ষষ্ঠ অধ্যায়

প্রোগ্রামিংয়ের মাধ্যমে সমস্যার সমাধান

এ অধ্যায় শেষে আমরা...

- সমস্যা সমাধান সম্পর্কে জানতে পারব;
- প্রোগ্রামিং ভাষা সম্পর্কে জানতে পারব;
- পাইথন প্রোগ্রামিং ভাষা ব্যবহার করে কোডিং করতে পারব;
- পাইথন প্রোগ্রামিং ভাষা ব্যবহার করে সাধারণ সমস্যার সমাধান করতে পারব।

সমস্যা সমাধানে প্রোগ্রামিং

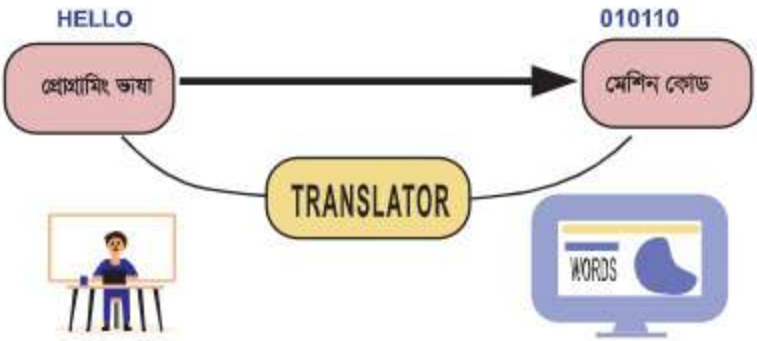
বাস্তবজীবনে আমরা নানান ধরনের সমস্যার সম্মুখীন হই। আমাদেরকে ঐ সকল সমস্যার সমাধান করে সামনের দিকে এগিয়ে যেতে হয়। সমস্যা সমাধান হল চ্যালেঞ্জ বা বাধা শনাক্ত করার প্রক্রিয়া, যাতে সমস্যার প্রকৃতি বোঝা যায় এবং এটি অতিক্রম করার উপায় খুঁজে বের করা হয়। এর সাথে পরিপার্শ্বিক পরিস্থিতি বিশ্লেষণ করা, চিন্তা করা এবং সমস্যার সম্ভাব্য সমাধান খুঁজে বের করে নিয়ে আসার বিষয়টি জড়িত। এই প্রক্রিয়াতে সাধারণত সমস্যা সংজ্ঞায়িত করা, সমস্যার সম্ভাব্য সমাধান নিয়ে চিন্তাভাবনা করা, সেই বিকল্প সমাধানগুলো মূল্যায়ন করা এবং তারপরে সেরা সমাধানটি বাস্তবায়ন করা অন্তর্ভুক্ত থাকে।

প্রোগ্রামিং ভাষা

আমরা মনের ভাব প্রকাশের জন্য বিভিন্ন ভাষা ব্যবহার করি যেমন- বাংলা, ইংরেজি, ফ্রেঞ্চ, ল্যাটিন, স্প্যানিশ ইত্যাদি। কিন্তু কম্পিউটার আমাদের এই সকল ভাষা সরাসরি বুঝতে পারে না। কম্পিউটারকে যেকোনো নির্দেশ দিতে গেলে কম্পিউটার বুঝতে পারে এমন ভাষায় নির্দেশ লিখতে হয়। কম্পিউটারসহ যেকোনো ইলেক্ট্রনিক ডিভাইস শুধুমাত্র ০ আর ১ কে বুঝতে পারে।

কিন্তু শুধু ০ আর ১ দিয়ে নিজেদের নির্দেশগুলো লিখে ফেলাও আমাদের জন্য কঠিন। আমরা মানুষরা সাধারণত যেসকল ভাষা ব্যবহার করি, সেগুলো শুধুমাত্র ০ আর ১ দিয়ে তৈরি নয়। তাহলে কম্পিউটারের সাথে কীভাবে আমরা যোগাযোগ করব? এমন কিছু ভাষা আছে, যেখানে ওই ভাষার রীতিনীতি অনুসরণ করে নির্দেশ লিখলে কম্পিউটার সেই ভাষাকে সহজেই নিজের বোঝার উপযোগী হিসাবে রূপান্তর করে নিয়ে নির্দেশগুলো বুঝতে পারে। এই ভাষাগুলোকে বলা হয় প্রোগ্রামিং ভাষা। অনেক রকম প্রোগ্রামিং ভাষা আছে। যেমন- সি, সি++, পাইথন, জাভা ইত্যাদি।

আমরা যেকোনো একটি প্রোগ্রামিং ভাষা শিখলে সেই ভাষার মাধ্যমে কম্পিউটারকে প্রয়োজনমত বিভিন্ন নির্দেশ দিতে পারব। আমাদের কম্পিউটারের একটি সফটওয়্যার অ্যাপ্লিকেশনে প্রথমে আমরা নির্দিষ্ট কোনো প্রোগ্রামিং ভাষায় নির্দেশগুলো লিখে ফেলব। কম্পিউটারে এমন একটি রূপান্তর ব্যবস্থা থাকে যা সেই প্রোগ্রামিং ভাষার নির্দেশগুলোকে মেশিন কোডে রূপান্তর করে।



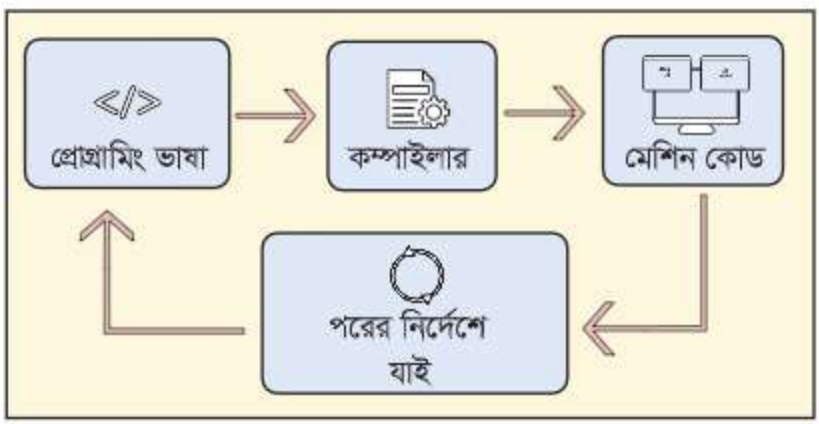
মেশিন কোড বা মেশিন ভাষা কি?

মূলত ০ আর ১ এর সমন্বয়ে তৈরি বাইনারি কোডকেই মেশিন কোড বা মেশিন ভাষা বলা হয়, যা আমাদের কম্পিউটার সরাসরি বুঝতে পারে। কাজেই প্রোগ্রামিং ভাষা ও মেশিন কোডের মধ্যে রূপান্তরের কাজটি অনুবাদক নামক এক ধরনের প্রোগ্রামের সাহায্যে করা হয়ে থাকে। অনুবাদকের মাধ্যমে মেশিন কোডে রূপান্তরের ফলে আমাদের নির্দেশগুলো কম্পিউটার বুঝতে পারবে এবং সেই নির্দেশ অনুসরণ করে নির্ধারিত কাজ সম্পন্ন করতে পারবে।

কম্পিউটারে থাকা প্রোগ্রামিং ভাষার রূপান্তর ব্যবস্থা বা অনুবাদক প্রোগ্রাম আবার দুই রকমের হতে পারে-

ক) কম্পাইলার (Compiler) : কিছু রূপান্তর ব্যবস্থায় আমরা যতগুলো নির্দেশ দিব, যদি নির্দেশগুলো নির্ভুল হয় তাহলে সবগুলো নির্দেশ একসাথে মেশিন কোডে রূপান্তরিত হবে। এই রূপান্তর ব্যবস্থাকে বলা হয় কম্পাইল (Compile) করা। আর যে সফটওয়্যার রূপান্তর করল, সেই রূপান্তরকারী প্রোগ্রামটি হচ্ছে একটি কম্পাইলার (Compiler)। তবে কম্পাইলার যদি পুরো নির্দেশের কোথাও ভুল পায়, তাহলে রূপান্তর করতে পারে না। সবগুলো নির্দেশ নির্ভুল দিলে তখনই রূপান্তরের কাজটি করতে পারে।

খ) ইন্টারপ্রেটার (Interpreter) : কিছু রূপান্তর ব্যবস্থায় আমরা যত নির্দেশই দেই না কেন, সব একসাথে রূপান্তর হবে না। একটি একটি করে নির্দেশ ধারাবাহিকভাবে রূপান্তরিত হতে থাকবে। এই রূপান্তরব্যবস্থাকে বলা হয় ইন্টারপ্রেট (Interpret) করা। আর যে রূপান্তরের কাজটি করছে তাকে বলা হয় ইন্টারপ্রেটার (Interpreter)। ইন্টারপ্রেটার একটি একটি করে নির্দেশ রূপান্তর করতে থাকবে। কোনো নির্দেশে ভুল পেলে সেই নির্দেশে আসার পর থেমে যাবে।



কোন প্রোগ্রামিং ভাষাটি শিখব?

বর্তমানে প্রচলিত প্রোগ্রামিং ভাষাগুলোর মধ্য থেকে যেকোনো একটি প্রোগ্রামিং ভাষা প্রথমে শিখলেই হবে। কারণ সব প্রোগ্রামিং ভাষার মূল গঠন একই রকমের, শুধু ভাষাগুলোতে বিভিন্ন নির্দেশ লেখার নিয়মে একটু-আধটু ভিন্নতা থাকে। যেমন সি প্রোগ্রামিং ভাষায় প্রতিটি নির্দেশ (স্টেটমেন্ট) শেষ হবার পর একটি সেমিকোলন চিহ্ন দিতে হয়, কিন্তু পাইথনে এই কাজ করতে হয় না। এরকম কিছু পার্থক্য থাকলেও চিন্তার কিছু নেই। তুমি একটি প্রোগ্রামিং ভাষা শিখে নিলে এরপর তোমার জন্য অন্য প্রোগ্রামিং ভাষাগুলো শেখা খুবই সহজ হয়ে যাবে। তবে সাধারণভাবে প্রোগ্রামিং ভাষা পছন্দের বিষয়টি কাজের ধরন এবং কিছুটা আগ্রহের উপরও নির্ভর করে। প্রোগ্রামিং যারা নতুন শিখতে চায় তাদের জন্য পাইথন একটি দুর্দান্ত সূচনা হতে পারে।

সহজে শেখার জন্য পাইথন বেশ মজার একটি প্রোগ্রামিং ভাষা। এটি শিক্ষানবিস-বান্ধব, এর একটি সাধারণ সিনট্যাক্স রয়েছে এবং এটি ওয়েব ডেভেলপমেন্ট, ডেটা সায়েন্স এবং অটোমেশনের মতো ক্ষেত্রেসমূহে ব্যাপকভাবে ব্যবহৃত হয়।

পাইথনে প্রোগ্রামিংয়ের যাত্রা শুরু:

পাইথন ভাষার নির্দেশ বা কোড লেখার জন্য প্রথমে প্রোগ্রামিং পরিবেশ তৈরির জন্য কিছু কাজ করতে হবে-

১. পাইথন অ্যাপ্লিকেশন ডাউনলোড কে আমাদের কম্পিউটারে ইনস্টল করতে হবে। এই লিংকে থেকে পাইথন ডাউনলোড করা যায়।

https://www.python.org/downloads/

এরপর সেখান থেকে সবচেয়ে সাম্প্রতিক ভার্সন নামিয়ে নেই।

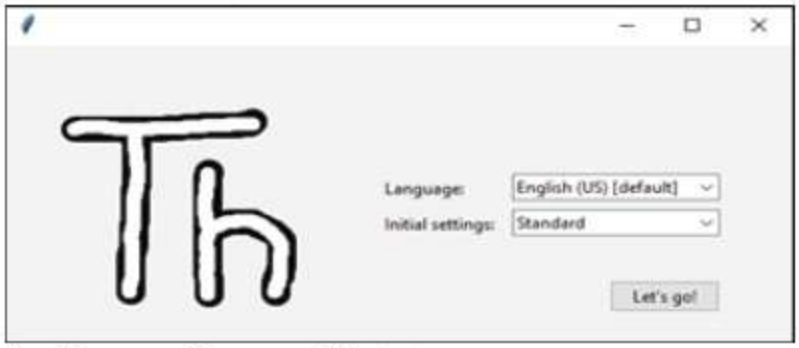
২. অ্যাপ্লিকেশন নামানো হয়ে গেলে এটি ইনস্টল করে ফেলি। ইনস্টলের সময় নিচের ছবির মতো একটি উইন্ডো দেখতে পাব-

এখন ইনস্টল উইন্ডোর নিচে থাকা অপশনগুলো ক্লিক করে টিক চিহ্ন দিয়ে দিব। তারপর Install Now অপশনে ক্লিক করব। এ সময় ইনস্টল হবার অনুমতি চাইলে সেটাও অনুমতি দিয়ে দিব।

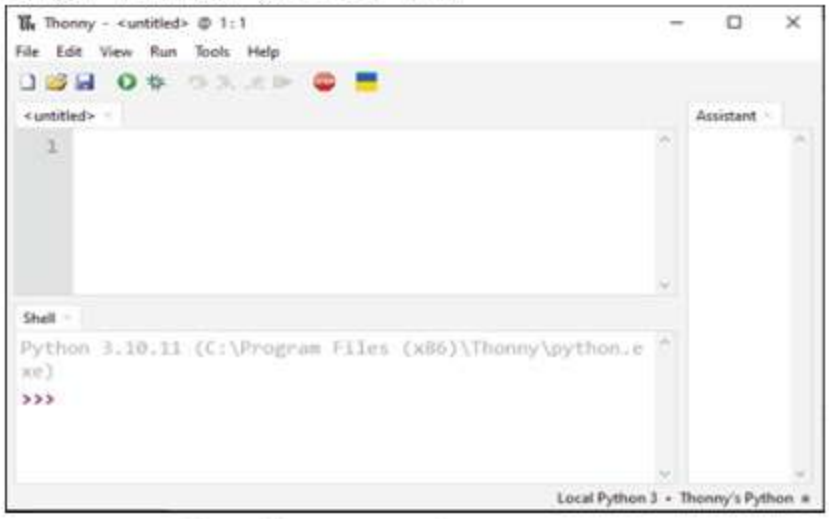
৩. এরপর আমাদের মেসেজ দেখাবে যে আমাদের সেটআপ সফল হয়েছে।

৪. পাইথন আমাদের কম্পিউটারে যুক্ত হলো। কিন্তু আমাদের আরেকটি সফটওয়্যার এপ্লিকেশন লাগবে যেখানে আমরা আমাদের নির্দেশ লিখে কম্পিউটারকে বুঝিয়ে দিব। সেজন্য এই লিংকে যাই-https://thonny-.org/ ; এই লিংক থেকে Thonny সফটওয়্যারটি নামিয়ে নেই ও ইনস্টল করে ফেলি।

৫. Thonny সফটওয়্যার এরপর চালু করি। নিচের মতো উইন্ডো দেখতে পাব-



Let's go! বাটনে ক্লিক করলে নিচের মতো উইন্ডো আসবে-

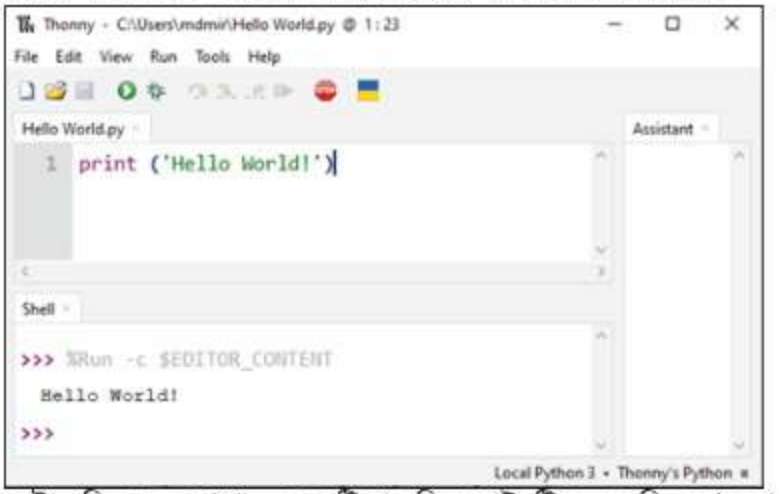


৬. এই উইন্ডোতে থাকা গুরুত্বপূর্ণ কয়েকটি অংশ বুঝে নেই-

৭. এবারে আমরা একটা প্রোগ্রাম লিখি, যার কাজ হবে আউটপুট হিসেবে Hello World! প্রিন্ট করা। আউটপুট হিসেবে কোনো কিছু প্রিন্ট করতে হলে print () ব্যবহার করতে হয়। আমরা যে টেক্সট প্রিন্ট ক তে চাই, সেটা print () এর ভিতরে Single Quotation (' ') দিয়ে তার মধ্যে লিখব। তাহলে Hello World! প্রিন্ট করতে লেখি-

print (' Hello World! ')

এরপর রান বাটনে ক্লিক করলে নিচে আউটপুট হিসেবে Hello World! লেখা উঠবে।



৮. এবারে সেভ বাটনে ক্লিক করে প্রোগ্রামের একটি নাম দিয়ে ফাইলটি সেভ করি। তখন আমাদের ফাইলের নামও প্রদর্শন করবে প্রোগ্রামের উপরে।

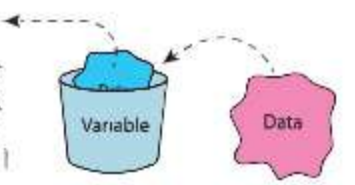
টেক্সট প্রদর্শন করা

আমরা যেমন ইংরেজি টেক্সট প্রিন্ট করলাম, একই ভাবে বাংলা লিখেও তুমি সেটা প্রিন্ট করতে পারবে। যেমন, নিচের লাইন লিখে রান করে দেখ তো কী দেখা যায়-

print (' আমি বাংলাদেশকে ভালবাসি ')

কাজেই print() ফাংশন মনিটরের স্ক্রিনে বা অন্যান্য স্ট্যান্ডার্ড আউটপুট ডিভাইসে নির্দিষ্ট বার্তা প্রিন্ট করে। বার্তাটি একটি স্ট্রিং বা অন্য কোনো বস্তু হতে পারে, স্ক্রিনে লেখার আগে অবজেক্টটি একটি স্ট্রিংয়ে রূপান্তরিত হবে।

প্রোগ্রামে চলক বা ভ্যারিয়েবলের ব্যবহার (Variable)



কোনো উপাত্ত বা ডেটা যদি প্রোগ্রামের ভিতর সঞ্চয় করতে হয় তাহলে আমরা চলক বা ভ্যারিয়েবল ব্যবহার করতে পারব। চলক বা ভ্যারিয়েবল হলো একটি বক্সের মতো, যার ভিতরে নির্দিষ্ট কোনো একটি ডেটা জমা রাখা যায়।

ভ্যারিয়েবল শব্দটির অর্থ পরিবর্তনশীল, এটি চলক নামেও পরিচিত। কাজেই আমরা চাইলে প্রোগ্রামে একটি লাইনে ভ্যারিয়েবলের মধ্যে একটি ডেটা জমা রাখার পর অন্য লাইনে সেই ডেটা পরিবর্তন করে ভিন্ন আরেকটি ডেটা জমা রাখতে পারি। উল্লেখ্য পাইথন প্রোগ্রামে চলক বা ভরিয়েবলের কোনো নির্দিষ্ট ডেটা টাইপ ঘোষণা করার প্রয়োজন নেই, এমনকি ডেটা টাইপ সেট করার পরেও যে কোনো সময় তা পরিবর্তন করা যায়।

চলক বা ভ্যারিয়েবলের নামকরণ

আমাদের সবার যেমন এক একটি নির্দিষ্ট নাম আছে এবং এই নাম দিয়ে আমরা একে অপরকে চিনতে পারি, ঠিক তেমনি প্রোগ্রামে প্রতিটি ভ্যারিয়েবলের একটি নির্দিষ্ট নাম দিতে হয়, যে নাম দিয়ে পুরো প্রোগ্রামে আমরা ভ্যারিয়েবলটি চিনতে পারব ও তা ব্যবহা করতে পারব। যেমন, আমরা যদি চাই number নামে একটি ভ্যারিয়েবল তৈরি করব, যেখানে ভ্যারিয়েবলের মান হিসেবে 9 জমা রাখতে চাই। তাহলে আমরা লিখব-

number=9

ভ্যারিয়েবলের নাম দেওয়ার সময় আমাদের কিছু নিয়মকানুন মেনে চলতে হবে। যথা-

১। ভ্যারিয়েবলের নাম সবসময় একটি শব্দ হবে। অর্থাৎ একাধিক শব্দ দিয়ে আমরা ভ্যারিয়েবলের নাম লিখতে পারব না। তবে আমরা চাইলে দুটি শব্দের মধ্যে থাকা স্পেসগুলো বাদ দিয়ে তাদের একটি শব্দ হিসেবে ভ্যারিয়েবলের নাম দেওয়া যাবে। আবার চাইলে দুটি শব্দের মধ্যে থাকা স্পেস বাদ দিয়ে তাদের মধ্যে একটি আন্ডারস্কোর (_) চিহ্ন দিয়েও ভ্যারিয়েবলটির নামকরণ করা যাবে।

২। ভ্যারিয়েবলের নামের প্রথম অক্ষর অবশ্যই a-z অথবা A-Z অথবা আন্ডারস্কোর (_) হতে হবে। প্রথম অক্ষর কোনো সংখ্যা (0-9) বা অন কোনো প্রতীক চিহ্ন (যেমন * বা - ইত্যাদি) হতে পারবে না। প্রথম অক্ষরের পর বাকি অক্ষরগুলোতে যেকোনো সংখ্যা (0-9) বা a-z অথবা A-Z অথবা আন্ডারস্কোর (_) ব্যবহার করা যাবে।

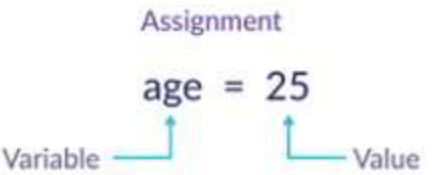
৩। ভ্যারিয়েবলের নামে অন্য কোনো প্রতীক চিহ্ন যেমন @, $, %, ^ ইত্যাদি ব্যবহার করা যাবে না।

৪। পাইথন একটি কেস সেন্সেটিভ (Case Sensitive) প্রোগ্রামিং ভাষা। তাই একই অক্ষর ছোটো হাতের ও বড়ো হাতের হলে পাইথন তাদের দুটি ভিন্ন ভ্যারিয়েবল হিসেবে বিবেচনা করবে। যেমন, My_variable আর my_variable দুটি ভিন্ন ভ্যারিয়েবল হিসেবে বিবেচিত হবে।

৫। কীওয়ার্ড হলো পাইথন প্রোগ্রামের সংরক্ষিত শব্দ। আমরা ভ্যারিয়েবলের নাম, ফাংশনের নাম বা অন্য কোনো শনাক্তকারী হিসেবে কীওয়ার্ড ব্যবহার করতে পারি না। পাইথন ৩.১০ তে ৩৫টি কী-ওয়ার্ড বা সংরক্ষিত শব্দ আছে। যেমন- False, True, None, and, as, break, class, continue, if, else, except, while, return, for, global ইত্যাদি।

ভ্যারিয়েবলের ভুল নামকরণ	ভ্যারিয়েবলের সঠিক নামকরণ
My Variable	MyVariable অথবা My_Variable
National ID	NationalID অথবা National_ID
this variable is cool	this_variable_is_cool
z!yan	zlyan অথবা z_yan
9abc	abc9
$variable	_variable
@My_name	My_name
print	কী ওয়ার্ডকে ভ্যারিয়েবলের নাম হিসাবে ব্যবহার করা যাবে না। তবে Myprint অথবা print1 সঠিক।

ভ্যারিয়েবলে ডেটা সংরক্ষণ করা অথবা ভ্যালু অ্যাসাইন করা



ভ্যারিয়েবলের মধ্যে ডেটা জমা করার জন্য আমরা = চিহ্ন ব্যবহার করি। একে বলা হয় ভ্যালু অ্যাসাইন করা।

ধরি, আমাদের একটি ভ্যারিয়েবল আছে age। এখন age এর মান যদি 25 রাখতে চাই, তাহল প্রোগ্রামে আমরা লিখব-

```
age=25
```

এই মানটি যদি প্রিন্ট করতে চাই তাহলে লিখব,

```
print (age)
```

এবার আমরা যদি পাশের প্রোগ্রামটি লিখে রান করি,

```
age=25
print (age)
```

তাহলে আমরা চিত্রের মতো আউটপুট পাব।

আবার, পুরো প্রোগ্রামে একটি ভ্যারিয়েবলের মান একাধিকবার পরিবর্তন করা সম্ভব। ভ্যারিয়েবলের মধ্যে নতুন মান অ্যাসাইন করা হলে আগের মান মুছে যায় ও সর্বশেষ অ্যসাইন করা মানটি জমা থাকে।

নিচের প্রোগ্রামটি যদি রান করি, আউটপুট তাহলে কি হবে?

```
value_now=1
print(value_now)
value_now=2
print(value_now)
value_now=3
print(value_now)
```

এই প্রোগ্রামের আউটপুট নিচের ঘরে লিখে ফেলি-

```
1
2
3
```

লক্ষ্য করি, উপরের প্রোগ্রামে একই ভ্যারিয়েবল value_now কে আমরা বারবার প্রিন্ট করেছি। কিন্তু একেক সময়ে ভ্যারিয়েবলটির ভেতরে জমা থাকা ডেটা ভিন্ন ছিল, তাই প্রিন্ট করার পর ভিন্ন মান পেয়েছি।

ডেটাটাইপ (Data Type)

আমরা জানি, ডেটা প্রক্রিয়াকরণের জন্য কম্পিউটার প্রসেসরকে পূর্ণ সংখ্যা, ভগ্নাংশযুক্ত সংখ্যা, বর্ণ, শব্দ, বুলিয়ান বা লজিক্যাল ডেটা ইত্যাদি নিয়ে কাজ করতে হয়। ডেটা টাইপ বলতে সাধারণত কোন ডেটা কম্পিউটার মেমরিতে কী পরিমাণ মেমরি (বিট বা বাইট) দখল করে তা নির্ধারণ করা এবং প্রসেসর কীভাবে ডেটাকে প্রসেস করে তা ব্যাখ্যা করাকে বুঝায়। পাইথন প্রোগ্রামে বিভিন্ন ধরনের ডেটাটাইপ (Data Type) রয়েছে। নিচে কিছু ডেটা টাইপ উল্লেখ করা হলো।

নিচে কিছু ডেটা টাইপ উল্লেখ করা হলো।

ক) **int** : পূর্ণসংখ্যাকে ইংরেজিতে ইন্টিজার নাম্বার (Integer number) বলা হয়। তাই ভ্যারিয়েবলে পূর্ণসংখ্যা জমা রাখলে তার ডেটাটাইপকে বলা হয় int। এখানে int হলো integer এর সংক্ষিপ্ত রূপ। এমন একটি উদাহরণ হলো-

```
EmpId_no = 5
```

খ) **float** : এই ভ্যারিয়েবলে আমরা ভগ্নাংশযুক্ত সংখ্যা জমা রাখতে পারি। এমন সংখ্যাকে ইংরেজিতে floating number (ফ্লোটিং নাম্বার) বলা হয়। তাই ভ্যারিয়েবলে ভগ্নাংশযুক্ত সংখ্যা জমা রাখলে তার ডেটাটাইপকে বলা হয় float। এমন একটি উদাহরণ হলো-

```
Age=45.50
```

গ) **str**: ভ্যারিয়েবলে যদি কোনো টেক্সট বা অক্ষর জাতীয় তথ্য জমা রাখতে চাই, তাহলে সেটিকে string (স্ট্রিং) বলা হয়। আর এ ধরনের তথ্য str ডেটাটাইপের অন্তর্ভুক্ত হয়। এখানে str হলো string এর সংক্ষিপ্ত রূপ। আমরা স্ট্রিং ভ্যারিয়েবলে যে টেক্সট রাখতে চাই তা Single Quotation এর মধ্যে জমা রাখব। উদাহরণ-

```
a='c'
b='This is a string variable'
```

ঘ) **bool**: bool হলো boolean এর সংক্ষিপ্ত রূপ। কোনো ভ্যারিয়েবলে যদি সত্য (True) অথবা মিথ্যা (False) কে ডেটা হিসেবে জমা রাখতে হয়, তাহলে সেটি হবে বুলিয়ান (Boolean) ডেটা। এখানে bool ডেটাটাইপে দুটি মাত্র তথ্য রাখা যায়- True ও False। এমন উদাহরণ নিচে দেখি-

```
a=True
```

পাইথন প্রোগ্রামে int, float, str এবং bool ছাড়াও আরও কিছু ডেটাটাইপ আছে, যদি কখনো প্রোগ্রাম লেখার সময় প্রয়োজন হয় আমরা সেগুলো সম্পর্কে তখন জানতে পারব। ভ্যারিয়েবলের মধ্যে আমরা যে ডেটা জমা রাখি, সেটা জমা হয় কম্পিউটারের মেমোরিতে। তাই যখন আমরা প্রোগ্রামের মধ্যে কোথাও ভ্যারিয়েবল ব্যবহার করব, কম্পিউটার মেমোরিতে জমা থাকা ভ্যারিয়েবলটির মান তখন প্রোগ্রামে ব্যবহার হবে।

ডেটা টাইপের রূপান্তর : টাইপ কাস্টিং (Casting)

কোনো ভ্যারিয়েবলের ডেটাটাইপ কোনটি, সেটি সহজেই বের করা যায় type () এর মাধ্যমে। আমরা যদি একটি ভ্যারিয়েবলকে type () ফাংশনের ভিতরে রেখে প্রিন্ট করি, তাহলে ওই ভ্যারিয়েবলের ডেটাটাইপ পেয়ে যাব। যেমন নিচের প্রোগ্রামটি যদি রান করি,

```
test_variable = 73.07
print (test_variable)
print(type(test_variable))
```

আউটপুট পাব নিচের মতো-

```
73.07

<class ' float '>
```

অর্থাৎ, আমরা বুঝতে পারলাম test_variable নামক ভ্যারিয়েবলের কাছে জমা করা তথ্য 73.07 এবং এটি একটি float ডেটাটাইপের অন্তর্ভুক্ত ভ্যারিয়েবল।

প্রোগ্রামিংয়ের ক্ষেত্রে এমন সময় আসতে পারে যখন কোনো ভেরিয়েবলের ডেটা টাইপ পরিবর্তন করতে হয় অথবা সুনির্দিষ্ট করতে হয়। এটি কাস্টিং (casting) এর মাধ্যমে করা যেতে পারে। পাইথনে কাস্টিং কনস্ট্রাক্টর (constructor) ফাংশন ব্যবহার করে তা করা হয়।

নিচে তিনটি গুরুত্বপূর্ণ টাইপ কাস্টিং ফাংশন উল্লেখ করা হলো।

`int()`- ফ্লোট বা স্ট্রিং থেকে পূর্ণসংখ্যা তৈরি করে।

`float()`-স্ট্রিং, ফ্লোট বা পূর্ণসংখ্যা থেকে ভগ্নাংশযুক্ত সংখ্যা তৈরি করে।

`str()`-বিভিন্ন ধরনের ডেটা থেকে স্ট্রিং তৈরি করে।

এখন নিচের উদাহরণগুলো লক্ষ্য কর।

```
x = int(1)   # x এর মান হবে 1
y = int(2.8) # y এর মান হবে 2
z = int("3") # z এর মান হবে 3
```

```
x = float(1)    # x এর মান হবে 1.0
y = float(2.8)  # y এর মান হবে 2.8
z = float("3")  # z এর মান হবে 3.0
w = float("4.2") # w এর মান হবে 4.2
```

```
x = str("s1") # x এর মান হবে 's1'
y = str(2)    # y এর মান হবে '2'
z = str(3.0)  # z এর মান হবে '3.0'
```

প্রোগ্রামে ডেটা ইনপুট নেওয়া

আমরা যদি প্রোগ্রামের ব্যবহারকারীর নিকট থেকে ডেটা ইনপুট নিতে চাই তাহলে কি করব? input () ব্যবহার করে এই কাজটি করা খুবই সহজ। প্রোগ্রাম নির্বাহ করার সময় যখন input() ফাংশন আসে তখন প্রোগ্রাম এক্সিকিউটশন করা বন্ধ করে দেয় এবং ব্যবহারকারী কিছু ইনপুট দিলে তখন তা আবার চলতে থাকে। আমরা যদি নিচের মতো লেখি-

```
test_input=input ()
```

তাহলে test_input ভ্যারিয়েবলটি আমাদের নিকট থেকে একটি ইনপুট গ্রহণ করবে। তবে, input () প্রোগ্রামের ভিতরে ইনপুট হিসেবে কোনো সংখ্যা, অক্ষর ইত্যাদি যেটাই গ্রহণ করুক না কেন, সেটিকে স্ট্রিং (str) ডেটাটাইপে গ্রহণ করবে। এখন আমরা একটি ভ্যারিয়েবল ইনপুট নিয়ে সেটি প্রিন্ট করি।

```
test_input= input()
print(test_input)
print(type(test_input))
```

এই প্রোগ্রামের আউটপুট তাহলে কি হবে? তুমি যা ইনপুট দিবে সেটিই কিন্তু আউটপুট হিসেবে প্রিন্ট হবে। কিন্তু খেয়াল করে দেখ, তুমি কোনো পূর্ণসংখ্যা বা ভগ্নাংশ ইনপুট দিলেও সেটির ডেটাটাইপ হিসেবে str প্রিন্ট হচ্ছে। অর্থাৎ তুমি যে তথ্যই জমা দাও, input () তথ্যটিকে একটি স্ট্রিং হিসেবে গ্রহণ করবে। কিন্তু তুমি যদি চাও ইনপুটটি যেন স্ট্রিং হিসেবে জমা না হয়ে ইন্টিজার বা ফ্লোট ডেটা হিসেবে জমা হয় তাহলে তোমাকে তথ্যটি ওই নির্দিষ্ট ডেটাটাইপে রূপান্তর করে নিতে হবে। নিচের মতো প্রোগ্রাম করে আমরা যখন কোনো তথ্য ইনপুট নিচ্ছি, input() ফাংশনকে সরাসরি int ডেটাটাইপে রূপান্তর করা যাবে-

```
test_input = int(input())
print(test_input)
print(type(test_input))
```

প্রোগ্রামটি রান করার পর খেয়াল করে দেখো int(input()) লেখার কারণে test_input ভ্যারিয়েবলটি int ডেটা টাইপে রূপান্তরিত হচ্ছে।

আবার, আমরা চাইলে একটি তথ্য ইনপুট নেবার সময় নির্দিষ্ট কমেন্ট বা নির্দেশনা দিয়ে দিতে পারব। সেজন্য input() এর ভিতরে ' ' দিয়ে সেই কমান্ড লিখে দিতে পারি। যেমন, আমরা যদি লেখি-

```
test_input = intput('Provide a sentence as an input:')
print(test_input)
print(type(test_input))
```

তাহলে প্রোগ্রামটি রান করার পর প্রথমে আমাদের কাছে একটি ইনপুট কমান্ড প্রদর্শন করবে

Provide a sentence as an input

এখন আমরা যদি চাই, আমাদের ইনপুট নেওয়া তথ্য প্রিন্ট করার আগে ও পরে আরও শব্দ বা বাক্য প্রিন্ট করতে পারি। এজন্য যে শব্দ বা বাক্য প্রিন্ট করতে চাই, সেটি print() এর মধ্যে ' ' এর ভিতরে লিখব এবং তারপর , (কমা) চিহ্ন দিয়ে আমাদের ভ্যারিয়েবলের নাম লিখে দিব। যেমন-

```
test_input=int(input('Insert an integer number:'))

print(' This is your integer number:', test_input)
```

তাহলে নিচের মতো আউটপুট দেখতে পাব-

Insert an integer number: 1245
This is your integer number: 1234

এখানে খেয়াল করে দেখো, print() এর মধ্যে যখন আমরা টেক্সট প্রিন্ট করছি, তখন সেটি ' ' বা একক উদ্ধৃতির ভিতরে লিখেছি। আর যখন আমরা একটি ভ্যারিয়েবল প্রিন্ট করেছি, সেটিকে কোনো ' ' বা একক উদ্ধৃতির মধ্যে না রেখে সরাসরি ভ্যারিয়েবলটির নাম লিখেছি।

গাণিতিক অপারেশন (Arithmetic Operation)

আমরা গাণিতিক অপারেশন (যোগ, বিয়োগ, গুণ, ভাগ ও ভাগশেষ) করতে সহজেই নিচের অপারেটরগুলো ব্যবহার করতে পারি-

অপারেটর	কাজ
+	এটি যোগ (Addition) অপারেটর। এই অপারেটর ব্যবহার করে অপারেটরটির দুপাশে থাকা দুটি ভ্যারিয়েবলকে যোগফল বের করা যাবে।
-	এটি বিয়োগ (Subtraction) অপারেটর। এই অপারেটর ব্যবহার করে অপারেটরটির দুপাশে থাকা দুটি ভ্যারিয়বলের বিয়োগফল বের করা যাবে।
*	এটি গুণ (Multiplication) অপারেটর। এই অপারেটর ব্যবহার করে অপারেটরটির দুপাশে থাকা দুটি ভ্যারিয়েরবলের গুণফল বের করা যাবে।
/	এটি ভাগ (Division) অপারেটর। এই অপারেটর ব্যবহার করে অপারেটরটির বামপাশে থাকা ভ্যারিয়বলকে ডানপাশে থাকা ভ্যারিয়েবল দিয়ে ভাগ করে ভাগ ফল বের করা যাবে।
%	এটি মডুলো (modulo) অপারেটর। এই অপারেটর ব্যবহার করে অপারেটরটির বামপাশে থাকা ভ্যারিয়েবলকে ডানপাশে থাকা ভ্যারিয়েবল দিয়ে ভাগ করে ভাগশেষ বের করা যাবে।

ব্যবহারিক সমস্যা-১ : দুটি সংখ্যা ইনপুট নিয়ে তাদের যোগফল প্রিন্ট করার জন্য একটি প্রোগ্রাম তৈরি করি।
সমাধান : দুটি পূর্ণসংখ্যা ইনপুট নিয়ে তাদের যোগফল প্রিন্ট করার প্রোগ্রামটি নিচে দেওয়া হলো-

```
num1 = int(input('Insert the first integer:'))
num2 = int(input('Insert the second integer:'))
result = num1+num2
print('The sum of', num1,'and', num2, 'is', result)
```

উপরের প্রোগ্রামটি রান করলে তুমি পছন্দমতো দুটি সংখ্যা ইনপুট দিতে পারবে এবং তারপর সংখ্যা দুটির যোগফল হিসাব করে নিচের মতো ফলাফল দেখাবে।

```
Insert the first integer: 50
Insert the second integer: 100
The sum of 50 and 100 is 150
```

উপরের প্রোগ্রামটি কিছু পরিবর্তন করে বিয়োগফল ও গুণফল ছাপানোর একটি প্রোগ্রাম তৈরি কর।

Comparison বা তুলনা করার অপারেটর এবং তাদের ব্যবহার-

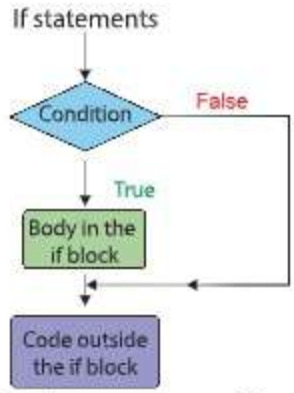
রিলেশনাল অপারেটর	উদাহরণ	ব্যবহার
== (Equal)	Op1== 10 Op1== (Op2 + Op3)	ডান দিকের অপার‍্যান্ড এবং বাম দিকের অপার‍্যান্ড সমান হলে।
!= (Not Equal)	Op1 != 10 Op1 != (Op2 - Op3)	ডান দিকের অপার‍্যান্ড এবং বাম দিকের অপার‍্যান্ড অসমান হলে।
< (Less Than)	Op1<10 Op1< (Op2 - 10)	একাধিক অপার‍্যান্ড-বিশিষ্ট গ্রুপ এক্সপ্রেশন তৈরির জন্য।
<= (Less or Equal	Op1 <= 10 Op1<= (Op2-Op3)	ডান দিকের অপার‍্যান্ডের চেয়ে বাম দিকের অপার‍্যান্ড ছোটো বা সমান হলে।
> (Greater Than)	Op1>10 Op1> (Op2 + Op3)	ডান দিকের অপার‍্যান্ডের চেয়ে বাম দিকের অপার‍্যান্ড বড়ো হলে।
>= (Greater or equal)	Op1>=10 Op1>= (+Op2+Op3)	ডান দিকের অপার‍্যান্ডের চেয়ে বাম দিকের অপার‍্যান্ড বড়ো, বা সমান হলে।

এখানে, Op1, Op2, Op3, যে কোন বৈধ ভেরিয়েবল। রিলেশনাল অপারেটরের সাথে এক বা একাধিক ভেরিয়েবল অথবা এ্যারেথমেটিক এক্সপ্রেশন সহযোগে রিলেশনাল এক্সপ্রেশন তৈরি করা হয়।

প্রোগ্রামে শর্তের ব্যবহার

আমরা কোন পরীক্ষার ফলাফল হিসাব করার একটি প্রোগ্রাম তৈরি করব। ধরা যাক ঐ পরীক্ষায় পাশের নম্বর হল ৪০। এখন যে ৪০ এর চেয়ে কম পাবে সে ফেল করবে এবং যে ৪০ বা তার চেয়ে বেশি পাবে সে পাশ করবে। কাজেই এখানে একটি শর্ত বিবেচনা করতে হবে এবং শর্তের উপর পাশ কিংবা ফেল নির্ভর করছে। পাইথন প্রোগ্রামে শর্ত সাপেক্ষে কোন কার্য বা স্টেটমেন্ট সম্পাদনের জন্য if স্টেটমেন্ট ব্যবহৃত হয়। প্রোগ্রামে "যদি" অর্থে if স্টেটমেন্টর ব্যবহার তুলনা করা যেতে পারে। if স্টেটমেন্ট ব্যবহারের ফরম্যাট হলো-

```
if condition:
    statement(s) to be executed if condition is true
statement just below if
```



if স্টেটমেন্টে ব্যবহৃত শর্ত (condition) সাধারণত এক বা একাধিক লজিক্যাল বা রিলেশনাল এক্সপ্রেশন হতে পারে যা if পরবর্তী লেখা হয়। if (condition) স্টেটমেন্টের পর কোলন থাকবে।

এই শর্তের মান যদি সত্য হয় তবে if পরবর্তী লাইনে বর্ণিত কাজগুলো সম্পাদিত হবে। একে if এর বডি (Body) বা মূল অংশও বলা হয়। এখানে যে কোনো বৈধ সিম্পল বা কম্পাউন্ড স্টেটমেন্ট থাকতে পারে।

শর্তের মান যদি মিথ্যা হয় তবে if এর মূল অংশ বা if এর বডিতে প্রোগ্রামের নিয়ন্ত্রণ যাবে না। ফলে বডিতে বর্ণিত কাজগুলো সম্পাদিত না হয়ে পরবর্তী স্টেটমেন্টসমূহ সম্পাদিত হবে।

ব্যবহারিক সমস্যা # ২ : কোন পরীক্ষায় পাশের সর্বনিম্ন নম্বর হচ্ছে ৪০। কোনো শিক্ষার্থী যদি পরীক্ষায় ৪০ বা তার চেয়ে বেশি নম্বর অর্জন করে তবে পাশ অন্যথায় ফেল হিসেবে গণ্য করা হবে। শিক্ষার্থীর অর্জিত নম্বরকে ইনপুট হিসেবে ব্যবহার করে আউটপুট হিসেবে উক্ত শিক্ষার্থীর পাশ বা ফেলের সিদ্ধান্ত প্রদানের জন্য একটি পাইথন প্রোগ্রাম লেখো।

সমাধান : নিচে প্রোগ্রামটির কোড দেওয়া হলো।

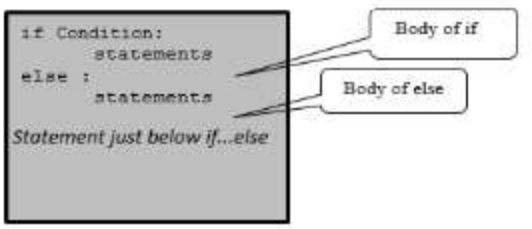
```
mark = int(input('Please enter your mark: '))
if mark >= 40:
    print('You have passed')
if mark < 40:
    print('You have failed')
```

প্রোগ্রামটি রান করে প্রাপ্ত নমুনা ফলাফল-

```
Please enter your mark: 85
You have passed in the subject.
```

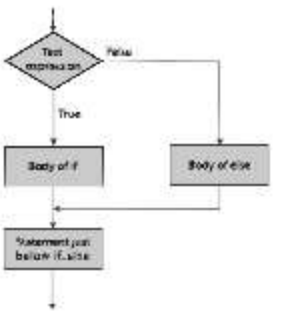
if...else স্টেটমেন্ট

পাইথন প্রোগ্রামে শর্ত সাপেক্ষে কোনো স্টেটমেন্ট বা কার্য সম্পাদনের জন্য if স্টেটমেন্ট ব্যবহৃত হয়। if স্টেটমেন্টের সাথে "অন্যথায়" অর্থে else স্টেটমেন্ট ব্যবহৃত হয়। এজন্য এই স্টেটমেন্টকে if... else স্টেটমেন্ট বলা হয়। if... else স্টেটমেন্ট ব্যবহারের ফরম্যাট হলো-



উল্লেখ্য যে, if...else কন্ট্রোলে ব্যবহৃত শর্ত (Condition) সাধারণত এক বা একাধিক লজিক্যাল বা রিলেশনাল এক্সপ্রেশন হয়। if (Condition) স্টেটমেন্টের পরে কোলন বসে।

এই Condition বা শর্তের মান যদি সত্য হয় তবে Body of if এ বর্ণিত কাজগুলো সম্পাদিত হবে। আর যদি মিথ্যা হয় তাহলে Body of else এ বর্ণিত কাজসমূহ সম্পাদিত হয়। এখানে statements এ যে কোন বৈধ সিম্পল বা কম্পাউন্ড স্টেটমেন্ট হতে পারে। চিত্রে if...else স্টেটমেন্টের প্রবাহচিত্র দেখান হলো-



ব্যবহারিক সমস্যা # ৩ : পাইথন ভাষার একটি প্রোগ্রাম তৈরি করতে হবে যাতে কোন নাগরিকের বয়স ইনপুট দেওয়া হলে তিনি নির্বাচনের ভোটার হওয়ার উপযুক্ত কিনা তা যাচাই করবে।

সমাধান : ন বাংলাদেশের যে কোন নাগরিক ১৮ বয়স হলেই নির্বাচনে ভোটার হিসাবে বিবেচিত হয়। নিচে প্রোগ্রামটির কোড দেওয়া হলো।

```
age = int(input('Please enter your age: '))
if age >= 18:
    print('You are eligible for voting')
else :
    print('You are not eligible for voting')
```

ব্যবহারিক সমস্যা # ৪ : কোন সংখ্যা জোড় না বিজোড় তা নির্ণয় করার জন্য একটি পাইথন প্রোগ্রাম লোখো।

সমাধান : নিচে প্রোগ্রামটির কোড দেওয়া হলো।

```
num = int(input('Insert a number= '))
if num%2!= 0:
    print('The number is odd')
else :
    print(' The number is even')
```

প্রোগ্রামটি রান করে প্রাপ্ত নমুনা ফলাফল-

```
Insert a number = 67
The number is odd.
```

ব্যবহারিক সমস্যা # ৫ : কোন বর্ষ অধিবর্ষ কিনা তা নির্ণয় করার জন্য একটি প্রোগ্রাম লেখো।

সমাধান: কোনো বর্ষ লিপ ইয়ার (Leap year) কিনা তা জানতে হলে বর্ষ সংখ্যাটিকে ৪০০ দিয়ে ভাগ করতে হয়, ভাগশেষ শূন্য হলে তা লিপ-ইয়ার, অন্যথায় বর্ষ সংখ্যাকে ১০০ দিয়ে ভাগ করতে হয়, এবার যদি ভাগশেষ শূন্য হয় তবে তা লিপ ইয়ার নয় এবং যদি ভাগশেষ শূন্য না হয়, তবে বর্ষ সংখ্যাটিকে পুনরায় ৪ দিয়ে ভাগ করতে হয়। এক্ষণে ভাগশেষ শূন্য হলে বর্ষটি লিপইয়ার, নতুবা লিপ ইয়ার নয়।

```
year = int(input('Enter the year (4 Digit) to check :'))
if (year%400 == 0 or (year%100 != 0 and year %4 == 0)):
    print(year, 'is a leap year.')
else :
    print(year, 'is not a leap year.')
```

প্রোগ্রামটি রান করে প্রাপ্ত নমুনা ফলাফল-

```
Enter the year (4 Digit) to check :2012
2012  is a leap year.
Enter the year (4 Digit) to check :1990
1990  is not a leap year.
```

ব্যবহারিক সমস্যা # ৬ : মিটার থেকে ফুট এবং ফুট থেকে মিটারে রূপান্তর করার জন্য একটি প্রোগ্রাম লেখো।

সমাধান : নিচে প্রোগ্রামটির কোড দেওয়া হলো।

```
print('1:Feet to Meters, 2:Meters to Feet.')
choice = int(input('Enter Choice: '))
if choice == 1:
    num = float(input('Enter number of feet: '))
    print('Meters: ', round((num/3.28),3))
else  :
    num = float(input('Enter number of meters: '))
    print ('Feet: ', round((num*3.28),3))
```

প্রোগ্রামটি রান করে প্রাপ্ত নমুনা ফলাফল -

```
1:Feet to Meters, 2:Meters to =Feet. Enter Choice:1
Enter number of feet: 25
Meters: 7.622

1:Feet to Meters, 2:Meters to Feet.
Enter Choice: 2
Enter number of Meters: 60
Feet : 196.800
```

একাধিক শর্ত যাচাই করার জন্য পাইথন প্রোগ্রামে "অন্যথায় যদি" অর্থে if...else স্টেটমেন্টের সাথে স্টেটমেন্ট ব্যবহৃত হয়। elif স্টেটমেন্ট else এবং if এর মাঝে থাকে। প্রোগ্রামে একাধিক if স্টেটমেন্ট ব্যবহারের বিকল্প হিসেবে elif স্টেটমেন্ট জনপ্রিয়। অর্থাৎ প্রোগ্রামে একাধিক শর্ত যাচাই করার জন্য elif ব্যবহৃত হয়। স্টেটমেন্ট ব্যবহারের ফরম্যাট হলো-

```
if (Conditional):
       Block1
elif (Conditional2):
       Block2
... ... ...
else :
       DefaultBlock;

BlockN;
... ... ...
```

ব্যবহারিক সমস্যা # ৭ : প্রদত্ত কোনো সংখ্যা ধনাত্মক, ঋণাত্মক না শূন্য তা নির্ণয় করার জন্যএকটি পাইথন প্রোগ্রাম লেখো।

সমাধান নিচে প্রোগ্রামটির কোড দেওয়া হলো।

```
num = int(input('Insert a number: '))
if num > 0:
    print('The number is positive')
elif num < 0:
    print('The number is negative')
else :
    print('It is zero')
```

প্রোগ্রামটি রান করে প্রাপ্ত নমুনা ফলাফল-

```
Insert a number : 7
The number is positive.
```

match স্টেটমেন্ট

পাইথন প্রোগ্রামে elif স্টেটমেন্টের অনুরূপ কাজে অর্থাৎ একাধিক স্টেটমেন্ট থেকে নির্দিষ্ট কোনো স্টেটমেন্ট নির্বাহের জন্য match স্টেটমেন্ট ব্যবহৃত হয়। মূলত বেশি সংখ্যক elif স্টেটমেন্ট ব্যবহারের পরিবর্তে match স্টেটমেন্ট ব্যবহৃত হয়। match স্টেটমেন্টের সাথে অতিরিক্ত case ও break স্টেটমেন্ট ব্যবহৃত হতে পারে। elif স্টেটমেন্টে কোনো কন্ডিশনাল কিংবা রিলেশনাল এক্সপ্রেশনের উপর ভিত্তি করে উপযুক্ত স্টেটমেন্ট নির্বাচন করা হয়। কিন্তু match স্টেটমেন্টে সাধারণত কোনো বৈধ ভেরিয়েবলের মানের ভিত্তিতে উপযুক্ত স্টেটমেন্ট নির্বাচন করা হয়। match স্টেটমেন্টের ফরম্যাট হলো-

```
match MatchExp:
       case Value1 :
              Block1
       case Value2 :
              Block2
       case Value3 :
              Block3
       case _ :
              DefaultBlock
BlockN
... ... ...
```

এখানে, match স্টেটমেন্টের সাথে ব্যবহৃত এক্সপ্রেশনকে ম্যাচ এক্সপ্রেশন (matchExp) বলা হয়। case স্টেটমেন্টে ম্যাচ এক্সপ্রেশন (matchExp) ভেরিয়েবলের সম্ভাব্য মান দেওয়া হয়। case কীওয়ার্ড ও ম্যাচ এক্সপ্রেশন ভেরিয়েবলের সম্ভাব্য মানের মাঝে ন্যূনতম একটি স্পেস থাকে এবং কোলন দ্বারা শেষ হয়। case স্টেটমেন্টে শেষে সাধারণত একটি default স্টেটমেন্ট থাকে যাকে কোলন দ্বারা শেষ হয়।

ব্যবহারিক সমস্যা # ৮ : ইংরেজি ছোটো হাতের কোনো অক্ষর vowel না consonants তা জানার জন্য একটি প্রোগ্রাম লেখো।

সমাধান: ইনপুট হিসাবে ইংরেজি ছোটো হাতের কোনো অক্ষর টাইপ করা হবে। নিচে প্রোগ্রামটির কোড দেওয়া হলো।

```
char = input('Enter the letter: ')
match char:
        case "a" :
                print(char, ' is a vowel ')
        case "e" :
                print(char, ' is a vowel ')
        case "i" :
                print(char, ' is a vowel ')
        case "o" :
                print(char, ' is a vowel ')
        case "u" :
                print(char, ' is a vowel ')
        case _:
print(char, ' is a consonants')
```

প্রোগ্রামটি রান করে প্রাপ্ত নমুনা ফলাফল-

Enter the letter: a
a is a vowel
Enter the letter: d
d is a consonants

বিকল্প পদ্ধতির সমাধান: উপরের প্রোগ্রামটিকে ইনপুট হিসেবে শুধুমাত্র ইংরেজি ছোটো হাতের কোনো অক্ষর টাইপ করা হলে ফলাফল দেখাবে। নিচের প্রোগ্রামটিতে ইংরেজি ছোটো হাতের বা বড়ো হাতের যে কোনো অক্ষর টাইপ করা হলে ফলাফল দেখাবে। নিচে প্রোগ্রামটির কোড দেওয়া হলো।

```
char = input('Enter the letter: ')
vowels = ['a', 'e', 'i', 'o', 'u', 'A', 'E', 'I', 'O', 'U']
if char in vowels:
    print(char, ' is a vowel')
else :
    print(char, ' is a consonants')
```

ব্যবহারিক সমস্যা # ৯ : তিনটি পূর্ণ সংখ্যার মধ্য থেকে বড় সংখ্যাটি বের করার জন্য একটি প্রোগ্রাম লেখো।

সমাধান : নিচে প্রোগ্রামটির কোড দেওয়া হলো।

```
num1 = int(input('Enter integer number 1: '))
num2 = int(input('Enter integer number 2: '))
num3 = int(input('Enter integer number 3: '))
if (num1 >= num2) and (num1 >= num3):
    print(num1, 'is the largest number')
elif (num2 >= num1) and (num2 >= num3):
    print(num2, 'is the largest number')
else :
    print(num3, 'is the largest number')
```

প্রোগ্রামটি রান করে প্রাপ্ত নমুনা ফলাফল

```
Enter integer number 1: 10
Enter integer number 2: 20
Enter integer number 3: 30
30 is the largest number.
```

একই কাজ বার বার করা : প্রোগ্রামে লুপের ব্যবহার

আমাদের জীবনে আমরা যেমন কিছু কাজ বার বার করি তেমনি প্রোগ্রামেও কিছু কাজ বার বার করতে হয়। প্রোগ্রামে একই কাজ একাধিক বার সম্পন্ন করতে হলে লুপ (Loop) ব্যবহার করতে হয়।

স্টেটমেন্ট দুই বা ততোধিক বার সম্পাদনের জন্য লুপ কন্ট্রোল স্টেটমেন্ট ব্যবহৃত হয়। পাইথন প্রোগ্রামে লুপ নির্বাহের জন্য ব্যবহৃত অন্যতম লুপ কন্ট্রোল স্টেটমেন্টসমূহ হলো-

- for স্টেটমেন্ট
- while স্টেটমেন্ট
- continue, break ও pass স্টেটমেন্ট।

নিম্নে বিভিন্ন স্টেটমেন্ট সম্পর্কে আলোচনা করা হলো-

for লুপ স্টেটমেন্ট

পাইথন প্রোগ্রামে কোনো স্টেটমেন্ট দুই বা ততোধিক বার সম্পাদন করার জন্য for স্টেটমেন্ট ব্যবহৃত হয়। লুপ কতবার নির্বাহ করা হবে তা জানা থাকলেই কেবলমাত্র for লুপ ব্যবহার উপযোগী। সাধারণত কোনো ভেরিয়েবল ব্যবহার করে for লুপের আবর্তন সংখ্যা গণনা করা হয়। এরূপ ভেরিয়েবলকে কাউন্টার ভেরিয়েবল বলা হয়। নিম্নে for স্টেটমেন্টের ফরম্যাট হলো-

```
for CounterInitialization in Condition :
        // Statement(s)
```

CounterInitialization অংশে কাউন্টার ভেরিয়েবলের প্রারম্ভিক মান দেওয়া হয়, একে Initialization বা লুপের শুরু বলা হয়। Condition অংশে কাউন্টার ভেরিয়েবলের চূড়ান্ত মান কিংবা চূড়ান্ত মান নির্ধারণের শর্ত দেওয়া হয়। কাউন্টার ভেরিয়েবল চূড়ান্ত মানে না পৌঁছা পর্যন্ত কিংবা শর্ত সত্য থাকা পর্যন্ত for লুপের সাথে সংশ্লিষ্ট স্টেটমেন্ট সম্পাদিত হতে থাকে।

ব্যবহারিক সমস্যা # ১০ : একটি নির্দিষ্ট টেক্সটকে একাধিকবার প্রদর্শনের জন্য একটি প্রোগ্রাম লেখো।

সমাধান : ধরা যাক, ICT টেক্সটি ৫ বার প্রদর্শন করা হবে।

```
for i in range(1,6):
    print('ICT')
```

বিকল্প পদ্ধতির সমাধান ১: (Increment by 1)

```
for i in range(1,6,1):
    print('ICT')
```

বিকল্প পদ্ধতির সমাধান ২: (Decrement by 1)

```
for i in range(6,1,-1):
    print('ICT')
```

প্রোগ্রামটি রান করে প্রাপ্ত নমুনা ফলাফল-

```
ICT
ICT
ICT
ICT
ICT
```

ব্যবহারিক সমস্যা # ১১ : একটি প্রোগ্রাম লিখ যা কোনো শব্দ ৫০ বার দেখাবে।

সমাধান : নিচে প্রোগ্রামটির কোড দেওয়া হলো।

```
var = input(('Write a word: '))
for i in range(1,51):
    print(var)
```

প্রোগ্রামটি রান করে প্রাপ্ত নমুনা ফলাফল-

```
Write a word: Computer
Computer  Computer  Computer  Computer  Computer
Computer  Computer  Computer  Computer  Computer
.
.
.
Computer  Computer  Computer  Computer  Computer
Computer  Computer  Computer  Computer  Computer
```

ব্যবহারিক সমস্যা # ১২ : ১ থেকে ১০০ পর্যন্ত সংখ্যাগুলোর মধ্যে বিজোড় সংখ্যাগুলো বের করার একটি প্রোগ্রাম লিখ।

সমাধান: নিচে প্রোগ্রামটির কোড দেওয়া হলো।

```
for i in range(1, 101):
    if (i % 2 != 0):
        print(i)
```

প্রোগ্রামটি রান করলে নিচের ফলাফল প্রদর্শিত হবে-

1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 57 59 61 63 65 67 69 71 73 75 77 79 81 83 85 87 89 91 93 95 97 99

ব্যবহারিক সমস্যা # ১৩ : একটি প্রোগ্রাম লিখ যা নিম্নের ন্যায় ফলাফল দেখাবে।

```
1
12
123
1234
.
.
.
123456789
```

সমাধান : নিচে প্রোগ্রামটির কোড দেওয়া হলো।

```
for i in range(1, 10):
    for j in range(1, i + 1):
        print(j, end=" ")
    print()
```

প্রোগ্রামটি রান করলে নিচের নমুনা ফলাফল প্রদর্শিত হবে-

```
1
1 2
1 2 3
1 2 3 4
1 2 3 4 5
1 2 3 4 5 6
1 2 3 4 5 6 7
1 2 3 4 5 6 7 8
1 2 3 4 5 6 7 8 9
```

ব্যবহারিক সমস্যা # ১৪ : কোনো একটি সংখ্যার নামতা বা গুণের টেবিল নির্ণয়ের একটি প্রোগ্রাম লেখো।

সমাধান : নিচে প্রোগ্রামটির কোড দেওয়া হলো।

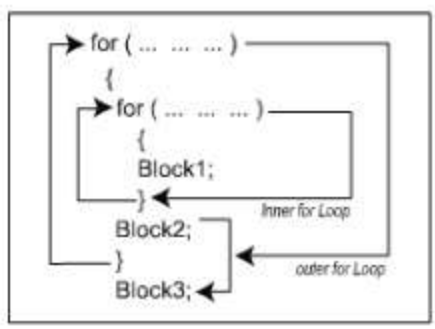
```
num = int(input('Enter a number: '))
for i in range(1,11):
    prod = num*i
    print(num, '*', i, '=', prod)
```

প্রোগ্রামটি রান করে নিচের নমুনা ডেটা ইনপুট দিলে ফলাফল প্রদর্শিত হবে-

Enter a number : 3

```
3*1=3
3*2=6
3*3=9
3*4=12
3*5=15
3*6=18
3*7=21
3*8=24
3*9=27
3*10=30
```

একটি for লুপ স্টেটমেন্টের মধ্যে অন্য কোন কোন for স্টেটমেন্ট থাকতে পারে। এরূপ for লুপকে নেস্টেড for লুপ এবং মাধ্যবর্তী for লুপকে ইনার (Inner) for লুপ, বহিস্থ for লুপকে আউটার (Outer) for লুপ বলা হয়। আউটার লুপের পূর্বে ইনার for লুপের কাজ শেষ হয়।

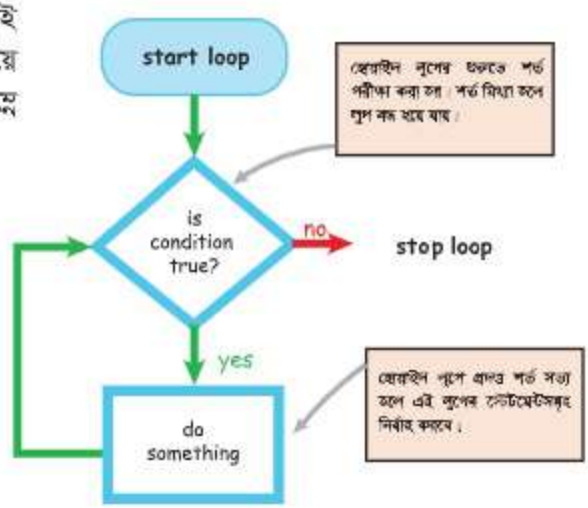


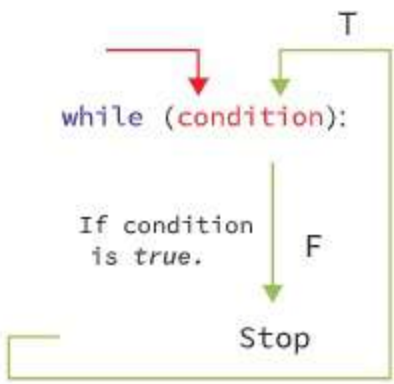
while লুপ

পাইথন প্রোগ্রামে শর্ত সাপেক্ষে দুই বা ততোধিক বার কোনো স্টেটমেন্ট সম্পাদন করার জন্য while লুপ স্টেটমেন্ট ব্যবহৃত হয়। এই লুপের শুরুতেই শর্ত পরীক্ষা করা হয়।

প্রদত্ত শর্ত সত্য অথবা অশূন্য (Non Zero) হলেই কেবল লুপ নির্বাহ হবে; অন্যথায় লুপ থেকে বের হয়ে যাবে। কোনো স্টেটমেন্ট নির্বাহ করবে না। নিম্নে while লুপ নিয়ন্ত্রণের ফরম্যাট দেওয়া হলো-

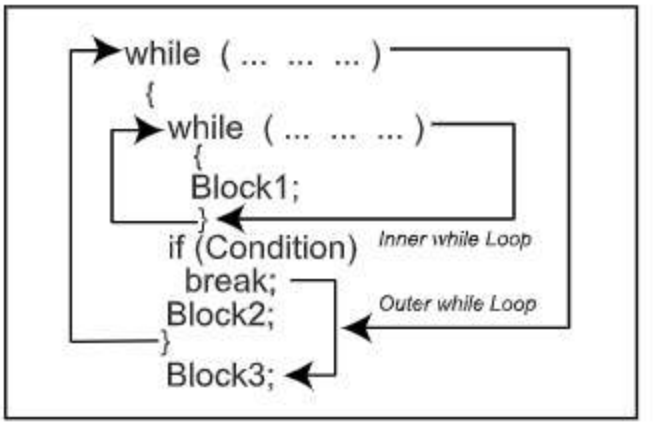
```
CounterInitialization;
while Condition:
        // Statement(s)
CounterDecrement/Increment;
                          )
```





CounterInitialization অংশে উপযুক্ত ডেটা টাইপসহ কাউন্টার প্রারম্ভিক মান নির্ধারণ করা হয়। Condition অংশে কাউন্টার ভেরিয়েবলের চূড়ান্ত মান কিংবা চূড়ান্ত মান নির্ধারণের জন্য শর্ত দেওয়া হয় এবং CounterDecrement/Increment অংশে প্রতিবার আবর্তন কালে কাউন্টার ভেরিয়েবলের হ্রাস/বৃদ্ধির মান নির্ধারণ করা হয়। Condition সত্য থাকা পর্যন্ত while লুপের সাথে সংশ্লিষ্ট স্টেটমেন্ট নির্বাহিত হতে থাকে। যখনই শর্ত মিথ্যা হয়ে যায় তখনই লুপ থেকে বের হয়ে যায়।

while লুপের সাথে সংশ্লিষ্ট স্টেটমেন্ট সাধারণত কম্পাউন্ড স্টেটমেন্ট হয়ে থাকে। প্রয়োজনে একটি while লুপের মধ্যে অপর কোনো while লুপ বা for লুপ ব্যবহার করা যেতে পারে। এরূপ মধ্যবর্তী while লুপকে নেস্টেড while লুপ বলা হয়।



নেস্টেড while স্টেটমেন্টের প্রবাহচিত্র

while লুপ অনেকটা for স্টেটমেন্ট বিকল্প হিসেবে ব্যবহৃত হয়। for স্টেটমেন্টের মত পূর্বে ঘোষিত কোনো কাউন্টার ভেরিয়েবল ব্যবহার করে স্টেটমেন্টের আবর্তন সংখ্যা গণনা করা হয়।

ব্যবহারিক সমস্যা # ১৫ : হোয়াইল লুপ ব্যবহার করে ICT লেখাটিকে একাধিকবার প্রদর্শনের জন্য একটি প্রোগ্রাম লেখো।

সমাধান : ধরা যাক, ICT টেক্সটি ৩ বার প্রদর্শন করা হবে।

```
count=1
while (count<=3):
    print('ICT')
    count=count+1
```

প্রোগ্রামটি রান করে প্রাপ্ত নমুনা ফলাফল

```
ICT
ICT
ICT
```

নিচে while লুপ স্টেটমেন্ট ব্যবহার করে ১ থেকে ১০০ পর্যন্ত এবং ১ থেকে *n* (যেখানে *n* হলো কোনো ধনাত্মক পূর্ণ সংখ্যা) পর্যন্ত ধারার যোগফল নির্ণয়ের দুটি প্রোগ্রাম দেওয়া হলো।

ব্যবহারিক সমস্যা # ১৬ : ১+২+৩+৪+.....+১০০ ধারার যোগফল নির্ণয়ের একটি প্রোগ্রাম লিখ।

অথবা, ১ থেকে ১০০ পর্যন্ত সকল ধনাত্মক সখ্যার যোগফল নির্ণয়ের প্রোগ্রাম লিখ।

সমাধান: নিচে প্রোগ্রামটির কোড দেওয়া হলো।

```
sum=0
count=1
while (count<=100):
    sum = sum + count
    count=count+1
print('The sum of the value is ', sum)
```

প্রোগ্রামটি রান করে নিচের ফলাফল প্রদর্শিত হবে-

The sum of the value is 5050

continue স্টেটমেন্ট

পাইথন প্রোগ্রামে শর্তযুক্ত অথবা শর্তবিহীনভাবে কোনো স্টেটমেন্ট বা লুপের পুনরাবৃত্তি করার জন্য continue স্টেটমেন্ট ব্যবহৃত হয়। continue স্টেটমেন্টের ফরম্যাট হলো-

```
continue
```

continue স্টেটমেন্ট প্রোগ্রাম পয়েন্টারকে পূর্ববর্তী স্টেটমেন্ট বা লুপের প্রারম্ভে স্থানান্তর করে। continue স্টেটমেন্ট if, elif, for, while ইত্যাদি ছাড়া সরাসরি কাজ করতে পারে না। এ জন্য সাধারণত if, elif স্টেটমেন্টের সাথে সম্পর্কিত শর্ত সাপেক্ষে কোন লুপের পুনরাবৃত্তি করার জন্য continue স্টেটমেন্ট ব্যবহৃত হয়। সে ক্ষেত্রে শর্তের মান সত্য হলে continue স্টেটমেন্ট কার্যকরী হয়, অন্যথায় কম্পাইলার continue স্টেটমেন্ট উপেক্ষা করে পরবর্তী স্টেটমেন্ট নির্বাহ করে।

continue স্টেটমেন্ট ব্যবহার করে ১ থেকে ১০ পর্যন্ত সংখ্যাসমূহের মধ্য থেকে যে কোনো একটি সংখ্যা বাদ দিয়ে অবশিষ্ট সংখ্যাসমূহ প্রদর্শনের একটি প্রোগ্রাম নিচে দেওয়া হলো।

ব্যবহারিক সমস্যা # ১৭ : ১ থেকে ১০ পর্যন্ত সংখ্যাসমূহের মধ্য থেকে যে কোনো একটি সংখ্যা বাদ দিয়ে অবশিষ্ট সংখ্যাসমূহ প্রদর্শনের জন্য একটি প্রোগ্রাম লিখ।

সমাধান : ধরা যাক, বাদ দেওয়া সংখ্যাটি হচ্ছে ৪। প্রোগ্রামটি নিচে দেওয়া হলো-

```
for num in range(1, 11, 1):
    if num == 4:
        continue
    else:
        print(num, end=" ")
print('Used continue to skip printing the value 4')
```

প্রোগ্রামটি রান করলে নিচের ফলাফল প্রদর্শিত হবে–

```
1 2 3 5 7 8 9 10
Used continue to skip printing the value 4
```

break স্টেটমেন্ট

পাইথন প্রোগ্রামে লুপের মধ্যে break স্টেটমেন্ট পেলে লুপ সেখানে তার কাজ শেষ করে লুপ থেকে বের হয়ে যাবে। অনেকটা চলন্ত গাড়িতে ব্রেক কষলে যেমন গাড়ির চলা থেমে যায় তেমনি লুপের মধ্যে break স্টেটমেন্ট পেলে লুপ নির্বাহ বন্ধ হয়ে যায়। অধিকাংশ ক্ষেত্রেই break স্টেটমেন্ট if, elif ইত্যাদির সাথে ব্যবহৃত হয়।

অনুশীলনী

১। অনুবাদকের কাজ হলো-

i) সবগুলো নির্দেশ একসাথে মেশিন কোডে রূপান্তরিত করা।

ii) একটি একটি করে নির্দেশ মেশিন কোডে রূপান্তরিত করা।

iii) মেশিন কোডকে পাইথন কোডে রূপান্তরিত করা।

নিচের কোনটি সঠিক?

ক. i ও ii খ. i ও iii

গ. ii ও iii ঘ. i, ii ও iii

২। পাইথন ভাষায় মনিটরে কোন লেখা প্রদর্শন করার জন্য নিচের কোন ফাংশনটি ব্যবহার করা হয়?

ক. `printf()` খ. `print()`

গ. `print` ঘ. `println()`

৩। নিচের কোনটি পাইথন ভাষায় ভেরিয়েবলের সঠিক নাম?

ক. 9abc খ. $variable

গ. print ঘ. National_ID

৪। পাইথন ভাষায় y = int(2.8) হলে print (y) এর মান কত হবে?

ক. ২.৮ খ. ২

গ. ২৮ ঘ. ২.০

৫। পাইথন ভাষায়, a=5, b=2, y = a%b হলে print (y) এর মান কত হবে?

ক. ১ খ. ২

গ. ৩ ঘ. ৫

৬। নিচের পাইথন কোডটি লক্ষ্য কর।

```
for i in range(1,6):
print('ICT')
```

এখানে ICT লেখাটি কতবার প্রদর্শিত হবে?

ক. ১ খ. ৫

গ. ৩ ঘ. ৬

৭। মেশিন কোড কী? প্রোগ্রাম কোডকে মেশিন কোডে রূপান্তরের উপায় লিখ।

৮। কাস্টিং কনস্ট্রাক্টর ফাংশন বলতে কী বুঝায়? কয়েকটি ফাংশনের নাম লিখ।

৯। ১ থেকে ১০০ পর্যন্ত সকল ধনাত্মক সংখ্যার যোগফল নির্ণয়ের জন্য for লুপ ও while লুপ ব্যবহার করে দুটি প্রোগ্রাম লিখ। প্রোগ্রামদ্বয়ের মধ্যে কোনটি সুবিধাজনক তা বিশ্লেষণ কর।

১০। break ও continue স্টেটমেন্ট ব্যাখ্যা কর।

সমাপ্ত