

## জনপ্রিয় কিছু উচ্চস্তরের প্রোগ্রামিং ভাষার পরিচিতিঃ

**সি (C):** ‘সি’ প্রোগ্রামিং ভাষা একটি স্ট্রাকচার্ড বা প্রোসিডিউর প্রোগ্রামিং ভাষা। মিদ লেভেল ল্যাস্‌সুয়েজ হিসেবে ‘সি’ অত্যন্ত জনপ্রিয়। “ডেনিশ রিচি” ‘সি’ প্রোগ্রামিং ভাষার জনক। ‘সি’ নামটা এসেছে মার্টিন রিচার্ডস (Martins Richards) এর উদ্ভাবিত বিসিপিএল (BCPL-Basic Combined Programming Language) ভাষা থেকে। BCPL সংক্ষেপে B নামে পরিচিত ছিল। পরে B এর উন্নয়নের ফলে C এর বিকাশ ঘটে।

**সি++(C++):** ১৯৮০ সালে Bjarne Stroustrup বেল ল্যাবরেটরিতে C ভাষার বৈশিষ্ট্যের সাথে অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং এর বৈশিষ্ট্য যুক্ত করে নতুন এক প্রোগ্রামিং ভাষা তৈরি করেন যা C++ নামে পরিজায়। এই ভাষাকে অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং ভাষা বলা হয়। এই ভাষার সাহায্যে বিভিন্ন সিস্টেম সফটওয়্যার, অ্যাপ্লিকেশন সফটওয়্যার, ডিভাইস ড্রাইভার ইত্যাদি তৈরি করা যায়।

**জাভাঃ** সান মাইক্রোসিস্টেম কোম্পানি জাভা প্রোগ্রামিং ভাষাটি তৈরি করেন। [James Gosling কে জাভা প্রোগ্রামিং ভাষার জনক বলা হয়। এটি একটি অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং ভাষা। বর্তমান যুগে জাভার জনপ্রিয়তার মূল কারণ হল এই ভাষা ব্যবহার করে লেখা প্রোগ্রাম যেকোন অপারেটিং সিস্টেমে চালানো যায়। এই প্রোগ্রামিং ভাষা ব্যবহার করে ওয়েব অ্যাপ্লিকেশন, ডেস্কটপ অ্যাপ্লিকেশন, মোবাইল অ্যাপ্লিকেশন ইত্যাদি তৈরি করা যায়।](#)

**পাইথনঃ** ১৯৯১ সালে Guido Van Rossum পাইথন প্রোগ্রামিং ভাষা তৈরি করেন। এই ভাষা একই সাথে অবজেক্ট ওরিয়েন্টেড ও স্ট্রাকচার্ড প্রোগ্রামিং ভাষার বৈশিষ্ট্য সাপোর্ট করে।

**ভিজুয়াল বেসিকঃ** ভিজুয়াল বেসিক একটি ইন্ডেন্ট ড্রাইভেন প্রোগ্রামিং ডিজাইন ভিত্তিক ভাষা। এটি মাইক্রোসফট তৈরি করে। এই ভাষা ব্যবহার করে খুব সহজেই কাস্টমাইজড অ্যাপ্লিকেশন সফটওয়্যার তৈরি করা যায়।

**অ্যালগলঃ** এর পূর্ণনাম Algorithmic Language। এটি ব্যবহৃত হতো মূলত বৈজ্ঞানিক গবেষণায় বিভিন্ন সমস্যার সমাধানে।

**ফোরট্রানঃ** Fortran এর অর্থ Formula Translator যা উচ্চস্তর প্রোগ্রামিং ভাষাগুলোর মধ্যে সবচেয়ে আদিমতম ভাষা। IBM এর গবেষক জন বাকাস IBM মেইনফ্রেম কম্পিউটার এর জন্য এটি তৈরি করেন। এটি গাণিতিক জটিল হিসাব-নিকাশের জন্য এবং প্রকৌশল বিজ্ঞানে গবেষণার কাজে বেশ জনপ্রিয়।

**ওরাকলঃ** ওরাকল একটি RDBMS(Relational Database Management System) সফটওয়্যার যা ওরাকল কর্পোরেশন তৈরি করে। এটি ডেটাবেজ সফটওয়্যারগুলোর মধ্যে সবচেয়ে জনপ্রিয়।

**মধ্যমস্তরের ভাষাঃ** যে প্রোগ্রামিং ভাষায় নিম্নস্তরের ভাষার সুবিধা যেমন- বিট পর্যায়ে প্রোগ্রামিং বা সিস্টেম সফটওয়্যার এর মাধ্যমে হার্ডওয়্যার নিয়ন্ত্রণ এবং উচ্চস্তরের ভাষার সুবিধা যেমন- অ্যাপ্লিকেশন সফটওয়্যার তৈরি করা যায় তাকে মধ্যম স্তরের ভাষা বলা হয়। মধ্যম স্তরের ভাষার উদাহরণ হল – C, Forth, Dbase, WordStar ইত্যাদি।

**চতুর্থ প্রজন্মের ভাষা(4<sup>th</sup> Generation Language-4GL):** 4GL এর পূর্ণরূপ Fourth Generation Language। চতুর্থ প্রজন্মের ভাষাকে অতি উচ্চ স্তরের ভাষা বলা হয়। চতুর্থ প্রজন্মের ভাষা হলো ডেটাবেজ সংক্রান্ত ভাষা। অর্থাৎ এই প্রজন্মের ভাষার সাহায্যে ডেটাবেজ তৈরি, আপডেট, ডিলেট সহ ডেটাবেজ সম্পর্কিত সকল কাজ সম্পাদন করা যায়। এই প্রজন্মের ভাষাকে non-procedural বা Functional Language বলা হয়। কারণ এই প্রজন্মের ভাষার ব্যবহারের ক্ষেত্রে যে তথ্যাবলি দরকার কেবল তা বলে দিলেই হয়, কীভাবে কুয়েরি করা যাবে তা বলার দরকার হয় না। এই প্রজন্মের ভাষার উদাহরণ হল SQL, Oracle ইত্যাদি।

**পঞ্চম প্রজন্মের ভাষা(5<sup>th</sup> Generation Language-5GL):** 5GL এর পূর্ণরূপ Fifth Generation Language। পঞ্চম প্রজন্মের ভাষাকে স্বাভাবিক ভাষা (Natural Language) ও বলা হয়। Artificial Intelligence বা কৃত্রিম বুদ্ধিমত্তা নির্ভর যন্ত্র তৈরিতে এই প্রজন্মের ভাষা ব্যবহৃত হয়। পঞ্চম প্রজন্মের ভাষায় লেখা প্রোগ্রামকে মেশিন ভাষায় রূপান্তরের জন্য ইন্টেলিজেন্ট কম্পাইলার ব্যবহৃত হয়। এই প্রজন্মের ভাষা ব্যবহার করে মানুষ যন্ত্রকে মৌখিক নির্দেশ দিতে পারে। পঞ্চম প্রজন্মের ভাষার উদাহরণ হচ্ছে PROLOG(PROgramming LOGic), LISP, Mercury ইত্যাদি।

#### পঞ্চম অধ্যায় পাঠ-২: অনুবাদক প্রোগ্রাম (অ্যাসেম্বলার, কম্পাইলার, ইন্টারপ্রেটার)

**অনুবাদক প্রোগ্রামঃ** যে প্রোগ্রাম উৎস(Source) প্রোগ্রামকে বস্তু(Object) প্রোগ্রামে রূপান্তর করে তাকে অনুবাদক প্রোগ্রাম বলে। মেশিন ভাষায় লেখা প্রোগ্রামকে বলা হয় বস্তু প্রোগ্রাম (Object Program) এবং অন্য যেকোনো ভাষায় লেখা প্রোগ্রামকে বলা হয় উৎস প্রোগ্রাম (Source program)।

অনুবাদক প্রোগ্রাম উৎস প্রোগ্রামকে ইনপুট হিসেবে নেয় এবং বস্তু প্রোগ্রামকে আউটপুট হিসেবে দেয়। প্রোগ্রাম অনুবাদের সময় উৎস প্রোগ্রামে যদি কোন ভুল থাকে, তবে তা সংশোধন করার জন্য ব্যবহারকারীকে Error Message দেয়।

#### অনুবাদক প্রোগ্রামের প্রকারভেদ-

- ১। অ্যাসেম্বলার(Assembler)
- ২। কম্পাইলার(Compiler)
- ৩। ইন্টারপ্রেটার(Interpreter)

**অ্যাসেম্বলারঃ** অ্যাসেম্বলার হলো এক ধরনের অনুবাদক প্রোগ্রাম যা অ্যাসেম্বলি ভাষায় লেখা প্রোগ্রামকে মেশিন ভাষায় রূপান্তর করে। এটি অ্যাসেম্বলি ভাষায় লেখা প্রোগ্রাম বা নেমোনিক কোডকে যান্ত্রিক ভাষায় রূপান্তর করে। এক্ষেত্রে প্রোগ্রামে কোনো ভুল থাকলে Error Message দেয়।

#### প্রধান কাজ সমূহ:

- ১। নেমোনিক কোডকে মেশিন ভাষায় অনুবাদ করা।
- ২। সাংকেতিক ঠিকানাকে মেশিন ভাষার ঠিকানায় রূপান্তর করা।
- ৩। সব নির্দেশ ও ডেটা প্রধান মেমোরিতে রাখা।
- ৪। প্রোগ্রামে কোনো ভুল থাকলে Error Message দেওয়া।
- ৫। প্রোগ্রামের সকল ভুল সংশোধনের পর প্রোগ্রাম কন্ট্রোলকে জানানো ইত্যাদি।

**কম্পাইলারঃ** কম্পাইলার হলো এক ধরনের অনুবাদক প্রোগ্রাম যা উচ্চস্তরের ভাষায় লেখা প্রোগ্রামকে মেশিন বা যান্ত্রিক ভাষায় রূপান্তর করে। অর্থাৎ উৎস প্রোগ্রামকে বস্তু প্রোগ্রামে রূপান্তর করে। কম্পাইলার দুই ধাপে অনুবাদকের কাজ সম্পন্ন করে –

প্রথম ধাপে কম্পাইলার উৎস প্রোগ্রামটি পড়ে এবং বস্তু প্রোগ্রামে রূপান্তর করে। এই ধাপে, সোর্স প্রোগ্রামে যদি কোন ভুল থাকে, তবে তা সংশোধন করার জন্য কম্পাইলার ব্যবহারকারীকে Error Message দেয়। এই Error Message কে কম্পাইলড টাইম ডায়াগনোস্টিক Error Message বলে। একবার প্রোগ্রাম কম্পাইল হয়ে গেলে পরবর্তীতে আর কম্পাইল করার প্রয়োজন হয় না। দ্বিতীয় ধাপে উপাত্ত বা ডেটার ভিত্তিতে ফলাফল প্রদর্শনের জন্য বস্তু প্রোগ্রামকে নির্বাহ করানো হয়।

#### কম্পাইলারের কাজঃ

- ১। উৎস প্রোগ্রামের স্টেটমেন্ট সমূহকে বস্তু প্রোগ্রামে বা মেশিন ভাষায় রূপান্তর।
- ২। সংশ্লিষ্ট সাব-রুটিন এর সাথে সংযোগের ব্যবস্থা প্রদান।
- ৩। প্রধান মেমোরির পরিসর চিহ্নিতকরণ।
- ৪। প্রোগ্রাম ভুল থাকলে অনুবাদের সময় ভুলের তালিকা প্রণয়ন।

#### কম্পাইলারের সুবিধাঃ

- ১। কম্পাইলার সম্পূর্ণ প্রোগ্রামটিকে একসাথে অনুবাদ করে। ফলে প্রোগ্রাম নির্বাহ দ্রুত হয়।
- ২। কম্পাইলারের মাধ্যমে রূপান্তরিত প্রোগ্রাম সম্পূর্ণরূপে মেশিন ভাষায় রূপান্তরিত হয়।
- ৩। একবার প্রোগ্রাম কম্পাইল করা হলে পরবর্তীতে আর কম্পাইলের প্রয়োজন হয় না।

#### কম্পাইলারের অসুবিধাঃ

- ১। কম্পাইলার প্রোগ্রামের সবগুলো ভুল একসাথে প্রদর্শন করে ফলে প্রোগ্রাম সংশোধনে বেশি সময় লাগে।
- ২। প্রোগ্রাম ডিবাগিং ও টেস্টিং এর কাজ ধীরগতি সম্পন্ন।
- ৩। কম্পাইলার বড় ধরনের প্রোগ্রাম হওয়ায় ইহা সংরক্ষণে মেমোরিতে বেশি জায়গা প্রয়োজন।

**ইন্টারপ্রেটারঃ** ইন্টারপ্রেটারও কম্পাইলারের মতো এক ধরনের অনুবাদক প্রোগ্রাম যা উচ্চস্তরের ভাষায় লেখা প্রোগ্রামকে লাইন বা লাইন মেশিন বা যান্ত্রিক ভাষায় রূপান্তর করে। এক্ষেত্রে কম্পাইলারের সাথে পার্থক্য হল, কম্পাইলার সম্পূর্ণ সোর্স প্রোগ্রামকে একসাথে অবজেক্ট প্রোগ্রামে রূপান্তর করে এবং সর্বশেষ ফলাফল প্রদান করে কিন্তু ইন্টারপ্রেটার সোর্স প্রোগ্রামটিকে লাইন-বাই-লাইন অবজেক্ট প্রোগ্রামে রূপান্তর করে এবং তাৎক্ষণিক ফলাফল প্রদর্শন করে।

#### ইন্টারপ্রেটারের কাজঃ

- ১। উৎস প্রোগ্রামের স্টেটমেন্ট সমূহকে বস্তু প্রোগ্রামে বা মেশিন ভাষায় রূপান্তর।
- ২। সংশ্লিষ্ট সাব-রুটিন এর সাথে সংযোগের ব্যবস্থা প্রদান।
- ৩। প্রধান মেমোরির পরিসর চিহ্নিতকরণ।

- ৪। প্রোগ্রাম ভুল থাকলে অনুবাদের সময় ভুলের তালিকা প্রণয়ন।

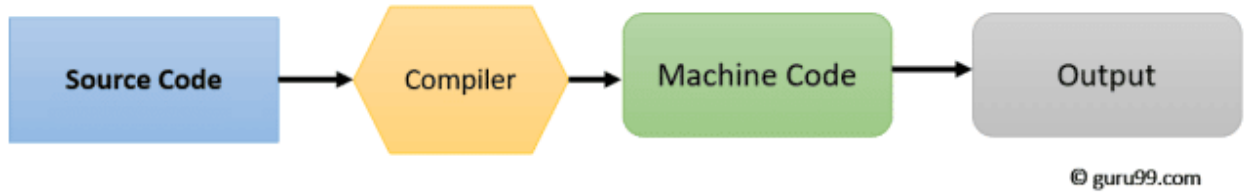
### ইন্টারপ্রেটারের সুবিধাঃ

- ১। ইন্টারপ্রেটার এর সবচেয়ে বড় সুবিধা হল এটি ইউজার ফ্রেন্ডলি।
- ২। এটি ব্যবহারে প্রোগ্রামের ভুল সংশোধন করা এবং পরিবর্তন করা সহজ হয়।
- ২। ইন্টারপ্রেটার প্রোগ্রাম আকারে ছোট হয় বলে মেমোরিতে কম জায়গা দখল করে।
- ৩। এটি সাধারণত ছোট কম্পিউটারে ব্যবহার করা হয়।

### ইন্টারপ্রেটারের অসুবিধাঃ

- ১। ইন্টারপ্রেটার যেহেতু প্রোগ্রাম লাইন-বাই-লাইন অনুবাদ করে, তাই অনুবাদ করতে কম্পাইলারের তুলনায় বেশি সময় প্রয়োজন।
- ২। ইন্টারপ্রেটার এর মাধ্যমে রূপান্তরিত প্রোগ্রাম সম্পূর্ণরূপে মেশিন প্রোগ্রামে রূপান্তরিত হয় না।
- ৩। প্রত্যেকবার প্রোগ্রাম নির্বাহের সময় অনুবাদ করার প্রয়োজন হয়।

### How Compiler Works



### How Interpreter Works



### কম্পাইলার এবং ইন্টারপ্রেটারের মধ্যে পার্থক্যঃ

#### কম্পাইলার

কম্পাইলার একটি অনুবাদক প্রোগ্রাম যা উচ্চস্তরের প্রোগ্রামিং ভাষায় লেখা একটি সম্পূর্ণ প্রোগ্রামকে একসাথে অনুবাদ করে।

ফলে প্রোগ্রাম নির্বাহের জন্য কম সময় প্রয়োজন।

#### ইন্টারপ্রেটার

ইন্টারপ্রেটারও এক ধরনের অনুবাদক প্রোগ্রাম যা উচ্চস্তরের প্রোগ্রামিং ভাষায় লেখা একটি প্রোগ্রামকে লাইন বাই লাইন অনুবাদ করে।

ফলে প্রোগ্রাম নির্বাহের জন্য বেশি সময় প্রয়োজন।

কম্পাইলার দ্বারা একটি প্রোগ্রাম একবার অনুবাদ করা হলে প্রতিবার কাজের পূর্বে পুনরায় অনুবাদ করার প্রয়োজন হয় না।

কম্পাইলার দ্বারা একটি প্রোগ্রাম অনুবাদ করলে পূর্ণাঙ্গ যান্ত্রিক প্রোগ্রামে রূপান্তরিত হয়।

ডিবাগিং এবং টেস্টিং এর ক্ষেত্রে ধীর গতি সম্পন্ন।

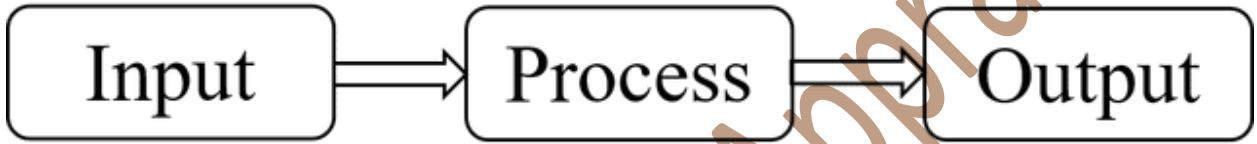
ইন্টারপ্রেটার দ্বারা একটি প্রোগ্রাম অনুবাদ করা হলে প্রত্যেকবার প্রোগ্রাম নির্বাহের সময় অনুবাদ করার প্রয়োজন হয়।

ইন্টারপ্রেটার দ্বারা একটি প্রোগ্রাম অনুবাদ করলে পূর্ণাঙ্গ যান্ত্রিক প্রোগ্রামে রূপান্তরিত হয় না।

ডিবাগিং ও টেস্টিং এর ক্ষেত্রে দ্রুত গতি সম্পন্ন।

### পঞ্চম অধ্যায় পাঠ-৩ প্রোগ্রামের সংগঠন এবং প্রোগ্রাম তৈরির ধাপসমূহ।

**প্রোগ্রাম সংগঠনঃ** প্রতিটি পূর্ণাঙ্গ প্রোগ্রামের তিনটি অপরিহার্য অংশ থাকে, যা পারস্পারিক সম্পর্কের মাধ্যমে একটি পূর্ণাঙ্গ প্রোগ্রাম গঠিত হয়। যেমন-



- ১। **ইনপুট:** প্রতিটি প্রোগ্রামে প্রসেস বা প্রক্রিয়া করার জন্য ইনপুট নেওয়ার ব্যবস্থা থাকতে হবে।
- ২। **প্রসেস বা প্রক্রিয়া:** ব্যবহারকারীর কাছ থেকে ইনপুট নিয়ে প্রসেস বা প্রক্রিয়া করার ব্যবস্থা থাকতে হবে।
- ৩। **আউটপুট:** ইনপুট নিয়ে প্রসেস করে আউটপুট দেখানোর ব্যবস্থাও থাকতে হবে।

**একটি আদর্শ প্রোগ্রামের নিম্নোক্ত বৈশিষ্ট্য সমূহ থাকতে হয়ঃ**

- ১। প্রোগ্রাম অবশ্যই সহজ ও বোধগম্য হতে হবে, যাতে অন্যকোন প্রোগ্রামার পরবর্তীতে আপডেট করতে পারে।
- ২। প্রোগ্রামটি রান করার জন্য সময় ও মেমোরি নূনতম হতে হবে।
- ৩। প্রোগ্রাম সহজে সম্প্রসারণযোগ্য হতে হবে।
- ৪। ডিবাগিং এবং টেস্টিং করা সহজতর হতে হবে।
- ৫। প্রোগ্রাম সহজে রক্ষণাবেক্ষণযোগ্য হতে হবে।

### প্রোগ্রাম তৈরির ধাপসমূহঃ

একটি প্রোগ্রাম তৈরির মাধ্যমে সাধারণত একটি নির্দিষ্ট সমস্যার সমাধান করা হয়ে থাকে। তাই একটি প্রোগ্রাম তৈরি করার জন্য কতগুলো ধাপ অনুসরণ করলে সমস্যাটি সহজে সমাধান করা যায়। ধাপগুলো নিম্নোক্ত আলোচনা করা হল-

- ১। সমস্যা নির্দিষ্টকরণ
- ২। সমস্যা বিশ্লেষণ
- ৩। প্রোগ্রাম ডিজাইন
- ৪। প্রোগ্রাম উন্নয়ন

- ৫। প্রোগ্রাম বাস্তবায়ন
- ৬। ডকুমেন্টেশন
- ৭। প্রোগ্রাম রক্ষণাবেক্ষণ

**সমস্যা নির্দিষ্টকরণঃ** একটি প্রোগ্রাম তৈরির মূল লক্ষ হল কোন একটি সমস্যার সমাধান করা। তাই প্রোগ্রাম তৈরির মাধ্যমে কোন সমস্যা সমাধানের পূর্বে সমস্যাটি অবশ্যই ভালোভাবে চিহ্নিত করতে হবে। সমস্যা নির্দিষ্টকরণের ক্ষেত্রে সমস্যাটি কী, সমস্যার বিষয়বস্তু কী ইত্যাদি বিষয়ে জানতে হবে। ভালভাবে সমস্যা নির্দিষ্ট করতে না পারলে প্রোগ্রাম যতই উন্নত হোক না কেন কাঙ্ক্ষিত সমাধান পাওয়া সম্ভব না।

**সমস্যা বিশ্লেষণঃ** সমস্যা নির্দিষ্ট করার পরের ধাপটি হল সমস্যা বিশ্লেষণ। সমস্যাটি কীভাবে সমাধান করা যায়, কতভাবে সমাধান করা যায়, যদি একাধিক ভাবে সমাধান করা যায় তাহলে কোনটি সবচেয়ে ইফেক্টিভ সমাধান তা বিশ্লেষণ করাই হল সমস্যা বিশ্লেষণ। এক্ষেত্রে সমস্যাটিকে ছোট ছোট অংশে ভাগ করে সমাধান করার চেষ্টা করা হয়। সমস্যা সমাধানের ক্ষেত্রে কতকগুলো বিষয় এই ধাপে বিবেচনা করা হলে থাকে। বিষয়গুলো হল-

- ১। কোন বিষয়গুলো প্রোগ্রাম ডেভেলপমেন্টের জন্য প্রয়োজন।
- ২। কোন পদ্ধতিতে প্রোগ্রাম ডিজাইন করা হবে।
- ৩। প্রোগ্রাম তৈরির ক্ষেত্রে কোন প্রোগ্রামিং ভাষাটি উপযুক্ত হবে।
- ৪। সমস্যায় কোন ধরনের ইনপুট এবং কোন ধরনের আউটপুট হবে ইত্যাদি।

**প্রোগ্রাম ডিজাইনঃ** প্রোগ্রাম ডিজাইন বলতে বুঝায়- সমস্যা বিশ্লেষণ ধাপে সমস্যাটিকে যে ছোট ছোট অংশে ভাগ করা হয়েছে তাদের পারস্পরিক সম্পর্ক ও সামগ্রিক সমাধান বের করে তার অ্যালগোরিদম অথবা ফ্লোচার্ট তৈরি করা।

**প্রোগ্রাম উন্নয়নঃ** সমস্যা সমাধানের জন্য যে অ্যালগোরিদম বা ফ্লোচার্ট তৈরি করা হয়েছে তা কোনো একটি উচ্চস্তরের প্রোগ্রামিং ভাষা ব্যবহার করে প্রোগ্রামে রূপদান করাকে বলা হয় প্রোগ্রাম উন্নয়ন। এক্ষেত্রে উচ্চস্তরের প্রোগ্রামিং ভাষা হিসেবে C, C++, java, python ইত্যাদি ব্যবহৃত হয়।

**প্রোগ্রাম বাস্তবায়নঃ** প্রোগ্রাম বাস্তবায়ন ধাপে প্রোগ্রাম এর টেস্টিং এবং ডিবাগিং করা হয়ে থাকে।

- ১। টেস্টিং
- ২। ডিবাগিং

**১। টেস্টিংঃ** প্রোগ্রাম টেস্টিং হচ্ছে, কোনো প্রোগ্রাম উন্নয়ন বা কোডিং সম্পন্ন করার পর প্রোগ্রামটির যে ধরনের আউটপুট বা ফলাফল হওয়া উচিত তা ঠিকমতো আসছে কিনা বা রান করছে কিনা তা যাচাই করা। এই ধাপে ভিন্ন ভিন্ন ইনপুট দিয়ে আউটপুটের অবস্থা পর্যবেক্ষণ করা হয়। এক্ষেত্রে যদি কোন অসঙ্গতি পাওয়া যায় তবে বুঝতে হবে প্রোগ্রাম কোডিংয়ের কোথাও ভুল হয়েছে। প্রোগ্রামে সাধারণত নিচের ভুলগুলো পরিলক্ষিত হয়। যথা:

- ১। ব্যাকরণগত ভুল
- ২। যৌক্তিক ভুল
- ৩। রান টাইম বা এক্সিকিউশন টাইম ভুল



**সিনট্যাক্স ভুল/ব্যাকরণগত ভুলঃ** প্রোগ্রামের মধ্যে প্রোগ্রামিং ভাষার ব্যাকরণগত যেসব ভুল থাকে তাকে বলা হয় সিনট্যাক্স ভুল। যেমন- বানান ভুল,কমা, ব্র্যাকেট ঠিকমতো না দেয়া, কোনো চলকের মান না জানানো প্রভৃতি। এসব ভুল সংশোধন করা খুবই সহজ, কারণ সিনট্যাক্স ভুল হলে অনুবাদক প্রোগ্রাম ভুলের বার্তা ছাপায়। যেমন- প্রোগ্রামে printf() কমান্ডের পরিবর্তে print() লেখা। সিনট্যাক্স ভুলকে কম্পাইল টাইম ভুলও বলা হয়।

**লজিক্যাল বা যৌক্তিক ভুলঃ** প্রোগ্রামে যুক্তির ভুল থাকলে তাকে বলে লজিক্যাল ভুল। সাধারণত সমস্যা ঠিকমতো না বুঝার জন্যই এ ভুল হয়। যেমন-  $a > b$  এর স্থলে  $a < b$  বা  $s = a + b$  এর স্থানে  $s = a - b$  লিখলে লজিক্যাল ভুল হয়। লজিক্যাল ভুলের ক্ষেত্রে একটি উত্তর পাওয়া যায় যদিও তা ভুল। এক্ষেত্রে অনুবাদক প্রোগ্রাম কোনো ভুলের বার্তা ছাপায় না বলে লজিক্যাল ভুল সংশোধন করা খুব কঠিন।

**রান টাইম বা এক্সিকিউশন টাইম ভুলঃ** রান টাইম ভুল প্রোগ্রাম এক্সিকিউশনের সময় ঘটে। যেমন- শূন্য দিয়ে ভাগ করা কিংবা ঋণাত্মক সংখ্যার বর্গমূল বা লগারিদম বের করা, ডাইনামিক মেমোরি অ্যালোকেশনের সময় অপরিপূর্ণ মেমোরি থাকা ইত্যাদি। অনুবাদক প্রোগ্রাম অনুবাদ করার সময় এই ধরনের ভুল নির্ণয় করতে পারে না। এই ধরনের ভুল সম্বলিত প্রোগ্রাম রান করবে কিন্তু প্রোগ্রাম বন্ধ হয়ে যেতে পারে। রান টাইম ভুল নির্ণয় এবং সংশোধন করা কঠিন। যৌক্তিক ভুল এক ধরনের রান-টাইম ভুল কারণ এই ধরনের ভুল কম্পাইলার নির্ণয় করতে পারে না বা ডিবাগিং এর মাধ্যমে নির্ণয় করা যায় না।

**২। ডিবাগিংঃ** আমরা প্রোগ্রাম টেস্টিং এর ক্ষেত্রে প্রোগ্রামে বিভিন্ন ধরনের ভুল সম্পর্কে জেনেছি। প্রোগ্রামে যেকোনো ভুল চিহ্নিত করতে পারলে সেই ভুলকে বলা হয় বাগ (Bug)। উক্ত ভুল বা Bug কে সমাধান করাকে বলা হয় ডিবাগ (Debug)। অর্থাৎ প্রোগ্রামের ভুল খুঁজে বের করে তা সমাধান করার পদ্ধতিকে বলা হয় ডিবাগিং। এক্ষেত্রে ডিবাগিং এর মাধ্যমে Syntax Error সমাধান করা সহজ কিন্তু Logical Error এবং Run-time Error সমাধান করা তুলনামূলক জটিল। ১৯৪৫ সালে মার্চ-১ কম্পিউটারের ভিতরে একটি মথপোকা ঢুকে বাসা বাধে ফলে কম্পিউটারটি অকার্যকর হয়ে যায়। তখন থেকে ডিবাগিং কথাটির উৎপত্তি।

**ডকুমেন্টেশনঃ** প্রোগ্রাম ডেভেলপমেন্টের সময় ভবিষ্যতে প্রোগ্রাম রক্ষণাবেক্ষণের কথা ভেবে প্রোগ্রামের বিভিন্ন অংশের বিবরণ কमेंট হিসেবে লিখে রাখতে হয়। প্রোগ্রামের বিভিন্ন অংশের বিবরণ কमेंট হিসেবে লিপিবদ্ধ করাকে প্রোগ্রাম ডকুমেন্টেশন বলে। প্রোগ্রামের ডকুমেন্টেশন লেখা থাকলে যেকোন প্রোগ্রামার খুব সহজেই প্রোগ্রামের আপডেট করতে পারে। প্রোগ্রাম রক্ষণাবেক্ষণে ডকুমেন্টেশনের গুরুত্ব অপরিসীম। ডকুমেন্টেশনে নিম্নলিখিত বিষয়সমূহ অন্তর্ভুক্ত করা হয়ঃ

- ১. প্রোগ্রামের বর্ণনা।
- ২. অ্যালগোরিদম বা ফ্লোচার্ট
- ৩. লিখিত প্রোগ্রাম
- ৪. নির্বাহের জন্য প্রয়োজনীয় কাজের তালিকা
- ৫. ফলাফল

**প্রোগ্রাম রক্ষণাবেক্ষণঃ** সময়ের সাথে পরিবেশ-পরিস্থিতি পরিবর্তনের কারণে প্রোগ্রামের পরিবর্তন বা আধুনিকীকরণ করা প্রয়োজন হয়। এ ধরনের কাজ রক্ষণাবেক্ষণ ধাপের অন্তর্ভুক্ত। যেমন-

- ১। ভুল সংশোধন
- ২। কর্মক্ষমতা বৃদ্ধি

- ৩। প্রোগ্রামের নতুন ফিচার যুক্ত করা
- ৪। অপ্রয়োজনীয় অংশ বাদ দেওয়া ইত্যাদি।

### পঞ্চম অধ্যায় পাঠ-৪ অ্যালগোরিদম, ফ্লোচার্ট এবং সূডোকোড।

**অ্যালগোরিদমঃ** কোনো একটি নির্দিষ্ট সমস্যা সমাধানের জন্য যুক্তিসম্মত সসীম সংখ্যক পর্যায়ক্রমিক ধারা বর্ণনাকে একত্রে অ্যালগোরিদম বলা হয়। কোনো সমস্যাকে কম্পিউটার প্রোগ্রামিং দ্বারা সমাধান করার পূর্বে কাগজে-কলমে সমাধান করার জন্যই অ্যালগোরিদম ব্যবহার করা হয়। আরব গণিতবিদ ‘আল খারিজমী’ এর নাম অনুসারে অ্যালগোরিদম নামকরণ করা হয়েছে।

#### অ্যালগোরিদম তৈরির শর্তঃ

- ১। ইনপুট এবং আউটপুট স্পষ্টভাবে নির্ধারণ করতে হবে।
- ২। অ্যালগোরিদমের প্রত্যেকটি ধাপ স্পষ্ট হতে হবে যাতে সহজে বোঝা যায়।
- ৩। সসীম সংখ্যক ধাপে সমস্যার সমাধান হতে হবে।
- ৪। অ্যালগোরিদম ব্যাপকভাবে প্রয়োগ উপযোগী হতে হবে।
- ৫। অ্যালগোরিদমে কোন কম্পিউটার কোড থাকা যাবে না। বরং অ্যালগোরিদম এমনভাবে লিখতে হবে যা একই ধরনের প্রোগ্রামিং ভাষার জন্য ব্যবহার করা যাবে।

#### অ্যালগোরিদম তৈরির সুবিধাঃ

- ১। এটি একটি ধাপ-ভিত্তিক উপস্থাপনা ফলে সহজে প্রোগ্রামের উদ্দেশ্য বোঝা যায়।
- ২। একটি অ্যালগোরিদম একটি নির্দিষ্ট পদ্ধতি ব্যবহার করে।
- ৩। এটি কোনও প্রোগ্রামিং ভাষার উপর নির্ভরশীল নয়, তাই প্রোগ্রামিং জ্ঞান ছাড়াই যেকারো পক্ষে এটি বোঝা সহজ।
- ৪। একটি অ্যালগোরিদমের প্রতিটি ধাপের নিজস্ব লজিকাল ক্রম আছে তাই এটি ডিবাগ করা সহজ।
- ৫। প্রোগ্রাম পরিবর্তন ও পরিবর্তনে সহায়তা করে।

#### দুটি সংখ্যার গড় নির্ণয়ের অ্যালগোরিদম-

- ধাপ-১: শুরু করি।
- ধাপ-২: ইনপুট হিসেবে a ও b চলকের মান গ্রহণ করি।
- ধাপ-৩:  $avg = (a+b)/2$  নির্ণয় করি।
- ধাপ-৪: ফলাফল হিসেবে avg চলকের মান প্রদর্শন করি।
- ধাপ-৫: শেষ করি।

**ফ্লোচার্ট বা প্রবাহ চিত্রঃ** যে চিত্রভিত্তিক পদ্ধতিতে বিশেষ কতকগুলো চিহ্নের সাহায্যে কোনো একটি নির্দিষ্ট সমস্যার সমাধান করা হয় তাকে ফ্লোচার্ট বলা হয়। অন্যভাবে বলা যায়, অ্যালগোরিদমের চিত্ররূপই হল ফ্লোচার্ট। ফ্লোচার্টের সাহায্যে প্রোগ্রাম বোঝা সহজ হয় বলে এটি প্রোগ্রামার ও ব্যবহারকারীর মাঝে সংযোগ রক্ষার জন্য ব্যবহৃত হয়।



## ফ্লোচার্ট তৈরি করার নিয়মাবলীঃ

- ১। প্রতিটি ফ্লোচার্টের অবশ্যই একটি শুরু এবং শেষ থাকবে।
- ২। নিয়ন্ত্রণ প্রবাহ অবশ্যই টপ থেকে শুরু হবে।
- ৩। নিয়ন্ত্রণ প্রবাহ অবশ্যই বোটম থেকে শেষ হবে।
- ৪। প্রচলিত চিহ্ন বা প্রতীক ব্যবহার করে ফ্লোচার্ট তৈরি করতে হবে।
- ৫। তীর চিহ্ন দিয়ে নিয়ন্ত্রণ প্রবাহ দেখাতে হবে।
- ৬। ফ্লোচার্টে কোন প্রোগ্রামিং ভাষা ব্যবহার করা যাবে না।
- ৭। চিহ্ন বা প্রতীক গুলো ছোট বড় হলে ক্ষতি নাই তবে আকৃতি ঠিক থাকতে হবে।

## ফ্লোচার্টের সুবিধাঃ

- ১। একটি প্রোগ্রামের যুক্তির মধ্যে যোগাযোগের চমৎকার উপায় হলো ফ্লোচার্ট।
- ২। ফ্লোচার্ট ব্যবহার করে সমস্যা বিশ্লেষণ করা সহজ।
- ৩। প্রোগ্রাম উন্নয়নের সময়, ফ্লোচার্ট একটি বুথ্রিন্টের ভূমিকা পালন করে, যা প্রোগ্রামের উন্নয়ন প্রক্রিয়াকে আরও সহজ করে তোলে।
- ৪। ফ্লোচার্ট এর সাহায্যে প্রোগ্রাম বা সিস্টেম রক্ষণাবেক্ষণ সহজ হয়।
- ৫। ফ্লোচার্টকে যেকোন প্রোগ্রামিং ভাষার কোডে রূপান্তর করা সহজ।

## ফ্লোচার্টের প্রকারভেদঃ ফ্লোচার্টকে প্রধানত দুইভাগে ভাগ করা যায়। যথা-

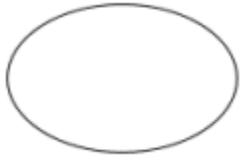
- ১। সিস্টেম ফ্লোচার্ট – সিস্টেম ফ্লোচার্টে ডেটা গ্রহণ, প্রক্রিয়াকরণ, সংরক্ষণ এবং ফলাফল প্রদর্শনের প্রবাহ দেখানো হয়। কোন সিস্টেমের কার্যপ্রণালী বোঝাতে সিস্টেম ফ্লোচার্ট ব্যবহৃত হয়।
- ২। প্রোগ্রাম ফ্লোচার্ট – প্রোগ্রাম ফ্লোচার্টে প্রোগ্রামের বিভিন্ন ধাপের বিস্তারিত বিবরণ দেওয়া হয়। এছাড়া প্রোগ্রামের ভুল নির্ণয় ও সংশোধনে প্রোগ্রাম ফ্লোচার্ট ব্যবহৃত হয়।

## ফ্লোচার্ট গঠনের মৌলিক ধরণ বা স্ট্রীকচারঃ

- ১। সরল অনুক্রম (Simple Sequence) – এই স্ট্রীকচারে প্রোগ্রামের নির্দেশগুলো সরল অনুক্রমে ধারাবাহিকভাবে নির্বাহ হয়ে থাকে।
- ২। নির্বাচন বা সিলেকশন (Selection)- কোন একটি শর্তের সত্য বা মিথ্যার উপর ভিত্তি করে সিদ্ধান্ত নিয়ে কার্য নির্বাহের ক্ষেত্রে এই স্ট্রীকচার ব্যবহৃত হয়।
- ৩। পুনরাবৃত্তি বা লুপ (Loop)- একই ধরণের কাজ পুনরাবৃত্তি করার জন্য এই স্ট্রীকচার ব্যবহৃত হয়।
- ৪। জাম্প (Jump)- এই স্ট্রীকচারে প্রোগ্রামের প্রবাহ সরল অনুক্রমের পরিবর্তে কোন শর্তের সত্য বা মিথ্যার উপর ভিত্তি করে উপরের বা নিচের নির্দিষ্ট কোন নির্দেশ নির্বাহ হতে থাকে।

## ফ্লোচার্টে ব্যবহৃত প্রতিক সমূহ এবং এদের ব্যবহারঃ

প্রতিক	প্রতিকের নাম	ব্যবহার
--------	-----------------	---------



ডিম্বক  
(Oval)

এটি ফ্লোচার্টের শুরু এবং শেষ নির্দেশ করে।



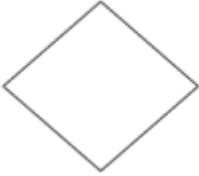
সামন্তরিক

ইনপুট নেওয়া ও আউটপুট প্রদর্শন নির্দেশ করে।



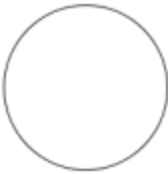
আয়তাকার

এই প্রতীক কোন প্রক্রিয়াকরণ নির্দেশ করে।



হীরক

যখন শর্তের সাপেক্ষে কোন প্রক্রিয়া সম্পন্ন করতে হয় তখন ব্যবহৃত হয়। এই প্রতীকের মধ্যে শর্ত লেখা হয়।



বৃত্ত

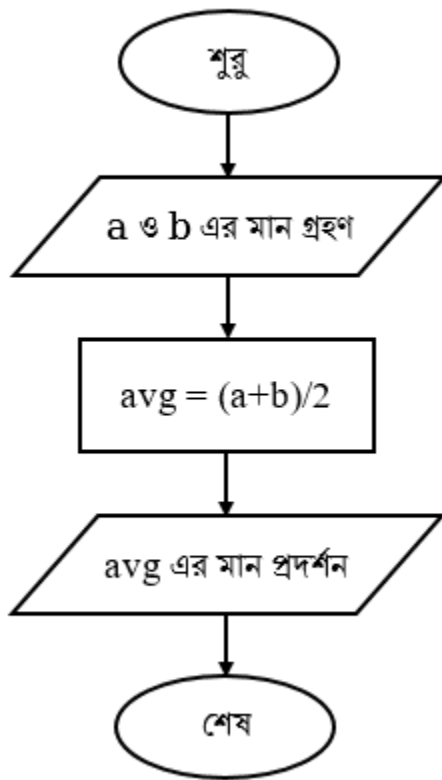
একাদিক শাখা যখন একটি বিন্দুতে মিলিত হয় তখন এই প্রতীক ব্যবহৃত হয়।



তীর চিহ্ন

প্রবাহের দিক নির্দেশে ব্যবহৃত হয়।

দুটি সংখ্যার গড় নির্ণয়ের ফ্লোচার্ট-



**সূডোকোড (Pseudo Code):** একটি প্রোগ্রামের কার্যপ্রণালী বর্ণনা বা উপস্থাপনার জন্য ইংরেজি ভাষায় লেখা কতগুলো নির্দেশনার সমষ্টিকে একত্রে সূডোকোড বলে। সূডোকোডকে অ্যালগোরিদমের পূর্ব-পস্তুতি বা অনেক সময় অ্যালগোরিদমের বিকল্প হিসেবে বিবেচনা করা হয়। সূডো(Pseudo) একটি গ্রীক শব্দ যার অর্থ হচ্ছে ছদ্ম।

**দুটি সংখ্যার গড় নির্ণয়ের সূডোকোড-**

- Start
- Input a and b
- $avg = (a+b)/2$
- Print avg
- Stop

পঞ্চম অধ্যায় পাঠ-৫ সাধারণ গাণিতিক সমস্যা সম্পর্কিত অ্যালগোরিদম এবং ফ্লোচার্ট।

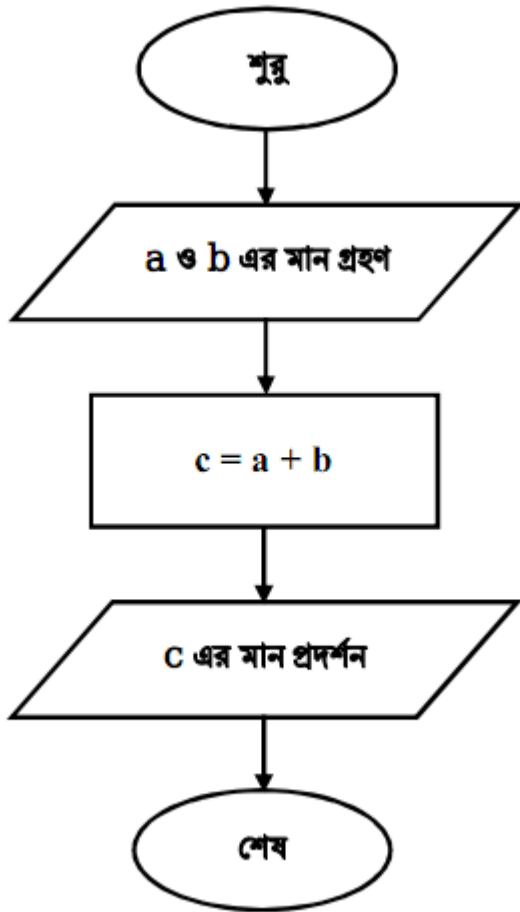
১। দুইটি সংখ্যা ইনপুট নিয়ে যোগফল নির্ণয় করার অ্যালগোরিদম ও ফ্লোচার্ট।

**অ্যালগোরিদমঃ**

- ধাপ-১: শুরু করি।

- ধাপ-২: ইনপুট হিসেবে a ও b চলকের মান গ্রহণ করি।
- ধাপ-৩:  $c=a+b$  নির্ণয় করি।
- ধাপ-৪: ফলাফল হিসেবে c চলকের মান প্রদর্শন করি।
- ধাপ-৫: শেষ করি।

ফ্লোচার্টঃ



অনুশীলনঃ পাঁচটি সংখ্যা ইনপুট নিয়ে যোগফল নির্ণয় করার অ্যালগোরিদম ও ফ্লোচার্ট তৈরি কর।

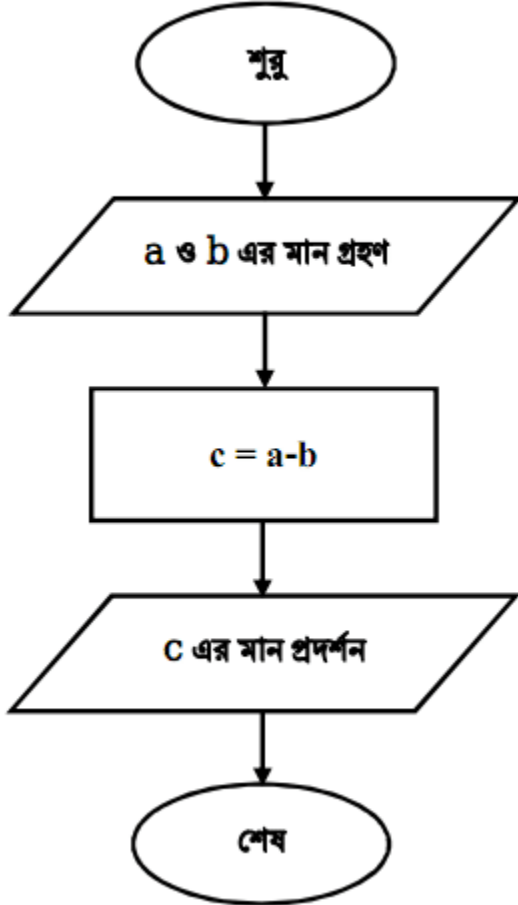
২। দুইটি সংখ্যা ইনপুট নিয়ে বিয়োগফল নির্ণয় করার অ্যালগোরিদম ও ফ্লোচার্ট।

অ্যালগোরিদমঃ

- ধাপ-১: শুরু করি।
- ধাপ-২: ইনপুট হিসেবে a ও b চলকের মান গ্রহণ করি।

- ধাপ-৩:  $c=a-b$  নির্ণয় করি।
- ধাপ-৪: ফলাফল হিসেবে  $c$  চলকের মান প্রদর্শন করি।
- ধাপ-৫: শেষ করি।

ফ্লোচার্টঃ

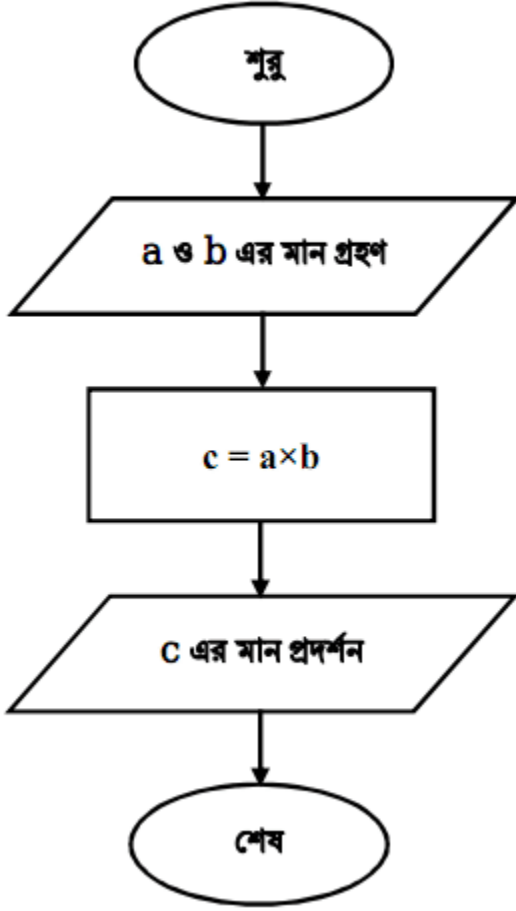


৩। দুইটি সংখ্যা ইনপুট নিয়ে গুণফল নির্ণয় করার অ্যালগোরিদম ও ফ্লোচার্ট।

অ্যালগোরিদমঃ

- ধাপ-১: শুরু করি।
- ধাপ-২: ইনপুট হিসেবে  $a$  ও  $b$  চলকের মান গ্রহণ করি।
- ধাপ-৩:  $c=a \times b$  নির্ণয় করি।
- ধাপ-৪: ফলাফল হিসেবে  $c$  চলকের মান প্রদর্শন করি।
- ধাপ-৫: শেষ করি।

ফ্লোচার্টঃ



অনুশীলনঃ পাঁচটি সংখ্যা ইনপুট নিয়ে গুণফল নির্ণয় করার অ্যালগোরিদম ও ফ্লোচার্ট তৈরি কর।

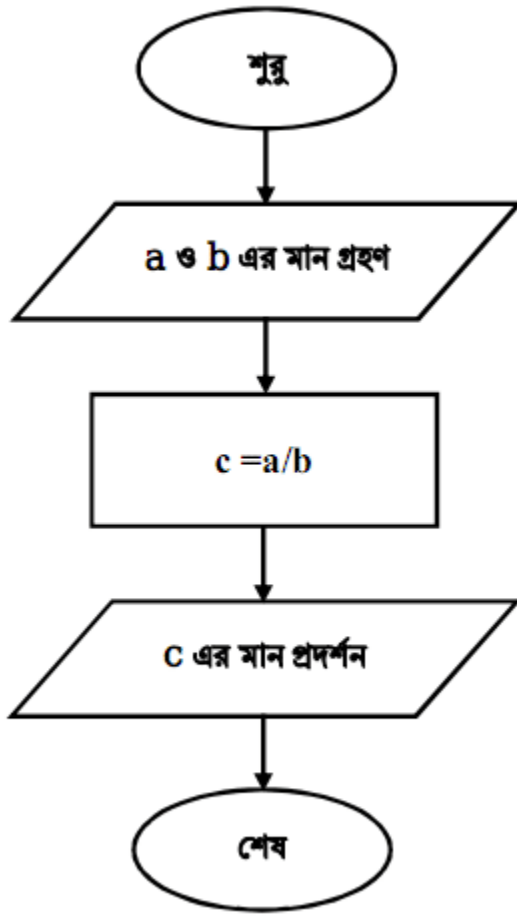
৪। দুইটি সংখ্যা ইনপুট নিয়ে ভাগফল নির্ণয় করার অ্যালগোরিদম ও ফ্লোচার্ট।

অ্যালগোরিদম

- ধাপ-১: শুরু করি।
- ধাপ-২: ইনপুট হিসেবে a ও b চলকের মান গ্রহণ করি।
- ধাপ-৩:  $c = a / b$  নির্ণয় করি।
- ধাপ-৪: ফলাফল হিসেবে c চলকের মান প্রদর্শন করি।
- ধাপ-৫: শেষ করি।

ফ্লোচার্টঃ





অনুশীলনঃ পাঁচটি সংখ্যা ইনপুট নিয়ে তাদের গড় নির্ণয় করার অ্যালগোরিদম ও ফ্লোচার্ট তৈরি কর।

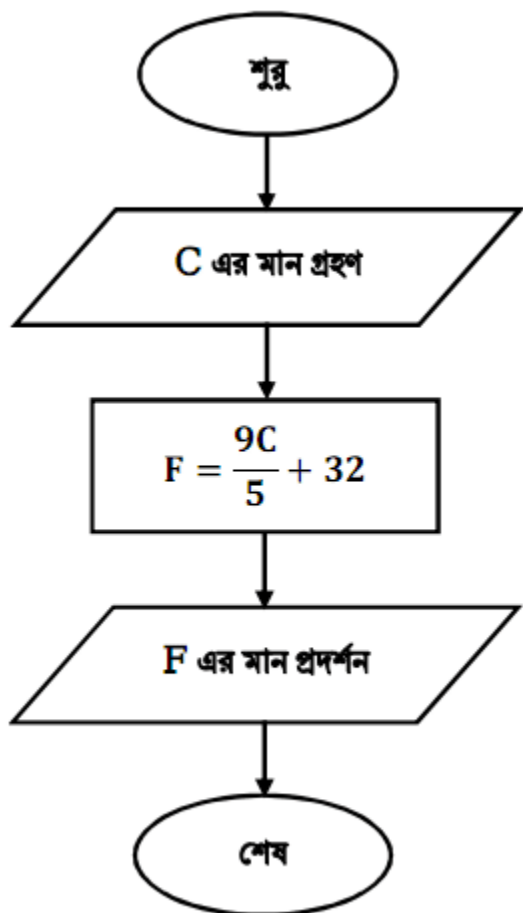
৫। সেলসিয়াস স্কেলের তাপমাত্রাকে ফারেনহাইট স্কেলের তাপমাত্রায় রূপান্তরের অ্যালগোরিদম ও ফ্লোচার্ট।

তাপমাত্রা পরিমাপের বিভিন্ন স্কেলের মধ্যে সম্পর্ক-  $C/5 = F-32 / 9 = K-273 / 5$

অ্যালগোরিদমঃ

- ধাপ-১: শুরু করি।
- ধাপ-২: ইনপুট হিসেবে C চলকে সেলসিয়াস স্কেলের তাপমাত্রা গ্রহণ করি।
- ধাপ-৩:  $F=(9C/5)+32$  সমীকরণে C চলকের মান বসিয়ে F চলকের মান নির্ণয় করি।
- ধাপ-৪: ফলাফল হিসেবে F চলকের মান প্রদর্শন করি।
- ধাপ-৫: শেষ করি।

ফ্লোচার্টঃ



**অনুশীলনঃ** সেলসিয়াস স্কেলের তাপমাত্রাকে কেলভিন স্কেলের তাপমাত্রায় রূপান্তরের অ্যালগোরিদম ও ফ্লোচার্ট তৈরি কর।

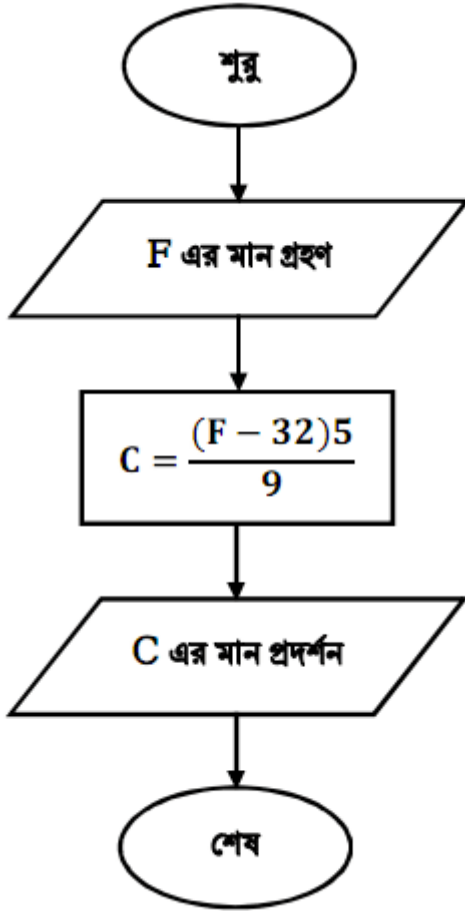
**৬। ফারেনহাইট স্কেলের তাপমাত্রাকে সেলসিয়াস তাপমাত্রায় রূপান্তরের অ্যালগোরিদম ও ফ্লোচার্ট।**

তাপমাত্রা পরিমাপের বিভিন্ন স্কেলের মধ্যে সম্পর্ক-  $C/5 = F-32 / 9 = K-273 / 5$

**অ্যালগোরিদমঃ**

- ধাপ-১: শুরু করি।
- ধাপ-২: ইনপুট হিসেবে F চলকে ফারেনহাইট স্কেলের তাপমাত্রা গ্রহণ করি।
- ধাপ-৩:  $C = ((F-32)5)/9$  সমীকরণে F চলকের মান বসিয়ে C চলকের মান নির্ণয় করি।
- ধাপ-৪: ফলাফল হিসেবে C চলকের মান প্রদর্শন করি।
- ধাপ-৫: শেষ করি।

ফ্লোচার্টঃ



অনুশীলনঃ ফারেনহাইট স্কেলের তাপমাত্রাকে কেলভিন তাপমাত্রায় রূপান্তরের অ্যালগোরিদম ও ফ্লোচার্ট তৈরি কর।

৭। ত্রিভুজের ভূমি ও উচ্চতা দেওয়া থাকলে ক্ষেত্রফল নির্ণয়ের অ্যালগোরিদম ও ফ্লোচার্ট।

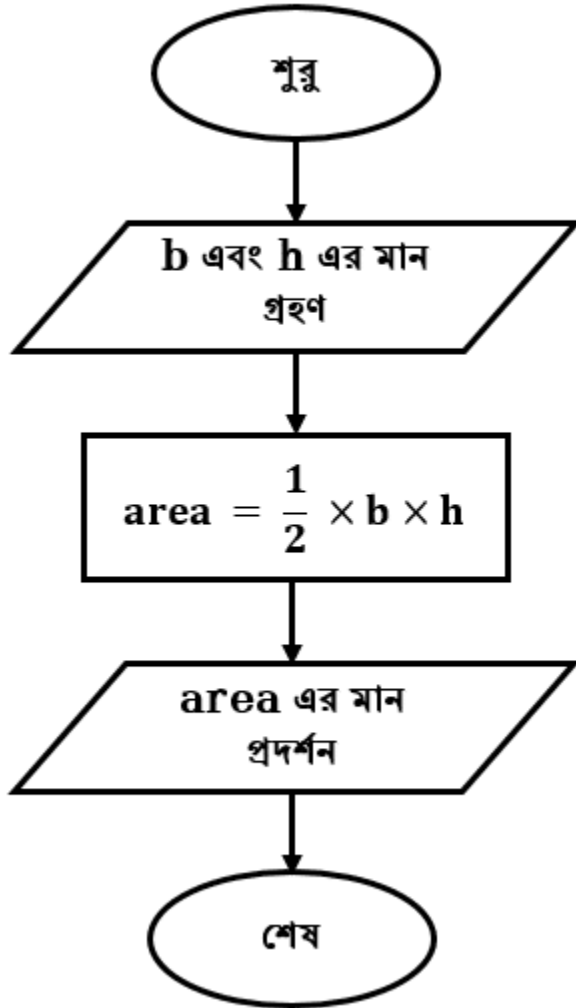
ত্রিভুজের ভূমি ও উচ্চতা দেওয়া থাকলে ক্ষেত্রফল নির্ণয়ের সূত্র, ক্ষেত্রফল  $= \frac{1}{2} \times \text{ভূমি} \times \text{উচ্চতা}$ ।

অ্যালগোরিদমঃ

- ধাপ-১: শুরু করি।
- ধাপ-২: ইনপুট হিসেবে b এবং h চলকে যথাক্রমে ত্রিভুজের ভূমি ও উচ্চতার মান গ্রহণ করি।
- ধাপ-৩:  $\text{area} = \frac{1}{2} \times b \times h$  সমীকরণে b এবং h চলকের মান বসিয়ে area চলকের মান নির্ণয় করি।
- ধাপ-৪: ফলাফল হিসেবে area চলকের মান প্রদর্শন করি।

- ধাপ-৫: শেষ করি।

ফ্লোচার্টঃ



৮। ত্রিভুজের তিনটি বাহুর দৈর্ঘ্য যথাক্রমে a, b এবং c দেওয়া থাকলে ক্ষেত্রফল নির্ণয়ের অ্যালগোরিদম ও ফ্লোচার্ট।

ত্রিভুজের তিনটি বাহুর দৈর্ঘ্য যথাক্রমে a, b এবং c দেওয়া থাকলে ক্ষেত্রফল নির্ণয়ের সূত্র

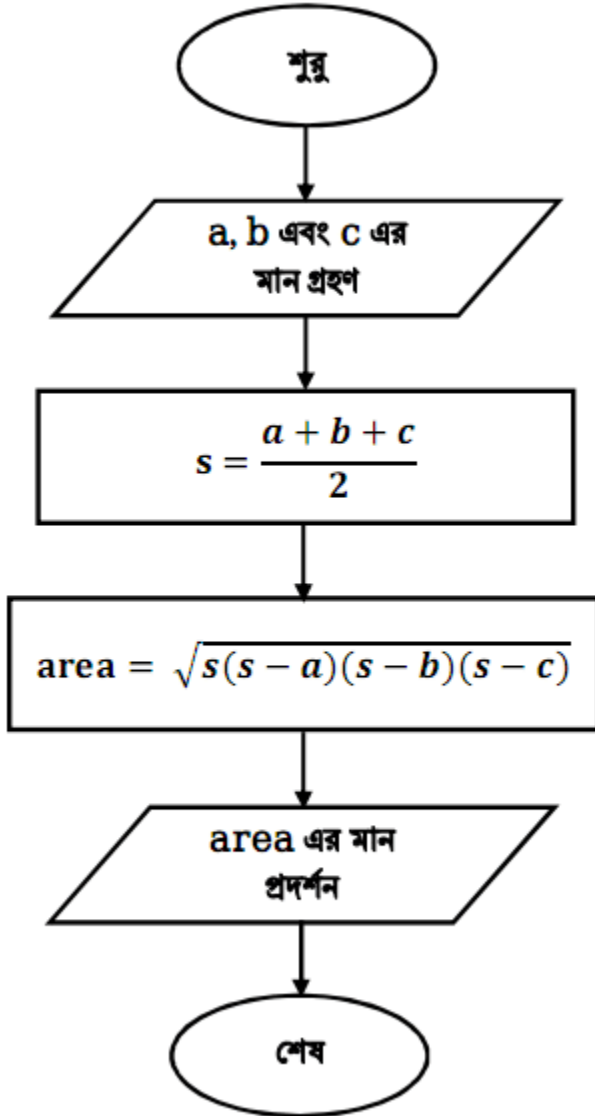
$$area = \sqrt{s(s-a)(s-b)(s-c)} \quad [\text{এখানে } s = \text{অর্ধপরিসীমা}]$$

$$\text{অর্ধপরিসীমা } s = (a+b+c)/2$$

অ্যালগোরিদমঃ

- ধাপ-১: শুরু করি।
- ধাপ-২: ইনপুট হিসেবে a, b এবং c চলকে যথাক্রমে ত্রিভুজের তিন বাহুর মান গ্রহণ করি।
- ধাপ-৩:  $s = (a+b+c)/2$  সমীকরণে a, b এবং c চলকের মান বসিয়ে ত্রিভুজের অর্ধপরিসীমা s চলকের মান নির্ণয় করি।
- ধাপ-৪:  $\text{area} = \sqrt{s(s-a)(s-b)(s-c)}$  সমীকরণের সাহায্যে area চলকের মান নির্ণয় করি।
- ধাপ-৫: ফলাফল হিসেবে area চলকের মান প্রদর্শন করি।
- ধাপ-৬: শেষ করি।

ফ্লোচার্টঃ



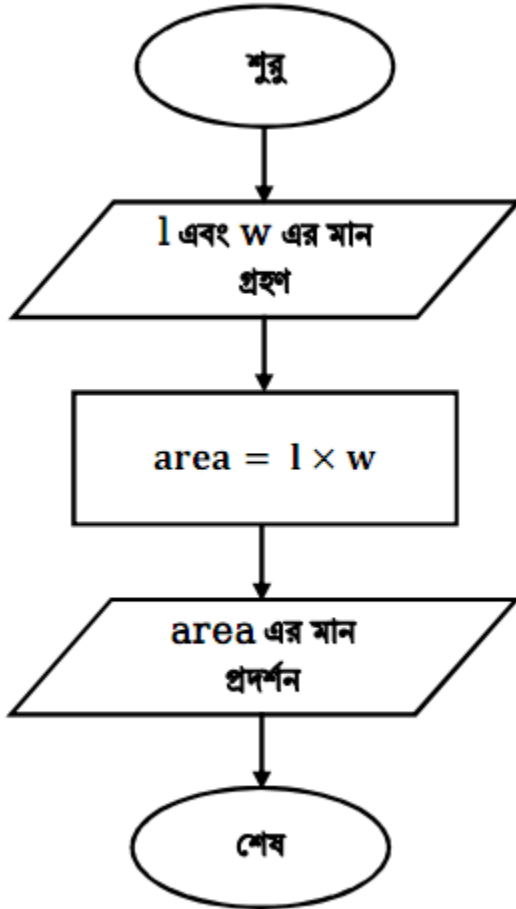
৯। আয়তক্ষেত্রের ক্ষেত্রফল নির্ণয়ের অ্যালগোরিদম ও ফ্লোচার্ট।

আয়তক্ষেত্রের দৈর্ঘ্য ও প্রস্থের মান দেওয়া থাকলে ক্ষেত্রফল নির্ণয়ের সূত্র, ক্ষেত্রফল=দৈর্ঘ্য  $\times$  প্রস্থ

অ্যালগোরিদমঃ

- ধাপ-১: শুরু করি।
- ধাপ-২: ইনপুট হিসেবে  $l$  এবং  $w$  চলকে যথাক্রমে আয়তক্ষেত্রের দৈর্ঘ্য ও প্রস্থের মান গ্রহণ করি।
- ধাপ-৩:  $\text{area} = l \times w$  সমীকরণে  $l$  এবং  $w$  চলকের মান বসিয়ে  $\text{area}$  চলকের মান নির্ণয় করি।
- ধাপ-৪: ফলাফল হিসেবে  $\text{area}$  চলকের মান প্রদর্শন করি।
- ধাপ-৫: শেষ করি।

ফ্লোচার্টঃ



অনুশীলনঃ সামান্তরিকের ক্ষেত্রফল নির্ণয়ের অ্যালগোরিদম ও ফ্লোচার্ট লেখ।

অনুশীলনঃ বর্গের ক্ষেত্রফল নির্ণয়ের অ্যালগোরিদম ও ফ্লোচার্ট লেখ।



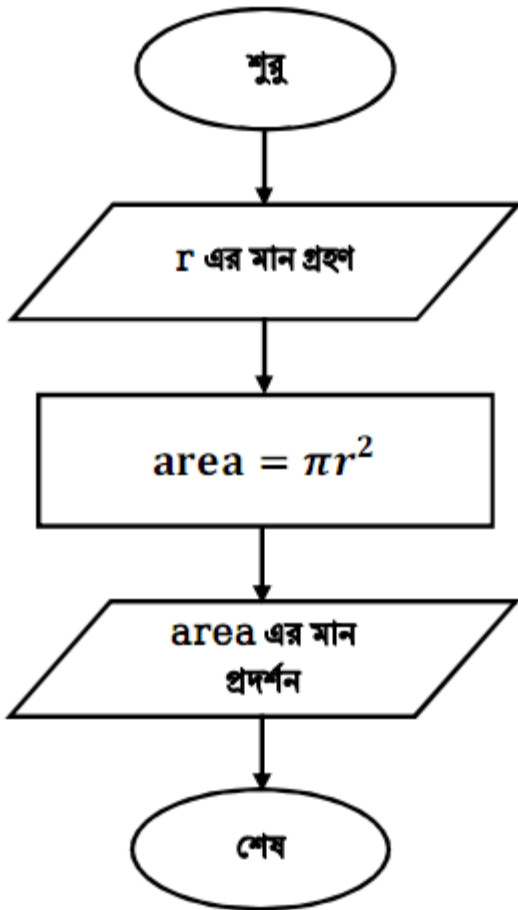
১০। বৃত্তের ক্ষেত্রফল নির্ণয়ের অ্যালগোরিদম ও ফ্লোচার্ট।

বৃত্তের ব্যাসার্ধের মান দেওয়া থাকলে ক্ষেত্রফল নির্ণয়ের সূত্র, ক্ষেত্রফল= $\pi r^2$

অ্যালগোরিদমঃ

- ধাপ-১: শুরু করি।
- ধাপ-২: ইনপুট হিসেবে  $r$  চলকে বৃত্তের ব্যাসার্ধের মান গ্রহণ করি।
- ধাপ-৩:  $\text{area} = \pi r^2$  সমীকরণে  $r$  চলকের মান বসিয়ে  $\text{area}$  চলকের মান নির্ণয় করি।
- ধাপ-৪: ফলাফল হিসেবে  $\text{area}$  চলকের মান প্রদর্শন করি।
- ধাপ-৫: শেষ করি।

ফ্লোচার্টঃ



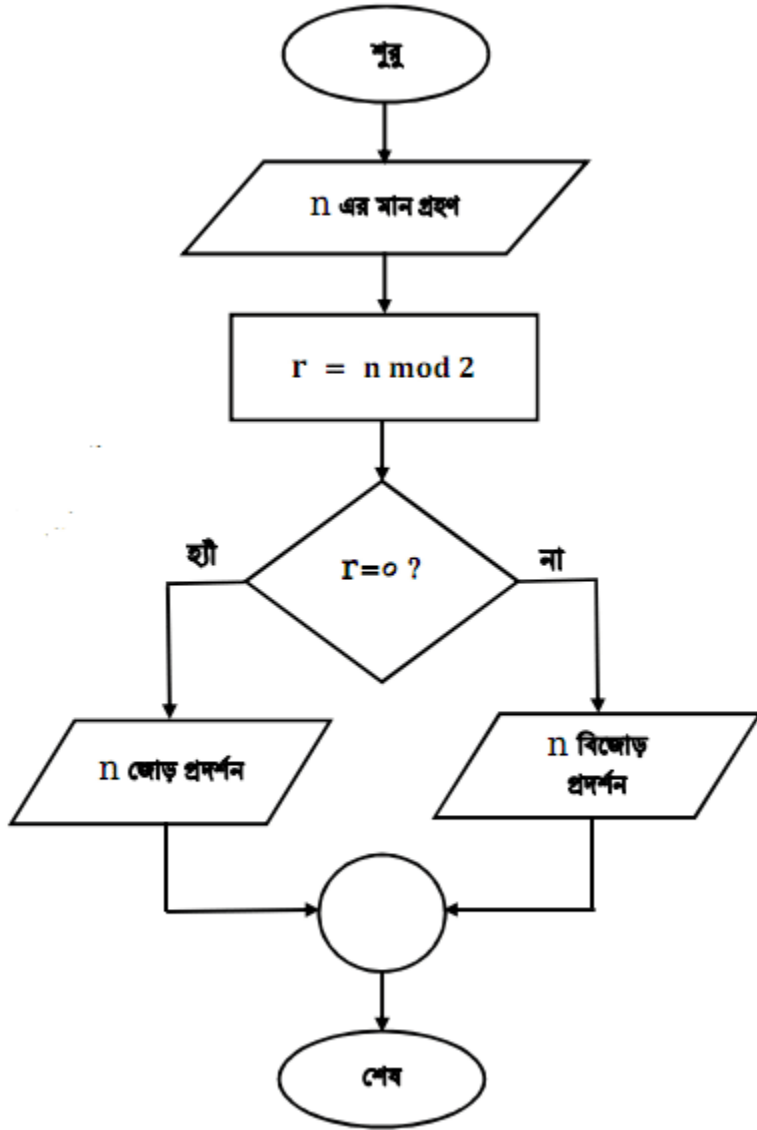
# পঞ্চম অধ্যায় পাঠ-৬ কন্ডিশনাল কন্ট্রোল স্টেটমেন্ট সম্পর্কিত অ্যালগোরিদম এবং ফ্লোচার্ট সমূহ।

১। কোন একটি পূর্ণ সংখ্যা জোড়/বিজোড় নির্ণয়ের অ্যালগোরিদম ও ফ্লোচার্ট।

অ্যালগোরিদম:

- ধাপ-১: শুরু করি।
- ধাপ-২: ইনপুট হিসেবে  $n$  চলকে একটি সংখ্যা গ্রহণ করি।
- ধাপ-৩:  $r = n \bmod 2$  নির্ণয় করি।
- ধাপ-৪: যদি  $r=0$  হয়, তাহলে সংখ্যাটি জোড় প্রদর্শন করি এবং ৬নং ধাপে যাই, অন্যথায় ৫নং ধাপে যাই।
- ধাপ-৫: সংখ্যাটি বিজোড় প্রদর্শন করি।
- ধাপ-৬: শেষ করি।

ফ্লোচার্টঃ

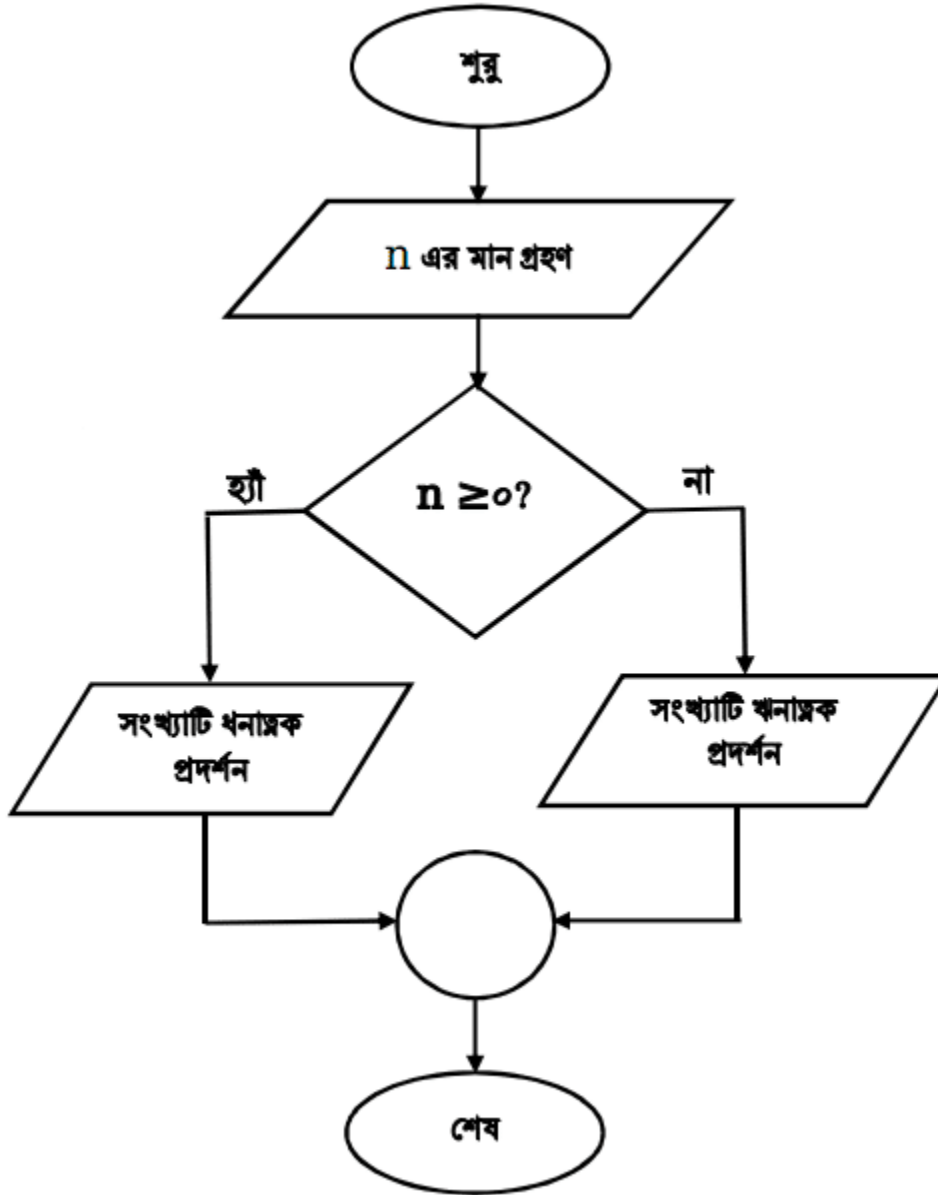


২। কোন সংখ্যা ধনাত্মক/ঋণাত্মক নির্ণয়ের অ্যালগোরিদম ও ফ্লোচার্ট।

অ্যালগোরিদমঃ

- ধাপ-১: শুরু করি।
- ধাপ-২: ইনপুট হিসেবে n চলকে একটি সংখ্যা গ্রহণ করি।
- ধাপ-৩: যদি  $(n \geq 0)$  হয়, তাহলে সংখ্যাটি ধনাত্মক প্রদর্শন করি এবং ৫নং ধাপে যাই, অন্যথায় ৪নং ধাপে যাই।
- ধাপ-৪: সংখ্যাটি ঋণাত্মক প্রদর্শন করি।
- ধাপ-৫: শেষ করি।

ফ্লোচার্টঃ



৩। কোন একটি সাল লিপ ইয়ার(অধিবর্ষ) নির্ণয়ের অ্যালগোরিদম ও ফ্লোচার্ট ।

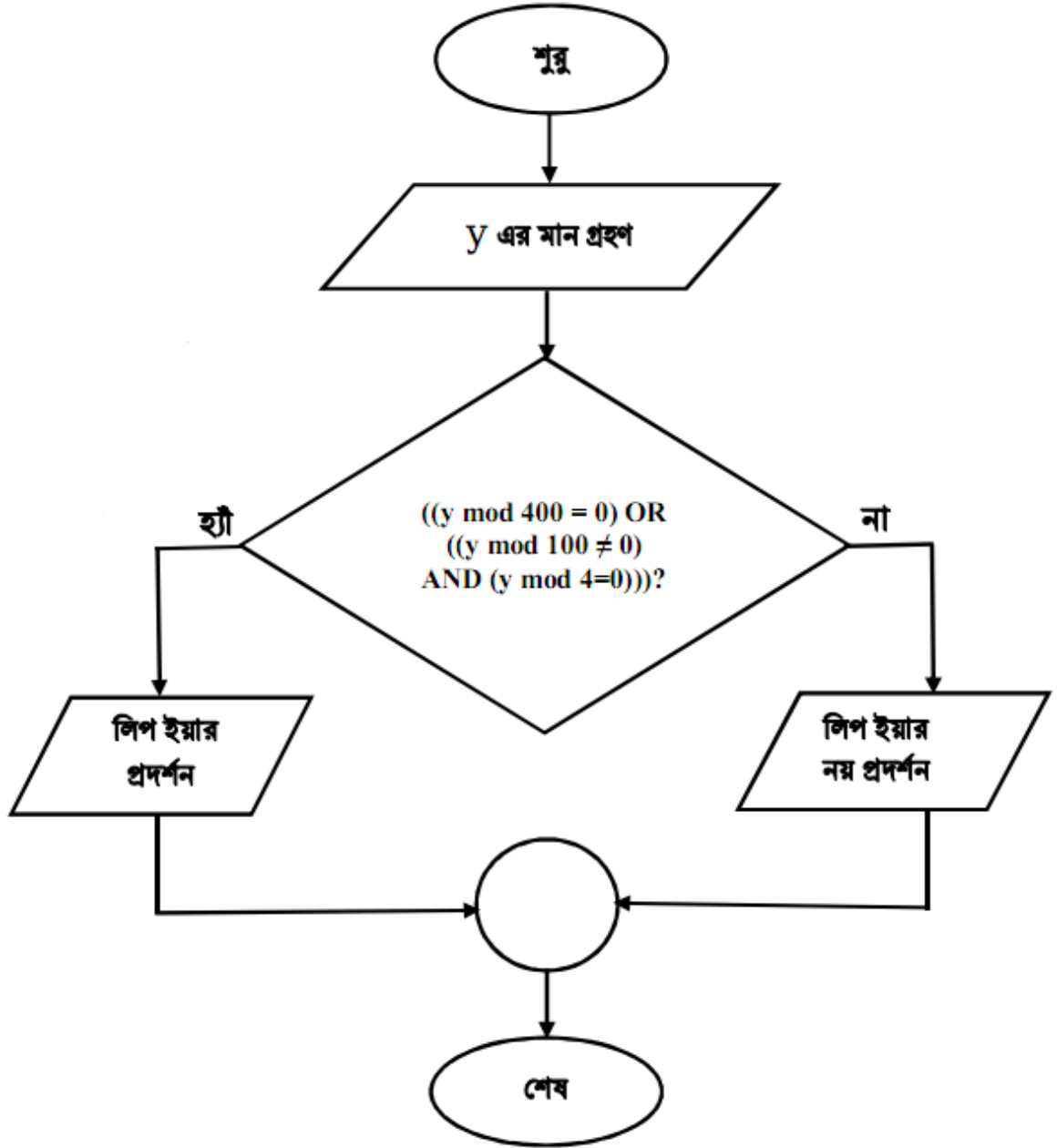
অধিবর্ষ বা লিপ ইয়ার হচ্ছে একটি বিশেষ বছর, যাতে সাধারণ বছরের তুলনায় একটি দিন বেশি থাকে। জ্যোতির্বিজ্ঞানিক বছর বা পৃথিবী যে সময়ে সূর্যের চারপাশে একবার ঘুরে আসে তার সময়কাল হচ্ছে প্রায় ৩৬৫ দিন ৫ ঘন্টা ৪৮ মিনিট ৪৭ সেকেন্ড, অথচ প্রচলিত গ্রেগরীয় বর্ষপঞ্জি মতে বছর হিসাব করা হয় ৩৬৫ দিনে। এভাবে প্রতিবছর প্রায় ছয় ঘন্টা সময় গোনার বাইরে থেকে যায় ও চার বছরে সেটা প্রায় এক দিনের সমান হয়। এই ঘাটতি পুষিয়ে নেয়ার জন্য প্রতি চার বছর পরপর ৩৬৬ দিনে বছর হিসাব করা হয়। গ্রেগরীয়

বর্ষপঞ্জিমে, প্রতি চার বছরে একবার ফেব্রুয়ারি মাসে ও বাংলা সনমতে ফাল্গুন মাসে এই অতিরিক্ত ১ দিন যোগ হয়।। যেমন: ২০১২ একটি অধিবর্ষ ও এর ফেব্রুয়ারি মাস হয়েছে ২৯ দিনে।

#### অ্যালগোরিদমঃ

- ধাপ-১: শুরু করি।
- ধাপ-২: ইনপুট হিসেবে  $y$  চলকে একটি সাল গ্রহণ করি।
- ধাপ-৩: যদি  $((y \bmod 400 = 0) \text{ OR } ((y \bmod 100 \neq 0) \text{ AND } (y \bmod 4 = 0)))$  হয়, তাহলে ৪নং ধাপে যাই, অন্যথায় ৫নং ধাপে যাই।
- ধাপ-৪: সালটি লিপ ইয়ার প্রদর্শন করি এবং ৬নং ধাপে যাই।
- ধাপ-৫: সালটি লিপ ইয়ার নয় প্রদর্শন করি।
- ধাপ-৬: শেষ করি।

#### ফ্লোচার্টঃ



প্রোগ্রাম দেখতে ক্লিক কর

৪। দুটি সংখ্যার মধ্যে বড় সংখ্যা নির্ণয়ের অ্যালগোরিদম ও ফ্লোচার্ট।

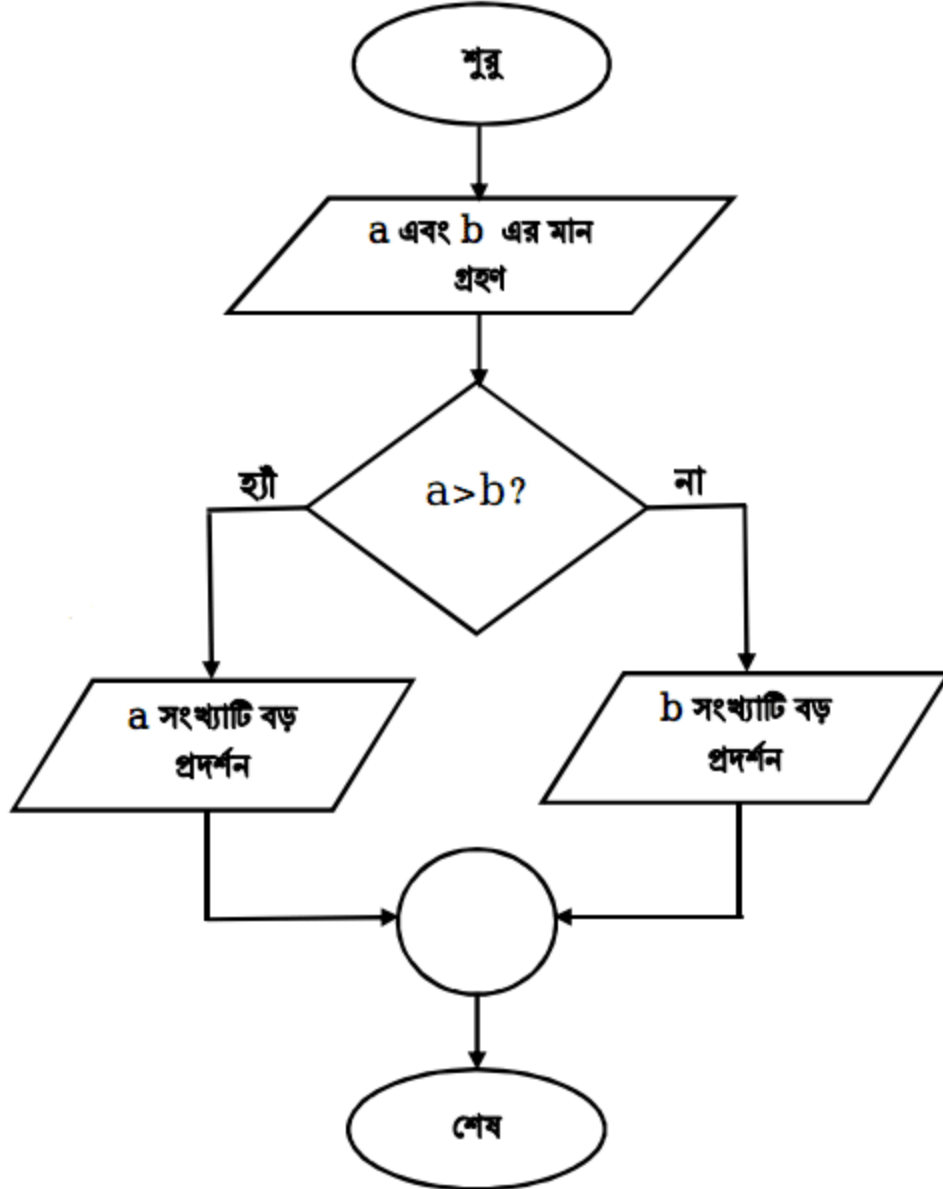
অ্যালগোরিদমঃ

- ধাপ-১: শুরু করি।



- ধাপ-২: ইনপুট হিসেবে a এবং b চলকে দুটি সংখ্যা গ্রহণ করি।
- ধাপ-৩: যদি  $(a > b)$  হয়, তাহলে a সংখ্যাটি বড় প্রদর্শন করি এবং ৫নং ধাপে যাই, অন্যথায় ৪নং ধাপে যাই।
- ধাপ-৪: b সংখ্যাটি বড় প্রদর্শন করি।
- ধাপ-৫: শেষ করি।

ফ্লোচার্টঃ



প্রোগ্রাম দেখতে ক্লিক কর

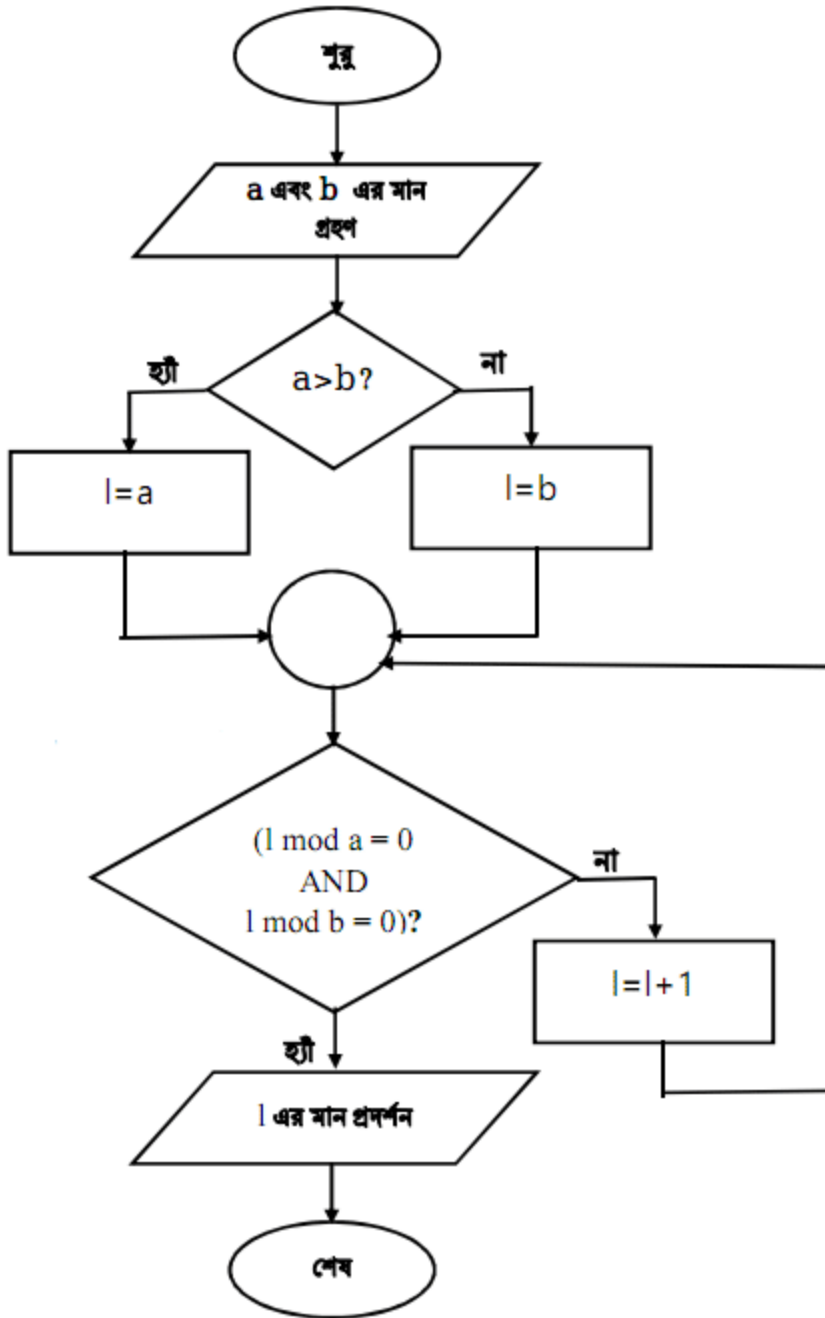
## ৫। দুটি পূর্ণ সংখ্যার ল. সা. গু. নির্ণয়ের অ্যালগোরিদম ও ফ্লোচার্ট।

ল.সা.গু শব্দের পূর্ণরূপ হল লঘিষ্ঠ সাধারণ গুণিতক। একটি সংখ্যা কোন সংখ্যা দ্বারা বিভাজ্য হলে প্রথম সংখ্যাটিকে দ্বিতীয় সংখ্যার গুণিতক বলে আর দ্বিতীয় সংখ্যাটিকে প্রথম সংখ্যার গুণনীয়ক বলে। যেমনঃ ১২ কে ৬ দ্বারা ভাগ করলে নিঃশেষে বিভাজ্য হবে সেক্ষেত্রে ১২ সংখ্যাটি ৬ এর গুণিতক আর ১২ এর গুণনীয়ক হচ্ছে ৬।

### অ্যালগোরিদমঃ

- ধাপ-১: শুরু করি।
- ধাপ-২: ইনপুট হিসেবে a এবং b চলকে দুটি সংখ্যা গ্রহণ করি।
- ধাপ-৩: যদি  $(a > b)$  হয়, তাহলে  $l = a$ , অন্যথায়  $l = b$  রাখি।
- ধাপ-৪: যদি  $((l \bmod a = 0) \text{ AND } (l \bmod b = 0))$  হয়, তাহলে ৫নং ধাপে যাই, অন্যথায় ৬নং ধাপে যাই।
- ধাপ-৫: ফলাফল হিসেবে l চলকের মান প্রদর্শন করি এবং ৭নং ধাপে যাই।
- ধাপ-৬: l চলকের মান ১ বৃদ্ধি করি এবং পুনরায় ৪নং ধাপে যাই।
- ধাপ-৭: শেষ করি।

### ফ্লোচার্টঃ



প্রোগ্রাম দেখতে ক্লিক কর

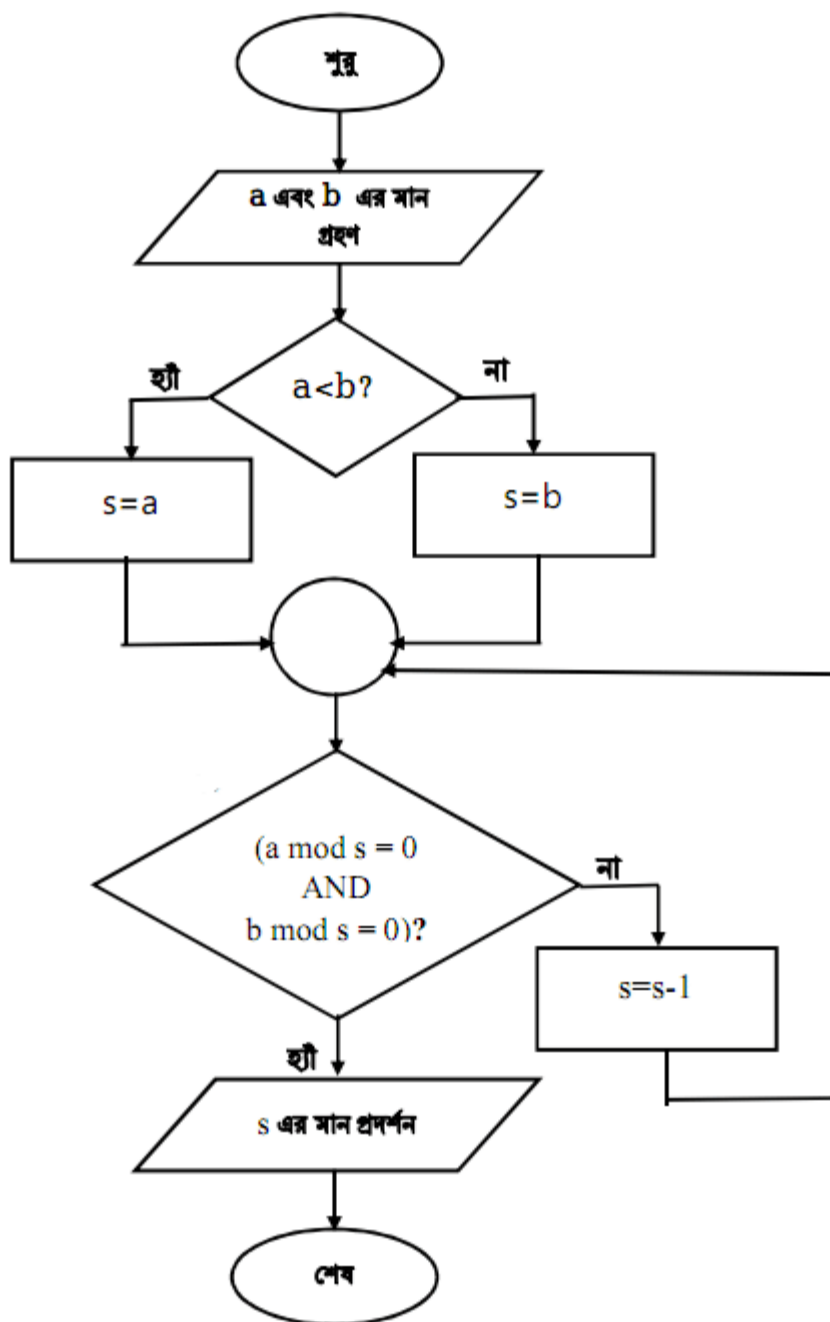
৬। দুটি পূর্ণ সংখ্যার গ. সা. গু. নির্ণয়ের অ্যালগোরিদম ও ফ্লোচার্ট।

গ.সা.গু. শব্দের পূর্ণরূপ হল গরিষ্ঠ সাধারণ গুণনীয়ক। একটি সংখ্যা কোন সংখ্যা দ্বারা বিভাজ্য হলে প্রথম সংখ্যাটিকে দ্বিতীয় সংখ্যার গুণিতক বলে আর দ্বিতীয় সংখ্যাটিকে প্রথম সংখ্যার গুণনীয়ক বলে। যেমনঃ ১২ কে ৬ দ্বারা ভাগ করলে নিঃশেষে বিভাজ্য হবে সেক্ষেত্রে ১২ সংখ্যাটি ৬ এর গুণিতক আর ১২ এর গুণনীয়ক হচ্ছে ৬।

#### অ্যালগোরিদমঃ

- ধাপ-১: শুরু করি।
- ধাপ-২: ইনপুট হিসেবে  $a$  এবং  $b$  চলকে দুটি সংখ্যা গ্রহণ করি।
- ধাপ-৩: যদি  $(a < b)$  হয়, তাহলে  $s = a$ , অন্যথায়  $s = b$  রাখি।
- ধাপ-৪: যদি  $((a \bmod s = 0) \text{ AND } (b \bmod s = 0))$  হয়, তাহলে ৫নং ধাপে যাই,
- অন্যথায় ৬নং ধাপে যাই।
- ধাপ-৫: ফলাফল হিসেবে  $s$  চলকের মান প্রদর্শন করি এবং ৭নং ধাপে যাই।
- ধাপ-৬:  $s$  চলকের মান ১ হ্রাস করি এবং পুনরায় ৪নং ধাপে যাই।
- ধাপ-৭: শেষ করি।

#### ফ্লোচার্টঃ



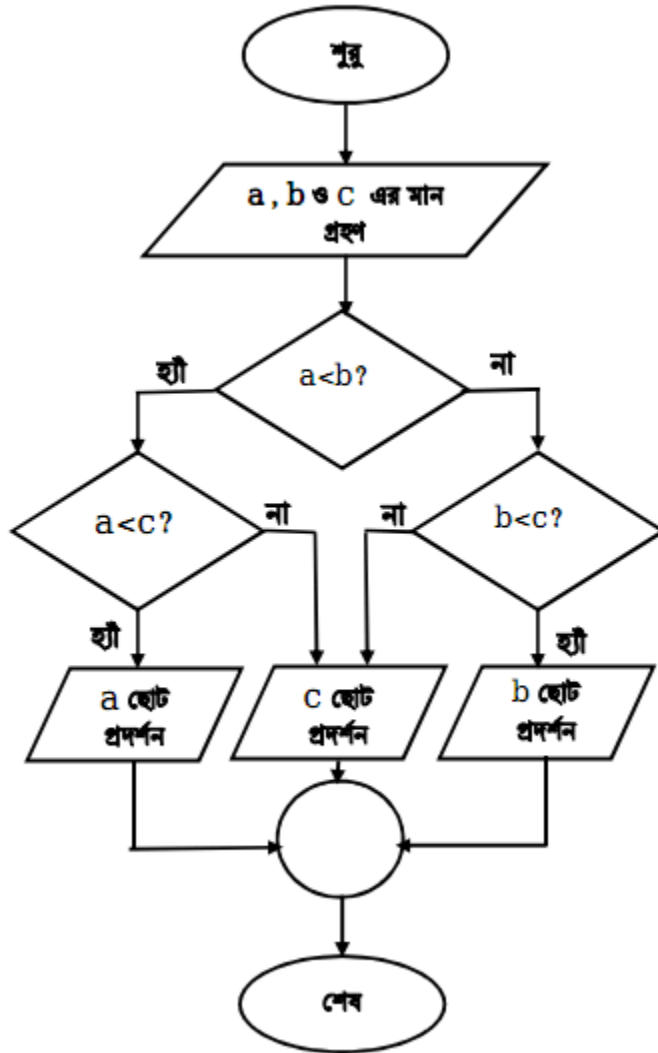
প্রোগ্রাম দেখতে ক্লিক কর

৭। তিনটি সংখ্যার মধ্যে সবচেয়ে ছোট সংখ্যা নির্ণয়ের অ্যালগোরিদম ও ফ্লোচার্ট।

অ্যালগোরিদমঃ

- ধাপ-১: শুরু করি।
- ধাপ-২: ইনপুট হিসেবে a, b ও c চলকে তিনটি সংখ্যা গ্রহণ করি।
- ধাপ-৩: যদি  $a < b$  হয়, তাহলে ৪ নং ধাপে যাই, অন্যথায় ৫নং ধাপে যাই।
- ধাপ-৪: যদি  $a < c$  হয়, তাহলে a ছোট প্রদর্শন করি এবং ৬নং ধাপে যাই, অন্যথায় c ছোট প্রদর্শন করি এবং ৬নং ধাপে যাই।
- ধাপ-৫: যদি  $b < c$  হয়, তাহলে b ছোট প্রদর্শন করি, অন্যথায় c ছোট প্রদর্শন করি।
- ধাপ-৬: শেষ করি।

ফ্লোচার্টঃ



প্রোগ্রাম দেখতে ক্লিক কর

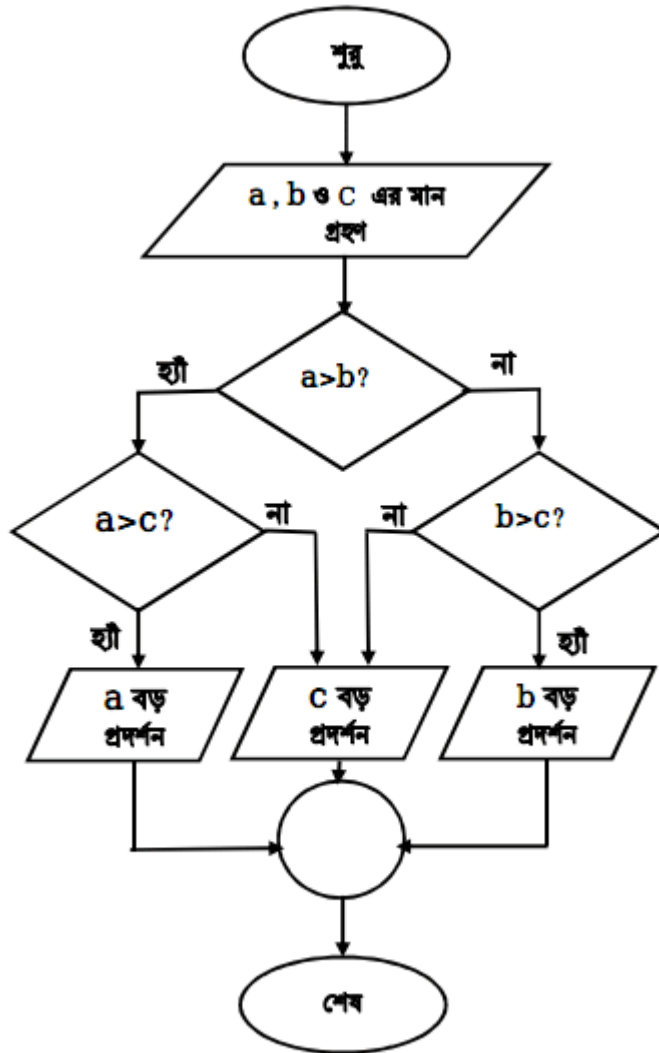


৮। তিনটি সংখ্যার মধ্যে সবচেয়ে বড় সংখ্যা নির্ণয়ের অ্যালগোরিদম ও ফ্লোচার্ট।

অ্যালগোরিদমঃ

- ধাপ-১: শুরু করি।
- ধাপ-২: ইনপুট হিসেবে  $a$ ,  $b$  ও  $c$  চলকে তিনটি সংখ্যা গ্রহণ করি।
- ধাপ-৩: যদি  $a > b$  হয়, তাহলে ৪ নং ধাপে যাই, অন্যথায় ৫নং ধাপে যাই।
- ধাপ-৪: যদি  $a > c$  হয়, তাহলে  $a$  বড় প্রদর্শন করি এবং ৬নং ধাপে যাই, অন্যথায়  $c$  বড় প্রদর্শন করি এবং ৬নং ধাপে যাই।
- ধাপ-৫: যদি  $b > c$  হয়, তাহলে  $b$  বড় প্রদর্শন করি, অন্যথায়  $c$  বড় প্রদর্শন করি।
- ধাপ-৬: শেষ করি।

ফ্লোচার্টঃ



# পঞ্চম অধ্যায় পাঠ-৭ লুপ কন্ট্রোল স্টেটমেন্ট সম্পর্কিত অ্যালগোরিদম এবং ফ্লোচার্ট সমূহ।

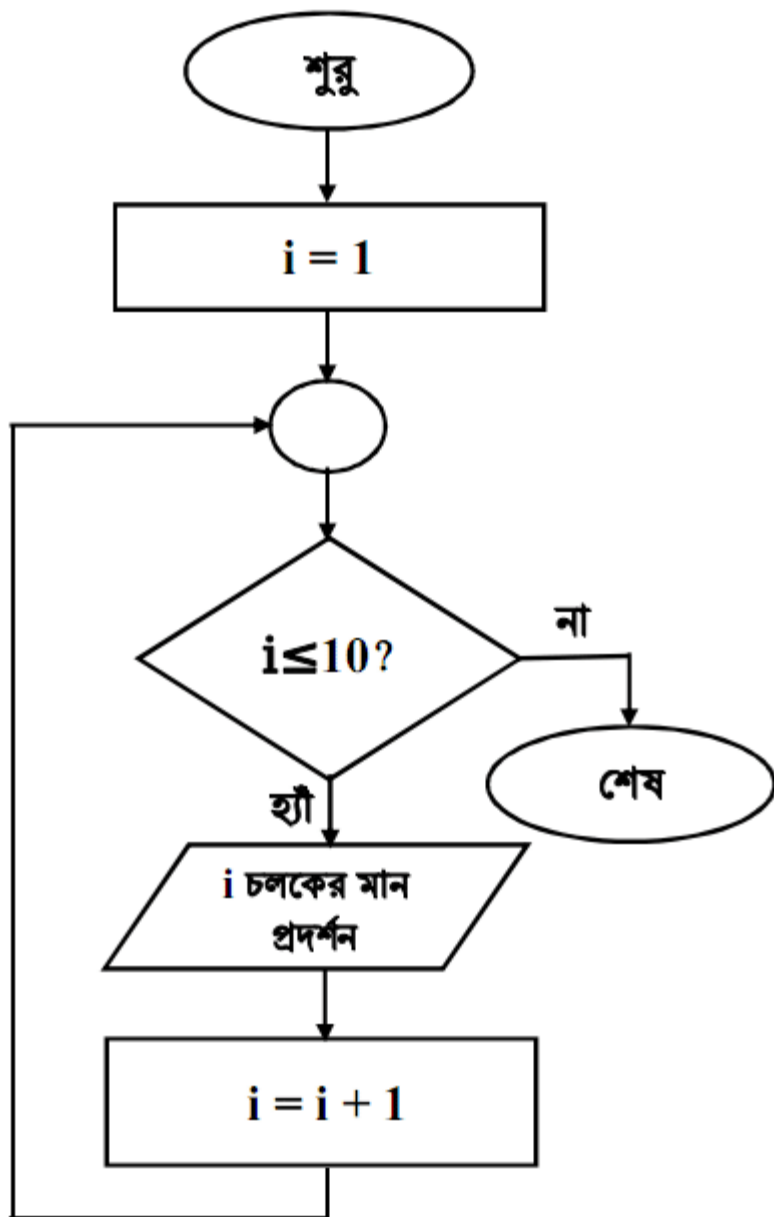
১। ১ থেকে ১০ পর্যন্ত সংখ্যা দেখানোর অ্যালগোরিদম ও ফ্লোচার্ট। অথবা

১ ২ ৩ ৪ ৫ ৬ ৭ ৮ ৯ ১০ ধারাটি তৈরির অ্যালগোরিদম ও ফ্লোচার্ট।

অ্যালগোরিদমঃ

- ধাপ-১: শুরু করি।
- ধাপ-২:  $i$  চলকের মান ১ দ্বারা সূচনা করি।
- ধাপ-৩: যদি  $i \leq 10$  হয়, তাহলে ৪নং ধাপে যাই, অন্যথায় ৬নং ধাপে যাই।
- ধাপ-৪:  $i$  চলকের মান প্রদর্শন করি।
- ধাপ-৫:  $i$  চলকের মান ১ বৃদ্ধি করি এবং পুনরায় ৩নং ধাপে যাই।
- ধাপ-৬: শেষ করি।

ফ্লোচার্টঃ



প্রোগ্রাম দেখতে ক্লিক কর

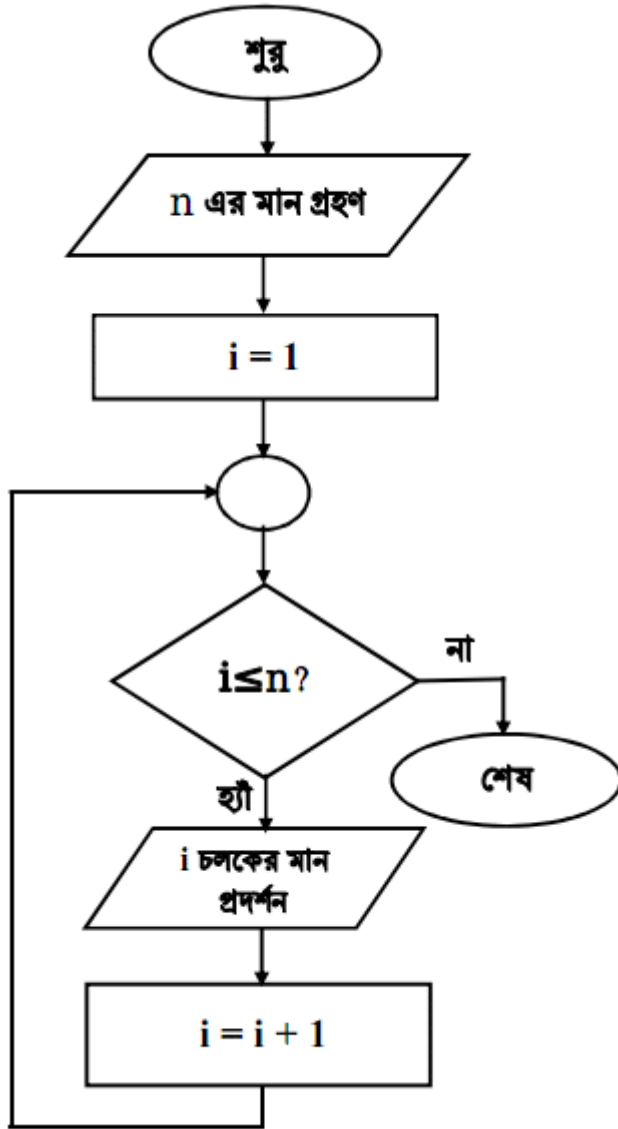
২। ১ থেকে  $n$  পর্যন্ত সংখ্যা দেখানোর অ্যালগোরিদম ও ফ্লোচার্ট। অথবা

১ ২ ৩ ৪ ৫ -----  $n$  ধারাটি তৈরির অ্যালগোরিদম ও ফ্লোচার্ট।

অ্যালগোরিদমঃ

- ধাপ-১: শুরু করি।
- ধাপ-২: ইনপুট হিসেবে  $n$  চলকের মান গ্রহণ করি।
- ধাপ-৩:  $i$  চলকের মান ১ দ্বারা সূচনা করি।
- ধাপ-৪: যদি  $i \leq n$  হয়, তাহলে ৫নং ধাপে যাই, অন্যথায় ৭নং ধাপে যাই।
- ধাপ-৫:  $i$  চলকের মান প্রদর্শন করি।
- ধাপ-৬:  $i$  চলকের মান ১ বৃদ্ধি করি এবং পুনরায় ৪নং ধাপে যাই।
- ধাপ-৭: শেষ করি।

ফ্লোচার্টঃ



প্রোগ্রাম দেখতে ক্লিক কর

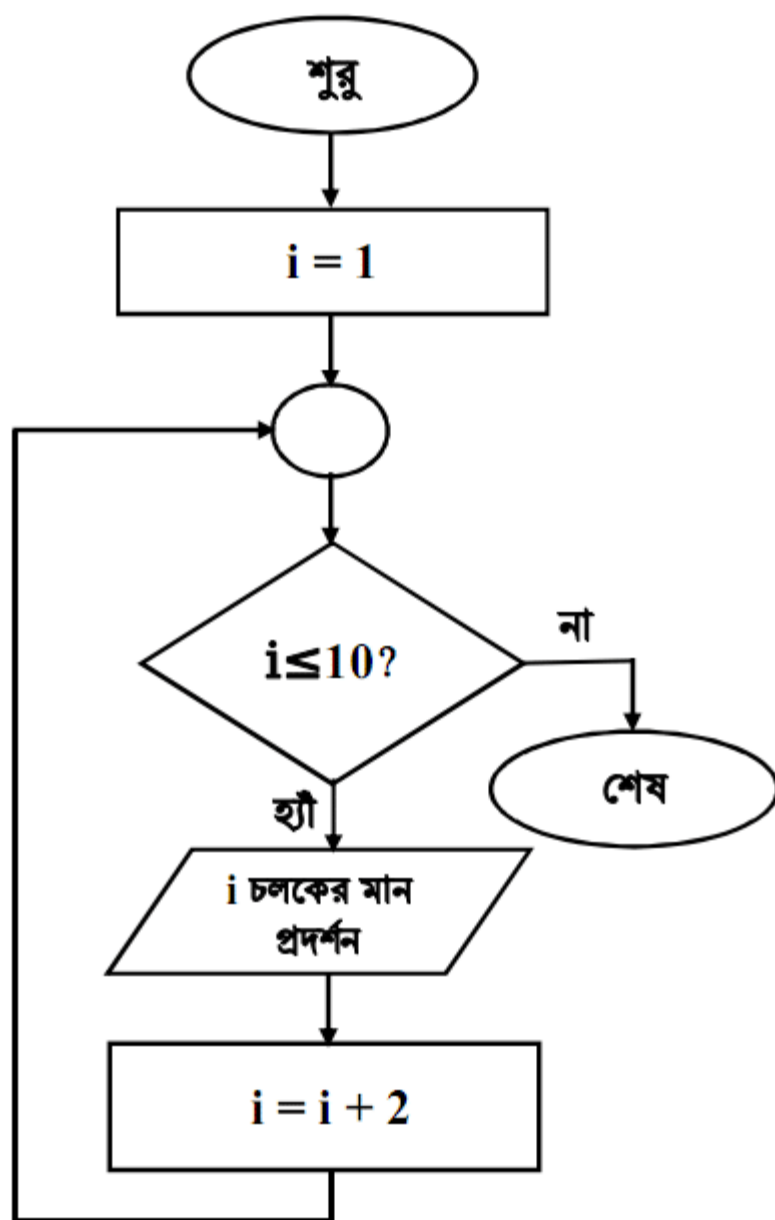
৩। ১ থেকে ১০ এর মধ্যে অবস্থিত বিজোড় সংখ্যাগুলো দেখানোর অ্যালগোরিদম ও ফ্লোচার্ট। অথবা

১ ৩ ৫ ৭ ৯ ধারাটি তৈরির অ্যালগোরিদম ও ফ্লোচার্ট।

অ্যালগোরিদমঃ

- ধাপ-১: শুরু করি।
- ধাপ-২:  $i$  চলকের মান ১ দ্বারা সূচনা করি।
- ধাপ-৩: যদি  $i \leq 10$  হয়, তাহলে ৪নং ধাপে যাই, অন্যথায় ৬নং ধাপে যাই।
- ধাপ-৪:  $i$  চলকের মান প্রদর্শন করি।
- ধাপ-৫:  $i$  চলকের মান ২ বৃদ্ধি করি এবং পুনরায় ৩নং ধাপে যাই।
- ধাপ-৬: শেষ করি।

ফ্লোচার্টঃ



প্রোগ্রাম দেখতে ক্লিক কর

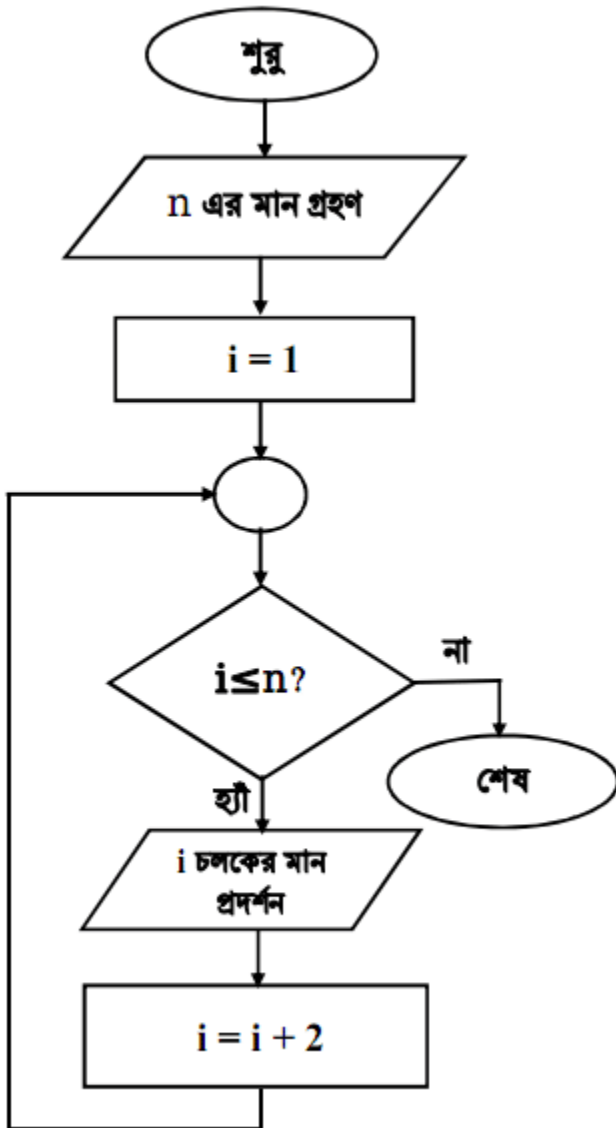
৪। ১ থেকে  $n$  এর মধ্যে অবস্থিত বিজোড় সংখ্যাগুলো দেখানোর অ্যালগোরিদম ও ফ্লোচার্ট। অথবা

১ ৩ ৫ ৭ ...  $n$  ধারাটি তৈরির অ্যালগোরিদম ও ফ্লোচার্ট।

অ্যালগোরিদমঃ

- ধাপ-১: শুরু করি।
- ধাপ-২: ইনপুট হিসেবে  $n$  চলকের মান গ্রহণ করি।
- ধাপ-৩:  $i$  চলকের মান ১ দ্বারা সূচনা করি।
- ধাপ-৪: যদি  $i \leq n$  হয়, তাহলে ৫নং ধাপে যাই, অন্যথায় ৭নং ধাপে যাই।
- ধাপ-৫:  $i$  চলকের মান প্রদর্শন করি।
- ধাপ-৬:  $i$  চলকের মান ২ বৃদ্ধি করি এবং পুনরায় ৪নং ধাপে যাই।
- ধাপ-৭: শেষ করি।

ফ্লোচার্টঃ



প্রোগ্রাম দেখতে ক্লিক কর

৫। ১ থেকে ১০ এর মধ্যে অবস্থিত জোড় সংখ্যাগুলো দেখানোর অ্যালগোরিদম ও ফ্লোচার্ট। অথবা

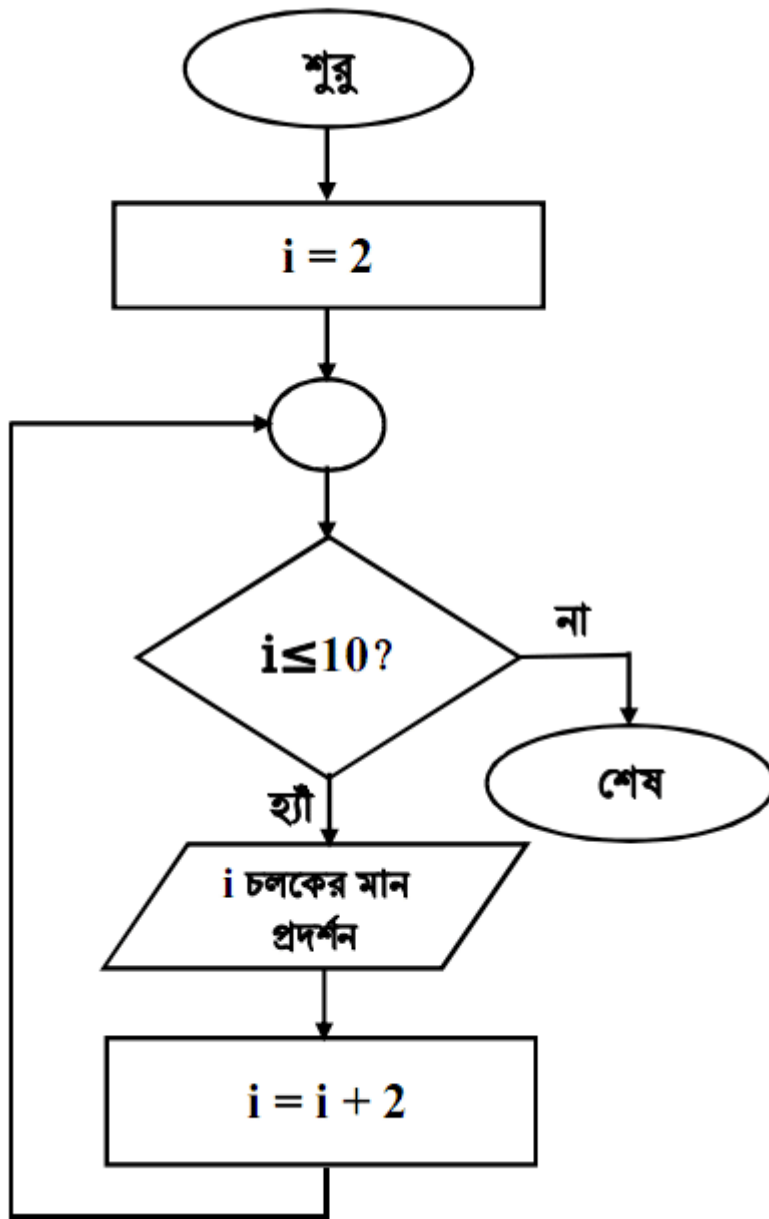
২ ৪ ৬ ৮ ১০ ধারাটি তৈরির অ্যালগোরিদম ও ফ্লোচার্ট।

অ্যালগোরিদমঃ

- ধাপ-১: শুরু করি।
- ধাপ-২:  $i$  চলকের মান ২ দ্বারা সূচনা করি।
- ধাপ-৩: যদি  $i \leq 10$  হয়, তাহলে ৪নং ধাপে যাই, অন্যথায় ৬নং ধাপে যাই।
- ধাপ-৪:  $i$  চলকের মান প্রদর্শন করি।
- ধাপ-৫:  $i$  চলকের মান ২ বৃদ্ধি করি এবং পুনরায় ৩নং ধাপে যাই।
- ধাপ-৬: শেষ করি।

ফ্লোচার্টঃ





প্রোগ্রাম দেখতে ক্লিক কর

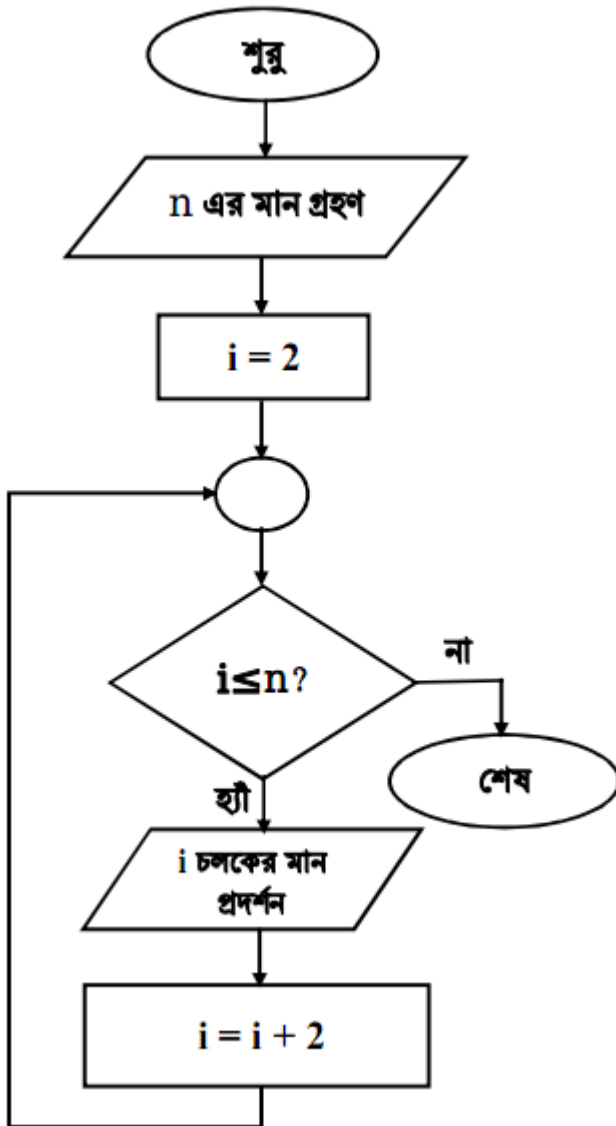
৬। ১ থেকে  $n$  এর মধ্যে অবস্থিত জোড় সংখ্যাগুলো দেখানোর অ্যালগোরিদম ও ফ্লোচার্ট। অথবা

২ ৪ ৬ ---  $n$  ধারাটি তৈরির অ্যালগোরিদম ও ফ্লোচার্ট।

অ্যালগোরিদমঃ

- ধাপ-১: শুরু করি।
- ধাপ-২: ইনপুট হিসেবে  $n$  চলকের মান গ্রহণ করি।
- ধাপ-৩:  $i$  চলকের মান ২ দ্বারা সূচনা করি।
- ধাপ-৪: যদি  $i \leq n$  হয়, তাহলে ৫নং ধাপে যাই, অন্যথায় ৭নং ধাপে যাই।
- ধাপ-৫:  $i$  চলকের মান প্রদর্শন করি।
- ধাপ-৬:  $i$  চলকের মান ২ বৃদ্ধি করি এবং পুনরায় ৪নং ধাপে যাই।
- ধাপ-৭: শেষ করি।

ফ্লোচার্টঃ



প্রোগ্রাম দেখতে ক্লিক কর

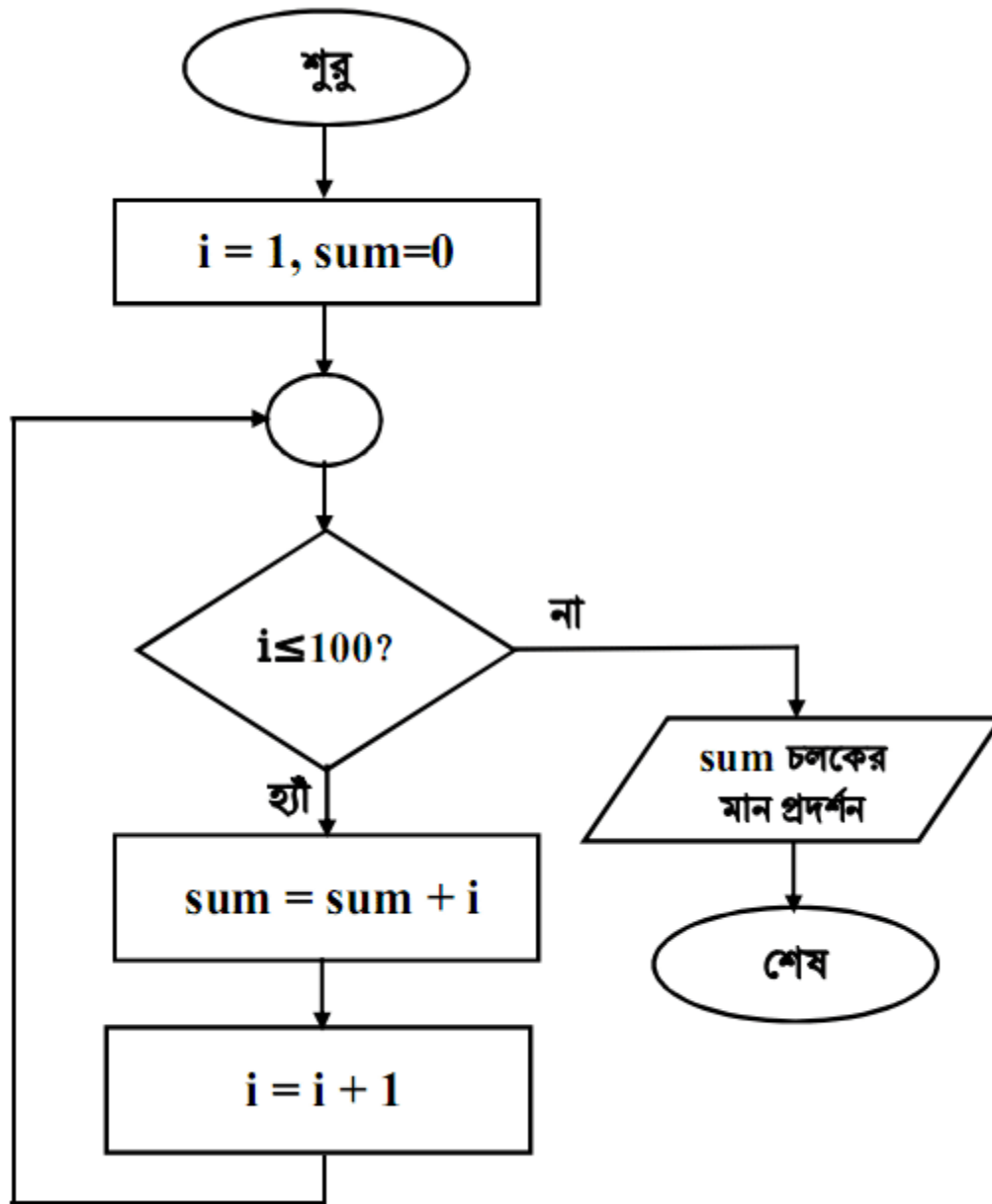
৭। ১ থেকে ১০০ পর্যন্ত সংখ্যা গুলোর যোগফল নির্ণয়ের অ্যালগোরিদম ও ফ্লোচার্ট। অথবা

$1+2+3+ \dots +100$  ধারার যোগফল নির্ণয়ের অ্যালগোরিদম ও ফ্লোচার্ট।

অ্যালগোরিদমঃ

- ধাপ-১: শুরু করি।
- ধাপ-২:  $i$  চলকের মান ১ দ্বারা এবং  $sum$  চলকের মান ০ দ্বারা সূচনা করি।
- ধাপ-৩: যদি  $i \leq 100$  হয়, তাহলে ৪নং ধাপে যাই, অন্যথায় ৬নং ধাপে যাই।
- ধাপ-৪:  $sum = sum + i$  নির্ণয় করি।
- ধাপ-৫:  $i$  চলকের মান ১ বৃদ্ধি করি এবং পুনরায় ৩নং ধাপে যাই।
- ধাপ-৬:  $sum$  চলকের মান প্রদর্শন করি।
- ধাপ-৭: শেষ করি।

ফ্লোচার্টঃ



প্রোগ্রাম দেখতে ক্লিক কর

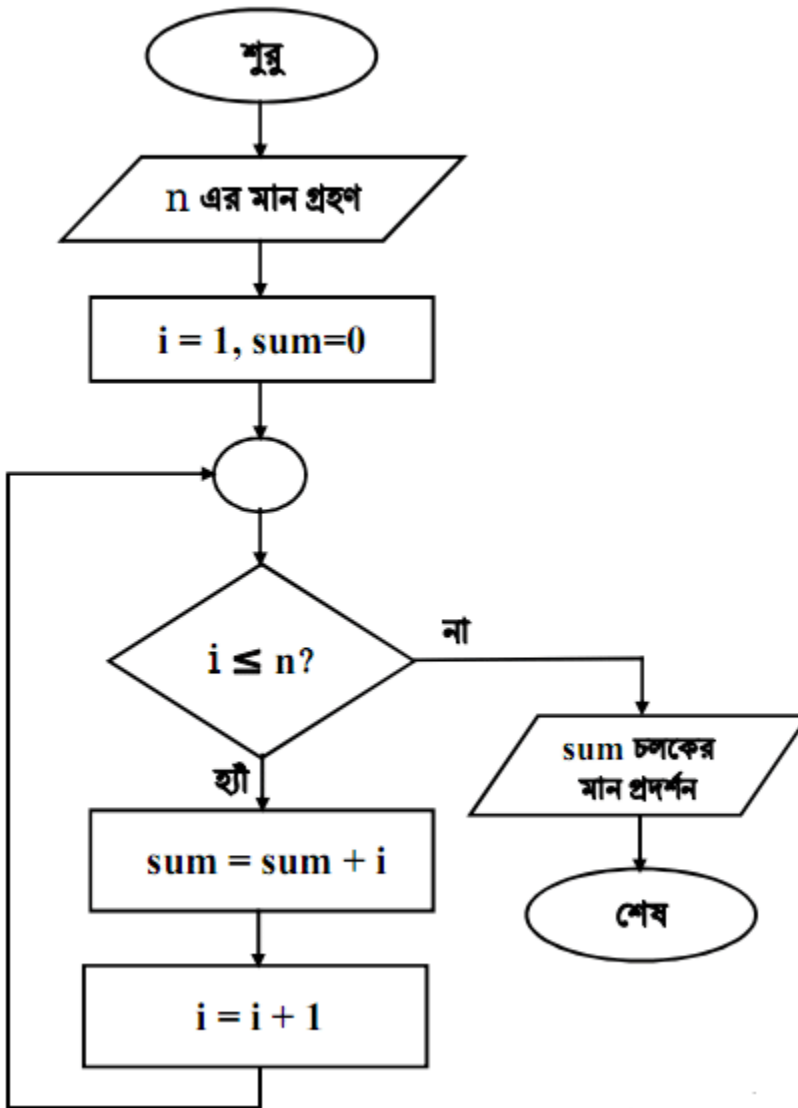
৮। ১ থেকে n পর্যন্ত সংখ্যা গুলোর যোগফল নির্ণয়ের অ্যালগোরিদম ও ফ্লোচার্ট। অথবা

১+২+৩+ --- +n ধারার যোগফল নির্ণয়ের অ্যালগোরিদম ও ফ্লোচার্ট।

অ্যালগোরিদমঃ

- ধাপ-১: শুরু করি।
- ধাপ-২: ইনপুট হিসেবে n চলকের মান গ্রহণ করি।
- ধাপ-৩: i চলকের মান 1 দ্বারা এবং sum চলকের মান 0 দ্বারা সূচনা করি।
- ধাপ-৪: যদি  $i \leq n$  হয়, তাহলে ৫নং ধাপে যাই, অন্যথায় ৭নং ধাপে যাই।
- ধাপ-৫:  $sum = sum + i$  নির্ণয় করি।
- ধাপ-৬: i চলকের মান 1 বৃদ্ধি করি এবং পুনরায় ৪নং ধাপে যাই।
- ধাপ-৭: sum চলকের মান প্রদর্শন করি।
- ধাপ-৮: শেষ করি।

ফ্লোচার্টঃ



প্রোগ্রাম দেখতে ক্লিক কর

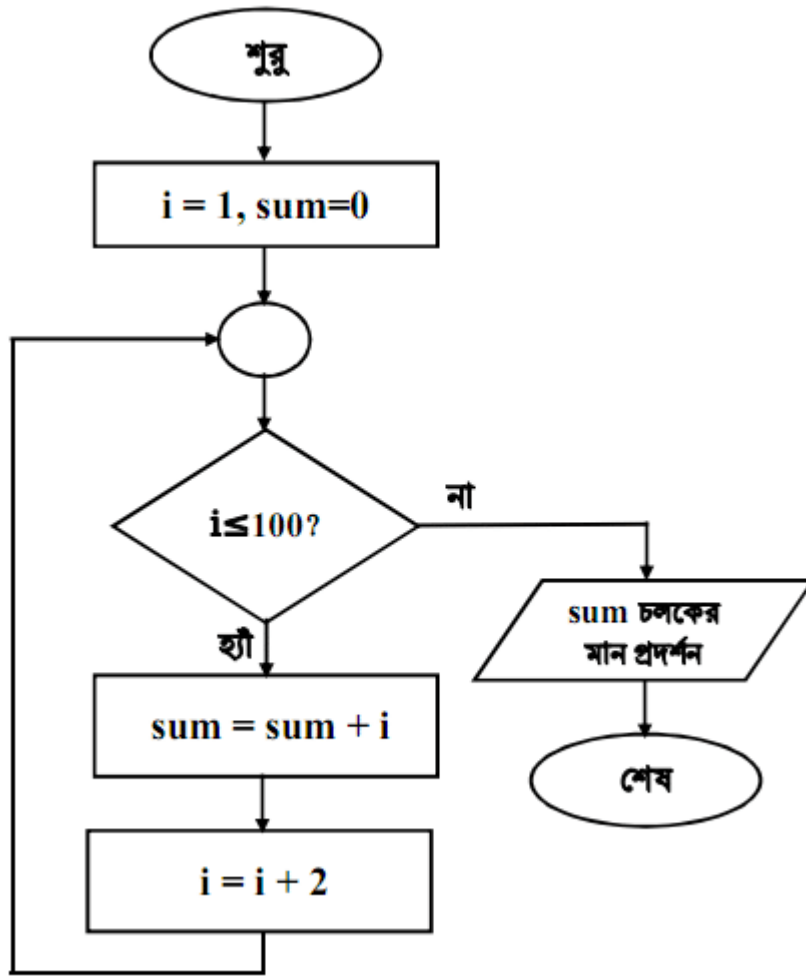
৯। ১ থেকে ১০০ এর মধ্যে অবস্থিত বিজোড় সংখ্যা গুলোর যোগফল নির্ণয়ের অ্যালগোরিদম ও ফ্লোচার্ট। অথবা

১+২+৫+ - - - - -+১০০ ধারার যোগফল নির্ণয়ের অ্যালগোরিদম ও ফ্লোচার্ট।

অ্যালগোরিদমঃ

- ধাপ-১: শুরু করি।
- ধাপ-২:  $i$  চলকের মান ১ দ্বারা এবং  $sum$  চলকের মান ০ দ্বারা সূচনা করি।
- ধাপ-৩: যদি  $i \leq 100$  হয়, তাহলে ৪নং ধাপে যাই, অন্যথায় ৬নং ধাপে যাই।
- ধাপ-৪:  $sum = sum + i$  নির্ণয় করি।
- ধাপ-৫:  $i$  চলকের মান ২ বৃদ্ধি করি এবং পুনরায় ৩নং ধাপে যাই।
- ধাপ-৬:  $sum$  চলকের মান প্রদর্শন করি।
- ধাপ-৭: শেষ করি।

ফ্লোচার্টঃ



প্রোগ্রাম দেখতে ক্লিক কর

১০। ১ থেকে ১০০ এর মধ্যে অবস্থিত জোড় সংখ্যা গুলোর যোগফল নির্ণয়ের অ্যালগোরিদম ও ফ্লোচার্ট। অথবা

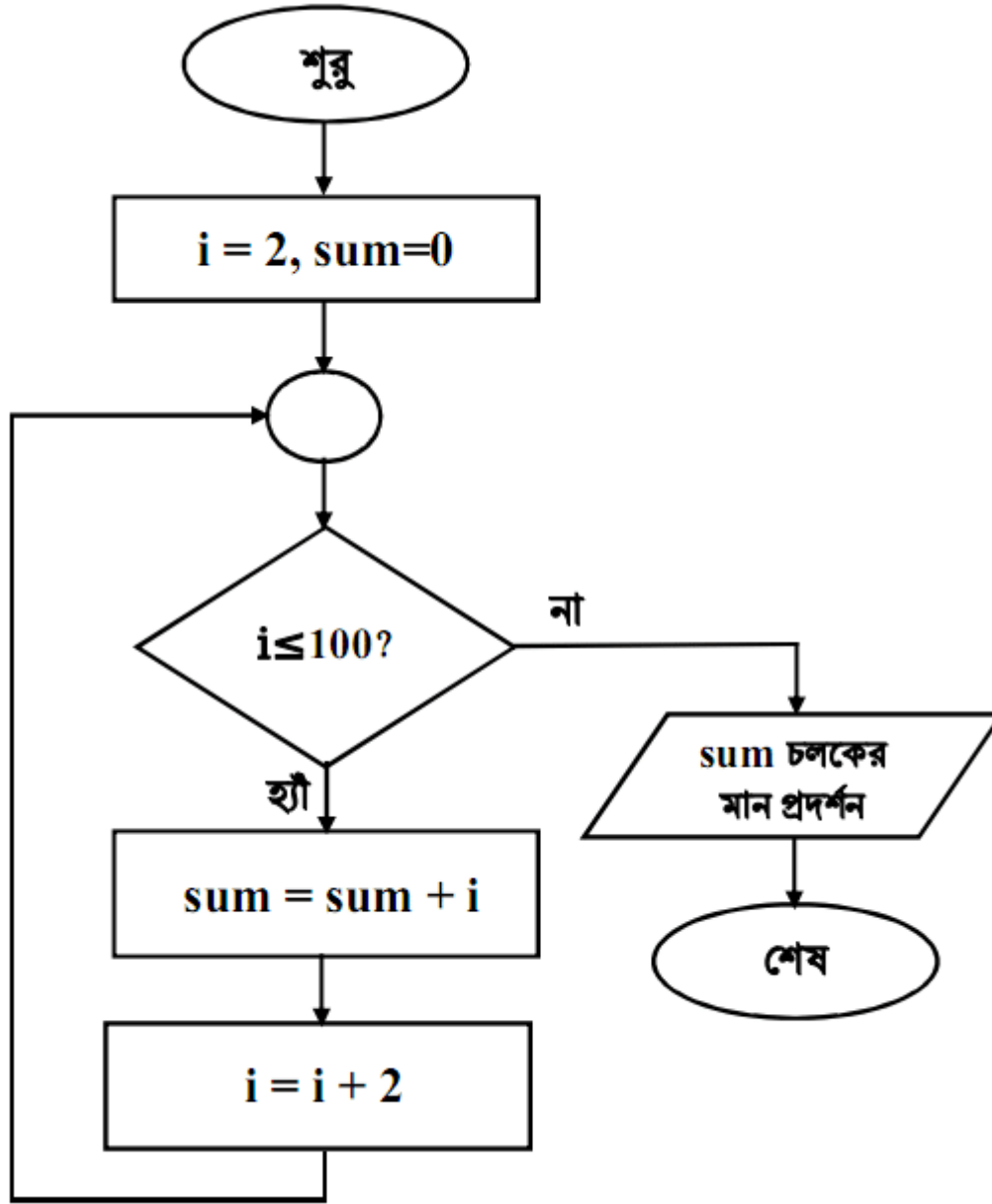
২+৪+৬+-----+১০০ ধারার যোগফল নির্ণয়ের অ্যালগোরিদম ও ফ্লোচার্ট।

অ্যালগোরিদমঃ

- ধাপ-১: শুরু করি।
- ধাপ-২: i চলকের মান ১ দ্বারা এবং sum চলকের মান ০ দ্বারা সূচনা করি।
- ধাপ-৩: যদি  $i \leq 100$  হয়, তাহলে ৪নং ধাপে যাই, অন্যথায় ৬নং ধাপে যাই।
- ধাপ-৪:  $sum = sum + i$  নির্ণয় করি।
- ধাপ-৫: i চলকের মান ১ বৃদ্ধি করি এবং পুনরায় ৩নং ধাপে যাই।
- ধাপ-৬: sum চলকের মান প্রদর্শন করি।

- ধাপ-৭: শেষ করি।

ফ্লোচার্টঃ



## পঞ্চম অধ্যায় পাঠ-৮ প্রোগ্রাম ডিজাইন মডেল।

প্রোগ্রাম ডিজাইন মডেলঃ সহজ উপায়ে কার্যকরী প্রোগ্রাম তৈরির জন্য যে বিশেষ নীতিমালা বা পদ্ধতি অনুসরণ করা হয় তাকে প্রোগ্রাম ডিজাইন মডেল বলে। কয়েকটি জনপ্রিয় প্রোগ্রাম ডিজাইন মডেল-



- ১। স্ট্রাকচার্ড বা প্রসিডিউর প্রোগ্রামিং মডেল
- ২। অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং(OOP) মডেল
- ৩। ভিজুয়াল প্রোগ্রামিং মডেল
- ৪। ইভেন্ট ড্রাইভেন প্রোগ্রামিং মডেল

**স্ট্রাকচার্ড প্রোগ্রামিংঃ** ১৯৬৬ সালে স্ট্রাকচার্ড প্রোগ্রামিং এর প্রথম ধারণা দেন Corrado Bohm এবং Guiseppe Jacopini। এই দুই গণিতবিদ ব্যাখ্যা করেন যে, যেকোন প্রোগ্রাম শুধুমাত্র তিনটি স্ট্রাকচার যেমন- decisions, sequences, এবং loops এর সাহায্যে লেখা যায়। পরবর্তিতে ১৯৭০ সালে Edsger W.Dijkstra ব্যপকভাবে ব্যবহৃত স্ট্রাকচার্ড প্রোগ্রামিং পদ্ধতি উন্নয়ন করেন, যেখানে একটি সমস্যাকে বিভিন্ন ছোট ছোট মডিউল বা অংশে ভাগ করে একটি বড় সমস্যার সমাধান করা হয়। প্রতিটি মডিউলকে ফাংশন বলা হয়। এক্ষেত্রে মডিউল বা ফাংশন গুলোর মধ্যে একটি প্রধান মডিউল বা মেইন ফাংশন বিবেচনা করা হয় যা অন্য সকল মডিউলকে কল করতে পারে, আবার এক মডিউল অন্য মডিউলকেও কল করতে পারে। কিন্তু প্রধান মডিউলকে অন্য মডিউলগুলো কল করতে পারে না। এই মডেলে প্রোগ্রামের নিয়ন্ত্রণ উপর থেকে নিচের দিকে পরিচালিত হয় অর্থাৎ টপ-ডাউন পদ্ধতি অনুসরণ করে। স্ট্রাকচার্ড প্রোগ্রামিং হল ইম্পট্রাকশন কেন্দ্রিক প্রোগ্রামিং পদ্ধতি। অর্থাৎ এই পদ্ধতিতে প্রোগ্রামের ডেটা গুলো ইম্পট্রাকশন দ্বারা নিয়ন্ত্রিত হয়।

উদাহরণঃ সি, কোবল, প্যাসকেল, ফরট্রান, কিউবেসিক ইত্যাদি প্রোগ্রামিং ভাষায় স্ট্রাকচার্ড প্রোগ্রামিং ডিজাইন অনুসরণ করে প্রোগ্রাম লেখা যায় এজন্য এই প্রোগ্রামিং ভাষা গুলোকে স্ট্রাকচার্ড প্রোগ্রামিং ভাষা বলা হয়।

#### স্ট্রাকচার্ড প্রোগ্রামিং এর সুবিধাঃ

- ১। প্রোগ্রাম তৈরি করা সহজ। কারণ কম শ্রম এবং সময় প্রয়োজন হয়।
- ২। প্রোগ্রাম পড়া এবং বুঝা সহজ।
- ৩। মেইন্টেইন সহজ।
- ৪। যেহেতু প্রোগ্রামকে বিভিন্ন মডিউলে ভাগ করা হয়, তাই প্রয়োজনে একটি মডিউলকে বার বার ব্যবহার করা যায়।
- ৫। বিভিন্ন মডিউলে ভাগ করায় ডিবাগিং বা প্রোগ্রামের ভুল সংশোধন করা সহজ।

**অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিংঃ** অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং হল এমন একটি প্রোগ্রামিং পদ্ধতি যা স্ট্রাকচার্ড প্রোগ্রামিং এর সুবিধার পাশাপাশি অতিরিক্ত বিশেষ কিছু সুবিধা যেমন- এনক্যাপ্সুলেশন, পলিমরফিজম ও ইনহেরিটেন্স প্রভৃতি ফিচার ব্যবহার করে প্রোগ্রাম লেখার সুবিধা প্রদান করে। অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং হল ডেটা কেন্দ্রিক প্রোগ্রামিং পদ্ধতি। অর্থাৎ এই পদ্ধতিতে প্রোগ্রামের ইম্পট্রাকশন গুলো ডেটা দ্বারা নিয়ন্ত্রিত হয়। একটি প্রোগ্রামিং ভাষাকে পরিপূর্ণ অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং ভাষা বলা যাবে তখনই, যখন প্রোগ্রামিং ভাষাটি ক্লাস, অবজেক্ট, এনক্যাপ্সুলেশন, পলিমরফিজম ও ইনহেরিটেন্স প্রভৃতি ফিচারগুলো সাপোর্ট করবে। উদাহরণঃ জাভা, পাইথন, সি++ সি# ইত্যাদি হল অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং ভাষা। অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং ভাষার বৈশিষ্ট্যসমূহঃ

**অবজেক্টঃ** অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং ভাষায় যেকোন ব্যক্তি বা বস্তুকে অবজেক্ট বলা হয়। যেমন – একটি গাড়ি কে অবজেক্ট বলা যায়। প্রতিটি অবজেক্ট এর কিছু বৈশিষ্ট্য(attribute) ও আচরণ(behavior) থাকে। যেমন একটি গাড়ির কালার, মডেল ইত্যাদি হল বৈশিষ্ট্য, আবার গাড়িটি সামনে চলতে পারে এবং পিছনে চলতে পারে এগুলো হল আচরণ।

**ক্লাসঃ** অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং ভাষায় ক্লাস হল ভেরিয়েবল ও মেথডের সমন্বয়ে একটি টেমপ্লেট বা ব্ল-প্রিন্ট যা কোন অবজেক্ট এর বৈশিষ্ট্য(attribute) ও আচরণ(behavior) উপস্থাপনের জন্য তৈরি করা হয়।

**এনক্যাপ্সুলেশনঃ** অবজেক্ট এর বৈশিষ্ট্য(attribute) ও আচরণ(behavior) কে একত্র করে ক্লাস তৈরি করাকে বলা হয় এনক্যাপ্সুলেশন।

**পলিমরফিজমঃ** পলিমরফিজম মানে হল বহুরূপ। একাদিক কোড মডিউলের নাম এক হলেও ভিন্ন ভিন্ন রূপ থাকতে পারে, এক্ষেত্রে কোন মডিউলটি কাজ করবে তা নির্ভর করে ডেটা পাঠানোর উপর।

**ইনহেরিটেন্সঃ** অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং ভাষায় ইনহেরিটেন্স এমন একটি ফিচার, যার কারণে একটি ক্লাসের বৈশিষ্ট্য অপর একটি ক্লাস ব্যবহার করতে পারে, একে বলা হয় ইনহেরিট করা। যে ক্লাস কে ইনহেরিট করা হয় তাকে বলে বেজ ক্লাস এবং যে ক্লাস অন্য ক্লাসকে ইনহেরিট করে তাকে বলে ডিরাইভড ক্লাস।

**ভিজুয়াল প্রোগ্রামিংঃ** ভিজুয়াল প্রোগ্রামিং হল এমন একটি প্রোগ্রামিং পদ্ধতি যেখানে বিভিন্ন চিত্রভিত্তিক নির্দেশ বা কমান্ড ব্যবহার করে প্রোগ্রাম রচনা করা হয়। স্ট্রাকচার্ড বা অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং এর উপর ভিত্তি করেই ভিজুয়াল প্রোগ্রামিং মডেল তৈরি। মাইক্রোসফট কোম্পানির তৈরিকৃত ভিজুয়াল বেসিক হল প্রথম ভিজুয়াল প্রোগ্রামিং মডেল।

**ইভেন্ট ড্রাইভেন প্রোগ্রামিংঃ** সকল ভিজুয়াল প্রোগ্রাম হচ্ছে ইভেন্ট ড্রাইভেন প্রোগ্রাম। এই মডেলে কী-বোর্ডের কোন কী চাপ দেওয়া, কোন বিশেষ কন্ট্রলের উপর মাউস ক্লিক করা ইত্যাদি কাজ গুলো হচ্ছে এক একটি ইভেন্ট। প্রতিটি ইভেন্টের জন্য পৃথক পৃথক কোড মডিউল থাকে। ব্যবহারকারী যখন কোন ইভেন্ট একটিভ করেন তখন ঐ ইভেন্টের জন্য নির্ধারিত কোড মডিউলটি নির্বাহ হয়।

**পঞ্চম অধ্যায় পাঠ-৯ ‘সি’ প্রোগ্রামিং ভাষার প্রাথমিক ধারণা।**

**‘সি’ প্রোগ্রামিং ভাষার প্রাথমিক ধারণাঃ** ‘সি’ প্রোগ্রামিং ভাষা একটি স্ট্রাকচার্ড বা প্রোসিডিউর প্রোগ্রামিং ভাষা যা “ডেনিশ রিচি” ডেভলোপ করেন। এই ভাষাটি বেল ল্যাবরেটরিতে UNIX অপারেটিং সিস্টেম তৈরি করার সময় তৈরি করেন। মিদ লেভেল ভাষা হিসেবে ‘সি’ অত্যন্ত জনপ্রিয়। ‘সি’ ভাষাটি ১৯৭২ সালে DEC PDP-11 নামক কম্পিউটারে সর্বপ্রথম বাস্তবায়ন করা হয়। ‘সি’ নামটা এসেছে মার্টিন রিচার্ডস (Martins Richards) এর উদ্ভাবিত বিসিপিএল (BCPL-Basic Combined Programming Language) ভাষা থেকে। BCPL সংক্ষেপে B নামে পরিচিত ছিল। পরে B এর উন্নয়নের ফলে C এর বিকাশ ঘটে।

‘সি’ ভাষাকে সকল আধুনিক প্রোগ্রামিং ভাষার মাতৃ-ভাষা (mother language) বলা হয়। কারণ অধিকাংশ প্রোগ্রামিং ভাষা যেমন- C++, Java, C#, ইত্যাদি ‘সি’ ভাষার সিনট্যাক্স অনুসরণ করে। ‘সি’ ভাষা অ্যারে, স্ট্রিং, ফাংশন, ফাইল ইত্যাদির ধারণা দেয় যা C++, Java, C#, ইত্যাদি ভাষায় ব্যবহৃত হয়।

**‘সি’ প্রোগ্রামিং ভাষা একটি স্ট্রাকচার্ড বা প্রোসিডিউর অরিয়েন্টেড প্রোগ্রামিং ভাষা**

‘সি’ কে স্ট্রাকচার্ড প্রোগ্রামিং ভাষা বলা হয়, কারণ ‘সি’ তে একটি প্রোগ্রামকে কতগুলো ছোট ছোট অংশে বিভক্ত করে প্রতিটি অংশের জন্য আলাদাভাবে ভেরিয়েবল, স্ট্রাকচার, ফাংশন ইত্যাদি বর্ণনা করা যায় এবং প্রয়োজনে if, while, for, goto ইত্যাদি কন্ট্রোল স্টেটমেন্টের মাধ্যমে বিভিন্ন অংশের মধ্যে সমন্বয় সাধন করা যায়, কিংবা কোন ফাংশন বা স্ট্রাকচার পুনঃব্যবহার করা যায়। তাই ‘সি’ প্রোগ্রামিং ভাষাকে একটি স্ট্রাকচার্ড প্রোগ্রামিং ভাষা বলা হয়। আনস্ট্রাকচার্ড ভাষায় (যেমন- বেসিক) এভাবে মূল সমস্যাকে একাধিক অংশে বিভক্ত করে প্রতিটি আলাদা অংশের জন্য আলাদাভাবে ফাংশন বর্ণনা করা যায় না।

‘সি’ কে প্রোসিডিউর অরিয়েন্টেড প্রোগ্রামিং ভাষাও বলা হয়, কারণ একটি প্রোগ্রামকে কতগুলো ছোট ছোট অংশে বিভক্ত করে প্রতিটি অংশের জন্য আলাদাভাবে ভেরিয়েবল, স্ট্রাকচার, ফাংশন ইত্যাদি বর্ণনা করা যায় এবং এই ছোট ছোট অংশগুলো পর্যায়ক্রমে নির্বাহের মাধ্যমে একটি সমস্যার সমাধান করে।

### ‘সি’ একটি মধ্যস্তরের প্রোগ্রামিং ভাষা

‘সি’ প্রোগ্রামিং ভাষায় নিম্নস্তরের ভাষার সুবিধা যেমন- বিট পর্যায়ে প্রোগ্রামিং বা সিস্টেম সফটওয়্যার এর মাধ্যমে হার্ডওয়্যার নিয়ন্ত্রণ এবং উচ্চস্তরের ভাষার সুবিধা যেমন- অ্যাপ্লিকেশন সফটওয়্যার তৈরি করা যায়। অর্থাৎ উচ্চস্তরের ভাষার সুবিধা পাওয়া যায় আবার নিম্নস্তরের ভাষার সুবিধাও পাওয়া যায়। তাই এই প্রোগ্রামিং ভাষাকে মধ্যস্তরের প্রোগ্রামিং ভাষা বলা হয়।

### ‘সি’ প্রোগ্রামিং ভাষা একটি general purpose প্রোগ্রামিং ভাষা

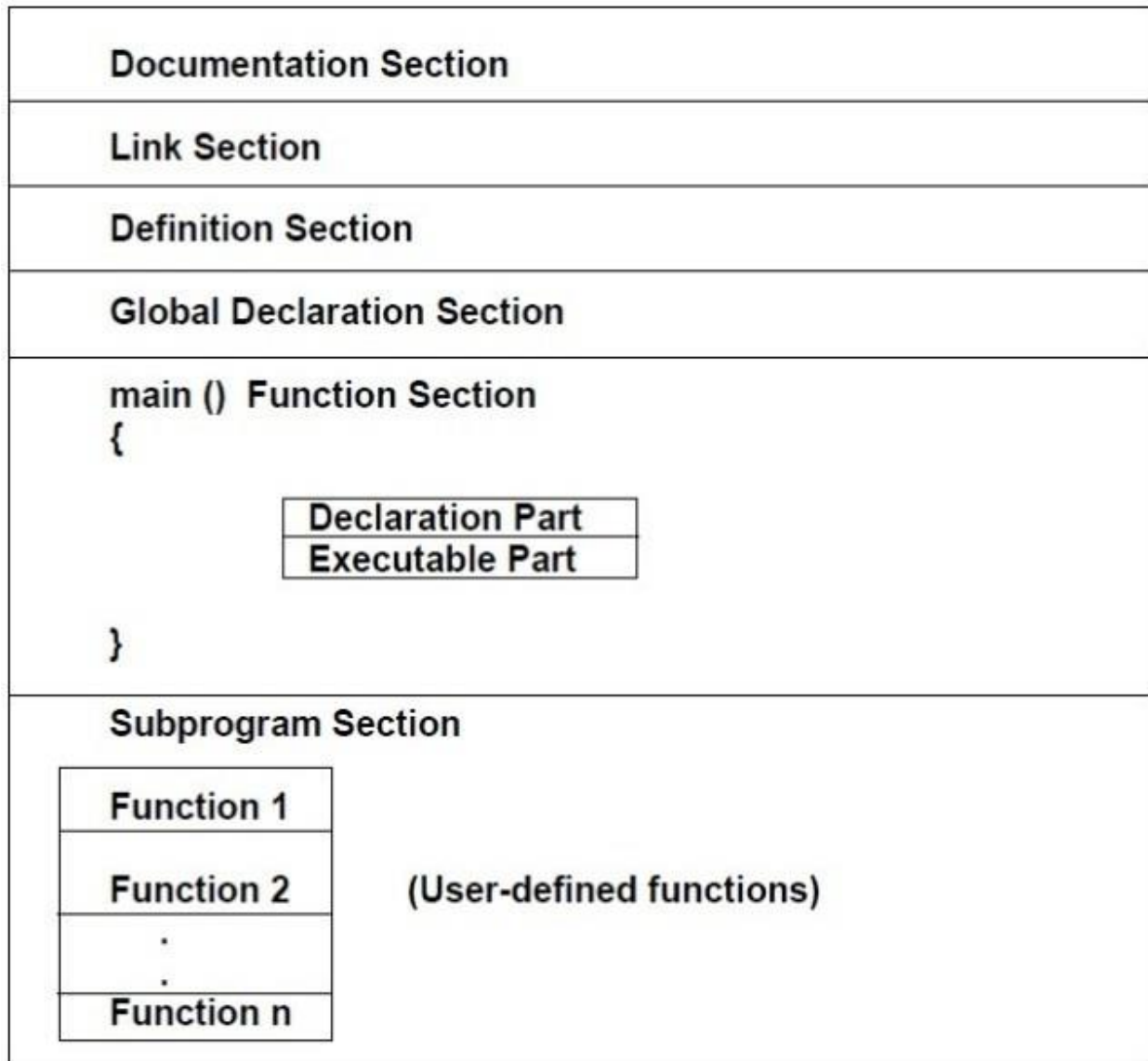
‘সি’ প্রোগ্রামিং ভাষাটি সব ধরনের কাজের জন্য ব্যবহৃত হয়। একজন প্রোগ্রামারের যেসব সুবিধা দরকার, যেমন- বিভিন্ন ডেটা ব্যবহারের ব্যাপক স্বাধীনতা, স্বল্প সংখ্যক কী-ওয়ার্ড, দ্রুত ও দক্ষতার সাথে প্রোগ্রাম চালানো এবং একই সাথে উচ্চ ও নিম্নস্তরের ভাষা সমন্বয় করা ইত্যাদি সব রকম সুবিধাই ‘সি’ প্রোগ্রামিং ভাষাতে আছে। তাই যেকোনো ধরনের প্রোগ্রাম লিখতে ‘সি’ প্রোগ্রামিং ভাষা ব্যবহার করা যায়। এই জন্য ‘সি’ ভাষাকে একটি General Purpose প্রোগ্রামিং ভাষা বলা হয়।

### ‘সি’ প্রোগ্রামিং ভাষার বৈশিষ্ট্যঃ

- ১। সকল ‘সি’ প্রোগ্রামের কাজ main() ফাংশন থেকে শুরু হয় এবং এটি প্রতিটি প্রোগ্রামের জন্য অত্যাবশ্যকীয়।
- ২। ‘সি’ প্রোগ্রামিং ভাষা একটি case sensitive ভাষা; অর্থাৎ uppercase letter এবং lowercase letter ভিন্ন অর্থ বহন করে।
- ৩। ‘সি’ প্রোগ্রামের প্রতিটি স্টেটমেন্ট এর শেষে সেমিকোলন( ; ) দিতে হয়।
- ৪। ‘সি’ প্রোগ্রামিং ভাষাকে মধ্যস্তরের প্রোগ্রামিং ভাষা বলা হয়।
- ৫। ‘সি’ প্রোগ্রামিং ভাষাকে General purpose language ও বলা হয়।
- ৬। ‘সি’ প্রোগ্রামিং ভাষাকে একটি স্ট্রাকচার্ড বা প্রোসিডিউর প্রোগ্রামিং ভাষা বলা হয়।
- ৭। ‘সি’ প্রোগ্রামিং ভাষায় পর্যাপ্ত সংখ্যক লাইব্রেরি ফাংশন এবং পর্যাপ্ত সংখ্যক অপারেটর রয়েছে যা যেকোনো জটিল প্রোগ্রাম লিখতে ব্যবহৃত হয়।
- ৮। ‘সি’ প্রোগ্রামিং ভাষায় লেখা প্রোগ্রাম যন্ত্র নির্ভরশীল নয়।

- ৯। ‘সি’ প্রোগ্রামিং ভাষার গুরুত্বপূর্ণ বৈশিষ্ট্য হল; এটি নিজেই নিজের বৈশিষ্ট্য বর্ধিত করতে পারে।

‘সি’ প্রোগ্রামিং ভাষায় লেখা একটি প্রোগ্রামের সাধারণ গঠনঃ



**Documentation Section:** এটি প্রোগ্রামের ঐচ্ছিক অংশ। এই অংশে প্রোগ্রামের নাম, বিষয়বস্তু, প্রোগ্রামারের নাম, ব্যবহারের নিয়ম [৩](#) প্রোগ্রামের উদ্দেশ্য কमेंটস এর মাধ্যমে লেখা হয়। ‘সি’ প্রোগ্রামিং ভাষায় কमेंট লেখার জন্য দুইটি পদ্ধতি আছে।

একাদিক লাইনের ক্ষেত্রে – /\*.....\*/ এবং

সিঙ্গেল লাইনের ক্ষেত্রে- //.....

**Link Section:** এটি প্রোগ্রামের অত্যাবশ্যকীয় অংশ। প্রোগ্রামে ব্যবহৃত লাইব্রেরী ফাংশনগুলোর হেডার ফাইল এই অংশে সংযুক্ত করা হয়। হেডার ফাইল যুক্ত করার নিয়ম হল- `#include<header_file_name.h>`।

**Definition Section:** এই অংশে কনস্ট্যান্ট ঘোষণা করা হয়। কনস্ট্যান্ট ঘোষণা [করার নিয়ম](#) হল-

```
#define constant_name constant_value
```

**Global Declaration Section:** এই অংশে গ্লোবাল চলক ঘোষণা করা হয়। এছাড়া ইউজার ডিফাইন্ড ফাংশনও ঘোষণা করা হয়।

**main() ফাংশন Section:** main() ফাংশন হলো প্রতিটি 'সি' প্রোগ্রামের প্রধান ফাংশন। এটি একটি ইউজার ডিফাইন্ড ফাংশন, কারণ এই ফাংশনের ডেফিনেশন প্রোগ্রামার নিজে লিখে। 'সি' প্রোগ্রামের মূল অংশ এই ফাংশনের আওতায় {} বন্ধনীর মধ্যে লিখতে হয়। এই ফাংশন ছাড়া কোনো 'সি' প্রোগ্রাম লেখা সম্ভব নয়। main() ফাংশনের দুটি অংশ থাকে। একটি Declaration Part এবং অন্যটি Execution Part। Declaration Part-এ প্রয়োজনীয় চলক, অ্যারে, পয়েন্টার, ফাইল ইত্যাদি ঘোষণা করা হয় যা নির্বাহ অংশে ব্যবহার করা হয় এবং Execution Part এ প্রোগ্রাম নির্বাহ হওয়ার জন্য কমপক্ষে একটি স্টেটমেন্ট থাকতে হয়। উভয় অংশের প্রত্যেক স্টেটমেন্টের শেষে সেমিকোলন(;) থাকতে হয়।

**Subprogram Section:** এই অংশে এক বা একাধিক ইউজার-ডিফাইন্ড ফাংশন থাকে যা main() ফাংশন থেকে Call করা হয়।

‘সি’ প্রোগ্রামিং ভাষায় লেখা একটি প্রোগ্রামের বিভিন্ন অংশের বিশ্লেষণঃ

দুটি সংখ্যা ইনপুট নিয়ে যোগফল নির্ণয় করে যোগফল প্রিন্ট করার জন্য একটি 'সি' প্রোগ্রাম

```
1 #include<stdio.h>
2 #include<conio.h>
3 main()
4 {
5     int x, y, sum;
6     scanf("%d %d",&x,&y);
7     sum = x + y;
8     printf("Sum = %d", sum);
9     getch();
10 }
```

**প্রোগ্রাম বিশ্লেষণ:**

১। 'সি' প্রোগ্রামে যেসব লাইব্রেরি ফাংশন ব্যবহার করা হয় তাদের ডেফিনেশন যে হেডার ফাইলে থাকে প্রোগ্রামের শুরুতে সেই হেডার ফাইলের নাম লিংক সেকশনে সংযুক্ত করতে হয়। প্রোগ্রামের ভিতরে printf() এবং scanf() নামে দুটি লাইব্রেরি ফাংশন ব্যবহার করা হয়েছে। ফাংশন দুটির ডেফিনেশন stdio.h নামক হেডার ফাইলে রয়েছে। তাই প্রোগ্রামের শুরুতে `#include<stdio.h>` ব্যবহার করা হয়েছে।

২। প্রোগ্রামের ভিতরে getch( ) লাইব্রেরি ফাংশন ব্যবহার করা হয়েছে। এই ফাংশনটির ডেফিনেশন conio.h নামক হেডার ফাইলে রয়েছে। তাই #include<conio.h> হেডার ফাইলটি সংযুক্ত করা হয়েছে।

৩। main ( ) ফাংশন প্রোগ্রামের মূল ফাংশন। main( ) ফাংশন থেকেই প্রোগ্রামের কার্যকারিতা শুরু হয়। প্রতিটি প্রোগ্রামে একটি main ( ) ফাংশন অবশ্যই থাকতে হবে।

৪। '{' দ্বিতীয় ব্রাকেটটি main ( ) ফাংশনটির কার্যক্রম শুরু বুঝানোর জন্য ব্যবহার করা হয়েছে।

৫। integer (পূর্ণসংখ্যা) টাইপের x, y ও sum নামে তিনটি ভেরিয়েবল ঘোষণা করা হয়েছে।

৬। scanf() ফাংশনটির মাধ্যমে ব্যবহারকারীর কাছ থেকে x ও y চলকের মান ইনপুট নেওয়া হয়েছে।

৭। x ও y চলকের মান যোগ করে sum চলকে রাখা হয়েছে।

৮। printf( ) ফাংশনটি ব্যবহার করে sum চলকের মান প্রদর্শন করা হয়েছে।

৯। getch( ) লাইব্রেরি ফাংশনটির কাজ হলো আউটপুট ব্যবহারকারী না সরানো পর্যন্ত ধরে রাখা।

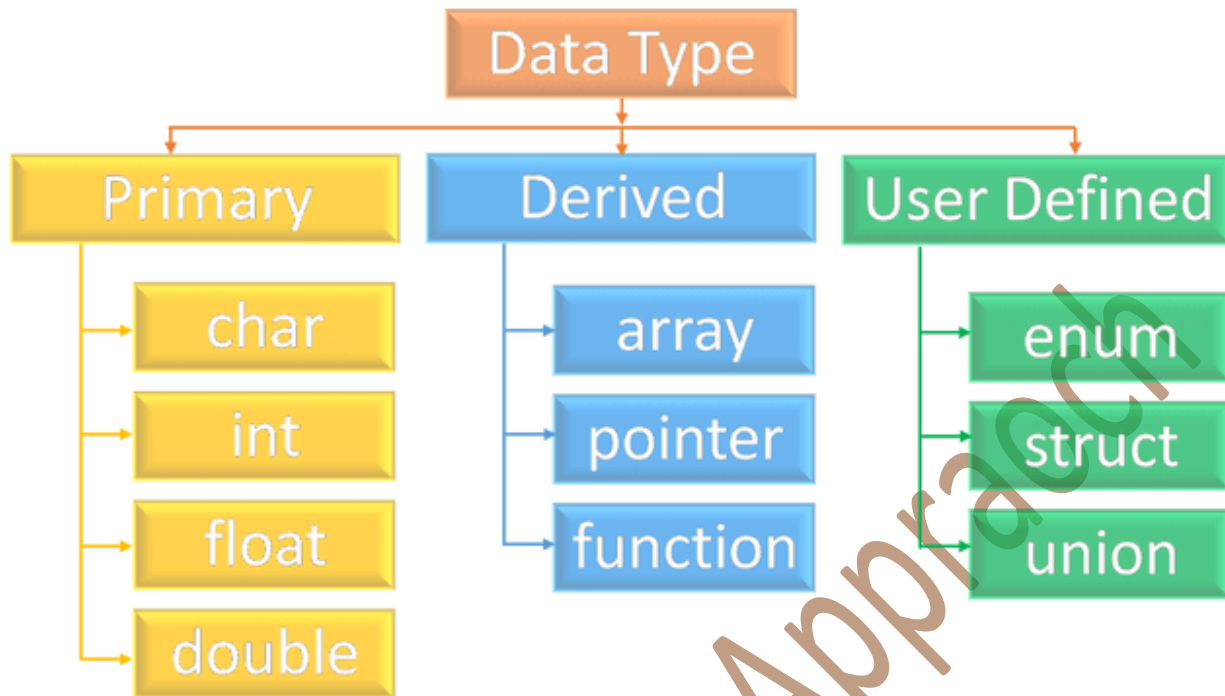
১০। '}' ব্রাকেটটি main( ) ফাংশনের কার্যক্রম শেষ বুঝানোর জন্য ব্যবহার করা হয়েছে।

**পঞ্চম অধ্যায় পাঠ-১০ ডেটা টাইপ, টোকেন, কি-ওয়ার্ড, কনস্ট্যান্ট ও ভেরিয়েবল।**

**ডেটা টাইপঃ** ডেটা টাইপ ডেটার ধরনকে নির্দেশ করে; যেমন- পূর্ণসংখ্যা, ভগ্নাংশ, ক্যারেঙ্কার ইত্যাদি। প্রতিটি ডেটা টাইপের ভিন্ন ভিন্ন পরিমাণ মেমোরি প্রয়োজন হয় এবং প্রতিটি ডেটা টাইপের উপর নির্দিষ্ট অপারেশন সম্পন্ন হয়।

**‘সি’ প্রোগ্রামে নিম্নোক্ত ডেটা টাইপগুলো ব্যবহৃত হয়ঃ**





**Primary অথবা Basic অথবা Built-in ডেটা টাইপ:**

- **char:** এই ডেটা টাইপ একটি ক্যারেকটার সংরক্ষণ করে। যেমন- 'A', 'a', '+' ইত্যাদি।
- **int:** এই ডেটা টাইপ পূর্ণসংখ্যা সংরক্ষণ করতে ব্যবহৃত হয়। যেমন- 10, 300, 6000 ইত্যাদি।
- **float:** এই ডেটা টাইপ সিঙ্গেল প্রিসিশন বিশিষ্ট ডেসিম্যাল সংখ্যা( ভগ্নাংশ মান সহ) সংরক্ষণ করতে ব্যবহৃত হয়। যেমন- 9.81, 345.7633 ইত্যাদি।
- **double:** এই ডেটা টাইপ ডাবল প্রিসিশন বিশিষ্ট ডেসিম্যাল সংখ্যা( ভগ্নাংশ মান সহ) সংরক্ষণ করতে ব্যবহৃত হয়। যেমন- 843.345678, 3293.837234 ইত্যাদি।

**‘void’ data type:** ‘void’ ডেটা টাইপ বলতে বুঝায় কোন ভ্যালু নেই। একটি ফাংশন কোন কিছুই রিটার্ন করবে না বুঝাতে এই ডেটা টাইপ ব্যবহৃত হয়।

**বিভিন্ন ডেটা টাইপের মেমোরি পরিসর, ডেটা রেঞ্জ এবং ফরম্যাট স্পেসিফায়ারঃ**

DATA TYPE	MEMORY (BYTES)	RANGE	FORMAT SPECIFIER
char	1	-128 to 127	%c
unsigned char	1	0 to 255	%c

int	2	−32,768 to 32,767	%d
unsigned int	2	0 to 65,535	%u
long int	4	−2,147,483,648 to 2,147,483,647	%ld
unsigned long int	4	0 to 4,294,967,295	%lu
float	4		%f
double	8		%lf
long double	10		%Lf

**ডেটা টাইপ মডিফায়ারঃ** মডিফায়ার হলো কিওয়ার্ড যা মৌলিক ডেটা টাইপের( float ব্যতীত ) পূর্বে ব্যবহার করে ডেটার ব্যাপ্তি এবং চলকের মেমোরি পরিসর কমানো বা বাড়ানো যায়। যেমন- signed ও unsigned মডিফায়ার char ও int টাইপ ডেটার জন্য এবং long ও short মডিফায়ার int ও double টাইপ ডেটার জন্য ব্যবহৃত হয়।

**টোকেনঃ** যে কোন প্রোগ্রাম কতগুলো স্টেটমেন্ট নিয়ে গঠিত। আবার প্রতিটি স্টেটমেন্ট কতগুলো word বা character এর সমষ্টি। ‘সি’ প্রোগ্রামে ব্যবহৃত word বা character সমূহকে একত্রে টোকেন বলে। অন্যভাবে বলা যায়; টোকেন হলো একটি প্রোগ্রামের ক্ষুদ্রতম উপাদান যা কম্পাইলারের কাছে অর্থবহ। বিভিন্ন টোকেন সমূহ –

- 1. Keywords ( eg: auto, break,int,short, while etc.)
- 2. Identifiers (eg: main, total, etc.)
- 3. Constants (eg: 9.81, 3.1416, 10, 20, etc.)
- 4. Strings (eg: “total”, “hello” etc.)
- 5. Special symbols (eg: (), {}, #, \$, @, &, etc.)
- 6. Operators (eg: +, /,-,\*, etc.)

**কিওয়ার্ড(Keywords):** কিওয়ার্ড হলো একটি প্রোগ্রামিং ভাষার পূর্ব-নির্ধারিত বা সংরক্ষিত কিছু শব্দ। প্রতিটি কিওয়ার্ড প্রোগ্রামে একটি নির্দিষ্ট কাজ সম্পাদন করে থাকে। যেহেতু কিওয়ার্ডগুলো কম্পাইলারের কাছে পরিচিত তাই তাদেরকে চলকের নাম হিসেবে ব্যবহার করা যায় না। কিওয়ার্ড সবসময় ছোট হাতের অক্ষরে লেখা হয়। ‘সি’ ভাষা ৩২ কিওয়ার্ড সাপোর্ট করে যা নিচে দেওয়া হলঃ

auto	double	int	struct
break	else	long	switch
case	enum	register	typedef
char	extern	return	union
const	float	short	unsigned
continue	for	signed	void
default	goto	sizeof	volatile
do	if	static	while



**আইডেন্টিফায়ার (Identifier):** একটি প্রোগ্রামের প্রতিটি উপাদানের একটি নাম দেওয়া হয় যাকে বলা হয় আইডেন্টিফায়ার। প্রোগ্রামের একটি নির্দিষ্ট উপাদানকে চিহ্নিত করতে বা চলক, ফাংশন, অ্যারে ইত্যাদির নাম দিতে এটি ব্যবহৃত হয়। আইডেন্টিফায়ার হলো ইউজার-ডিফাইন্ড নাম যা 'সি' এর স্ট্যান্ডার্ড ক্যারেক্টার সেটের সমন্বয়ে লেখা হয়। একটি প্রোগ্রামের প্রতিটি আইডেন্টিফায়ারের অদ্বিতীয় নাম থাকতে হয়।

**আইডেন্টিফায়ার লেখার জন্য নিচের নিয়মগুলো অনুসরণ হয়ঃ**

- ১। প্রথম ক্যারেক্টার অবশ্যই অ্যালফাবেট অথবা underscore( \_ ) হতে হবে। যেমন: Count, a7, \_number ইত্যাদি সঠিক। কিন্তু 22Roll সঠিক নয়।
- ২। কেবলমাত্র অ্যালফাবেটিক ক্যারেক্টার (A-Z, a-z), সংখ্যা(০-৯) এবং underscore( \_ ) ব্যবহার করা যাবে। যেমন- roll\_number, roll22 ইত্যাদি সঠিক। কিন্তু don't\_use, my@roll, &a ইত্যাদি সঠিক নয়।
- ৩। কোন কি-ওয়ার্ড বা রিজার্ভ ওয়ার্ড ব্যবহার করা যাবে না। যেমন: for, while, if ইত্যাদি ব্যবহার করা যাবে না।
- ৪। কোন white space ক্যারেক্টার বা ফাঁকা স্থান গ্রহণযোগ্য নয়। যেমন: roll\_number সঠিক। কিন্তু roll number সঠিক নয়।
- ৫। ছোট বা বড় হাতের অক্ষর ব্যবহার করা যায়। তবে ছোট এবং বড় হাতের অক্ষর ভিন্ন অর্থ বহন করে। যেমন: number এবং NUMBER এক নয়।
- ৬। ৩১ ক্যারেক্টারের অধিক হতে পারে। তবে প্রথম ৩১ ক্যারেক্টার তাৎপর্যপূর্ণ।

**চলক (Variable):** চলক বা ভেরিয়েবল হলো মেমরির লোকেশনের নাম বা ঠিকানা। প্রোগ্রামে যখন কোনো ডেটা নিয়ে কাজ করা হয়, প্রাথমিকভাবে সেগুলো কম্পিউটারের র‍্যামে অবস্থান করে। পরবর্তী সময়ে সেগুলো পুনরুদ্ধার বা পুনব্যবহারের জন্য ঐ নাম বা ঠিকানা জানা প্রয়োজন হয়। সুতরাং প্রোগ্রামে ডেটা নিয়ে কাজ করার সময় প্রতিটি ডেটার জন্য একটি চলক ব্যবহার করতে হয়। প্রতিবার প্রোগ্রাম নির্বাহের সময় মেমরিতে চলকগুলোর অবস্থান এবং সংরক্ষিত মান পরিবর্তন হয় বা হতে পারে বলে এদেরকে ভেরিয়েবল বা চলক বলা হয়। প্রোগ্রামে কোন চলক ব্যবহারের পূর্বে তা ঘোষণা করতে হয়। চলক ঘোষণার ক্ষেত্রে চলকের নাম লেখার সময় আইডেন্টিফায়ার লেখার নিয়মগুলো অনুসরণ করা হয়। চলক ঘোষণার ফরম্যাট –

- Data\_type variable\_name;
- যেমনঃ int number;

**ডিক্লারেশনের উপর ভিত্তি করে ভেরিয়েবলকে দুই ভাগে ভাগ করা যায়। যথা:**

- ১। লোকাল ভেরিয়েবল
- ২। গ্লোবাল ভেরিয়েবল

**১। লোকাল ভেরিয়েবল:** কোনো ফাংশনের মধ্যে ভেরিয়েবল ডিক্লেয়ার করলে তাকে উক্ত ফাংশনের লোকাল ভেরিয়েবল বা স্থানীয় চলক বলা হয়। ফাংশনের মধ্যে ঘোষণাকৃত চলক উক্ত ফাংশনের বাইরে ব্যবহার করা যায়

না। লোকাল ভেরিয়েবলের কর্মকান্ড শুধুমাত্র সংশ্লিষ্ট ফাংশনেই সীমাবদ্ধ থাকে। ভিন্ন ভিন্ন ফাংশনে একই নামের লোকাল ভেরিয়েবল থাকতে পারে।

**২। গ্লোবাল ভেরিয়েবল:** সকল ফাংশনের বাইরে প্রোগ্রামের শুরুতে ঘোষণাকৃত ভেরিয়েবলকে গ্লোবাল ভেরিয়েবল বলা হয়। গ্লোবাল ভেরিয়েবল সাধারণত প্রোগ্রামের শুরুতেই ডিক্লেয়ার করা হয়। এ ধরনের ভেরিয়েবলের কর্মকান্ড কোনো ফাংশনের মধ্যে সীমাবদ্ধ নয় বলে একে গ্লোবাল বা সার্বজনীন ভেরিয়েবল বলে।

**Constants (কনস্ট্যান্ট):** প্রোগ্রাম নির্বাহের সময় “সি” প্রোগ্রামিং ভাষায় এমন কিছু মান আছে যা কখনো পরিবর্তন হয় না। যেমন  $\pi$  এর মান হলো বা ৩.১৪১৬ যা কখনো পরিবর্তন হয় না। প্রোগ্রাম নির্বাহের সময় যে রাশির মান অপরিবর্তীত থাকে তাকে কনস্ট্যান্ট বা ধ্রুবক বলে।

‘সি’ প্রোগ্রামিং ভাষায় দুইভাবে কনস্ট্যান্ট ঘোষণা করা যায়। যথা-

- ১। `const` কীওয়ার্ড ব্যবহার করে
- ২। `#define` প্রিপ্রসেসর ব্যবহার করে

**`const` কীওয়ার্ড ব্যবহার করে ধ্রুবক ঘোষণার ফরম্যাট হলো:**

- `const ConstType ConstName = ConstValue;`
- যেমনঃ `const float PI=3.1416;`

**`#define` প্রিপ্রসেসর ব্যবহার করে ধ্রুবক ঘোষণার ফরম্যাট হলো:**

- `#define ConstName ConstValue`
- `#define PI 3.1416`

**স্ট্রিং(Strings):** স্ট্রিং হলো কতগুলো ক্যারেক্টারের সমষ্টি যার শেষ উপাদান হলো `null` ক্যারেক্টার()। এই `null` ক্যারেক্টার স্ট্রিং এর শেষ নির্দেশ করে। স্ট্রিং সবসময় ডাবল কোটেশনের (“”) সাহায্যে আবদ্ধ থাকে।

**স্ট্রিং ডিক্লোরেশন করার পদ্ধতিঃ**

- `char string[20] = {'s','t','u','d','y',' '};`
- `char string[20] = “demo”;`
- `char string [] = “demo”;`

**Special symbols:**

**বিশেষ চিহ্ন (Special symbols):** নিচের বিশেষ চিহ্নগুলো ‘সি’ ভাষায় ব্যবহৃত হয়; যার প্রত্যেকটির বিশেষ অর্থ আছে। তাই অন্য উদ্দেশ্যে ব্যবহার করা যায় না। `[] () {} , ; * = #`

- **Brackets[]**: অ্যারে এলিমেন্টের রেফারেন্স বুঝাতে ওপেনিং এবং ক্লোজিং ব্র্যাকেট ব্যবহৃত হয়। এটি সিঙ্গেল এবং মাল্টি-ডাইমেনশনাল সাবস্ক্রিপ্ট নির্দেশ করে।
- **Parentheses()**: ফাংশন কল এবং ফাংশন প্যারামিটার নির্দেশ করতে এই বিশেষ চিহ্ন ব্যবহৃত হয়।
- **Braces{}**: ওপেনিং এবং ক্লোজিং কার্লি ব্রেস যথাক্রমে একটি কোড ব্লকের শুরু ও শেষ নির্দেশ করে।
- **comma (,)**: একাধিক উপাদানকে পৃথক করতে এই চিহ্ন ব্যবহৃত হয়।
- **semi colon (;)**: একাধিক স্টেটমেন্টকে পৃথক করতে এই চিহ্ন ব্যবহৃত হয়।
- **asterick (\*)**: পয়েন্টার ভেরিয়েবল তৈরি করতে এই বিশেষ চিহ্ন ব্যবহৃত হয়।
- **assignment operator**: ভ্যালু অ্যাসাইন করতে এটি ব্যবহৃত হয়।
- **pre processor(#)**: প্রোগ্রাম ফাইল লিঙ্ক করতে কম্পাইলার অটোমেটিক্যালি এই চিহ্ন ব্যবহার করে।

**অপারেটর (Operators):** পরবর্তী পাঠে অপারেটর সম্পর্কে বিস্তারিত আলোচনা করা হবে।

**পঞ্চম অধ্যায় পাঠ-১১ ‘সি’ প্রোগ্রামিং ভাষার অপারেটরসমূহ এবং রাশিমালা।**

**অপারেটরঃ** ‘সি’ প্রোগ্রামিং ভাষায় গাণিতিক এবং যৌক্তিক কাজ সম্পাদন করার জন্য কতগুলো বিশেষ চিহ্ন বা সিম্বল ব্যবহৃত হয়, এই সিম্বল বা চিহ্নগুলোকে অপারেটর বলা হয়। অপারেটরগুলো যার উপর কাজ করে তাকে অপারেণ্ড বলা হয়। যেমনঃ  $A + B * 5$  এই এক্সপ্রেশনটিতে  $+$ ,  $*$  হলো অপারেটর ও  $A$ ,  $B$  হলো অপারেণ্ড,  $5$  হলো ধ্রুবক এবং  $A + B * 5$  হলো এক্সপ্রেশন।

**অপারেটর কতগুলো অপারেণ্ড নিয়ে কাজ করে তার উপর ভিত্তি করে তিন প্রকার। যথা-**

- ১। ইউনারি(Unary) অপারেটর
- ২। বাইনারি(Binary) অপারেটর
- ৩। টারনারি(Ternary) অপারেটর

**ইউনারি(Unary) অপারেটরঃ** যেসব অপারেটর শুধুমাত্র একটি অপারেণ্ড নিয়ে কাজ করে তাদেরকে ইউনারি(Unary) অপারেটর বলে। যেমনঃ Increment ( $++$ ) and decrement ( $--$ ) operators

**বাইনারি(Binary) অপারেটরঃ** যেসব অপারেটর দুইটি অপারেণ্ড নিয়ে কাজ করে তাদেরকে বাইনারি(Binary) অপারেটর বলে। যেমনঃ

- 1. Arithmetic operators ( $+$ ,  $-$ ,  $*$  etc.)
- 2. Relational Operators ( $<$ ,  $>$ ,  $==$ )
- 3. Logical Operators ( $\&\&$ ,  $\|\|$ )
- 4. Assignment Operators ( $=$ ,  $+=$ ,  $-=$ )
- 5. Bitwise Operators ( $\&$ ,  $\|\|$ )

**টারনারি(Ternary) অপারেটরঃ** যেসব অপারেটর তিনটি অপারেণ্ড নিয়ে কাজ করে তাদেরকে টারনারি(Ternary) অপারেটর বলে। যেমনঃ Conditional Operators( $?:$ )

কাজের প্রকৃতির উপর ভিত্তি করে ‘সি’ প্রোগ্রামিং ভাষার অপারেটর সমূহ:

- ১। গাণিতিক অপারেটর (Arithmetic Operators)
- ২। রিলেশনাল অপারেটর (Relational Operators)
- ৩। লজিক্যাল অপারেটর (Logical Operators)
- ৪। অ্যাসাইনমেন্ট অপারেটর (Assignment Operators)
- ৫। ইনক্রিমেন্ট এবং ডিক্রিমেন্ট অপারেটর (Increment and Decrement Operators)
- ৬। কন্ডিশনাল অপারেটর (Conditional Operators)
- ৭। বিট ওয়াইজ অপারেটর (Bitwise Operators)
- ৮। বিশেষ অপারেটর (Special Operator)

**গাণিতিক অপারেটর (Arithmetic Operators):** ‘সি’ প্রোগ্রামে বিভিন্ন গাণিতিক কাজ (যেমন-যোগ, বিয়োগ, গুণ, ভাগ প্রভৃতি) করার জন্য যেসব অপারেটর ব্যবহৃত হয়, সেসব অপারেটরকে গাণিতিক অপারেটর বলা হয়।

অপারেটর (Operator)	নাম (Name)	ব্যবহার (Uses)
+	plus	যোগ করার জন্য ব্যবহৃত হয়।
-	minus	বিয়োগ করার জন্য ব্যবহৃত হয়।
/	division	ভাগ করে ভাগফল নির্ণয়ের জন্য ব্যবহৃত হয়।
*	multiplier	গুণ করার জন্য ব্যবহৃত হয়।
%	modulus	ভাগশেষ বের করার জন্য ব্যবহৃত হয়।

‘সি’ প্রোগ্রামিং ভাষায় গাণিতিক অপারেটর গুলোর অগ্রগণ্যতা হলঃ \*, /, %, +, -

**রিলেশনাল অপারেটর (Relational Operators):** প্রোগ্রাম নির্বাহের সময় দুটি চলকের মধ্যে তুলনার ক্ষেত্রে রিলেশনাল অপারেটর ব্যবহৃত হয়। রিলেশন বলতে একটি অপারেন্ড অপার অপারেন্ড থেকে ছোট কিংবা বড় বা সমান ইত্যাদি বুঝায়।

রিলেশনাল অপারেটর	কাজ	উদাহরণ	বর্ণনা
<	Less than (ছোট)	$a < b$	a এর মান b মানের চেয়ে ছোট
<=	Less than or equal (ছোট বা সমান)	$a <= b$	a এর মান b মানের চেয়ে ছোট অথবা সমান
>	Greater than (বড়)	$a > b$	a এর মান b মানের চেয়ে বড়
>=	Greater than or equal (বড় বা সমান)	$a >= b$	a এর মান b মানের চেয়ে বড় অথবা সমান।
==	Equal to (সমান)	$a == b$	a এর মান b মানের সমান
!=	Not equal to (অসমান)	$a != b$	a ও b এর মান সমান নয়

**লজিক্যাল অপারেটর (Logical Operators):** প্রোগ্রামে যুক্তিমূলক এক্সপ্রেশন নিয়ে কাজ করার জন্য যেসব অপারেটর ব্যবহার করা হয় সেগুলোকে লজিক্যাল অপারেটর বলা হয়।

অপারেটর এর চিহ্ন	অপারেটর এর নাম	কার্যপদ্ধতি	উদাহরণ
&&	AND অপারেটর	যদি উভয় অপারেন্ড এর মান শূন্য না হয় তবেই শর্তটি সত্য বা true হবে।	(A && B) is true.
	OR অপারেটর	যদি দুটি অপারেন্ড এর কমপক্ষে একটি মান শূন্য না হয় তবেই শর্তটি সত্য বা true হবে।	(A    B) is true.
!	NOT অপারেটর	অপারেন্ড এর মান বিপরীত অর্থে ব্যবহৃত হয়। যদি একটি শর্ত সত্য বা true হয় এবং সেক্ষেত্রে লজিক্যাল NOT অপারেটর ব্যবহারের ফলে শর্তটি মিথ্যা বা false হবে।	!(A && B) is false.

**অ্যাসাইনমেন্ট অপারেটর (Assignment Operators):** কোনো এক্সপ্রেশন বা ভেরিয়েবলের মানকে অন্য কোনো ভেরিয়েবলের মান হিসেবে নির্ধারণ করতে যেসব অপারেটর ব্যবহার করা হয়, সেগুলোকে অ্যাসাইনমেন্ট অপারেটর বলা হয়।

Simple assignment operator	Short hand assignment operator
$a = a +$	$a += 1$
$a = a - 1$	$a -= 1$
$a = a * b$	$a *= b$
$a = a / b$	$a /= b$
$a = a \% b$	$a \% = b$

**ইনক্রিমেন্ট এবং ডিক্রিমেন্ট অপারেটর (Increment and Decrement Operators):**

‘সি’ প্রোগ্রামে দুটি গুরুত্বপূর্ণ অপারেটর ব্যবহার করা হয় যা সাধারণত অন্য ভাষায় ব্যবহার করা হয় না। অপারেটর দুটি হচ্ছে Increment ( ++ ) and Decrement ( — ) Operators । ইনক্রিমেন্ট অপারেটর ব্যবহার করা হয় কোন একটি ভেরিয়েবলের মান ১ বৃদ্ধি করতে এবং ডিক্রিমেন্ট অপারেটর ব্যবহার করা হয় কোন একটি ভেরিয়েবলের মান ১ হ্রাস করতে। ইনক্রিমেন্ট এবং ডিক্রিমেন্ট উভয় অপারেটর একটি অপারেন্ডের উপর কাজ করে। তাই এদেরকে ইউনারি অপারেটর বলা হয়।

**ইনক্রিমেন্ট অপারেটরের প্রকারভেদঃ**

- pre-increment
- post-increment

**pre-increment ( ++ variable ): pre** ইনক্রিমেন্ট এর ক্ষেত্রে চলকের মান আগে বৃদ্ধি করে এবং তারপর আপডেট মানটি নিয়ে কাজ করে।

```

Example of pre-increment
1 #include<stdio.h>
2 #include<conio.h>
3
4 main()
5 {
6     int x,i;
7     i=10;
8     x=++i;
9     printf("x: %d",x);
10    printf("i: %d",i);
11    getch();
12 }

```

**Output:**

```

1 x: 11
2 i: 11

```

**post-increment (variable ++):** post ইনক্রিমেন্ট এর ক্ষেত্রে চলকের বর্তমান মান নিয়ে কাজ করে তারপর চলকের মান বৃদ্ধি করে।

```

Example of post-increment
1 #include<stdio.h>
2 #include<conio.h>
3
4 void main()
5 {
6     int x,i;
7     i=10;
8     x=i++;
9     printf("x: %d",x);
10    printf("i: %d",i);
11    getch();
12 }

```

**Output:**

```

1 x: 10
2 i: 11

```

**ডিক্রিমেন্ট অপারেটরের প্রকারভেদঃ**

- pre-decrement
- post-decrement

**Pre-decrement (– variable):** pre ডিক্রিমেন্ট এর ক্ষেত্রে চলকের মান আগে হ্রাস করে এবং তারপর আপডেট মানটি নিয়ে কাজ করে।

```

Example of pre-decrement
1 #include<stdio.h>
2 #include<conio.h>
3
4 void main()
5 {
6     int x,i;
7     i=10;
8     x=--i;
9     printf("x: %d",x);
10    printf("i: %d",i);
11    getch();
12 }

```

**Output:**

```

1 x: 9
2 i: 9

```

**post-decrement (variable --):** post ডিক্রিমেন্ট এর ক্ষেত্রে চলকের বর্তমান মান নিয়ে কাজ করে তারপর চলকের মান হ্রাস করে।

```

Example of post-decrement
1 #include<stdio.h>
2 #include<conio.h>
3
4 void main()
5 {
6     int x,i;
7     i=10;
8     x=i--;
9     printf("x: %d",x);
10    printf("i: %d",i);
11    getch();
12 }

```

**Output:**

```

1 x: 10
2 i: 9

```

**কন্ডিশনাল অপারেটর (Conditional Operators):**

‘সি’ প্রোগ্রামে শর্ত সাপেক্ষে কোন কাজ করার জন্য কন্ডিশনাল অপারেটর ব্যবহৃত হয়। কন্ডিশনাল অপারেটরের গঠন নিম্নরূপঃ

**Syntax :** (Condition? true\_value: false\_value);

**Example :** (A < 0 ? Negative : Positive);

উপরের উদাহরণে, যদি A , 0 এর চেয়ে ছোট হয় তাহলে Negative রিটার্ন করবে অন্যথায় Positive রিটার্ন করবে।





কন্ডিশন সত্য হলে Expr1 সম্পাদিত হবে এবং কন্ডিশন মিথ্যা হলে Expr2 সম্পাদিত হবে। এই অপারেটরকে টারনারি অপারেটরও বলা হয়।

**বিট ওয়াইজ অপারেটর (Bitwise Operators):** ‘সি’ প্রোগ্রামে বিট পরীক্ষা করার জন্য বা কোনো বিট ডানে বা বামে সরানোর জন্য বিট ওয়াইজ অপারেটর ব্যবহার করা হয়। Float বা Double টাইপের ডেটার ক্ষেত্রে বিট ওয়াইজ অপারেটর ব্যবহার করা যায় না।

Operator	Meaning	Example
&	Bitwise AND	a&b
	Bitwise OR	a b
^	Bitwise Exclusive OR	a^b
~	Bitwise NOT	~a
<<	Left Shift	a<<1
>>	Right Shift	a>>1

বিট ওয়াইজ অপারেটর এর সত্যক সারণিঃ

A	B	A   B	A & B	A ^ B	~A
0	0	0	0	0	1
0	1	1	0	1	1
1	0	1	0	1	0
1	1	1	1	0	0

**বিশেষ অপারেটর (Special Operator):** ‘সি’ প্রোগ্রামে বিশেষ কিছু কাজের জন্য ব্যবহৃত অপারেটরকে বিশেষ অপারেটর বলে। যেমনঃ

## অপারেটর বর্ণনা

চলকের অ্যাড্রেস পেতে এই অপারেটর ব্যবহৃত হয়।

&

উদাহরণ : &a, a এর অ্যাড্রেস দিবে।

চলকের পয়েন্টার হিসেবে ব্যবহৃত হয়।

\*

উদাহরণ : \* a যেখানে, \* হলো a চলকের পয়েন্টার।

চলকের মেমোরি সাইজ রিটার্ন করে।

Sizeof ()

উদাহরণ: size of (char); ১ রিটার্ন করবে।

**রাশিমালা (Expression):** চলক, ধ্রুবক ও বিভিন্ন অপারেটরের সমন্বয়ে রাশিমালা বা Expression তৈরি হয়।  
গাণিতিক Expression গুলো 'সি' প্রোগ্রামে নিম্নরূপে লেখা হয়-

গাণিতিক এক্সপ্রেশন	'সি' ভাষায় এক্সপ্রেশন
$ax^2+bx+c$	$a*x*x+b*x+c$
$Y=a^3+b^3+c^3$	$Y=a*a*a+b*b*b+c*c*c$
$F=x+\frac{y}{z}+c$	$F=x+(y/z)+c$
$Y=(a^n)^m$	$Y=pow((pow(a,n)),m)$
$Y=\sqrt{b^2-4ac}$	$Y=sqrt(b*b-4*a*c)$
$Y= a-b +c$	$Y=abs(a-b)+c$

## Expression বা রাশিমালায় অপারেটরের precedence এবং associativity:

একটি রাশিমালায় একাধিক অপারেটর থাকলে কম্পাইলার অপারেটরগুলোর অগ্রগণ্যতা দেয় এবং অগ্রগণ্যতা অনুযায়ী কাজ করে। প্রোগ্রামিং ভাষায় একে অপারেটরের precedence বলে। 'সি' প্রোগ্রামিং ভাষায় গাণিতিক অপারেটর গুলোর অগ্রগণ্যতা হল: \*, /, %, +, - । অর্থাৎ  $x+y*x-a$  এই রাশিমালায় আগে গুণের কাজ, তারপর যোগের কাজ এবং সবশেষে বিয়োগের কাজ সম্পন্ন হবে।

আবার একটি রাশিমালায় একই precedence বিশিষ্ট একাধিক অপারেটর থাকলে কম্পাইলার কিছু অপারেটরের ক্ষেত্রে বামদিক থেকে ডানদিকে এবং কিছু অপারেটরের ক্ষেত্রে ডানদিক থেকে বামদিকে কাজ করে। প্রোগ্রামিং ভাষায় একে অপারেটর associativity বলে। যেমন-  $x+y+z+a$  রাশিমালায় '+' অপারেটর একাধিকবার ব্যবহৃত হয়েছে। এই '+' অপারেটরের associativity হল বাম থেকে ডান। তাই উপরের রাশিমালায় প্রথমে x ও y যোগ করে তার সাথে z এর যোগ এবং অবশেষে প্রাপ্ত যোগফলের সাথে a এর যোগ।

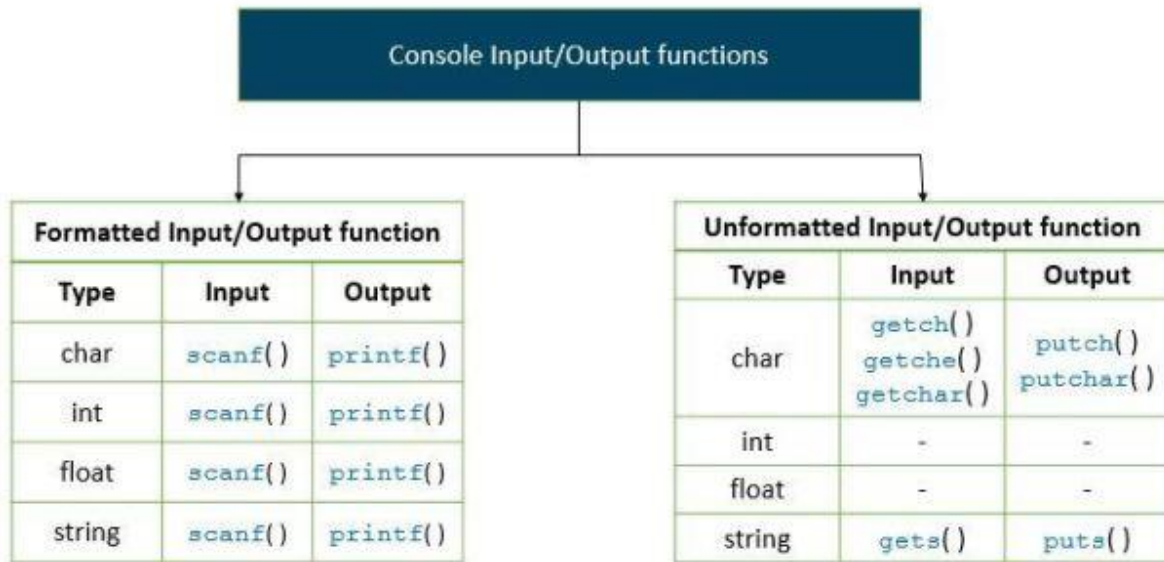
নিচের টেবিলে, সর্বোচ্চ precedence (অগ্রাধিকার) বিশিষ্ট অপারেটর টেবিলের শীর্ষে অবস্থিত এবং সর্বনিম্ন precedence (অগ্রাধিকার) বিশিষ্ট অপারেটর টেবিলের সর্বনিম্ন নীচে অবস্থিত।

Category	Operator	Associativity
Postfix	() [] -> . ++ --	Left to right
Unary	+ - ! ~ ++ -- (type)* & sizeof	Right to left
Multiplicative	* / %	Left to right
Additive	+ -	Left to right
Shift	<< >>	Left to right
Relational	< <= > >=	Left to right
Equality	== !=	Left to right
Bitwise AND	&	Left to right
Bitwise XOR	^	Left to right
Bitwise OR		Left to right
Logical AND	&&	Left to right
Logical OR		Left to right
Conditional	?:	Right to left
Assignment	= += -= *= /= %=>>= <<= &= ^=  =	Right to left
Comma	,	Left to right

পঞ্চম অধ্যায় পাঠ-১২ 'সি' প্রোগ্রামিং ভাষায় ইনপুট এবং আউটপুট ফাংশন সমূহ

'সি' প্রোগ্রামিং ভাষায় ইনপুট এবং আউটপুট ফাংশনসমূহঃ

কোন প্রোগ্রামে ডেটা প্রক্রিয়া করার জন্য প্রথমে ডেটা ইনপুট নিতে হয়। প্রোগ্রামে ডেটা ইনপুট নেওয়ার জন্য ব্যবহৃত ফাংশনকে ইনপুট ফাংশন বলে। আবার প্রক্রিয়া পরবর্তী তথ্য আউটপুটে প্রদর্শনের জন্য ব্যবহৃত ফাংশনকে আউটপুট ফাংশন বলে। 'সি' প্রোগ্রামিং ভাষায় ইনপুট নেওয়া এবং আউটপুট দেখানোর জন্য বিভিন্ন লাইব্রেরী ফাংশন রয়েছে। ফাংশনসমূহঃ



**ফরমেট স্পেসিফায়ারঃ** ‘সি’ প্রোগ্রামের কোন চলকে ফরমেটেড আকারে ডেটা গ্রহণ বা ফরমেটেড আকারে কোন চলকের মান প্রদর্শনের জন্য যথাক্রমে ইনপুট ও আউটপুট ফাংশনে যে সকল ক্যারেক্টার সেট ব্যবহৃত হয় তাদের ফরমেট স্পেসিফায়ার বলা হয়। প্রতিটি ফরমেট স্পেসিফায়ার পার্সেন্টেজ ক্যারেক্টার(%) দিয়ে শুরু হয়।

**বিভিন্ন ডেটা টাইপের জন্য ফরমেটেড ইনপুট ও আউটপুট ফাংশনে ব্যবহৃত ফরমেট স্পেসিফায়ারসমূহঃ**

ফরমেট স্পেসিফায়ার	ব্যবহার	উদাহরণ
%c	char টাইপের ডেটা ইনপুট/আউটপুটের জন্য	<code>scanf("%c",&amp;a); printf("%c",a);</code>
%d	int টাইপের ডেটা ইনপুট/আউটপুটের জন্য	<code>scanf("%d",&amp;a); printf("%d",a);</code>
%f	float টাইপের ডেটা ইনপুট/আউটপুটের জন্য	<code>scanf("%f",&amp;a); printf("%f",a);</code>
%lf	double টাইপের ডেটা ইনপুট/আউটপুটের জন্য	<code>scanf("%lf",&amp;a); printf("%lf",a);</code>
%ld	long int টাইপের ডেটা ইনপুট/আউটপুটের জন্য	<code>scanf("%ld",&amp;a); printf("%ld",a);</code>
%u	unsigned int টাইপের ডেটা ইনপুট/আউটপুটের জন্য	<code>scanf("%u",&amp;a); printf("%u",a);</code>
%o	Octal ডেটা ইনপুট/আউটপুটের জন্য	<code>scanf("%o",&amp;a); printf("%o",a);</code>
%x	Hexadecimal ডেটা ইনপুট/আউটপুটের জন্য	<code>scanf("%x",&amp;a); printf("%x",a);</code>

**scanf() ফাংশনের ব্যবহারঃ**

পূর্বে ঘোষণাকৃত একটি চলকে ডেটা ইনপুট নেওয়ার জন্য `scanf()` ফাংশন ব্যবহারের ফরমেটঃ

scanf("format\_specifier ", &variable\_name);

উদাহরণঃ

- a চলকে char টাইপের ডেটা ইনপুট নেওয়ার জন্য scanf() ফাংশনঃ scanf("%c", &a);
- a চলকে int টাইপের ডেটা ইনপুট নেওয়ার জন্য scanf() ফাংশনঃ scanf("%d", &a);
- a চলকে float টাইপের ডেটা ইনপুট নেওয়ার জন্য scanf() ফাংশনঃ scanf("%f", &a);
- a চলকে double টাইপের ডেটা ইনপুট নেওয়ার জন্য scanf() ফাংশনঃ scanf("%lf", &a);

পূর্বে ঘোষণাকৃত একাধিক চলকের ডেটা একসাথে ইনপুট নেওয়ার জন্য scanf() ফাংশন ব্যবহারের ফরমেটঃ

scanf(" format\_specifier1 format\_specifier2....", &variable\_name1, &variable\_name2.....);

উদাহরণঃ

একসাথে একাধিক চলকে একই ধরনের ডেটা ইনপুট নেওয়ার জন্য scanf() ফাংশনের ব্যবহারঃ

- a,b ও c চলকে int টাইপের ডেটা ইনপুট নেওয়ার জন্য scanf() ফাংশনঃ
- scanf("%d %d %d", &a, &b, &c);

একসাথে একাধিক চলকে ভিন্ন ভিন্ন ধরনের ডেটা ইনপুট নেওয়ার জন্য scanf() ফাংশনের ব্যবহারঃ

- a,b ও c চলকে যথাক্রমে int, float ও double টাইপের ডেটা ইনপুট নেওয়ার জন্য scanf() ফাংশনঃ
- scanf("%d %f %lf", &a, &b, &c);

**printf() ফাংশনের ব্যবহারঃ**

printf() ফাংশন দুইভাবে ব্যবহার করা যায়। প্রথমত, কোন কিছু হুবহু আউটপুটে দেখানো। দ্বিতীয়ত, কোন এক বা একাধিক চলকের মান আউটপুটে দেখানো।

কোন কিছু হুবহু আউটপুটে দেখানোর জন্য printf() ফাংশনের ফরমেটঃ

আউটপুটে দেখানোর প্রয়োজনীয় টেক্সটটি printf(" "); ফাংশনের ডাবল কোটেশনের মধ্যে লিখতে হয়। যেমন-

printf(" Output text should be here ");

কোন একটি চলকের মান আউটপুটে দেখানোর জন্য printf() ফাংশনের ফরমেটঃ

printf("format\_specifier", variable\_name);

উদাহরণঃ

- a চলকের char টাইপের ডেটা আউটপুটে দেখানোর printf() ফাংশনঃ printf(“%c”, a);
- a চলকের int টাইপের ডেটা আউটপুটে দেখানোর printf() ফাংশনঃ printf(“%d”, a);
- a চলকের float টাইপের ডেটা আউটপুটে দেখানোর printf() ফাংশনঃ printf(“%f”, a);
- a চলকের double টাইপের ডেটা আউটপুটে দেখানোর printf() ফাংশনঃ printf(“%lf”, a);

একাধিক চলকের মান একসাথে আউটপুটে দেখানোর জন্য printf() ফাংশনের ফরমেটঃ

printf(“format\_specifier1, format\_specifier2....”, variable\_name1, variable\_name2...);

একসাথে একাধিক চলকের একই ধরনের ডেটা আউটপুটে দেখানোর printf() ফাংশনের ব্যবহারঃ

- a, b ও c চলকের ডেটা আউটপুটে int টাইপের দেখানোর জন্য printf() ফাংশনঃ
- printf(“%d %d %d”, a, b, c);

একসাথে একাধিক চলকের ভিন্ন ভিন্ন ধরনের ডেটা আউটপুটে দেখানোর printf() ফাংশনের ব্যবহারঃ

- a, b ও c চলকের ডেটা আউটপুটে যথাক্রমে int, float ও double টাইপের দেখানোর printf() ফাংশনের ব্যবহারঃ
- printf(“%d %f %lf”, a, b, c);

**ব্যাকস্ল্যাশ ক্যারেঞ্জারঃ** বিশেষ কিছু ক্যারেঞ্জার আছে যেগুলো printf() ফাংশনের মাধ্যমে সরাসরি প্রদর্শন করা যায় না। এই বিশেষ ক্যারেঞ্জারগুলো প্রদর্শনের জন্য অতিরিক্ত একটি ক্যারেঞ্জার( ) ব্যবহৃত হয় যাকে ব্যাকস্ল্যাশ ক্যারেঞ্জার বলে।

**ব্যাকস্ল্যাশ ক্যারেঞ্জারের ব্যবহারঃ**

ব্যাকস্ল্যাশ ক্যারেঞ্জার	ব্যবহার	উদাহরণ	আউটপুট
n	আউটপুট নতুন লাইনে প্রদর্শনের জন্য	printf(“Learning ICT”);	Learning ICT
t	আউটপুটে horizontal tab প্রদর্শনের জন্য	printf(“Learning gt ICT”);	Learning ICT
v	আউটপুটে vertical	printf(“Learning gv ICT”);	Learning ICT

	tab	প্রদর্শ	
		নের জন্য	
a	সতর্ক সংকেত	printf("aLearnin	Alert!
	প্রদানের জন্য	ng ICT");	
	আউটপুটে		
„	ডাবল	printf("Learnin	Learning "ICT"
	কোটেসন( " )	g "ICT" ");	
	প্রদর্শনের জন্য		
	আউটপুটে		
,	সিঙ্গেল	printf("Learnin	Learning 'ICT'
	কোটেসন( ' )	g 'ICT' ");	
	প্রদর্শনের জন্য		
	আউটপুটে		
	ব্যাকস্ল্যাশ	printf("Learnin	Learning ICT
	ক্যারেক্টার ( )	g ICT ");	
	প্রদর্শনের জন্য		
	আউটপুটে		
?	প্রশ্নবোধক	printf(" Are	
	চিহ্ন ( ? )	you learning	Are you learning ICT?
	প্রদর্শনের জন্য	ICT? ");	

পঞ্চম অধ্যায় পাঠ-১৩ সাধারণ গাণিতিক সমস্যা সম্পর্কিত প্রোগ্রামসমূহ।

১। দুইটি সংখ্যা ইনপুট নিয়ে যোগফল নির্ণয় করার জন্য 'সি' প্রোগ্রামিং ভাষায় একটি প্রোগ্রাম।

```

1 #include<stdio.h>
2 #include<conio.h>
3 main()
4 {
5     int a, b, c;
6     printf(" Enter Two numbers: ");
7     scanf("%d %d",&a,&b);
8     c = a+b;
9     printf("Summation = %d",c);
10    getch();
11 }

```

অনুশীলনঃ পাঁচটি সংখ্যা ইনপুট নিয়ে যোগফল নির্ণয় করার প্রোগ্রাম তৈরি কর।

২। দুইটি সংখ্যা ইনপুট নিয়ে বিয়োগফল নির্ণয় করার জন্য সি প্রোগ্রামিং ভাষায় একটি প্রোগ্রাম।

```

1 #include<stdio.h>
2 #include<conio.h>
3 main()
4 {
5     int a, b, c;
6     printf(" Enter Two numbers: ");
7     scanf("%d %d",&a,&b);
8     c = a-b;
9     printf("Subtraction= %d",c);
10    getch();
11 }

```

৩। দুইটি সংখ্যা ইনপুট নিয়ে গুণফল নির্ণয় করার জন্য সি প্রোগ্রামিং ভাষায় একটি প্রোগ্রাম।

```

1 #include<stdio.h>
2 #include<conio.h>
3 main()
4 {
5     int a, b, c;
6     printf(" Enter Two numbers: ");
7     scanf("%d %d",&a,&b);
8     c = a*b;
9     printf("Multiplication= %d", c);
10    getch();
11 }

```

অনুশীলনঃ পাঁচটি সংখ্যা ইনপুট নিয়ে গুণফল নির্ণয় করার প্রোগ্রাম তৈরি কর।

৪। দুইটি সংখ্যা ইনপুট নিয়ে ভাগফল নির্ণয় করার জন্য সি প্রোগ্রামিং ভাষায় একটি প্রোগ্রাম।

```

1 #include<stdio.h>
2 #include<conio.h>
3 main()
4 {
5     int a, b;
6     float c;
7     printf("Enter Two numbers: ");
8     scanf("%d %d",&a,&b);
9     c = a/b;
10    printf("Division= %f",c);
11    getch();
12 }

```

অনুশীলনঃ পাঁচটি সংখ্যা ইনপুট নিয়ে তাদের গড় নির্ণয় করার প্রোগ্রাম তৈরি কর।

৫। সেলসিয়াস স্কেলের তাপমাত্রাকে ফারেনহাইট স্কেলের তাপমাত্রায় রূপান্তরের প্রোগ্রাম।



তাপমাত্রা পরিমাপের বিভিন্ন স্কেলের মধ্যে সম্পর্ক-  $C/5 = F-32 / 9 = K-273 / 5$

```
1 #include<stdio.h>
2 #include<conio.h>
3 main()
4 {
5     int c, f;
6     printf("Enter celcius temperature :");
7     scanf("%d",&c);
8     f=9*c/5+32;
9     printf("Ferenheight temperature:%d",f);
10    getch();
11 }
```

অনুশীলনঃ সেলসিয়াস স্কেলের তাপমাত্রাকে কেলভিন স্কেলের তাপমাত্রায় রূপান্তরের প্রোগ্রাম তৈরি কর।

৬। ফারেনহাইট স্কেলের তাপমাত্রাকে সেলসিয়াস স্কেলের তাপমাত্রায় রূপান্তরের প্রোগ্রাম।

তাপমাত্রা পরিমাপের বিভিন্ন স্কেলের মধ্যে সম্পর্ক-  $C/5 = F-32 / 9 = K-273 / 5$

```
1 #include<stdio.h>
2 #include<conio.h>
3 main()
4 {
5     int c, f;
6     printf("Enter Ferenheight temperature :");
7     scanf("%d",&f);
8     c=5*(f-32)/9;
9     printf("Celcius temperature:%d",c);
10    getch();
11 }
```

অনুশীলনঃ ফারেনহাইট স্কেলের তাপমাত্রাকে কেলভিন তাপমাত্রায় রূপান্তরের প্রোগ্রাম তৈরি কর।

৭। ত্রিভুজের ভূমি ও উচ্চতা দেওয়া থাকলে ক্ষেত্রফল বের করার প্রোগ্রাম লেখ।

```
1 #include<stdio.h>
2 #include<conio.h>
3 main()
4 {
5     int b,h;
6     float area;
7     printf("Enter Base & Height:");
8     scanf("%d %d",&b,&h);
9     area=.5*b*h;
10    printf("\nThe area is %.2f",area);
11    getch();
12 }
```

৮। ত্রিভুজের তিনটি বাহুর দৈর্ঘ্য যথাক্রমে a,b এবং c দেওয়া আছে। ত্রিভুজের ক্ষেত্রফল বের করার প্রোগ্রাম।

```

1 #include<stdio.h>
2 #include<conio.h>
3 #include<math.h>
4 main()
5 {
6     int a, b, c;
7     float s, area;
8     printf("Enter three integer values:");
9     scanf("%d %d %d", &a,&b,&c);
10    s = (a + b + c)/2;
11    area = sqrt(s*(s-a)*(s-b)*(s-c));
12    printf("Area of triangle is = %f", area);
13    getch();
14 }

```

৯। আয়তক্ষেত্রের ক্ষেত্রফল নির্ণয় করার প্রোগ্রাম।

```

1 #include<stdio.h>
2 #include<conio.h>
3 main()
4 {
5     int l,w,area;
6     printf("Enter the length & width: ");
7     scanf("%d %d",&l,&w);
8     area=l*w;
9     printf("\nThe area is %d",area);
10    getch();
11 }

```

১০। বৃত্তের ক্ষেত্রফল নির্ণয় করার প্রোগ্রাম।

```

1 #include<stdio.h>
2 #include<conio.h>
3 main ( )
4 {
5     int r;
6     float area;
7     printf ("Enter integer value for radius:");
8     scanf ("%d", &r) ;
9     area = 3.1416*r*r;
10    printf("\n Area of circle =%f", area);
11    getch();
12 }

```

প্রশ্নের ধরণঃ

- ১। কোন অ্যালগোরিদম বা ফ্লোচার্ট দেওয়া থাকলে তার প্রোগ্রাম লিখতে হতে পারে।
- ২। কোন প্রোগ্রাম দেওয়া থাকবে তার ভুল নির্ণয় এবং সমাধান করতে হতে পারে।

পঞ্চম অধ্যায় পাঠ-১৪ 'সি' প্রোগ্রামিং ভাষায় কন্ডিশনাল কন্ট্রোল স্টেটমেন্ট।

**কন্ট্রোল স্টেটমেন্টঃ** ‘সি’ প্রোগ্রামিং ভাষায় স্টেটমেন্টসমূহ সাধারণত স্বয়ংক্রিয়ভাবে পর্যায়ক্রমে নির্বাহ হয়। কিন্তু বিভিন্ন পরিস্থিতিতে প্রোগ্রামের নির্বাহ নিয়ন্ত্রণ (যেমন- এক বা একাধিক স্টেটমেন্ট একাধিক বার নির্বাহ, শর্ত সাপেক্ষে কোন এক বা একাধিক স্টেটমেন্ট নির্বাহ, এক স্টেটমেন্ট থেকে অন্য স্টেটমেন্টে প্রোগ্রামের নিয়ন্ত্রণ স্থানান্তর ইত্যাদি) করার প্রয়োজন হয়। যে সকল স্টেটমেন্ট এর সাহায্যে প্রোগ্রাম স্টেটমেন্টসমূহের পর্যায়ক্রমিক নির্বাহ নিয়ন্ত্রণ করা যায়, তাদেরকে কন্ট্রোল স্টেটমেন্ট বলে।

### কন্ট্রোল স্টেটমেন্ট সমূহঃ

- ১। কন্ডিশনাল কন্ট্রোল স্টেটমেন্ট/ ডিসিশন কন্ট্রোল স্টেটমেন্ট
- ২। লুপ কন্ট্রোল স্টেটমেন্ট
- ৩। জাম্পিং কন্ট্রোল স্টেটমেন্ট

**কন্ডিশনাল কন্ট্রোল স্টেটমেন্টঃ** ‘সি’ প্রোগ্রামে শর্তসাপেক্ষে এক বা একাধিক স্টেটমেন্ট নির্বাহের জন্য কন্ডিশনাল কন্ট্রোল স্টেটমেন্ট ব্যবহৃত হয়। কন্ডিশনাল কন্ট্রোল স্টেটমেন্টে ব্যবহৃত শর্ত সত্য হলে প্রোগ্রামে এক ধরনের ফলাফল পাওয়া যায় এবং মিথ্যা হলে অন্য ধরনের ফলাফল পাওয়া যায়।

### ‘সি’ প্রোগ্রামিং ভাষায় কন্ডিশনাল কন্ট্রোল স্টেটমেন্টসমূহঃ

- ১। if স্টেটমেন্ট
- ২। if-else স্টেটমেন্ট
- ৩। else if স্টেটমেন্ট
- ৪। nested if-else স্টেটমেন্ট
- ৫। switch স্টেটমেন্ট

**if স্টেটমেন্টঃ** প্রোগ্রামে শর্ত সাপেক্ষে এক বা একাধিক স্টেটমেন্ট নির্বাহের জন্য if স্টেটমেন্ট ব্যবহার করা হয়। এক্ষেত্রে if স্টেটমেন্ট তার কন্ডিশনটি চেক করে। যদি কন্ডিশন সত্য হয় তাহলে বডি'র মধ্যে অবস্থিত স্টেটমেন্টসমূহ নির্বাহ হয়। আর যদি কন্ডিশন মিথ্যা হয় তাহলে বডি'র মধ্যে অবস্থিত স্টেটমেন্টসমূহ নির্বাহ হয় না। if স্টেটমেন্ট ব্যবহারের ফরম্যাট হলো-

```
if statement
1 if(condition)
2 {
3     statement(s)
4 }
```

কীবোর্ড থেকে কোনো সংখ্যা ইনপুট দিয়ে দেখবে সংখ্যাটি ধনাত্মক কিনা?

```

1 #include<stdio.h>
2 #include<conio.h>
3
4 main()
5 {
6     int a;
7     printf("Enter a number");
8     scanf("%d",&a);
9     if(a>0)
10         printf("The given number is Positive");
11     getch();
12 }

```

**if-else স্টেটমেন্ট:** if-else স্টেটমেন্টের ক্ষেত্রে if এর কন্ডিশনটি সত্য হলে নির্দিষ্ট স্টেটমেন্টসমূহ নির্বাহ হয়। আর যদি প্রোগ্রামের কোন কন্ডিশন সত্য না হয়, তাহলে else এর স্টেটমেন্টসমূহ নির্বাহ হয়। ‘সি’ প্রোগ্রামে ‘অন্যথায়’ অর্থে else স্টেটমেন্ট ব্যবহৃত হয়। else স্টেটমেন্টে কোন কন্ডিশন থাকে না। if-else স্টেটমেন্ট ব্যবহারের ফরম্যাট হলো-

```

1 if(condition)
2 {
3     statement(s)
4 }
5 else
6 {
7     statement(s)
8 }

```

কীবোর্ড থেকে কোনো সংখ্যা ইনপুট দিয়ে দেখবে সংখ্যাটি ধনাত্মক না ঋণাত্মক।

```

1 #include<stdio.h>
2 #include<conio.h>
3
4 main()
5 {
6     int a;
7     printf("Enter a number");
8     scanf("%d",&a);
9     if(a>=0)
10         printf("The given number is Positive");
11     else
12         printf("The given number is Negative");
13     getch();
14 }

```

**else if স্টেটমেন্ট:** প্রোগ্রামে যদি একাধিক কন্ডিশন যাচাই করতে হয় তাহলে প্রথম কন্ডিশন যাচাই করার জন্য if স্টেটমেন্ট ব্যবহার করা হয়। তারপরের কন্ডিশন গুলো যাচাই করার জন্য else if স্টেটমেন্ট ব্যবহার করা হয়। ‘সি’ প্রোগ্রামে “অন্যথায় যদি” অর্থে else if স্টেটমেন্ট ব্যবহার করা হয়। else if স্টেটমেন্ট ব্যবহারের ফরম্যাট হলো-

```

1  if(Condition1)
2  {
3      statement(s)
4  }
5  else if(Condition2)
6  {
7      statement(s)
8  }
9  .....
10 .....
11
12 else
13 {
14     statement(s)
15 }

```

কীবোর্ড থেকে কোনো সংখ্যা ইনপুট দিয়ে দেখবে সংখ্যাটি শূন্য, ধনাত্মক অথবা ঋনাত্মক।

```

1  #include<stdio.h>
2  #include<conio.h>
3
4  main()
5  {
6      int a;
7      printf("Enter a number");
8      scanf("%d",&a);
9      if(a==0)
10         printf("The given number is Zero");
11     else if(a>0)
12         printf("The given number is Positive");
13     else
14         printf("The given number is Negative");
15     getch();
16 }

```

**nested if-else স্টেটমেন্টঃ** একটি if-else স্টেটমেন্টের মধ্যে যখন অন্য এক বা একাধিক if-else স্টেটমেন্ট ব্যবহৃত হয় তখন তাকে nested if-else স্টেটমেন্ট বলে। nested if-else স্টেটমেন্ট ব্যবহারের ফরম্যাট হলো-

```

1  if(Condition1)
2  {
3      if(Condition2)
4      {
5          statement(s)
6      }
7  }

```

লিপ ইয়ার( অধিবর্ষ) নির্ণয়ের প্রোগ্রাম।

```

1 #include<stdio.h>
2 #include<conio.h>
3
4 main()
5 {
6     int y;
7     printf("Enter a year");
8     scanf("%d",&y);
9     if(y%4==0)
10    {
11        if(y%100!=0)
12            printf("The given year is a leap year");
13    }
14    else if(y%400)
15        printf("The given year is a leap year");
16    else
17        printf("The given year is not a leap year");
18    getch();
19 }

```

**নোটঃ** প্রতিটি কন্ট্রোল স্টেটমেন্টের জন্য একাধিক স্টেটমেন্ট ব্যবহৃত হলে স্টেটমেন্টসমূহ কার্লিব্রেস {} এর মধ্যে লিখতে হয়। আর যদি একটি স্টেটমেন্ট ব্যবহৃত হয় তাহলে কার্লিব্রেস {} এর মধ্যে না লিখলেও সমস্যা নেই। উপরের উদাহরণে প্রতিটি কন্ট্রোল স্টেটমেন্টের জন্য একটি স্টেটমেন্ট লেখা আছে তাই কার্লিব্রেস {} এর মধ্যে লেখা হয় নি।

**switch স্টেটমেন্টঃ** প্রোগ্রামে একাধিক কন্ডিশন লেখার ক্ষেত্রে else if স্টেটমেন্ট ব্যবহার করা হয়। অনেক বড় প্রোগ্রামের ক্ষেত্রে else if স্টেটমেন্ট ব্যবহার করে প্রোগ্রাম লেখা কষ্টকর। তাই এই ক্ষেত্রে else if স্টেটমেন্ট এর পরিবর্তে switch স্টেটমেন্ট ব্যবহার করে প্রোগ্রাম লেখা সহজ। switch স্টেটমেন্ট ব্যবহারের ফরম্যাট হলো-

```

1 switch(expression)
2 {
3     case constant_1:
4         statements_for_constant_1;
5         break;
6
7     case constant_2:
8         statements_for_constant_2;
9         break;
10    -----
11    -----
12    case constant_n:
13        statements_for_constant_n;
14        break;
15
16    default: // Default case is optional
17        default_statements;
18        break;
19 }

```

switch স্টেটমেন্টের expression এর মানের সাথে যে case constant টি ম্যাচিং করবে সেই case এর কোড ব্লকটি নির্বাহ হবে। যদি কোন case constant এর সাথে না মিলে তাহলে default কোড ব্লকটি নির্বাহ হবে।

**switch স্টেটমেন্ট সম্পর্কে গুরুত্বপূর্ণ তথ্যঃ**

- ১। switch স্টেটমেন্টে case constant গুলোর মান char('A', 'B'../ 'a','b'.../'+', '- ... ইত্যাদি) অথবা int(১,২,৩...) টাইপ হতে হবে। case constant এর মান char টাইপের হলে তা সিঙ্গেল কোটেশনের( ' ' ) মধ্যে লিখতে হয়।
- ২। একটি switch স্টেটমেন্টের case constant গুলোর মান অবশ্যই ভিন্ন ভিন্ন হতে হবে।
- ৩। case constant গুলোর শেষে কোলন ( : ) থাকতে হবে।
- ৪। প্রতিটি case এর শেষে break কিওয়ার্ড থাকতে হবে।
- ৫। একাধিক case এর জন্য একই কোড ব্লক হতে পারবে।
- ৬। একটি মাত্র ডিফল্ট case থাকতে পারে।

**উদাহরণ-১ঃ** দুইটি সংখ্যা ইনপুট নিয়ে switch স্টেটমেন্টের সাহায্যে যোগ অথবা বিয়োগ অথবা গুন অথবা ভাগ করার একটি প্রোগ্রাম।

```

1  #include<stdio.h>
2  #include<conio.h>
3
4  main()
5  {
6      char ope;
7      int a,b;
8      printf("Enter Operator Either + or - or * or /");
9      scanf("%c",&ope);
10     printf("Enter Two numbers");
11     scanf("%d %d",&a,&b);
12
13     switch(ope)
14     {
15     case '+':
16         printf("Sum=%d",a+b);
17         break;
18     case '-':
19         printf("Subtraction=%d",a-b);
20         break;
21     case '*':
22         printf("Multiplication=%d",a*b);
23         break;
24     case '/':
25         printf("Quotient=%d",a/b);
26         break;
27     default:
28         printf("Your Operation is not matched");
29         break;
30     }
31     getch();
32 }

```

পঞ্চম অধ্যায় পাঠ-১৫ কন্ডিশনাল কন্ট্রোল স্টেটমেন্ট সম্পর্কিত প্রোগ্রাম সমূহ।

১। কোন সংখ্যা জোড়/বিজোড় নির্ণয়ের জন্য সি প্রোগ্রামিং ভাষায় একটি প্রোগ্রাম।

```

1 #include<stdio.h>
2 #include<conio.h>
3 main()
4 {
5     int n;
6     printf("Enter a number: ");
7     scanf("%d", &n);
8
9     if (n%2==0)
10        printf("\nThe number %d is even.",n);
11    else
12        printf("\nThe number %d is odd.",n);
13    getch();
14 }

```

২। কোন সংখ্যা ধনাত্মক/ঋণাত্মক নির্ণয়ের জন্য সি প্রোগ্রামিং ভাষায় একটি প্রোগ্রাম।

```

1 #include<stdio.h>
2 #include<conio.h>
3 main()
4 {
5     int n;
6     printf("Enter a number: ");
7     scanf("%d", &n);
8
9     if (n>=0)
10        printf("\nThe number %d is positive.",n);
11    else
12        printf("\nThe number %d is Negative.",n);
13    getch();
14 }

```

৩। কোন একটি সাল লিপ ইয়ার(Leap Year ) নির্ণয়ের জন্য সি প্রোগ্রামিং ভাষায় একটি প্রোগ্রাম।

অধিবর্ষ বা লিপ ইয়ার হচ্ছে একটি বিশেষ বছর, যাতে সাধারণ বছরের তুলনায় একটি দিন বেশি থাকে। জ্যোতির্বিজ্ঞানিক বছর বা পৃথিবী যে সময়ে সূর্যের চারপাশে একবার ঘুরে আসে তার সময়কাল হচ্ছে প্রায় ৩৬৫ দিন ৫ ঘণ্টা ৪৮ মিনিট ৪৭ সেকেন্ড, অথচ প্রচলিত গ্রেগরীয় বর্ষপঞ্জি মতে বছর হিসাব করা হয় ৩৬৫ দিনে। এভাবে প্রতিবছর প্রায় ছয় ঘণ্টা সময় গোনার বাইরে [থেকে যায় ও চার](#) বছরে সেটা প্রায় এক দিনের সমান হয়। এই ঘাটতি পুষিয়ে নেয়ার জন্য প্রতি চার বছর পরপর ৩৬৬ দিনে বছর হিসাব করা হয়। গ্রেগরীয় বর্ষপঞ্জি মতে, প্রতি চার বছরে একবার ফেব্রুয়ারি মাসে ও বাংলা সনমতে ফাল্গুন মাসে এই অতিরিক্ত ১ দিন যোগ হয়। যেমন: ২০১২ একটি অধিবর্ষ ও এর ফেব্রুয়ারি মাস হয়েছে ২৯ দিনে।



```

Leap Year
1 #include<stdio.h>
2 #include<conio.h>
3
4 main()
5 {
6     int y;
7     printf("Enter a year:");
8     scanf("%d",&y);
9
10    if ((y%400==0)||((y%100!=0)&&(y%4==0)))
11        printf("%d is a Leap year", y);
12    else
13        printf("%d is not a Leap year", y);
14    getch();
15 }

```

৪। দুটি সংখ্যার মধ্যে বড় সংখ্যা নির্ণয়ের জন্য সি প্রোগ্রামিং ভাষায় একটি প্রোগ্রাম।

```

1 #include<stdio.h>
2 #include<conio.h>
3 main()
4 {
5     int a, b;
6     printf("Enter 1st value :");
7     scanf("%d",&a);
8     printf("Enter 2nd value :");
9     scanf("%d",&b);
10
11    if(a>b)
12        printf("Largest Number is : %d", a);
13    else
14        printf("Largest Number is: %d", b);
15    getch();
16 }

```

৫। দুটি পূর্ণ সংখ্যার ল.সা.গু. নির্ণয়ের জন্য সি প্রোগ্রামিং ভাষায় একটি প্রোগ্রাম।

ল.সা.গু শব্দের পূর্ণরূপ হল লঘিষ্ঠ সাধারণ গুণিতক। একটি সংখ্যা কোন সংখ্যা দ্বারা বিভাজ্য হলে প্রথম সংখ্যাটিকে দ্বিতীয় সংখ্যার গুণিতক বলে আর দ্বিতীয় সংখ্যাটিকে প্রথম সংখ্যার গুণনীয়ক বলে। যেমনঃ ১২ কে ৬ দ্বারা ভাগ করলে নিঃশেষে বিভাজ্য হবে সেক্ষেত্রে ১২ সংখ্যাটি ৬ এর গুণিতক আর ১২ এর গুণনীয়ক হচ্ছে ৬।

```

1 #include<stdio.h>
2 #include<conio.h>
3 main()
4 {
5     int a,b,l;
6     printf("Enter the two numbers: ");
7     scanf("%d %d",&a,&b);
8     if(a>b)
9         l=a;
10    else
11        l=b;
12    again:
13    if(l%a==0 && l%b==0)
14        printf("LCM of %d and %d is %d",a,b,l);
15    else
16    {
17        l=l+1;
18        goto again;
19    }
20    getch();
21 }

```

৬। দুটি পূর্ণ সংখ্যার গ.সা.গু. নির্ণয়ের জন্য সি প্রোগ্রামিং ভাষায় একটি প্রোগ্রাম।

গ.সা.গু. শব্দের পূর্ণরূপ হল গরিষ্ঠ সাধারণ গুণনীয়ক। একটি সংখ্যা কোন সংখ্যা দ্বারা বিভাজ্য হলে প্রথম সংখ্যাটিকে দ্বিতীয় সংখ্যার গুণিতক বলে আর দ্বিতীয় সংখ্যাটিকে প্রথম সংখ্যার গুণনীয়ক বলে। যেমনঃ ১২ কে ৬ দ্বারা ভাগ করলে নিঃশেষে বিভাজ্য হবে সেক্ষেত্রে ১২ সংখ্যাটি ৬ এর গুণিতক আর ১২ এর গুণনীয়ক হচ্ছে ৬।

```
1 #include<stdio.h>
2 #include<conio.h>
3 main()
4 {
5     int a,b,s;
6     printf("Enter the two numbers: ");
7     scanf("%d %d",&a,&b);
8
9     if(a<b)
10        s=a;
11    else
12        s=b;
13
14    again:
15    if(a%s==0 && b%s==0)
16        printf("GCD of %d and %d is %d",a,b,s);
17    else
18    {
19        s=s-1;
20        goto again;
21    }
22    getch();
23 }
```

৭। তিনটি সংখ্যার মধ্যে সবচেয়ে ছোট সংখ্যা নির্ণয়ের জন্য সি প্রোগ্রামিং ভাষায় একটি প্রোগ্রাম।

```
1 #include<stdio.h>
2 #include<conio.h>
3 main()
4 {
5     int a,b,c;
6     printf("Enter three integer numbers:");
7     scanf("%d %d %d", &a, &b, &c);
8
9     if(a<b)
10    {
11        if(a<c)
12            printf("\n Smallest number is: %d", a);
13        else
14            printf("\n Smallest number is: %d", c);
15    }
16    else
17    {
18        if(b<c)
19            printf("\n Smallest number is: %d", b);
20        else
21            printf("\n Smallest number is: %d", c);
22    }
23    getch();
24 }
```

৮। তিনটি সংখ্যার মধ্যে মাঝারি সংখ্যা নির্ণয়ের জন্য সি প্রোগ্রামিং ভাষায় একটি প্রোগ্রাম।

```
Middle number among three numbers
1 #include<stdio.h>
2 #include<conio.h>
3 main()
4 {
5     int a,b,c;
6     printf("Enter three integer numbers:");
7     scanf("%d %d %d", &a, &b, &c);
8
9     if(a>b)
10    {
11        if(b>c)
12            printf("%d is middle one",b);
13        else if(c>a)
14            printf("%d is middle one",a);
15        else
16            printf("%d is middle one",c);
17    }
18    else
19    {
20        if(b<c)
21            printf("%d is middle one",b);
22        else if(c<a)
23            printf("%d is middle one",a);
24        else
25            printf("%d is middle one",c);
26    }
27    getch();
28 }
```

৯। তিনটি সংখ্যার মধ্যে সবচেয়ে বড় সংখ্যা নির্ণয়ের জন্য সি প্রোগ্রামিং ভাষায় একটি প্রোগ্রাম।

```
The Largest Number Among Three Numbers
1 #include<stdio.h>
2 #include<conio.h>
3 main()
4 {
5     int a,b,c;
6     printf("Enter three integer numbers:");
7     scanf("%d %d %d", &a, &b, &c);
8
9     if(a>b)
10    {
11        if(a>c)
12            printf("\n Largest number is: %d", a);
13        else
14            printf("\n Largest number is: %d", c);
15    }
16    else
17    {
18        if(b>c)
19            printf("\n Largest number is: %d", b);
20        else
21            printf("\n Largest number is: %d", c);
22    }
23    getch();
24 }
```

পঞ্চম অধ্যায় পাঠ-১৬ ‘সি’ প্রোগ্রামিং ভাষায় লুপ কন্ট্রোল স্টেটমেন্ট।

লুপঃ প্রোগ্রামের এক বা একাধিক স্টেটমেন্ট নির্দিষ্ট সংখ্যক বার পুনরাবৃত্তি করাকে লুপ বা লুপিং বলে।

লুপের প্রকারভেদ:

- সসীম লুপ – যদি কোন লুপ নির্দিষ্ট সংখ্যক বার পুনরাবৃত্তি হয়, তখন তাকে সসীম লুপ বলে।

- **অসীম লুপ** – যদি কোন লুপ অনবরত পুনরাবৃত্তি হতে থাকে, অর্থাৎ কখনো শেষ না হয় তবে তাকে অসীম লুপ বলে।
- **মধ্যবর্তী লুপ** – একটি লুপের মধ্যে যদি অপর একটি লুপ থাকে তাহলে তাকে মধ্যবর্তী বা ন্যেস্টেড লুপ বলে।

**লুপ কন্ট্রোল স্টেটমেন্টঃ** প্রোগ্রামের এক বা একাধিক স্টেটমেন্ট নির্দিষ্ট সংখ্যক বার পুনরাবৃত্তি করার জন্য যে কন্ট্রোল স্টেটমেন্ট ব্যবহৃত হয় তাকে লুপ কন্ট্রোল স্টেটমেন্ট বলে।

**লুপ কন্ট্রোল স্টেটমেন্ট সমূহঃ**

- ১। for লুপ স্টেটমেন্ট
- ২। while লুপ স্টেটমেন্ট
- ৩। do...while লুপ স্টেটমেন্ট

প্রত্যেক লুপ কন্ট্রোল স্টেটমেন্টের দুটি অংশ থাকে। যথা-

- ১। লুপ ডিক্লারেশন
- ২। লুপ বডি

**লুপ ডিক্লারেশন তিনটি প্রধান অংশ-**

- ১। **Initialization Statement-** এই অংশে ভেরিয়েবলের প্রাথমিক মান নির্ধারণ করা হয়।
- ২। **Test Expression** – এই অংশে শর্ত লেখা হয়। শর্ত মিথ্যা না হওয়া পর্যন্ত লুপ বডি পুনরাবৃত্তি হয়।
- ৩। **Update Statement-** প্রতিবার লুপ বডি পুনরাবৃত্তির পর ভেরিয়েবলের মান হ্রাস/বৃদ্ধি নির্ধারণ করা হয় এই অংশে।

**লুপ বডি-** যে স্টেটমেন্টগুলো পুনরাবৃত্তি হবে তা { } এর মধ্যে থাকে, যা লুপ বডি হিসেবে বিবেচিত হয়।

লুপ স্টেটমেন্টের লুপ বডি এবং টেস্ট কন্ডিশনের অবস্থানের ভিত্তিতে লুপ স্টেটমেন্টসমূহকে দুই ভাগে ভাগ করা যায়। যথা-

- ১। এন্ট্রি কন্ট্রোল লুপ স্টেটমেন্ট
- ২। এক্সিট কন্ট্রোল লুপ স্টেটমেন্ট

**এন্ট্রি কন্ট্রোল লুপ স্টেটমেন্টঃ** লুপ বডি নির্বাহের পূর্বে টেস্ট কন্ডিশন যাচাই করা হয়। কন্ডিশন সত্য হলেই কেবলমাত্র লুপ বডি নির্বাহ হয়। উদাহরণঃ for লুপ স্টেটমেন্ট, while loop স্টেটমেন্ট।

**এক্সিট কন্ট্রোল লুপ স্টেটমেন্টঃ** প্রথমবার টেস্ট কন্ডিশন যাচাই না করেই লুপ বডি নির্বাহ হয়। তারপর কন্ডিশন যাচাই করা হয়। কন্ডিশন সত্য হলে লুপ বডি নির্বাহ হয়। উদাহরণঃ do-while loop স্টেটমেন্ট।

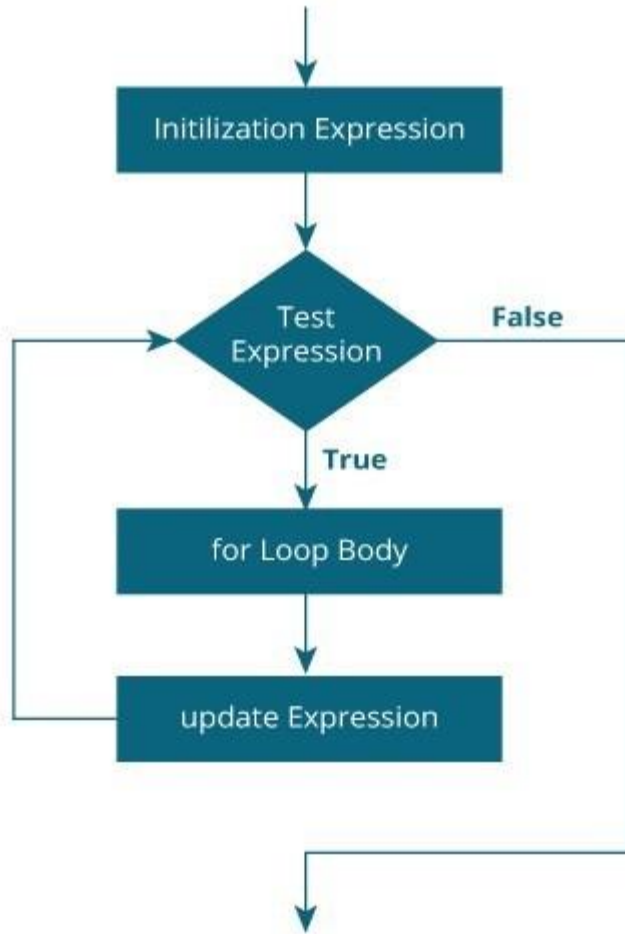
**for লুপ স্টেটমেন্টঃ** 'সি' প্রোগ্রামে শর্ত সাপেক্ষে এক বা একাধিক স্টেটমেন্ট নির্দিষ্ট সংখ্যকবার নির্বাহ করতে for লুপ স্টেটমেন্ট ব্যবহার করা হয়। লুপ বডির কোড নির্বাহের পূর্বে কন্ডিশন চেক করে। লুপ কতবার নির্বাহ হবে তা জানা থাকলেই কেবলমাত্র for লুপ ব্যবহার করা যায়। নিম্নে for লুপ স্টেটমেন্টের ফরম্যাট দেওয়া হলো-

```
1 for (initializationStatement; testExpression; updateStatement)
2 {
3     // codes
4 }
```

**for loop যেভাবে কাজ করে-**

- ১। প্রথমে Initialization Statement নির্বাহ হয়। লুপ স্টেটমেন্টে Initialization Statement কেবলমাত্র একবার নির্বাহ হয়।
- ২। তারপর Test Expression চেক করে। যদি Test Expression সত্য হয় তাহলে লুপ বডির কোড নির্বাহ হয় এবং Update Expression এর মান আপডেট হয়। Test Expression মিথ্যা না হওয়া পর্যন্ত ধাপ-২ পুনরাবৃত্তি হতে থাকে।
- ৩। যদি Test Expression মিথ্যা হয়, তাহলে প্রোগ্রাম নির্বাহ লুপ থেকে বের হয়ে আসে।

**for লুপের ফ্লোচার্টঃ**



এবারে আমরা for loop ব্যবহার করে Hello World লেখাটি ৫ বার প্রিন্ট করার প্রোগ্রাম তৈরি করব।

```
1 #include<stdio.h>
2 #include<conio.h>
3
4 main()
5 {
6     int i;
7     for(i=1; i<=5; i++)
8     {
9         printf("Hello World\n");
10    }
11    getch();
12 }
```

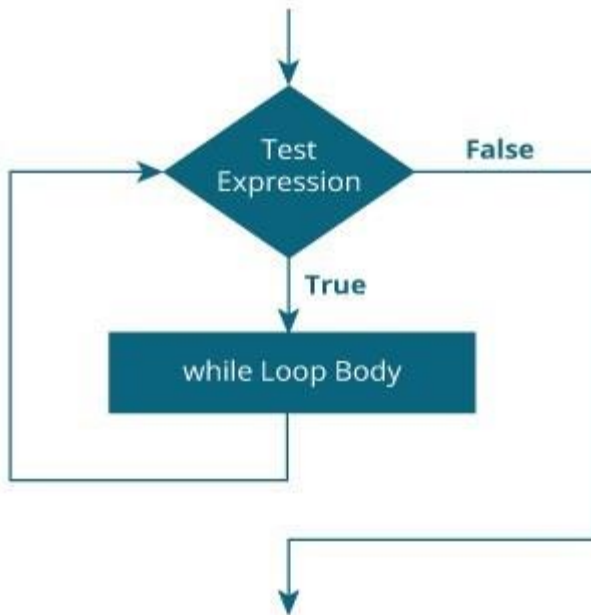
**while loop স্টেটমেন্টঃ** 'সি' প্রোগ্রামে শর্ত সাপেক্ষে এক বা একাধিক স্টেটমেন্ট নির্দিষ্ট সংখ্যকবার নির্বাহ করতে while loop স্টেটমেন্ট ব্যবহার করা হয়। লুপ বডির কোড নির্বাহের পূর্বে কন্ডিশন চেক করে while loop কে for loop এর বিকল্প হিসাবে ব্যবহার করা যায়। লুপ কতবার নির্বাহ হবে তা অজানা থাকলে while লুপ ব্যবহার করা হয়। while loop স্টেটমেন্টের ফরম্যাট হলো-

```
1 while (testExpression)
2 {
3     //codes
4 }
```

**while loop যেভাবে কাজ করে-**

- ১। প্রথমে Test Expression চেক করে। যদি Test Expression সত্য হয় তাহলে লুপ বডির কোড নির্বাহ হয় এবং পুনরায় Test Expression চেক করে। Test Expression মিথ্যা না হওয়া পর্যন্ত এই প্রক্রিয়া পুনরাবৃত্তি হতে থাকে।
- ২। যদি Test Expression মিথ্যা হয়, তাহলে প্রোগ্রাম নির্বাহ লুপ থেকে বের হয়ে আসে।

**while লুপের ফ্লোচার্টঃ**



for লুপের মত করে While লুপের গঠনঃ

```
1 initializationStatement;
2 while(testExpression)
3 {
4     // codes;
5
6     updateStatement;
7 }
```

এবারে আমরা While loop ব্যবহার করে Hello World লেখাটি ৫ বার প্রিন্ট করার প্রোগ্রাম তৈরি করব।

```
1 #include<stdio.h>
2 #include<conio.h>
3
4 main()
5 {
6     int i;
7     i=1;
8     while(i<=5)
9     {
10         printf("Hello World\n");
11         i++;
12     }
13     getch();
14 }
```

**do-while loop স্টেটমেন্ট:** ‘সি’ প্রোগ্রামে শর্ত সাপেক্ষে এক বা একাধিক স্টেটমেন্ট নির্দিষ্ট সংখ্যকবার নির্বাহ করতে do-while loop স্টেটমেন্ট ব্যবহার করা হয়। do-while লুপের কন্ডিশন যাচাই না করে লুপ বডি অন্ততপক্ষে একবার নির্বাহ হয়। কারণ এখানে কন্ডিশন পরে যাচাই হয়। do-while loop টি do loop নামেও পরিচিত। তবে প্রোগ্রামে for এবং while লুপের চেয়ে do-while loop লুপ কম ব্যবহৃত হয়। do-while loop এর গঠন হচ্ছে-

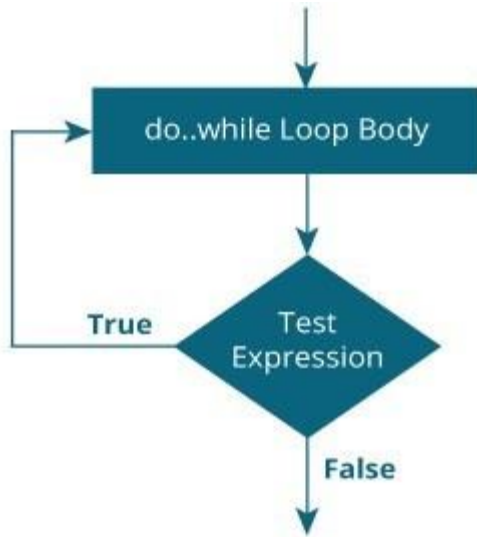
```
1 do
2 {
3     // codes
4 }
5 while (testExpression);
```

**do-while loop যেভাবে কাজ করে-**

- ১। প্রথমে লুপ বডির কোড একবার নির্বাহ হয়।
- ২। তারপর Test Expression চেক করে। যদি Test Expression সত্য হয় তাহলে লুপ বডির কোড নির্বাহ হয় এবং পুনরায় Test Expression চেক করে। Test Expression মিথ্যা না হওয়া পর্যন্ত এই প্রক্রিয়া পুনরাবৃত্তি হতে থাকে।
- ৩। যদি Test Expression মিথ্যা হয়, তাহলে প্রোগ্রাম নির্বাহ লুপ থেকে বের হয়ে আসে।

**do-while লুপের ফ্লোচার্টঃ**





for লুপের মত করে do-while লুপের গঠনঃ

```

1 initializationStatement;
2 do
3 {
4     // codes;
5
6     updateStatement;
7 }while(testExpression);
  
```

এবারে আমরা do-while loop ব্যবহার করে Hello World লেখাটি ৫ বার প্রিন্ট করার প্রোগ্রাম তৈরি করব।

```

1 #include<stdio.h>
2 #include<conio.h>
3
4 main()
5 {
6     int i;
7     i=1;
8     do
9     {
10         printf("Hello World\n");
11         i++;
12     }while(i<=5);
13     getch();
14 }
  
```

পঞ্চম অধ্যায় পাঠ-১৭ লুপ কন্ট্রোল স্টেটমেন্ট সম্পর্কিত প্রোগ্রাম সমূহ।

১।১ থেকে ১০ পর্যন্ত সংখ্যা দেখানোর প্রোগ্রাম। অথবা

১ ২ ৩ ৪ ৫ ৬ ৭ ৮ ৯ ১০ ধারাটি তৈরির প্রোগ্রাম।

```
1  /* for loop ব্যবহার করে প্রোগ্রাম */
2  #include<stdio.h>
3  #include<conio.h>
4  main()
5  {
6      int i;
7      for(i=1; i<=10; i++)
8      {
9          printf("%d\t", i);
10     }
11     getch();
12 }
13
14 /* while loop ব্যবহার করে প্রোগ্রাম */
15 #include<stdio.h>
16 #include<conio.h>
17 main()
18 {
19     int i;
20     i=1;
21     while(i<=10)
22     {
23         printf("%d\t", i);
24         i++;
25     }
26     getch();
27 }
28
29
30 /* do while loop ব্যবহার করে প্রোগ্রাম */
31 #include<stdio.h>
32 #include<conio.h>
33 main()
34 {
35     int i;
36     i=1;
37     do
38     {
39         printf("%d\t", i);
40         i++;
41     } while(i<=10);
42     getch();
43 }
```

১ ২ ৩ ৪ ৫ ----- n ধারাটি তৈরির প্রোগ্রাম।

```
1 /* for loop ব্যবহার করে প্রোগ্রাম */
2 #include<stdio.h>
3 #include<conio.h>
4 main()
5 {
6     int i, n;
7     printf("Enter value of n: ");
8     scanf("%d",&n);
9     for(i=1;i<=n; i=i+1)
10     {
11         printf("%d\t",i);
12     }
13     getch();
14 }
15
16 /* while loop ব্যবহার করে প্রোগ্রাম */
17 #include<stdio.h>
18 #include<conio.h>
19 main()
20 {
21     int i, n;
22     printf("Enter value of n: ");
23     scanf("%d",&n);
24     i=1;
25     while(i<=n)
26     {
27         printf("%d\t",i);
28         i=i+1;
29     }
30     getch();
31 }
32
33 /* do while loop ব্যবহার করে প্রোগ্রাম */
34 #include<stdio.h>
35 #include<conio.h>
36 main()
37 {
38     int i, n;
39     printf("Enter value of n: ");
40     scanf("%d",&n);
41     i=1;
42     do
43     {
44         printf("%d\t",i);
45         i=i+1;
46     } while(i<=n);
47     getch();
48 }
```

৩। ১ থেকে ১০ এর মধ্যে অবস্থিত বিজোড় সংখ্যাগুলো দেখানোর প্রোগ্রাম। অথবা

১ ৩ ৫ ৭ ৯ ধারাটি তৈরির প্রোগ্রাম।

```
1  /* for loop ব্যবহার করে প্রোগ্রাম */
2  #include<stdio.h>
3  #include<conio.h>
4  main()
5  {
6      int i;
7      for(i=1;i<=10; i=i+2)
8      {
9          printf("%d\t",i);
10     }
11     getch();
12 }
13
14
15
16 /* while loop ব্যবহার করে প্রোগ্রাম */
17 #include<stdio.h>
18 #include<conio.h>
19 main()
20 {
21     int i;
22     i=1;
23     while(i<=10)
24     {
25         printf("%d\t",i);
26         i=i+2;
27     }
28     getch();
29 }
30
31
32 /* do while loop ব্যবহার করে প্রোগ্রাম */
33 #include<stdio.h>
34 #include<conio.h>
35 main()
36 {
37     int i;
38     i=1;
39     do
40     {
41         printf("%d\t",i);
42         i=i+2;
43     } while(i<=10);
44     getch();
45 }
```

৪। ১ থেকে n এর মধ্যে অবাস্তব বিজোড় সংখ্যাগুলো দেখানোর প্রোগ্রাম। অথবা

১ ৩ ৫ ৭ ---n ধারাটি তৈরির প্রোগ্রাম।

```
1  /* for loop ব্যবহার করে প্রোগ্রাম */
2  #include<stdio.h>
3  #include<conio.h>
4  main()
5  {
6      int i,n;
7      printf("Enter Value of n: ");
8      scanf("%d",&n);
9      for(i=1;i<=n; i=i+2)
10     {
11         printf("%d\t",i);
12     }
13     getch();
14 }
15
16
17
18 /* while loop ব্যবহার করে প্রোগ্রাম */
19 #include<stdio.h>
20 #include<conio.h>
21 main()
22 {
23     int i,n;
24     printf("Enter Value of n: ");
25     scanf("%d",&n);
26     i=1;
27     while(i<=10)
28     {
29         printf("%d\t",i);
30         i=i+2;
31     }
32     getch();
33 }
34
35
36 /* do while loop ব্যবহার করে প্রোগ্রাম */
37 #include<stdio.h>
38 #include<conio.h>
39 main()
40 {
41     int i,n;
42     printf("Enter Value of n: ");
43     scanf("%d",&n);
44     i=1;
45     do
46     {
47         printf("%d\t",i);
48         i=i+2;
49     } while(i<=10);
50     getch();
51 }
```

```
1  /* for loop ব্যবহার করে প্রোগ্রাম */
2  #include<stdio.h>
3  #include<conio.h>
4  main()
5  {
6      int i;
7      for(i=2;i<=10; i=i+2)
8      {
9          printf("%d\t ",i);
10     }
11     getch();
12 }
13
14
15 /* while loop ব্যবহার করে প্রোগ্রাম */
16 #include<stdio.h>
17 #include<conio.h>
18 main()
19 {
20     int i;
21     i=2;
22     while(i<=10)
23     {
24         printf("%d\t ",i);
25         i=i+2;
26     }
27     getch();
28 }
29
30
31 /* do while loop ব্যবহার করে প্রোগ্রাম */
32 #include<stdio.h>
33 #include<conio.h>
34 main()
35 {
36     int i;
37     i=2;
38     do
39     {
40         printf("%d\t ",i);
41         i=i+2;
42     } while(i<=10);
43     getch();
44 }
```

৬।১ থেকে n এর মধ্যে অবস্থিত জোড় সংখ্যাগুলো দেখানোর প্রোগ্রাম। অথবা

২ ৪ ৬ --- n ধারাটি তৈরির প্রোগ্রাম।

```
1  /* for loop ব্যবহার করে প্রোগ্রাম */
2  #include<stdio.h>
3  #include<conio.h>
4  main()
5  {
6      int i,n;
7      printf("Enter Value of n: ");
8      scanf("%d",&n);
9      for(i=2;i<=n; i=i+2)
10     {
11         printf("%d\t",i);
12     }
13     getch();
14 }
15
16 /* while loop ব্যবহার করে প্রোগ্রাম */
17 #include<stdio.h>
18 #include<conio.h>
19 main()
20 {
21     int i,n;
22     printf("Enter Value of n: ");
23     scanf("%d",&n);
24     i=2;
25     while(i<=n)
26     {
27         printf("%d\t",i);
28         i=i+2;
29     }
30     getch();
31 }
32
33
34
35 /* do while loop ব্যবহার করে প্রোগ্রাম */
36 #include<stdio.h>
37 #include<conio.h>
38 main()
39 {
40     int i,n;
41     printf("Enter Value of n: ");
42     scanf("%d",&n);
43     i=2;
44     do
45     {
46         printf("%d\t",i);
47         i=i+2;
48     } while(i<=n);
49     getch();
50 }
51
```

৭। ১ থেকে ১০০ পর্যন্ত সংখ্যা গুলোর যোগফল দেখানোর প্রোগ্রাম। অথবা

১+২+৩+৪+-----+১০০ দ্বারা যোগফল দেখানোর প্রোগ্রাম।

```
1  /* for loop ব্যবহার করে প্রোগ্রাম */
2  #include<stdio.h>
3  #include<conio.h>
4  main()
5  {
6      int i,s=0;
7      for(i=1;i<=100; i=i+1)
8      {
9          s=s+i;
10     }
11     printf("Sum=%d ",s);
12     getch();
13 }
14
15
16
17 /* while loop ব্যবহার করে প্রোগ্রাম */
18 #include<stdio.h>
19 #include<conio.h>
20 main()
21 {
22     int i,s=0;
23     i=1;
24     while(i<=100)
25     {
26         s=s+i;
27         i=i+1;
28     }
29     printf("Sum=%d ",s);
30     getch();
31 }
32
33
34
35 /* do while loop ব্যবহার করে প্রোগ্রাম */
36 #include<stdio.h>
37 #include<conio.h>
38 main()
39 {
40     int i,s=0;
41     i=1;
42     do
43     {
44         s=s+i;
45         i=i+1;
46     } while(i<=100);
47     printf("Sum=%d ",s);
48     getch();
49 }
```



৮।১ থেকে n পর্যন্ত সংখ্যা গুলোর যোগফল দেখানোর প্রোগ্রাম। অথবা

1+2+3+8+-----+n ধারার যোগফল দেখানোর প্রোগ্রাম।

```
1  /* for loop ব্যবহার করে প্রোগ্রাম */
2  #include<stdio.h>
3  #include<conio.h>
4  main()
5  {
6      int i,n,s=0;
7      printf("Enter Value of n: ");
8      scanf("%d",&n);
9      for(i=1;i<=n; i=i+1)
10     {
11         s=s+i;
12     }
13     printf("Sum=%d ",s);
14     getch();
15 }
16
17
18
19 /* while loop ব্যবহার করে প্রোগ্রাম */
20 #include<stdio.h>
21 #include<conio.h>
22 main()
23 {
24     int i,n,s=0;
25     printf("Enter Value of n: ");
26     scanf("%d",&n);
27     i=1;
28     while(i<=n)
29     {
30         s=s+i;
31         i=i+1;
32     }
33     printf("Sum=%d ",s);
34     getch();
35 }
36
37
38
39 /* do while loop ব্যবহার করে প্রোগ্রাম */
40 #include<stdio.h>
41 #include<conio.h>
42 main()
43 {
44     int i,n,s=0;
45     printf("Enter Value of n: ");
46     scanf("%d",&n);
47     i=1;
48     do
49     {
50         s=s+i;
51         i=i+1;
52     } while(i<=n);
53     printf("Sum=%d ",s);
54     getch();
55 }
```

৯।১ থেকে ১০০ এর মধ্যে অবাস্তব ভাজোড় সংখ্যা গুলোর যোগফল দেখানোর প্রোগ্রাম। অথবা

১+৩+৫+-----+১০০ ধারার যোগফল দেখানোর প্রোগ্রাম।

```
1  /* for loop ব্যবহার করে প্রোগ্রাম */
2  #include<stdio.h>
3  #include<conio.h>
4  main()
5  {
6  int i,s=0;
7  for(i=1; i<=100; i=i+2)
8  {
9      s=s+i;
10 }
11 printf("Sum=%d ",s);
12 getch();
13 }
14
15
16 /* while loop ব্যবহার করে প্রোগ্রাম */
17 #include<stdio.h>
18 #include<conio.h>
19 main()
20 {
21 int i,s=0;
22 i=1;
23 while(i<=100)
24 {
25     s=s+i;
26     i=i+2;
27 }
28 printf("Sum=%d ",s);
29 getch();
30 }
31
32
33 /* do while loop ব্যবহার করে প্রোগ্রাম */
34 #include<stdio.h>
35 #include<conio.h>
36 main()
37 {
38 int i,s=0;
39 i=1;
40 do
41 {
42     s=s+i;
43     i=i+2;
44 } while(i<=100);
45 printf("Sum=%d ",s);
46 getch();
47 }
```

১০। ১ থেকে ১০০ এর মধ্যে অবাস্তব জোড় সংখ্যা গুলোর যোগফল দেখানোর প্রোগ্রাম। অথবা

২+৪+৬+-----+১০০ ধারার যোগফল দেখানোর প্রোগ্রাম।

```
1  /* for loop ব্যবহার করে প্রোগ্রাম */
2  #include<stdio.h>
3  #include<conio.h>
4  main()
5  {
6      int i,s=0;
7      for(i=2;i<=100; i=i+2)
8      {
9          s=s+i;
10     }
11     printf("Sum=%d ",s);
12     getch();
13 }
14
15
16
17 /* while loop ব্যবহার করে প্রোগ্রাম */
18 #include<stdio.h>
19 #include<conio.h>
20 main()
21 {
22     int i,s=0;
23     i=2;
24     while(i<=100)
25     {
26         s=s+i;
27         i=i+2;
28     }
29     printf("Sum=%d ",s);
30     getch();
31 }
32
33
34
35 /* do while loop ব্যবহার করে প্রোগ্রাম */
36 #include<stdio.h>
37 #include<conio.h>
38 main()
39 {
40     int i,s=0;
41     i=2;
42     do
43     {
44         s=s+i;
45         i=i+2;
46     } while(i<=100);
47     printf("Sum=%d ",s);
48     getch();
49 }
```

১১। কোন একটি ধনাত্মক সংখ্যার ফ্যাক্টোরিয়াল নির্ণয়ের প্রোগ্রাম।

```
1  #include <stdio.h>
2  main()
3  {
4      int n, i;
5      long int fact = 1;
6      printf("Enter an integer: ");
7      scanf("%d",&n);
8      if(n<0)
9          printf("Error! Factorial of a negative number doesn't exist.");
10     else
11     {
12         for (i=1; i<=n; i++)
13         {
14             fact=fact * i;
15         }
16         printf("Factorial= %ld", fact);
17     }
18 }
```

# পঞ্চম অধ্যায় পাঠ-১৮ continue স্টেটমেন্ট, break স্টেটমেন্ট ও goto স্টেটমেন্ট।

**continue স্টেটমেন্ট:** 'সি' প্রোগ্রামে লুপ কন্ট্রোল স্টেটমেন্টের লুপ বডির এক বা একাধিক স্টেটমেন্ট নির্বাহ না হয়ে পুনরায় প্রথম থেকে নির্বাহের জন্য continue স্টেটমেন্ট ব্যবহৃত হয়। continue স্টেটমেন্ট শর্তযুক্ত এবং শর্তবিহীন উভয় ভাবে ব্যবহার করা যায়। তবে শর্তবিহীন continue স্টেটমেন্ট অসীম লুপের সৃষ্টি করে। continue স্টেটমেন্টের ফরম্যাটঃ

```
1 continue;
```

continue স্টেটমেন্ট যেভাবে কাজ করে-



for লুপে continue স্টেটমেন্ট ব্যবহার করে ১ থেকে ১০ এর মধ্যে অবস্থিত বিজোড় সংখ্যাগুলো দেখানোর প্রোগ্রাম।

```
1 /* for loop ব্যবহার করে প্রোগ্রাম */
2 #include<stdio.h>
3 #include<conio.h>
4 main()
5 {
6     int i;
7     for(i=1; i<=10; i=i+1)
8     {
9         if(i%2==0)
10            continue;
11         printf("%d\t", i);
12     }
13     getch();
14 }
```

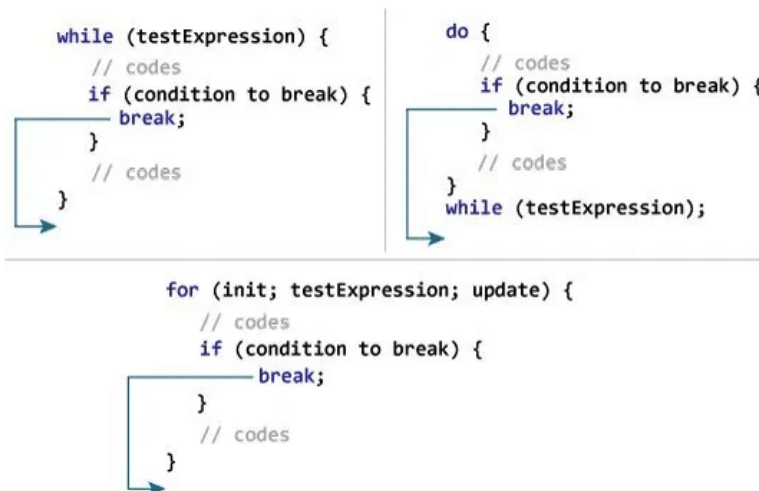
উপরের প্রোগ্রামে লুপ বডির Test Expression যখন সত্য হয় তখন continue স্টেটমেন্টটি কাজ করে। অর্থাৎ printf() ফাংশনটি নির্বাহ না করে প্রোগ্রামের নির্বাহ আবার লুপের প্রথম থেকে শুরু হয়।

**break স্টেটমেন্টঃ** লুপ কন্ট্রোল স্টেটমেন্টের লুপ বডি'র স্টেটমেন্টগুলো সাধারণত Test Expression মিথ্যা না হওয়া পর্যন্ত পুনরাবৃত্তি করতে থাকে। কিন্তু Test Expression মিথ্যা হওয়ার পূর্বেই লুপ থেকে বের হওয়ার জন্য break স্টেটমেন্ট ব্যবহার করা হয়। break স্টেটমেন্ট loops অথবা switch স্টেটমেন্টে ব্যবহৃত হয়। যখন break স্টেটমেন্ট কাজ করে তখন প্রোগ্রাম কন্ট্রোল লুপ থেকে বের হয়ে যায় এবং লুপের বাইরে প্রথম স্টেটমেন্ট থেকে প্রোগ্রাম নির্বাহ হতে থাকে। নেস্টেড লুপের ক্ষেত্রে প্রোগ্রাম কন্ট্রোল, প্রথমে ভেতরের লুপ থেকে বের হয়ে আসে এবং পরে বাইরের লুপ থেকে বের হয়ে আসে। সাধারণত break স্টেটমেন্ট এমন একটি অবস্থায় ব্যবহার করা হয় যখন লুপটি কতবার পুনরাবৃত্তি হবে তা আমাদের কাছে অজানা অথবা কোন শর্তের ভিত্তিতে প্রোগ্রাম কন্ট্রোল লুপ থেকে বের হয়ে আসা।

break স্টেটমেন্টের ফরম্যাটঃ

```
1 break;
```

break স্টেটমেন্ট যেভাবে কাজ করে-



নিচের প্রোগ্রামটি লক্ষ কর-

```

1  /* for loop ব্যবহার করে প্রোগ্রাম */
2  #include<stdio.h>
3  #include<conio.h>
4  main()
5  {
6      int i;
7      for(i=1; i<=5; i=i+1)
8      {
9          printf("Bangladesh\n");
10         if(i==3)
11             break;
12     }
13     getch();
14 }

```

উপরের প্রোগ্রামের লুপটি পাঁচ বার পুনরাবৃত্তি হওয়ার পরিবর্তে তিনবার পুনরাবৃত্তি হবে। কারণ যখন লুপ বডি'র কন্ডিশনটি সত্য হবে তখন break স্টেটমেন্টটি কাজ করবে অর্থাৎ প্রোগ্রাম কন্ট্রোল লুপ থেকে বের হয়ে যাবে। ফলে Bangladesh লেখাটি তিনবার দেখাবে।

**goto স্টেটমেন্টঃ** goto স্টেটমেন্টকে জাম্পিং স্টেটমেন্ট বলা হয়। ‘সি’ ভাষায় প্রোগ্রাম নির্বাহের নিয়ন্ত্রণ শর্তযুক্ত বা শর্তবিহীন ভাবে এক স্টেটমেন্ট থেকে উপরে বা নিচে অপর কোন স্টেটমেন্টে বা প্রোগ্রামের পূর্বনির্ধারিত কোন স্থানে স্থানান্তরের জন্য goto স্টেটমেন্ট ব্যবহার করা হয়। একটি নির্দিষ্ট শর্তের জন্য প্রোগ্রামের একটি নির্দিষ্ট অংশ পুনরাবৃত্তি করতেও goto স্টেটমেন্ট ব্যবহার করা যায়। এছাড়া মাল্টিপল লুপ ব্রেক করতেও goto স্টেটমেন্ট ব্যবহার করা যায় যা একটি সিঙ্গেল break স্টেটমেন্ট দিয়ে সম্ভব নয়। যাইহোক, goto স্টেটমেন্টের এর ব্যবহার খুবই কম। কারণ এটি প্রোগ্রামকে জটিল ও বিব্রান্তি করে।

goto স্টেটমেন্টের ফরম্যাটঃ



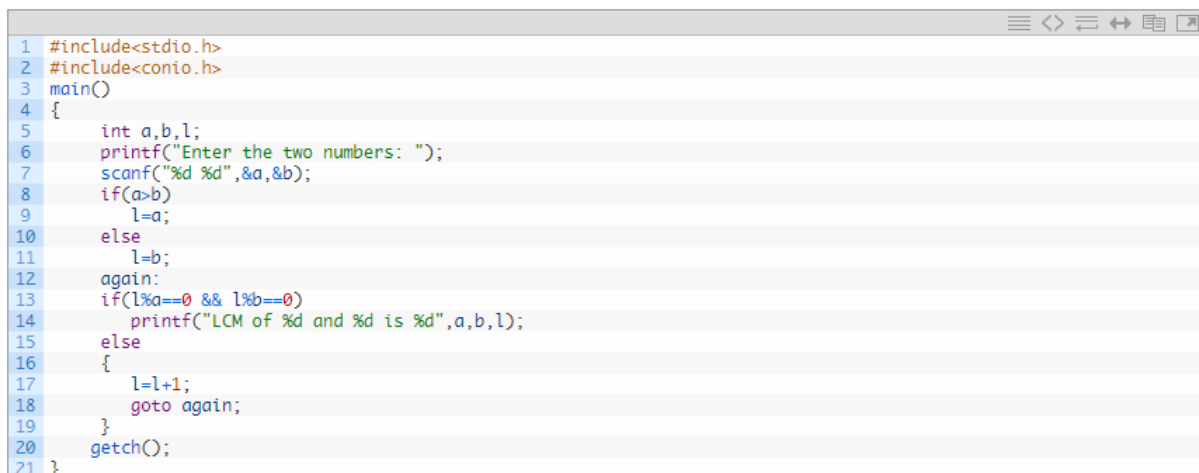
```

1 label:;
2 -----
3 -----
4 goto label;;
5 ;
6 অথবা
7 ;
8 goto label;;
9 -----
10 -----
11 label:;

```

এখানে label প্রোগ্রামারের দেওয়া একটি আইডেন্টিফায়ার, যেখানে প্রোগ্রাম নির্বাহের নিয়ন্ত্রণ স্থানান্তরিত হবে। এই আইডেন্টিফায়ার লেখার জন্য আইডেন্টিফায়ার লেখার নিয়ম মেনে লিখতে হবে। মনে রাখতে হবে, আইডেন্টিফায়ার (label) এর পরে সেমিকোলন(;) না হয়ে কোলন(:) হয়।

দুটি পূর্ণ সংখ্যার ল.সা.গু. নির্ণয়ের জন্য ‘সি’ প্রোগ্রামিং ভাষায় লেখা নিচের প্রোগ্রামটি লক্ষ করি।



```

1 #include<stdio.h>;
2 #include<conio.h>;
3 main();
4 {
5     int a,b,l;
6     printf("Enter the two numbers: ");
7     scanf("%d %d",&a,&b);
8     if(a>b)
9         l=a;
10    else
11        l=b;
12    again:
13    if(l%a==0 && l%b==0)
14        printf("LCM of %d and %d is %d",a,b,l);
15    else
16    {
17        l=l+1;
18        goto again;
19    }
20    getch();
21 }

```

উপরের প্রোগ্রামটিতে goto স্টেটমেন্ট ব্যবহার করে প্রোগ্রাম নির্বাহের নিয়ন্ত্রণ উপরে নির্দিষ্ট জায়গায় স্থানান্তর করা হয়েছে।

**পঞ্চম অধ্যায় পাঠ-১৯ ‘সি’ প্রোগ্রামিং ভাষায় অ্যারে।**

**অ্যারেঃ** অ্যারে হলো এক ধরনের ডেটা স্ট্রাকচার, যা একই ধরনের বা সমপ্রকৃতির চলকের সমাবেশ। অ্যারে একটি ডিরাইভড ডেটা টাইপ। একই টাইপের অনেকগুলো চলক নিয়ে কাজ করার প্রয়োজন হলে তখন চলক ঘোষনার পরিবর্তে অ্যারে ঘোষণা করা হয়। অ্যারেকে সাধারণত দুই ভাগে ভাগ করা হয়। যথা:

- ১। একমাত্রিক অ্যারে
- ২। বহুমাত্রিক অ্যারে (দ্বিমাত্রিক অ্যারে,...)

**একমাত্রিক অ্যারে:** একমাত্রিক অ্যারে হলো এক ধরনের লিনিয়ার অ্যারে। অ্যারের অন্তর্ভুক্ত উপাদান বা ডেটাগুলো যদি একটি মাত্র কলাম ও একাধিক সারি অথবা একটি মাত্র সারি এবং একাধিক কলামে উপস্থাপন করা হয় তাকে একমাত্রিক অ্যারে বলা হয়। একমাত্রিক অ্যারে ঘোষণার ফরম্যাটঃ

**Data\_Type Array\_Name [ array\_size ];**

array\_size বলতে বুঝায় অ্যারেতে কয়টি চলক থাকবে। এই array\_size অবশ্যই একটি পূর্ণসংখ্যা হতে হবে এবং এর মান শূন্য(০) এর চেয়ে বড় হতে হবে। Array\_Name লেখার ক্ষেত্রে চলক ঘোষণার নিয়ম মেনে লিখতে হবে। Data\_Type বলতে বুঝায় চলকগুলোতে কী ধরনের ডেটা থাকবে এবং এটি 'সি' ভাষার যেকোন বৈধ ডেটা টাইপ হতে পারে।

উদাহরণঃ

- int id [5];
- float marks[5];

**ব্যখ্যাঃ** যখন int id [5]; অ্যারে ঘোষণা করা হয়, তখন int টাইপের নিচের মত পাঁচটি চলক ঘোষণা হয়।

- int id [0];
- int id [1];
- int id [2];
- int id [3];
- int id [4];

**অ্যারের মান নির্ধারণঃ** অ্যারের মান তিনটি উপায়ে নির্ধারন করা যায়। যথা-

- ১। অ্যারে ঘোষণার সময়
- ২। অ্যারে ঘোষণার পরে
- ৩। প্রোগ্রাম নির্বাহের সময়

**অ্যারে ঘোষণার সময় মান নির্ধারণঃ** int id [5]; এই অ্যারের মান অ্যারে ঘোষণার সময় নিম্নোক্ত উপায়ে নির্ধারণ করা যায়।

int id [5]={101, 102, 103, 104, 105};

মানগুলো অ্যারেতে নিম্নোক্ত ভাবে নির্ধারণ হবে-

- int id [0]=101;
- int id [1]=102;

- `int id [2]=103;`
- `int id [3]=104;`
- `int id [4]=105;`

অ্যারে ঘোষণার পরে মান নির্ধারণঃ `int id [5];` এই অ্যারের মান অ্যারে ঘোষণার পর নিম্নোক্ত উপায়ে নির্ধারণ করা যায়।

`int id [5]; // অ্যারে ঘোষণা`

তারপর অ্যারের মান নির্ধারণ-

- `id [0]=101;`
- `id [1]=102;`
- `id [2]=103;`
- `id [3]=104;`
- `id [4]=105;`

প্রোগ্রাম নির্বাহের সময় মান নির্ধারণঃ অ্যারের সিঙ্গেল চলকের মান নির্ধারণ করা যায়। আবার লুপ স্টেটমেন্ট ব্যবহার করে অ্যারের সবগুলো চলকের মানও নির্ধারণ করা যায়। লুপ ব্যবহার করে `int id [5];` অ্যারের মান নির্ধারণ-

```

1  #include<stdio.h>
2  #include<conio.h>
3
4  main()
5  {
6      int i;
7      int id[5];
8      for(i=0; i<5; i++)
9      {
10         printf("Enter id");
11         scanf("%d",&id[i]);
12     }
13     getch();
14 }
```

দ্বিমাত্রিক অ্যারে: যে অ্যারের উপাদানগুলো একই সাথে একাধিক সারি ও একাধিক কলামে উপস্থাপন করা হয় তাকে দ্বিমাত্রিক অ্যারে বলা হয়। দ্বিমাত্রিক অ্যারে ঘোষণার ফরম্যাটঃ

**`Data_Type Array_Name [ row_size ][ column_size ];`**

`row_size` এবং `column_size` যথাক্রমে দ্বিমাত্রিক অ্যারের সারি এবং কলাম সংখ্যা বুঝায়। এই `row_size` এবং `column_size` অবশ্যই একটি পূর্ণসংখ্যা হতে হবে এবং এর মান শূন্য(০) এর চেয়ে বড় হতে হবে। `Array_Name` লেখার ক্ষেত্রে চলক ঘোষণার নিয়ম মেনে লিখতে হবে। `Data_Type` বলতে বুঝায় চলকগুলোতে কী ধরনের ডেটা থাকবে এবং এটি 'সি' ভাষার যেকোন বৈধ ডেটা টাইপ হতে পারে।



**উদাহরণঃ** একটি 3×4 ম্যাট্রিক্স এর ডেটাগুলো অ্যারেতে রাখার জন্য নিম্নোক্ত ভাবে দ্বিমাত্রিক অ্যারে ঘোষণা করতে হবে-

```
int mat [3][4];
```

**অ্যারে ব্যবহারের সুবিধা:**

- ১। একই প্রকৃতির অনেকগুলো ডেটা আইটেম একটি সিঙ্গেল নামের মাধ্যমে উপস্থাপন করা যায়।
- ২। বিভিন্ন ডেটা স্ট্রাকচার যেমন- লিংক লিস্ট, স্ট্যাক, কিউ, ট্রি, গ্রাফ ইত্যাদি বাস্তবায়নে অ্যারে ব্যবহার করা যায়।
- ৩। প্রোগ্রাম নির্বাহ দ্রুত হয়।
- ৪। উপাদানগুলোর জন্য অতিরিক্ত মেমোরি স্পেস বরাদ্দ করে না। ফলে অ্যারেতে মেমোরি ওভারফ্লো ও মেমোরি আন্ডারফ্লো হয় না।
- ৫। দ্বিমাত্রিক অ্যারের সাহায্যে ম্যাট্রিক্স উপস্থাপন করা যায়।

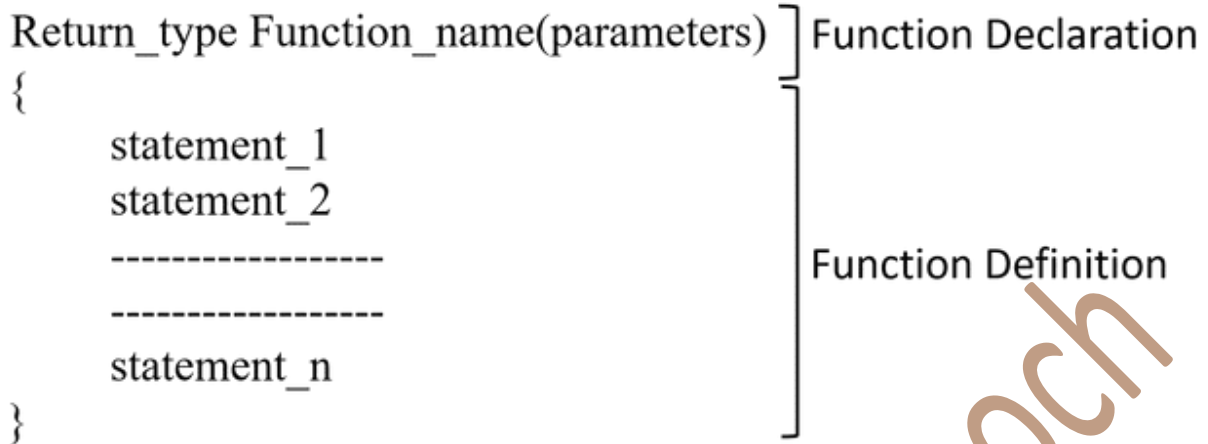
**অ্যারে ব্যবহারের অসুবিধা:**

- ১। অ্যারেতে কতগুলো উপাদান থাকবে তা পূর্বে থেকেই জানা থাকতে হয়।
- ২। অ্যারে একটি স্ট্যাটিক স্ট্রাকচার। অর্থাৎ প্রোগ্রাম নির্বাহের সময়ে ঘোষণাকৃত অ্যারের সাইজ কখনো পরিবর্তন করা যায় না।
- ৩। যেহেতু অ্যারের সাইজ পরিবর্তন করা যায় না, তাই প্রকৃত ডেটা অপেক্ষা অ্যারের সাইজ অনেক বেশি ঘোষণা করলে মেমোরির অপচয় হবে। আবার প্রকৃত ডেটা অপেক্ষা অ্যারের সাইজ কম ঘোষণা করলেও সমস্যা সৃষ্টি করতে পারে।
- ৪। অ্যারের উপাদানগুলো মেমোরিতে পাশাপাশি অবস্থান করে। তাই ডেটা ইনসার্ট এবং ডিলেট করা কঠিন এবং সময় সাপেক্ষ।

## পঞ্চম অধ্যায় পাঠ-২০ ‘সি’ প্রোগ্রামিং ভাষায় ফাংশন।

**ফাংশন:** ফাংশন হলো কতগুলো স্টেটমেন্টের সমষ্টি যা একত্রে একটি নির্দিষ্ট কাজ সম্পাদন করে। প্রতিটি ফাংশন ইনপুট নেয়, প্রসেস করে এবং একটি আউটপুট দেয়। প্রতিটি ‘সি’ প্রোগ্রাম এরূপ এক বা একাধিক ফাংশনের সমষ্টি।

**Syntax of a function:**



**ফাংশন ডিক্লারেশনঃ** ফাংশন ডিক্লারেশন ফাংশনের রিটার্ন টাইপ, ফাংশনের নাম, ফাংশনের প্যারামিটার সম্পর্কে কম্পাইলারকে তথ্য দেয়।

- **ফাংশনের রিটার্ন টাইপ** – প্রতিটি ফাংশন একটি মান রিটার্ন করে থাকে। ফাংশনটি কী টাইপের ডেটা রিটার্ন করবে তা Return\_type নির্ধারণ করে। কিছু ফাংশন কাজক্ষত অপারেশন করলেও অনেক সময় কোন মান রিটার্ন করে না। এই ক্ষেত্রে Return\_type হয় void।
- **ফাংশনের নাম**– ফাংশনের নাম একটি আইডেন্টিফায়ার যা যেকোন নাম হতে পারে। তবে ফাংশনের নাম অর্থপূর্ণ হওয়া উচিত, যাতে নাম দেখেই ফাংশনের উদ্দেশ্য বুঝা যায়। ফাংশনের নাম লেখার ক্ষেত্রে আইডেন্টিফায়ার এর নিয়ম অনুসরণ করে লেখতে হয়।
- **ফাংশন প্যারামিটার**– প্যারামিটার অংশে ডেটা টাইপ সহ চলক থাকে, যা ফাংশনটি কী ধরনের কয়টি ডেটা ইনপুট নিবে তা নির্ধারণ করে। এটি অপশনাল অর্থাৎ ফাংশনে প্যারামিটার থাকতেও পারে নাও পারে।

**ফাংশন ডেফিনেশনঃ** ফাংশনটি যে কাজ করবে তার জন্য প্রয়োজনীয় কোড এই অংশে লেখা হয়।

‘সি’ প্রোগ্রামে ব্যবহৃত ফাংশন সমূহকে দুটি ভাগে ভাগ করা হয়। যথা:

- ১। লাইব্রেরি ফাংশন
- ২। ইউজার-ডিফাইন্ড ফাংশন

**লাইব্রেরি ফাংশনঃ** লাইব্রেরি ফাংশন হলো এমন একটা ফাংশন যার ডেফিনেশন প্রোগ্রামারকে লিখতে হয় না। যেগুলো বিশেষ কিছু কার্য সম্পাদনের জন্য ‘সি’ কম্পাইলারে বিল্ট-ইন থাকে। শুধু প্রয়োজনে সেগুলোকে তাদের নিজস্ব ফরম্যাট অনুযায়ী main () ফাংশনে ব্যবহার বা কল করা হয়। যেমন- scanf(), printf(), gets(), puts(), getchar(), putchar(), abs(), pow(b,p), sqrt(), sin(), cos(), tan(), rand() ইত্যাদি হচ্ছে লাইব্রেরি ফাংশন। এক্ষেত্রে প্রতিটি লাইব্রেরী ফাংশনের নিজস্ব হেডার ফাইল প্রোগ্রামের শুরুতে লিখে দিতে হয়। লাইব্রেরি ফাংশন দুই প্রকার।

**সংখ্যাচক ফাংশন :** যে ফাংশন গাণিতিক কার্য সম্পাদন করে তাকে সংখ্যাচক ফাংশন বলে।  
যেমন- pow(b,p), sqrt(), sin(), rand() ইত্যাদি।

**স্ট্রিং ফাংশন:** যে ফাংশন কোনো স্ট্রিং নিয়ে কাজ করে তাকে স্ট্রিং ফাংশন বলে। যেমন- strcpy(), strcat(), strcmp() ইত্যাদি।

### বিভিন্ন লাইব্রেরী ফাংশন ও তাদের হেডার ফাইলঃ

লাইব্রেরী ফাংশন	হেডার ফাইল
scanf(),printf(), gets(), puts(), getchar(), putchar()	<stdio.h>
sqrt(), pow(), abs(), sin(), cos(), tan(), rand()	<math.h>
clrscr(), getch()	<conio.h>
strcpy(), strcat(), strcmp()	<string.h>

**ইউজার-ডিফাইন্ড ফাংশন:** ইউজার-ডিফাইন্ড ফাংশন হলো এমন একটা ফাংশন যার ডেফিনেশন প্রোগ্রামারকে লিখতে হয়। যেমন – main() ফাংশন একটি ইউজার-ডিফাইন্ড ফাংশন। কারণ এর ডেফিনেশন প্রোগ্রামার লিখে থাকে। একটি প্রোগ্রামে অবশ্যই একটি main() ফাংশন থাকতে হবে। কারণ কম্পাইলার প্রথমে main() ফাংশন খুঁজে বের করে এবং সেখান থেকে কাজ শুরু করে। main() ফাংশন ছাড়াও প্রোগ্রামে প্রয়োজনে ইউজার-ডিফাইন্ড ফাংশন তৈরি করে ব্যবহার করা যায়।

### ফাংশনের প্রয়োজনীয়তা:

- ১। ফাংশনের সাহায্যে প্রোগ্রাম সংক্ষিপ্ত আকারে রচনা করা যায়।
- ২। ফাংশন ব্যবহারে একই ধরনের কাজের জন্য একই ধরনের স্টেটমেন্ট বার বার লেখার প্রয়োজন হয় না। অর্থাৎ কোড পুনর্ব্যবহার করা যায়।
- ৩। প্রোগ্রামের ভুল সংশোধন বা ডিবাগিং করা সহজ হয়।
- ৪। ব্যবহারকারী তার প্রয়োজন অনুযায়ী ফাংশন তৈরি করে কার্য সম্পাদন করতে পারে।

**ফাংশনের উপাদানঃ** ‘সি’ প্রোগ্রামে কোনো ইউজার ডিফাইন্ড ফাংশন ব্যবহার করতে গেলে সাধারণত নিম্ন লিখিত চারটি বিষয় বিবেচনা করতে হয়।

- (১) ফাংশন বর্ণনা
- (২) ফাংশন কল
- (৩) ফাংশনের প্রোটোটাইপ
- (৪) ফাংশনের রিটার্ন টাইপ ও রিটার্ন স্টেটমেন্ট

**রিকার্সিভ ফাংশনঃ** একটি ফাংশন অন্য যেকোনো ফাংশনকে যেকোনো সংখ্যক বার কল করতে পারে। আবার একটি ফাংশন নিজেও নিজেই কল করতে পারে। যখন কোনো ফাংশন নিজেই নিজেই কল করে তখন সেই ফাংশনকে রিকার্সিভ ফাংশন বলা হয় এবং এই প্রক্রিয়াকে রিকার্সন বলা হয়। যখন রিকার্সন ব্যবহার করা হয়, তখন প্রোগ্রামারকে অবশ্যই সচেতন থাকতে হবে যেন একটি এক্সিট কন্ডিশন থাকে, অন্যথায় অসীম লুপের সৃষ্টি হবে। বিভিন্ন গাণিতিক সমস্যা যেমন- একটি সংখ্যার ফ্যাক্টোরিয়াল গণনা, ফিবোনাচি সিরিচ তৈরি ইত্যাদি সমাধান করতে রিকার্সিভ ফাংশন খুবই কার্যকর।

**রিকার্সিভ ফাংশন ব্যবহারের সুবিধা:**

- ১। ফাংশনের অপ্রয়োজনীয় কলিং করতে হয় না।
- ২। রিকার্সনের সাহায্যে সমস্যাকে সহজভাবে সমাধান করা যায়, যেখানে ইটারেটিভ(লুপিং) সমাধান অনেক বড় এবং জটিল।
- ৩। একই সমাধান প্রয়োগের ক্ষেত্রে এটি খুবই কার্যকরী।

**রিকার্সিভ ফাংশন ব্যবহারের অসুবিধা:**

- ১। রিকার্সিভ সমাধান সর্বদা লজিক্যাল এবং এর সমস্যা সনাক্ত করা কঠিন।
- ২। ইটারেটিভ(লুপিং) সমাধানের চেয়ে রিকার্সিভ ফাংশন নির্বাহে বেশি সময় লাগে।
- ৩। বেশি স্ট্যাক স্পেসের প্রয়োজন হয়।

**ফাংশন চেনার সহজ উপায়ঃ** ফাংশন চেনার সহজ উপায় হলো ফাংশনের নামের শেষে এক জোড়া প্রথম বন্ধনী ‘( )’ থাকে, এই প্রথম বন্ধনীর মধ্যে অনেক কিছু থাকতে পারে, আবার নাও থাকতে পারে। প্রতিটি ফাংশনের একটি নাম থাকে, যে নামে কম্পাইলার তাকে সনাক্ত করে। প্রোগ্রাম নির্বাহের সময়ে কম্পাইলার যখন কোনো ফাংশন কল করে তখন মূল প্রোগ্রামের কাজ স্থগিত রেখে কল ফাংশনে নির্বাহ শুরু করে এবং নির্বাহ শেষে মূল ফাংশনে প্রত্যাবর্তন পূর্বক পরবর্তী লাইন থেকে নির্বাহ চালিয়ে যায়। তবে এই প্রক্রিয়ায় অতিরিক্ত কিছুটা সময় ব্যয় হয়। তাই ছোট কোনো প্রোগ্রামের জন্য সাধারণত ফাংশন ব্যবহার করা হয় না।

**পঞ্চম অধ্যায় – অনুধাবন প্রশ্ন ও উত্তর সমূহ**

০, ১ দিয়ে লেখা ভাষা ব্যাখ্যা কর। / মেশিন ভাষায় লিখিত প্রোগ্রাম দ্রুত নির্বাহ হয় কেন?/ “শব্দ ছাড়া শুধু মাত্র সংখ্যার মাধ্যমে ভাষা প্রকাশ সম্ভব”- ব্যাখ্যা কর।

কম্পিউটারের নিজস্ব ভাষা হচ্ছে মেশিন ভাষা। এটি কম্পিউটারের মৌলিক ভাষা। এই ভাষায় শুধু মাত্র ০ এবং ১ ব্যবহার করা হয় বলে এই ভাষায় দেওয়া যেকোনো নির্দেশ কম্পিউটার সরাসরি বুঝতে পারে। মেশিন ভাষার সবচেয়ে বড় সুবিধা হচ্ছে সরাসরি কম্পিউটারের সাথে যোগাযোগ করা যায়। মেশিন ভাষায় লেখা প্রোগ্রাম নির্বাহের জন্য কোনো প্রকার অনুবাদক প্রোগ্রামের প্রয়োজন হয় না। ফলে দ্রুত কাজ করে। মেশিন ভাষায় লিখিত প্রোগ্রামে অতি অল্প মেমোরি প্রয়োজন হয়। কম্পিউটারের ভেতরের গঠন ভালোভাবে বুঝতে হলে এই ভাষা

জানতে হয়। এ ভাষার সবচেয়ে বড় অসুবিধা হচ্ছে এক ধরনের কম্পিউটারের জন্য লিখিত প্রোগ্রাম অন্য ধরনের কম্পিউটারে ব্যবহার করা যায় না। মেশিন ভাষাকে নিম্ন স্তরের ভাষাও বলা হয়।

**“লো-লেভেল প্রোগ্রামিং ভাষার দুর্বলতায় হাই-লেভেল প্রোগ্রামিং ভাষার উৎপত্তির কারণ”/ “উচ্চ স্তরের ভাষা মেশিন ভাষা থেকে উন্নতর”-ব্যাখ্যা কর।**

নিম্নস্তরের ভাষা যেমন মেশিন ভাষা এবং অ্যাসেম্বলি ভাষায় যথাক্রমে ০,১ এবং বিভিন্ন নেমোনিক এর সাহায্যে প্রোগ্রাম লেখা হয়। নিম্নস্তরের ভাষায় লিখিত কোনো প্রোগ্রাম সাধারণত বোঝা কষ্টকর। এ ভাষায় প্রোগ্রাম লিখতে প্রচুর সময় লাগে এবং ভুল হবার সম্ভাবনা খুব বেশি থাকে। ভুল হলে তা বের করা এবং ভুল-ত্রুটি দূর করা খুব কঠিন। এ ভাষার সবচেয়ে বড় অসুবিধা হচ্ছে এক ধরনের কম্পিউটারের জন্য লিখিত প্রোগ্রাম অন্য ধরনের কম্পিউটারে ব্যবহার করা যায় না। কিন্তু উচ্চস্তরের ভাষায় প্রোগ্রাম লেখা সহজ ও লিখতে সময় কম লাগে। এতে ভুল হবার সম্ভাবনা কম থাকে এবং প্রোগ্রামের ত্রুটি বের করে তা সংশোধন করা সহজ। এ ভাষায় প্রোগ্রাম লেখার জন্য কম্পিউটারের ভেতরের সংগঠন সম্পর্কে ধারণা থাকার প্রয়োজন নেই। এক মডেলের কম্পিউটারের জন্য লিখিত প্রোগ্রাম অন্য মডেলের কম্পিউটারে চলে। উপরের আলোচনা থেকে দেখা যায় নিম্নস্তরের ভাষার অসুবিধাসমূহ উচ্চস্তরের ভাষায় নেই। তাই বলা যায় লো-লেভেল ল্যাংগুয়েজের দুর্বলতাই হাই-লেভেল ল্যাংগুয়েজের উৎপত্তির কারণ বা উচ্চ স্তরের ভাষা মেশিন ভাষা থেকে উন্নতর।

**হাই-লেভেল প্রোগ্রামিং ভাষায় প্রোগ্রামিং করা সহজ-ব্যাখ্যা কর।**

মেশিন ও অ্যাসেম্বলি ভাষায় রচিত প্রোগ্রামের প্রধান অসুবিধা হল- এক ধরনের কম্পিউটারে রচিত প্রোগ্রাম অন্য ধরনের কম্পিউটারে ব্যবহার করা যায় না। আবার এই লো-লেভেল ভাষায় রচিত প্রোগ্রাম কম্পিউটার বুঝলেও মানুষের পক্ষে তা বোঝা অনেক কষ্টসাধ্য। অপরদিকে হাই লেভেল প্রোগ্রামিং ভাষায় প্রোগ্রাম রচনা করলে এই ধরনের কোনো সমস্যার সৃষ্টি হয় না। এই ভাষায় লিখিত প্রোগ্রাম যেকোনো কম্পিউটারে ব্যবহার করা যায়। এই ভাষায় রচিত প্রোগ্রাম মানুষের পক্ষে বোঝা অনেক সহজ। প্রোগ্রাম লিখতে সময় অনেক কম লাগে। লো-লেভেল ভাষায় প্রোগ্রাম লিখতে যেখানে ৪/৫ টি নির্দেশের দরকার হয়, সেখানে হাই-লেভেল ভাষায় মাত্র একটি বাক্য লিখলেই চলে। প্রোগ্রাম লেখার সময় কম্পিউটার হার্ডওয়্যার সম্পর্কে ধারণা থাকার প্রয়োজন নেই। তাই বলা যায় হাই-লেভেল প্রোগ্রামিং ভাষায় প্রোগ্রামিং করা সহজ।

**অ্যাসেম্বলি ভাষা মেশিন ভাষা থেকে উন্নত কেন?**

যান্ত্রিক ভাষায় প্রোগ্রাম রচনা করার মতো কঠিন কাজকে অপেক্ষাকৃত সহজ করার জন্য এবং সেই সাথে রচিত প্রোগ্রামের ভুল সংশোধন ও পরিবর্তনের কাজ সহজতর করে প্রোগ্রামিং এ গতি সঞ্চারণের লক্ষ্যে অ্যাসেম্বলি ভাষার উন্নয়ন সাধিত হয়। অ্যাসেম্বলি ভাষায় রচিত প্রোগ্রাম মেশিন ভাষার তুলনায় অনেক সহজ হয়। প্রোগ্রাম রচনায় তুলনামূলক সময় অনেক কম লাগে। সহজে ভুল সংশোধন করা যায় এবং প্রোগ্রাম পরিবর্তন করা যায়।

অ্যালগরিদম কোডিং এর পূর্বশর্ত-ব্যাখ্যা কর। /প্রোগ্রাম কোডিং-এ অ্যালগরিদমের গুরুত্ব লেখ।

কোনো একটি নির্দিষ্ট সমস্যা সমাধানের জন্য যুক্তিসম্মত ও ধাপে ধাপে সমাধান করার যে পদ্ধতি, তাকে অ্যালগরিদম বলা হয়। অপরদিকে কোনো সমস্যাকে কম্পিউটার দ্বারা সমাধান করার জন্য প্রোগ্রামিং ভাষায় নির্দেশনা দেওয়াকেই বলে কোডিং। এক্ষেত্রে কোনো সমস্যাকে কম্পিউটার দ্বারা সমাধান করার পূর্বে অ্যালগরিদম অনুসরণ করলে যে সুবিধাগুলো পাওয়া যায়, তা হলো- সহজে প্রোগ্রামের উদ্দেশ্যে বোঝা যায়। সহজে প্রোগ্রামের ভুল নির্ণয় করা যায়। প্রোগ্রাম প্রবাহের দিক বুঝা যায়। জটিল প্রোগ্রাম সহজে রচনা করা যায়। প্রোগ্রাম পরিবর্তন ও পরিবর্তনে সহায়তা করে। অর্থাৎ কোডিং করার পূর্বে অ্যালগরিদম অনুসরণ করলে অনেক সুবিধা পাওয়া যায়। তাই বলা যায় অ্যালগরিদম কোডিং বা প্রোগ্রামিং এর পূর্বশর্ত।

“অ্যালগরিদমের চেয়ে ফ্লোচার্টের মাধ্যমে সমস্যা শনাক্ত করা সহজ”-ব্যাখ্যা কর।

সমস্যা সমাধানের ধাপসমূহের লিখিত পদ্ধতিকে অ্যালগরিদম বলে। অ্যালগরিদম বর্ণনা নির্ভর। অ্যালগরিদম তৈরির পূর্বে সুডোকোড তৈরির প্রয়োজন হতে পারে। অ্যালগরিদমে প্রোগ্রাম বুঝতে সময় বেশি লাগে। অন্যদিকে চিত্র বা সাংকেতিক চিহ্নের মাধ্যমে সমস্যা সমাধানের ধাপসমূহকে ফ্লোচার্ট বলে। ফ্লোচার্ট চিত্র নির্ভর। ফ্লোচার্ট রচনার ক্ষেত্রে সুডোকোডের কোন প্রয়োজন হয় না। ফ্লোচার্টে প্রোগ্রাম বুঝতে সময় অনেক কম লাগে। সুতরাং বলা যায় যে, অ্যালগরিদমের চেয়ে ফ্লোচার্টের মাধ্যমে সমস্যা শনাক্ত করা সহজ।

প্রবাহচিত্রে ব্যবহৃত প্রতীকগুলো ব্যাখ্যা কর। / ফ্লোচার্ট হলো চিত্রভিত্তিক অ্যালগোরিদম –ব্যাখ্যা কর।

প্রবাহচিত্রে ব্যবহৃত প্রতীকগুলোর কাজ নিচে দেওয়া হলঃ



চিত্রটি দিয়ে কোনো প্রোগ্রামের শুরু কিংবা শেষ বোঝায়।



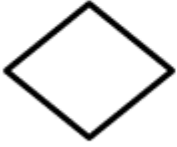
চিত্রটি দিয়ে কোনো প্রোগ্রামের ইনপুট কিংবা আউটপুট বোঝায়।



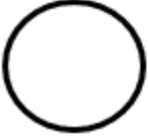
চিত্রটি দিয়ে কোনো প্রোগ্রামের প্রক্রিয়াকরণ বোঝায়।



চিত্রটি দিয়ে কোনো প্রবাহচিত্রের প্রবাহের দিক বোঝায়।



চিত্রটি দিয়ে কোনো প্রোগ্রামের কোনো বিষয়ের সিদ্ধান্ত নেওয়া বোঝায়।



চিত্রটি দিয়ে কোনো প্রোগ্রামের একাধিক ধাপের সংযোগ বোঝায়।

**সি প্রোগ্রামিং ভাষাকে মিজ লেভেল প্রোগ্রামিং ভাষা বলা হয় কেন?**

কম্পিউটারের সাহায্যে কোন সমস্যা সমাধান তথা প্রোগ্রাম রচনার জন্য ব্যবহৃত শব্দ, বর্ণ, অংক, চিহ্ন প্রভৃতির সমন্বয়ে গঠিত রীতিনীতিকে প্রোগ্রামিং ভাষা বলা হয়। “সি” প্রোগ্রামিং ভাষাকে মিজ লেভেল ভাষা বলা হয়। কারণ এতে নিম্নস্তরের ভাষার সুবিধা যেমন বিট, বাইট ও মেমোরি অ্যাড্রেস নিয়ে কাজ করা যায়। আবার উচ্চস্তরের ভাষার সুবিধা যেমন বিভিন্ন ডেটা টাইপ নিয়ে কাজ করা যায়। অর্থাৎ “সি” প্রোগ্রামিং ভাষায় নিম্নস্তরের ভাষার সুবিধার পাশাপাশি উচ্চস্তরের ভাষার সুবিধাও পাওয়া যায়। তাই “সি” প্রোগ্রামিং ভাষাকে মিজ লেভেল প্রোগ্রামিং ভাষা বলা হয়।

**সি একটি উচ্চ স্তরের প্রোগ্রামিং ভাষা-ব্যাখ্যা কর।**

**অনুবাদক প্রোগ্রাম হিসেবে কম্পাইলার বেশি উপযোগী-ব্যাখ্যা কর।**



যে প্রোগ্রামের সাহায্যে উৎস (Source) প্রোগ্রামকে বস্তু (Object) প্রোগ্রামে পরিণত করা হয় তাকে অনুবাদক প্রোগ্রাম বলে। মেশিন ভাষায় লেখা প্রোগ্রামকে বলা হয় বস্তু প্রোগ্রাম (Object Program) এবং অন্য যেকোনো প্রোগ্রামিং ভাষায় লেখা প্রোগ্রামকে বলা হয় উৎস প্রোগ্রাম (Source program)। কম্পাইলার হলো এক ধরনের অনুবাদক প্রোগ্রাম যা হাইলেভেল ভাষায় লিখিত প্রোগ্রামকে মেশিন ভাষায় রূপান্তর করে। অর্থাৎ সোর্স প্রোগ্রামকে অবজেক্ট প্রোগ্রামে রূপান্তর করে। অনুবাদক প্রোগ্রাম হিসেবে কম্পাইলার বেশি উপযোগী কারণ- কম্পাইলার সম্পূর্ণ প্রোগ্রামটিকে একসাথে অনুবাদ করে ফলে প্রোগ্রাম নির্বাহের গতি দ্রুত হয়। প্রোগ্রাম নির্বাহে কম সময় লাগে, কম্পাইলারের মাধ্যমে রূপান্তরিত প্রোগ্রাম সম্পূর্ণরূপে মেশিন প্রোগ্রামে রূপান্তরিত হয়, একবার প্রোগ্রাম কম্পাইল করা হলে পরবর্তিতে আর কম্পাইলের প্রয়োজন হয় না, প্রোগ্রামে কোন ভুল থাকলে তা মনিটরে একসাথে প্রদর্শন করে। উপরোক্ত বৈশিষ্ট্য থেকে বলা যায় অনুবাদক প্রোগ্রাম হিসেবে কম্পাইলার বেশি উপযোগী।

প্রত্যেকবার প্রোগ্রাম নির্বাহের সময় কম্পাইল করা প্রয়োজন – ব্যাখ্যা কর।

কম্পাইলারের তুলনায় ইন্টারপ্রেটার কোন ক্ষেত্রে ভালো – ব্যাখ্যা কর।

উৎস প্রোগ্রামকে কম্পাইল করার প্রয়োজন হয় কেন? ব্যাখ্যা কর।

সি প্রোগ্রামিং ভাষা একটি কেস সেনসিটিভ ভাষা – বুঝিয়ে লিখ।

সি প্রোগ্রামিং ভাষা একটি উচ্চস্তরের ভাষা। এই ভাষা প্রায় মানুষের ভাষার কাছাকাছি তাই এই ভাষার সাহায্যে প্রোগ্রাম রচনা করা সহজ। সি প্রোগ্রামিং ভাষাটির একটি বৈশিষ্ট্য হল এটি case sensitive প্রোগ্রামিং ভাষা। অর্থাৎ এই ভাষায় লেখা প্রোগ্রামে ছোট হাতের অক্ষর এবং বড় হাতের অক্ষর আলাদা অর্থ বহন করে। যেমন: a এবং A দুটি আলাদা চলক। তাই বলা হয় সি প্রোগ্রামিং ভাষা একটি case sensitive প্রোগ্রামিং ভাষা।

চলক তৈরির ক্ষেত্রে কিছু বিধিবদ্ধ নিয়ম কানুন রয়েছে – ব্যাখ্যা কর।/সি ভাষায় int roll@no; স্টেটমেন্টটি বৈধ/অবৈধ ব্যাখ্যা কর।

চলক হলো মেমোরি লোকেশনের নাম বা ঠিকানা। অর্থাৎ প্রোগ্রামে ডেটা নিয়ে কাজ করার সময় প্রতিটি ডেটার জন্য একটি চলকের ব্যবহার করতে হয়। প্রতিবার প্রোগ্রাম নির্বাহের সময় মেমরিতে ভেরিয়েবলগুলো অবস্থান এবং সংরক্ষিত মান পরিবর্তন হয় বা হতে পারে বলে এদেরকে ভেরিয়েবল বা চলক বলা হয়। একটি প্রোগ্রামের শুরুতে প্রয়োজনীয় সংখ্যক চলক বা ভেরিয়েবল ঘোষণা করা হয়। এই চলক ঘোষণা করার কিছু নিয়ম আছে। যেমন-

- ১. ভেরিয়েবল এর নাম হিসাবে কেবল অ্যালফাবেটিক ক্যারেক্টার (A-Z,a-z), সংখ্যা (0-9) এবং আন্ডারস্কোর ( ) ব্যবহার করা যায়। যেমন: Roll\_number, Counta7 ইত্যাদি।



- ২. ভেরিয়েবলের নামের প্রথম ক্যারেক্টার অবশ্যই অক্ষর হতে হবে। যেমন: Roll\_number, number, amount, Roll\_22 সঠিক ভেরিয়েবল। কিন্তু 22Roll সঠিক ভেরিয়েবল নয়।
- ৩. ভেরিয়েবল এ কোন প্রকার স্পেশাল ক্যারেক্টার যেমন !, @, #, %, &, \$ ইত্যাদি ব্যবহার করা যাবে না। যেমন- my@roll, ashek\$mizan\$amir, &a ইত্যাদি অবৈধ ভেরিয়েবল।
- ৪. কোন কী-ওয়ার্ড বা রিজার্ভ ওয়ার্ড ভেরিয়েবল এর নাম হিসাবে ব্যবহার করা যাবে না। যেমন: for, while ইত্যাদি। তাই বলা যায় চলক তৈরির ক্ষেত্রে কিছু নিয়ম কানুন রয়েছে।

সি প্রোগ্রামিং ভাষার লাইব্রেরি ফাংশনের হেডার ফাইল বলতে কী বুঝায়?

সি প্রোগ্রামের সোর্স কোডের লিঙ্ক সেকশনে লাইব্রেরিতে সংরক্ষিত যে সব ফাইলকে যুক্ত করা হয় তাদেরকে header ফাইল বলে। এই ফাইলের এক্সটেনশন হলো “.h” যেমন- stdio.h সি প্রোগ্রামিং ভাষার একটি হেডার ফাইল। উক্ত ফাইলে printf() বা scanf() ফাংশনের বর্ণনা রয়েছে। ফলে printf() ফাংশন প্রোগ্রামে ব্যবহার করলে stdio.h হেডার ফাইলটি প্রোগ্রামে অন্তর্ভুক্ত করতে হয়। এখানে #include<stdio.h> লিখে প্রোগ্রামে উক্ত হেডার ফাইলটি সংযুক্ত করা হয়েছে।

সি প্রোগ্রামিং ভাষায় লিখিত প্রোগ্রামে #include<stdio.h> আবশ্যিক কেন? ব্যাখ্যা কর।

একটি আদর্শ প্রোগ্রামে ইনপুট, প্রসেস ও আউটপুট এর ব্যবস্থা থাকতে হয়। সি প্রোগ্রামিং ভাষায় লেখা একটি প্রোগ্রামে ইনপুট এবং আউটপুট এর জন্য যথাক্রমে scanf() এবং printf() নামক লাইব্রেরী ফাংশন ব্যবহার করা হয়। এই ফাংশন দুটির জন্য #include<stdio.h> হেডার ফাইল ব্যবহার করা হয়। তাই বলা যায়, সি প্রোগ্রামিং ভাষায় লিখিত প্রোগ্রামে #include<stdio.h> আবশ্যিক।

সি প্রোগ্রামিং ভাষায় লিখিত প্রোগ্রামে main() ফাংশনের গুরুত্ব লেখ।

প্রোগ্রামে লাইব্রেরী ও ইউজার ডিফাইন্ড ফাংশন এক নয়-ব্যাখ্যা কর।

প্রোগ্রামে যখন কতগুলো স্টেটমেন্ট কোনো নামে একটি ব্লকের মধ্যে অবস্থান করে কোনো নির্দিষ্ট কাজ সম্পাদন করে তখন ব্লকটিকে ফাংশন বলা হয়। একটি ফাংশনের দুইটি অংশ থাকে। ফাংশন ডিক্লারেশন ও ফাংশন ডেফিনেশন। যে ফাংশনের ডেফিনেশন প্রোগ্রামার দ্বারা নির্ধারিত হয়, সে ফাংশনকে ইউজার ডিফাইন্ড ফাংশন বলে। অন্যদিকে যে ফাংশনের ডেফিনেশন পূর্ব নির্ধারিত থাকে, তাকে লাইব্রেরী ফাংশন বলে। প্রোগ্রামে কোন লাইব্রেরী ফাংশন ব্যবহার করলে তার জন্য নির্দিষ্ট হেডার ফাইল প্রোগ্রামের শুরুতে যুক্ত করতে হয়। তাই বলা যায় লাইব্রেরী ফাংশন এবং ইউজার ডিফাইন্ড ফাংশন এক নয়।

আউটপুট ফাংশন বলতে কি বুঝায়?

সি প্রোগ্রামে যখন কোনো নির্দিষ্ট কাজ সম্পাদনের জন্য কতগুলো স্টেটমেন্ট কোনো নামে একটি ব্লকের মধ্যে রাখা হয় তখন তাকে ফাংশন বলা হয়। প্রতিটি ফাংশন একটি নির্দিষ্ট কাজ করে। আউটপুট ফাংশন হচ্ছে এমন একটি ফাংশন যা মনিটরে কোন কিছু প্রদর্শন করে। যেমন- printf() হচ্ছে একটি আউটপুট ফাংশন যা প্রোগ্রামে কোনকিছু আউটপুট হিসেবে দেখায়।

**সি প্রোগ্রামিং ভাষায় ব্যবহৃত ইনপুট ও আউটপুট ফাংশনগুলো লেখ।**

সি প্রোগ্রামে যখন কোনো নির্দিষ্ট কাজ সম্পাদনের জন্য কতগুলো স্টেটমেন্ট কোনো নামে একটি ব্লকের মধ্যে রাখা হয় তখন তাকে ফাংশন বলা হয়। প্রতিটি ফাংশন একটি নির্দিষ্ট কাজ সম্পাদন করে। সি প্রোগ্রামে কোনো মান ইনপুট হিসেবে গ্রহণ করার জন্য scanf(), gets() ও getchar() ফাংশন ব্যবহৃত হয়। মনিটরে কোন কিছু প্রদর্শন করার জন্য printf(), puts() ও putchar() আউটপুট ফাংশন ব্যবহৃত হয়।

**printf(“%d %d”, a, b); স্টেটমেন্টটি ব্যাখ্যা কর।**

printf(“%d %d”, a, b); স্টেটমেন্টটিকে সি প্রোগ্রামিং ভাষায় আউটপুট স্টেটমেন্ট বলা হয়। আউটপুট স্টেটমেন্ট হলো যার মাধ্যমে ব্যবহারকারীকে আউটপুট দেখানো হয়। উপরের স্টেটমেন্টটির মাধ্যমে দুটি চলক a ও b এর মান প্রদর্শন করা হয়েছে।

**scanf(“%f”, &a); স্টেটমেন্টটি ব্যাখ্যা কর।**

scanf(“%f”, &a); এই স্টেটমেন্ট কে “সি” প্রোগ্রামিং ভাষায় ইনপুট স্টেটমেন্ট বলা হয়। ইনপুট স্টেটমেন্ট হলো যার মাধ্যমে প্রোগ্রাম ইউজারের কাছ থেকে ডেটা ইনপুট নেয়। এই স্টেটমেন্টের মাধ্যমে প্রোগ্রাম ইউজারের কাছ থেকে একটি ভগ্নাংশ ধরনের সংখ্যা ইনপুট নিয়ে a চলকে সংরক্ষণ করে যা পরবর্তীতে ব্যবহার করে।

**printf() ও scanf() উদাহরণ দ্বারা ব্যাখ্যা কর।**

সি প্রোগ্রামিং ভাষায় বহুল ব্যবহৃত ও গুরুত্বপূর্ণ ফাংশন হলো printf() এবং scanf()। প্রোগ্রামে কোনো মান ইনপুট নেওয়ার জন্য scanf() ফাংশন ব্যবহৃত হয় এবং কোনো ফলাফল প্রদর্শনের জন্য printf() ফাংশন ব্যবহৃত হয়। যেমন- scanf(“%d”, &a); এবং printf(“%d”, a); স্টেটমেন্ট দুটির সাহায্যে যথাক্রমে a চলকে ইনপুট নেওয়া ও a চলকের মান আউটপুটে দেখানো হয়।

**Integer এর পরিবর্তে কখন Long Integer ব্যবহার করা হয়? বুঝিয়ে লেখ।**

integer হল একধরনের ডেটা টাইপ যা পূর্ণসংখ্যা ইনপুট নেওয়ার জন্য সি প্রোগ্রামে ব্যবহৃত হয়। এটি সাধারণত ভেরিয়েবল ডিক্লারেশন এর সময় ভেরিয়েবল এর সামনে লিখে দিতে হয়। যেমনঃ `int a=32;` এর রেঞ্জ -৩২৭৬৮ থেকে +৩২৭৬৭ পর্যন্ত। এর চেয়ে বড় কোন রেঞ্জের সংখ্যা ইনপুট নেওয়ার জন্য long integer ব্যবহৃত হয়।

**সি ভাষায় Float type বলতে কি বুঝ? উদাহরণ সহ লিখ।**

যদি কোনো ডেটার মান সংখ্যাসূচক কিন্তু ভগ্নাংশ হয় তবে তার জন্য ব্যবহৃত চলক অথবা ধ্রুবকের ডেটা টাইপকে float বলা হয়। এ ধরনের চলকের জন্য মেমোরিতে ৪ বাইট জায়গা লাগে। যেমন- ধরি `x=৩২`, `printf("%f", x);` তাহলে আউটপুট হবে `x=32.000000`

**কখন ইউনারি অপারেটর ব্যবহার করা হয়? ব্যাখ্যা কর।**

সি ভাষায় গাণিতিক এবং যৌক্তিক কাজ নিয়ন্ত্রণ করার জন্য কতগুলো বিশেষ চিহ্ন ব্যবহৃত হয়, এগুলোকে অপারেটর বলা হয়। আর যা ডেটা ধারণ করে তাকে অপারেণ্ড বলা হয়। যে সকল অপারেটর একটি মাত্র অপারেণ্ড নিয়ে কাজ করে তাদের ইউনারি অপারেটর বলে। অর্থাৎ যখন প্রোগ্রামিং এর ক্ষেত্রে একটি মাত্র অপারেণ্ড নিয়ে কাজ করতে হয়, সেই ক্ষেত্রে ইউনারি অপারেটর ব্যবহার করা হয়। যেমন- ইনক্রিমেন্টাল অপারেটর (`++`) এবং ডিক্রিমেন্টাল অপারেটর (`-`) ইত্যাদি।

**প্রোগ্রামে অপারেটরের গুরুত্ব লেখ।**

সি ভাষায় গাণিতিক এবং যৌক্তিক কাজ নিয়ন্ত্রণ করার জন্য কতগুলো বিশেষ চিহ্ন/symbol (যেমন- `+`, `-`, `*`, `/`, `++`, `-`, `<`, `>`, `>=` ইত্যাদি) ব্যবহৃত হয় যা অপারেটর নামে পরিচিত। এই অপারেটর সমূহ না থাকলে সি ভাষায় কখনো গাণিতিক বা যৌক্তিক কাজ করা সম্ভব হতো না।

**`i++` এবং `++i` ব্যাখ্যা কর।**

`i++` হলো পোস্ট ইনক্রিমেন্ট বা পোস্টফিক্স। পোস্টফিক্স প্রথমে বাম পাশে চলকের মান অ্যাসাইন করে তারপর অপারেণ্ডের মান 1 বৃদ্ধি করে। `++i` হলো প্রিইনক্রিমেন্ট বা প্রিফিক্স। প্রিফিক্স প্রথমে অপারেণ্ডের সাথে 1 যোগ করে। তারপর ফলাফলকে বাম পাশের চলকে অ্যাসাইন করে।

**“=” এবং “==” এর মধ্যে পার্থক্য লেখ।**

“=” হচ্ছে অ্যাসাইনমেন্ট অপারেটর। কোনো চলকের মানকে বা কোন সংখ্যাকে অন্য কোনো চলকের মান হিসেবে নির্ধারণ করতে যেসব অপারেটর ব্যবহার করা হয়, তাকে অ্যাসাইনমেন্ট অপারেটর বলে। যেমন-  $C=A+B$ । এখানে A ও B চলকের মান যোগ করে যোগফল C চলকে অ্যাসাইন করা হয়েছে। “= =” হলো রিলেশনাল অপারেটর। দুটি অপারেটরের মধ্যে তুলনা করতে যে অপারেটরসমূহ ব্যবহৃত হয় তাদেরকে সম্পর্কযুক্ত অপারেটর বলা হয়। যেমন-  $a==b$  হলো a এবং b এর মান সমান।

**for loop এবং Do while loop এর মধ্যে কোনটি ব্যবহার করা সহজ?**

for এবং do while loop এর মধ্যে for loop ব্যবহার করা সহজ। কারণ সি প্রোগ্রামে কোন স্টেটমেন্ট দুই বা ততোধিক বার ব্যবহার করার জন্য for loop ব্যবহৃত হয়। loop কতবার নির্বাহ করা হবে তা জানা থাকলে for loop ব্যবহার করা উপযোগী। তাই do while loop এর চেয়ে for loop ব্যবহার করা সহজ।

**“গ্লোবাল ভেরিয়েবলকে প্রোগ্রামের সকল অংশে জুড়ে দেওয়া সম্ভব”-ব্যাখ্যা কর।**

কোনো চলক বা ভেরিয়েবল যদি main() ফাংশনের বাইরে ঘোষণা করা হয় তবে সেই ভেরিয়েবলের কার্যকারিতা main() ফাংশন বা ঐ প্রোগ্রামে ব্যবহৃত সকল ফাংশনের মধ্যে বিস্তৃত থাকে। আর এই ধরনের চলককে গ্লোবাল ভেরিয়েবল বলা হয়। এই ধরনের চলক প্রোগ্রামের সকল ফাংশনেই ব্যবহার করা সম্ভব।

**“অ্যারে এবং চলক এক নয়”-ব্যাখ্যা কর।**

একই ধরনের বা সমপ্রকৃতির চলকের সমাবেশকে অ্যারে বলা হয়। অ্যারের একটি নাম থাকে, ইনডেক্স নাম্বার এবং আইটেম সমূহকে বন্ধনীর মধ্যে উপস্থাপন করা হয়। অ্যারে একমাত্রিক ও বহুমাত্রিক হতে পারে। মেমোরি অ্যাড্রেস সরাসরি ব্যবহার না করে একটি নাম দিয়ে ঐ নামের অধীনে ডেটা রাখা হয়। এই ডেটা পরিবর্তনশীল তাই ঐ নামকে চলক বলা হয়। চলক হলো প্রোগ্রামের দেওয়া মেমোরির কয়েক বাইট স্থানের একটি নাম। তাই বলা যায় যে, অ্যারে এবং চলক এক নয়।

**অ্যারে প্রোগ্রামের জটিলতা কমায়-ব্যাখ্যা কর।**

**একই জাতীয় একাধিক ডেটা একটি চলকের আন্ডারে রাখা সম্ভব -ব্যাখ্যা কর।**

**পঞ্চম অধ্যায় – জ্ঞানমূলক প্রশ্ন ও উত্তর সমূহ।**

## প্রোগ্রাম কী?

কম্পিউটার বা যন্ত্রের সাহায্যে কোন সমস্যা সমাধানের জন্য প্রোগ্রামিং ভাষায় লেখা প্রয়োজনীয় নির্দেশমালার সমষ্টিকে প্রোগ্রাম (program) বলা হয়।

## প্রোগ্রামিং কী?

প্রোগ্রাম রচনার পদ্ধতি বা কৌশলকে প্রোগ্রামিং (programming) বা কোডিং বলা হয়।

## প্রোগ্রামার কী?

যে বা যিনি কম্পিউটার বা যন্ত্রের সাহায্যে কোন সমস্যা সমাধানের জন্য প্রোগ্রামিং ভাষায় প্রয়োজনীয় নির্দেশমালা বা প্রোগ্রাম লিখে তাকে প্রোগ্রামার বলা হয়।

## প্রোগ্রামিং ভাষা কী?

কম্পিউটারের মাধ্যমে কোন সমস্যা সমাধান তথা প্রোগ্রাম রচনার জন্য ব্যবহৃত শব্দ, বর্ণ, অংক, চিহ্ন প্রভৃতির সমন্বয়ে গঠিত রীতিনীতিকে প্রোগ্রামিং ভাষা (Programming Language) বলা হয়।

## 4GL কী?

4GL বলতে 4<sup>th</sup> Generation Language বা চতুর্থ প্রজন্মের ভাষা বুঝায়। 4GL এর সাহায্যে সহজেই অ্যাপ্লিকেশন তৈরি করা যায় বলে একে Rapid Application Development (RAD) টুলও বলা হয়।

## উচ্চস্তরের প্রোগ্রামিং ভাষা কী?

যে প্রোগ্রামিং ভাষার প্রতীক এবং শব্দ সমূহ সাধারণত গাণিতিক ও ইংরেজি ভাষার মত এবং যা মানুষের জন্য সহজে বোধগম্য, সে প্রোগ্রামিং ভাষাকে উচ্চস্তরের প্রোগ্রামিং ভাষা বলে। এটি মেশিন নির্ভর নয়।

## মেশিন ভাষা কী?

কম্পিউটারের নিজস্ব ভাষা হচ্ছে মেশিন ভাষা। এটি কম্পিউটারের মৌলিক ভাষা। এই ভাষায় শুধু মাত্র ০ এবং ১ ব্যবহার করা হয় বলে এই ভাষায় দেওয়া কোনো নির্দেশ কম্পিউটার সরাসরি বুঝতে পারে। এর সাহায্যে সরাসরি কম্পিউটারের সাথে যোগাযোগ করা যায়।

## অ্যাসেম্বলি ভাষা কী?

যে প্রোগ্রামিং ভাষায় প্রোগ্রাম লেখার জন্য বিভিন্ন নেমোনিক বা সংকেত ব্যবহার করা হয়, সে প্রোগ্রামিং ভাষাকে অ্যাসেম্বলি ভাষা বলা হয়। এটি দ্বিতীয় প্রজন্মের এবং নিম্নস্তরের প্রোগ্রামিং ভাষা।

## নেমোনিক কী?

অ্যাসেম্বলি ভাষায় ব্যবহৃত বিভিন্ন সংকেতকে নিমোনিক বলে। যেমনঃ ADD, SUB, MUL ও DIV ইত্যাদি।

### অনুবাদক প্রোগ্রাম কী?

যে প্রোগ্রামের সাহায্যে উৎস (Source) প্রোগ্রামকে বস্তু (Object) প্রোগ্রামে পরিণত করা হয় তাকে অনুবাদক প্রোগ্রাম বলে। মেশিন ভাষায় লেখা প্রোগ্রামকে বলা হয় বস্তু প্রোগ্রাম (Object Program) এবং অন্য যেকোনো ভাষায় লেখা প্রোগ্রামকে বলা হয় উৎস প্রোগ্রাম (Source program)।

### আসেম্বলার কী?

আসেম্বলার একটি অনুবাদক প্রোগ্রাম যা অ্যাসেম্বলি ভাষায় লিখিত প্রোগ্রামকে মেশিন ভাষায় অনুবাদ করার জন্য ব্যবহার করা হয়। এটি অ্যাসেম্বলি ভাষায় লিখিত প্রোগ্রামকে যান্ত্রিক ভাষায় রূপান্তর করে অর্থাৎ, নিমোনিক কোডকে মেশিন ভাষায় অনুবাদ করে।

### কম্পাইলার কী?

কম্পাইলার হলো এক ধরনের অনুবাদক প্রোগ্রাম যা হাইলেভেল ভাষায় লিখিত প্রোগ্রামকে মেশিন ভাষায় রূপান্তর করে। কম্পাইলার সম্পূর্ণ প্রোগ্রামটিকে একসাথে অনুবাদ করে।

### ইন্টারপ্রেটার কী?

ইন্টারপ্রেটার হলো এক ধরনের অনুবাদক প্রোগ্রাম যা কম্পাইলারের মতো হাইলেভেল ভাষায় লিখিত প্রোগ্রামকে মেশিন ভাষায় রূপান্তর করে। তবে ইন্টারপ্রেটার লাইন বাই লাইন অনুবাদ করে।

### ডিবাগিং কী?

প্রোগ্রামে যে কোনো ভুল চিহ্নিত করতে পারলে ভুলকে বলা হয় বাগ (Bug)। উক্ত বাগকে সমাধান করার প্রক্রিয়াকে বলা হয় ডিবাগ (Debug) বা ডিবাগিং।

### প্রোগ্রাম টেস্টিং?

প্রোগ্রাম টেস্টিং হচ্ছে কোনো প্রোগ্রাম কোডিং সম্পন্ন করার পর প্রোগ্রামটির বিভিন্ন ইনপুটের জন্য যে ধরনের আউটপুট বা ফলাফল হওয়া উচিত তা ঠিকমতো আসছে কিনা বা রান করছে কিনা তা যাচাই করা।

### সিনট্যাক্স ভুল/ব্যাকরণগত ভুল?

প্রোগ্রামে প্রোগ্রামিং ভাষার সিনট্যাক্স বা ব্যাকরণগত যেসব ভুল থাকে তাকে বলা হয় সিনট্যাক্স ভুল। যেমন- বানান ভুল, কমা, ব্রাকেট ঠিকমতো না দেয়া, কোনো চলকের মান না জানানো প্রভৃতি। এসব ভুল সংশোধন করা খুবই সহজ, কারণ সিনট্যাক্স ভুলের বেলায় অনুবাদক প্রোগ্রাম ভুলের বার্তা ছাপায়।

### লজিক্যাল ভুল কী?

প্রোগ্রামে যুক্তির ভুল থাকলে তাকে বলে লজিক্যাল ভুল। সাধারণত সমস্যা ঠিকমতো না বোঝার জন্যই এ ভুল হয়। যেমন-  $a > b$  এর স্থলে  $a < b$  বা  $s = a + b$  এর স্থানে  $s = a - b$  লিখলে লজিক্যাল ভুল হয়।

### অ্যালগরিদম কী?

কোনো একটি নির্দিষ্ট সমস্যা সমাধানের জন্য যুক্তিসম্মত ও সসীম সংখ্যক ধাপে সমাধান করার যে পদ্ধতি, তাকে অ্যালগরিদম বলা হয়। কোনো সমস্যাকে কম্পিউটার প্রোগ্রামিং দ্বারা সমাধান করার পূর্বে কাগজে-কলমে সমাধান করার জন্যই অ্যালগরিদম ব্যবহার করা হয়।

### ফ্লোচার্ট কী?

যে চিত্রভিত্তিক পদ্ধতিতে বিশেষ কতকগুলো চিহ্নের সাহায্যে কোনো একটি নির্দিষ্ট সমস্যার সমাধান করা হয় তাকে ফ্লোচার্ট বলা হয়।

### সুডোকোড কী?

প্রোগ্রাম রচনার ধারাবাহিক বর্ণনাই হল সুডোকোড। সুডোকোড মানুষের ভাষা ব্যবহার করে লেখা হয়।

### OOP কী?

OOP এর পূর্ণরূপ হল Object Oriented Programming.

### চলক কী?

প্রোগ্রাম নির্বাহের সময় যার মান পরিবর্তনশীল তাকে চলক বলে। প্রোগ্রামে ডেটা নিয়ে কাজ করার সময় প্রতিটি ডেটার জন্য একটি ভেরিয়েবল বা চলক ব্যবহার করা হয়।

### ধ্রুবক কী?

প্রোগ্রাম নির্বাহের সময় যার মান অপরিবর্তীত থাকে তাকে ধ্রুবক বলে। যেমন  $\pi$  এর মান হলো বা ৩.১৪২৮৫। কখনো  $\pi$  এর মানের কোনো পরিবর্তন হয় না।

### কী-ওয়ার্ড কী?

প্রত্যেক প্রোগ্রামিং ভাষার নিজস্ব কিছু সংরক্ষিত শব্দ আছে যা প্রোগ্রাম রচনার সময় ব্যবহার করা হয়। এই সংরক্ষিত শব্দগুলোকে কী-ওয়ার্ড বলা হয়।

### ডেটা টাইপ কী?

ডেটার ধরনকে ডেটা টাইপ বলে। যেমন- int, float, double ইত্যাদি।

### অপারেটর কী?



প্রোগ্রামিং ভাষায় গাণিতিক এবং যৌক্তিক কাজ নিয়ন্ত্রণ করার জন্য কতগুলো বিশেষ চিহ্ন ব্যবহৃত হয়, এগুলোকে অপারেটর বলা হয়। যেমন- +, -, \*, % ইত্যাদি।

### কন্ট্রোল স্টেটমেন্ট কী?

সি প্রোগ্রামের স্টেটমেন্টসমূহ সাধারণত পর্যায়ক্রমিকভাবে নির্বাহ হয়। কিন্তু কখনও কখনও স্টেটমেন্টসমূহ দুই বা ততোধিকবার নির্বাহ এবং শর্ত সাপেক্ষে নির্বাহের প্রয়োজন হয়। এক্ষেত্রে স্টেটমেন্টসমূহের নির্বাহ প্রোগ্রামার কর্তৃক নিয়ন্ত্রিত হয়। যে স্টেটমেন্টের সাহায্যে এরূপ স্টেটমেন্টসমূহের নির্বাহ নিয়ন্ত্রণ করা হয় তাকে কন্ট্রোল স্টেটমেন্ট বলে।

### লুপ কন্ট্রোল স্টেটমেন্ট কী?

সি প্রোগ্রামে কোন স্টেটমেন্ট একাধিকবার নির্বাহের জন্য যে কন্ট্রোল স্টেটমেন্ট ব্যবহৃত হয় তাকে লুপ কন্ট্রোল স্টেটমেন্ট বলে।

### লুপ কী?

প্রোগ্রামের অংশ বিশেষ কোনো শর্তে না পৌঁছা পর্যন্ত নির্দিষ্ট সংখ্যক বার পুনরাবৃত্তি করাকে লুপিং বা লুপ বলা হয়।

### অ্যারে কী?

অ্যারে হলো একই ধরনের বা সমপ্রকৃতির চলকের সমাবেশ।

### ফাংশন কী?

প্রোগ্রামে যখন কতগুলো স্টেটমেন্ট কোনো নামে একটি ব্লকের মধ্যে অবস্থান করে কোনো নির্দিষ্ট কাজ সম্পাদন করে তখন ব্লকটিকে ফাংশন বলা হয়।