্ত প্রশ্ন

**১. ইউনিট টেস্টিং এর প্রয়জনীয়তা সংক্ষেপে লেখ।** [বাকাশিবো-২০২৪ (অনুষ্ঠিত-২৫)]

**উত্তর:** ফটওয়্যার ডেভেলপমেন্ট লাইফ সাইকেলে (SDLC) ইউনিট টেস্টিং একটি অত্যন্ত গুরুত্বপূর্ণ ধাপ। এর প্রধান প্রয়োজনীয়তাগুলো নিচে বর্ণনা করা হলো:

- ভুল বা বাগ (Bug) দ্রুত শনাক্তকরণ: কোড লেখার সাথে সাথেই তা পরীক্ষা করা হয় বলে ডেভেলপমেন্টের প্রাথমিক ধাপেই ত্রুটিগুলো ধরা পড়ে। এতে পরবর্তী বড় ধরনের জটিলতা এড়ানো সম্ভব হয়।

- ডিবাগিং সহজতর করা: যেহেতু প্রতিটি ছোট অংশ (Unit) আলাদাভাবে পরীক্ষা করা হয়, তাই কোনো সমস্যা হলে ডেভেলপার ঠিক কোন লাইনে বা কোন ফাংশনে সমস্যা হয়েছে তা খুব দ্রুত চিহ্নিত করতে পারেন।

- কোড রিফ্যাক্টরিং ও আপডেট: কোড পরিবর্তন বা নতুন ফিচার যোগ করার সময় আগের কোড ঠিকঠাক কাজ করছে কি না তা নিশ্চিত হওয়া যায়। অর্থাৎ, নতুন কোড যোগ করার ফলে পুরনো সিস্টেমে কোনো সমস্যা তৈরি হচ্ছে কি না তা বোঝা যায়।

- খরচ ও সময় সাশ্রয়: ডেভেলপমেন্টের শুরুতে ভুল ধরা পড়লে তা সংশোধন করা সহজ এবং সস্তা। কিন্তু সফটওয়্যার পুরোপুরি তৈরি হওয়ার পর বা গ্রাহকের কাছে পৌঁছানোর পর ভুল ধরা পড়লে তা সংশোধন করা অনেক ব্যয়বহুল ও সময়সাপেক্ষ।

- কোডের মান ও নির্ভরযোগ্যতা বৃদ্ধি: নিয়মিত ইউনিট টেস্টিং করলে কোডের মান উন্নত হয় এবং সফটওয়্যারটি ব্যবহারের জন্য আরও বেশি নির্ভরযোগ্য হয়ে ওঠে।

- ডকুমেন্টেশন হিসেবে কাজ করা: ইউনিট টেস্টগুলো দেখলে অন্য ডেভেলপাররা সহজেই বুঝতে পারেন যে একটি নির্দিষ্ট ফাংশন বা মেথড কী কাজ করার জন্য তৈরি করা হয়েছে।

## রচনামূলক প্রশ্ন

**১. পাইথনে ইউনিট টেস্টিং এর গুরুত্ব বর্ণনা কর।**

**উত্তর:** পাইথনে ইউনিট টেস্টিং হলো একটি পদ্ধতি যেখানে প্রোগ্রামের ক্ষুদ্রতম অংশগুলো (যেমন: ফাংশন বা ক্লাস) সঠিকভাবে কাজ করছে কি না তা যাচাই করা হয়। এর গুরুত্ব অপরিসীম:

- বাগ (Bug) দ্রুত শনাক্তকরণ: কোড লেখার প্রাথমিক পর্যায়েই ভুল ধরা পড়ে, ফলে বড় ধরনের বিপর্যয় এড়ানো যায়।
- কোড রিফ্যাক্টরিং: কোড যখন পরিমার্জন বা উন্নত করা হয়, তখন ইউনিট টেস্টিং নিশ্চিত করে যে পরিবর্তনের ফলে আগের ফিচারগুলো নষ্ট হয়ে যায়নি।
- ডিবাগিং সহজ করে: যেহেতু প্রতিটি ফাংশন আলাদাভাবে পরীক্ষা করা হয়, তাই কোনো সমস্যা হলে ঠিক কোথায় ভুল হয়েছে তা দ্রুত খুঁজে বের করা সম্ভব হয়।
- খরচ ও সময় সাশ্রয়: প্রজেক্টের শেষ পর্যায়ে ভুল ধরা পড়লে তা ঠিক করা ব্যয়বহুল ও সময়সাপেক্ষ। ইউনিট টেস্টিং শুরুতে এই ঝুঁকি কমিয়ে দেয়।
- ডকুমেন্টেশন: ইউনিট টেস্টগুলো দেখলে অন্য ডেভেলপাররা সহজেই বুঝতে পারেন যে কোডটির কোন অংশ কী আউটপুট দেওয়ার কথা।

**২. ইউনিট টেস্টিং এর Structure দেখাও।** [বাকাশিবো-২০২২ (অনুষ্ঠিত-২৩)]

**উত্তর:** পাইথনের বিল্ট-ইন unittest মডিউল ব্যবহার করে ইউনিট টেস্টিং করা হয়। একটি স্ট্যান্ডার্ড ইউনিট টেস্টের কাঠামো মূলত ৪টি প্রধান অংশের সমন্বয়ে গঠিত:

ইউনিট টেস্টিং কাঠামোর প্রধান অংশসমূহ: ১. Setup: টেস্ট শুরু করার আগে প্রয়োজনীয় এনভায়রনমেন্ট বা ডেটা প্রস্তুত করা। ২. Test Case: আসল টেস্ট কোড যেখানে ইনপুট দিয়ে আউটপুট পরীক্ষা করা হয়। ৩. Assertion: প্রাপ্ত আউটপুট এবং প্রত্যাশিত আউটপুট তুলনা করা। ৪. Teardown: টেস্ট শেষ হওয়ার পর অস্থায়ী ডেটা বা ফাইল মুছে ফেলে পরিষ্কার করা।

**কাঠামোর উদাহরণ (কোড):**

```
import unittest

# ১. টেস্ট কেস ক্লাস তৈরি
class TestMyMath(unittest.TestCase):

    # ২. সেটিংস বা সেটআপ (ঐচ্ছিক)
    def setUp(self):
        self.value = 10

    # ৩. টেস্ট মেথড (অবশ্যই 'test' দিয়ে শুরু হতে হবে)
    def test_addition(self):
        result = self.value + 5
        self.assertEqual(result, 15)  # Assertion

    # ৪. ক্লিয়ারিং বা টিয়ারডাউন (ঐচ্ছিক)
    def tearDown(self):
        pass

if __name__ == '__main__':
    unittest.main()
```

# অধ্যায় ১০ পাইথন রেগুলার এক্সপ্রেশন

এই অধ্যায়ের প্রধান সূচিপত্র (contants)


| বিষয় | পৃষ্ঠা নং |
|---|---|
| ❖ সূত্রাবলী ও শর্টকার্ট টেকনিক | - |
| ❖ অতি সংক্ষিপ্ত প্রশ্নাবলি | ৬৭ |
| ❖ সংক্ষিপ্ত প্রশ্নাবলি | ৬৮ |
| ❖ রচনামূলক প্রশ্নাবলি | ৭০ |


## বোর্ড প্রশ্নাবলির বিশ্লেষন (Board Questions Analysis)

এই অধ্যায় থেকে বিগত তিন বছরের ডিপ্লোমা ইন ইঞ্জিনিয়ারিং, ডিপ্লোমা ইন টেক্সটাইল, ডিপ্লোমা ইন এগ্রিকালচার, ডিপ্লোমা ইন মেরিন পরীক্ষায় মোট প্রশ্ন....নিচে 'Board Questions Analysis' অংশে এই অধ্যায় থেকে কোন সালে কোন বোর্ডে কতটি প্রশ্ন হয়েছে তা দেখানো হলো।

| সাল / বোর্ড | অতিসংক্ষিপ্ত | সংক্ষিপ্ত | রচনামূলক |
|---|---|---|---|
| ২০২৪ | -- | -- | -- |
| ২০২৩ | ১টি | -- | ২টি |
| ২০২২ | ১টি | -- | ১টি |

" কখনো সপ্ন দেখা বন্ধ করো না,
পরিশ্রমই তোমাকে সাফল্যের দিকে নিয়ে যাবে "

মেরি কুরি

## অতি সংক্ষিপ্ত প্রশ্ন

**১. পাইথন রেগুলার এক্সপ্রেশন কী?** [বাকাশিবো-২০২২ (অনুষ্ঠিত-২৩)]

**উত্তর:** রেগুলার এক্সপ্রেশন (Regular Expression বা সংক্ষেপে Regex) হলো অক্ষরের একটি বিশেষ সিকুয়েন্স বা প্যাটার্ন, যা কোনো টেক্সট বা স্ট্রিংয়ের মধ্যে নির্দিষ্ট কোনো শব্দ বা বিন্যাস খুঁজে বের করতে, এডিট করতে বা ম্যানিপুলেট করতে ব্যবহৃত হয়। পাইথনে রেগুলার এক্সপ্রেশন ব্যবহারের জন্য বিল্ট-ইন re মডিউল ব্যবহার করা হয়। এটি ইমেইল ভ্যালিডেশন, পাসওয়ার্ড চেক বা বড় কোনো টেক্সট থেকে তথ্য খোঁজার কাজে অত্যন্ত কার্যকর।

**২. মেটাক্যারেক্টার কী?**

**উত্তর:** রেগুলার এক্সপ্রেশনে ব্যবহৃত যেসব ক্যারেক্টার বা প্রতীকের নিজস্ব সাধারণ অর্থের বাইরে বিশেষ কোনো অর্থ বা কাজ থাকে, তাদের মেটাক্যারেক্টার (Metacharacter) বলা হয়। এই ক্যারেক্টারগুলো সরাসরি কোনো বর্ণ হিসেবে কাজ না করে সার্চ প্যাটার্ন তৈরি করতে সাহায্য করে। যেমন: . (ডট) সাধারণ একটি ফোটা হলেও Regex-এ এটি যেকোনো একটি ক্যারেক্টারকে নির্দেশ করে।

**৩. মেটাক্যারেক্টার কী কী?** [বাকাশিবো-২০২৩ (পরি)]

**উত্তর:** পাইথন রেগুলার এক্সপ্রেশনে বহুল ব্যবহৃত কিছু মেটাক্যারেক্টার হলো:

| মেটাক্যারেক্টার | নাম | কাজ |
|---|---|---|
| . | ডট (Dot) | যেকোনো একটি ক্যারেক্টার (নিউলাইন বাদে)। |
| ^ | ক্যারেট (Caret) | স্ট্রিং-এর শুরু বোঝাতে ব্যবহৃত হয়। |
| $ | ডলার (Dollar) | স্ট্রিং-এর শেষ বোঝাতে ব্যবহৃত হয়। |
| * | স্টার (Star) | কোনো ক্যারেক্টার ০ বা তার বেশিবার থাকলে। |
| + | প্লাস (Plus) | কোনো ক্যারেক্টার ১ বা তার বেশিবার থাকলে। |
| ? | প্রশ্নবোধক | কোনো ক্যারেক্টার ০ বা ১ বার থাকলে। |
| [] | ব্র্যাকেট | ক্যারেক্টারের সেট বোঝাতে (যেমন: [a-z])। |

| | | |
|---|---|---|
| \| | পাইপ | অথবা' (OR) বোঝাতে ব্যবহৃত হয়। |

**৪. স্পেশাল সিকুয়েন্স এর তালিকা তৈরি কর।**

**উত্তর:** স্পেশাল সিকুয়েন্স বলতে একটি ব্যাকস্ল্যাশ \ এবং তার পরে একটি নির্দিষ্ট ক্যারেক্টারের সমন্বয়কে বোঝায়, যার বিশেষ অর্থ থাকে। নিচে একটি তালিকা দেওয়া হলো:

- \d : যেকোনো একটি ডিজিট বা সংখ্যা খুঁজে বের করে (০-৯)।
- \D : সংখ্যা নয় এমন যেকোনো ক্যারেক্টার খুঁজে বের করে।
- \s : যেকোনো হোয়াইটস্পেস (Space, Tab, Newline) খুঁজে বের করে।
- \S : হোয়াইটস্পেস নয় এমন ক্যারেক্টার খুঁজে বের করে।
- \w : যেকোনো শব্দ ক্যারেক্টার (A-Z, a-z, 0-9 এবং underscore _) খুঁজে বের করে।
- \W : শব্দ ক্যারেক্টার নয় এমন কিছু (যেমন: @, #, $, space) খুঁজে বের করে।
- \A : যদি নির্দিষ্ট ক্যারেক্টারগুলো স্ট্রিং-এর একদম শুরুতে থাকে।
- \Z : যদি নির্দিষ্ট ক্যারেক্টারগুলো স্ট্রিং-এর একদম শেষে থাকে।

## সংক্ষিপ্ত প্রশ্ন

**১. পাইথন রেগুলার এক্সপ্রেশন কী? সংক্ষেপে লেখ।**

**উত্তর:** পাইথন রেগুলার এক্সপ্রেশন (Regular Expression বা সংক্ষেপে Regex) হলো বিশেষ ক্যারেক্টার বা চিহ্নের একটি বিন্যাস (Pattern), যা কোনো টেক্সট বা স্ট্রিংয়ের ভেতর নির্দিষ্ট কোনো তথ্য খুঁজে বের করতে, যাচাই করতে (Validation) বা পরিবর্তন করতে ব্যবহৃত হয়। পাইথনে re মডিউল ব্যবহার করে রেগুলার এক্সপ্রেশনের কাজ করা হয়। এটি ইমেইল অ্যাড্রেস ভ্যালিডেশন, ফোন নম্বর ফরমেট চেক করা বা বড় কোনো ডকুমেন্ট থেকে নির্দিষ্ট শব্দ খোঁজার কাজে অপরিহার্য।

**২. পাইথন রেগুলার এক্সপ্রেশন এর বিল্ট – ইন মেথড তালিকা তৈরি কর।**

**উত্তর:** পাইথনের re মডিউলে বেশ কিছু বিল্ট-ইন মেথড বা ফাংশন রয়েছে। প্রধান মেথডগুলোর তালিকা ও কাজ নিচে দেওয়া হলো:

| মেথড (Method) | কাজ (Description) |
|---|---|
| findall() | স্ট্রিংয়ের মধ্যে প্যাটার্নের সাথে মিলে যায় এমন সব অংশের একটি তালিকা (List) প্রদান করে। |
| search() | স্ট্রিংয়ের যেকোনো স্থানে প্যাটার্নটি প্রথমবার যেখানে পাওয়া যায় তার একটি 'Match object' দেয়। |
| split() | প্যাটার্ন অনুযায়ী স্ট্রিংটিকে ভাগ করে একটি তালিকা তৈরি করে। |
| sub() | স্ট্রিংয়ের এক বা একাধিক ম্যাচিং অংশকে অন্য কোনো স্ট্রিং বা টেক্সট দিয়ে প্রতিস্থাপন করে। |
| match() | স্ট্রিংয়ের শুধুমাত্র শুরুতেই প্যাটার্নটি আছে কি না তা পরীক্ষা করে। |
| finditer() | সকল ম্যাচিং অংশকে একটি ইটারেটর (Iterator) হিসেবে রিটার্ন করে। |

**৩. Metacharacters - এর তালিকা তৈরি কর।**

**উত্তর:** মেটাক্যারেক্টার হলো এমন কিছু বিশেষ চিহ্ন যাদের নিজস্ব সাধারণ অর্থের বাইরে Regex-এ বিশেষ অর্থ থাকে। নিচে এদের তালিকা ও কাজ দেওয়া হলো:

- [] (Square Brackets): ক্যারেক্টারের একটি সেট নির্দেশ করে। যেমন: [a-m]।
- \ (Backslash): স্পেশাল সিকুয়েন্স বোঝাতে বা বিশেষ চিহ্নকে সাধারণ বর্ণে রূপান্তর করতে ব্যবহৃত হয়।
- . (Dot): নিউলাইন বাদে যেকোনো একটি ক্যারেক্টারকে নির্দেশ করে।
- ^ (Caret): স্ট্রিং বা লাইনের শুরু বোঝাতে ব্যবহৃত হয়।
- $ (Dollar): স্ট্রিং বা লাইনের শেষ বোঝাতে ব্যবহৃত হয়।
- * (Star): কোনো ক্যারেক্টার ০ (শূন্য) বা তার বেশিবার থাকলে তা ম্যাচ করে।
- + (Plus): কোনো ক্যারেক্টার ১ বা তার বেশিবার থাকলে তা ম্যাচ করে।
- ? (Question Mark): কোনো ক্যারেক্টার ০ বা ১ বার থাকলে তা ম্যাচ করে।

- {} (Braces): নির্দিষ্ট সংখ্যক বার কোনো ক্যারেক্টার থাকলে তা বোঝায়। যেমন: a{2} মানে পরপর দুটি 'a'।
- | (Pipe): 'অথবা' (OR) লজিক হিসেবে কাজ করে। যেমন: cat|dog।
- () (Parentheses): গ্রুপ বা ক্যারেক্টারগুলোকে একত্রে দলভুক্ত করার জন্য ব্যবহৃত হয়।

## রচনামূলক প্রশ্ন

**১. RegEx ব্যবহার করে নির্দিষ্ট Word খোঁজা ও গণনা করার Python program লেখ।** [বাকাশিবো-২০২৩ (অনুষ্ঠিত-২৪)

**উত্তর:** এই প্রোগ্রামে আমরা re.findall() মেথড ব্যবহার করব, যা নির্দিষ্ট প্যাটার্নের সবকটি ম্যাচকে একটি লিস্ট হিসেবে রিটার্ন করে। এরপর len() ফাংশন দিয়ে তার সংখ্যা গণনা করব।

```
import re

text = "Python is powerful, Python is easy, and Python is popular."

word_to_find = "Python"

# RegEx ব্যবহার করে শব্দ খোঁজা

matches = re.findall(word_to_find, text)

# ফলাফল প্রদর্শন

print(f"খোঁজা হয়েছে: {word_to_find}")

print(f"কতবার পাওয়া গেছে: {len(matches)}")
```

**২. Regex ব্যবহারপূর্বক নিম্নোক্ত String হতে Number-গুলোকে Extract করার জন্য Program লেখ।** [বাকাশিবো-২০২৩ (পরি)]

**উত্তর:** স্ট্রিং থেকে সংখ্যা বা ডিজিট বের করার জন্য আমরা স্পেশাল সিকুয়েন্স \d+ ব্যবহার করি।

```
import re

my_string = "The price of item 1 is 500 dollars and item 2 is 1250 dollars."
```

```
# \d+ মানে হলো এক বা একাধিক সংখ্যা (digits)

numbers = re.findall(r'\d+', my_string)

print("নিষ্কাশিত সংখ্যাগুলো হলো:", numbers)
```

**৩. উদাহরণসহ পাইথন রেগুলার এক্সপ্রেশন-এর বিল্ট-ইন মেথডগুলো বর্ণনা কর।**

অথবা, **উদাহরণসহ পাইথন রেগুলার এক্সপ্রেশনগুলো বর্ণনা কর।** [বাকাশিবো-২০২২ (অনুষ্ঠিত-২৩]

**উত্তর:** পাইথনের re মডিউলে বেশ কিছু গুরুত্বপূর্ণ মেথড রয়েছে। নিচে উদাহরণসহ তাদের বর্ণনা দেওয়া হলো:

১. re.findall()

এটি স্ট্রিংয়ের মধ্যে প্যাটার্নের সাথে মিলে যায় এমন সব অংশকে একটি লিস্ট (List) আকারে রিটার্ন করে।

- উদাহরণ:

```
import re

res = re.findall("ai", "The rain in Spain")

print(res) # আউটপুট: ['ai', 'ai']
```

২. re.search()

এটি পুরো স্ট্রিংয়ে প্যাটার্নটি খোঁজে এবং প্রথম যেখানে মিল পায় তার একটি 'Match Object' রিটার্ন করে। মিল না পেলে None রিটার্ন করে।

- উদাহরণ:

```
import re

res = re.search("\s", "Hello World") # প্রথম স্পেসটি খুঁজবে

print("স্পেস পাওয়া গেছে পজিশন:", res.start())
```

৩. re.split()

এই মেথডটি প্যাটার্ন অনুযায়ী স্ট্রিংটিকে ভেঙে বিভিন্ন অংশে ভাগ করে একটি লিস্ট তৈরি করে।

- উদাহরণ:

```
import re

res = re.split("\s", "Python is fun")

print(res) # আউটপুট: ['Python', 'is', 'fun']
```

৪. re.sub()

এটি স্ট্রিংয়ের এক বা একাধিক ম্যাচিং অংশকে অন্য কোনো শব্দ বা ক্যারেক্টার দিয়ে প্রতিস্থাপন (Replace) করে।

- উদাহরণ:

```
import re

res = re.sub("\s", "-", "Python is fun")

print(res) # আউটপুট: Python-is-fun
```

৫. re.match()

এটি শুধুমাত্র স্ট্রিংয়ের শুরুতে প্যাটার্নটি আছে কি না তা যাচাই করে। শুরুতে না থাকলে এটি কিছুই রিটার্ন করে না (এমনকি যদি স্ট্রিংয়ের মাঝখানে মিল থাকে তাও নয়)।

# অধ্যায় ১১ অ্যাপ্লিকেশন সফটওয়্যার

এই অধ্যায়ের প্রধান সূচিপত্র (contants)


| বিষয় | পৃষ্ঠা নং |
|---|---|
| ❖ সূত্রাবলী ও শর্টকার্ট টেকনিক | - |
| ❖ অতি সংক্ষিপ্ত প্রশ্নাবলি | ৭৪ |
| ❖ সংক্ষিপ্ত প্রশ্নাবলি | ৭৪ |
| ❖ রচনামূলক প্রশ্নাবলি | ৭৭ |


## বোর্ড প্রশ্নাবলির বিশ্লেষন (Board Questions Analysis)

এই অধ্যায় থেকে বিগত তিন বছরের ডিপ্লোমা ইন ইঞ্জিনিয়ারিং, ডিপ্লোমা ইন টেক্সটাইল, ডিপ্লোমা ইন এগ্রিকালচার, ডিপ্লোমা ইন মেরিন পরীক্ষায় মোট প্রশ্ন....নিচে 'Board Questions Analysis' অংশে এই অধ্যায় থেকে কোন সালে কোন বোর্ডে কতটি প্রশ্ন হয়েছে তা দেখানো হলো।

| সাল / বোর্ড | অতিসংক্ষিপ্ত | সংক্ষিপ্ত | রচনামূলক |
|---|---|---|---|
| ২০২৪ | -- | -- | -- |
| ২০২৩ | ১টি | -- | ১টি |
| ২০২২ | -- | ১টি | -- |

## অতি সংক্ষিপ্ত প্রশ্ন

**১. অ্যাপ্লিকেশন সফটওয়্যার বলতে কী বুঝায়?**

**উত্তর:** অ্যাপ্লিকেশন সফটওয়্যার (Application Software) হলো এক ধরনের কম্পিউটার প্রোগ্রাম যা ব্যবহারকারীকে কোনো নির্দিষ্ট কাজ (যেমন: চিঠি লেখা, হিসাব করা, ছবি এডিট করা বা ইন্টারনেট ব্রাউজ করা) সম্পন্ন করতে সাহায্য করে। এটি মূলত ব্যবহারকারীর সরাসরি প্রয়োজনে ব্যবহৃত হয়। সিস্টেম সফটওয়্যার (যেমন: উইন্ডোজ) কম্পিউটার চালায়, আর অ্যাপ্লিকেশন সফটওয়্যার ব্যবহারকারীর কাজ করে দেয়। উদাহরণস্বরূপ: MS Word, Google Chrome, Photoshop ইত্যাদি।

**২. অ্যাপ্লিকেশন সফটওয়্যারের প্রকারভেদগুলো লেখ।** [বাকাশিবো-২০২৩ (পরি)]

**উত্তর:** কাজের ধরন এবং ব্যবহারের ক্ষেত্র অনুযায়ী অ্যাপ্লিকেশন সফটওয়্যারকে প্রধানত দুই ভাগে ভাগ করা যায়:

সাধারণ বা প্যাকেজ সফটওয়্যার (General Purpose / Package Software): যেসব সফটওয়্যার সাধারণ ব্যবহারকারীদের সাধারণ কাজের জন্য তৈরি করা হয় এবং বাজারে কিনতে পাওয়া যায়।

- উদাহরণ: মাইক্রোসফট অফিস (Word, Excel), মিডিয়া প্লেয়ার, ওয়েব ব্রাউজার।
- বিশেষ বা কাস্টমাইজড সফটওয়্যার (Customized / Tailor-made Software): যেসব সফটওয়্যার কোনো নির্দিষ্ট প্রতিষ্ঠান বা ব্যক্তির নিজস্ব প্রয়োজন মেটানোর জন্য বিশেষভাবে তৈরি করা হয়।
- উদাহরণ: ব্যাংকিং সফটওয়্যার, হাসপাতালের ম্যানেজমেন্ট সিস্টেম, কোনো প্রতিষ্ঠানের নিজস্ব হিসাবরক্ষণ সফটওয়্যার।

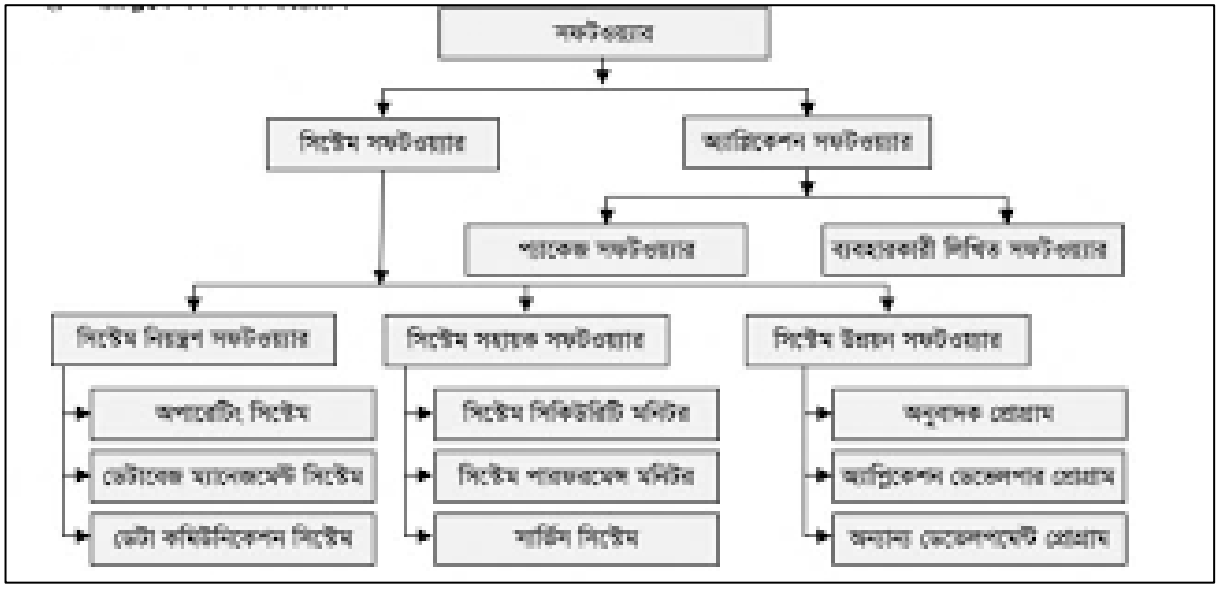
এছাড়াও কাজের ক্ষেত্র অনুযায়ী আরও কিছু জনপ্রিয় প্রকারভেদ হলো:

- ওয়ার্ড প্রসেসিং সফটওয়্যার: (যেমন: MS Word)
- স্প্রেডশিট সফটওয়্যার: (যেমন: MS Excel)
- ডেটাবেস সফটওয়্যার: (যেমন: Oracle, MySQL)
- গ্রাফিক্স সফটওয়্যার: (যেমন: Adobe Photoshop)

## সংক্ষিপ্ত প্রশ্ন

**১. অ্যাপ্লিকেশন সফটওয়্যারের শ্রেণিবিভাগ তৈরি কর।**

**উত্তর:**অ্যাপ্লিকেশন সফটওয়্যারকে মূলত তাদের ব্যবহারের উদ্দেশ্য, প্রাপ্তিযোগ্যতা এবং কাজের ধরন অনুযায়ী প্রধানত দুই ভাগে ভাগ করা যায়। নিচে বিস্তারিত আলোচনা করা হলো:




**২. অ্যাপ্লিকেশন সফটওয়্যারের প্রকারভেদগুলো গুলো লেখ।** । [বাকাশিবো-২০২২ (অনুষ্ঠিত-২৩)]

**উত্তর:** অ্যাপ্লিকেশন সফটওয়্যারকে সাধারণত তাদের ব্যবহারের ক্ষেত্র এবং উদ্দেশ্য অনুযায়ী প্রধানত দুইটি ভাগে ভাগ করা হয়। নিচে বিস্তারিত বর্ণনা দেওয়া হলো:

সাধারণ উদ্দেশ্যমূলক সফটওয়্যার (General Purpose / Package Software)

এই সফটওয়্যারগুলো সাধারণ ব্যবহারকারীদের সাধারণ কাজের জন্য তৈরি করা হয়। এগুলো বাজারে সাধারণত প্যাকেজ হিসেবে পাওয়া যায়। এর প্রধান শাখাগুলো হলো:

- ওয়ার্ড প্রসেসিং সফটওয়্যার: ডকুমেন্ট, চিঠি বা রিপোর্ট তৈরির জন্য। (যেমন: MS Word, Google Docs)।
- স্প্রেডশিট সফটওয়্যার: জটিল গাণিতিক হিসাব ও বাজেট তৈরির জন্য। (যেমন: MS Excel)।
- ডেটাবেস সফটওয়্যার: তথ্য সংগ্রহ ও সংরক্ষণের জন্য। (যেমন: Oracle, MS Access, MySQL)।
- প্রেজেন্টেশন সফটওয়্যার: তথ্যকে স্লাইডের মাধ্যমে উপস্থাপনের জন্য। (যেমন: MS PowerPoint)।
- গ্রাফিক্স ও মাল্টিমিডিয়া: ছবি এডিট ও গান-ভিডিও চালানোর জন্য। (যেমন: Photoshop, VLC Player)।
- ওয়েব ব্রাউজার: ইন্টারনেট ব্যবহারের জন্য। (যেমন: Chrome, Firefox)।

বিশেষ উদ্দেশ্যমূলক বা কাস্টমাইজড সফটওয়্যার (Customized / Special Purpose Software)

কোনো নির্দিষ্ট প্রতিষ্ঠান বা ব্যক্তির বিশেষ চাহিদা অনুযায়ী যখন কোনো সফটওয়্যার তৈরি করা হয়, তখন তাকে কাস্টমাইজড সফটওয়্যার বলে। এর উদাহরণগুলো হলো:

- ব্যাংকিং সফটওয়্যার: ব্যাংকের লেনদেন পরিচালনার জন্য।
- হসপিটাল ম্যানেজমেন্ট সফটওয়্যার: হাসপাতালের রোগী ও ডাক্তারদের তথ্য ব্যবস্থাপনার জন্য।
- ইনভেন্টরি সিস্টেম: কোনো দোকানের মালামাল বা স্টকের হিসাব রাখার জন্য।
- রেলওয়ে রিজার্ভেশন সিস্টেম: ট্রেনের টিকিট বুকিং করার জন্য।

ব্যবহারের ধরন অনুযায়ী আরও কিছু প্রকারভেদ:

- ইউটিলিটি সফটওয়্যার: সিস্টেমের সুরক্ষা ও রক্ষণাবেক্ষণের কাজ করে (যেমন: এন্টিভাইরাস)।
- এন্টারপ্রাইজ সফটওয়্যার: বড় প্রতিষ্ঠানের ব্যবসায়িক কার্যক্রম পরিচালনার জন্য (যেমন: ERP)।

**৩. অ্যাপ্লিকেশন সফটওয়্যারের কাজ লেখ।**

**উত্তর:** অ্যাপ্লিকেশন সফটওয়্যার মূলত ব্যবহারকারীর সরাসরি প্রয়োজন মেটাতে কাজ করে। এর প্রধান কাজগুলো হলো:

- তথ্য ব্যবস্থাপনা (Information Management): বিশাল পরিমাণ ডেটা বা তথ্যকে সুশৃঙ্খলভাবে সাজিয়ে রাখা এবং প্রয়োজনে তা খুঁজে বের করা।
- ডকুমেন্ট তৈরি: চিঠি, রিপোর্ট, নোটিশ বা যেকোনো ধরনের লিখিত দলিল তৈরি ও ফরমেটিং করা।
- হিসাব-নিকাশ: গাণিতিক ও যৌক্তিক সমস্যার সমাধান এবং জটিল আর্থিক হিসাব সহজে সম্পন্ন করা।
- যোগাযোগ: ইমেইল আদান-প্রদান, ভিডিও কনফারেন্সিং বা সোশ্যাল মিডিয়ার মাধ্যমে যোগাযোগ রক্ষা করা।
- শিক্ষা ও গবেষণা: ইন্টারনেটের মাধ্যমে তথ্য সংগ্রহ এবং বিভিন্ন শিক্ষামূলক সিমুলেশন বা প্রেজেন্টেশন তৈরি করা।
- বিনোদন: গান শোনা, মুভি দেখা বা গেম খেলার মাধ্যমে বিনোদন প্রদান করা।
- ডিজাইন ও সৃষ্টিশীল কাজ: স্থাপত্য নকশা (CAD), গ্রাফিক্স ডিজাইন এবং ভিডিও এডিটিং এর মতো সৃষ্টিশীল কাজ সম্পন্ন করা।

## রচনামূলক প্রশ্ন

**১. অ্যাপ্লিকেশন সফটওয়্যারের কার্যাবলি লেখ।** [বাকাশিবো-২০২৩ (অনুষ্ঠিত-২৪)]

**উত্তর:** অ্যাপ্লিকেশন সফটওয়্যার মূলত ব্যবহারকারীর নির্দিষ্ট কাজগুলো সম্পন্ন করার জন্য ডিজাইন করা হয়। এর প্রধান কার্যাবলি নিচে দেওয়া হলো:

- তথ্য ব্যবস্থাপনা: বিশাল পরিমাণ ডেটা বা তথ্যকে সুশৃঙ্খলভাবে সংরক্ষণ, বিন্যাস এবং প্রয়োজনে তা দ্রুত খুঁজে বের করতে সাহায্য করে।
- ডকুমেন্ট তৈরি ও ফরমেটিং: প্রতিবেদন, চিঠি, সিভি বা যেকোনো ধরনের টেক্সট ভিত্তিক ফাইল তৈরি, এডিট এবং প্রিন্ট করতে সহায়তা করে।
- গাণিতিক হিসাব-নিকাশ: জটিল আর্থিক হিসাব, বাজেট তৈরি এবং বৈজ্ঞানিক সমস্যার সমাধান দ্রুত ও নির্ভুলভাবে সম্পন্ন করে।
- যোগাযোগ রক্ষা: ইন্টারনেট ব্যবহার করে ইমেইল আদান-প্রদান, ভিডিও কনফারেন্সিং এবং সামাজিক যোগাযোগ মাধ্যমে যুক্ত হতে সাহায্য করে।
- সৃষ্টিশীল কাজ: ছবি এডিট করা, গ্রাফিক্স ডিজাইন, ভিডিও এডিটিং এবং আর্কিটেকচারাল নকশা তৈরিতে ব্যবহৃত হয়।
- শিক্ষা ও বিনোদন: মাল্টিমিডিয়া কন্টেন্ট (গান, মুভি) চালানো এবং শিক্ষামূলক প্রেজেন্টেশন ও সিমুলেশন তৈরিতে গুরুত্বপূর্ণ ভূমিকা পালন করে।

**২. অ্যাপ্লিকেশন সফটওয়্যারের প্রকারভেদগুলো বর্ণনা কর।**

**উত্তর:** অ্যাপ্লিকেশন সফটওয়্যারকে তাদের ব্যবহারের উদ্দেশ্য ও ক্ষেত্র অনুযায়ী প্রধানত দুই ভাগে ভাগ করা যায়:
ক) সাধারণ উদ্দেশ্যমূলক সফটওয়্যার (General Purpose / Package Software): এগুলো সাধারণ ব্যবহারকারীদের সাধারণ কাজের জন্য তৈরি। এগুলো বাজারে প্যাকেজ আকারে কিনতে পাওয়া যায়। যেমন:

- ওয়ার্ড প্রসেসর: MS Word.
- স্প্রেডশিট: MS Excel.
- ডেটাবেস: MySQL, Oracle.
- ওয়েব ব্রাউজার: Google Chrome.

খ) বিশেষ উদ্দেশ্যমূলক বা কাস্টমাইজড সফটওয়্যার (Customized / Tailor-made Software): কোনো

নির্দিষ্ট প্রতিষ্ঠানের চাহিদা অনুযায়ী যখন কোনো সফটওয়্যার বিশেষভাবে তৈরি করা হয়। যেমন:

- ব্যাংকিং সফটওয়্যার: ব্যাংকের লেনদেনের জন্য।
- হসপিটাল ম্যানেজমেন্ট সফটওয়্যার: রোগীর তথ্য সংরক্ষণের জন্য।
- ইনভেন্টরি সিস্টেম: দোকানের স্টক ম্যানেজমেন্টের জন্য।

**৩. অ্যাপ্লিকেশন সফটওয়্যারের ব্যবহার বর্ণনা কর।**

**উত্তর:** বর্তমান সময়ে জীবনের প্রতিটি ক্ষেত্রে অ্যাপ্লিকেশন সফটওয়্যারের ব্যাপক ব্যবহার রয়েছে। উল্লেখযোগ্য কিছু ব্যবহার নিচে দেওয়া হলো:

- অফিস ব্যবস্থাপনায়: দাপ্তরিক চিঠিপত্র লেখা, ডাটা এন্ট্রি করা এবং কর্মীদের বেতনের হিসাব রাখতে এর ব্যবহার অপরিহার্য।
- শিক্ষা ক্ষেত্রে: অনলাইন ক্লাস পরিচালনা, ডিজিটাল পদ্ধতিতে শিক্ষা উপকরণ (স্লাইড) তৈরি এবং স্কুল-কলেজের রেজাল্ট শিট তৈরির জন্য ব্যবহৃত হয়।
- চিকিৎসা ক্ষেত্রে: রোগীর প্রেসক্রিপশন তৈরি, এক্স-রে বা প্যাথলজি রিপোর্ট বিশ্লেষণ এবং হাসপাতালের সামগ্রিক ব্যবস্থাপনায় এটি ব্যবহৃত হয়।
- ব্যাংকিং ও ব্যবসায়: অনলাইন টাকা লেনদেন, এটিএম বুথ পরিচালনা এবং বড় বড় শিল্প কারখানার প্রোডাকশন ও স্টকের হিসাব রাখতে এটি ব্যবহৃত হয়।
- যোগাযোগ ও বিনোদন: ভিডিও কলিং অ্যাপস (Zoom, Skype) এবং বিনোদনের জন্য ইউটিউব বা মিডিয়া প্লেয়ারের ব্যবহার এর প্রকৃষ্ট উদাহরণ।
- ডিজাইন ও ইঞ্জিনিয়ারিং: ভবন বা যন্ত্রপাতির নকশা তৈরিতে (AutoCAD) এবং বিজ্ঞাপন বা সিনেমার জন্য বিশেষ ইফেক্ট তৈরিতে এর ব্যবহার ব্যাপক।