



# প্রোগ্রামিং ভাষা

Programming Language



অধ্যয়ন স্থান

- প্রোগ্রাম
- অনুবাদক
- অ্যালগরিদম
- ফ্লোচার্ট
- ডেটা টাইপ
- চলক
- ধুবক
- অপারেটর
- লুপিং
- অ্যারে

আমরা কম্পিউটারে অসংখ্য বৈচিত্র্যময় কাজ করে থাকি। লেখালেখির কাজ, গান শোনা, ভিডিও দেখা, গেইম খেলা, গ্রাফিক্স আরও কত কী। এই সকল কাজ করার জন্য আমরা ভিন্ন ভিন্ন সফটওয়্যার বা প্রোগ্রাম ব্যবহার করি। এই প্রোগ্রামগুলো তৈরি হলো কীভাবে? মূলত প্রোগ্রাম তৈরি করা হয় প্রোগ্রামিং ভাষার সাহায্যে। এ অধ্যায়ে আমরা প্রোগ্রামিং ভাষা সম্পর্কে বিস্তারিত জানব এবং 'সি' প্রোগ্রামিং ভাষা আয়ত্ত করব।



## এ অধ্যায়ের পাঠগুলো পড়ে যা যা শিখব

- প্রোগ্রামের ধারণা
- বিভিন্ন স্তরের প্রোগ্রামিং ভাষা
- ব্যবহারিক : প্রোগ্রামের সংগঠন প্রদর্শন
- প্রোগ্রাম অ্যালগরিদম ও ফ্লোচার্ট প্রস্তুত করা
- 'সি' প্রোগ্রামিং ভাষা ব্যবহার করে প্রোগ্রাম প্রস্তুত করা

## পাঠ পরিকল্পনা

পাঠ ১, ২	প্রোগ্রামের ধারণা ও প্রোগ্রামের ভাষা
পাঠ ৩, ৪	বিভিন্ন উচ্চস্তরের ভাষা সম্পর্কে আলোচনা
পাঠ ৫	চতুর্থ প্রজন্মের ভাষা
পাঠ ৬	প্রোগ্রামের সংগঠন ও প্রোগ্রাম তৈরির ধাপসমূহ
পাঠ ৭-১০	অ্যালগরিদম ও ফ্লোচার্ট
পাঠ ১১, ১২	প্রোগ্রাম ডিজাইন মডেল
পাঠ ১৩, ১৪	'সি' প্রোগ্রাম
পাঠ ১৫, ১৬	'সি' ভাষায় ব্যবহৃত ডেটা টাইপ
পাঠ ১৭-১৯	'সি' ভাষায় ব্যবহৃত ধুবক ও চলক
পাঠ ২০	রাশিমালা ও কি-ওয়ার্ড
পাঠ ২১-২৩	ইনপুট / আউটপুট স্টেটমেন্ট
পাঠ ২৪	ব্যবহারিক নির্দেশাবলী ও কিছু প্রোগ্রাম প্রক্যাটস
পাঠ ২৫, ২৬	কন্ট্রোল ও কন্ডিশনাল স্টেটমেন্ট
পাঠ ২৭, ২৮	লুপ ও লুপের ব্যবহার
পাঠ ২৯, ৩০	অ্যারে ও অ্যারের ব্যবহার
পাঠ ৩১, ৩২	ফাংশন ও ফাংশনের ব্যবহার

## পাঠ ১ ও ২

## প্রোগ্রামের ধারণা ও প্রোগ্রামের ভাষা

## ৫.১ প্রোগ্রামের ধারণা (Concept of Program)

কম্পিউটার ব্যবহার করার মূল উদ্দেশ্য হচ্ছে সমস্যা সমাধান করা। কোন সমস্যা সমাধানের জন্য কম্পিউটারের বোধগম্য ভাষায় নির্দেশ লেখা হয় যা সোর্স কোড নামে পরিচিত। এরূপ সারিবদ্ধ সুশৃঙ্খল এক গুচ্ছ নির্দেশ মালার সমষ্টিকে প্রোগ্রাম বলে। কম্পিউটার এ নির্দেশমালা সমূহকে পর্যায়ক্রমিকভাবে পালনের মাধ্যমে নির্দিষ্ট সমস্যার সমাধান করে। কম্পিউটারের মাধ্যমে সমস্যা সমাধানের উদ্দেশ্যে ব্যবহৃত কতগুলো নির্দেশের সমষ্টিকে বলা হয় প্রোগ্রাম। কম্পিউটার প্রোগ্রাম তৈরির বিশেষ ধরনের কৌশলকে কম্পিউটার প্রোগ্রামিং বলে। যিনি প্রোগ্রামিং করেন তাকে প্রোগ্রামার বলে। প্রোগ্রামার যে ভাষায় প্রোগ্রামিং করে তাকে বলা হয় প্রোগ্রামের ভাষা বা প্রোগ্রামিং ভাষা। মূলত ধারাবাহিকভাবে লিখিত কমান্ড বা নির্দেশসমূহকেই প্রোগ্রাম বলা হয়।

বিখ্যাত ইংরেজ কবি লর্ড বাইরনের কন্যা অ্যাডা লাভলেস (Ada Lovelace) বারনোলি নম্বর (Bernoulli Number) ব্যবহার করে ধারাবাহিকভাবে হিসাবের জন্য একটি প্রোগ্রাম রচনা করেন। তাঁর লিখিত প্রোগ্রামটি প্রথম প্রোগ্রাম হিসেবে বিবেচিত হয় এবং এজন্য তাঁকে প্রথম প্রোগ্রামার বলা হয়। পরবর্তীতে এ কাজের স্বীকৃতি স্বরূপ অ্যাডা লাভলেস নামে একটি প্রোগ্রামিং ভাষার নামকরণ করা হয় যা এডা (Ada) নামে পরিচিত।



চিত্র : অ্যাডা লাভলেস  
[জন্ম: ১০ ডিসেম্বর, ১৮১৫  
মৃত্যু: ২৭ নভেম্বর, ১৮৫২]

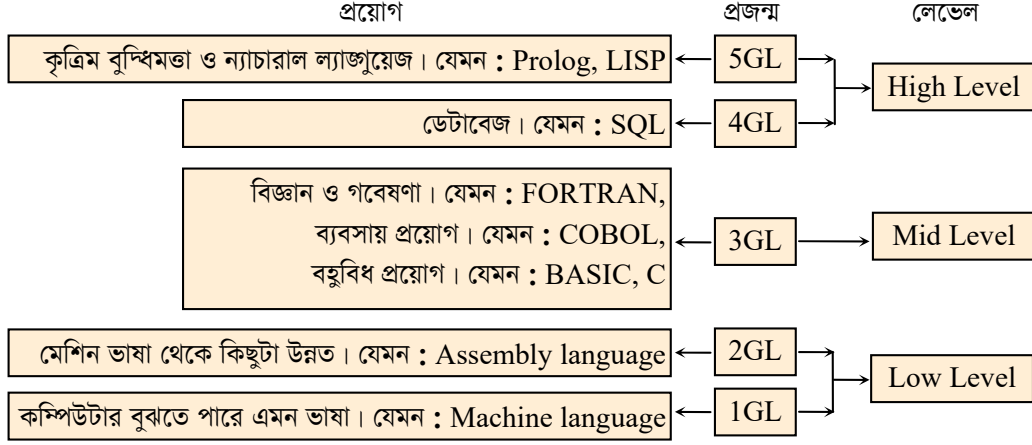
## ৫.২ প্রোগ্রামের ভাষা (Programming Language)

মনেরভাব প্রকাশের মাধ্যম হচ্ছে ভাষা। দৈনন্দিন জীবনে আমরা পরস্পরের সাথে যোগাযোগ বা বিভিন্ন রকম সমস্যা সমাধানের জন্য নানা রকম ভাষা ব্যবহার করি। আমরা একে অপরের সাথে ভাব বিনিময়ের জন্য যেমন বিভিন্ন ভাষা ব্যবহার করি, তেমনি কম্পিউটারকে আমাদের প্রয়োজনীয় নির্দেশাবলী দেয়ার জন্য বিশেষ ধরনের ভাষা ব্যবহার করা হয়। মানুষের ভাষা কম্পিউটার বুঝতে পারেনা, কম্পিউটার বোঝে কম্পিউটারের নিজস্ব ভাষা। এসব ভাষাই মূলত প্রোগ্রামের ভাষা। কম্পিউটারের ভাষায় ব্যবহৃত বর্ণ হচ্ছে ০ এবং ১। কম্পিউটার বুঝতে পারে এমন কিছু নির্দিষ্ট সংকেত ও চিহ্ন ব্যবহার করে বিশেষ নিয়মানুসারে সাজিয়ে প্রোগ্রাম লিখতে হয়। প্রোগ্রাম তৈরির জন্য ব্যবহৃত এ সকল সংকেত, চিহ্ন ও নিয়মগুলোকে একত্রে প্রোগ্রামের ভাষা বলে। যেমন মানুষের মধ্যে ভাষার ভিন্নতা আছে, তেমনি কম্পিউটারেও অনেক ধরনের ভাষা ব্যবহৃত হয়। তবে ভাষা যাই হোক না কেন, তা ০ এবং ১ দিয়েই কম্পিউটারকে বোঝাতে হয়।

কম্পিউটারের সাথে মানুষের যোগাযোগের মাধ্যম হচ্ছে কম্পিউটারের ভাষা। কম্পিউটারের ভাষা হচ্ছে মূলত প্রোগ্রামের ভাষা। প্রোগ্রামের ভাষা বলতে আমরা বুঝি কতগুলো নির্দেশ যা কম্পিউটারকে নিয়ন্ত্রণ করবে এবং কম্পিউটার কী ধরনের কাজ করবে, ডেটা কোথায় রাখবে, ফলাফল কী হবে ইত্যাদি নির্ধারণ করা।

বৈশিষ্ট্য অনুযায়ী প্রোগ্রামের ভাষাকে পাঁচটি প্রজন্মে (Generation) ভাগ করা যায়। যথা—

১. প্রথম প্রজন্মের ভাষা (১৯৪৫) : মেশিন ভাষা (Machine language)
২. দ্বিতীয় প্রজন্মের ভাষা (১৯৫০) : অ্যাসেম্বলি ভাষা (Assembly language)
৩. তৃতীয় প্রজন্মের ভাষা (১৯৬০) : উচ্চতর ভাষা (High level language)
৪. চতুর্থ প্রজন্মের ভাষা (১৯৭০) : অতি উচ্চতর ভাষা (Very high level language)
৫. পঞ্চম প্রজন্মের ভাষা (১৯৮০) : স্বাভাবিক বা ন্যাচারাল ভাষা (Natural language)।



চিত্র : বিভিন্ন প্রজন্মের ভাষা

### ৫.৩ মেশিন ভাষা বা যান্ত্রিক ভাষা (Machine Language)

সর্বপ্রথম ১৯৪৫ সালে মেশিন ভাষা চালু হয়। এটিকে কম্পিউটারের প্রথম প্রজন্মের ভাষা বলা হয়। কম্পিউটারের নিজস্ব ভাষা হচ্ছে মেশিন ভাষা। এটি কম্পিউটারের মৌলিক ভাষা। মেশিন ভাষায় শুধুমাত্র ০ এবং ১ ব্যবহার করা হয় বলে কোনো নির্দেশ কম্পিউটার সরাসরি বুঝতে পারে। এটির মাধ্যমে সরাসরি কম্পিউটারের সাথে যোগাযোগ করা যায়। মেশিন ভাষায় বিট, বাইট ও মেমোরি অ্যাড্রেস ব্যবহার করা হয়। সুতরাং মেশিন ভাষায় যেসব নির্দেশ দেওয়া হয় তাদের চার ভাগে ভাগ করা যায়।

- যেমন-
- |                                   |  |
|-----------------------------------|--|
| ১. গাণিতিক (Arithmetic)           | : যোগ, বিয়োগ, গুণ, ভাগ                    |
| ২. নিয়ন্ত্রণ (Control)           | : লোড (Load), স্টোর (Store) ও জাম্প (Jump) |
| ৩. ইনপুট-আউটপুট (input output)    | : পড় (Read) ও লেখ (Write)                 |
| ৪. প্রত্যক্ষ ব্যবহার (Direct use) | : আরম্ভ (Start), থামা (Halt) ও শেষ (End)   |

#### সুবিধা:

১. মেশিন ভাষার সবচেয়ে বড় সুবিধা হচ্ছে সরাসরি কম্পিউটারের সাথে যোগাযোগ করা যায়।
২. মেশিন ভাষায় লেখা প্রোগ্রাম নির্বাহের জন্য অনুবাদক প্রোগ্রামের প্রয়োজন হয় না ফলে দ্রুত কাজ করে।
৩. মেশিন ভাষায় লিখিত প্রোগ্রামে অতি অল্প মেমোরি প্রয়োজন হয়।
৪. কম্পিউটারের ভিতরের গঠন ভালোভাবে বুঝতে হলে এই ভাষা জানতে হয়।

#### অসুবিধা:

১. মেশিন ভাষায় লিখিত কোনো প্রোগ্রাম সাধারণত বোঝা যায় না।
২. শুধু ০ ও ১ ব্যবহার করা হয় বলে প্রোগ্রাম লেখা কষ্টসাধ্য।
৩. এ ভাষায় প্রোগ্রাম লিখতে প্রচুর সময় লাগে এবং ভুল হবার সম্ভাবনা খুব বেশি থাকে।
৪. ভুল হলে তা বের করা এবং ত্রুটিমুক্ত করা খুব কঠিন।
৫. এক ধরনের কম্পিউটারের জন্য লিখিত প্রোগ্রাম অন্য ধরনের কম্পিউটারে ব্যবহার করা যায় না।

## ৫.৪ অ্যাসেম্বলি ভাষা (Assembly Language)

অ্যাসেম্বলি ভাষার প্রচলন শুরু হয় ১৯৫০ সাল থেকে। অ্যাসেম্বলি ভাষাকে সাংকেতিক (Symbolic) ভাষাও বলা হয়। দ্বিতীয় প্রজন্মের কম্পিউটারে এই ভাষা ব্যাপকভাবে প্রচলিত ছিল। মেশিন ভাষার মতো অ্যাসেম্বলি ভাষায় ০ ও ১ ব্যবহার না করে কতকগুলো বিটের সমষ্টি নিয়ে গঠিত ইংরেজি বর্ণের সাহায্যে বিশেষ কোডে কম্পিউটারকে নির্দেশ দেওয়া হয়। অ্যাসেম্বলি ভাষার ক্ষেত্রে নির্দেশ ও ডেটার অ্যাড্রেস বাইনারি বা হেক্সা সংখ্যার সাহায্যে না দিয়ে সংকেতের সাহায্যে দেওয়া হয়। এই সংকেতকে বলে সাংকেতিক কোড (Symbolic Code) বা নেমোনিক (Nemonic)। এটি অনেকটা সহজবোধ্য।

যেমন: 'যোগ' বা Addition করাকে লেখা হয় ADD  
 'বিয়োগ' বা Subtraction করাকে লেখা হয় SUB  
 'গুণ' বা Multiply কে লেখা হয় MUL  
 'ভাগ' বা Division কে লেখা হয় DIV ইত্যাদি।

অ্যাসেম্বলি ভাষায় প্রতিটি নির্দেশের চারটি অংশ থাকে। ক. লেবেল, খ. অপ-কোড, গ. অপারেন্ড ও ঘ. কমেন্ট।

লেবেল	অপকোড	অপারেন্ড	কমেন্ট
-------	-------	----------	--------

**লেবেল:** লেবেলে নির্দেশের সাংকেতিক ঠিকানা থাকে। যেমন- জাম্পের সময় পরবর্তী নির্দেশের ঠিকানা লেবেলে দেওয়া হয়, তবে লেবেল সব সময় নাও থাকতে পারে। লেবেলের ১-২টি অ্যালফানিউমেরিক বর্ণ থাকে। এই বর্ণের মধ্যে কোন ফাঁক থাকে না। নির্দেশ নেমোনিক (যেমন LDA) ও রেজিস্টারের নাম লেবেল হিসেবে ব্যবহার করা যায় না।

**অপকোড:** এতে নির্দেশ নেমোনিক থাকে। এ নেমোনিকগুলো বিভিন্ন কম্পিউটারে বিভিন্ন হতে পারে, নিম্নে উল্লেখ করা হলো।

নির্দেশ নেমোনিক	পূর্ণরূপ ও উচ্চারণ	ব্যাখ্যা
LDA	Load Accumulator- লোড অ্যাকিউমুলেটর	প্রধান মেমোরির কোনো নির্দিষ্ট অবস্থানের সংখ্যা অ্যাকিউমুলেটরে রাখার নির্দেশ দেওয়া হয়।
ADD	ADD-অ্যাড	দুটি অপারেন্ড-এর মধ্যে যোগ করার নির্দেশ বুঝানো হয়।
CLR	CLEAR-ক্লিয়ার	অ্যাকিউমুলেটর খালি করার কমান্ড।
STA	Store Accumulator- স্টোর অ্যাকিউমুলেটর	অ্যাকিউমুলেটরে ডেটা সংরক্ষণ করার নির্দেশ।
SUB	SUBtract-বিয়োগ	দুটি অপারেন্ড-এর মধ্যে বিয়োগ করার নির্দেশ বুঝানো হয়।
MUL	MULTiply-গুণ	দুটি অপারেন্ড এর মধ্যে গুণ করার নির্দেশ বুঝানো হয়।
DIV	DIVide-ভাগ	দুটি অপারেন্ড এর মধ্যে ভাগ করার নির্দেশ বুঝানো হয়।
OR	OR-অর	দুটি অপারেন্ড এর মধ্যে লজিক্যাল অর অপারেশন বুঝায়।
JMU	JUMP-জাম্প	নিঃশর্তভাবে প্রোগ্রামের নির্দিষ্ট স্থানে যাওয়ার নির্দেশ।
INP	INPUT-ইনপুট	ডেটা বা নির্দেশ গ্রহণ করে মেমোরির নির্দিষ্ট স্থানে রাখা।
OUT	OUTPUT-আউটপুট	মেমোরির কোন নির্দিষ্ট বিষয়কে আউটপুটে পাঠানোর নির্দেশ।
STP	STOP-থামা	প্রোগ্রামকে থামানোর নির্দেশ।

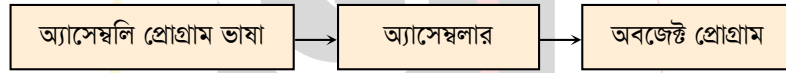
**অপারেভ:** অপকোড যার উপর কাজ করে তাকে অপারেভ বলে। অপারেভের অবস্থানের ঠিকানা বুঝানোর জন্য সাধারণত আলফানিউমেরিক বর্ণ ব্যবহার করা হয়। যেমন— A, B, X, Y, AM, XY ইত্যাদি।

**মন্তব্য:** কমেন্ট বা মন্তব্য নির্দেশের কোন অংশ নয়। মন্তব্য আসলে প্রত্যেক নির্দেশের ব্যাখ্যা যা পরবর্তি প্রোগ্রামার বা অন্য কেউ প্রোগ্রামের সঠিক অর্থ সহজে বুঝতে পারে। অপারেভ ফিল্ডের পর কোলন (:) বা সেমিকোলন (;) দিয়ে মন্তব্য লেখা যায়।

**উদাহরণ:** A ও B যোগ করে C অবস্থানে রাখতে হবে। এখানে A বা B এর অবস্থানের অ্যাড্রেসকেও যথাক্রমে A বা B বলা হয়। নিচে A ও B যোগ করে C অবস্থানে রাখার জন্য অ্যাসেম্বলি ভাষার প্রোগ্রাম দেওয়া হলো।

CLR	অ্যাকিউমুলেটর খালি করা।
INP: A	A সংখ্যাটিকে ইনপুট থেকে প্রধান মেমোরি A অবস্থানে রাখা।
INP: B	B সংখ্যাটিকে ইনপুট থেকে প্রধান মেমোরি B অবস্থানে রাখা।
LDA: A	অ্যাকিউমুলেটরে A রাখা।
ADD: B	B কে অ্যাকিউমুলেটরের সংখ্যার সাথে যোগ করে যোগফল অ্যাকিউমুলেটরে রাখা।
STA : C	অ্যাকিউমুলেটরের সংখ্যা C অবস্থানে রাখা।
OUT : C	ফলাফল C চলকের মাধ্যমে প্রদর্শন করা।
STP	থামা।

অ্যাসেম্বলি ভাষায় লিখিত প্রোগ্রাম কম্পিউটারে সরাসরি বুঝতে পারে না। এজন্য এ জাতীয় প্রোগ্রামকে মেশিন ভাষায় রূপান্তর করতে হয়। এ রূপান্তরের কাজে বিশেষ প্রোগ্রাম ব্যবহার করা হয়। যে প্রোগ্রামের সাহায্যে অ্যাসেম্বলি ভাষাকে মেশিন ভাষায় রূপান্তর করা হয় তাকে অ্যাসেম্বলার বলে। নিম্নের চিত্রে অ্যাসেম্বলি ভাষায় প্রোগ্রাম নির্বাহ প্রক্রিয়া দেখানো হলো—



#### অ্যাসেম্বলি ভাষার সুবিধা:

- অ্যাসেম্বলি ভাষায় প্রোগ্রাম রচনা করা যান্ত্রিক ভাষার তুলনায় অনেক সহজ।
- প্রোগ্রাম রচনা করতে কম সময় লাগে।
- মেমোরি অ্যাড্রেসের বিবরণের দরকার হয় না।
- প্রোগ্রাম পরিবর্তন করা সহজ।
- মেশিনের অভ্যন্তরীণ গঠন সম্পর্কে জানা যায়।

#### অ্যাসেম্বলি ভাষার অসুবিধা:

- প্রোগ্রাম রচনার সময় প্রোগ্রামারকে মেশিন সম্পর্কে ধারণা থাকতে হয়।
- ভুল ত্রুটি বের করা কষ্টসাধ্য ব্যাপার।
- ভিন্ন ভিন্ন মেশিনে ভিন্ন ভিন্ন অ্যাসেম্বলি ভাষা ব্যবহার করতে হয়।
- অনুবাদক প্রোগ্রামের প্রয়োজন হয়।
- সরাসরি মেশিন বুঝতে পারে না।

### যান্ত্রিক ভাষা ও অ্যাসেম্বলি ভাষার মধ্যে পার্থক্য:

পার্থক্যের বিষয়	যান্ত্রিক ভাষা	অ্যাসেম্বলি ভাষা
১. সংজ্ঞা	বাইনারি সংখ্যা ০ ও ১ দিয়ে তৈরি ভাষাকে যান্ত্রিক ভাষা বলে।	সংক্ষিপ্ত সাংকেতিক চিহ্ন বা সহায়ক নাম দিয়ে লিখিত ভাষাকে অ্যাসেম্বলি ভাষা বলে।
২. প্রোগ্রাম রচনা	যান্ত্রিক ভাষা সংক্ষিপ্ত আকারে লেখা যায়।	এ ভাষা খুব কম সময়ে রচনা করা যায়।
৩. নির্ভর	এ ভাষা কম্পিউটার নির্ভর ভাষা।	এ ভাষা যন্ত্রনির্ভর ভাষা।
৪. ভিন্ন ভিন্ন কম্পিউটার	এ ভাষায় এক কম্পিউটারের জন্য লিখিত প্রোগ্রাম অন্য কম্পিউটারে চালানো যায় না।	ভিন্ন ভিন্ন যন্ত্রের জন্য ভিন্ন ভিন্ন অ্যাসেম্বলি ভাষা ব্যবহৃত হয়।

### ৫.৫ মধ্যম স্তরের ভাষা (Mid Level Language)

যে প্রোগ্রামিং ভাষার মধ্যে লো-লেভেল ও হাই-লেভেল উভয় ভাষার বৈশিষ্ট্য বিদ্যমান তাকে মধ্যম স্তরের ভাষা বলে। কম্পিউটারের হার্ডওয়্যার নিয়ন্ত্রণ ও সিস্টেম প্রোগ্রাম রচনার জন্য বিট পর্যায়ে প্রোগ্রামিং ভাষা হচ্ছে মধ্যম স্তরের ভাষা। এ ভাষায় উচ্চতর ভাষার সুবিধা পাওয়া যায়। আবার নিম্নস্তরের ভাষায়ও প্রোগ্রাম রচনা করা যায়। মধ্যম স্তরের কয়েকটি ভাষা হলো— C, FORTH, Macro-Assembler।

#### মধ্যম স্তরের ভাষার সুবিধা:

১. এই ভাষায় লিখিত প্রোগ্রাম বোঝা প্রোগ্রামারদের কাছে সহজসাধ্য।
২. যেকোনো ধরনের কম্পিউটারে নির্বাহ করা সম্ভব।
৩. একবার লিখিত প্রোগ্রাম পরবর্তীতে পরিবর্তন করা সহজ।
৪. ভুল হওয়ার সম্ভাবনা কম থাকে এবং ভুল সংশোধন করা সম্ভব।

#### মধ্যম স্তরের ভাষার অসুবিধা:

১. অনুবাদক প্রোগ্রামের প্রয়োজন হয়।
২. এ ভাষা মেশিন সরাসরি বুঝতে পারে না।
৩. প্রোগ্রাম নির্বাহ করতে মেশিন ভাষার তুলনায় বেশি সময় লাগে।
৪. উচ্চতর ভাষার তুলনায় এই ভাষা কঠিন।



#### কাজ :

আইসিটি স্যার ক্লাসে বললেন— অনেক আগে ০ এবং ১ ব্যবহার করে প্রোগ্রাম লেখা হত। আবার বিভিন্ন প্রকার সাংকেতিক চিহ্ন ব্যবহার করেও প্রোগ্রাম লেখা হত।

প্রশ্ন: উক্ত প্রোগ্রামিং ভাষা দুটির মধ্যে তুলনামূলক পার্থক্য বিশ্লেষণ কর।



## ৫.৬ উচ্চস্তরের ভাষা (High Level Language)

মেশিন ও অ্যাসেম্বলি ভাষায় লিখিত প্রোগ্রাম অন্য কোনো কম্পিউটারে ব্যবহার করা যায় না। এমনকি এ ধরনের প্রোগ্রাম সহজে বোঝা কষ্টকর। এসকল অসুবিধা থেকে রক্ষার জন্য পরবর্তিতে উচ্চ স্তরের ভাষার উদ্ভব ঘটে। উচ্চতর ভাষা বা হাই লেভেল ভাষার সাথে মানুষের ভাষার (যেমন: ইংরেজি) মিল আছে। হাই লেভেল ভাষা মানুষ সহজেই দ্রুত লিখতে, বলতে ও মনে রাখতে পারে। এই স্তরের ভাষায় লিখিত প্রোগ্রাম বিভিন্ন ধরনের মেশিনে ব্যবহার করা সম্ভব। অর্থাৎ, এই ভাষা কম্পিউটার সংগঠনের নিয়ন্ত্রণের উদ্দেশ্যে, এ কারণে এসব ভাষাকে উচ্চতর ভাষা বলা হয়। এটি মানুষ সহজে বুঝতে পারলেও কম্পিউটার সরাসরি বুঝতে পারে না বলে অনুবাদক প্রোগ্রামের সাহায্যে একে মেশিন ভাষায় রূপান্তরিত করে নিতে হয়। উদাহরণ: Qbasic, Pascal, C/C++, JAVA ইত্যাদি।

### ৫.৬.১ উচ্চস্তরের ভাষার প্রকারভেদ (Classifications of High Level Language)

বিভিন্ন ভাষার প্রয়োগ ক্ষেত্র বিভিন্ন। প্রয়োগের ভিত্তিতে উচ্চস্তরের ভাষাকে সাধারণত নিম্নলিখিত দুই ভাগে ভাগ করা যায়।

- সাধারণ কাজের ভাষা (General Purpose Language)
- বিশেষ কাজের ভাষা (Special Purpose Language)

যেসব ভাষা সব ধরনের কাজের উপযোগী করে তৈরি করা হয় তা সাধারণ কাজের ভাষা নামে পরিচিত। যেমন BASIC, PASCAL, C ইত্যাদি। আর যেসব ভাষা বিশেষ বিশেষ কাজের উপযোগী করে তৈরি করা হয় তা বিশেষ কাজের ভাষা নামে পরিচিত। যেমন— COBOL, ALGOL, FORTRAN ইত্যাদি।

**উচ্চস্তরের ভাষার ব্যবহার:**

- বড় প্রোগ্রাম তৈরি করতে
- বৃহৎ ডেটা প্রসেসিং এর প্রোগ্রাম তৈরি করতে
- যেসব ক্ষেত্রে প্রচুর মেমোরির প্রয়োজন সেসব ক্ষেত্রের সফটওয়্যার তৈরি করতে
- জটিল গাণিতিক হিসাব নিকাশের সফটওয়্যার তৈরি করতে
- অ্যাপ্লিকেশন প্যাকেজ সফটওয়্যার তৈরি করতে
- বিভিন্ন ধরনের অটোমেটিক প্রসেস কন্ট্রোল করতে।

**সুবিধা:** ১. উচ্চস্তরের ভাষায় প্রোগ্রাম লেখা সহজ ও লিখতে কম সময় লাগে।

২. ভুল হবার সম্ভাবনা কম থাকে এবং প্রোগ্রামের ত্রুটি সংশোধন করাও সহজ।

৩. এ ভাষায় প্রোগ্রাম লেখার জন্য কম্পিউটারের ভিতরের সংগঠন সম্পর্কে ধারণা থাকার প্রয়োজন নেই।

৪. এক মডেলের কম্পিউটারের জন্য লিখিত প্রোগ্রাম অন্য মডেলের কম্পিউটারে চলে।

**অসুবিধা:** ১. উচ্চস্তরের ভাষায় সরাসরি কম্পিউটারের সাথে যোগাযোগ করা যায় না।

২. প্রোগ্রামকে অনুবাদ করে কম্পিউটারকে বুঝিয়ে দিতে হয়।

৩. বেশি মেমোরি প্রয়োজন হয়।

নিচে কিছু উচ্চস্তরের ভাষা সম্পর্কে আলোচনা করা হলো:

### ৫.৬.২ সি (C)

১৯৭০ সালে আমেরিকার বেল ল্যাবরেটরির গবেষক ডেনিস রিচি C ভাষা উদ্ভাবন করেন। এটি সিস্টেম প্রোগ্রাম তৈরিতে বেশ জনপ্রিয়। “C” কে কম্পিউটার ভাষার জনক বলা হয়। এ ভাষার অনেক সংস্করণ রয়েছে। যেমন: C, ANSI C, Turbo C, Visual C ইত্যাদি। পরবর্তী পাঠে C ভাষা নিয়ে বিস্তারিত আলোচনা করা হয়েছে।

```
#include<stdio.h>
void main(){
    int a, b, sum;
    printf("Enter two number :");
    scanf("%d%d",&a,&b);
    sum=a+b;
    printf("Sum=%d",sum);
}
```

চিত্র: একটি সি প্রোগ্রাম

### ৫.৬.৩ সি++ (C++)

C ভাষায় নতুন বৈশিষ্ট্য ও সুবিধা প্রদান করে পরবর্তী সংস্করণে C++ ভাষায় পরিণত হয়। ১৯৮০ সালে বিয়ারনে স্ট্রাউসট্রুপ যুক্তরাষ্ট্রের (Bjarne Stroustrup) AT & T Bell Laboratory তে এটি তৈরি করেন। এতে নতুন যে সুবিধা সংযোজন করা হয় তা হচ্ছে অবজেক্ট ওরিয়েন্টেড ফিচার। মূলত ১৯৭৯ সাল থেকে এর গবেষণা শুরু হয় ‘C উইথ ক্লাশেস’ নামে। এ কারণে সি ++ কে সি-এর বর্ধিত সংস্করণ বা সুপারসেট বলা হয়। টেক্সট এডিটর তৈরি, কম্পাইলার ও ইন্টারপ্রেটার তৈরি, ডেটাবেজ হ্যান্ডলিং, কমিউনিকেশন সিস্টেম ডিজাইন, ডিস্ট্রিবিউটেড সিস্টেম ডিজাইন, রিয়েল-টাইম সিস্টেম ডিজাইন ও উইন্ডোভিত্তিক অ্যাপ্লিকেশনসমূহ সি++ এর অনন্য অবদান। সি++ দিয়ে তৈরি করা হয়েছে এক্স উইন্ডো সিস্টেম, কিউট ইত্যাদির মত গ্রাফিক্যাল ডিসপ্লে ম্যানেজমেন্ট প্রোগ্রামিং। অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং এর দুনিয়ায় একটি কিংবদন্তি এবং জাভার সৃষ্টির প্রেরণা হলো সি++। মজিলা ফায়ারফক্স আর ক্রোমিয়াম ব্রাউজারও কিন্তু বেশিরভাগই সি++ এ লেখা। ১৯৮৫ সাল থেকে এর বাণিজ্যিক ব্যবহার শুরু হয়।

```
#include<iostream.h>
void main(){
    int a, b, sum;
    cout<<"Enter two number:";
    cin>>a>>b;
    sum=a+b;
    cout<<"Sum="<<sum;
}
```

চিত্র: একটি সি++ প্রোগ্রাম

### ৫.৬.৪ ভিজুয়াল বেসিক (Visual Basic)

বিশ্বখ্যাত সফটওয়্যার নির্মাতা প্রতিষ্ঠান মাইক্রোসফট ১৯৯১ সালে উইন্ডোজ অপারেটিং সিস্টেমে কাস্টমার অ্যাপ্লিকেশন সফটওয়্যার তৈরির জন্য ভিজুয়াল বেসিক নির্মাণ করে। এটি মূলত বেসিক প্রোগ্রামিং ভাষার “গ্রাফিক্যাল ইউজার ইন্টারফেস” ভাষা। ভিজুয়াল প্রোগ্রামিং -এ ইভেন্ট ড্রাইভেন ফিচার সংযোজন করা হয়, ফলে যে কেউ ইচ্ছা করলে ভিজুয়াল বেসিক ব্যবহার করে অল্প সময়ে কাস্টমাইজড অ্যাপ্লিকেশন সফটওয়্যার তৈরি করতে পারে। পরবর্তীতে এর অনেক ভার্শন তৈরি হয়। বর্তমানে VB.Net হচ্ছে ভার্সুয়াল রিয়েলিটি সমসাময়িক ভার্শন।

```
Sub Main()
    Dim firstNum As Integer
    Dim secondNum As Integer
    Dim sum As Integer
    Console.WriteLine("Enter first number:")
    firstNum = Console.ReadLine
    Console.WriteLine("Enter second number:")
    secondNum = Console.ReadLine
    sum = firstNum + secondNum
    Console.WriteLine("Sum=" & sum)
    Console.ReadLine()
End Sub
```

চিত্র: একটি ভিজুয়াল বেসিক প্রোগ্রাম



### ৫.৬.৫ জাভা (Java)

সান মাইক্রোসিস্টেমের তৈরি অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং ভাষা হচ্ছে জাভা। এটি ক্লাস বেজড ও সম্পূর্ণ অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং ভাষা। ১৯৯১ সালের শেষের দিকে জেমস গসলিং -এর নেতৃত্বে একদল বিশেষজ্ঞ কর্তৃক জাভা ভাষার উৎপত্তি। প্রথমে এর নাম ছিলো OAK। মূলত এটি OAK প্রোগ্রামিং ভাষার পরবর্তী সংস্করণ যা ১৯৯৫ সালের জুন মাসে বাজারে আসে। সান মাইক্রোসিস্টেম OAK প্রোগ্রামিং ভাষা তৈরি করেছিলেন মূলত হ্যান্ডহেল্ড ইলেক্ট্রনিক ডিভাইসের (হাতে বহনকারী ইলেক্ট্রনিক যন্ত্র) জন্য। কারণবশত এটি জনপ্রিয়তা পায় নি। পরবর্তীতে নাম পরিবর্তন করে এতে WWW এর ফিচারসমূহ সংযোজন করা হয়। ফলে ইন্টারনেটভুক্ত যন্ত্রসমূহে জাভা প্লাটফর্ম নামে নতুন একটা পরিবেশ তৈরি হয়। যা সব ধরনের অপারেটিং সিস্টেম সাপোর্ট করে।

```
import java.util.Scanner;
public class JavaProgram
{
    public static void main(String args[])
    {
        int a, b, sum;
        Scanner scan = new Scanner(System.in);
        System.out.print("Enter Two Numbers:");
        a = scan.nextInt();
        b = scan.nextInt();
        sum = a + b;
        System.out.print("Sum=" + sum);
    }
}
```

চিত্র: একটি জাভা প্রোগ্রাম

### ৫.৬.৬ ওরাকল (Oracle)

ডেটাবেজ সংক্রান্ত সফটওয়্যার তৈরির সবচেয়ে বড় প্রতিষ্ঠান ওরাকল কর্পোরেশন। ১৯৭৭ সালে এড ওয়াটস ও বব মাইনারকে সাথে নিয়ে Oracle তৈরি করেন ল্যারি এলিসন। ওরাকল একটি RDBMS প্রোগ্রাম। সিকিউরিটির দিক দিয়ে সবচেয়ে শক্তিশালী ও বেশি ডেটা ধারণক্ষম RDBMS হচ্ছে ওরাকল।

```
DECLARE
Var1 integer;
Var2 integer;
Var3 integer;
BEGIN
Var1:=&var1;
Var2:=&var2;
Var3:=var1+var2;
DBMS_OUTPUT.PUT_LINE(Var3);
END;
```

চিত্র: একটি ওরাকল প্রোগ্রাম

### ৫.৬.৭ অ্যালগল (Algol)

১৯৫৮ সালে ইউরোপিয়ান ও আমেরিকান কম্পিউটার বিজ্ঞানীদের যৌথ উদ্যোগে অ্যালগল (ALGOritomic Language) প্রোগ্রামিং ল্যাঙ্গুয়েজ তৈরি হয়। এটি ব্যবহার হতো মূলত গবেষণার জন্য। অন্যান্য প্রোগ্রামিং ভাষাকে অ্যালগরিদমের বর্ণনার কাজে সহযোগিতা করত।

```
BEGIN
integer a;
integer b;
integer sum;
READ Int(a);
READ Int(b);
sum:=a + b;
PRINT Int(sum)
END
```

চিত্র: একটি অ্যালগল প্রোগ্রাম

### ৫.৬.৮ ফোরট্রান (Fortran)

উচ্চস্তরের ভাষার মধ্যে সবচেয়ে আদিমতম ভাষা হচ্ছে ফোরট্রান (FORmula TRANslation)। ১৯৫৩ সালে আইবিএম (IBM) এর গবেষক জন ব্যাকাস IBM 704 মেইন ফ্রেম কম্পিউটারের জন্য এটি তৈরি করেন। গাণিতিক জটিল হিসাব- নিকাশের সুবিধার জন্য তৈরি করা হয়েছিল। এটি প্রকৌশল সংক্রান্ত গবেষণার কাজে এখনো বেশ জনপ্রিয়।

```
INTEGER a,b,sum
PRINT *, 'Enter Two Numbers:'
READ *, a,b
sum = a + b
PRINT *, 'Sum=' , sum
END
```

চিত্র: একটি ফোরট্রান প্রোগ্রাম

### ৫.৬.৯ পাইথন (Python)

পাইথন একটি মাল্টিপ্যার্যাডিজম প্রোগ্রামিং ভাষা যা একই সাথে অবজেক্ট ওরিয়েন্টেড ও স্ট্রাকচার্ড ফিচার সাপোর্ট করে। ডাইনামিক ওয়েব অ্যাপ্লিকেশনসহ অনেক কিছু তৈরিতে পাইথন প্রোগ্রামিং ভাষা ব্যবহার করা হয়। ১৯৯১ সালে নেদারল্যান্ড-এর কম্পিউটার বিজ্ঞানী গুইডো ভ্যান রোসাম এই প্রোগ্রামিং ভাষা তৈরি করেন। ব্রিটিশ কমেডি শো “মাল্টি পাইথন” -এর নামে পাইথন ভাষার নামকরণ করেন।

```
print("Enter two numbers: ");
val1 = int(input());
val2 = int(input());
sum = val1 + val2;
print("Sum=",sum);
```

চিত্র: একটি পাইথন প্রোগ্রাম

### অ্যাসেম্বলি ভাষা ও উচ্চস্তরের ভাষার মধ্যে পার্থক্য

অ্যাসেম্বলি ভাষা	উচ্চস্তরের ভাষা
১. সংক্ষিপ্ত সাংকেতিক চিহ্ন বা Nemonic Code দিয়ে লিখিত ভাষাকে অ্যাসেম্বলি ভাষা বলে।	১. ইংরেজি ভাষা ব্যবহার করে যে প্রোগ্রাম তৈরি করা হয় তাকে উচ্চস্তরের ভাষা বলে।
২. এ ভাষার ভুল সহজে নির্ণয় করা যায় না।	২. এ ভাষার ভুল নির্ণয় করা সহজ।
৩. এ ভাষা যন্ত্রনির্ভর ভাষা।	৩. এ ভাষা ব্যবহারকারী নির্ভর ভাষা।
৪. ভিন্ন ভিন্ন যন্ত্রের জন্য ভিন্ন ভিন্ন অ্যাসেম্বলি ভাষা ব্যবহৃত হয়।	৪. এ ভাষায় লিখিত প্রোগ্রাম যে কোনো কম্পিউটারে ব্যবহার করা যায়।

### মেশিন ভাষা / নিম্নস্তরের ভাষা ও উচ্চস্তরের ভাষার মধ্যে পার্থক্য

মেশিন ভাষা / নিম্নস্তরের ভাষা	উচ্চস্তরের ভাষা
১. কম্পিউটার এ ভাষা সরাসরি বুঝতে পারে।	১. কম্পিউটার এ ভাষা সরাসরি বুঝতে পারে না।
২. এক মডেলের মেশিনের জন্য লিখিত প্রোগ্রাম অন্য কোনো মেশিন বুঝতে পারে না।	২. যে কোনো মডেলের জন্য লিখিত প্রোগ্রাম অন্য মডেলের মেশিন বুঝতে পারে।
৩. মেশিন ভাষার প্রোগ্রামারকে কম্পিউটারের লজিক্যাল গঠন সম্পর্কে ভাল জ্ঞান থাকতে হয়।	৩. উচ্চস্তরের ভাষার প্রোগ্রামারকে কম্পিউটারের লজিক্যাল গঠন সম্পর্কে জ্ঞান না থাকলেও হয়।
৪. মেশিন ভাষায় প্রোগ্রাম লেখা খুবই কঠিন ও সময় সাপেক্ষ।	৪. উচ্চস্তরের ভাষায় প্রোগ্রাম লেখা সহজ ও কম সময় প্রয়োজন হয়।
৫. সরাসরি বাইনারি ভাষায় প্রোগ্রাম লেখা হয়ে থাকে। ফলে অনুবাদের প্রয়োজন হয় না।	৫. মানুষের বোধগম্য ভাষায় প্রোগ্রাম লেখা হয়। তবে অনুবাদক প্রোগ্রাম দ্বারা বাইনারি ভাষায় রূপান্তর করে নিতে হয়।
৬. কম পরিমাণ লজিক ও মেমোরিতে এ ভাষায় লিখিত প্রোগ্রাম নির্বাহ করা যায়।	৬. এ ভাষায় লিখিত প্রোগ্রাম নির্বাহ করতে বেশি পরিমাণ লজিক ও মেমোরি প্রয়োজন হয়।



#### কাজ:

১. কোন স্তরের ভাষা বিভিন্ন ডিভাইসে ব্যবহার করতে অসুবিধা হয় না? ব্যাখ্যা কর।
২. এই পাঠে উল্লিখিত প্রোগ্রামিং ভাষাগুলোর মধ্যে বৈশিষ্ট্যগত পার্থক্য নিরূপণ কর।

### ৫.৭ চতুর্থ প্রজন্মের ভাষা (4th Generation Language-4GL)

কম্পিউটারে বা ডিজিটাল ডিভাইসে অতি সহজে ব্যবহারের জন্য বিশেষ ভাষাকে চতুর্থ প্রজন্মের ভাষা বা 4GL বলা হয়। এটি Non-Procedural ভাষা অর্থাৎ এই ভাষায় শুধু বলে দিতে হয় যে কী ফলাফল প্রয়োজন। কীভাবে কার্য সমাধান করতে হবে তা ব্যাখ্যা করার প্রয়োজন নেই। এ প্রজন্মের ভাষার জন্য প্রসেসিং ক্ষমতা বেশি দরকার হয়। 4GL এর সাহায্যে সহজেই অ্যাপ্লিকেশন তৈরি করা যায় বলে একে Rapid Application Development (RAD) টুলও বলা হয়। ডেটাবেজকে নিয়ন্ত্রণ, পরিচালনা, কুয়েরি ও রিপোর্ট তৈরির জন্য এই ভাষা ব্যবহার করা হয়।

উদাহরণ: SQL (Structured Query language), Visual Basic, Oracle, NOMAD, RPG III, Focus, Intellect BPM ইত্যাদি।

#### 4GL এর বৈশিষ্ট্য

- এটি মুক্ত প্রকৃতির ভাষা।
- ডেটাবেজ ব্যবহার করে ডেটা সংরক্ষণ, অনুসন্ধান, উত্তোলন, ডেটা প্রবেশ, মডিফাই, ডিলিট, আপডেট ইত্যাদি কাজ করার জন্য ব্যবহার করা হয়।
- এ ধরনের ভাষাকে মেশিন ভাষায় রূপান্তরের জন্য ইন্টেলিজেন্ট কম্পাইলারের প্রয়োজন হয়।
- এই ভাষায় কাজ করা অত্যন্ত সহজ, কেননা এই ভাষা প্রায় মানুষের ভাষার ন্যায় বা ইংরেজি ভাষার ন্যায়।
- এই ভাষায় ব্যবহারকারীর নিজস্ব চিন্তাভাবনা প্রয়োগের সুযোগ নেই।

#### চতুর্থ প্রজন্মের ভাষার অসুবিধা

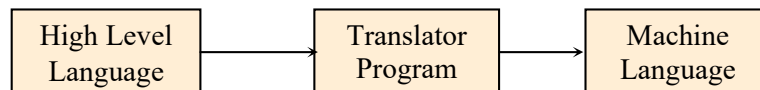
১. প্রোগ্রামিং করার সময় বেশি মেমোরির প্রয়োজন হয়।
২. কম্পিউটারের হার্ডওয়্যারের ধারণক্ষমতার উপর নির্ভরশীল।
৩. বেশি ডিস্ক স্পেস এর প্রয়োজন হয়।
৪. প্রোগ্রামের আকার অধিকাংশ সময়ে বড় হয়।

### পঞ্চম প্রজন্মের ভাষা (5GL)

স্বাভাবিক ভাষা (Natural Language): পঞ্চম প্রজন্মের প্রোগ্রামের ভাষা হিসেবে মানুষের স্বাভাবিক ভাষা বা ন্যাচারাল ল্যাঙ্গুয়েজকে ব্যবহারের চেষ্টা চলছে। মানুষের ভাষার মতো স্বাভাবিক ভাষা কম্পিউটারে ব্যবহারের জন্য এখনো অনেক পরীক্ষা-নিরীক্ষা চলছে। এ ধরনের ভাষাকে মেশিনের ভাষায় রূপান্তরের জন্য ব্যবহৃত অনুবাদককে বুদ্ধিমান বা ইন্টেলিজেন্ট কম্পাইলার বলা হয়। কৃত্রিম বুদ্ধিমত্তায় সাধারণত পঞ্চম প্রজন্মের ভাষা ব্যবহৃত হয়। যেমন— Prolog, OPS5, Mercury হলো 5GL-এর উদাহরণ।

### ৫.৮ অনুবাদক প্রোগ্রাম (Translator Program)

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



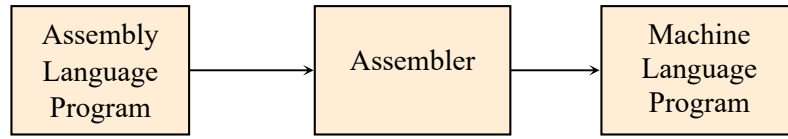
চিত্র: অনুবাদক প্রোগ্রামের কাজের ধারা

কম্পিউটার মেশিন ভাষা ছাড়া অন্য কোনো ভাষা বুঝতে পারে না। আবার মানুষের পক্ষে মেশিন ভাষায় প্রোগ্রাম করা অত্যন্ত কষ্টসাধ্য। তাই মানুষ হাই লেভেল ভাষায় অথবা অ্যাসেম্বলি ভাষায় প্রোগ্রাম লিখে পরে অনুবাদক প্রোগ্রাম দ্বারা একে মেশিন ভাষায় রূপান্তর করে। অনুবাদক প্রোগ্রাম তিন প্রকার:

১. অ্যাসেম্বলার (Assembler), ২. কম্পাইলার (Compiler), ৩. ইন্টারপ্রেটার (Interpreter)

#### ৫.৮.১ অ্যাসেম্বলার (Assembler)

অ্যাসেম্বলি ভাষায় লিখিত প্রোগ্রামকে মেশিন ভাষায় অনুবাদ করার জন্য অ্যাসেম্বলার ব্যবহার করা হয়। এটি অ্যাসেম্বলি ভাষায় লিখিত প্রোগ্রামকে যান্ত্রিক ভাষায় রূপান্তর করে। অর্থাৎ নেমোনিক কোডকে মেশিন ভাষায় অনুবাদ করে। প্রোগ্রামে কোনো ভুল থাকলে Error Message দেয়।



চিত্র: অ্যাসেম্বলারের কাজের ধারা

অ্যাসেম্বলারের প্রধান কাজসমূহ:

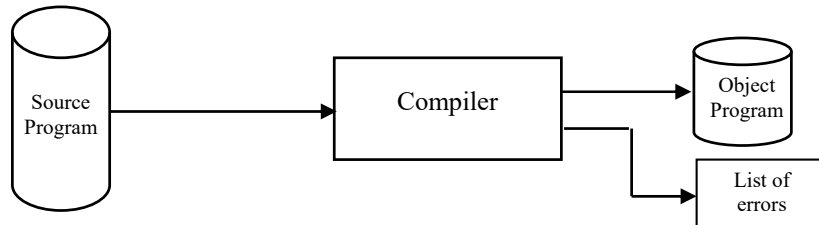
১. নেমোনিক কোডকে মেশিন ভাষায় অনুবাদ করা।
২. অ্যাসেম্বলি অ্যাড্রেসকে মেশিন ভাষায় লেখা অ্যাড্রেসে পরিণত করা।
৩. প্রোগ্রামে কোনো ভুল থাকলে Error Message দেওয়া।
৪. সব নির্দেশ ও ডেটা প্রধান মেমোরিতে রাখা।
৫. প্রত্যেকটি নির্দেশনা পরীক্ষা করে দেখা, ঠিক না থাকলে তা জানানো।

#### ৫.৮.২ কম্পাইলার (Compiler)

কম্পাইলার হলো এক ধরনের অনুবাদক যা উচ্চতর ভাষায় লিখিত প্রোগ্রামকে মেশিন ভাষায় রূপান্তর করে। অর্থাৎ সোর্স প্রোগ্রামকে অবজেক্ট প্রোগ্রামে রূপান্তর করার প্রোগ্রামকে কম্পাইলার বলে। কম্পাইলার দুই ধাপে অনুবাদকের কাজ সম্পন্ন করে।

**প্রথম ধাপে:** কম্পাইলার উৎস প্রোগ্রামের প্রত্যেকটি লাইন পড়ে এবং অবজেক্ট প্রোগ্রামে রূপান্তর করে। এই ধাপে কম্পাইলার সোর্স প্রোগ্রামে যদি ভুল থাকে, তবে তা সংশোধন করার জন্য 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:** ফলাফল প্রদর্শনের জন্য উপাত্ত বা ডেটার ভিত্তিতে অবজেক্ট প্রোগ্রামকে নির্বাহ করানো হয়। কম্পিউটার দিয়ে অনুবাদক প্রক্রিয়া চিত্রের মাধ্যমে দেখানো হলো।



চিত্র: কম্পাইলারের কাজ

#### কম্পাইলারের কাজ:

- উৎস প্রোগ্রামের স্টেটমেন্টসমূহকে মেশিনের ভাষায় রূপান্তর করা।
- সংশ্লিষ্ট সাব-রুটিন এর সাথে সংযোগের ব্যবস্থা প্রদান করা।
- প্রধান স্মৃতির পরিসর চিহ্নিতকরণ করা।
- প্রয়োজন হলে কাগজে সোর্স প্রোগ্রাম ও অবজেক্ট প্রোগ্রামের লিখিতরূপ প্রস্তুতকরণ করা।
- প্রোগ্রাম ভুল থাকলে অনুবাদের সময় ভুলের তালিকা প্রণয়ন করা।

#### কম্পাইলারের সুবিধা:

- 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 কোনো ভুল থাকলে তা মনিটরে একসাথে প্রদর্শন করে।

#### কম্পাইলারের অসুবিধা:

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

#### ৫.৮.৩ ইন্টারপ্রেটার (Interpreter)

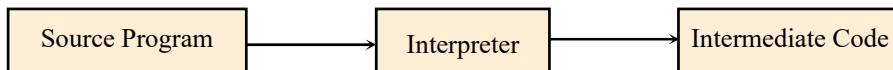
ইন্টারপ্রেটারও কম্পাইলারের মতো উচ্চতর ভাষাকে মেশিন ভাষায় রূপান্তর করে। কম্পাইলার যেমন প্রথমে সোর্স প্রোগ্রামকে অবজেক্ট প্রোগ্রামে রূপান্তর করে এবং সর্বশেষ ফলাফল প্রদান করে। কিন্তু ইন্টারপ্রেটার সোর্স প্রোগ্রামকে অবজেক্ট প্রোগ্রামে রূপান্তর করে না, ইন্টারপ্রেটার লাইন নির্বাহ করে এবং তাৎক্ষণিক ফলাফল প্রদর্শন করে। যেমন-BASIC ভাষায় লিখিত প্রোগ্রামের তিনটি লাইন

LET A = 6

LET B = 2+A

PRINT B

ইন্টারপ্রেটার যখন প্রথম লাইনে প্রবেশ করবে, তখন কম্পিউটার A চলকের জন্য মেমোরি এলাকা তৈরি করে তার মধ্যে 6 জমা রাখে। দ্বিতীয় লাইনে প্রবেশ করলে B চলকের জন্য মেমোরি এলাকা তৈরি করে তার মধ্যে 8 জমা রাখবে। ৩য় লাইনে প্রবেশ করার পর কম্পিউটার স্ক্রিনে B এর মধ্যে 8 প্রদর্শন করবে। এভাবে ইন্টারপ্রেটার প্রোগ্রামের লাইন অনুসারে প্রোগ্রামকে নির্বাহ করে।



চিত্র: ইন্টারপ্রেটারের কর্মপ্রক্রিয়া

**ইন্টারপ্রেটারের সুবিধা:**

- এটি ব্যবহারে প্রোগ্রামের ভুল সংশোধন করা এবং পরিবর্তন করা সহজ হয়।
- Interpreter Program আকারে ছোট হয় এবং মেমোরি স্থানে কম জায়গা দখল করে।
- এটি সাধারণত ছোট কম্পিউটারে ব্যবহার করা হয়।

**ইন্টারপ্রেটারের অসুবিধা:**

- ইন্টারপ্রেটার ব্যবহারে প্রোগ্রাম কার্যকরী করতে কম্পাইলারের তুলনায় বেশি সময় লাগে।
- এটির মাধ্যমে রূপান্তরিত প্রোগ্রাম সম্পূর্ণরূপে মেশিন প্রোগ্রামে রূপান্তরিত হয় না।
- প্রতিটি কাজের পূর্বে অনুবাদ করার প্রয়োজন হয়।

**কম্পাইলার ও ইন্টারপ্রেটারের পার্থক্য**

কম্পাইলার	ইন্টারপ্রেটার
১. সম্পূর্ণ প্রোগ্রামটিকে একসাথে অনুবাদ করে।	১. এক লাইন এক লাইন করে অনুবাদ করে।
২. কম্পাইলার দ্রুত কাজ করে।	২. ইন্টারপ্রেটার ধীরে কাজ করে।
৩. সবগুলো ভুল একসাথে প্রদর্শন করে।	৩. প্রতিটি লাইনের ভুল প্রদর্শন করে এবং ভুল পাওয়া মাত্রই কাজ বন্ধ করে দেয়।
৪. ভুল-ত্রুটি দূর করার ক্ষেত্রে সময় বেশি লাগে।	৪. ভুল-ত্রুটি দূর করার ক্ষেত্রে দ্রুত কাজ করে।
৫. কাজ করতে প্রধান মেমোরিতে বেশি জায়গা প্রয়োজন হয়।	৫. প্রধান মেমোরিতে কম জায়গা প্রয়োজন হয়।
৬. বড় ধরনের কম্পিউটারে বেশি ব্যবহৃত হয়।	৬. অপেক্ষাকৃত ছোটো কম্পিউটারে বেশি ব্যবহৃত হয়।
৭. কম্পাইলারের মাধ্যমে প্রোগ্রামকে রূপান্তরের পর তা পূর্ণাঙ্গ মেশিন ভাষায় রূপান্তরিত হয়। একে বলা হয় অবজেক্ট প্রোগ্রাম।	৭. ইন্টারপ্রেটারের মাধ্যমে প্রোগ্রামকে রূপান্তরের পর তা একটি মধ্যবর্তী অবস্থানে পৌঁছে। একে বলা হয় ইন্টারমিডিয়েট কোড।
৮. একবার কম্পাইল করার পর দ্বিতীয়বার কম্পাইল করার প্রয়োজন হয় না।	৮. ইন্টারপ্রেটারের ক্ষেত্রে পুনরায় রূপান্তরের প্রয়োজন হয়।

**কাজ:**

মাহী ও রাহি প্রোগ্রামার। দু'জনের প্রোগ্রাম তৈরির পদ্ধতি দু'ধরনের। রাহির প্রোগ্রাম ভুল সংশোধন করে সম্পূর্ণ প্রোগ্রাম পড়ার পর আর মাহীর প্রোগ্রাম ভুল সংশোধন করে প্রতিটি লাইন পৃথক পৃথক ভাবে।

অপরদিকে জামীর প্রোগ্রাম লেখার জন্য ইংরেজি শব্দ ব্যবহার করে।

প্রশ্ন ১: জামীর ব্যবহৃত প্রোগ্রামিং ভাষার সুবিধা ও অসুবিধা লেখ।

প্রশ্ন ২: প্রোগ্রাম নির্বাহের ক্ষেত্রে মাহী ও রাহির ব্যবহৃত প্রোগ্রামের মধ্যে কোনটি দ্রুত গতিসম্পন্ন?

বিশ্লেষণপূর্বক মতামত দাও।

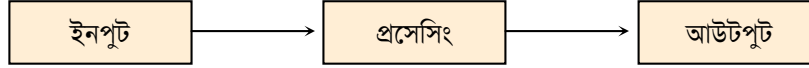


## পাঠ ৬

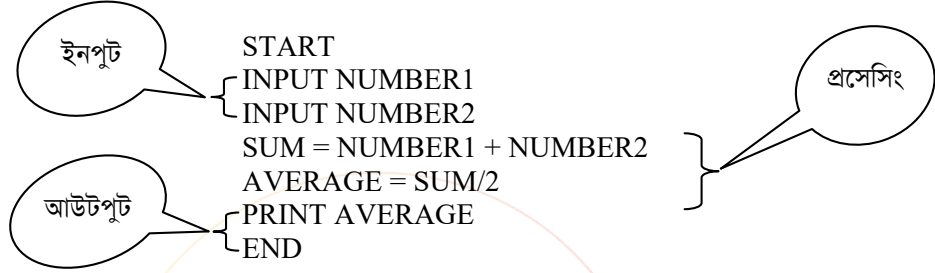
## প্রোগ্রামের সংগঠন ও প্রোগ্রাম তৈরির ধাপসমূহ

### ৫.৯ প্রোগ্রামের সংগঠন (Organization of a Program)

প্রোগ্রামের গঠনরীতিকে প্রোগ্রামের সংগঠন বলা হয়। অর্থাৎ সমস্যা সমাধানের উদ্দেশ্যে প্রোগ্রামকে কিভাবে গঠন করা যায় তা বুঝায়। প্রত্যেক প্রোগ্রামের তিনটি অংশ থাকে। প্রত্যেকটি অংশের পারস্পরিক সমন্বয়ের মাধ্যমে তৈরি হয় পূর্ণাঙ্গ প্রোগ্রাম। প্রোগ্রামের তিনটি অংশ হচ্ছে: ইনপুট, প্রসেসিং ও আউটপুট।



ফলাফল লাভের উদ্দেশ্যে কম্পিউটারে যেসব ডেটা নির্দেশ দেওয়া হয় তা হলো ইনপুট। প্রদত্ত ডেটাকে নির্দেশ অনুযায়ী প্রক্রিয়াকরণ করা হয় যা হলো প্রসেসিং। প্রসেসিং থেকে প্রাপ্ত ফলাফল প্রদর্শন করা হলো আউটপুট। একটি উদাহরণের মাধ্যমে নিচে প্রোগ্রামের সংগঠন দেখানো হলো:



#### ৫.৯.১ একটি আদর্শ প্রোগ্রামের বৈশিষ্ট্য (Features of an Ideal Program)

আদর্শ প্রোগ্রাম বলতে প্রোগ্রামের যাবতীয় বৈশিষ্ট্য বা গুণাবলীকে বুঝায়। একটি আদর্শ প্রোগ্রামে সাধারণত পাঁচটি পর্ব থাকে। যথা-১. পরিচয় পর্ব; ২. বর্ণনা; ৩. ইনপুট; ৪. প্রসেসিং; ৫. আউটপুট।

একটি আদর্শ প্রোগ্রামের যেসকল গুণাবলি থাকা দরকার নিম্নে তা উল্লেখ করা হলো:

- প্রোগ্রামের অ্যালগরিদম সরলভাবে প্রণয়ন করা যাতে প্রোগ্রামের ধাপগুলো সহজেই বুঝা যায়।
- প্রোগ্রামের প্রবাহচিত্র স্পষ্টভাবে উপস্থাপন করা যাতে প্রোগ্রাম নির্বাহের ধারা বুঝা যায়।
- নির্দিষ্ট সমস্যার জন্য প্রয়োজনে উপযুক্ত প্রোগ্রামিং ভাষা নির্বাচন করা।
- প্রোগ্রামের শুরুতেই প্রোগ্রামের উদ্দেশ্য, প্রোগ্রামারের নাম, ধ্রুবক, চলক ইত্যাদির পরিচয় রাখা। এতে পরবর্তী প্রোগ্রামার সহজেই প্রোগ্রামের উদ্দেশ্য ও কাজ সম্পর্কে ধারণা পেতে পারে।
- চলক হিসাবে অর্থপূর্ণ শব্দ ব্যবহার করা যাতে চলকের উদ্দেশ্য বুঝতে অসুবিধা না হয়।
- প্রোগ্রামকে অকারণে দীর্ঘ না করা।
- গুরুত্বপূর্ণ অংশ বা যেকোনো ফাংশন লিখলে তার সাথে মন্তব্য দিতে হয়।
- তথ্য প্রদানের ব্যবস্থা করতে হয়।
- তথ্য প্রক্রিয়াকরণের ব্যবস্থা করতে হয়।
- অবশ্যই ফলাফল প্রাপ্তির সুবিধা করতে হয়।

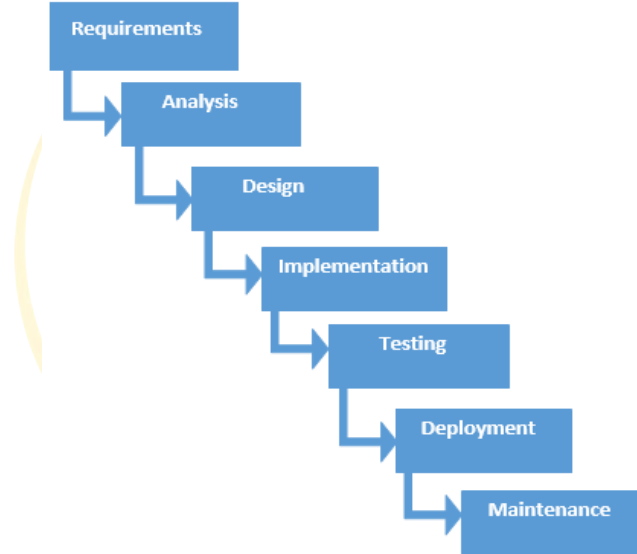
### ৫.১০ প্রোগ্রাম তৈরির ধাপ (Steps of Developing a Program)

কম্পিউটারের সাহায্যে কোন বিভিন্ন সমস্যা সমাধানের জন্যে কতকগুলো পদক্ষেপ নিতে হয়। এ পদক্ষেপ সমূহকে প্রোগ্রাম তৈরির ধাপ বলে। প্রোগ্রাম তৈরির ধাপগুলো নিম্নরূপ-

১. সমস্যা নির্দিষ্টকরণ (Problem Specification)
২. সমস্যা বিশ্লেষণ (Problem Analysis)
৩. প্রোগ্রাম ডিজাইন (Program Design)

তথ্য ও যোগাযোগ প্রযুক্তি (বোর্ড)-২৬ক

৪. প্রোগ্রাম উন্নয়ন (Program Development)
  ৫. প্রোগ্রাম বাস্তবায়ন (Program Implementation)
  ৬. ডকুমেন্টেশন (Documentation)
  ৭. প্রোগ্রাম রক্ষণাবেক্ষণ (Program Maintenance)
১. **সমস্যা নির্দিষ্টকরণ (Problem Specification):** প্রোগ্রামিং সমস্যা সমাধানের পূর্বে তা অবশ্যই ভালোভাবে চিহ্নিত করতে হয়। কোন ধরনের ইনপুট দিতে হবে এবং কী ধরনের আউটপুট প্রয়োজন সে বিষয়ে সিদ্ধান্ত নেওয়া হয়। সমস্যা সমাধানের জন্য যে সকল তথ্য প্রয়োজন তা বিভিন্ন পদ্ধতি অবলম্বন করে সংগ্রহ করতে হয়। অর্থাৎ সঠিকভাবে সমস্যা নির্দিষ্ট করতে পারলে কাজিত প্রোগ্রাম তৈরি করা যায়।
  ২. **সমস্যা বিশ্লেষণ (Problem Analysis):** সমস্যা নির্দিষ্টকরণের পর সমস্যাটিকে ক্ষুদ্র ক্ষুদ্র অংশে ভাগ করতে হয়। এই ধাপে নিম্নলিখিত বিষয়গুলোর ওপর গুরুত্ব দিতে হয়-
    - ক. সমস্যার গাণিতিক মডেল তৈরি করা।
    - খ. সমস্যার সমাধানে কত সময় লাগবে তা নিরূপণ করা।
 সমস্যা বিশ্লেষণের ক্ষেত্রে বিভিন্ন টুলস বা পদ্ধতি ব্যবহার করা হয়। যেমন— ডেটা ফ্লো ডায়াগ্রাম (DFD), ডিসিশন ট্রি, ডিসিশন টেবিল, স্ট্রাকচার্ড ইংলিশ ইত্যাদি।
  ৩. **প্রোগ্রাম ডিজাইন (Program Design):** প্রোগ্রাম বিশ্লেষণ ধাপে যে ছোটো ছোটো ভাগগুলো করা হয়েছে তাদের পারস্পরিক সম্পর্ক ও সামগ্রিক সমাধান বের করতে হয়। প্রোগ্রাম ডিজাইনে নিম্নলিখিত বিষয়গুলো অন্তর্ভুক্ত:
    - ক. ইনপুট ডিজাইন, খ. আউটপুট ডিজাইন ও গ. ইনপুট ও আউটপুটের মধ্যে সম্পর্ক ডিজাইন।
 ডিজাইনের ক্ষেত্রে অ্যালগরিদম, ফ্লোচার্ট ও সুডোকোডের সাহায্যে সমস্যার সমাধান দিতে হয়।



চিত্র: প্রোগ্রাম তৈরির ধাপসমূহ

৪. **প্রোগ্রাম উন্নয়ন (Program Development):** কম্পিউটারের বোধগম্য ভাষায় প্রোগ্রাম রচনাকে কোডিং বলা হয়। নির্দিষ্ট কোনো সমস্যা সমাধানের জন্য তৈরিকৃত অ্যালগরিদম ও ফ্লোচার্টের আলোকে প্রোগ্রামিং ভাষার মাধ্যমে প্রোগ্রাম রচনা করতে হয়। যেমন— C, Pascal, Q Basic ইত্যাদি বিভিন্ন ধরনের প্রোগ্রামিং ভাষার মাধ্যমে কোডিং করা যায়।
৫. **প্রোগ্রাম বাস্তবায়ন (Program Implementation):** কোডিং-এর পর প্রতিটি অংশকে পরীক্ষা নিরীক্ষা করে দেখতে হয়। প্রয়োজনীয় সংশোধনের মাধ্যমে প্রোগ্রামকে বাস্তবায়ন করতে সম্পূর্ণভাবে প্রস্তুত করতে হয়। বাস্তবায়ন অংশের দুটি গুরুত্বপূর্ণ কাজ হচ্ছে:

ক. **টেস্টিং:** এ ধাপে ভুল-ত্রুটি পরীক্ষা করা হয়। প্রোগ্রাম টেস্টিং হচ্ছে কোডিং সম্পন্ন করার পর প্রোগ্রামটির যে ধরনের আউটপুট বা ফলাফল হওয়া উচিত তা ঠিকমতো আসছে কিনা বা রান করছে কিনা তা যাচাই করা। ভিন্ন ভিন্ন ইনপুট দিয়ে আউটপুটের অবস্থা পর্যবেক্ষণ করা হয়। এক্ষেত্রে যদি কোনো অসঙ্গতি পাওয়া যায় তবে বুঝতে হয় প্রোগ্রাম কোডিংয়ের কোথাও ভুল হয়েছে। প্রোগ্রামে সাধারণত তিন ধরনের ভুল পরিলক্ষিত হয়। যথা:

১. ব্যাকরণগত ভুল (Syntax Error)

২. যৌক্তিক ভুল (Logical Error)

৩. নির্বাহজনিত ভুল (Execution Error or Runtime Error)

১. **ব্যাকরণগত ভুল (Syntax Error):** যে ভাষায় প্রোগ্রাম লেখা হয় তার নিজস্ব কতগুলো নিয়ম থাকে। নিয়মবহির্ভূত কোনো কোডিং হয়ে থাকলে তাকে ব্যাকরণগত ভুল বলা হয়। যেমন: C প্রোগ্রামিং ভাষায় কোনো স্টেটমেন্টের পর সেমিকোলন (;) দিতে হয়। এক্ষেত্রে সেমিকোলন (;) না দিলে তা হবে ব্যাকরণগত ভুল। ব্যাকরণগত ভুল থাকলে কম্পিউটার Error Message দিবে এবং প্রোগ্রামের কোথায় কী ভুল হয়েছে তা জানিয়ে দিবে। ভুল সংশোধন না করা হলে কম্পিউটার প্রোগ্রাম নির্বাহ করবে না।
২. **যৌক্তিক ভুল (Logical Error):** প্রোগ্রামে কোনো লজিক ভুল হলে ফলাফল ঠিকই আসবে কিন্তু তা সঠিক হবে না। এধরনের ভুলকে যৌক্তিক ভুল বলা হয়। ধরা যাক,  $X > Y$  এর স্থলে  $X < Y$  লিখলে কম্পিউটার কোনো Error Message দিবে না কিন্তু ফলাফল ভুল প্রদর্শিত হবে।
৩. **নির্বাহজনিত ভুল (Execution Error or Runtime Error):** প্রোগ্রাম নির্বাহের সময় ভুল ডেটা ইনপুট দিলে অথবা ডেটার ফরম্যাট ঠিক না থাকলে আউটপুট বা ফলাফল ভুল আসবে অথবা প্রোগ্রাম নির্বাহ হবে না। এধরনের ভুলকে নির্বাহজনিত ভুল বলা হয়।

খ. **ডিবাগিং:** এ ধাপে ভুল সংশোধন করা হয়। এ ধাপে ভুল সংশোধন করা হয়। প্রোগ্রামে যে কোনো ভুল চিহ্নিত করতে পারলে তাকে বলা হয় বাগ (Bug)। উক্ত বাগ সমাধান করাকে বলা হয় ডিবাগ (Debug)। এক্ষেত্রে Syntax Error সমাধান করা সহজ। কিন্তু Logical Error সমাধান করা তুলনামূলক জটিল।



জেনে রাখো

প্রোগ্রাম থেকে ভুল-ত্রুটি খুঁজে বের করে তা সমাধান করাকে ডিবাগিং (debugging) বলা হয়। 'Bug' অর্থ পোকা। ডিবাগিং অর্থ পোকা দূর করা। ডিবাগিং করার জন্য প্রোগ্রামটিকে প্রথমে শুরু থেকে শেষ পর্যন্ত ভালোভাবে পরীক্ষা করা হয়।

৬. **ডকুমেন্টেশন (Documentation):** প্রোগ্রাম সংশোধনের পর প্রোগ্রামকে ভবিষ্যতে রক্ষণের জন্য লিপিবদ্ধ করাকে প্রোগ্রাম ডকুমেন্টেশন বলে। প্রোগ্রাম রক্ষণাবেক্ষণে ডকুমেন্টেশনের গুরুত্ব অপরিসীম। ডকুমেন্টেশনে নিম্নলিখিত বিষয়সমূহ অন্তর্ভুক্ত করা হয়।
  - ক. প্রোগ্রামের বর্ণনা।
  - খ. ফ্লোচার্ট
  - গ. লিখিত প্রোগ্রাম
  - ঘ. নির্বাহের জন্য প্রয়োজনীয় কাজের তালিকা
  - ঙ. পরীক্ষণ ও ফলাফল

৭. **প্রোগ্রাম রক্ষণাবেক্ষণ (Program maintenance):** সময়ের সাথে সাথে পরিবেশ-পরিস্থিতি পরিবর্তনের কারণে প্রোগ্রামের পরিবর্তন বা আধুনিকীকরণ করা প্রয়োজন হয়। এ ধরনের কাজ রক্ষণাবেক্ষণ ধাপের অন্তর্ভুক্ত। এছাড়া প্রোগ্রাম সংশ্লিষ্ট বিভিন্ন গুরুত্বপূর্ণ ডকুমেন্টেশনের কাজ এ ধাপে সম্পন্ন করা হয়।



কাজ :

তিনটি সংখ্যার মধ্যে বড় সংখ্যা নির্ণয়ের ক্ষেত্রে প্রোগ্রাম তৈরির ধাপগুলোর প্রয়োগ দেখাও।

## পাঠ ৭-১০

## ব্যবহারিক: অ্যালগরিদম ও ফ্লোচার্ট

## ৫.১০.১ অ্যালগরিদম (Algorithm)

কোনো একটি নির্দিষ্ট সমস্যা সমাধানের জন্য যুক্তিসম্মত ও পর্যায়ক্রমিকভাবে ধাপে ধাপে সমাধান করার যে পদ্ধতি, তাকে অ্যালগরিদম বলা হয়। অ্যালগরিদম শব্দটি এসেছে মুসলিম গণিতবিদ “মুসা আল খারিজমী”-এর নাম থেকে। একজন প্রোগ্রামার প্রোগ্রামিং সমস্যার ধরনের উপর ভিত্তি করে পর্যায়ক্রমিকভাবে ছোট ছোট অংশে ভাগ করে অ্যালগরিদম রচনা করেন। ফলে খুব সহজেই প্রোগ্রাম নির্বাহের ধাপগুলো স্পষ্টভাবে বুঝা যায়। ফলে প্রোগ্রাম কোডিং করতে অনেক সহজ হয়। অর্থাৎ অ্যালগরিদমকে যত সহজভাবে উপস্থাপন করা যায় প্রোগ্রামকে তত সহজভাবে লেখা যায়। এটির এক বা একাধিক ইনপুট থাকবে কিন্তু একটি মাত্র আউটপুট হবে।

অ্যালগরিদম তৈরির শর্ত বা নিয়ম:

১. অ্যালগরিদমটি সহজবোধ্য হতে হবে।
২. প্রত্যেকটি ধাপ স্পষ্ট হতে হবে যাতে সহজে বোঝা যায়।
৩. সসীম সংখ্যক ধাপে সমস্যার সমাধান হতে হবে।
৪. অ্যালগরিদম ব্যাপকভাবে প্রয়োগ উপযোগী হতে হবে।

অ্যালগরিদমের সুবিধা:

১. সহজে প্রোগ্রামের উদ্দেশ্য বোঝা যায়।
২. সহজে প্রোগ্রামের ভুল নির্ণয় করা যায়।
৩. প্রোগ্রামের প্রবাহের দিক বুঝা যায়।
৪. জটিল প্রোগ্রাম সহজে রচনা করা যায়।
৫. প্রোগ্রাম পরিবর্তন ও পরিবর্তনে সহায়তা করে।

## ৫.১০.২ ফ্লোচার্ট বা প্রবাহ চিত্র (Flowchart)

কোনো একটি নির্দিষ্ট সমস্যা সমাধানের পর্যায়ক্রমিক ধাপগুলোকে বিশেষ কতগুলো চিহ্নের সাহায্যে প্রকাশ করাকে বলা হয় ফ্লোচার্ট বা প্রবাহচিত্র। ফ্লোচার্টের প্রোগ্রাম বোঝা সহজ হয় বলে প্রোগ্রামার ও ব্যবহারকারীর মাঝে সংযোগ রক্ষার জন্য এটি ব্যবহৃত হয়। অ্যালগরিদম একটি সমস্যাকে ধাপে ধাপে সমাধান করে এবং ফ্লোচার্ট ধাপগুলোরই চিত্রভিত্তিক রূপান্তর।

ফ্লোচার্টের বৈশিষ্ট্য বা সুবিধা:

একটি উন্নতমানের ফ্লোচার্টে নিম্নলিখিত বৈশিষ্ট্যসমূহ থাকে-

১. সহজে প্রোগ্রামের উদ্দেশ্য বোঝা যায়।
২. প্রোগ্রামের ভুল নির্ণয়ে সহায়তা করে।
৩. প্রোগ্রাম রচনায় সহায়তা করে।
৪. প্রোগ্রাম পরিবর্তন এবং পরিবর্তনে সহায়তা করে।
৫. সহজে ও সংক্ষেপে জটিল প্রোগ্রাম লেখা যায়।

ফ্লোচার্ট তৈরি করার নিয়মাবলী (Rules of drawing Flowchart):


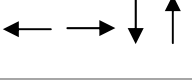

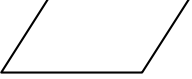



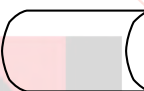
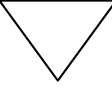

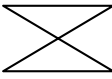

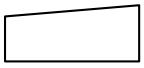

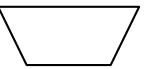


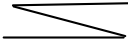
১. ফ্লোচার্ট তৈরি করার জন্য প্রচলিত প্রতীক ব্যবহার করা উচিত।
২. তীর চিহ্ন দিয়ে উপর থেকে নিচে বা বাম থেকে ডান দিকে প্রবাহ দেখানো উচিত।
৩. ফ্লোচার্ট তৈরি করার সময় সংযোগ চিহ্ন যত কম হয় ততই ভালো।
৪. ফ্লোচার্ট সহজে বোধগম্য হওয়া উচিত।

৫. ফ্লোচার্ট নির্দিষ্ট কোনো প্রোগ্রামের ভাষায় লেখা উচিত নয়।
৬. চিহ্নগুলো ছোট বড় হলে ক্ষতি নাই তবে আকৃতি ঠিক থাকতে হবে।
৭. প্রয়োজনে চিহ্নের সাথে মন্তব্য দিতে হবে।

ফ্লোচার্টের প্রকারভেদ: ফ্লোচার্টকে প্রধানত দুভাগে ভাগ করা যায়। যেমন—

১. সিস্টেম ফ্লোচার্ট
২. প্রোগ্রাম ফ্লোচার্ট

১. সিস্টেম ফ্লোচার্ট: সিস্টেম ফ্লোচার্টে উপাত্ত গ্রহণ, প্রক্রিয়াকরণ, স্মৃতিতে সংরক্ষণ ও ফলাফল প্রদর্শনের প্রবাহ দেখানো হয়। অর্থাৎ যে ফ্লোচার্টের মাধ্যমে কোনো ব্যবস্থার সংগঠনকে সহজে তুলে ধরা যায় তাকে সিস্টেম ফ্লোচার্ট বলে। সিস্টেম ফ্লোচার্টে ব্যবহৃত প্রতিকসমূহ নিম্নে ছকের মাধ্যমে দেখানো হলো—

প্রতীক	অর্থ	প্রতীক	অর্থ
	প্রক্রিয়াকরণ		প্রবাহের দিক
	পাঞ্জকার্ড		গ্রহণ/নির্গমন
	ডকুমেন্ট		পাঞ্জ টেপ
	চৌম্বক টেপ		অনলাইন স্মৃতি
	অফ-লাইন স্মৃতি		প্রদর্শন
	কোলেট বা সংযুক্তি		সটিং বা সাজানো
	ম্যানুয়েল ইনপুট		মার্জ বা একত্রিকরণ
	ম্যানুয়েল কাজ		সহায়ক ক্রিয়া
	কী অপারেশন		যোগাযোগ মাধ্যম

চিত্র : সিস্টেম ফ্লোচার্টে ব্যবহৃত প্রতীক

২. **প্রোগ্রাম ফ্লোচার্ট:** প্রোগ্রাম ফ্লোচার্টে প্রোগ্রামের বিভিন্ন ধাপের বিস্তারিত বিবরণ চিত্রের মাধ্যমে প্রদর্শিত করাকেই প্রোগ্রাম ফ্লোচার্ট বলা হয়। প্রোগ্রাম ফ্লোচার্ট ব্যবহার করে প্রোগ্রাম রচনা করা হয়। এছাড়া প্রোগ্রামের তুল নির্ণয় ও সংশোধনের জন্য এই ফ্লোচার্ট ব্যবহার করা হয়।

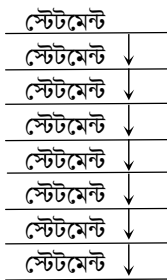
প্রবাহ চিত্রে অনেক রকম প্রতীক চিহ্ন ব্যবহার করা হয়। তার মধ্যে বহুল ব্যবহৃত চিহ্নগুলোর বর্ণনা নিচে দেওয়া হল—

প্রতীক	অর্থ	প্রতীক	অর্থ
	শুরু/শেষ		প্রক্রিয়াকরণ
	সিদ্ধান্ত		পূর্বনির্ধারিত প্রক্রিয়া
	ইনপুট/আউটপুট		সংযোগ
	প্রবাহের দিক		টিকা

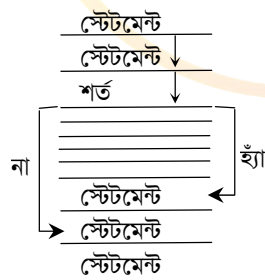
#### ৫.১০.৩ ফ্লোচার্টের মৌলিক গঠন (Basic Structure of Flowchart):

ফ্লোচার্টের ব্যবহৃত মৌলিক প্রতীক চিহ্ন ব্যবহার করে যে কয়েকটি ভাবে ফ্লোচার্ট অংকন করা যায় তাকে ফ্লোচার্টের মৌলিক গঠন বা ছাঁচ বলে। ফ্লোচার্টের মৌলিক গঠন চারটি। যথা:

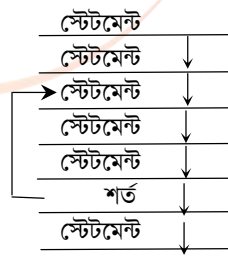
১. **সরল অনুক্রম (Simple sequence):** এটি একটি সরল স্ট্রাকচার। এই স্ট্রাকচারে সকল নির্দেশগুলি ধারাবাহিক অনুক্রমে সাজানো থাকে।
২. **নির্বাচন (Selection):** যে সকল ক্ষেত্রে সিদ্ধান্তের প্রয়োজন বা তুলনা করে কার্য নির্বাহ করতে হয় সেক্ষেত্রে এই স্ট্রাকচার ব্যবহার করা যায়।
৩. **লুপ বা চক্র (Repetition or loop):** প্রোগ্রামে একই ধরনের কাজ বারবার করার প্রয়োজন হলে লুপ বা চক্র ব্যবহার করা হয়। প্রোগ্রামের সুবিধার্থে বিভিন্ন রকম লুপ ব্যবহার করা হয়।
৪. **জাম্প (Jump):** সরল অনুক্রমকে পরিবর্তন করে প্রোগ্রামের মধ্যে এক লাইন থেকে পরবর্তী লাইনে না গিয়ে উপরে বা নিচে অন্য কোন লাইন থেকে কাজ শুরু করলে তাকে জাম্প বলে।



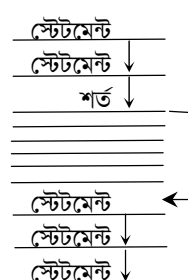
চিত্র: সরলঅনুক্রম



চিত্র: সিদ্ধান্ত বা নির্বাচন



চিত্র: লুপ



চিত্র: জাম্প



### অ্যালগরিদম ও ফ্লোচার্টের পার্থক্য

অ্যালগরিদম	ফ্লোচার্ট
১. যে পদ্ধতিতে পর্যায়ক্রমিক ধাপে অগ্রসর হয়ে কোনো একটি নির্দিষ্ট সমস্যার সমাধান করা হয় তাকে অ্যালগরিদম বলে।	১. যে পদ্ধতিতে চিত্রের সাহায্যে কতকগুলো চিহ্ন ব্যবহার করে সমস্যার ধারাবাহিক সমাধান করা হয় তাকে ফ্লোচার্ট বলে।
২. এটি বর্ণনামূলক।	২. এটি চিত্রভিত্তিক।
৩. এর দ্বারা প্রোগ্রাম বোঝা কঠিন।	৩. এর দ্বারা প্রোগ্রাম বোঝা সহজ।
৪. প্রোগ্রাম প্রবাহের দিক বোঝা যায় না।	৪. প্রোগ্রাম প্রবাহের দিক সহজে বোঝা যায়।
৫. প্রোগ্রামের ভুল-ত্রুটি দূর করা কঠিন।	৫. প্রোগ্রামের ভুল-ত্রুটি দূর করা সহজ।

নিচে কিছু সমস্যার জন্য অ্যালগরিদম ও ফ্লোচার্ট দেখানো হলো:

উদাহরণ-১. তিনটি সংখ্যার গড় নির্ণয়ের জন্য অ্যালগরিদম ও ফ্লোচার্ট অঙ্কন কর।

অ্যালগরিদম:

ধাপ-১: প্রোগ্রাম শুরু করি।

ধাপ-২: ইনপুট হিসেবে a, b এবং c এর মান গ্রহণ করি।

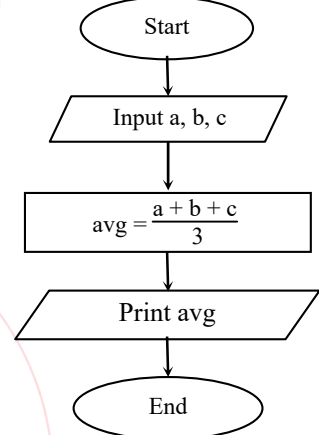
ধাপ-৩:  $avg = (a + b + c)/3$  সূত্র ব্যবহার করে

avg এর মান নির্ণয় করি।

ধাপ-৪: avg এর মান ছাপাই।

ধাপ-৫: প্রোগ্রাম শেষ করি।

ফ্লোচার্ট:



উদাহরণ-২. দুটি সংখ্যার বিয়োগফল নির্ণয়ের জন্য অ্যালগরিদম, ফ্লোচার্ট তৈরি কর।

অ্যালগরিদম:

ধাপ-১: প্রোগ্রাম শুরু

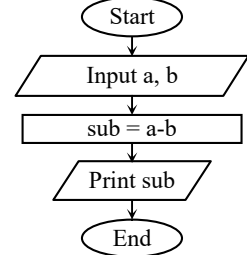
ধাপ-২: a, b এর মান গ্রহণ

ধাপ-৩:  $sub = a - b$  নির্ণয়

ধাপ-৪: sub এর মান ছাপাই

ধাপ-৫: প্রোগ্রাম শেষ

ফ্লোচার্ট:



উদাহরণ-৩. বড় হাতের অক্ষরকে ছোট হাতের অক্ষরের রূপান্তরের জন্য অ্যালগরিদম, ফ্লোচার্ট তৈরি কর।

অ্যালগরিদম:

ধাপ-১: প্রোগ্রাম শুরু

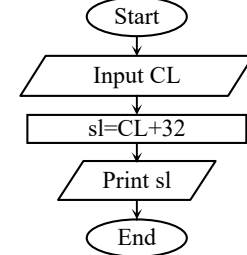
ধাপ-২: ইনপুট হিসাবে বড় হাতের অক্ষর CL এর মান গ্রহণ করি।

ধাপ-৩: ছোট হাতের অক্ষর  $sl = CL + 32$  নির্ণয়

ধাপ-৪: sl এর মান ছাপাই

ধাপ-৫: প্রোগ্রাম শেষ

ফ্লোচার্ট:



অ্যাসকি কোডে A এর দশমিক মান হচ্ছে 65 অন্যদিকে a দশমিক মান হচ্ছে 97। ফলে দেখা যাচ্ছে, উভয়ই দশমিক মানের পার্থক্য হচ্ছে 32। অর্থাৎ বড় হাতের (Capital Letter) প্রতিটি অক্ষরের দশমিক মানের সাথে ছোট হাতের (Small Letter) প্রতিটি অক্ষরের দশমিক মানের পার্থক্য 32। সুতরাং বড়হাতের প্রতিটি অক্ষরের সাথে 32 যোগ করলে অক্ষরটি ছোট হাতের অক্ষরে পরিণত হবে। আর ছোট হাতের অক্ষরের হতে 32 বিয়োগ করলে অক্ষরটি বড়হাতের অক্ষরে পরিণত হবে।

উদাহরণ-৪. ছোট হাতের অক্ষরকে বড়হাতের অক্ষরে রূপান্তরের জন্য অ্যালগরিদম, ফ্লোচার্ট তৈরি কর।

অ্যালগরিদম:

ধাপ-১: প্রোগ্রাম শুরু

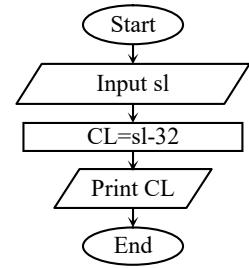
ধাপ-২: ইনপুট হিসাবে slএর মান গ্রহণ কর।

ধাপ-৩:  $CL = sl - 32$  নির্ণয়।

ধাপ-৪: CL এর মান ছাপাই।

ধাপ-৫: প্রোগ্রাম শেষ।

ফ্লোচার্ট:



উদাহরণ-৫. কোনো সংখ্যা জোড় না বিজোড় নির্ণয়ের জন্য অ্যালগরিদম, ফ্লোচার্ট তৈরি কর।

অ্যালগরিদম:

ধাপ-১: প্রোগ্রাম শুরু কর।

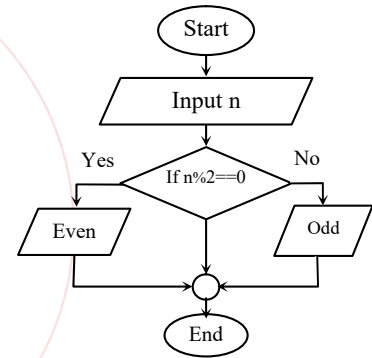
ধাপ-২: ইনপুট হিসেবে n এর মান গ্রহণ কর।

ধাপ-৩: যদি  $(n \% 2 == 0)$  হয় তবে 'Even' ছাপাই,  
নেই ধাপে যাই।

ধাপ-৪: 'Odd' ছাপাই।

ধাপ-৫: প্রোগ্রাম শেষ কর।

ফ্লোচার্ট:



[বি.দ্র: সিভায়ায় % চিহ্নটি ভাগ শেষ হিসাবে ব্যবহৃত হয়]

উদাহরণ-৬. কোনো সংখ্যা ধনাত্মক না ঋণাত্মক তা নির্ণয়ের জন্য অ্যালগরিদম, ফ্লোচার্ট তৈরি কর।

অ্যালগরিদম:

ধাপ-১: প্রোগ্রাম শুরু কর।

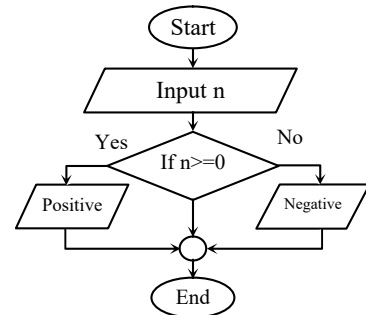
ধাপ-২: ইনপুট হিসেবে n এর মান গ্রহণ কর।

ধাপ-৩: যদি  $(n \geq 0)$  হয় তবে 'Positive' ছাপাই,  
নেই ধাপে যাই।

ধাপ-৪: 'Negative' ছাপাই।

ধাপ-৫: প্রোগ্রাম শেষ কর।

ফ্লোচার্ট:



উদাহরণ-৭: ১, ২, ৩ ..... n ধারাটি নির্ণয়ের জন্য অ্যালগরিদম, ফ্লোচার্ট লিখ।

অ্যালগরিদম:

ধাপ-১: শুরু করি।

ধাপ-২: ইনপুট হিসেবে n এর মান গ্রহণ করি।

ধাপ-৩: a = 1 ধরি।

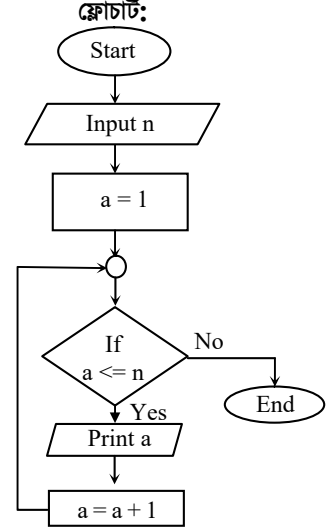
ধাপ-৪: যদি  $a \leq n$  হয় তবে ৫ নং ধাপে যাই।

অন্যথায় ৬ নং ধাপে যাই।

ধাপ-৫: a = a + 1 নির্ণয় করি।

ধাপ-৬: a এর মান ছাপাই।

ধাপ-৭: শেষ করি।



উদাহরণ-৮: ১, ২, ৩ ..... n ধারাটির যোগফল নির্ণয়ের জন্য অ্যালগরিদম, ফ্লোচার্ট লিখ।

অ্যালগরিদম:

ধাপ-১: প্রোগ্রাম শুরু করি।

ধাপ-২: ইনপুট হিসেবে n এর মান গ্রহণ করি।

ধাপ-৩: s = 0, a = 1 ধরি।

ধাপ-৪: যদি  $a \leq n$  হয় তবে ৫ নং ধাপে যাই।

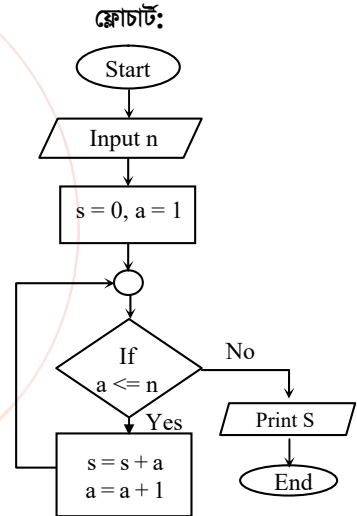
অন্যথায় ৬ নং ধাপে যাই।

ধাপ-৫: s = s + a, a = a + 1 নির্ণয় করি। ৮ নং ধাপে

ফেরত যাই।

ধাপ-৬: s এর মান ছাপাই।

ধাপ-৭: প্রোগ্রাম শেষ করি।



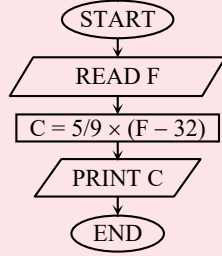
### সূডোকোড (Pseudocode)

সূডো (Pseudo) একটি গ্রীক শব্দ যার অর্থ ছদ্ম বা যা সত্য নয়। প্রোগ্রামের ধরণ ও কার্যাবলি তুলে ধরার জন্য প্রোগ্রামিং এর মতো কিন্তু প্রোগ্রামিং নয় এমন কিছুসংখ্যক নির্দেশ/কোড বা স্টেটমেন্টের সমাহারকেই সূডোকোড বলে।

এটি প্রোগ্রাম ডিজাইনের ক্ষেত্রে পূর্ববর্তী ধাপ হিসেবে ব্যবহৃত হয়। সূডোকোডের মাধ্যমে সহজ ইংরেজি ভাষায় প্রোগ্রামের বিভিন্ন ধাপ বর্ণনা করা হয় যা দেখতে কোনো প্রোগ্রামিং ভাষার কোডিংয়ের মতো মনে হয়। সূডোকোড

নির্দিষ্ট কোনো প্রোগ্রামিং ভাষার উপর নির্ভরশীল নয়। এ পদ্ধতিতে একটি প্রোগ্রামকে এমনভাবে উপস্থাপন করা হয় যেন সকলে তা সহজে বুঝতে পারে। সূডোকোডকে অনেক সময় অ্যালগরিদমের বিকল্প হিসেবে বিবেচনা করা হয়। তিনটি সংখ্যার গড় নির্ণয়ের জন্য সূডোকোড হচ্ছে—

```
START
INPUT X,Y,Z
Total = X+Y+Z
Average = Total/3
OUTPUT Average
END
```



প্রশ্ন-১: চিত্রটির ধারণা প্রোগ্রাম তৈরি ধাপের সাথে কিভাবে সম্পর্কিত? বিশ্লেষণ কর।

নিচের অনুচ্ছেদটি লক্ষ্য কর।

ধাপ-১ : প্রোগ্রাম শুরু

ধাপ-২: দুইটি সংখ্যা পড়।

ধাপ-৩: দুইটি সংখ্যা যোগ করে প্রথম সংখ্যার সাথে গুণ কর।

ধাপ-৪: ফলাফল ছাপাও।

ধাপ-৫: প্রোগ্রাম শেষ।

প্রশ্ন-২: উদ্দীপকে উল্লিখিত প্রক্রিয়াটি প্রোগ্রাম তৈরির ধাপের সাথে কীভাবে সম্পর্কিত বিশ্লেষণ কর।

## পাঠ-১১ ও ১২ প্রোগ্রাম ডিজাইন মডেল

### ৫.১১ প্রোগ্রাম ডিজাইন মডেল (Program Design Model)

প্রোগ্রামের গঠন রীতিনীতিকে প্রোগ্রামের মডেল বলা হয়। সাজিয়ে-গুছিয়ে প্রোগ্রাম লেখা এবং সহজে বোঝার জন্য প্রোগ্রাম রচনার ক্ষেত্রে কয়েকটি মডেল ব্যবহার করা হয়। এই মডেলগুলো প্রোগ্রামের অনুধাবনযোগ্যতা বৃদ্ধি করে। উল্লেখযোগ্য কয়েকটি মডেল সম্পর্কে আলোচনা করা হলো:

#### ৫.১১.১ স্ট্রাকচার্ড প্রোগ্রামিং (Structured Programming)

ডাচ কম্পিউটার বিজ্ঞানী এডগার ওয়েইবে ডেইক্স্ট্রা (Edsger Wybe Dijkstra) [১১ মে, ১৯৩০-৬ আগস্ট, ২০০২] প্রথম বড় আকারের প্রোগ্রাম উন্নয়নের উদ্দেশ্যে স্ট্রাকচার্ড প্রোগ্রামিং-এর ধারণা দেন। স্ট্রাকচার্ড মডেলে পুরো সমস্যাকে বিভিন্ন অংশ বা মডিউলে ভাগ করা হয়। প্রতিটি মডিউলকে ছোট আকারের সমস্যা ভাষা যেতে পারে। স্ট্রাকচার্ড প্রোগ্রামিং এর সুবিধা হলো- এতে বড় আকারের সমস্যা সহজে সমাধান করা যায়। একবার কোনো কোড লিখে তা একাধিকবার ব্যবহার করা যায়। এতে সময় অপচয় রোধ করা যায়। প্রোগ্রামের নির্দিষ্ট কাঠামো থাকায় ডিবাগিং বা প্রোগ্রামের ভুল সংশোধন করা সহজ হয়। স্ট্রাকচার্ড প্রোগ্রামিং-এ একটি মূল প্রোগ্রাম থাকে যা বিভিন্ন মডিউলকে কল করে। এক মডিউল আবার অন্য মডিউলকে কল করতে পারে। BASIC, Fortran, COBOL, PASCAL, C ইত্যাদি স্ট্রাকচার্ড প্রোগ্রামিং ভাষা। স্ট্রাকচার্ড প্রোগ্রামে তিন ধরনের কাঠামো ব্যবহৃত হয়ে থাকে—

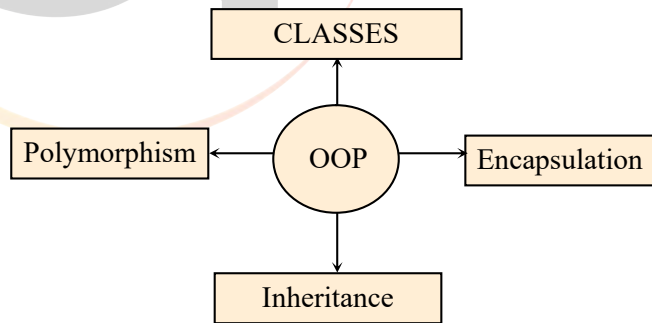
- **পর্যায়ক্রমিক কাঠামো:** এ কাঠামোতে প্রোগ্রামের বা মডিউলের একটির পর একটি নির্দেশ ধারাবাহিকভাবে নির্বাহ হয়। নির্দেশের ধারাবাহিকতা বা পর্যায় কখনো বিঘ্নিত হয় না।
- **সিদ্ধান্তমূলক কাঠামো:** এ কাঠামো একটি নির্দিষ্ট শর্তের ওপর নির্ভর করে। শর্তটি সত্য হলে, একটি স্টেটমেন্ট বা নির্দেশ নির্বাহ হয়; আর মিথ্যা হলে অন্য আরেকটি স্টেটমেন্ট নির্বাহ হয়। সিদ্ধান্তমূলক কাজের প্রয়োজনে এ কাঠামো ব্যবহার করা হয়ে থাকে। একাধিক সিদ্ধান্ত নেওয়ার বেলাতেও এ কাঠামোটি ব্যবহার করা যায়।
- **চক্রাবর্ত কাঠামো:** এ কাঠামোকে লুপ বলা হয়। এক বা একাধিক নির্দেশ বারবার লিখতে হয় না। এক বা একাধিক নির্দেশকে শর্তহীনভাবে নির্দিষ্ট সংখ্যক বার বা শর্তের অধীন অনির্দিষ্ট সংখ্যক বার নির্বাহ করা যায়।

#### ৫.১১.২ অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং (Object Oriented Programming-OOP)

প্রোগ্রামিং এ কোনো সমস্যা সমাধানের জন্য যখন object তৈরি করা হয় এবং এই অবজেক্টের মধ্যে যাবতীয় সমস্যার উপকরণ ও সমাধান পাওয়া যায় তখন তাকে অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং বলা হয়। অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং মডেলে ডেটা ও সংশ্লিষ্ট কোডকে একক হিসেবে বিবেচনা করা হয়। এ ধরনের একককে ক্লাস (Class) বলে। ক্লাসে কোনো ডেটা রেখে নির্বাহ করতে হলে নির্দিষ্ট ক্লাসের অবজেক্ট তৈরি করতে হয়। কোনো প্রোগ্রাম উন্নয়নের সময় ক্লাসগুলো এমনভাবে নির্মাণ করা হয়, যাতে তা বাস্তব সমস্যাকে ভালোভাবে উপস্থাপন করতে পারে।

ছোট আকারের প্রোগ্রাম রচনার জন্য OOP

মডেল কোনো বিশেষ সুবিধা দেয় না। কিন্তু বড় ধরনের প্রোগ্রাম (কয়েক হাজার লাইনের অধিক) উন্নয়নের জন্য OOP অপরিহার্য মডেল। OOP-এর বিশেষ সুবিধা হলো- ইনহেরিটেন্সের মাধ্যমে প্রচলিত ক্লাসকে বর্ধিত করে নতুন ও উন্নত ক্লাস তৈরি করা যায়। ডেটা লুকানো থাকে বলে অপ্রত্যাশিত পরিবর্তন সম্ভব নয়। সহজেই ছোট থেকে বড় প্রোগ্রাম উন্নয়ন করা যায়। সকল প্রোগ্রামিং ভাষাই অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং সমর্থন করে না। OOP প্রোগ্রামিং ভাষার



উদাহরণ হলো C++, Java, C# ইত্যাদি। কোনো প্রোগ্রামিং ভাষাকে পরিপূর্ণ অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং ভাষা হতে হলে কমপক্ষে তিনটি বৈশিষ্ট্য থাকতে হয়। এগুলো হলো—

১. **ইনহেরিটেন্স (Inheritance):** ইনহেরিটেন্স এমন একটি ক্ষমতা যার মাধ্যমে কোনো প্রচলিত ক্লাসের কোনো পরিবর্তন না করে পরিবর্তিত নতুন ক্লাস তৈরি করা যায়। নতুন ক্লাস মূল ক্লাসের প্রয়োজনীয় বৈশিষ্ট্য ধারণ করে থাকে।
২. **এনক্যাপসুলেশন (Encapsulation):** কোনো চলকের ডেটা এবং ইনস্ট্রাকশন একত্রিত অবস্থায় থাকাকে এনক্যাপসুলেশন বলে। যে চলকের জন্য যে ডেটা সেই চলকের বাইরে তার আর কোনো অস্তিত্ব নেই। ফলে ডেটার ওপর প্রোগ্রামের অন্য অংশের কোনো প্রভাব পড়ে না।
৩. **পলিমরফিজম (Polymorphism):** পলিমরফিজম অর্থ হচ্ছে বহুরূপ। এ বৈশিষ্ট্যের জন্য কোনো কোড মডিউলের নাম এক হলেও একাধিক রূপ থাকতে পারে। কখন কোন রূপটি ব্যবহৃত হবে তা কম্পাইলার নির্ণয় করবে। একই অপারেটর ভিন্ন ধরনের ডেটার ওপর প্রয়োগ করা যেতে পারে। সঠিক ডেটা নিরূপণ করা কম্পাইলারের দায়িত্ব।

### ৫.১১.৩ ভিজুয়াল প্রোগ্রামিং (Visual Programming)

গ্রাফিক্যাল ইউজার ইন্টারফেস (GUI) সমৃদ্ধ পরিবেশকে বলা হয় ভিজুয়াল (Visual) বা দৃশ্যমান। চিত্রভিত্তিক বা ভিজুয়াল পরিবেশে কাজ করা সহজ বলে চিত্রভিত্তিক প্রোগ্রামিং ভাষা উদ্ভাবিত হয়েছে। চিত্রভিত্তিক পরিবেশ হলেও প্রোগ্রামিং-এর ক্ষেত্রে মূলত স্ট্রাকচার্ড প্রোগ্রামিং অথবা OOP মডেল ব্যবহৃত হয়। তবে প্রকৃত প্রোগ্রামিং মডেল প্রোগ্রাম থেকে আড়ালে থাকায় চিত্রভিত্তিক প্রোগ্রামিং বেশ জনপ্রিয়। মাইক্রোসফট কোম্পানির ভিজুয়াল বেসিক হলো প্রথম চিত্রভিত্তিক প্রোগ্রামিং মডেল। বর্তমানে ভিজুয়াল বেসিক ও ডেলফি দুটিই জনপ্রিয় ভিজুয়াল প্রোগ্রামিং সফটওয়্যার। ভিজুয়াল প্রোগ্রামিং প্রকৃতপক্ষে এক ধরনের অ্যাপ্লিকেশন প্রোগ্রাম। বড় আকারের বা দক্ষ প্রোগ্রাম রচনার জন্য এসব সফটওয়্যার খুব একটা ব্যবহৃত হয় না। তবু চিত্রভিত্তিক প্রোগ্রামিং-এর জনপ্রিয়তা দিন দিন বৃদ্ধি পাচ্ছে। ভিজুয়াল প্রোগ্রামিং এর উদাহরণ হলো— Visual C ++, Visual Basic, Microsoft Access, Oracle developers ইত্যাদি।

### ৫.১১.৪ ইভেন্ট ড্রাইভেন প্রোগ্রামিং (Event Driven Programming)

স্ট্রাকচার্ড এবং অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং মডেলের মূল বৈশিষ্ট্য হচ্ছে, প্রোগ্রামের শুরু থেকে শেষ পর্যন্ত নির্বাহ হওয়া। প্রোগ্রাম নির্বাহের প্রবাহ সিদ্ধান্তমূলক ও লুপ কাঠামো ব্যবহার করে কিছুটা নিয়ন্ত্রণ করা গেলেও ব্যবহারকারী প্রোগ্রাম পরিচালনাকালে পুরোপুরি স্বাধীনতা পায় না। উদাহরণস্বরূপ ১০,০০০টি সংখ্যা যোগ করার একটা প্রোগ্রাম স্ট্রাকচার্ড অথবা OOP মডেলে লিখলে তা ব্যবহারকারীকে সম্পূর্ণ সুবিধা দিতে পারে না। যেমন যোগ করাকালীন যেকোনো মুহূর্তে ব্যবহারকারী প্রক্রিয়াটিকে বাতিল করতে চাইলে, বা নতুন করে শুরু করতে চাইলে তা সম্ভব হয় না। ব্যবহারকারীকে এ ধরনের সুবিধা দেওয়ার জন্য উদ্ভাবিত হয়েছে ইভেন্ট ড্রাইভেন প্রোগ্রামিং মডেল। প্রোগ্রামের সাথে ব্যবহারকারীকে যেকোনো ধরনের মিথস্ক্রিয়াকে (Interaction) ইভেন্ট বলে। মাউস সরানো, ক্লিক করা, কি-বোর্ডে কোনো কি (Key) চাপা, প্রতিটিই এক একটি ইভেন্ট।

ইভেন্ট ড্রাইভেন প্রোগ্রামিং মডেলে কোনো প্রোগ্রামের সব নির্দেশ ধারাবাহিকভাবে নির্বাহ হয় না। প্রোগ্রামটিতে বিভিন্ন ইভেন্টের জন্য সংশ্লিষ্ট কোড মডিউল থাকে যা কেবল ঐ ইভেন্ট উৎপন্ন হলেই নির্বাহ হয়। অন্যথায় তা অবসর বসে থাকে। যেমন- মাইক্রোসফট ওয়ার্ডে মেনু ও টুলবার প্রদর্শিত হয়। প্রোগ্রামের শুরুতে এগুলো কিছুই করে না। কিন্তু ব্যবহারকারী যখনই কোনো মেনুবার/টুলবারের বোতামে ক্লিক করে তখনই একটি ইভেন্টের সৃষ্টি হয়। সাথে সাথে ইভেন্টের সংশ্লিষ্ট কোড দ্বারা মডিউলটি নির্বাহ হয়। সব ভিজুয়াল প্রোগ্রামই হচ্ছে ইভেন্ট ড্রাইভেন প্রোগ্রাম। যেমন: Visual Basic, Microsoft Access, Oracle Developers ইত্যাদি।



কাজ:

প্রত্যেক প্রোগ্রাম মডেলের তিনটি করে বৈশিষ্ট্য ও আওতাভুক্ত দুটি প্রোগ্রাম ভাষার নাম লেখ।



## ৫.১২ ‘সি’ প্রোগ্রামিং ভাষা (Programming Language-C)



চিত্র: ডেনিস রিচি  
[জন্ম: ৯ সেপ্টেম্বর, ১৯৪১  
মৃত্যু: ১২ অক্টোবর, ২০১১]

সি একটি স্ট্রাকচার্ড বা প্রোসিডিউর অরিয়েন্টেড প্রোগ্রামিং ল্যাঙ্গুয়েজ। এটি মিদ লেভেল ল্যাঙ্গুয়েজ হিসেবে জনপ্রিয়। সি নামটা এসেছে মার্টিন রিচার্ডস (Martins Richards) এর উদ্ভাবিত বিসিপিএল (BCPL-Basic Combined Programming Language) ভাষা থেকে যা প্রাথমিকভাবে ক্যামব্রিজ বিশ্ববিদ্যালয়ে রিসার্চ অরিয়েন্টেড কাজে ব্যবহৃত হতো। BCPL সংক্ষেপে বি নামে পরিচিত ছিলো। পরে বি এর উন্নয়নের ফলে সি এর বিকাশ ঘটে। ১৯৭০ সালে যুক্তরাষ্ট্রের টিএন্ডটি বেল ল্যাবোরেটরিতে (AT&T Bell Laboratory) ডেনিস রিচি (Dennis Ritchie) DEC PDP-IT কম্পিউটারে ব্যবহারের জন্য ইউনিক্স (UNIX) অপারেটিং সিস্টেম ব্যবহার করে সি (C) প্রোগ্রাম ভাষা উদ্ভাবন করেন। প্রথম দিকে সি কেবল ইউনিক্স অপারেটিং সিস্টেম পরিবেশে লেখা হতো। ১৯৭৮ সালে ডেনিস রিচির লেখা “দ্যা সি প্রোগ্রামিং ল্যাঙ্গুয়েজ” বইটি প্রকাশের পর এবং মাইক্রোকম্পিউটারের জনপ্রিয়তা বাড়ার সাথে সাথে সি এর ব্যাপক প্রচলন শুরু হয়।

### ৫.১২.১ সি প্রোগ্রামিং এর প্রাথমিক ধারণা (Primary Concept of C programming)

‘সি’ দিয়ে সহজে উচ্চ স্তরের এবং নিম্নস্তরের ভাষার মধ্যে সমন্বয় করা যায়। আবার উচ্চ স্তরের ভাষার (যেমন- ফরট্রান) মতো বিট, বাইট, ও মেমোরি অ্যাড্রেসের পরিবর্তে বিভিন্ন ডেটা টাইপ ভেরিয়েবল নিয়ে কাজ করা যায়। তাছাড়া সি এর প্রোগ্রামিং কৌশল নিম্নস্তরের ভাষার মত কঠিন নয় আবার উচ্চ স্তরের ভাষার মত সহজও নয়। সি দিয়ে ইচ্ছামতো হার্ডওয়ার নিয়ন্ত্রণ করে প্রোগ্রাম তৈরি করা যায় এবং এইসব প্রোগ্রামগুলি বেশ নমনীয় হয়। এই জন্য ‘সি’ কে মধ্যবর্তী (Mid Level) কম্পিউটারের ভাষা বলা হয়।

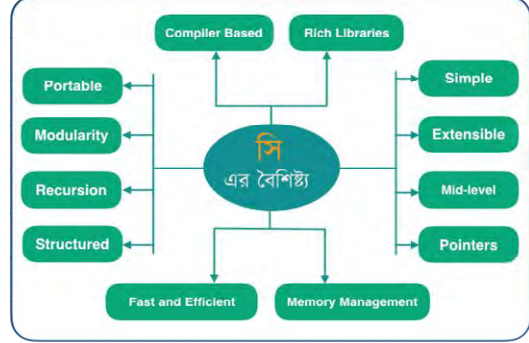
‘সি’ কে স্ট্রাকচার্ড বা প্রোসিডিউর অরিয়েন্টেড প্রোগ্রামিং ল্যাঙ্গুয়েজ বলা হয়, কারণ ‘সি’তে মূল সমস্যাকে কতগুলো ছোট ছোট অংশে বিভক্ত করে আলাদাভাবে ভেরিয়েবল, স্ট্রাকচার, ফাংশন ইত্যাদি বর্ণনা করা যায়। প্রয়োজনে if, while, for, goto ইত্যাদি কন্ট্রোল স্ট্রাকচারের মাধ্যমে বিভিন্ন অংশের মধ্যে সমন্বয় সাধন করা যায়, কিংবা কোনো ফাংশন বা স্ট্রাকচার পুনঃব্যবহার করা যায়। আনস্ট্রাকচার্ড ভাষায় (যেমন, বেসিক) এভাবে মূল সমস্যাকে একাধিক অংশে বিভক্ত করে আলাদাভাবে ফাংশন বর্ণনা করা যায় না। ‘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’ ভাষাতে আছে। তাই যেকোনো ধরনের প্রোগ্রাম লিখতে সি ভাষা ব্যবহার করা যায়। এই জন্য ‘সি’ ভাষাকে একটি **General Purpose প্রোগ্রামিং ভাষা** বলা হয়। এজন্য ‘সি’-কে যাবতীয় উচ্চ স্তরের ভাষা শেখার সিঁড়ি হিসেবে অবহিত করা হয়।

### ৫.১২.২ ‘সি’ প্রোগ্রামিং ভাষার বৈশিষ্ট্য (Characteristics of ‘C’ language)

কম্পিউটার প্রোগ্রাম ডিজাইনে সি ভাষা একটি সুশৃঙ্খল পদ্ধতি এবং কাঠামো প্রদান করেছে। ‘সি’ ভাষার নিম্নলিখিত বৈশিষ্ট্য পাওয়া যায়-

১. **প্রোসিডিউরাল ল্যাংগুয়েজ:** সি এর মতো প্রোসিডিউরাল (procedural) ল্যাংগুয়েজে পূর্বনির্ধারিত কিছু ইন্সট্রাকশন ধাপে ধাপে সম্পন্ন হয়। একটি কাজ সম্পন্ন করার জন্য একটি আদর্শ সি প্রোগ্রামে এক বা একের অধিক প্রোসিডিউর বা ফাংশন থাকতে পারে।
২. **সি প্রোগ্রাম দ্রুততর:** সি সরাসরি কম্পিউটার হার্ডওয়ারের মাধ্যমে প্রোগ্রাম সম্পাদনে সম্মতি দেয়। হাই-লেভেল ল্যাংগুয়েজে অনেক ধরনের বৈশিষ্ট্য থাকে যেমন- গার্বজ কালেকশন (Garbage Collection) এবং ডাইনামিক টাইপিং (Dynamic Typing)। ফলে সেই প্রোগ্রাম ধীর গতি সম্পন্ন হয়।



৩. **বহনযোগ্য:** সি প্রোগ্রাম বহনযোগ্য (portable), এই কথার অর্থ হলো “একবার লিখে সকল প্লাটফর্মে কম্পাইল করা যায়”। এক সিস্টেম (যেমন- উইন্ডোজ)-এর জন্য লেখা প্রোগ্রাম কোনো ধরনের পরিবর্তন ছাড়াই অন্য প্লাটফর্ম (যেমন- লিনাক্স)-এ কম্পাইল করা যায়।
৪. **মডিউলারিটি (Modularity):** সি প্রোগ্রামকে ভিন্ন ভিন্ন ফাংশনে ভাগ করে লাইব্রেরির মধ্যে রাখা যায়। প্রোগ্রামিং এর এই ধারণা মডিউলারিটি (modularity) নামে পরিচিত। তাই সি একটি মডিউলার প্রোগ্রামিং ভাষা।
৫. **স্ট্যাটিক্যালি টাইপ ল্যাংগুয়েজ:** সি ভাষায় ভ্যারিয়েবলের টাইপ রান টাইমে নয় বরং কম্পাইল টাইমে চেক হয়। তাই সি কে একটি স্ট্যাটিক্যালি টাইপ ল্যাংগুয়েজ (Statically Typed Language) বলে। ইহা সফটওয়্যার ডেভেলপমেন্ট সাইকেলের সময় ভুল (Error) খুঁজতে সহায়তা করে। এছাড়া ডাইনামিক্যালি টাইপ ল্যাংগুয়েজের তুলনায় স্ট্যাটিক্যালি টাইপ ল্যাংগুয়েজ সাধারণত দ্রুততর হয়।

#### ৫.১২.৩ ‘সি’ প্রোগ্রামিং ভাষার সুবিধা (Advantage of C Programming Language)

- এ ভাষার স্টেটমেন্ট গুলো ইংরেজি ভাষার মতো হওয়ায় শেখা সহজ।
- প্রোগ্রাম রচনা করা সহজ।
- প্রোগ্রামের মধ্যে যেকোনো স্থানে কমেন্ট দেওয়া যায়।
- উচ্চস্তরের ও মেশিন ভাষার প্রোগ্রামে লেখা যায়।
- অল্প মেমোরির প্রয়োজন হয়।
- প্রোগ্রাম ডিবাগিং করা সহজ।
- মেনুর সাহায্যে বিভিন্ন নির্দেশ ব্যবহার করে কাজ করা যায়।
- একই সাথে একাধিক ফাইল ও উইন্ডো নিয়ে কাজ করা যায়।
- ব্যবহারকারীর তৈরি ফাংশন ব্যবহারের সুবিধা।
- দ্রুত প্রোগ্রাম নির্বাহ করা যায়।
- কোনো লাইনের নম্বর দিতে হয় না।

#### ৫.১২.৪ ‘সি’ প্রোগ্রামিং ভাষার অসুবিধা (Disadvantage of C Programming Language)

- ‘সি’ ভাষা কেস সেনসিটিভ ভাষা ফলে ছোট হাতের অক্ষর এবং বড় হাতের অক্ষরের মধ্যে পার্থক্য পরিলক্ষিত হয়। এই ভাষায় প্রোগ্রাম সব সময় ছোট হাতের অক্ষরে লিখতে হয়।
- পর্যাপ্ত আধুনিক ফাংশন নেই ফলে আধুনিক প্রোগ্রামিং এনভায়রনমেন্টকে হ্যান্ডেলিং করা যায় না।
- সি ভাষায় নেম স্পেস অগ্রাহ্য করে।

- সি ভাষায় সঠিকভাবে চলক ঘোষণা করতে হয়।
- লাইব্রেরি ফাংশনের হেডার ফাইলগুলো ঠিকমত ডিক্লেয়ার করতে হয়।
- 'সি' প্রোগ্রামিং ল্যাংগুয়েজ অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ফিচারকে সমর্থন করে না।
- প্রোগ্রাম রান করার সময় চেকিং করা যায় না।

সি দিয়ে তৈরি করা হয়েছে এমন কিছু প্রোগ্রাম-এর তালিকাঃ

বহুল ব্যবহৃত অপারেটিং সিস্টেম উইন্ডোজ (Windows) 'সি' প্রোগ্রামিং ভাষা দিয়ে তৈরি করা হয়েছে। নিম্নে আরও কিছু তালিকা দেওয়া হলো:

- অপারেটিং সিস্টেম (Windows, DOSBox)
- কম্পাইলার (Compilers)
- অ্যাসেম্বলার (Assemblers)
- টেক্সট এডিটর (Text Editors)
- প্রিন্ট স্পুলার (Print Spoolers)
- নেটওয়ার্ক ড্রাইভার (Network Driver)
- ডেটাবেজ (Database)
- ভাষাবূপান্তরক (Interpreters)

যেখানে 'সি' প্রোগ্রাম লিখতে এবং রান করতে হয়: প্রোগ্রামিং সি লেখার জন্য IDE (Integrated Development Environment) এর দরকার হয়। IDE হলো একটি সাধারণ টেক্সট এডিটরের মতো যেখানে প্রোগ্রাম রান করার জন্য নানা রকম টুলস (Tools) দেওয়া থাকে। কিছু IDE এর উদাহরণ হলো—

- Code::Blocks (কোড ব্লকস)
- Microsoft Visual C++ ( মাইক্রোসফট ভিজ্যুয়াল সি/ সি++)
- Bloodshed Dev-C++ ( ব্লাডশেড ডেভ-সি++)
- Turbo C/C++ (টার্বো সি/ সি++)
- Borland C/C++ ( বোরল্যান্ড সি/ সি++) )

#### ৫.১২.৫ প্রোগ্রাম কম্পাইলিং (Compiling of Programs)

C ভাষায় লিখিত কোনো প্রোগ্রামকে সোর্স কোড বা সোর্স প্রোগ্রাম বলা হয়। সোর্স প্রোগ্রামকে কম্পাইলারের সাহায্যে এক সাথে সম্পূর্ণরূপে মেশিন ভাষায় অনুবাদ করে একটি অবজেক্ট প্রোগ্রাম ও একটি এক্সিকিউশন ফাইলে রূপান্তর করা হয়। এক্সিকিউশন ফাইলটিই হচ্ছে মূল ফাইল যার দ্বারা প্রোগ্রাম রান (Run) করানো হয়। আর এই প্রক্রিয়াকে বলে কম্পাইলিং। প্রোগ্রাম কম্পাইলিং এর মাধ্যমে প্রোগ্রামকে ত্রুটিমুক্ত করা হয়।

কম্পাইলিংয়ের কাজসমূহ:

- উৎস প্রোগ্রামকে অনুবাদ করে অবজেক্ট প্রোগ্রাম তৈরি করা।
- প্রোগ্রামকে লিংক করা। অর্থাৎ প্রোগ্রামের সঙ্গে প্রয়োজনীয় রুটিন যোগ করা। রুটিন হলো প্রোগ্রামের ছোট অংশ যাতে কোনো নির্দিষ্ট কাজ করার জন্য নির্দেশ দেয়া থাকে।
- প্রোগ্রামে কোনোভুল থাকলে তা প্রকাশ করা।
- প্রধান মেমোরিতে প্রোগ্রামের জন্য প্রয়োজনীয় জায়গা তৈরি করা।

সি প্রোগ্রাম কম্পাইল করার ধাপসমূহ (Steps of Programs Compiling): সি ভাষাকে কম্পাইল করতে বিভিন্ন ধরনের কম্পাইলারের প্রচলন রয়েছে। কম্পাইলার হচ্ছে একটি সফটওয়্যার মতো যা প্রোগ্রামিং ভাষাকে মেশিন বা কম্পিউটারের ভাষায় অর্থপূর্ণ করে তোলে। কয়েকটি জনপ্রিয় সি কম্পাইলার হলোঃ যেমন- GCC-GNU

Compiler, Clang, Intel C++ Compiler, Borland Turbo C ইত্যাদি। Turbo কম্পাইলারের ক্ষেত্রে কম্পাইলিং করার পদ্ধতি নিম্নরূপ:

- এই কম্পাইল করার জন্য Compile মেনু হতে Compile সাব মেনু সিলেক্ট করতে হয় বা Alt+F9 কী একত্রে চাপতে হয়। অবশ্য শুধু F9-কী press করেও কম্পাইল করা যায়।
- প্রোগ্রাম কম্পাইল করার পর প্রোগ্রাম রান করার জন্য Run মেনু থেকে Run সাব মেনু সিলেক্ট করতে হয় অথবা Ctrl+F9 কি একত্রে চাপলে প্রোগ্রামটি রান হবে। অন্যান্য কম্পাইলের ক্ষেত্রে কমান্ড ভিন্ন হতে পারে।

### ৫.১২.৬ প্রোগ্রামের গঠন (Structure of Programs)

এক বা একাধিক ফাংশন নিয়ে 'সি' প্রোগ্রাম গঠিত হয়। তবে 'সি' প্রোগ্রামে অবশ্যই main ফাংশন থাকতে হয়। একটি সি প্রোগ্রামে প্রধানত দু'টি অংশ থাকে যথা: হেডার ফাইল ও main ফাংশন। তবে বড় প্রোগ্রামের ক্ষেত্রে main ফাংশনে ব্যবহৃত ফাংশনসমূহ আলাদা সোর্স ফাইলে বর্ণিত হয় এবং main ফাংশনকে পৃথক ফাইলে রাখা হয়। এরপর main ফাংশনে হেডার ফাইলসহ অন্যান্য ফাইলসমূহ সংযুক্ত করা হয়।

নিম্নে সি-এর গঠন প্রণালি দেখানো হলো-

Documentation Section	মন্তব্য লেখা
Link Section	হেডার ফাইল সংযুক্ত রাখা
Defination Section	ধ্রুবমানের ব্যবহার
Global Declaration Section	গ্লোবাল ভেরিয়েবল ঘোষণা
Main Function Section return-type main() { Declaration Part Execution Part }	মেইন ফাংশন বিভাগ মেইন ফাংশন শুরুর ব্রাকেট ভেরিয়েবল ঘোষণা প্রসেস ভ্যালু মেইন ফাংশন শেষ ব্রাকেট
Sub-function Section Function 1 Function 2 .. .. Function N	ব্যবহারকারীর নিজস্ব ফাংশন লেখা ব্যবহারকারীর নিজস্ব ফাংশন, Function 1

- Documentation Section:** সি প্রোগ্রামের এটি ঐচ্ছিক অংশ। এখানে প্রোগ্রামের নাম, বিষয়বস্তু, প্রোগ্রামারের নাম, ব্যবহারের নিয়ম ও প্রোগ্রামের উদ্দেশ্য সংযুক্ত করা হয়। সাধারণত কমেন্টস এর মাধ্যমে এগুলো উল্লেখ করা হয়। সি ভাষায় কমেন্ট লেখার জন্য /\*.....\*/ এবং //....., ব্যবহার করা হয়। যেমন: // This is my first program in C.

- Link Section:** এটি প্রোগ্রামের অত্যাৱশ্যকীয় অংশ। প্রোগ্রামের বিভিন্ন স্থানে ব্যবহৃত ফাংশনগুলোর হেডার ফাইল এ অংশে সংযুক্ত করা হয়। হেডার ফাইল সংযোগের সিনটেক্স হলো-

```
#include<header_file_name>
```

ইনপুট ফাংশন scanf() ও আউটপুট ফাংশন Printf() এর হেডার ফাইল হলো stdio.h যা নিম্নের মতো লেখা হয়-

```
#include<stdio.h >
```

- **Defination Section:** এ অংশে কনস্ট্যান্ট ঘোষণা করা হয়। PI একটি কনস্ট্যান্ট যার মান 3.1416। PI কনস্ট্যান্টকে নিম্নোক্ত ভাবে ঘোষণা করা হয়।  
#define PI 3.1416 অথবা const float PI=3.1416;
- **Global Declaration Section:** এ অংশে গ্লোবাল চলক ঘোষণা করা হয়।
- **Main Function Section :** main() হলো 'সি' প্রোগ্রামের প্রধান ফাংশন। এটি একটি ইউজার ডিফাইন্ড বা ব্যবহারকারী বর্ণিত ফাংশন। 'সি' প্রোগ্রামের মূল অংশ main () ফাংশনের আওতায় {} বন্ধনীর মধ্যে লিখতে হয়। 'সি' প্রোগ্রাম যত বড় বা ছোট হোক না কেন, ফাংশন সংলগ্ন দ্বিতীয় বন্ধনীর পরবর্তী স্টেটমেন্ট থেকে প্রোগ্রাম নির্বাহ শুরু হয়। এই ফাংশন ছাড়া কোনো 'সি' প্রোগ্রাম লেখা সম্ভব নয়। main() ফাংশনের দুটি অংশ থাকে। যথা: একটি Declaration Part এবং অন্যটি Execution Part। Declaration Part-এ প্রয়োজনীয় চলক, অ্যারে, পয়েন্টার, ফাইল ইত্যাদি ঘোষণা করা হয় যা নির্বাহ অংশে ব্যবহার করা যায়। Execution Part-এ প্রোগ্রাম নির্বাহ হওয়ার জন্য কমপক্ষে একটি স্টেটমেন্ট থাকতে হয়। উভয় অংশের প্রত্যেক স্টেটমেন্টের শেষে সেমিকোলন (;) থাকতে হয়।
- **Sub-function Section:** এক বা একাধিক ইউজার ডিফাইন্ড ফাংশন থাকলে তা main() ফাংশন থেকে কল করা হয়। ব্যবহারকারী কোনো ফাংশন তৈরি করে প্রোগ্রামে ব্যবহার করতে চাইলে তা main() ফাংশনের দ্বিতীয় বন্ধনীর বাইরে (উপরে বা নিচে) তৈরি করতে হয়।

একটি প্রোগ্রামের সাহায্যে 'সি' ভাষার স্ট্রাকচার বোঝানো হলো:

```
// This program calculate the area of circle
#include<stdio.h>
#define PI 3.1416
int r;
float area(int r);
int main()
{
    printf("Type the radius: ");
    scanf("%d",&r);
    printf("Area of Circle=%.2f",area(r));
    return 0;
}
float area(int r)
{
    float area;
    area=PI*r*r;
    return area;
}
```

Diagram labels for the code above:

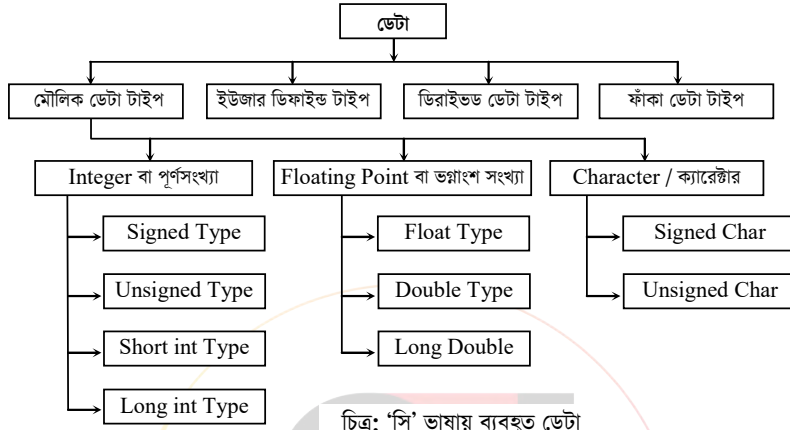
- // This program calculate the area of circle → Documentation Section
- #include<stdio.h> → Link Section
- #define PI 3.1416 → Defination Section
- int r; → Global Declaration Section
- float area(int r); → Global Declaration Section
- int main() → main() ফাংশন শুরু
- {
- printf("Type the radius: ");
- scanf("%d",&r); → ফাংশন কল
- printf("Area of Circle=%.2f",area(r));
- return 0;
- } → main() ফাংশন শেষ
- float area(int r) → Sub-function Section
- {
- float area;
- area=PI\*r\*r;
- return area;
- }

## ৫.১২.৭ ডেটা টাইপ (Data Type)

‘সি’ প্রোগ্রামে অনেক ধরনের ডেটা নিয়ে কাজ করা যায়, যেমন- পূর্ণ সংখ্যা, ভগ্নাংশ, ক্যারেক্টার, স্ট্রিং ইত্যাদি। ডেটার ধরন এবং মেমোরি পরিসর সংরক্ষণের ভিত্তিতে সি প্রোগ্রামে ব্যবহৃত ডেটাকে প্রধানত চারটি ভাগে ভাগ করা হয়।

যথা- i) char, ii) int, iii) float, iv) double।

এদেরকে বেসিক বা মৌলিক অথবা বিল্টইন ডেটা টাইপ বলা হয়। প্রয়োজনে আবার নিজস্ব ডেটা টাইপ তৈরি করে নেয়া যায়। এরূপ ডেটা টাইপকে ইউজার ডিফাইন্ড বা কাস্টম ডেটা টাইপ বলা হয়। চিত্রে ‘সি’ প্রোগ্রামে ব্যবহৃত বিভিন্ন প্রকার বিল্টইন, মডিফাইড এবং কাস্টম ডেটা টাইপের শ্রেণিবিন্যাস দেখানো হলো।



**I) char টাইপ:** সি প্রোগ্রামে ক্যারেক্টার টাইপ ডেটা (বর্ণ বা বর্ণমালা) নিয়ে কাজ করার জন্য char টাইপ ভেরিয়েবল ব্যবহার করা হয়। প্রতিটি char টাইপ ভেরিয়েবলের জন্য কম্পাইলার ১ বাইট বা ৮ বিট জায়গা সংরক্ষণ করে। char টাইপ ডেটার জন্য সিনট্যাক্স (Syntax) হলো-

**char VariableName;**

উদাহরণ : char ch = 'a';

এখানে a এর স্থলে যেকোনো চিহ্ন বা বর্ণ ব্যবহার করা যাবে।

**II) int টাইপ:** সি প্রোগ্রামে পূর্ণসংখ্যা (যেমন, ২০, -৪৬৭, ৮৯০) ইত্যাদি নিয়ে কাজ করার জন্য int টাইপ ভেরিয়েবল ব্যবহার করা হয়। প্রতিটি int টাইপ ভেরিয়েবলের জন্য কম্পাইলার ২ বাইট বা ১৬ বিট জায়গা সংরক্ষণ করে। int টাইপ ডেটার জন্য সিনট্যাক্স (Syntax) হলো-

**int VariablesName;**

উদাহরণ : int num1 = 5;

**III) float টাইপ:** সি প্রোগ্রামে রিয়েল বা ভগ্নাংশসহ কোনো সংখ্যা (যেমন, ২০.৩৪, -৪৬.৮৭, ৮৯.৭০) নিয়ে কাজ করার জন্য float টাইপ ভেরিয়েবল ব্যবহার করা হয়। প্রতিটি float টাইপ ভেরিয়েবলের জন্য কম্পাইলার ৪ বাইট বা ৩২ বিট জায়গা সংরক্ষণ করে। float ডেটা টাইপে দশমিক বিন্দুর পর ৬ ডিজিট পর্যন্ত গ্রহণযোগ্য। float টাইপ ডেটার জন্য সিনট্যাক্স (Syntax) হলো-

**float VariableName;**

উদাহরণ : float num1 = 5.06;



**IV) double টাইপ:** সি প্রোগ্রামে এক্সপোনেন্সিয়াল বা সায়েন্টিফিক ফরমেটের  $(4.5*20^{20}, -12*10^{-20})$  সংখ্যার ক্ষেত্রে double ডেটাটাইপ ব্যবহার করা হয়। double ডেটাটাইপে দশমিক বিন্দুর পর 14 ডিজিট পর্যন্ত গ্রহণযোগ্য। প্রতিটি double টাইপ ভেরিয়েবলের জন্য কম্পাইলার মেমোরিতে ৮ বাইট বা ৬৪ বিট জায়গা সংরক্ষণ করে। double টাইপ ডেটার জন্য সিনট্যাক্স (Syntax) হলো-

**double VariableName;**

উদাহরণ : double num1=23.5454353

নিম্নে বিভিন্ন ডেটাটাইপের জন্য ব্যবহৃত কীওয়ার্ড উদাহরণসহ দেখানো হলো-

ডেটাটাইপ	ব্যবহৃত কী ওয়ার্ড	উদাহরণ
Character	char	char a; char a, b, c;
Integer	int	int a; int a, b, c;
Float	float	float a; float a, b, c;
Double	double	double a; double a, b, c;

#### ৫.১২.৮ ডেটা টাইপ মডিফায়ার (Data type Modifier)

ভিন্ন ভিন্ন ডেটা টাইপের প্রয়োজনে একই ভেরিয়েবলের জন্য সংরক্ষিত মেমোরি পরিসর আলাদা হয়। যে সকল কীওয়ার্ড ব্যবহার করে ফ্লোট (float) ছাড়া অন্যান্য মৌলিক বা প্রাথমিক ডেটা টাইপের পরিসর ও সংরক্ষণের জন্য মেমোরি পরিমাণ বাড়ানো বা কমানো যায় এদেরকে ডেটা টাইপ মডিফায়ার বলে। সহজ কথায় বলা যায়, সি ভাষায় ব্যবহৃত মৌলিক ডেটা টাইপগুলোর জন্য সংরক্ষিত মেমোরি পরিসর, ডেটার প্রকৃতি এবং ধারণ ক্ষমতার পরিবর্তন বা পরিবর্ধন করার জন্য ব্যবহৃত কীওয়ার্ডসমূহকে মডিফায়ার বা টাইপ কোয়ালিফায়ার বলা হয়। সি'তে মোট চারটি মডিফায়ার আছে। যথা- signed, unsigned, short, long।

কোন টাইপের জন্য কোন মডিফায়ার ব্যবহৃত হয়: সাধারণত: char টাইপ ভেরিয়েবলের জন্য signed ও unsigned মডিফায়ার, int টাইপ ভেরিয়েবলের জন্য signed, unsigned, short ও long এবং double টাইপ ভেরিয়েবলের জন্য long মডিফায়ার ব্যবহৃত হয়। মডিফায়ার সর্বদা ডেটা টাইপের পূর্বে বসে। নিচের ছকে বিভিন্ন প্রকার বেসিক এবং মডিফাইড ডেটা টাইপ ভেরিয়েবলের জন্য সংরক্ষিত মেমোরি স্পেসের পরিমাণ এবং ডেটার রেঞ্জ দেয়া হলো।

মৌলিক ডেটা টাইপ	মডিফাইড ডেটা টাইপ	সাইজ (বিট)	ডেটা রেঞ্জ
ইন্টিজার বা পূর্ণসংখ্যা	Int or signed int	১৬	-৩২,৭৬৮ থেকে +৩২,৭৬৭ বা $-2^{31}$ থেকে $+(2^{31}-1)$
	Unsigned int	১৬	০ থেকে ৬৫,৫৩৬ বা ০ থেকে $+(2^{16}-1)$
	Signed long int	৩২	-২১৪,৭৪,৮৩,৬৪৮ থেকে +২১৪,৭৪,৮৩,৬৪৭ বা $-2^{31}$ থেকে $+(2^{31}-1)$
	Unsigned long int	৩২	০ থেকে ৪২৯,৪৯,৬৭,২৯৫ বা ০ থেকে $+(2^{32}-1)$
ফ্লোটিং পয়েন্ট বা ভগ্নাংশ	Float	৩২	$3.8 \times E-38$ থেকে $3.8 \times E+38$
	Double	৬৪	$1.9 \times E-308$ থেকে $1.9 \times E+308$
	Long double	৮০	$3.8 \times E-8932$ থেকে $3.8 \times E+8932$
ক্যারেক্টার	Char or signed char	৮	-১২৮ থেকে +১২৭ বা $-2^7$ থেকে $+(2^7-1)$
	Unsigned char	৮	০ থেকে ২৫৫ বা ০ থেকে $(2^8-1)$

চিত্র: 'সি' ডেটা টাইপ



কাজ:

একজন ছাত্রের নাম, বয়স, শ্রেণি, রোল নম্বর, বিভাগ, ফি ইত্যাদির জন্য কোন কোন ডেটা টাইপ ব্যবহৃত হবে তা ছকের মাধ্যমে দেখাও।

## পাঠ ১৭-১৯

## ‘সি’ ভাষায় ব্যবহৃত ধ্রুবক ও চলক

## সি প্রোগ্রামে ডেটার পরিচায়ক (Identifier)

প্রোগ্রামিংয়ের সুবিধার্থে সরাসরি সাংখ্যিক অ্যাড্রেস ব্যবহার না করে প্রতিটি অ্যাড্রেসকে একটি নাম দেওয়া হয়। এই নামকে পরিচায়ক বা আইডেন্টিফায়ার বলা হয়। আইডেন্টিফায়ার প্রধানত দুটো শ্রেণিতে ভাগ করা হয়। যথা-

- ধ্রুবক
- চলক

## ৫.১২.৯ ধ্রুবক (Constant)

ধ্রুবক যা একটি নির্দিষ্ট মান ধারণ করে। প্রোগ্রামে অপরিবর্তনশীল মানকে কনস্ট্যান্ট বা ধ্রুবক বলা হয়। সেক্ষেত্রে প্রোগ্রামে ঐ মানকে কনস্ট্যান্ট হিসেবে ঘোষণা করা হয়। প্রোগ্রাম নির্বাহের সময় কোনো অবস্থাতেই কনস্ট্যান্ট বা ধ্রুবকের মান পরিবর্তন করা যায় না। কোনো সংখ্যা বা মান দ্বারা কনস্ট্যান্টের মান নির্ধারণ করা যায় না। তবে কনস্ট্যান্ট দ্বারা ভেরিয়েবলের মান নির্ধারণ করা যায়। সি প্রোগ্রামে মোট দুইভাবে কনস্ট্যান্ট ঘোষণা করা যায়। যথা:

<b>১। Const কীওয়ার্ড ব্যবহার করে-</b> ফরম্যাট: <b>const DataType</b> <b>ConstName = ConstValue;</b> যেমন: const int Max = 50; const float PI = 3.1416; const char Ch = 'a';	<b>২। #define প্রিপ্রসেসর ব্যবহার করে-</b> ফরম্যাট: <b>#define ConstName</b> <b>ConstValue</b> যেমন: #define Max 50 #define PI 3.1416 #define Ch 'a'
--	--

define ব্যবহারের সময় কেবল কনস্ট্যান্টের নাম ও প্রারম্ভিক মান দিতে হয়। তবে কোনো টাইপ উল্লেখ করতে হয় না ও মাঝে সমান চিহ্ন ও শেষে সেমিকোলন বসে না।

কনস্ট্যান্ট প্রধানত দুই ধরনের, যথা : নিউমেরিক ও স্ট্রিং কনস্ট্যান্ট।

১. **সংখ্যাসূচক ধ্রুবক বা নিউমেরিক কনস্ট্যান্ট:** এই ধরনের ধ্রুবক 0-9 পর্যন্ত অংক বা \$ দ্বারা শুরু হয়। মান -2147483648 থেকে 294967295 এর মধ্যবর্তী যেকোনো ঋণাত্মক বা ধনাত্মক হতে পারে। ধ্রুবকে কমা ব্যবহার করা যায় না; তবে প্রয়োজনে দশমিক চিহ্ন ব্যবহার করা যায়। সংখ্যাসূচক ধ্রুবক আবার নিম্নোক্ত পাঁচভাগে ভাগ করা যায় :

- **ইন্টিজার কনস্ট্যান্ট :** এ ধরনের ধ্রুবক ধনাত্মক বা ঋণাত্মক যে কোনো পূর্ণসংখ্যা হতে পারে। এ ধ্রুবকে দশমিক বিন্দু থাকে না। যেমন- 546, 45, 20000, -32768, +6532767 ইত্যাদি।
- **ফ্লোটিং পয়েন্ট কনস্ট্যান্ট :** এ ধরনের ধ্রুবক ধনাত্মক বা ঋণাত্মক পূর্ণ বা ভগ্নাংশবিশিষ্ট সংখ্যা হতে পারে। পূর্ণসংখ্যার জন্য দশমিক চিহ্নের ব্যবহার আবশ্যিক নয়, তবে আংশিক মানের জন্য এর বিকল্প নেই। যেমন- 56, 56.0, 0.55, 50, 3.141592, -45.678, +65.32767 ইত্যাদি।
- **এক্সপোনেনশিয়াল কনস্ট্যান্ট:** এ ধরনের ধ্রুবকও ধনাত্মক এবং ঋণাত্মক উভয় ধরনের হতে পারে। এ রকম সংখ্যা 10 এর সূচক বা ঘাত (Power) হিসাবে লেখা হয় এবং E অক্ষর দিয়ে বোঝানো হয়। এখানে E দিয়ে 10 এর সাথে যে সূচক সংখ্যাটি থাকে তার মান লেখা হয়। সূচক সংখ্যাটি যদি ধনাত্মক হয় তাহলে E এর পরে যোগ (+) এবং ঋণাত্মক হলে বিয়োগ (-) ব্যবহার করা যায়। যেমন, 3.5E+3, 3.5E-5, ইত্যাদি।
- **অক্টাল কনস্ট্যান্ট:** এ ধরনের সংখ্যার পূর্বে একটি শূন্য (0) বসাতে হয়, যেমন- 0348, 01234 ইত্যাদি। তবে ফলাফলে এই অতিরিক্ত শূন্য অগ্রাহ্য হয়।
- **হেক্সাডেসিম্যাল কনস্ট্যান্ট:** এ ধরনের সংখ্যার পূর্বে 0x লিখতে হয়, যেমন- 0x12, 0xA2B ইত্যাদি। তবে ফলাফলে এই অতিরিক্ত 0x অগ্রাহ্য হয়।

২. অক্ষরসূচক ধ্রুবক বা স্ট্রিং কনস্ট্যান্ট: বর্ণ, অঙ্ক ও অন্যান্য চিহ্ন সাজিয়ে এই ধ্রুবক গঠিত হয়। এই ধ্রুবককে সিঙ্গেল অথবা ডাবল কোটেশন দ্বারা নির্দিষ্ট করা হয়।
  - **Single Character Constant:** একটি বর্ণ যা Single Quotation Mark দ্বারা চিহ্নিত করা হয়। যেমন: 'A', '5' ইত্যাদি।
  - **String Constant:** অঙ্ক, বর্ণ বা অন্যান্য চিহ্নেও সমন্বয়ে এ ধ্রুবক গঠিত হয়। এ ধ্রুবক সিঙ্গেল বা ডাবল কোটেশন দ্বারা নির্দিষ্ট করা যায়। স্ট্রিং ধ্রুবককে ০ হতে ২৫৫ টি অক্ষর থাকতে পারে। যেমন- "A", "Khoksa", "58.58" ইত্যাদি।
  - **Backslash Character Constant:** বিশেষ কিছু ক্যারেক্টার আছে (যেমন, /, ", \n, \r, \t ইত্যাদি) যেগুলো printf() ফাংশনের ডাবল কোটেশনের (" ") মধ্যে যেভাবে ব্যবহার করা হয় কিন্তু ফলাফলে সেবুপ প্রদর্শিত হয় না। printf() বা এরূপ কোনো ফাংশন দ্বারা এসব ক্যারেক্টার প্রদর্শনের জন্য এই ক্যারেক্টার গুলোর সাথে অতিরিক্ত একটি ব্যাকস্লাশ (\) ক্যারেক্টার ব্যবহার করতে হয়, এগুলোকে ব্যাকস্লাশ বা ইস্কেপ সিকুয়েন্স ক্যারেক্টার সেট বলা হয়।

নিম্নের ছকে বহুল ব্যবহৃত কয়েকটি ব্যাকস্লাশ ক্যারেক্টার সেটের তালিকা দেওয়া হলো।

ব্যাকস্লাশ ক্যারেক্টার	ব্যবহার	উদাহরণ	আউটপুট
\n	নতুনলাইন তৈরিকরে	printf("This is \n Dhaka College");	This is Dhaka College
\t	ট্যাব এর মতো কাজ করে	printf("This is \t Dhaka College");	This is    Dhaka College
\r	লাইনের শুরুতে প্রদর্শনের জন্য	printf("This is \r Dhaka College"); printf("\t\tThis is \r Dhaka College");	Dhaka College Dhaka College This is
\a	সংকেত (Alarm) দানের জন্য	printf("\a\aThis is Dhaka College");	This is Dhaka College (একটি শব্দ সংকেত শোনা যাবে)
\b	ব্যাকস্পেস এর কাজ করে	printf("This is \bDhaka College");	This is Dhaka College
\"	ডাবল কোটেশন (") ক্যারেক্টার প্রদর্শনের জন্য	printf("This is \"Dhaka College\" ");	This is "Dhaka College"
\'	সিঙ্গেল কোটেশন (') ক্যারেক্টার প্রদর্শনের জন্য	printf("This is \' Dhaka College\' ");	This is 'Dhaka College'
\\	ব্যাকস্লাশ (\) প্রদর্শনের জন্য	printf("This is \\ Dhaka College");	This is \ Dhaka College
\?	প্রশ্নবোধক চিহ্ন প্রদর্শনের জন্য	printf("This is \? Dhaka College");	This is ? Dhaka College

ছক: কয়েকটি ব্যাকস্লাশ ক্যারেক্টার ও তাদের ব্যবহার

**কনস্ট্যান্ট ব্যবহারের নিয়ম:** কনস্ট্যান্ট ব্যবহারের কতগুলো সুনির্দিষ্ট নিয়ম আছে। যেমন:-

- প্রতিটি কনস্ট্যান্টের নাম থাকে।
- কনস্ট্যান্ট ঘোষণার সময়ই তার মান নির্ধারণ করে দিতে হয়।
- প্রোগ্রাম নির্বাহের সময় কোনো অবস্থাতেই মান পরিবর্তন করা যায় না।
- প্রয়োজনে প্রোগ্রামের যে কোনো জায়গায় কনস্ট্যান্ট ব্যবহার করা যায়।
- printf() ফাংশন দ্বারা কনস্ট্যান্ট মান প্রদর্শনের জন্য উপযুক্ত ফরম্যাট স্পেসিফায়ার ব্যবহৃত হয়।

### প্রোগ্রামে ধ্রুবক ব্যবহারের সুবিধা (Advantages of Constant)

- ধ্রুবক ব্যবহারে প্রোগ্রামে ভুলের পরিমাণ কমে যায় ও প্রোগ্রাম সহজবোধ্য হয়।
- প্রোগ্রামের কোড টাইপ করতে সময় কম লাগে।

### ৫.১২.১০ চলক (Variable)

ভেরিয়েবল হলো মেমোরির (RAM) লোকেশনের নাম বা ঠিকানা। প্রোগ্রামে যখন কোনো ডেটা নিয়ে কাজ করা হয়, প্রাথমিকভাবে সেগুলো কম্পিউটারের র্যামে অবস্থান করে। পরবর্তী সময়ে সেগুলো পুনর্বিন্যাস বা পুনব্যবহারের জন্য ঐ নাম বা ঠিকানা জানা প্রয়োজন হয়। সুতরাং প্রোগ্রামে ডেটা নিয়ে কাজ করার সময় প্রতিটি ডেটার জন্য একটি ভেরিয়েবল ব্যবহার করতে হয়। প্রতিবার প্রোগ্রাম নির্বাহের সময় মেমোরিতে ভেরিয়েবলগুলো অবস্থান ও সংরক্ষিত মান পরিবর্তন হয় বা হতে পারে বলে এদেরকে ভেরিয়েবল বা চলক বলা হয়।

একটি ভেরিয়েবলের নিম্নলিখিত বৈশিষ্ট্য থাকতে হবে-

- একটি সুনির্দিষ্ট নাম থাকতে হবে।
- এটি মেমোরিতে নির্দিষ্ট পরিমাণ জায়গা নেবে।
- একটি নির্দিষ্ট ডেটা টাইপ থাকবে।

চলক ঘোষণার সিনটেক্স হলো-

**DataType VariableName;** উদাহরণ: int number1;

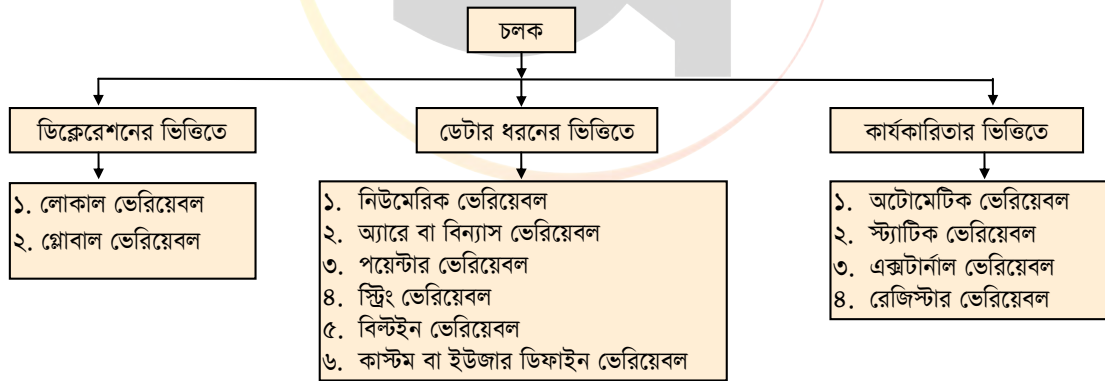
অথবা, **DataType VariableName=[Value];** উদাহরণ: int number1=60;

এখানে, ভেরিয়েবল বলতে number1 কে বুঝানো হয়।

### ভেরিয়েবল ব্যবহারের সুবিধা (Advantages of Variable):

ভেরিয়েবল ব্যবহার না করেও প্রোগ্রামে বিভিন্ন ধরনের ডেটা নিয়ে কাজ করা যায়। তবে সেক্ষেত্রে ডেটার স্বয়ংক্রিয় মান নির্ধারণ, পুনব্যবহার প্রভৃতি সুবিধা পাওয়া যায় না। উদাহরণ হিসেবে বলা যায়, কারো যদি লক্ষ লক্ষ বন্ধু থাকে, তবে যতই আন্তরিক হোক না কেন তারা কে কোন রুমে থাকে তা মনে রাখা সম্ভবপর নয়। কিন্তু তারা যদি তাদের নাম, রোল বা আইডি নাম্বারের অনুরূপ নামবিশিষ্ট রুমে থাকে তবে সহজেই তাদের খুঁজে বের করা সম্ভব হবে। মূলত প্রোগ্রামে ভেরিয়েবল ব্যবহারের মাধ্যমে মেমোরিতে ভেরিয়েবলের নামবিশিষ্ট লোকেশনে ডেটা সংরক্ষণ করা হয়। যা পরবর্তী সময়ে সেগুলো খুঁজে পাওয়া সহজ হয়।

**ভেরিয়েবল বা চলকের প্রকারভেদ (Classification of Variable):** বিভিন্ন দৃষ্টিকোণ থেকে ভেরিয়েবলকে ভাগ করা যায়। নিম্নে ছকের মাধ্যমে তা দেখানো হলো—



ঘোষণা বা অবস্থানের ভিত্তিতে দুই ধরনের ভেরিয়েবল ব্যবহৃত হয়। যথা:

- **লোকাল ভেরিয়েবল (Local Variable):** যখন কোনো ভেরিয়েবল কোনো ফাংশনের মধ্যে বা ফাংশন বডিতে ঘোষণা করা হয় তখন সেই ভেরিয়েবলকে ঐ ফাংশনের সাপেক্ষে লোকাল ভেরিয়েবল বলা হয়। লোকাল ভেরিয়েবলের

মান ও অস্তিত্ব শুধুমাত্র সংশ্লিষ্ট ফাংশনের মধ্যে সীমাবদ্ধ থাকে। এই মান অন্য ফাংশনে সরাসরি ব্যবহার করা যায় না। তবে লোকাল ভেরিয়েবল বিশিষ্ট কোনো ফাংশন অন্য কোনো ফাংশনে কল করে ব্যবহারকারী ফাংশনে পরোক্ষভাবে লোকাল ভেরিয়েবলের মান ব্যবহার করতে পারে। কম্পাইলার যতক্ষণ একটি ফাংশন নিয়ে কাজ করে ততক্ষণ পর্যন্ত ঐ ফাংশনের লোকাল ভেরিয়েবলগুলো সক্রিয় থাকে। কোনো ফাংশনের কার্যক্রম শেষে কম্পাইলার স্বয়ংক্রিয়ভাবে লোকাল ভেরিয়েবলগুলোর জন্য বরাদ্দকৃত মেমোরি পরিসর খালি করে দেয়। ফলে দুই বা ততোধিক ফাংশনে একই নাম ও ডেটা টাইপের লোকাল ভেরিয়েবল ব্যবহার করা যেতে পারে এবং তাতে কোনো সমস্যা হয় না। প্রয়োজনে লোকাল ভেরিয়েবলের কার্যক্রম ফাংশনের একটি নির্দিষ্ট ব্লকের মধ্যেও সীমাবদ্ধ করে দেয়া যায়।

- **গ্লোবাল ভেরিয়েবল (Global Variable):** যখন কোনো ভেরিয়েবল প্রোগ্রামের শুরুতে যেমন `main()` ফাংশনের পূর্বে ঘোষণা করা হয় তখন তাকে গ্লোবাল ভেরিয়েবল বলা হয়। গ্লোবাল ভেরিয়েবলের মান ও অস্তিত্ব কোনো নির্দিষ্ট ব্লক বা কোনো নির্দিষ্ট ফাংশনের মধ্যে সীমাবদ্ধ না থেকে পুরো প্রোগ্রামে বিস্তৃত থাকে। এ ধরনের ভেরিয়েবল ফাংশনের মধ্যে নয়; ফাংশনের উপরে ঘোষণা করা হয়। ফলে ভেরিয়েবলের মান, নাম ও ক্ষেত্র প্রোগ্রামে ব্যবহৃত সকল ফাংশনের জন্য সমানভাবে প্রযোজ্য।

**লোকাল ভেরিয়েবল ও গ্লোবাল ভেরিয়েবল এর মধ্যে পার্থক্য:**

লোকাল ভেরিয়েবল	গ্লোবাল ভেরিয়েবল
১. কোনো ফাংশনের মধ্যে ভেরিয়েবল ডিক্লেয়ার করলে তাকে উক্ত ফাংশনের লোকাল ভেরিয়েবল বলা হয়।	১. সকল ফাংশনের বাহিরে প্রোগ্রামের শুরুতে ডিক্লেয়ার করা ভেরিয়েবলকে গ্লোবাল ভেরিয়েবল বলা হয়।
২. কোনো ফাংশনের মধ্যে ডিক্লেয়ার করা লোকাল ভেরিয়েবল উক্ত ফাংশনের বাইরে ব্যবহার করা যায় না।	২. গ্লোবাল ভেরিয়েবলের কর্মকান্ড কোনো ফাংশনের মধ্যে সীমাবদ্ধ নয়।
৩. ভিন্ন ভিন্ন ফাংশনে একই নামের লোকাল ভেরিয়েবল থাকতে পারে।	৩. একটি প্রোগ্রামের মধ্যে একই নামের একটি মাত্র গ্লোবাল ভেরিয়েবল থাকতে পারে।
৪. ফাংশনের শুরুতে ডিক্লেয়ার করা হয়।	৪. সাধারণত প্রোগ্রামের শুরুতে ডিক্লেয়ার করা হয়।

**ডেটার ধরনের ওপর ভিত্তি করে সি ভাষায় মোটামুটি ছয় ধরনের ভেরিয়েবল ব্যবহৃত হয়। যথা:**

- নিউমেরিক ভেরিয়েবল:** যে ভেরিয়েবলের মান সংখ্যায় হয় তাকে সংখ্যাসূচক চলক বা নিউমেরিক ভেরিয়েবল বলা হয়। এরূপ ভেরিয়েবলের মান প্রোগ্রামে নির্দিষ্ট করে দেয়া যায় অথবা প্রোগ্রাম নির্বাহের সময় কীবোর্ড বা অন্য কোনো উৎস থেকে নেয়া যায়। সি ভাষায় ব্যবহৃত নিউমেরিক ভেরিয়েবলগুলো পূর্ণসংখ্যা (যেমন, 100, 200, 300, 1000, -4890, 12345 ইত্যাদি), দশমিক চিহ্নবিশিষ্ট সংখ্যা (যেমন, 100.0, 20.50, 23.00, -48.90, 12.45 ইত্যাদি), কিংবা এক্সপোনেনশিয়াল বা দশমিক চিহ্নবিশিষ্ট বৃহৎ সংখ্যা (যেমন,  $3.4 \times 10^{200}$  অর্থাৎ 3.5E-203 ইত্যাদি) হতে পারে।
- অ্যারে ভেরিয়েবল:** একই ধরনের কতগুলো ভেরিয়েবলের সমষ্টিকে অ্যারে ভেরিয়েবল বলা হয়। অ্যারে আবার একমাত্রিক, দ্বিমাত্রিক ও বহুমাত্রিক হতে পারে। যেমন: একমাত্রিক অ্যারের উদাহরণ হলো: `A [10]` যা মোট 10টি ভেরিয়েবলের সমষ্টি নির্দেশ করে। আবার দ্বিমাত্রিক অ্যারের উদাহরণ হলো: `A [2], [3]` যা মোট ৬টি ভেরিয়েবলের সমষ্টি নির্দেশ করে।
- পয়েন্টার ভেরিয়েবল:** পয়েন্টার এক প্রকার ভেরিয়েবল যা একই টাইপের অপর কোনো ভেরিয়েবলকে নির্দেশ করে; অর্থাৎ একই টাইপের অপর কোনো ভেরিয়েবলের মেমোরি অ্যাড্রেস ধারণ করে। পয়েন্টার ভেরিয়েবল ব্যবহারের ফলে প্রোগ্রামের জটিলতা অনেকাংশে হ্রাস পায়। প্রোগ্রাম নির্বাহে অপেক্ষাকৃত কম সময় লাগে।
- কাস্টম ভেরিয়েবল:** অনেক সময় প্রোগ্রামের জটিলতা হ্রাস করার জন্য প্রোগ্রামার তার প্রয়োজনে বিভিন্ন টাইপের ভেরিয়েবলের সমন্বয়ে নিজস্ব ডেটা টাইপ তৈরি করে নেন। প্রোগ্রামার কর্তৃক তৈরি ডেটা টাইপকে ইউজার-ডিফাইন্ড বা কাস্টম ভেরিয়েবল বলা হয়। এতে প্রোগ্রামের জটিলতা অনেকাংশে হ্রাস পায়। স্ট্রিকচার, ইউনিয়ন ও ইনুমারেশন সি-তে বহুল ব্যবহৃত কয়েকটি কাস্টম ডেটা টাইপ।



- V. **স্ট্রিং ভেরিয়েবল:** যখন এক বা একাধিক ক্যারেক্টার বা বর্ণ একটি ভেরিয়েবল হিসেবে ব্যবহার করা হয় তখন তাকে স্ট্রিং ভেরিয়েবল বলা হয়। যেমন, "Programming in C", "University of Dhaka" ইত্যাদি। মূলত Char টাইপ ভেরিয়েবলকে অ্যারে কিংবা পয়েন্টার ভেরিয়েবল হিসেবে ঘোষণা করে তাতে স্ট্রিং সংরক্ষণ করা হয়। যেমন,

```
Char    Ch1 [30] =    "Programming in C"
Char    *Ch2 =        "University of Dhaka"
```

- VI. **বিল্ট ইন ভেরিয়েবল:** বিল্ট ইন ভেরিয়েবল হলো এমন একটি ভেরিয়েবল যা কতগুলো লাইব্রেরি ফাংশন। এটি ডেটা আইটেমের মান সংরক্ষণে ব্যবহৃত হয়।

কার্যকারিতার উপর নির্ভর করে ভেরিয়েবলকে চার ভাগে ভাগ করা যায়। যথা—

- **অটোমেটিক ভেরিয়েবল:** প্রোগ্রামে বিল্ট-ইন ও মডিফাইড ডেটা টাইপের যে সকল লোকাল ভেরিয়েবল ঘোষণা করা হয় সেগুলো বা ক্ষণস্থায়ী প্রকৃতির। এসব ভেরিয়েবলের জন্য প্রোগ্রাম নির্বাহকালে কম্পাইলার প্রয়োজনীয় মেমোরি পরিসর বরাদ্দ করে এবং ফাংশন নির্বাহ শেষে স্বয়ংক্রিয়ভাবে বরাদ্দকৃত মেমোরি পরিসর খালি করে দেয়। এরূপ ভেরিয়েবলকে অটোমেটিক ভেরিয়েবল বলা হয়। অটোমেটিক ভেরিয়েবল ঘোষণার জন্য ভেরিয়েবলের ডেটা টাইপের পূর্বে auto ব্যবহৃত হয়। তবে কোনো লোকাল ভেরিয়েবল ঘোষণাকালে তার ডেটা টাইপের পূর্বে auto কীওয়ার্ড উল্লেখ না করলেও কম্পাইলার স্বয়ংক্রিয়ভাবে অটোমেটিক ভেরিয়েবল হিসেবে গণ্য করে। অর্থাৎ সি প্রোগ্রামে যে সকল লোকাল ভেরিয়েবল ব্যবহার করা হয় সেগুলোর সবই অটোমেটিক ভেরিয়েবল।
- **স্ট্যাটিক ভেরিয়েবল:** নিজস্ব ফাংশন ও ব্যবহারকারী ফাংশনসহ পুরো প্রোগ্রামে কোনো ভেরিয়েবলের সর্বশেষ মান ব্যবহার করার জন্য স্ট্যাটিক ভেরিয়েবল ঘোষণা করা হয়। সি প্রোগ্রামে অটোমেটিক ভেরিয়েবল বিশিষ্ট কোনো ফাংশন কল করা হলে প্রতিবার ফাংশন কলের জন্য ভেরিয়েবলগুলোর প্রারম্ভিক মান গৃহীত হয়। কিন্তু স্ট্যাটিক ভেরিয়েবল বিশিষ্ট কোনো ফাংশন একাধিকবার কল করা হলে কেবল প্রথমবার স্ট্যাটিক ভেরিয়েবলের জন্য দেয় প্রারম্ভিক মান গৃহীত হয়। পরবর্তীতে যতবার তা কল করা হয় স্ট্যাটিক ভেরিয়েবলের জন্য ফাংশনে দেয়া প্রারম্ভিক মান গৃহীত না হয়ে পূর্ববর্তী ফাংশন কলে অর্জিত সর্বশেষ মান গৃহীত হয়। কোনো ভেরিয়েবলকে স্ট্যাটিক হিসেবে ঘোষণার জন্য ভেরিয়েবলের ডেটা টাইপের পূর্বে static কীওয়ার্ড ব্যবহৃত হয়।

অটোমেটিক ভেরিয়েবল ঘোষণার ফরম্যাট হলো :

```
void main()
{
    auto int x,y,z;
    //... ..
}
অথবা
void main()
{
    int x,y,z;
    //... ..
}
```

স্ট্যাটিক ভেরিয়েবল ঘোষণার ফরম্যাট হলো :

```
void main()
{
    static int x,y,z;
    //... ..
}
```

- **এক্সটার্নাল ভেরিয়েবল:** ইহা একটি গ্লোবাল ভেরিয়েবল, যার মান কোনো ফাংশন বা মডিউলের মাধ্যমে পরিবর্তন করা যায়। যে ফাংশনে তা পরিবর্তিত হয়, সেখানে তা extern হিসাবে ঘোষণা করা হয়। যেমন- extern int x;
- **রেজিস্টার ভেরিয়েবল:** এ সব চলকের মান মেমোরিতে না রেখে দ্রুতগতির রেজিস্টারে রাখা হয়। ফলে ডেটা প্রসেস সহজ হয়। এধরনের চলক ঘোষণা করতে register কী-ওয়ার্ড ব্যবহৃত হয়। যেমন - register int x;

**ভেরিয়েবল ঘোষণা ও নামকরণের নিয়মাবলী:** প্রোগ্রামার প্রোগ্রাম রচনা করতে প্রয়োজনীয় সংখ্যক ভেরিয়েবল ঘোষণা করেন এবং তিনি তার ইচ্ছা অনুযায়ী ভেরিয়েবলের নামকরণ করতে পারেন না। কারণ ভেরিয়েবল ঘোষণা ও নামকরণের মধ্যে কিছু মৌলিক সীমাবদ্ধতা ও নিয়ম-কানুন রয়েছে। প্রোগ্রামে ডেটা নিয়ে কাজ করার সময় প্রতিটি ডেটার জন্য একটি ভেরিয়েবল ব্যবহার করতে হয়। আবার প্রতিটি ভেরিয়েবল নামের পূর্বে তার ডেটা টাইপ উল্লেখ করতে হয়। ডেটা টাইপ-সহ কোনো ভেরিয়েবলের নামকরণ প্রক্রিয়াকে ভেরিয়েবল ঘোষণা বলা হয়।

প্রোগ্রামে ভেরিয়েবল ঘোষণা ও নামকরণের জন্য যেসব নিয়ম-নীতি অনুসরণ করতে হয় তা নিম্নরূপ:

- ভেরিয়েবলের প্রথম অক্ষর অবশ্যই আলফাবেটিক ক্যারেক্টার (a, ...., z, A, ....., Z) হতে হবে। ভেরিয়েবল নাম ডিজিট বা অংক দিয়ে শুরু হতে পারে না। যেমন- Roll\_1 ও Roll\_10 বৈধ ভেরিয়েবল; কিন্তু 1Roll 2\_Roll অবৈধ।
- ভেরিয়েবলের মধ্যে স্পেশাল ক্যারেক্টার আন্ডারস্কোর ( ) ব্যবহার করা যায়। অন্য কোনো স্পেশাল ক্যারেক্টার (যেমন \$, !, @, #, %, \*, +, - ইত্যাদি) ব্যবহার করা যায় না। যেমন, my\_var বৈধ ভেরিয়েবল; কিন্তু my@var ও my&Roll অবৈধ।
- একই ফাংশনে একই নামে দুই বা ততোধিক ভেরিয়েবল ঘোষণা করা যায় না। তবে একই প্রোগ্রামে ব্যবহৃত দুই বা ততোধিক ফাংশনে একই নামে কোনো ভেরিয়েবল ঘোষণা করা যেতে পারে।
- ভেরিয়েবল নামের মধ্যে কোনো ফাঁকা জায়গা থাকতে পারে না। যেমন, RollNo, Roll, MyRoll ইত্যাদি বৈধ ভেরিয়েবল। কিন্তু Roll N ও Roll 1, My Roll অবৈধ।
- সি প্রোগ্রামে বড় ও ছোট হাতের অক্ষরগুলো আলাদা অর্থ বহন করে। তাই Roll\_1, roll\_10 ও MyRoll নামে ভেরিয়েবল ঘোষণা করে roll\_1, Roll\_10, Myroll নামে ব্যবহার করা যায় না।
- কোনো কীওয়ার্ডের নাম ভেরিয়েবল হিসেবে ব্যবহার করা যায় না। main কোনো কীওয়ার্ড না হলেও ভেরিয়েবল নাম হিসেবে main ব্যবহৃত হয় না। অবশ্য কীওয়ার্ড-সমূহের নামের এক বা একাধিক বর্ণ বড় হরফে লিখে আইডেন্টিফায়ারের নাম হিসেবে ব্যবহার করা যায়। তবে এরূপ না করাই উত্তম। যেমন, Int, Char, Main ইত্যাদি বৈধ ভেরিয়েবল। কিন্তু int, main ইত্যাদি অবৈধ।
- ভেরিয়েবল নামকরণে যেকোনো সংখ্যক ক্যারেক্টার ব্যবহার করা যায়। তবে ANSI নিয়ম অনুযায়ী ভেরিয়েবল নামকরণে ৩১টি ক্যারেক্টারের বেশি ব্যবহার না করাই ভালো।

**ভেরিয়েবলের মান নির্ধারণের ক্ষেত্রে সাধারণ ভুল:** মান নির্ধারণের ক্ষেত্রে সাধারণত ভেরিয়েবলের ডেটা টাইপ ও ডেটা টাইপের রেঞ্জের ভুল বেশি হয়। যেমন, এক টাইপ ভেরিয়েবলের জন্য অন্য টাইপ মান দেওয়া কিংবা ভেরিয়েবলের মানের রেঞ্জ অতিক্রম করা। যেকোনো কারণে ভেরিয়েবলের মানের রেঞ্জ অতিক্রম করলে প্রোগ্রামে ভুল ফলাফল আসতে পারে। তবে মজার ব্যাপার হলো এক্ষেত্রে কম্পাইলার কোনো সতর্ক বার্তা দেখায় না।

**সি ল্যাংগুয়েজে কনস্ট্যান্ট ও ভেরিয়েবল এর মধ্যে পার্থক্য**

কনস্ট্যান্ট	ভেরিয়েবল
১. কনস্ট্যান্ট অর্থ স্থির বা ধ্রুবক যা একটি নির্দিষ্ট মান ধারণ করে।	১. ভেরিয়েবল হলো একটা নাম, যে নামে কম্পাইলার নির্দিষ্ট ধরনের ডেটা রাখার জন্য মেমোরিতে জায়গা রাখে।
২. কনস্ট্যান্টে কমা ব্যবহার করা যায় না। তবে প্রয়োজনে দশমিক ব্যবহার করা যায়।	২. ভেরিয়েবলের মান নির্ধারণ করার সময় সংখ্যার মধ্যে কমা ব্যবহার করা যাবে।
৩. প্রোগ্রাম চালানোর সময় কোনোভাবেই কনস্ট্যান্ট এর মান পরিবর্তন করা যায় না।	৩. প্রোগ্রাম চালানোর সময় যখন প্রয়োজন ইচ্ছামত ভেরিয়েবল এর মান পরিবর্তন করা যায়।



**কাজ:**

- নিচের চলকগুলোর মধ্যে কোনগুলো বৈধ এবং বৈধ নয়। কারণ দর্শাও।  
num, num1, lnum, n1um, n um, n \_um, \_num, !num, n!um, ..num, n..um, \_n\_
- নিচের চলকগুলো কেন অবৈধ তা ব্যাখ্যা করো।  
i. int my@roll ii. int "5x" iii. int main iv. float marks 50 v. char fa-name;



## ৫.১২.১১ রাশিমালা (Expression)

সি ভাষায় গাণিতিক ও যৌক্তিক কাজ নিয়ন্ত্রণ করার জন্য কতগুলো বিশেষ সিম্বল (যেমন, +, -, \*, /, ++, --, <, >, >= ইত্যাদি) ব্যবহৃত হয়, এগুলোকে অপারেটর বলা হয়। আর যা ডেটা ধারণ করে তাকে অপারেণ্ড বলা হয়। অপারেণ্ড বা ডেটা ব্যবহার করে বিভিন্ন কর্ম সম্পাদনের জন্য অপারেটর ব্যবহৃত হয়। কতগুলো অপারেণ্ড, অপারেটর ও কনস্ট্যান্টের অর্থবোধক এবং সামঞ্জস্যপূর্ণ উপস্থাপনকে এক্সপ্রেশন বা বর্ণনা বলা হয়।

উদাহরণ হিসেবে বলা যায়,  $Average = (value1 + value2) / 2$ ; একটি এক্সপ্রেশন। এখানে Average, value1, value2 অপারেণ্ড; =, -, +, / অপারেটর ও 2 কনস্ট্যান্ট।

সি-তে ব্যবহৃত অপারেটরসমূহ: অপারেটরের সাথে সংযুক্ত অপারেণ্ড বা কনস্ট্যান্ট সংখ্যার ভিত্তিতে সি প্রোগ্রামে ব্যবহৃত অপারেটর সমূহকে তিনটি প্রধান শ্রেণিতে ভাগ করা হয়। যথা:

- ইউনারি অপারেটর
- বাইনারি অপারেটর এবং
- টারনারি অপারেটর

ইউনারি অপারেটর: যে সকল অপারেটরের সাথে কেবল একটি অপারেণ্ড বা কনস্ট্যান্ট সংযুক্ত থাকে তাদেরকে ইউনারি অপারেটর বলা হয়। নিম্নে একটি ছকে বহুল ব্যবহৃত কয়েকটি ইউনারি অপারেটর ও তাদের ব্যবহার উল্লেখ করা হলো।

অপারেটর	উদাহরণ	ব্যবহার
+	$v2 = v1;$ $v3 = (v1 - v2);$	অপারেণ্ডের ধনাত্মক মান বুঝাতে ব্যবহৃত হয়।
-	$v2 = -v1;$ $v3 = -(v1 + v2);$	অপারেণ্ডে ঋণাত্মক মান বুঝাতে ব্যবহৃত হয়।
++	++x; বা x++; যা x = x+1 বা x += 1 এর সমান।	অপারেণ্ডের মানের সাথে ১ যোগ হয়।
--	--x; বা x--; যা x = x-1 বা x -= 1 এর সমান।	অপারেণ্ডের মান হতে ১ বিয়োগ হয়।
!	!4 = 0 এবং !0 = 1	শূণ্য বাদে অন্য কোনো অপারেণ্ডের মান 0 করে দেয় কিন্তু শূণ্যের মান 1 করে দেয়।
~	$A = (10)_{10} = (00001010)_2$ $\sim A = (11110101)_2$ বা $(-11)_{10}$	অপারেণ্ডের মান ১'এর পরিপূরকে রূপান্তর করে তা আবার দশমিকে রূপান্তর করে দেখায়।

ইউনারি অপারেটরগুলো পোস্টফিক্স বা প্রিফিক্স নোটেশনে কাজ করে। পোস্টফিক্স নোটেশন অর্থ হলো অপারেটর অপারেণ্ডের পরে বসে। অন্যদিকে প্রিফিক্স নোটেশন অর্থ হলো অপারেটর অপারেণ্ডের পূর্বে বসে। কোন কোন অপারেটরগুলো পোস্টফিক্স, প্রিফিক্স বা উভয় নোটেশনে কাজ করে তা নিচে দেওয়া হলো।

অপারেটর	নোটেশন	অপারেটর	নোটেশন	অপারেটর	নোটেশন
+	প্রিফিক্স	++	উভয়	!	প্রিফিক্স
-	প্রিফিক্স	--	উভয়	~	প্রিফিক্স

**বাইনারি অপারেটর:** যে সকল অপারেটরের সাথে দুইটি অপারেণ্ড বা কন্সট্যান্ট সংযুক্ত থাকে তাদেরকে বাইনারি অপারেটর বলা হয়। যেমন: সি প্রোগ্রামে ইউনারি অপারেটর অপেক্ষা বাইনারি অপারেটরের ব্যবহার বেশি দেখা যায়। ইউনারি ও টারনারি অপারেটর ছাড়া বাকী সবগুলো বাইনারি অপারেটর। বাইনারি অপারেটর ইনফিক্স (infix) নোটেশনে কাজ করে। অর্থাৎ অপারেটরগুলো দুটো অপারেণ্ডের মাঝখানে ব্যবহৃত হয়।

**টারনারি অপারেটর:** যে সকল অপারেটর এক সাথে তিনটি অপারেণ্ড নিয়ে কাজ করে তাদেরকে টারনারি অপারেটর বলা হয়। টারনারি অপারেটরটি হলো-  $?:$  যা if-else স্টেটমেন্টের সংক্ষিপ্ত রূপ হিসাবে কাজ করে। এই অপারেটরটি ব্যবহারের সিনটেক্স হলো-

(condition) ? true result : false result;

এখানে condition সত্য হলে প্রোগ্রামে true result অংশ কাজ করবে অন্যথায় false result অংশ কাজ করবে।

যেমন:  $x = (a > b) ? a : b$ , এখানে a, b এর চেয়ে বড় হলে  $x = a$  হবে, নতুবা  $x = b$  হবে। এই উদাহরণে  $a = 10$ ;  $b = 20$  হলে x এর মান হবে 20।

কাজের উপর ভিত্তি করে বাইনারি অপারেটরকে আবার নিম্নোক্ত ভাগে ভাগ করা যায়।

**১. গাণিতিক অপারেটর (Arithmetic Operator):** ‘সি’ প্রোগ্রামে গাণিতিক কাজ যেমন-যোগ, বিয়োগ, গুণ, ভাগ ইত্যাদি সম্পন্ন করার জন্য অ্যারিথমেটিক অপারেটর ব্যবহৃত হয়।

গাণিতিক অপারেটর	উদাহরণ	গাণিতিক অপারেটর	উদাহরণ
+	যোগ করার জন্য $7 + 5 = 12$	/	ভাগ করার জন্য $7/5 = 1$
-	বিয়োগ করার জন্য $7 - 5 = 2$	%	ভাগশেষ নির্ণয়ের জন্য $7\%5 = 2$
×	গুণ করার জন্য $7 \times 5 = 35$		

এখানে + এবং- ছাড়া বাকীগুলো শুধুমাত্র বাইনারি অপারেটর হিসাবে কাজ করে। অর্থাৎ, এটি দু’টো অপারেণ্ড নিয়ে কাজ করে। + ও - কে ইউনারি কিংবা বাইনারি অপারেটর হিসেবেও ব্যবহার করা যায়।

প্রোগ্রামে ভাগের কাজ করার জন্য ( / ) অপারেটর ব্যবহার করা হয়। ( / ) অপারেটর দিয়ে এক্সপ্রেশন তৈরি করার সময় নিম্নের বিষয়গুলো খেয়াল রাখতে হয়,

- ১) দ্বিতীয় অপারেণ্ডেও মান অবশ্যই শূন্য হতে পারেনা।
- ২) উভয় অপারেণ্ডের ডেটাইপ যদি int হয়, তাহলে ভাগফল ও int টাইপের হয়।

এখানে % অপারেটর ছাড়া অন্য অপারেটর গুলোর অপারেণ্ড হিসেবে যেকোনো ডেটাইপ (int, float, double, char) ব্যবহার করা যায়। কিন্তু % অপারেটরের অপারেণ্ড হিসাবে অবশ্যই int টাইপের ডেটাইপ ব্যবহার করতে হয়।

উদাহরণ: যদি $a=10$ , $b=20$ , $c=7$ এবং $d=2$ হয়, তাহলে $a+b*c/d$ এক্সপ্রেশনের মান- $a+b*c/d$ $=10+20*7/2$ $=10+140/2$ $=10+70$ $=80$	উদাহরণ: যদি $a=10$ , $b=20$ , $c=30$ এবং $d=5$ হয়, তাহলে $a+(a*b-(c\%d)/2)*3-(c-d+(a-b))$ এক্সপ্রেশনের মান $a+(a*b-(c\%d)/2)*3-(c-d+(a-b))$ $=10+(10*20-(30\%5)/2)*3-(30-5+(10-20))$ $=10+(200-0/2)*3-(30-5-10)$ $=10+(200-0)*3-(25-10)$ $=10+200*3-15 = 10+600-15$ $=610-15$ $=595$
---	--

**কাজ:**



নিচের রাশিমালাগুলোর ফলাফল বের করো।

- $\text{pow}((6/2+4/2), (13\%5))-4*2+7$
- $\text{pow}(15/3-9/3), (17\%5))-4+3*2+7$
- $49/7-\text{pow}((18/3-3/3), (14\%4))-5+2*2+29$

**২. অ্যাসাইনমেন্ট অপারেটর (Assignment Operator):** সি প্রোগ্রামে ভেরিয়েবল বা এক্সপ্রেশনের মান অন্য কোনো ভেরিয়েবল বা এক্সপ্রেশনের মান হিসেবে ব্যবহার করতে অ্যাসাইনমেন্ট অপারেটর ব্যবহৃত হয়।

অপারেটর	উদাহরণ (int a=11,b=5)	একই রকম ব্যবহার	ফলাফল	অপারেটর	উদাহরণ (int a=11, b=5)	একই রকম ব্যবহার	ফলাফল
=	a = b	a = b	a = 5	* =	a *= b	a = a * b	a = 55
+ =	a += b	a = a + b	a = 16	/ =	a /= b	a = a / b	a = 2
- =	a -= b	a = a - b	a = 6	% =	a %= b	a = a % b	a = 1



**কাজ:**

```
int a=2,b=3,c=4;
a=a+b+c;
b=a+b+c;
c=a+b+c;
প্রশ্ন: a, b, c এর সর্বশেষ মান নির্ণয় কর।
```

**৩. রিলেশনাল অপারেটর (Relational Operator):** সি প্রোগ্রামিং এ রিলেশনাল অপারেটর দুটি অপারেন্ডের মধ্যে সম্পর্ক যাচাই করে। রিলেশন সত্যি (True) হলে 1 রিটার্ন করে, রিলেশন মিথ্যা (false) হলে 0 রিটার্ন করে। সিদ্ধান্ত গ্রহণ (Decision making) এবং লুপ (Loop) এ রিলেশনাল অপারেটর ব্যবহৃত হয়।

অপারেটর	অর্থ	উদাহরণ (int a=11, b=5)	ফলাফল
< (ছোট)	ডানদিকের অপারেন্ডের চেয়ে বামদিকের অপারেন্ড ছোট কিনা তা যাচাইয়ের জন্য।	a < b	False
<= (ছোট বা সমান)	ডানদিকের অপারেন্ডের চেয়ে বামদিকের অপারেন্ড ছোট বা সমান কিনা তা যাচাইয়ের জন্য।	a <= b	False
> (বড়)	ডানদিকের অপারেন্ডের চেয়ে বামদিকের অপারেন্ড বড় কিনা তা যাচাইয়ের জন্য।	a > b	True
>= (বড় বা সমান)	ডানদিকের অপারেন্ডের চেয়ে বামদিকের অপারেন্ড বড় বা সমান কিনা তা যাচাইয়ের জন্য।	a >= b	True
== (সমান)	ডানদিকের অপারেন্ড এবং বামদিকের অপারেন্ড সমান কিনা তা যাচাইয়ের জন্য।	a == b	False
!= (অসমান)	ডানদিকের অপারেন্ড এবং বামদিকের অপারেন্ড অসমান কিনা তা যাচাইয়ের জন্য।	a != b	True

অ্যাসাইনমেন্ট অপারেটর এর '=' এবং রিলেশনাল অপারেটর হিসাবে ব্যবহৃত সমতা চিহ্ন '==' এক নয়। অ্যাসাইনমেন্ট অপারেটর '=' কে কোন একটা ভেরিয়েবলের মান নির্ধারণ করতে ব্যবহৃত হয়। কিন্তু সমতা চিহ্ন '==' কে দুটি অপারেন্ড সমান কিনা তা তুলনা করতে ব্যবহার করা হয় তা অপারেন্ড হোক বা কোনো এক্সপ্রেশনই হোক।



**কাজ:**

x=5, y=6, z=7। তাহলে নিম্নের এক্সপ্রেশন গুলোর আউটপুট নির্ণয় কর।

i. (x+y)>z	ii. (x+y)<=z	iii. x!=y	iv. (x<y)&&(y==6)
v. (x<y)&&(z!=y)	vi. (x>y)   (z!=y)	vii. (x>y)&&(z!=y)	viii. (x<y)&&(z==y)
ix. (x<y)   (z==y)			

**৪. লজিক্যাল অপারেটর (Logical Operator):** সি প্রোগ্রামে বিভিন্ন ধরনের লজিক্যাল অপারেশন (যেমন- অর, অ্যান্ড, নট) সম্পন্ন করার জন্য লজিক্যাল অপারেটর ব্যবহার করা হয়। লজিক্যাল অর ও অ্যান্ড বাইনারি অপারেটর হলেও লজিক্যাল নট ইউনারি অপারেটর। অপারেটর গুলো int টাইপের ডেটা নিয়ে কাজ করে। লজিক্যাল এক্সপ্রেশনে ব্যবহৃত কোনো অপারেন্ড বা এক্সপ্রেশনের মানশূন্য ব্যতিত অন্য যে কোনো সংখ্যা হলে তার মান True বা এক(1) ধরা হয়, অন্যথায় False বা শূন্য (0) ধরা হয়। সি প্রোগ্রামিং-এ && (অ্যান্ড), || (অর) এবং !(নট) অপারেটর সমূহকে লজিক্যাল অপারেটর বলা হয়। সি প্রোগ্রামিং এ সিদ্ধান্ত গ্রহণে সচারচর লজিক্যাল অপারেটর ব্যবহৃত হয়। নিচে ছকের মাধ্যমে কয়েকটি লজিক্যাল অপারেশনের ফলাফল দেখানো হলো।

অপারেটর	অর্থ	উদাহরণ (a=11,b=5,c=15)	ফলাফল
&& ( লজিক্যাল অ্যান্ড )	উভয় অপারেন্ড True হলে ফলাফল True হবে।	((a==b)&&(b>c))	True বা 1
 ( লজিক্যাল অর )	যেকোনো একটি অপারেন্ড True হলে ফলাফল True হবে।	(b==c)    (a>c)	False বা 0
! ( লজিক্যাল নট )	অপারেন্ড False হলে ফলাফল True হবে।	!(b==c)	True বা 1



#### কাজ:

মনেকরি, a = 5, b = 5, c = 10। তাহলে নিম্নের এক্সপ্রেশন গুলোর আউটপুট নির্ণয় কর।

i. (a = b) && (c > b);	ii. (a = b) && (c < b);	iii. (a = b)    (c < b);
iv. (a != b)    (c < b);	v. !(a != b);	vi. !(a == b);

**৫. ইনক্রিমেন্টাল/ ডিক্রিমেন্টাল অপারেটর (Incremental/Decremental Operator):** ++ এবং -- কে যথাক্রমে ইনক্রিমেন্টাল ও ডিক্রিমেন্টাল অপারেটর বলা হয়। কোনো অপারেন্ডের মান 1 বৃদ্ধি বা হ্রাস করতে যথাক্রমে ইনক্রিমেন্টাল ও ডিক্রিমেন্টাল ব্যবহৃত হয়। যেমন, count++ এবং count--। এখানে Counter একটি int টাইপ ভেরিয়েবল। count++ স্টেটমেন্টের মাধ্যমে count = count + 1; এবং Counter-- স্টেটমেন্টের মাধ্যমে count = count-1; বোঝানো হয়। সাধারণত for এবং while লুপে ইনক্রিমেন্টাল এবং ডিক্রিমেন্টাল অপারেটর বেশি ব্যবহৃত হয়।

**ইনক্রিমেন্টাল ও ডিক্রিমেন্টাল অপারেটরের প্রিফিক্স ও পোস্টফিক্স নোটেশন:** ইনক্রিমেন্টাল বা ডিক্রিমেন্টাল অপারেটরের প্রিফিক্স (++count বা --count) নোটেশনের ক্ষেত্রে কম্পাইলার প্রথমে ভেরিয়েবলের প্রারম্ভিক মানের সাথে যথাক্রমে এক যোগ বা বিয়োগ করে, অতপর প্রোগ্রামের একই স্টেটমেন্ট এই বর্ধিত মান ব্যবহার করে। কিন্তু ইনক্রিমেন্টাল বা ডিক্রিমেন্টাল অপারেটরের পোস্টফিক্স (count++ বা count--) নোটেশনের ক্ষেত্রে কম্পাইলার প্রথমে প্রোগ্রামে ভেরিয়েবলের পুরাতন মান ব্যবহার করে, অতপর ভেরিয়েবলের মানের সাথে যথাক্রমে এক যোগ বা বিয়োগ করে। এই নতুন মান পরবর্তী স্টেটমেন্ট ধাপ থেকে কার্যকর হয়। অর্থাৎ কোনো চলকের আগে ++ বা -- ব্যবহার করলে আগে চলকের মান বৃদ্ধি বা হ্রাস পাবে এবং পরে চলক অপারেশনে অংশ নিবে। পক্ষান্তরে কোনো চলকের পরে ++ বা -- ব্যবহার করলে আগে চলক অপারেশনে অংশ নিবে এবং পরে চলকের মান বৃদ্ধি বা হ্রাস পাবে। যেমন :

<b>উদাহরণ :</b> <pre>main () { int x, y; y = 10; x = ++y; }</pre> <p>এক্ষেত্রে আগে y এর মান এক 1(এক) বৃদ্ধি পাবে। তাহলে y এর মান হলো 11; তারপর x = 11 হবে।</p>	<b>উদাহরণ :</b> <pre>main () { int x, y; y = 10; x = y++; }</pre> <p>এক্ষেত্রে প্রথমে x = 10 হবে, তারপর y এর মান 1(এক) বৃদ্ধি পেয়ে y = 11 হবে।</p>
---	--



**কাজ:** প্রোগ্রাম শেষে চলক গুলি যে মান ধারণ করবে তা দেখাও।

**কাজ ১:**  

```
main()
{
int a, b, s, d, m, mod,
di;
a=35;
b=8;
s=a+b;
d=a-b;
m=a*b;
mod=a%b;
di=a/b;
a++;
++a;
b--;
--b;
}
```

**কাজ ২:**  

```
কাজ ৪:
main()
{
int a,b,c;
a=5;
b=10+a;
b++;
++b;
c=b-a;
printf("c=%d",c);
}
```

**কাজ ৩:**  

```
main()
{
int a=21,b=8;
if(a>b)
b++;
if(b==9)
--a;
if(a<21&&b>=9)
a=10;
if(a!=10)
b=10;
}
```

**৬. বিটওয়াইজ অপারেটর (Bitwise Operator):** সি প্রোগ্রামের কার্যক্ষমতা বাড়াতে অনেক সময় বাইনারি ডেটা বা বিট নিয়ে কাজ করার জন্য যে সকল অপারেটর ব্যবহার করা হয় তাদেরকে বিটওয়াইজ অপারেটর বলে। বিটওয়াইজ অপারেটর কেবলমাত্র int টাইপের ডেটা নিয়ে কাজ করে।

নিচে ছকের মাধ্যমে বিটওয়াইজ অপারেটর সমূহের তালিকা ও ব্যবহার উল্লেখ করা হলো।

বিটওয়াইজ অপারেটর	বর্ণনা	উদাহরণ
& (বিটওয়াইজ AND)	দুইটি অপারেন্ডেও বিট সমূহের মধ্যে জোড়ায় জোড়ায় অ্যান্ড (AND) অপারেশন সম্পন্ন করে।	$A=(4)_{10} \Rightarrow A=(00000100)_2$ $B=(5)_{10} \Rightarrow B=(00000101)_2$ $A \& B=(00000100)_2$ বা $4_{10}$
 (বিটওয়াইজ OR)	দুইটি অপারেন্ডেও বিট সমূহের মধ্যে জোড়ায় জোড়ায় অর (OR) অপারেশন সম্পন্ন করে।	$A=(4)_{10} \Rightarrow A=(00000100)_2$ $B=(5)_{10} \Rightarrow B=(00000101)_2$ $A   B=(00000101)_2$ বা $5_{10}$
^ (বিটওয়াইজ XOR)	দুইটি অপারেন্ডের বিট সমূহের মধ্যে জোড়ায় জোড়ায় এক্সঅর অপারেশন সম্পন্ন করে।	$A=(4)_{10} \Rightarrow A=(00000100)_2$ $B=(5)_{10} \Rightarrow B=(00000101)_2$ $A \wedge B=(00000001)_2$

বিটওয়াইজ অপারেটর	বর্ণনা	উদাহরণ
<< (শিফট লেফট)	কোনো অপারেন্ডের বাইনারি বিট সমূহকে এক বা একাধিক বার বামদিকে সরানোর জন্য ব্যবহৃত হয়। ফলে প্রতিবার লেফটশিফট অপারেশনে কোনো সংখ্যার মান দ্বিগুণ হয়।	$A=(4)_{10} \Rightarrow A=(0000\ 0100)_2$ $A<<2=(0001\ 0000)_2$ $= (16)_{10}$
>> (শিফটরাইট)	কোনো অপারেন্ডেও বাইনারি বিট সমূহকে এক বা একাধিকবার ডানদিকে সরানোর জন্য ব্যবহৃত হয়। ফলে রাইটশিফট অপারেশনে কোনো সংখ্যার মান অর্ধেক হয়।	$A=(4)_{10} \Rightarrow A=(00000100)_2$ $A>>2=(00000001)_2$ $= (1)_{10}$
~ (১'এর পরিপূরককরেতা দশমিকে দেখায়)	কোনো অপারেন্ডেও বাইনারি বিট সমূহকে বিপরীত করে অর্থাৎ ০ থাকলে ১ এবং ১ থাকলে ০ করে ফেলে।	$A=(4)_{10} \Rightarrow A=(00000100)_2$ $\sim A=11111011$ [১'এর পরিপূরক] $=(-5)_{10}$ $\sim A+1=11111011$ [২'এর পরিপূরক] $=(-4)_{10}$



কাজ:  $A=10, B=12$  হলো নিম্নোক্ত রাশি সমূহের মান বের কর:  
 $A|B, A\&B, A\wedge B, A<<2, B>>2, \sim A, \sim A+1, \sim B, \sim B+1$

**৭. কন্ডিশনাল অপারেটর (Conditional Operator):** সি প্রোগ্রামে শর্ত সাপেক্ষে কোনো ভেরিয়েবলের মান অন্য কোনো ভেরিয়েবলের মান হিসাবে নির্ধারণ করার জন্য যে সকল অপারেটর ব্যবহার করা হয় তাকে কন্ডিশনাল অপারেটর বলে। অর্থাৎ এক জোড়া অপারেটর “? :” সি প্রোগ্রামে কন্ডিশনাল এক্সপ্রেশন গঠন করার জন্য ব্যবহার করা হয়। এই অপারেটরের গঠন নিম্নরূপ:

$exp1\ ?\ exp2\ :\ exp3$ এখানে $exp1, exp2$ , এবং $exp3$ হচ্ছে expression।	<b>উদাহরণ :</b> $x = 20, y = 15;$ $z = (x > y) ? x : y;$ এখানে, $(x > y)$ যদি সত্য হয় তাহলে $z = x$ , অন্যথায় $z = y$ হবে। শর্ত অনুযায়ী $z$ এর মান হবে ২০।
---	---

**৮. বিশেষ অপারেটর (Misc Operator):** C প্রোগ্রামে কিছু বিশেষ অপারেটর ব্যবহার করা হয় বিশেষ বিশেষ কাজের জন্য। যেমন—The comma operator (,), The size of operator (sizeof), Pointer operator (& and \*), Member selection operator (. and ->) ইত্যাদি। সি প্রোগ্রামে বিশেষ কাজের জন্য বিশেষ অপারেটর ব্যবহৃত হয়।

অপারেটর	বর্ণনা	উদাহরণ
sizeof()	কোনো চলকের আকার(Size) কী তা জানার জন্য ব্যবহার করা হয়।	sizeof(char)
&	চলকের ঠিকানা জানার জন্য ব্যবহার করা হয়।	&a
,	একই ধরনের একাধিক এক্সপ্রেশনকে সংযুক্ত বা আলাদা করতে কমা (,) অপারেটর ব্যবহার করা হয়।	int a,b,c; int a=4, b=9,c=5;

নিচে কিছু গাণিতিক এক্সপ্রেশন সমতুল্য সি ভাষায় টেবিলের মাধ্যমে দেখানো হলো:

গাণিতিক এক্সপ্রেশন	সমতুল্য সি ভাষায় এক্সপ্রেশন	গাণিতিক এক্সপ্রেশন	সমতুল্য সি ভাষায় এক্সপ্রেশন
$y=a+\frac{b}{c}+d$	$y=a+(b/c)+d$	$a=x^2+2xy+y^2$	$a=x*x+2*x*y+y*y$
$y=\frac{a+b}{b+c}$	$y=(a+b)/(c+d)$	$x=ay^2+by+cz^3$	$x=a*y*y+b*y+c*z*z*z$
$y=a^3+b^3$	$y=a*a*a+b*b*b$	$y=\sqrt{9b^2+4ac^2}$	$y=\text{sqrt}(9*b*b+4*a*c*c)$
$y=\sqrt{b^2-4ac}$	$y=\text{sqrt}(b*b-4*a*c)$		
$y=\left(\frac{a+b}{c+d}\right)^3$	$y=\text{pow}(((a+b)/(c+d)),3)$	$y=a+\frac{x}{ m-n }$	$y=a+x/\text{abs}(m-n)$
$y=a+\frac{1}{1+\frac{1}{a+b}}$	$y=a+(1/(1+1/(a+b)))$	$y=abc+\frac{a}{ab+bc}$	$y=a*b*c+a/(a*b+b*c)$
$y=(a^n)^m+d^nd^m$	$y=\text{pow}(\text{pow}(a,n),m)+\text{pow}(a,n)*\text{pow}(a,m)$	$y=\frac{a}{ab+\frac{de}{g}+b}$	$y=a/(a*b+(d*e/g)+b)$
$y=a+\frac{\frac{b}{c}}{\sqrt{m-n^2}}$	$y=a+(b/c)/\text{sqrt}(m-n*n)$	$y=\frac{ a-b }{ c-d }+\frac{x}{ m-n }$	$y=\text{abs}(a-b)/\text{abs}(c-d)+x/\text{abs}(m-n)$

### ৫.১২.১২ কিওয়ার্ড (Keyword)

প্রত্যেক প্রোগ্রামিং ভাষার নিজস্ব কিছু সংরক্ষিত শব্দ আছে যা প্রোগ্রাম রচনার সময় ব্যবহার করা হয়। এই সংরক্ষিত শব্দগুলোকে কিওয়ার্ড বলা হয়। C প্রোগ্রামে ৩২টি সংরক্ষিত শব্দ আছে যা স্টেটমেন্ট নামে পরিচিত। এদের প্রতিটির আলাদা অর্থ আছে এবং এদেরকে প্রোগ্রামে লেখার সময় ছোট হাতের অক্ষরে লিখতে হয়। সি প্রোগ্রামিং-এ বিভিন্ন ভার্শনে বিভিন্ন ধরনের কিওয়ার্ড ব্যবহৃত হয়। নিম্নের টেবিলে ভার্শন 'সি'৮৯ প্রোগ্রামে ব্যবহৃত কিওয়ার্ডগুলো দেখানো হলো:

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

ভার্শন 'সি'৯৯ এ আরোও পাঁচটি কিওয়ার্ড যুক্ত হয়। যথা—



<code>_bool</code>	<code>_imaginary</code>	<code>restrict</code>	<code>_complex</code>	<code>inline</code>
--------------------	-------------------------	-----------------------	-----------------------	---------------------

সি১১ তে যুক্ত হয় আরোও সাতটি কিওয়ার্ড।

<code>_alignas</code>	<code>_atomic</code>	<code>_noreturn</code>	<code>_thread_local</code>
<code>_alignof</code>	<code>_generic</code>	<code>_static_assert</code>	

এই সমস্ত কিওয়ার্ড ছাড়া কম্পাইলারের উপর ভিত্তি করে সি প্রোগ্রামিং ল্যাংগুয়েজ আরোও কিছু কিওয়ার্ড সমর্থন করে। ANSI সিতে ৪৭টি এবং C++ এর ৬৩টি কিওয়ার্ড আছে।

**কীওয়ার্ড ব্যবহারের নিয়ম :** কীওয়ার্ডসমূহ ব্যবহারের জন্য সুনির্দিষ্ট নিয়ম আছে। এর সামান্য ব্যতিক্রম হলে প্রোগ্রাম ভুল ফলাফল দিতে পারে। নিম্নে কীওয়ার্ড ব্যবহারের কয়েকটি নিয়ম উল্লেখ করা হলো।

- কীওয়ার্ডসমূহের নাম একটি একক শব্দ বা ওয়ার্ডে হয়, অর্থাৎ মাঝে কোনো ফাঁকা স্থান থাকে না।
- কীওয়ার্ডসমূহের প্রতিটি বর্ণ ছোট হাতের হয়, অর্থাৎ কীওয়ার্ডের নাম লিখতে ইংরেজি বড় হাতের অক্ষর ব্যবহার করা যায় না।
- কখনও যদি দুটো কীওয়ার্ড একত্রে ব্যবহৃত হয় তবে মাঝে ফাঁকা স্থান থাকে।



**কাজ:**

নিচের সমীকরণগুলোকে C ভাষায় লেখ।

১.  $y = \sqrt{b^2 - 4ac}$

২.  $ax^2 + bx + c = 0$

৩.  $\frac{ax^2}{bx} + c = 0$

## ৫.১২.১৩ সি ভাষায় ইনপুট-আউটপুট স্টেটমেন্ট (Input-Output Statement in C language)

সি ভাষাতে অনেক ধরনের লাইব্রেরি ফাংশন আছে। তার মধ্যে অন্যতম হলো Standard I/O লাইব্রেরি ফাংশন যা সকল ধরনের ইনপুট ও আউটপুটের কাজ সম্পন্ন করে।

**ইনপুট স্টেটমেন্টঃ** যে সকল স্টেটমেন্টের সাহায্যে সি প্রোগ্রামে ডাটা নেয়া বা ইনপুট করা হয় তাদেরকে বলা হয় ইনপুট স্টেটমেন্ট। যেমন- `getchar()`, `gets()`, `scanf()` ইত্যাদি।

**আউটপুট স্টেটমেন্টঃ** যে সকল স্টেটমেন্টের সাহায্যে প্রোগ্রামের ফলাফল মনিটরের পর্দায় প্রদর্শন করা হয় তাকে আউটপুট স্টেটমেন্ট বলে। যেমন- `putchar()`, `puts()`, `printf()` ইত্যাদি।

ইনপুট/ আউটপুট ফাংশন  
(Input/ Output Function)

ফরমেটেড ফাংশন (Formatted Function)			আনফরমেটেড ফাংশন (Unformatted Function)		
ডেটাটাইপ	ইনপুট	আউটপুট	ডেটাটাইপ	ইনপুট	আউটপুট
char	<code>scanf("%c",&amp;var);</code>	<code>printf("%c",var);</code>	char	<code>getch()</code> <code>var=getchar()</code>	<code>putch()</code> <code>putchar(var)</code>
string	<code>scanf("%s",&amp;var);</code>	<code>printf("%s",var);</code>	string	<code>gets(array_var)</code>	<code>puts(array_var)</code>
int	<code>scanf("%d",&amp;var);</code>	<code>printf("%d",var);</code>	int	-	-
float	<code>scanf("%f",&amp;var);</code>	<code>printf("%f",var);</code>	float	-	-
double	<code>scanf("%lf",&amp;var);</code>	<code>printf("%lf",var);</code>	double	-	-

- **getchar() ফাংশন:** কি-বোর্ডের সাহায্যে একটি করে ক্যারেক্টার বা অক্ষর ইনপুট করে তা কম্পিউটারের মেমোরিতে সংরক্ষণ করার জন্য ব্যবহৃত ফাংশন হলো `getchar()`। `getchar()` ফাংশনের সাধারণ গঠন বা সিনট্যাক্স (Syntax) হলোঃ **`variable_name= getchar();`**  
যেখানে `variable_name` হলো সি প্রোগ্রামে ব্যবহৃত গ্রহণযোগ্য char type এর চলক।
- **gets() ফাংশন:** কি-বোর্ড হতে স্ট্রিং জাতীয় ডাটা ইনপুট করে তা কম্পিউটারের মেমোরিতে সংরক্ষণ করার জন্য ব্যবহৃত ফাংশন হলো `gets()`। `getchar()` ফাংশনের সিনট্যাক্স (Syntax) হলোঃ **`gets(variable_name);`**  
যেখানে `variable_name` হলো সি প্রোগ্রামে ব্যবহৃত গ্রহণযোগ্য string type এর চলক।
- **scanf() ফাংশন:** প্রোগ্রাম চলার সময় ব্যবহারকারীর কাছ থেকে int, float, char ইত্যাদি টাইপের ডাটা ইনপুট নেয়ার জন্য বহুল ব্যবহৃত ইনপুট স্টেটমেন্ট হলো `scanf()`। `scanf()` এর সিনট্যাক্স (Syntax) হলোঃ **`scanf("F_S",&variable_name);`**  
এখানে F\_S হলো ফরমেট স্পেসিফিকার যা নির্দেশ করে ব্যবহারকারীর কাছ থেকে কোন টাইপের ডেটা নেয়া হবে। `variable_name` হলো পূর্ব ঘোষিত কোনো ভেরিয়েবলের নাম ও `&` হলো অ্যাড্রেস অপারেটর যা ভেরিয়েবলের জন্য গৃহীত মান মেমোরির ঐ লোকেশনে সংরক্ষণ করে। আর `&variable_name` নির্দেশ করে address of variable অর্থাৎ ব্যবহারকারী যে ডেটা ইনপুট করবে, তা উক্ত ভ্যারিয়েবলের জন্য নির্ধারিত মেমোরি অ্যাড্রেসে সংরক্ষিত হবে। অ্যাড্রেস অপারেটর (&) ব্যবহার না করলে গৃহীত মান মেমোরিতে ভেরিয়েবলের সঠিক লোকেশনে প্রেরিত হয় না বিধায় প্রোগ্রাম নির্বাহে সঠিক ফলাফল নাও আসতে পারে। আর এজন্য কম্পাইলার

কোনো বার্তা প্রদর্শন নাও করতে পারে। scanf() ফাংশনের মাধ্যমে একাধিক ডেটা ইনপুটে নেওয়া যায়। scanf() এর মাধ্যমে একাধিক ডেটা নেওয়ার সিনট্যাক্স হলোঃ

**printf(" F\_SF\_S ..... F\_S", variable\_1, variable\_2, ....., variable\_n);**

অবশ্য variable\_1, variable\_2, ....., variable\_n একই টাইপের ডেটার জন্য হতে পারে অথবা ভিন্ন ভিন্ন টাইপের ডেটার জন্যও হতে পারে।

- **putchar() ফাংশন:** সি প্রোগ্রামে একই সময়ে একটি ক্যারেক্টার মনিটরের পর্দায় প্রদর্শন করার জন্য putchar() ফাংশনটি ব্যবহৃত হয়। putchar() ফাংশনের সিনট্যাক্স হলোঃ **putchar(variable\_name);** যেখানে variable\_name হলো সি প্রোগ্রামে ব্যবহৃত গ্রহণযোগ্য একটি ভ্যারিয়েবল যার টাইপ অবশ্যই char হবে।
- **puts() ফাংশন:** ফাঁকা স্পেসসহ স্ট্রিং মনিটরে প্রদর্শন করার জন্য ফাংশন ব্যবহৃত হয়। puts() ফাংশনের সিনট্যাক্স হলোঃ **puts(variable\_name);** যেখানে variable\_name হলো সি প্রোগ্রামে ব্যবহৃত গ্রহণযোগ্য একটি ভ্যারিয়েবল যার টাইপ অবশ্যই string হবে।
- **printf() ফাংশন:** আউটপুট স্টেটমেন্ট হিসেবে সি ভাষায় বহুল ব্যবহৃত ফাংশন হলো printf() যার সাহায্যে স্ট্রিং বা স্ট্রিং সহবিভিন্ন টাইপের (যেমন int, float, char ইত্যাদি) ডাটার মান মনিটরের স্ক্রিনে প্রদর্শন করা যায়। অর্থাৎ printf() ফাংশনের বিভিন্ন ধরনের গঠন লক্ষ্য করা যায়। যথাঃ
  - (১) শুধুমাত্র স্ট্রিং প্রদর্শন করার জন্য printf() ফাংশনের সিনট্যাক্স হলোঃ **printf("String" );** এখানে String হিসেবে যেকোনো word বা character বা sentence হতে পারে। অর্থাৎ printf() ফাংশন এর প্রথমবন্ধনীর ( ) ভিতর ডবল কোটেশনের মধ্যে যা লেখা হয় printf() ফাংশনটি মনিটরের পর্দায় তাই প্রদর্শন করে।
  - (২) স্ট্রিংসহ বিভিন্ন টাইপের ডেটার ভ্যারিয়েবলের মান প্রদর্শন করার জন্য printf() ফাংশনের সাধারণ গঠন বা সিনট্যাক্স হলোঃ **printf("F\_S", variable\_name);** এখানে F\_S হলো ফরম্যাট স্পেসিফায়ার যা নির্দেশ করে variable\_name কোন ধরনের ডেটাইপের ডেটা নির্দেশ করছে। variable\_name চলক যে ডেটাইপের ঘোষণা করা করা হয়েছে, F\_S হবে সেই ডেটাইপের ফরম্যাট স্পেসিফায়ার। ফরম্যাট স্পেসিফায়ার ঠিক না থাকলে প্রোগ্রাম নির্বাহে সঠিক ফলাফল নাও আসতে পারে। আর এজন্য কম্পাইলার কোনোবর্তা প্রদর্শন নাও করতে পারে। printf() ফাংশনের মাধ্যমে একাধিক ডেটাকে আউটপুটে দেখানো যায়। অর্থাৎ printf() এর সিনট্যাক্সটি হলোঃ

**printf(" F\_SF\_S ..... F\_S", variable\_1, variable\_2, ....., variable\_n);**

অবশ্য variable\_1, variable\_2, ....., variable\_n একই টাইপের ডেটার জন্য হতে পারে অথবা ভিন্ন ভিন্ন টাইপের ডেটার জন্যও হতে পারে।

### ফরম্যাট স্পেসিফায়ার (Format Specifier)

সি প্রোগ্রামে ফরম্যাটেড (কাজিত আকারে) ভেরিয়েবলের মান গ্রহণ ও প্রদর্শনের জন্য যে সকল ক্যারেক্টার সেট ব্যবহৃত হয় তাদেরকে ফরম্যাট স্পেসিফায়ার (Format Specifier) বলা হয়। ফরম্যাট স্পেসিফায়ার হলো এক ধরনের কন্ট্রোল স্ট্রিং যা নির্দেশ করে ব্যবহারকারীর কাছ থেকে কোন টাইপের ডাটা নেওয়া হবে অথবা কোন ধরনের ডেটা আউটপুটে দেওয়া হবে। বিভিন্ন ধরনের ফরম্যাট স্পেসিফায়ার এর ব্যবহার দেয়া হলো :

ফরম্যাট স্পেসিফায়ার	কেন ব্যবহার করা হবে	উদাহরণ
%c	একটি char টাইপ মান ইনপুট/আউটপুট করার জন্য	scanf("%c",&data); printf("%c",data);
%d	int টাইপ মান ইনপুট / আউটপুট করার জন্য	scanf("%d",&data); printf("%d",data);
%ld	long int টাইপ মান ইনপুট/আউটপুট করার জন্য	scanf("%ld",&data); printf("%ld",data);
%e	float টাইপ মান এক্সপোনেনসিয়াল e নোটেসানে ইনপুট / আউটপুট করার জন্য	scanf("%e",&data); printf("%e",data);
%E	float টাইপ মান এক্সপোনেনসিয়াল E নোটেসানে ইনপুট / আউটপুট করার জন্য	scanf("%E",&data); printf("%E",data);
%f	float টাইপ মান ইনপুট / আউটপুট করার জন্য	scanf("%f",&data); printf("%f",data);
%lf	double টাইপ মান ইনপুট / আউটপুট করার জন্য	scanf("%lf",&data); printf("%lf",data);
%g	float টাইপ মান %f অথবা %e নোটেসানে ইনপুট / আউটপুট করার জন্য	scanf("%g",&data); printf("%g",data);
%G	float টাইপ মান %f অথবা %E নোটেসানে ইনপুট / আউটপুট করার জন্য	scanf("%G",&data); printf("%G",data);
%hd	Read a short integer value	scanf("%hd",&data); printf("%hd",data);
%o	অক্টাল টাইপ মান ইনপুট / আউটপুট করার জন্য	scanf("%o",&data); printf("%o",data);
%s	স্ট্রিং টাইপ মান ইনপুট/আউটপুট করার জন্য	scanf("%s",&data); printf("%s",data);
%[^\n]	স্ট্রিং টাইপ মান ইনপুট/আউটপুট করার জন্য	scanf("%[^\n]",data); printf("%s",data);
%u	unsigned int টাইপ মান ইনপুট / আউটপুট করার জন্য	scanf("%u",&data); printf("%u",data);

ফরম্যাট স্পেসিফায়ার	কেন ব্যবহার করা হবে	উদাহরণ
%lu	unsigned int টাইপ মান ইনপুট / আউটপুট করার জন্য	scanf("%lu",&data); printf("%lu",data);
%x	হেক্সাডেসিমেল টাইপ মান(a,b,...,f) ইনপুট / আউটপুট করার জন্য	scanf("%x",&data); printf("%x",data);
%X	হেক্সাডেসিমেল টাইপ মান(A,B,..,F) ইনপুট / আউটপুট করার জন্য	scanf("%X",&data); printf("%X",data);

ইনপুট-আউটপুট( I/O) এর জন্য সবচেয়ে বেশি ব্যবহৃত দুটি ফাংশন হলো **printf()** এবং **scanf()**। স্ট্যান্ডার্ড ইনপুট ডিভাইস(কীবোর্ড) এর মাধ্যমে ইউজার থেকে ফরম্যাটেড ইনপুট নেওয়ার জন্য **scanf()** ফাংশন ব্যবহৃত হয়। পক্ষান্তরে স্ট্যান্ডার্ড আউটপুট ডিভাইস(মনিটর) এ ফরম্যাটেড আউটপুট পাঠানোর জন্য **printf()** ফাংশন ব্যবহৃত হয়।

ইনপুট এর সিনট্যাক্স:

```
scanf("%c", &var_name); // Character ডেটা ইনপুট নেওয়ার জন্য ব্যবহৃত হয়।
scanf("%d", &var_name); // integer ডেটা ইনপুট নেওয়ার জন্য ব্যবহৃত হয়।
scanf("%f", &var_name); // float ডেটা ইনপুট নেওয়ার জন্য ব্যবহৃত হয়।
scanf("%lf", &var_name); // double ডেটা ইনপুট নেওয়ার জন্য ব্যবহৃত হয়।
scanf("%s", &var_name); // string ডেটা ইনপুট নেওয়ার জন্য ব্যবহৃত হয়।
```

আউটপুট এর সিনট্যাক্স:

```
printf("%c", var_name); // Character টাইপের ডেটা আউটপুট পাওয়ার জন্য ব্যবহৃত হয়।
printf("%d", var_name); // integer টাইপের ডেটা আউটপুট পাওয়ার জন্য ব্যবহৃত হয়।
printf("%f", var_name); // float টাইপের ডেটা আউটপুট পাওয়ার জন্য ব্যবহৃত হয়।
printf("%lf", var_name); // double টাইপের ডেটা আউটপুট পাওয়ার জন্য ব্যবহৃত হয়।
printf("%s", var_name); // string টাইপের ডেটা আউটপুট পাওয়ার জন্য ব্যবহৃত হয়।
```



কাজ:

১. scanf() ও gets এর মধ্যে পার্থক্য দেখাও।
২. printf() ও puts এর মধ্যে পার্থক্য দেখাও।
৩. %c, %f, %e, %u, %x, %hd, %lf সম্পর্কে লেখ।

## পাঠ ২৪

## ব্যবহারিক: ব্যবহারিক নির্দেশাবলি ও কিছু প্রোগ্রাম প্র্যাকটিস

অনুশীলনের জন্য main() ফাংশনের কিছু স্ট্রাকচার লক্ষ করি—

প্রথম প্রোগ্রামটি—

```
#include <stdio.h>

main()
{
    printf("Hello World!");
    return 0;
}
```

‘সি’ প্রোগ্রামে main() ফাংশন অবশ্যই থাকে। আর প্রত্যেকটি ফাংশনের একটি রিটার্ন টাইপ ঘোষণা করে দিতে হয়। কোনো ফাংশনের শুরুতে রিটার্ন টাইপ উল্লেখ না করলে তা অপারেটিং সিস্টেমকে একটি int টাইপ করবে বলে ধরে নেওয়া হয়। প্রথম প্রোগ্রামে কোনো রিটার্ন টাইপ উল্লেখ করা নেই। শুধুমাত্র রিটার্ন স্টেটমেন্টের সাহায্যে 0 রিটার্ন করা হয়েছে তবে এটা না করলেও কোনো সমস্যা হবে না।

দ্বিতীয় প্রোগ্রামটি—

```
#include<stdio.h>

int main( )
{
    printf ("Hello world !");
    return 0;
}
```

দ্বিতীয় প্রোগ্রামে int main() লেখা হয়েছে ফলে main() ফাংশন একটি ইন্টিজার মান রিটার্ন করবে তখন অবশ্যই প্রোগ্রামের শেষে return স্টেটমেন্ট ব্যবহার করতে হবে।

তৃতীয় প্রোগ্রামটি—

```
#include<stdio.h>

void main( )
{
    printf ("Hello world !");
}
```

তৃতীয় প্রোগ্রামে void নামক রিটার্ন টাইপ ব্যবহার করা হয়েছে যার ফলে ফাংশনের রিটার্ন টাইপ নাল(null) হবে। এক্ষেত্রে ফাংশনের কোথাও return ব্যবহার করা যাবে না। এটি ব্যবহার করলে এরর দেখাবে। রিটার্ন টাইপ কিছু না থাকলে সেক্ষেত্রে void ধরে নেওয়া হয়।

উপরোক্ত স্ট্রাকচার গুলোতে কোনো ক্ষেত্রে main() ফাংশনের শেষে রিটার্ন স্টেটমেন্ট ব্যবহৃত হয়েছে, কোনটির ক্ষেত্রে হয়নি, আবার কোনটির রিটার্ন টাইপের স্থলে void ব্যবহার করে রিটার্ন টাইপ বাতিল করে দেয়া হয়েছে। তবে ফাংশনের শেষে রিটার্ন টাইপ বাতিল না করে দিয়ে ফাংশনের শেষে একটি return; বা ‘return 0;’ স্টেটমেন্ট ব্যবহার করা বাঞ্ছনীয়।

কোনো ইউজার ডিফাইন্ড ফাংশনে কোন রিটার্ন স্টেটমেন্ট ব্যবহার না করলে কম্পাইলার সাধারণত প্রোগ্রাম কম্পাইল/নির্বাচন করলে কোন ভুলবার্তা দেখায় না। তবে কম্পাইল কালে "Function should have a return value" এরূপ একটি সতর্কবার্তা দেখায়। main() ফাংশনে কোন রিটার্ন স্টেটমেন্ট থাক বা না থাক তাতে প্রোগ্রামের ফলাফলে কোন পার্থক্য হয় না।

[বি.দ্রঃ CodeBlocks বহুল ব্যবহৃত C/C++ IDE(Integrated Development Environment) যেখানে C প্রোগ্রাম রান করার জন্য নানাবিধ টুলস (Tools) দেওয়া আছে। CodeBlocks-এ main ফাংশনের আগে return টাইপ না দিলেও কোনো এরর দেখায় না। আবার অনেক সময় int main() লেখার পর return 0; না লিখলেও কোনো এরর দেখায় না।



কারণ হলো, CodeBlocks একটি স্মার্ট IDE। return 0; না লিখলে সে নিজের মতো করে একটি মান return করে। তাই CodeBlocks কে একটি user friendly IDE বলা যায়।]

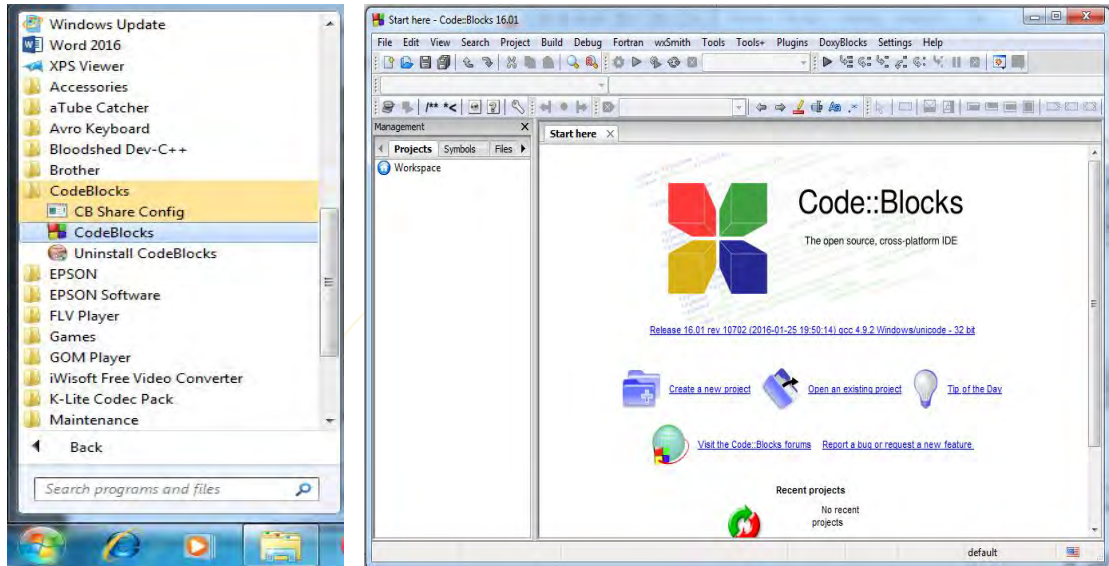
### সি ভাষায় আমাদের প্রথম প্রোগ্রাম

সি ভাষায় প্রোগ্রাম লেখার জন্য এডিটর প্রয়োজন হয়। বেশির ভাগ কম্পাইলারের সোর্স এডিটর থাকে, যেখানে প্রোগ্রাম লেখা যায় ও নির্বাহ করে ফলাফল দেখা যায়। যেমন- কোডব্লকস (CodeBlocks), টার্বো সি (Turbo C) ইত্যাদি। এখানে কোডব্লকস সফটওয়্যার ব্যবহার করে প্রোগ্রাম কিভাবে লেখা যায় তা নিয়ে আলোচনা করা হয়েছে।

### সি ভাষায় প্রোগ্রাম খোলা

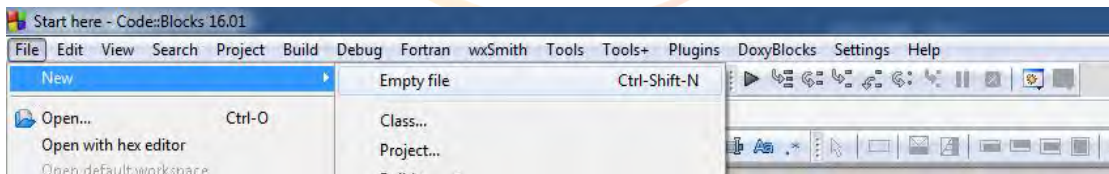
কোডব্লকস ব্যবহার করে সি প্রোগ্রাম লেখার জন্য নিচের ধাপসমূহ অনুসরণ করতে হবে-

১. প্রথমে Start→All Programs→CodeBlocks→CodeBlocks ক্লিক করলে কোডব্লকস প্রোগ্রাম চালু হবে।



চিত্র: কোডব্লকস প্রোগ্রাম চালু করণ

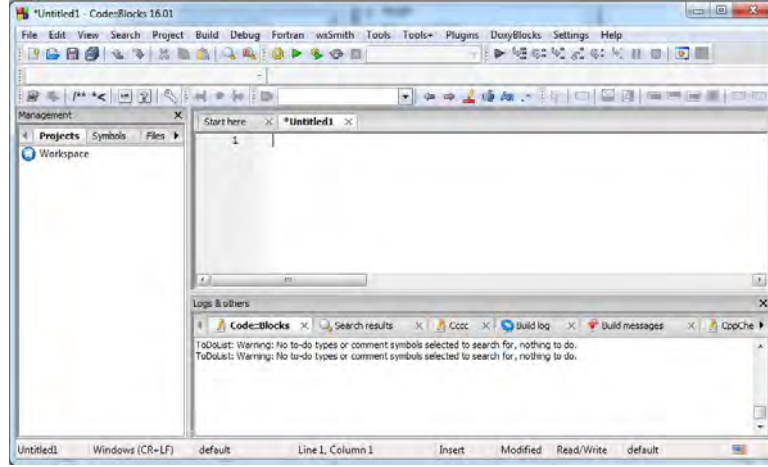
২. এবারে File মেনু হতে New→Empty File এ ক্লিক করি।



চিত্র: কোডব্লকস এ নতুন ফাইল খোলা

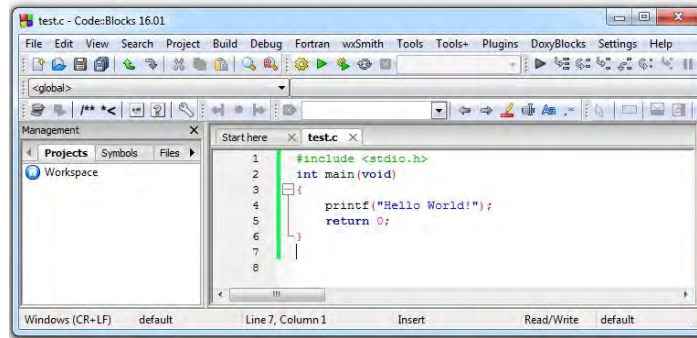
তাহলে নিচের মতো প্রোগ্রাম লেখার এডিটিং উইন্ডো আসবে। উপরের দিকে প্রোগ্রাম লেখার কার্সর দেখা যাবে।





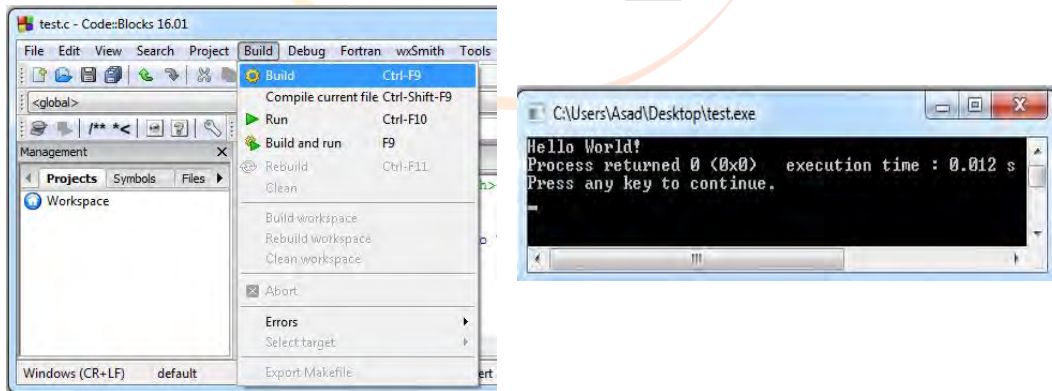
চিত্র: সি প্রোগ্রাম লেখার এডিটিং স্ক্রিন

৩. এখানে প্রয়োজনীয় কোড লিখতে হবে।  
 ৪. সি প্রোগ্রামের ফাইল সংরক্ষণ করার জন্য File মেনু হতে Save /Save as এ ক্লিক করলে ফাইল এর নাম লেখার ডায়ালগ বক্স আসবে। এক্ষেত্রে test.c নাম দিয়ে এন্টার কী চাপলে ফাইল সংরক্ষিত হয়ে যাবে।



চিত্র: নমুনাকৃত সি প্রোগ্রাম

৫. প্রোগ্রাম কম্পাইল করার জন্য Build→Compile Current file এ ক্লিক করতে হবে।



চিত্র: প্রোগ্রাম কম্পাইলিং ও রান করা

৬. প্রোগ্রাম Run করার জন্য Bulid→Run এ ক্লিক করতে হবে। ফলে আউটপুট উইন্ডো আসবে।

সি ভাষায় আমাদের প্রথম প্রোগ্রামের বিশ্লেষণ:

```
#include<stdio.h>
int main(void)
{
printf("Hello World!");
return 0;
}
```

উপরোক্ত প্রোগ্রামের ব্যাখ্যা নিম্নরূপ:

#include<stdio.h>	include অর্থ হচ্ছে কোনোকিছু যুক্ত করা। stdio এর পূর্ণরূপ হচ্ছে standard input output। stdio.h-এর .h দিয়ে বুঝানো হয় এটি একটি header ফাইল। অর্থাৎ সম্পূর্ণ লাইন দিয়ে বুঝানো হয় যে, standard input output কে যুক্ত করা।
int main(void)	এটিকে বলা হয় মেইন ফাংশন। আমরা যখন প্রোগ্রামটি রান করাবো তখন এ মেইন ফাংশন থেকে কাজ করা শুরু করবে। তাই সব প্রোগ্রামে একটি (শুধুমাত্র একটি) মেইন ফাংশন থাকতে হয়। মেইন ফাংশনের শুরুতে দ্বিতীয় বন্ধনী দিয়ে শুরু করতে হয়। মেইন ফাংশন শেষ করতেও হয় একটি দ্বিতীয় বন্ধনী দিয়ে। int main() মানে মূল ফাংশন ইন্টিজার (গাণিতিকপূর্ণ সংখ্যা) প্রদান করবে। আর (void) লিখার কারণে মূল ফাংশনে কোনো কিছু ইনপুট করতে হবে না।
{	এটি দ্বারা main() ফাংশনের শুরু বুঝানো হয়।
printf("Hello World!");	এখানে printf() হচ্ছে একটি ফাংশন। printf অর্থ হচ্ছে print formatted। এটি একটি লাইব্রেরি ফাংশন যাকে স্ট্যান্ডার্ড আউটপুট ফাংশন বলে। printf() এর কাজ হচ্ছে স্ক্রিনে কিছু প্রিন্ট করা। ডাবল কোটেশন চিহ্নেও ভিতরে যা লিখব তাই কনসোলে সে প্রিন্ট করবে। কম্পাইলারের যে উইন্ডোতে আউটপুট দেখা যায় সেটিকে কনসোল বলে। প্রথমেই একটি লাইন লিখেছি #include<stdio.h>। printf() ফাংশনটি কিভাবে কোনো কিছু প্রিন্ট করে তা লেখা রয়েছে এই stdio.h ফাইলে।
return 0;	main হচ্ছে একটা ফাংশন। প্রত্যেক ফাংশন এর একটা return মান থাকতে হয়। যা ফাংশন এর কাজ শেষে কিছু একটা রিটার্ন করে। return 0 মানে শূন্য রিটার্ন করা।
}	এটি দ্বারা main() ফাংশনের শেষ সীমা বুঝানো হয়।
printf("HelloWorld!"); ও return 0; কে বলে স্টেটমেন্ট (Statement)। 'সি' প্রোগ্রামিং-এ প্রতিটি স্টেটমেন্ট শেষে একটি করে সেমিকোলন (;) দিতে হয়। যদি সেমিকোলন দেওয়া হয় তাহলে কম্পাইলারে ভুল দেখাবে এবং প্রোগ্রামটি রান হবে না। এ ধরনের ভুল থাকলে কম্পাইল এরর (compile error) দেখায়।	

সি ভাষায় আমাদের দ্বিতীয় প্রোগ্রাম:

```
#include<stdio.h>
int main()
{
int a,b,sum;
printf("Enter first number:");
scanf("%d",&a);
printf("Enter second number:");
scanf("%d",&b);
sum=a+b;
printf("\nSum is: %d", sum);
return 0;
}
```

উপরোক্ত প্রোগ্রামের ব্যাখ্যা নিম্নরূপ:

#include<stdio.h>	প্রোগ্রামের মধ্যে দুইটি লাইব্রেরি ফাংশন ব্যবহার করা হয়েছে। যথা- printf() এবং scanf()। এই ফাংশন গুলো কিভাবে কাজ করে, তা stdio.h নামক হেডার ফাইলে বর্ণিত আছে। হেডার ফাইল শুরুতেই #include এর মাধ্যমে সংযুক্ত করা হয়েছে।
int main()	প্রত্যেকটি সি প্রোগ্রাম কম্পাইল ও নির্বাহ শুরু হয় main() ফাংশন থেকে। তাই প্রত্যেকটি সি প্রোগ্রাম লেখার সময় main() ফাংশন অবশ্যই লিখতে হয়।
{	এটি দ্বারা main() ফাংশনের শুরু বুঝানো হয়।
int a,b,sum;	প্রত্যেকটি সি প্রোগ্রামে সাময়িক ভাবে ডাটা রাখার জন্য কিছু ভ্যারিয়েবল ব্যবহৃত হয়। এই উদাহরণে দুইটি সংখ্যার মান রাখার জন্য a ও b দুইটি এবং ফলাফল রাখার জন্য sum ভ্যারিয়েবল ঘোষণা (Declare) করা হয়েছে। একাধিক ভ্যারিয়েবল ঘোষণা করার সময় দুটি ভ্যারিয়েবলের মাঝে কমা বসাতে হয় ও শেষে সেমিকোলন দিতে হয়।
printf("Enter first number:");	printf() ফাংশনের কাজ হলো কোন কিছু মনিটরে প্রদর্শন করা। এট ইনভার্টেড কমা র ভেতরের অংশ প্রদর্শন করে। নমুনা প্রোগ্রামে এই লাইনটি নিচের মতো করে আউটপুট দেখায়- Enter first number:
scanf("%d",&a);	scanf() ফাংশনের কাজ হলো কোন কি-বোর্ড থেকে কোন ভ্যালু নিয়ে কোন ভ্যারিয়েবলে রাখা। %d কে ফরম্যাট স্পেসিফায়ার বলে। ভ্যারিয়েবলে কোন ধরনের মান থাকবে যেমন- দশমিক, পূর্ণসংখ্যা, ক্যারেক্টার তা নির্ধারণ করতে ফরম্যাট স্পেসিফায়ার ব্যবহৃত হয়। আমাদের উদাহরণে, পূর্ণসংখ্যার জন্য %d ব্যবহৃত হয়েছে। ফলে ব্যবহারকারী প্রদত্ত পূর্ণসংখ্যার ইনপুট a ভ্যারিয়েবলে সংরক্ষিত হবে।
printf("Enter second number:");	নমুনা প্রোগ্রামে এই লাইনটি নিচের মতো করে আউটপুট প্রদর্শন করে- Enter Second number:
scanf("%d",&b);	ইউজার প্রদত্ত ইনপুট b ভ্যারিয়েবলে সংরক্ষিত হবে।
sum=a+b;	a ও b এর মান যোগ করে যোগফল sum ভ্যারিয়েবলে সংরক্ষিত হবে।
printf("\nSum is: %d", sum);	নমুনা প্রোগ্রামের ফলাফল প্রদর্শন করে।
return 0;	main হচ্ছে একটা ফাংশন। প্রত্যেক ফাংশন এর একটা return মান থাকতে হয়। যা ফাংশন এর কাজ শেষে কিছু একটা রিটার্ন করে। return 0 মানে শূন্য রিটার্ন করা।
}	এটি দ্বারা main() ফাংশনের শেষ সীমা বুঝানো হয়।

প্রোগ্রামের আরও কিছু উদাহরণ দেওয়া হলো।

<p><b>উদাহরণ-১.</b> শুধুমাত্র একটি ক্যারেক্টার ইনপুট দিয়ে তা আউটপুট দেখানোর জন্য [getchar() ও putchar()] ফাংশন এর ব্যবহার প্রোগ্রাম লিখ।</p> <pre>#include&lt;stdio.h&gt; int main() {     int ch;     printf("Input One Character :");     ch=getchar();     printf("Output :");     putchar(ch);     getch();     return 0; }</pre> <p><b>আউটপুট:</b> Input One Character :B Output:B</p>	<p><b>উদাহরণ-২.</b> একাধিক শব্দ বিশিষ্ট স্ট্রিং এর ইনপুট অপারেশনে gets( ) ফাংশন এবং আউটপুট অপারেশনে puts( )ফাংশন ব্যবহার করে একটি প্রোগ্রাম লিখ।</p> <pre>#include&lt;stdio.h&gt; int main() {     char a[25];     printf("Type string:");     gets(a);     printf("Output string:");     puts(a);     return 0; }</pre> <p><b>আউটপুট:</b> Type string:I LoveBangladesh. Output string: I LoveBangladesh.</p>
--	---

এই প্রোগ্রামে a[25] অ্যারে ব্যবহার করা হয় যা পরবর্তীতে আলোচনা করা হয়েছে। এই অ্যারে ঘোষণার জন্য 25টি ক্যারেক্টারইনপুট দেওয়া যাবে।

<p><b>উদাহরণ-৩.</b> ছোট হাতের অক্ষরকে বড় হাতের অক্ষরে রূপান্তর করার প্রোগ্রাম লিখ।</p> <pre>#include&lt;stdio.h&gt;  void main() {     char n;     printf("Enter any lower case character:");     scanf("%c",&amp;n);     printf("You entered : %c",n-32); }</pre> <p><b>আউটপুট:</b></p> <p>Enter any lower case character: a ↵ You entered: A</p>	<p><b>উদাহরণ-৪.</b> বড় হাতের অক্ষরকে ছোট হাতের অক্ষরে রূপান্তর করার প্রোগ্রাম লিখ।</p> <pre>#include&lt;stdio.h&gt;  void main() {     char n;     printf("Enter any uppercase character:");     scanf("%c",&amp;n);     printf("You entered: %c",n+32); }</pre> <p><b>আউটপুট:</b></p> <p>Enter any uppercase character: A ↵ You entered: a</p>
---	--

অ্যাসকি কোডে A এর দশমিক মান হচ্ছে 65 অন্যদিকে a দশমিক মান হচ্ছে 97। ফলে দেখা যাচ্ছে, উভয়ই দশমিক মানের পার্থক্য হচ্ছে 32। অর্থাৎ বড় হাতের (Capital Letter) প্রতিটি অক্ষরের দশমিক মানের সাথে ছোট হাতের (Small Letter) প্রতিটি অক্ষরের দশমিক মানের পার্থক্য 32। সুতরাং বড় হাতের প্রতিটি অক্ষরের সাথে 32 যোগ করলে অক্ষরটি ছোট হাতের অক্ষরে পরিণত হবে। আর ছোট হাতের অক্ষরের হতে ৩২ বিয়োগ করলে অক্ষরটি বড় হাতের অক্ষরে পরিণত হবে।

এবারে আমরা **sizeof()** কীওয়ার্ডের মাধ্যমে বিভিন্ন ডেটা টাইপের সাইজ দেখার জন্য একটি প্রোগ্রাম তৈরি করি।

```
#include <stdio.h>
```

```
int main()
```

```
{
    printf("Char size: %d bytes.",sizeof(char));
    printf("\nInteger size: %d bytes.",sizeof(int));
    printf("\nFloat size: %d bytes.",sizeof(float));
    printf("\nDouble size: %d
bytes.",sizeof(double));
    return 0;
}
```

**আউটপুট:**

Char size: 1 bytes.  
Integer size: 2 bytes.  
Float size: 4 bytes.  
Double size: 8 bytes.

ফরম্যাট স্পেসিফায়ার সি প্রোগ্রামের একটি গুরুত্বপূর্ণ বিষয়। কারণ শুধুমাত্র ফরম্যাট স্পেসিফায়ার ব্যবহার করে একরূপ ডেটা থেকে অন্যরূপ ডেটা রূপান্তর করা সম্ভব। নিচে কিছু উদাহরণ দেওয়া হলো:

**উদাহরণ-১:** কীবোর্ড থেকে একটি ডেসিম্যাল বা দশমিক সংখ্যা ইনপুট দিয়ে অক্টাল ও হেক্সাডেসিম্যাল সংখ্যায় রূপান্তর করার জন্য সি ভাষায় প্রোগ্রাম লিখ।

```
#include<stdio.h>
void main()
{
    int a;
    printf("Decimal number: ");
    scanf("%d",&a);
    printf(" \nOctal number:
    %o",a);
    printf(" \nHexadecimal
    number: %x",a);
}
```

**ফলাফল:**

Decimal number: 10 ↵  
Octal number: 12  
Hexadecimal number: a

**উদাহরণ-২:** কীবোর্ড থেকে একটি অক্টাল সংখ্যা ইনপুট দিয়ে ডেসিমেল ও হেক্সাডেসিম্যাল সংখ্যায় রূপান্তর করার জন্য সি ভাষায় প্রোগ্রাম লিখ।

```
#include<stdio.h>
void main()
{
    int a;
    printf("Octal number: ");
    scanf("%o",&a);
    printf(" \nDecimal number:
    %d",a);
    printf(" \nHexadecimal
    number: %x",a);
}
```

**ফলাফল:**

Octal number: 177 ↵  
Decimal number: 127  
Hexadecimal number: 7f

**উদাহরণ-৩:** কীবোর্ড থেকে একটি হেক্সাডেসিম্যাল সংখ্যা ইনপুট দিয়ে অক্টাল ও ডেসিম্যাল সংখ্যায় রূপান্তর করার জন্য সি ভাষায় প্রোগ্রাম লিখ।

```
#include<stdio.h>
void main()
{
    int a;
    printf("Hexadecimal
    number: ");
    scanf("%x",&a);
    printf(" \nDecimal
    number: %d",a);
    printf(" \nOctal number:
    %o",a);
}
```

**ফলাফল:**

Hexadecimal number: a ↵  
Decimal number: 10  
Octal number: 12

নিম্নে কিছু সমস্যার অ্যালগরিদম, ফ্লোচার্ট ও সি ভাষায় লিখিত প্রোগ্রাম দেওয়া হলো।

**উদাহরণ-১.** দুটি সংখ্যার যোগফল নির্ণয়ের জন্য অ্যালগরিদম, ফ্লোচার্ট ও প্রোগ্রাম লিখ।

**অ্যালগরিদম:**

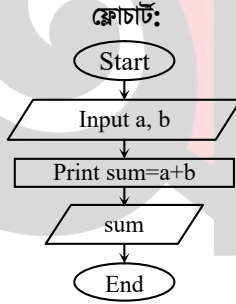
ধাপ-১: প্রোগ্রাম শুরু

ধাপ-২: a, b এর মান গ্রহণ

ধাপ-৩:  $sum = a + b$  নির্ণয়

ধাপ-৪: sum এর মান ছাপাই

ধাপ-৫: প্রোগ্রাম শেষ



**প্রোগ্রাম:**

```
#include<stdio.h>
void main()
{
    int a, b, sum;
    scanf("%d %d",&a,&b);
    sum = a + b;
    printf("%d",sum);
}
```

**উদাহরণ-২.** ত্রিভুজের ভূমি ও উচ্চতা দেওয়া থাকলে ক্ষেত্রফল নির্ণয়ের অ্যালগরিদম, ফ্লোচার্ট ও প্রোগ্রাম লিখ।

**অ্যালগরিদম:**

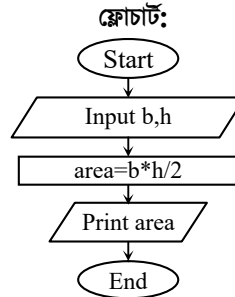
ধাপ-১: প্রোগ্রাম শুরু

ধাপ-২: b (ভূমি) এবং h (উচ্চতা) এর মান গ্রহণ

ধাপ-৩:  $area = b * h / 2$  নির্ণয় করি।

ধাপ-৪: area এর মান ছাপাই।

ধাপ-৫: প্রোগ্রাম শেষ করি।



**প্রোগ্রাম:**

```
#include<stdio.h>
void main()
{
    float b,h, area;
    scanf("%f %f",&b,&h);
    area = b * h / 2;
    printf("%f", area);
}
```

উদাহরণ-৩. ত্রিভুজের তিন বাহুর দৈর্ঘ্য দেওয়া থাকলে ক্ষেত্রফল নির্ণয়ের অ্যালগরিদম, ফ্লোচার্ট ও প্রোগ্রাম লিখ।

অ্যালগরিদম:

ধাপ-১: প্রোগ্রাম শুরু

ধাপ-২: a, b এবং c এর মান গ্রহণ

ধাপ-৩:  $s = (a+b+c)/2$  নির্ণয় করি।

ধাপ-৪:  $area = \sqrt{s(s-a)(s-b)(s-c)}$  নির্ণয়

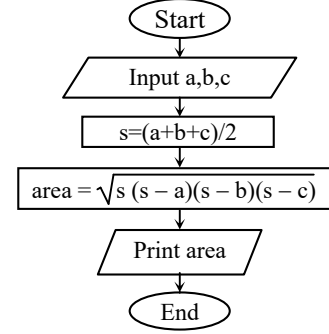
ধাপ-৫: area এর মান ছাপাই

ধাপ-৬: প্রোগ্রাম শেষ

প্রোগ্রাম:

```
#include<stdio.h>
#include<math.h>
main()
{
    float a, b, c, s, area;
    scanf("%f %f %f", &a,&b,&c);
    s = (a + b + c)/2;
    area = sqrt(s*(s-a)*(s-b)*(s-c));
    printf("Area of triangle is = %f", area);
}
```

ফ্লোচার্ট:



উদাহরণ-৪. বৃত্তের ক্ষেত্রফল নির্ণয়ের অ্যালগরিদম, ফ্লোচার্ট ও প্রোগ্রাম লিখ।

অ্যালগরিদম:

ধাপ-১: প্রোগ্রাম শুরু করি।

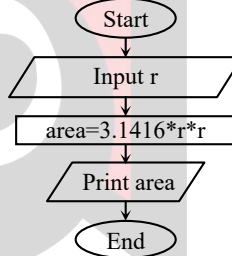
ধাপ-২: ইনপুট হিসেবে r (ব্যাসার্ধ)  
এর মান গ্রহণ করি।

ধাপ-৩:  $area = 3.1416 * r * r$  ব্যবহার করে  
area এর মান নির্ণয় করি।

ধাপ-৪: area এর মান ছাপাই।

ধাপ-৫: প্রোগ্রাম শেষ করি।

ফ্লোচার্ট:



প্রোগ্রাম:

```
#include<stdio.h>
main ( )
{
    float r, area;
    scanf ("%f", &r) ;
    area = 3.14*r*r;
    printf("%f", area);
}
```

উদাহরণ-৫. আয়তের ক্ষেত্রফল নির্ণয়ের অ্যালগরিদম, ফ্লোচার্ট ও প্রোগ্রাম লিখ।

অ্যালগরিদম:

ধাপ-১: প্রোগ্রাম শুরু করি।

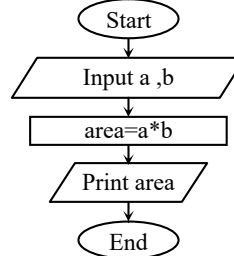
ধাপ-২: ইনপুট হিসেবে a (দৈর্ঘ্য) এবং b (প্রস্থ)  
এর মান গ্রহণ করি।

ধাপ-৩:  $area = a * b$  ব্যবহার করে  
area এর মান নির্ণয় করি।

ধাপ-৪: area এর মান ছাপাই।

ধাপ-৫: প্রোগ্রাম শেষ করি।

ফ্লোচার্ট:



প্রোগ্রাম:

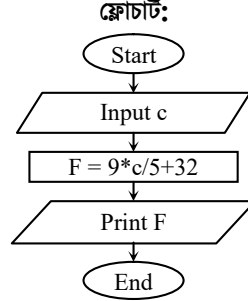
```
#include<stdio.h>
main()
{
    int a,b,area;
    scanf("%d %d",&a,&b);
    area=a*b;
    printf("%d", area);
}
```



উদাহরণ-৬. সেন্টিগ্রেড তাপমাত্রাকে ফারেনহাইট-এ রূপান্তরের জন্য অ্যালগরিদম, ফ্লোচার্ট ও প্রোগ্রাম।

অ্যালগরিদম:

- ধাপ-১: প্রোগ্রাম শুরু করি।  
 ধাপ-২: ইনপুট হিসেবে c এর মান গ্রহণ করি।  
 ধাপ-৩:  $F = 9 * c / 5 + 32$  ব্যবহার করে F এর মান নির্ণয় করি।  
 ধাপ-৪: F এর মান ছাপাই।  
 ধাপ-৫: প্রোগ্রাম শেষ করি।



প্রোগ্রাম:

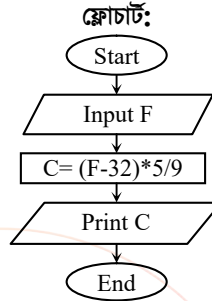
```

#include<stdio.h>
main()
{
    float c, F;
    scanf("%f",&c);
    F=9*c/5+32;
    printf("%f",F);
}
  
```

উদাহরণ-৭. ফারেনহাইট তাপমাত্রাকে সেন্টিগ্রেড -এ রূপান্তরের জন্য অ্যালগরিদম, ফ্লোচার্ট ও প্রোগ্রাম।

অ্যালগরিদম:

- ধাপ-১: প্রোগ্রাম শুরু করি।  
 ধাপ-২: ইনপুট হিসেবে F এর মান গ্রহণ করি।  
 ধাপ-৩:  $C = (F-32) * 5 / 9$  ব্যবহার করে C এর মান নির্ণয় করি।  
 ধাপ-৪: C এর মান ছাপাই।  
 ধাপ-৫: প্রোগ্রাম শেষ করি।



প্রোগ্রাম:

```

#include<stdio.h>
main()
{
    float C,F;
    scanf("%f",&F);
    C=5*(F-32)/9;
    printf("%f",C);
}
  
```

উদাহরণ-৮. কোনো পরিমাপ ফুটকে মিটারে প্রকাশ করার প্রোগ্রাম।

[সূত্র : ১ মিটার = ৩.২৮ ফুট]

```

#include<stdio.h>
main()
{
    float m,f;
    printf("Enter feet: ");
    scanf("%f",&f);
    m= f/3.28;
    printf("\n Meter is %.2f",m);
}
  
```

ফলাফল:

Enter feet: 6.56 ↵  
 Meter is 2.00

উদাহরণ-৯. কোনো পরিমাপ ইঞ্চিকে সেন্টিমিটারে প্রকাশ করার প্রোগ্রাম।

[সূত্র : ১ ইঞ্চি = ২.৫৪ সেন্টিমিটার]

```

#include<stdio.h>
main()
{
    int inch;
    float cm;
    printf ("Integer value for inches:");
    scanf ("%d", &inch) ;
    cm=inch*2.54;
    printf("\nCentimeter=%.2f", cm);
}
  
```

ফলাফল:

Integer value for inches: 5 ↵  
 Centimeter = 12.70



কাজ:

- সুমাইয়া ও জাফরিগের গণিত বিষয়ে প্রাপ্ত নম্বর যথাক্রমে x এবং y। যোগের মাধ্যমে সুমাইয়া ও জাফরিগের গণিতে প্রাপ্ত নম্বরের পার্থক্য নির্ণয় করা সম্ভব— সি ভাষার সাহায্যে বিশ্লেষণ করো।
- একটি গরু x মিটার দৈর্ঘ্যের একটি রশি বা দড়ি দিয়ে একটি খুঁটির সাথে এমনভাবে বাঁধা আছে যাতে গরুটি অবোধে চলাচল করতে পারে। গরুটি ১ পাকে সর্বোচ্চ যে পথ পরিভ্রমণ করতে পারে তার দৈর্ঘ্য এবং উক্ত পথ দ্বারা তৈরিকৃত ক্ষেত্রের ক্ষেত্রফল নির্ণয়ের জন্য প্রোগ্রাম লেখ।



### ৫.১২.১৪ কন্ডিশনাল স্টেটমেন্ট (Conditional Statement)

সি প্রোগ্রাম কতগুলো এক্সপ্রেশনের সমন্বয়ে গঠিত। প্রতিটি এক্সপ্রেশন কতগুলো টোকেন, কী-ওয়ার্ড, আইডেন্টিফায়ার অপারেটর, অপারেন্ড ইত্যাদি নিয়ে গঠিত। এরূপ এক্সপ্রেশন বা ফাংশন সমূহকে যখন সেমিকোলন (;) দিয়ে শেষ করা হয় তখন তাকে স্টেটমেন্ট বলে। সি ভাষায় স্টেটমেন্ট সমূহকে প্রধান দুটি ভাগে ভাগ করা যায়। যথা:

- **সিম্পল স্টেটমেন্ট (Simple Statement):** এক এক্সপ্রেশন কিংবা ফাংশন নিয়ে গঠিত স্টেটমেন্টকে সিম্পল স্টেটমেন্ট বলে। সিম্পল স্টেটমেন্ট সাধারণত সেমিকোলন (;) দ্বারা শেষ হয়। যেমন:

```
x=y+7;
```

```
printf("This is simple statment");
```

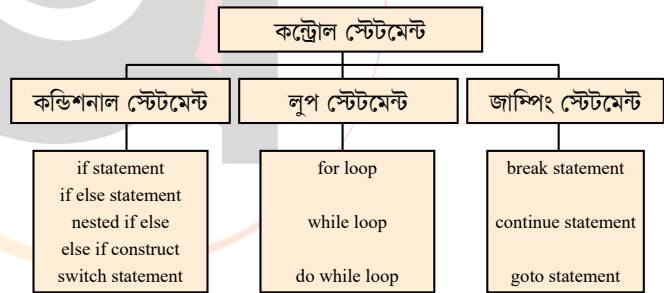
এদের প্রত্যেকটি একেকটা সিম্পল স্টেটমেন্ট।

- **কম্পাউন্ড স্টেটমেন্ট (Compound Statement) :** দুই বা ততোধিক সিম্পল স্টেটমেন্টকে যখন '{ }' বন্ধনীর মধ্যে লেখা হয় তখন তাকে কম্পাউন্ড স্টেটমেন্ট বলে। কম্পাউন্ড স্টেটমেন্টের জন্য ক্লোজিং দ্বিতীয় বন্ধনীর শেষে কোনো সেমিকোলন (;) দিতে হয় না। কম্পাউন্ড স্টেটমেন্টকে আবার ব্লক স্টেটমেন্টও বলা হয়। একটি কম্পাউন্ড স্টেটমেন্ট অন্য কোনো কম্পাউন্ড স্টেটমেন্টকেও ধারণ করতে পারে। যেমন:

```
{
    x=0;
    printf("This is compound statment");
    ++x;
}
```

### কন্ট্রোল স্ট্রাকচার (Control Structure)

'সি' প্রোগ্রামের কোনো স্টেটমেন্টকে দুই বা ততোধিকবার স্বয়ংক্রিয়ভাবে ও পর্যায়ক্রমে সম্পাদনের ক্ষেত্রে কন্ট্রোল স্ট্রাকচার ব্যবহার হয়। কোনো স্টেটমেন্টে শর্ত সাপেক্ষে অপর কোনো স্টেটমেন্টের ভিত্তিতে সম্পাদনের প্রয়োজন হয়, সেক্ষেত্রে এটি ব্যবহৃত হয়। এই সকল স্টেটমেন্টের নিয়ন্ত্রণ একজন প্রোগ্রামার কন্ট্রোল স্ট্রাকচার এর মাধ্যমে সমাধান করেন। 'সি' প্রোগ্রামে কন্ট্রোল স্টেটমেন্ট-সমূহকে প্রধান দু'ভাগে ভাগ করা যায়। যেমন-



১. কন্ডিশনাল কন্ট্রোল স্টেটমেন্ট (Conditional Control Statement)
২. লুপ কন্ট্রোল স্টেটমেন্ট (Loop Control Statement)
৩. জাম্পিং কন্ট্রোল স্টেটমেন্ট (Jumping Control Statement)

**কন্ডিশনাল কন্ট্রোল স্টেটমেন্ট (Conditional Control Statement):** 'সি' প্রোগ্রামে শর্ত সাপেক্ষে কোনো স্টেটমেন্ট সম্পাদনের জন্য কন্ডিশনাল কন্ট্রোল ব্যবহৃত হয়। কন্ডিশনাল কন্ট্রোলে ব্যবহৃত শর্ত সত্য হলে প্রোগ্রামে এক

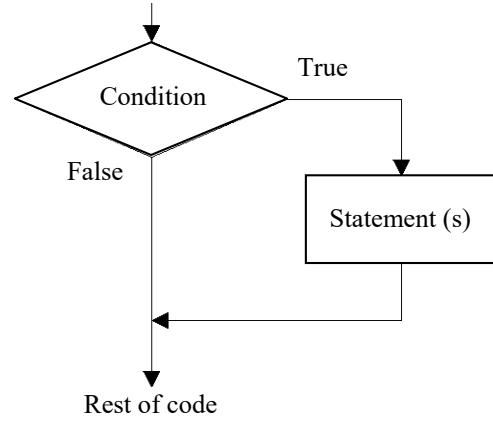
ধরনের ফলাফল পাওয়া যায় ও মিথ্যা হলে অন্য ধরনের ফলাফল পাওয়া যায়। অন্যতম কন্ডিশনাল কন্ট্রোল স্টেটমেন্টগুলো হচ্ছে:

- if স্টেটমেন্ট
- if.....else স্টেটমেন্ট
- else if স্টেটমেন্ট (বা nested if স্টেটমেন্ট)
- switch স্টেটমেন্ট

### if স্টেটমেন্ট

কোন সিদ্ধান্তমূলক কাজ উপযোগী প্রোগ্রাম তৈরির জন্য স্টেটমেন্ট ব্যবহৃত হয়। সাধারণত একটি শর্ত সাপেক্ষে কোনো কাজ সম্পাদনের জন্য if স্টেটমেন্ট ব্যবহৃত হয়। if স্টেটমেন্ট ব্যবহারের ফরম্যাট হলো-

```
if (Condition)
{
    Action1;
}
```



চিত্র : if স্টেটমেন্টের ফ্লোচার্ট

এখানে if প্রথমে Condition কে মূল্যায়ন করে। Condition যদি True (শূন্য না) হয় তাহলে if ব্লকের মধ্যে অবস্থিত Action1 সম্পাদিত হয়। আর Condition যদি False (শূন্য) হয় তাহলে if ব্লকের মধ্যে অবস্থিত Action1 এড়িয়ে যায়।

if স্টেটমেন্টে শর্ত সাধারণত এক বা একাধিক লজিক্যাল বা রিলেশনাল এক্সপ্রেশন হয় যা if পরবর্তী প্রথম বন্ধনীর মধ্যে লেখা হয়। if (Condition) স্টেটমেন্টের পর কোনো সেমিকোলন (;) হবে না। Action1 একটি স্টেটমেন্ট হতে পারে আবার একাধিক স্টেটমেন্ট হতে পারে। একটি স্টেটমেন্ট হলে দ্বিতীয় বন্ধনী দেবার প্রয়োজন নেই।

**উদাহরণ-১:** কোনো সংখ্যা ধনাত্মক না ঋণাত্মক তা নির্ণয়ের জন্য প্রোগ্রাম লেখ।

```
#include<stdio.h>
int main()
{
    int a;
    printf("Enter a value :");
    scanf("%d",&a);
    if (a>=0)
        printf("%d is a positive number.", a);
    if (a<0)
        printf("%d is a negative number.", a);
    return 0;
}
```

**ফলাফল:**

Enter a value : 15  
15 is a positive number.

**উদাহরণ-২:** কোনো সংখ্যা জোড় না বিজোড় তা নির্ণয়ের জন্য প্রোগ্রাম লেখ।

```
#include<stdio.h>
int main()
{
    int a;
    printf("Enter a value :");
    scanf("%d",&a);
    if (a%2==0)
        printf("%d is a Even number.", a);
    if (a%2==1)
        printf("%d is a Odd number.", a);
    return 0;
}
```

**ফলাফল:**

Enter a value : 16  
16 is a Even number.

উদাহরণ-৩: শুধুমাত্র if স্টেটমেন্ট ব্যবহার করে তিনটি সংখ্যার মধ্যে বড় সংখ্যা নির্ণয় করার প্রোগ্রাম [অ্যালগরিদম ও ফ্লোচার্ট সহ] লিখ।

ধাপ-১: প্রোগ্রাম শুরু।

ধাপ-২: তিনটি সংখ্যা a, b এবং c এর মান গ্রহণ।

ধাপ-৩: ধরি, max=a

ধাপ-৪: যদি max<b সত্য হয় তাহলে max=b।

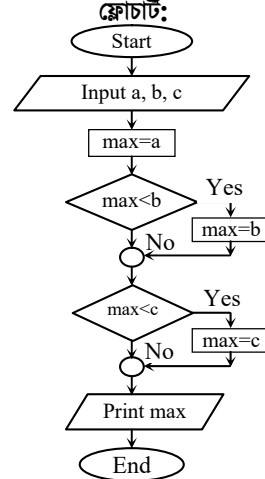
অন্যথায় ৫ নং ধাপে যেতে হবে।

ধাপ-৫: যদি max<c সত্য হয় তাহলে max=c।

অন্যথায় ৬ নং ধাপে যেতে হবে।

ধাপ-৬: ফলাফলে max এর মান ছাপাতে হবে।

ধাপ-৭: প্রোগ্রাম শেষ।



```

#include<stdio.h>
main()
{
    int a,b,c,max;
    scanf("%d %d %d",&a,&b,&c);
    max=a;
    if (max<b)
        max=b;
    if(max<c)
        max=c;
    printf("Maximum=%d",max);
}
    
```

### if.....else স্টেটমেন্ট

একটা লজিক্যাল টেস্ট যদি সত্য হয়, তাহলে কিছু কাজ করে। আর যদি মিথ্যে হয়, তাহলে অন্য কাজ করে। এ লজিক থেকেই if else ব্যবহার।

if....else স্টেটমেন্ট ব্যবহারের ফরম্যাট হলো-

if(Expression)

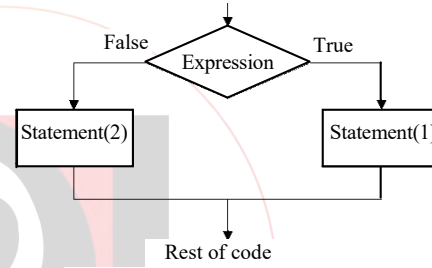
```

{
    Statement 1;
}
    
```

else

```

{
    Statement2;
}
    
```



চিত্র : if ..... else স্টেটমেন্টের ফ্লোচার্ট

if.... else কন্ট্রোলে ব্যবহৃত শর্ত (Expression) সাধারণত এক বা একাধিক লজিক্যাল বা রিলেশনাল এক্সপ্রেশন হয় যা if এর পরে প্রথম বন্ধনীর মধ্যে লেখা হয়। যদি Expression টি সত্য হয় তাহলে Statement1 কাজ করবে। আর যদি মিথ্যে হয় তাহলে Statement2 টি কাজ করবে। যেমন:

উদাহরণ-১. কোনো সংখ্যা জোড় (Even) না বিজোড় (Odd) নির্ণয়ের জন্য

অ্যালগরিদম, ফ্লোচার্ট ও প্রোগ্রাম লিখ।

অ্যালগরিদম:

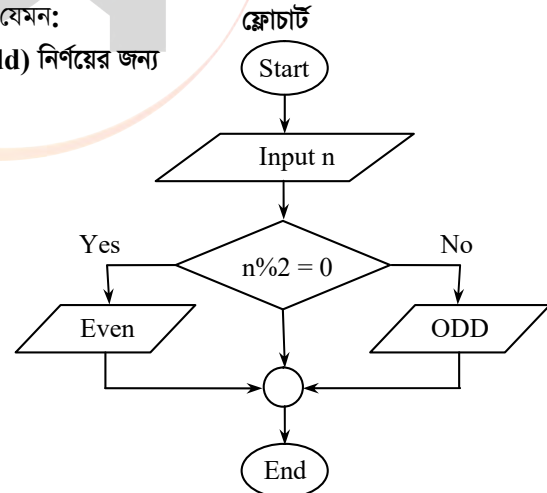
ধাপ-১: প্রোগ্রাম শুরু করি।

ধাপ-২: ইনপুট হিসেবে n এর মান গ্রহণ করি।

ধাপ-৩: যদি (n % 2 == 0) হয় তবে 'Even' ছাপাই, ৫নং ধাপে যাই।

ধাপ-৪: 'ODD' ছাপাই।

ধাপ-৫: প্রোগ্রাম শেষ করি।



তথ্য ও যোগাযোগ প্রযুক্তি (বোর্ড)-২৯ক

**Conditional statement ব্যবহার করে**

```
#include<stdio.h>
main()
{
    int n;
    scanf("%d",&n);
    if (n%2==0)
        printf("Even number.");
    else
        printf("Odd number.");
}
```

**Conditional operator ব্যবহার করে**

```
#include<stdio.h>
main()
{
    int n;
    scanf("%d",&n);
    (n%2==0)?printf("Even");printf("Odd");
}
```

উদাহরণ-২. কোনো সংখ্যা ধনাত্মক (Positive) না ঋণাত্মক (Negative) তা নির্ণয়ের জন্য অ্যালগরিদম, ফ্লোচার্ট ও প্রোগ্রাম লিখ।

অ্যালগরিদম:

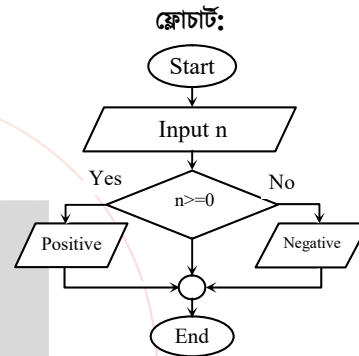
ধাপ-১: প্রোগ্রাম শুরু করি।

ধাপ-২: ইনপুট হিসেবে n এর মান গ্রহণ করি।

ধাপ-৩: যদি  $(n \geq 0)$  হয় তবে 'Positive' ছাপাই,  
নৈনং ধাপে যাই।

ধাপ-৪: 'Negative' ছাপাই।

ধাপ-৫: প্রোগ্রাম শেষ করি।

**Conditional statement ব্যবহার করে**

```
#include<stdio.h>
main()
{
    int n;
    scanf("%d",&n);
    if (n>=0)
        printf("Positive");
    else
        printf("Negative");
}
```

**Conditional operator ব্যবহার করে**

```
#include<stdio.h>
main()
{
    int n;
    scanf("%d",&n);
    (n>=0)?printf("Positive");printf("Negative");
}
```

উদাহরণ-৩. if-else ব্যবহার করে দুটি অসমান সংখ্যার মধ্যে বৃহত্তম সংখ্যা নির্ণয়ের জন্য অ্যালগরিদম, ফ্লোচার্ট ও প্রোগ্রাম লিখ।

অ্যালগরিদম:

ধাপ-১: প্রোগ্রাম শুরু করি।

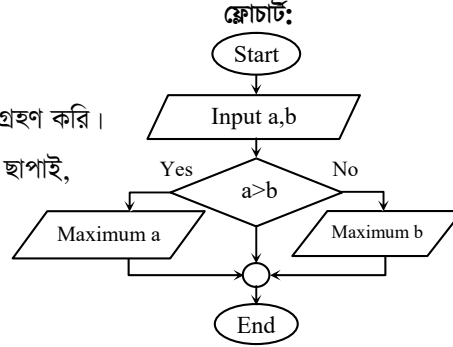
ধাপ-২: ইনপুট হিসেবে a,b এর মান গ্রহণ করি।

ধাপ-৩: যদি  $(a > b)$  হয় তবে 'a বড়' ছাপাই,

৫নং ধাপে যাই।

ধাপ-৪: 'b বড়' ছাপাই।

ধাপ-৫: প্রোগ্রাম শেষ করি।



প্রোগ্রাম:

```

#include<stdio.h>
main()
{
    int a,b;
    scanf("%d%d",&a,&b);
    if (a>b)
        printf("Max=%d",a);
    else
        printf("Max=%d",b);
}
  
```

উদাহরণ-৪. if-else ব্যবহার করে কোনো বর্ষ অধিবর্ষ (Leap year) কি না তা নির্ণয়ের জন্য অ্যালগরিদম, ফ্লোচার্ট ও প্রোগ্রাম লিখ।

অ্যালগরিদম:

ধাপ-১: প্রোগ্রাম শুরু করি।

ধাপ-২: ইনপুট হিসেবে y এর মান গ্রহণ করি।

ধাপ-৩: যদি  $(y \% 400 == 0) \parallel ((y \% 100 != 0) \&\& (y \% 4 == 0))$  হয়

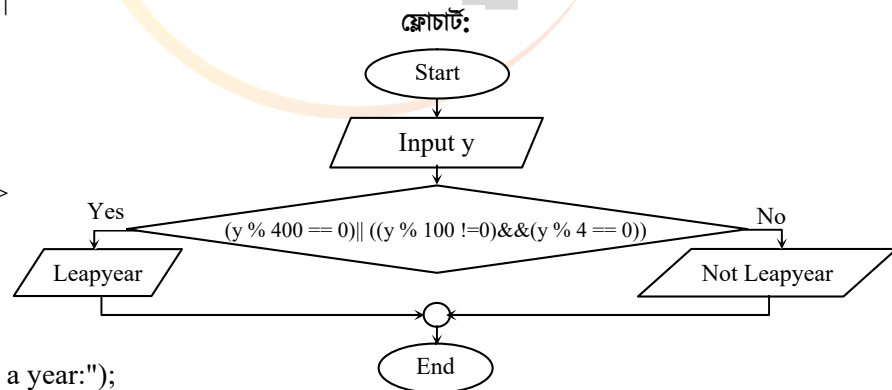
তাহলে ৪নং ধাপে যাই।

অন্যথায় ৫নং ধাপে যাই।

ধাপ-৪: LEAP YEAR ছাপাই এবং ৭ নং ধাপে গমন।

ধাপ-৫: NOT LEAP YEAR ছাপাই।

ধাপ-৭: প্রোগ্রাম শেষ।



প্রোগ্রাম:

```

#include<stdio.h>
  
```

```

main()
  
```

```

{
  
```

```

    int y;
  
```

```

    printf("\n Enter a year:");
  
```

```

    scanf("%d",&y);
  
```

```

if ((y % 400 == 0) || ((y % 100 != 0) && (y % 4 == 0)))
    printf("\n %d is a Leap year", y);
else
    printf("\n %d is not a Leap year", y);
}

```

### Nested if.... else স্টেটমেন্ট

একটি if.... else স্টেটমেন্টের মধ্যে অপর একটি if.... else স্টেটমেন্ট থাকতে পারে। এরূপ মধ্যবর্তী if.... else স্টেটমেন্টকে Nested if.... else স্টেটমেন্ট বলা হয়। নিচে ন্যেস্টেড if.... else এর সিনটেক্স দেওয়া হলো।

```

if (Condition1)

```

```

{

```

```

    if(Condition2)

```

```

        Action1;

```

```

    else

```

```

        Action2;

```

```

}

```

```

else

```

```

{

```

```

    if(Condition3)

```

```

        Action3;

```

```

    else

```

```

        Action4;

```

```

}

```

উদাহরণ: Nested if-else ব্যবহার করে তিনটি অসমান সংখ্যার মধ্যে বড় সংখ্যা নির্ণয়ের অ্যালগরিদম ও ফ্লোচার্ট।

অ্যালগরিদম:

ধাপ-১: প্রোগ্রাম শুরু।

ধাপ-২: তিনটি সংখ্যা a, b এবং c এর মান গ্রহণ।

ধাপ-৩: যদি  $a > b$  সত্য হয় তাহলে ৪ নং ধাপে যেতে হবে

অন্যথায় ৫ নং ধাপে যেতে হবে।

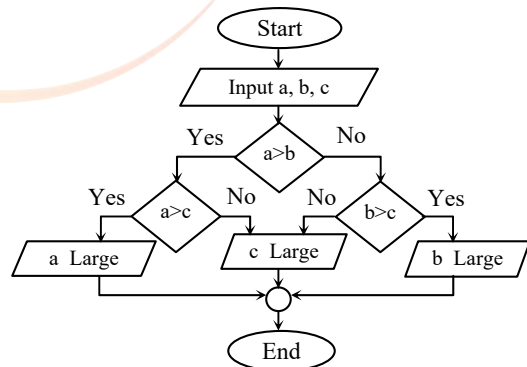
ধাপ-৪: যদি  $a > c$  সত্য হয় তাহলে a বড়।

অন্যথায় c বড় এবং ৬নং ধাপে যেতে হবে।

ধাপ-৫: যদি  $b > c$  সত্য হয় তাহলে b বড়।

অন্যথায় c বড়।

ফ্লোচার্ট:





ধাপ-৬: প্রোগ্রাম শেষ।

প্রোগ্রাম:

```
#include<stdio.h>
main()
{
    int a,b,c;
    scanf("%d %d %d",&a,&b,&c);
    if (a > b)
    {
        if(a > c)
            printf("Maximum: %d", a);
        else
            printf("Maximum: %d", c);
    }
    else
    {
        if(b > c)
            printf("Maximum:%d", b);
        else
            printf("Maximum:%d", c);
    }
}
```

মাত্র একবার printf() ব্যবহার করে উপরোক্ত প্রোগ্রামটি হবে নিম্নরূপ:

ধাপ-১: প্রোগ্রাম শুরু।

ধাপ-২: তিনটি সংখ্যা a, b এবং c এর মান গ্রহণ।

ধাপ-৩: যদি  $a > b$  সত্য হয় তাহলে ৪ নং ধাপে যেতে হবে

অন্যথায় ৫ নং ধাপে যেতে হবে।

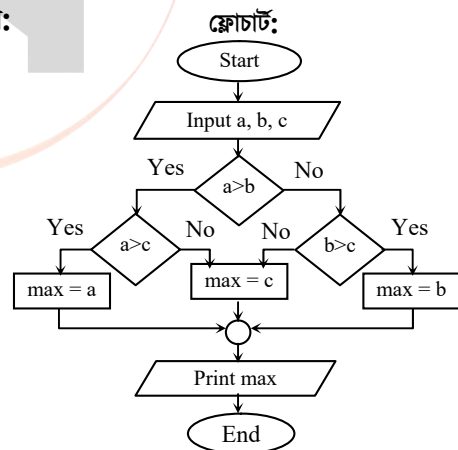
ধাপ-৪: যদি  $a > c$  সত্য হয় তাহলে  $\text{max} = a$ ।

অন্যথায়  $\text{max} = c$  এবং ৬ নং ধাপে যেতে হবে।

ধাপ-৫: যদি  $b > c$  সত্য হয় তাহলে  $\text{max} = b$ ।

অন্যথায়  $\text{max} = c$  এবং ৬ নং ধাপে যেতে হবে।

ধাপ-৬: ফলাফলে max এর মান ছাপাতে হবে।



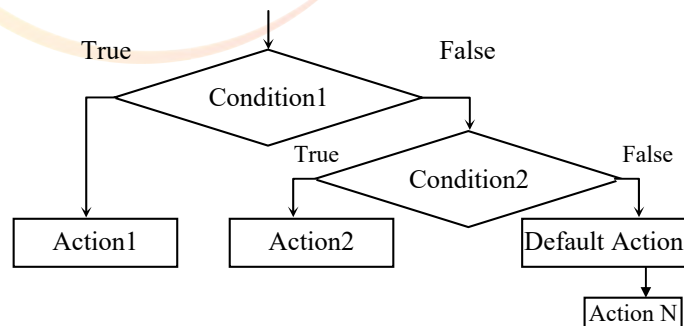
ধাপ-৭: প্রোগ্রাম শেষ।

```
#include<stdio.h>
main()
{
    int a,b,c,max;
    scanf("%d %d %d",&a,&b,&c);
    if (a > b)
    {
        if(a > c)
            max=a;
        else
            max=c;
    }
    else
    {
        if(b > c)
            max=b;
        else
            max=c;
    }
    printf("Maximum=%d",max);
}
```

#### if.....else if....else স্টেটমেন্ট

প্রোগ্রামে একাধিক শর্ত যাচাই করার জন্য if...else if...else স্টেটমেন্ট ব্যবহার করা হয়। 'সি' প্রোগ্রামে “অন্যথায় যদি” অর্থে if...else স্টেটমেন্টের সাথে else if স্টেটমেন্ট ব্যবহার হয়। else if স্টেটমেন্ট if...else স্টেটমেন্টের if এবং else স্টেটমেন্টের মাঝে বসে। if ও else স্টেটমেন্টের মাঝে একাধিক else if স্টেটমেন্ট থাকতে পারে। else if স্টেটমেন্ট ব্যবহারের ফরম্যাট হলো—

```
if (Condition 1)
{
    Action1;
}
else if (Condition 2)
{
    Action 2;
}
.....
```



চিত্র : else if স্টেটমেন্টের ফ্লোচার্ট

```
else
{
    Default Action ;
}
```

Action N;

যদি ১ম শর্তটি সত্য হয় তাহলে Action1 এক্সিকিউট করে। যদি ১ম শর্তটি মিথ্যা হয় তাহলে ২য় শর্তটি চেক করে এবং যদি ২য় শর্তটি সত্য হয় তাহলে Action 2 এক্সিকিউট করে। এভাবেই এই প্রক্রিয়াটি চলতে থাকে। যদি সবগুলো শর্ত মিথ্যা হয় তাহলে else এর Default Action এক্সিকিউট করে। যেমন:

**উদাহরণ-১: if-else if-else ব্যবহার করে তিনটি সংখ্যার মধ্যে বড় সংখ্যা নির্ণয়ের প্রোগ্রাম:**

**অ্যালগরিদম:**

ধাপ-১: প্রোগ্রাম শুরু।

ধাপ-২: তিনটি সংখ্যা a, b এবং c এর মান গ্রহণ।

ধাপ-৩: প্রথম সংখ্যাটি কি দ্বিতীয় ও তৃতীয় সংখ্যার চেয়ে বড়?

ক. হ্যাঁ, a বড় এবং ৬নং ধাপে যাও।

খ. না, ৪নং ধাপে যাও।

ধাপ-৪: দ্বিতীয় সংখ্যাটি কি তৃতীয় সংখ্যার চেয়ে বড়?

ক. হ্যাঁ, b বড় এবং ৬নং ধাপে যাও।

খ. না, ৫নং ধাপে যাও।

ধাপ-৫: c বড় এবং ৬নং ধাপে যাও।

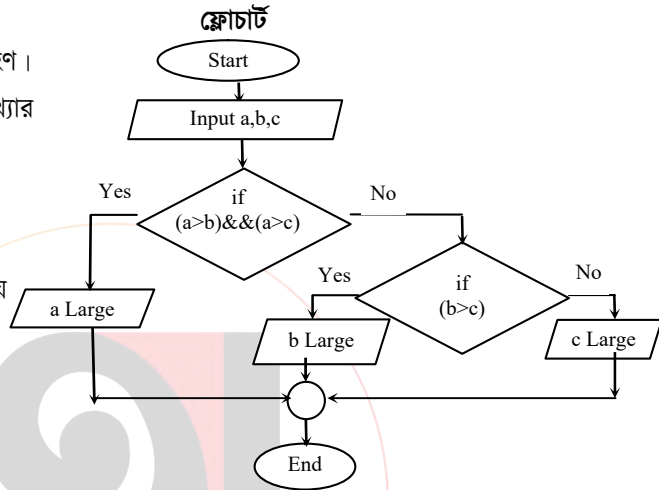
ধাপ-৬: প্রোগ্রাম শেষ।

**প্রোগ্রাম:**

```
#include<stdio.h>
```

```
main()
```

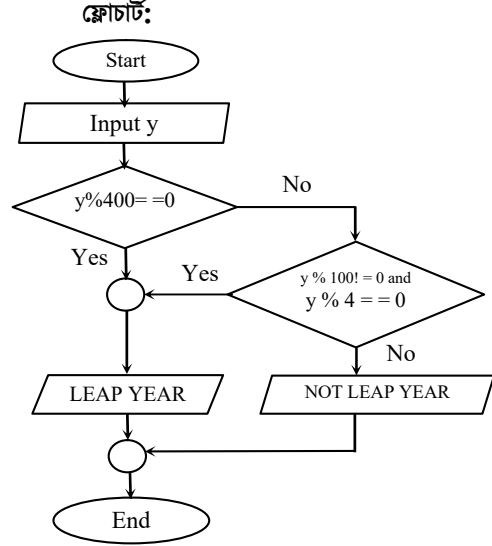
```
{
    int a,b,c;
    scanf("%d %d %d", &a, &b, &c);
    if ((a > b) && (a > c))
        printf("\n Largest Value is: %d", a);
    else if (b > c)
        printf("\n Largest Value is: %d", b);
    else
        printf("\n Largest Value is: %d", c);
}
```



উদাহরণ-২: if-else if-else ব্যবহার করে কোনো সাল লিপইয়ার কি-না তা নির্ণয়ের জন্য প্রোগ্রাম:

অ্যালগরিদম:

- ধাপ-১: প্রোগ্রাম শুরু করি।  
 ধাপ-২: ইনপুট হিসেবে y এর মান গ্রহণ করি।  
 ধাপ-৩: যদি  $(y \% 400 == 0)$  তবে ৬ নং ধাপে যাই।  
 ধাপ-৪: যদি  $(y \% 100 != 0)$  এবং  $y \% 4 == 0$  পাই,  
 তবে ৬ নং ধাপে যাই।  
 ধাপ-৫: NOT LEAP YEAR ছাপাই এবং ৭নং ধাপে গমন।  
 ধাপ-৬: LEAP YEAR ছাপাই।  
 ধাপ-৭: প্রোগ্রাম শেষ করি।



প্রোগ্রাম:

```
#include<stdio.h>
```

```
main()
```

```
{
```

```
    int y;
```

```
    printf("\n Enter a year:");
```

```
    scanf("%d",&y);
```

```
    if (y%400==0)
```

```
        printf("\n %d is a Leap year", y);
```

```
    else if ((y%100 != 0)&&(y%4==0))
```

```
        printf("\n %d is a Leap year", y);
```

```
    else
```

```
        printf("\n %d is not a Leap year", y);
```

```
}
```

**switch স্টেটমেন্ট**

একাধিক স্টেটমেন্ট থেকে নির্দিষ্ট কোনো স্টেটমেন্ট সম্পাদনের জন্য switch স্টেটমেন্ট ব্যবহার করা হয়। মূলত বেশি সংখ্যক else if স্টেটমেন্ট ব্যবহারের পরিবর্তে switch স্টেটমেন্ট ব্যবহৃত হয়। যখন অনেকগুলো if-else থাকে তখন if-else স্টেটমেন্ট ব্যবহার করার চেয়ে switch স্টেটমেন্ট ব্যবহার করাই উত্তম। switch স্টেটমেন্ট-এর সাথে

অতিরিক্ত case, break ও default স্টেটমেন্ট ব্যবহার হয়। else if স্টেটমেন্টে কোনো কন্ডিশনাল কিংবা রিলেশনাল এক্সপ্রেশনের ওপর ভিত্তি করে উপযুক্ত স্টেটমেন্ট নির্বাচন করা হয়। কিন্তু switch স্টেটমেন্টে সাধারণত কোনো বৈধ ভেরিয়েবলের মানের ভিত্তিতে উপযুক্ত স্টেটমেন্ট নেওয়া হয়। switch স্টেটমেন্ট-এর ফরম্যাট হলো-

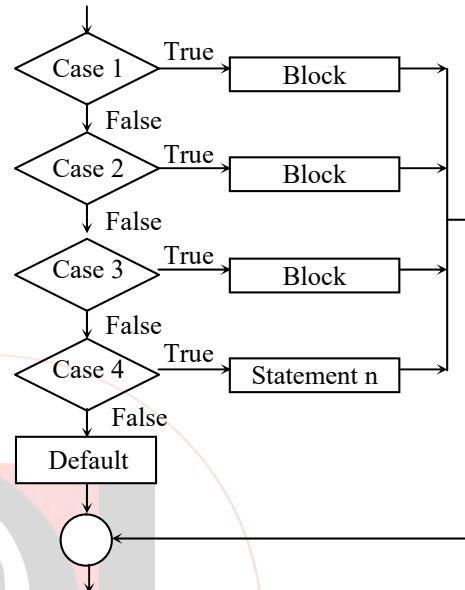
Data-Type Index Variable;

switch (expression)

```
{
case 1:
    Block 1;
    break;
case 2:
    Block 2;
    break;
case 3:
    Block 3;
    break;
....
....
case N :
    Block N;
    break;
default:
    Default Block;
}
```

switch স্টেটমেন্টে 4টি কী-ওয়ার্ড থাকে। যথা:

- switch
- case
- break
- default



চিত্র : Switch স্টেটমেন্টের ফ্লোচার্ট

**switch-** switch স্টেটমেন্টের মাধ্যমে কম্পাইলারকে নির্দেশ দেয়া হয় কোন জায়গা থেকে switch স্টেটমেন্ট এর কাজ শুরু করতে হবে।

**case-** switch কে যদি if এর সাথে তুলনা করতে হয় তবে case কে else if এর সাথে তুলনা করা যায়। n এর যে মানের সাথে যে case এর constant ভ্যালুর সাথে মিলে যাবে তার স্টেটমেন্ট গুলো ফলাফলে দেখাবে। ANSI স্ট্যান্ডার্ড অনুযায়ী switch() স্টেটমেন্টে মোট 256টি case স্টেটমেন্ট ব্যবহৃত হতে পারে।

**break-case** এর সাথে যে সকল স্টেটমেন্ট থাকে সেগুলো এক্সিকিউট হওয়ার পর break কী-ওয়ার্ড পেলে কম্পাইলার **switch** থেকে বের হয়।

**default-** switch এর ভিতর কোন case এর সাথে শর্ত না মিললে default এর ভিতরের স্টেটমেন্ট গুলো ফলাফল দেখাবে।

বিষয়টি একটু জটিল, তবে একটি উদাহরণ দেখলে অনেক সহজ হয়ে যাবে। আমরা একটা প্রোগ্রাম লিখব, যেখানে যদি আমরা r ইনপুট দিই, তাহলে লেখা উঠবে You select Red, যদি w ইনপুট দিই, তাহলে লেখা উঠবে You select White. যদি b ইনপুট দিই, তাহলে লেখা উঠবে You select Black. আর প্রোগ্রামটা লিখব আমরা switch case ব্যবহার করে।

```
#include<stdio.h>
int main()
{
    char colorCode;
    printf("Enter first word of Red, White or Black: \n");
    scanf("%c", &colorCode);
    switch ( colorCode ) {
        case 'r' :
            printf("You select Red.");
            break;
        case 'w':
            printf("You select White.");
            break;
        case 'b':
            printf("You select Black.");
            break;
        default:
            printf("Wrong choose!");
            break;
    }
    return 0;
}
```

উপরের প্রোগ্রামটি রান করি, তারপর r, w, b এ তিনটার মধ্যে যেকোনো একটি ইনপুট দিলে কালার দেখাবে। আর যদি অন্য কোনো ক্যারেক্টার ইনপুট দিই, তাহলে লেখা উঠবে Wrong choose!

**উদাহরণ: কোনএকটি অক্ষর বা লেটার vowel না consonant তা নির্ণয়ের প্রোগ্রাম।**

<pre>#include&lt;stdio.h&gt; void main() {     char ch;     printf("Enter any letter :");     ch=getchar();     switch(ch)     {         case 'a':</pre>	<p>অথবা</p> <pre>#include&lt;stdio.h&gt; void main() {     char ch;     printf("\n Enter any letter :");     ch=getchar();     switch(ch)</pre>
--	---



<pre> printf("\n The letter is a vowel."); break; case 'A': printf("\n The letter is a vowel."); break; case 'e': printf("\n The letter is a vowel."); break; case 'E': printf("\n The letter is a vowel."); break; case 'i': printf("\n The letter is a vowel."); break; case 'I': printf("\n The letter is a vowel."); break; case 'o': printf("\n The letter is a vowel."); break; case 'O': printf("\n The letter is a vowel."); break; case 'u': printf("\n The letter is a vowel."); break; case 'U': printf("\n The letter is a vowel."); break; default: printf("\n The letter is a consonant."); } } </pre>	<pre> { case 'a': case 'A': case 'e': case 'E': case 'i': case 'I': case 'o': case 'O': case 'u': case 'U': printf("\n The letter is a vowel."); break; default: printf("\n The letter is a consonant."); } } </pre> <p><b>ফলাফল:</b> Enter any letter: R The letter is a consonant.</p>
<p><b>ফলাফল:</b> Enter any letter: U The letter is a vowel.</p>	



**কাজ:**

১. জোড় বিজোড় সংখ্যা নির্ণয়ের জন্য প্রোগ্রাম লেখ।
২. একটি সালের ফেব্রুয়ারি মাস কত দিনে তা নির্ণয়ের জন্য প্রোগ্রাম লেখ।
৩. কোনো বিষয়ের প্রাপ্ত নম্বর থেকে গ্রেড নির্ণয়ের জন্য প্রোগ্রাম লেখ।

### ৫.১২.১৫ লুপ স্টেটমেন্ট (Loop Statement)

প্রোগ্রামের অংশ বিশেষ প্রদত্ত শর্তে না পৌঁছা পর্যন্ত নির্দিষ্ট সংখ্যকবার প্রদত্ত পুনরাবৃত্তি করাকে লুপিং বা চক্র নিয়ন্ত্রণ বলা হয়। লুপকে তিন ভাগে ভাগ করা হয়। যথা—

১. **অসীম লুপ (Endless Loop):** যদি কোনো লুপ অনবরত আবর্তন হতে থাকে, কখনো শেষ না হয় তবে তাকে অসীম লুপ বলে।
২. **সসীম লুপ (Finite Loop):** নির্দিষ্ট সংখ্যক আবর্তনের পর যে লুপ শেষ হয় তাকে সসীম লুপ বলে।
৩. **মধ্যবর্তী লুপ (Nested Loop):** একটি লুপের মধ্যে যদি আর একটি লুপ থাকে তাহলে তাকে মধ্যবর্তী লুপ (Nested Loop) বলে।

কোনো স্টেটমেন্টকে দুই বা ততোধিক বার সম্পাদনের জন্য যে সকল স্টেটমেন্ট ব্যবহৃত হয় তাকে লুপ কন্ট্রোল স্টেটমেন্ট বলে। লুপ স্টেটমেন্টসমূহ সাধারণত দুইটি অংশ থাকে। যথাঃ

- লুপ বডি (Loop Body) এবং
- টেস্ট কন্ডিশন (Test Condition)

লুপ স্টেটমেন্টের লুপ বডি ও টেস্ট কন্ডিশনের অবস্থানের ভিত্তিতে লুপ স্টেটমেন্টসমূহকে দুই ভাগে ভাগ করা হয়। যথা—

**এন্ট্রি কন্ট্রোল লুপ (Entry Control Loop) :** এন্ট্রি কন্ট্রোল লুপে লুপ বডির নির্বাহ শুরুর আগেই টেস্ট কন্ডিশন যাচাই করা হয়। কন্ডিশন সত্য না হলে লুপ বডি সম্পাদিত হয় না। এন্ট্রি কন্ট্রোল লুপ নির্বাহের জন্য প্রধানত দুইটি স্টেটমেন্ট ব্যবহার করা হয়। সেগুলো হচ্ছে- for লুপ স্টেটমেন্ট, while লুপ স্টেটমেন্ট।

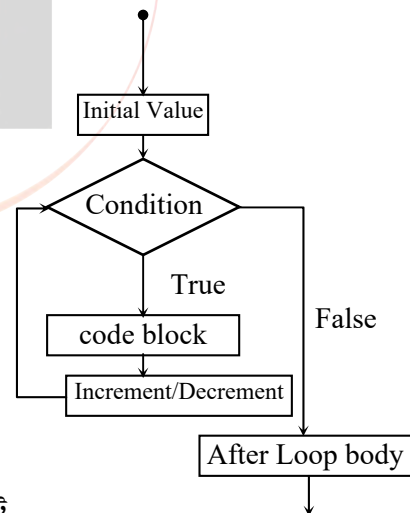
**এক্সিট কন্ট্রোল লুপ (Exit Control Loop):** এক্সিট কন্ট্রোল লুপে প্রথমে একবার লুপ নির্বাহ হয়। তারপর টেস্ট কন্ডিশন যাচাই করা হয়, কন্ডিশন সত্য হলে লুপ বডি সম্পাদিত হয়, কন্ডিশন সত্য না হলে লুপ বডি সম্পাদিত হয় না। এক্সিট কন্ট্রোল লুপের স্টেটমেন্ট হলো, do – while লুপ স্টেটমেন্ট

লুপ স্টেটমেন্ট নির্বাহের জন্য প্রধান বিবেচ্য বিষয়সমূহ হলোঃ

- কাউন্টার ভেরিয়েবল স্থাপন ও তার প্রারম্ভিক মান নির্ধারণ
- লুপ বডির স্টেটমেন্ট নির্বাহ
- লুপ বডির পরবর্তী নির্বাহ না হওয়ার জন্য শর্ত পরীক্ষা
- কাউন্টার ভেরিয়েবলের ইনক্রিমেন্ট বা ডিক্রিমেন্ট

#### for স্টেটমেন্ট

‘সি’ প্রোগ্রামে কোনো স্টেটমেন্ট দুই বা ততোধিকবার সম্পাদনের জন্য for স্টেটমেন্ট ব্যবহার করা হয়। সাধারণ কোনো ভেরিয়েবল ব্যবহার করে for লুপের আবর্তন সংখ্যা গণনা হয়। এরূপ ভেরিয়েবলকে কাউন্টার ভেরিয়েবল বলে। for স্টেটমেন্ট-এর ফরম্যাট দেখানো হলো—



চিত্র : for স্টেটমেন্টের ফ্লোচার্ট

```
Counter Declaration;
for (InitialValue; Condition; Decrement/Increment)
{
    Statement(s);
}
```

- **Counter Declaration** অংশে উপযুক্ত ডেটাইপসহ কাউন্টার ভেরিয়েবল ঘোষণা করা হয়।
- For Loop-এর শুরুতেই **InitialValue** স্টেটমেন্ট শুধু একবার এক্সিকিউট হয়।
- তারপর **Condition** এক্সিকিউট হয়। ইহা False হলে for লুপের সমাপ্তি ঘটে। কিন্তু Condition যদি True হয় তাহলে for লুপের কোডব্লক এক্সিকিউট হয় এবং **Decrement/Increment** এর ভ্যালু আপডেট হয়।
- Condition মিথ্যা (False) না হওয়া পর্যন্ত কাজ চালিয়ে যায়।

**নোট:** ইটারেশন (Iteration) সংখ্যা বা কতবার লুপ চলবে সেই সংখ্যা আগে থেকে জানা থাকলে সাধারণত for লুপ ব্যবহৃত হয়।

<p><b>উদাহরণ-১:</b> ১ ২ ৩ ৪ ৫ সংখ্যা গুলো প্রদর্শন করার প্রোগ্রাম লিখ।</p> <pre>#include&lt;stdio.h&gt; int main(void) {     int i;     for(i=1;i&lt;=5;i++)     printf("%d\t",i);     return 0; }</pre> <p><b>আউটপুট :</b></p> <pre>1      2      3      4      5</pre>	<p><b>উদাহরণ-২:</b> ১ ২ ৪ ৮ ১৬ সংখ্যাগুলো প্রদর্শন করার প্রোগ্রাম লিখ।</p> <pre>#include&lt;stdio.h&gt; int main(void) {     int i;     for(i=1;i&lt;=16;i=i*2)     printf("%d\t",i);     return 0; }</pre> <p><b>আউটপুটঃ</b></p> <pre>1      2      4      8      16</pre>
<p><b>উদাহরণ-৩:</b> <math>1+2+3+\dots+n</math> ধারার যোগফল নির্ণয়ের জন্য প্রোগ্রাম।</p> <pre>#include&lt;stdio.h&gt; main() {     int n,a,s=0;     printf("Enter a last term: ");     scanf("%d",&amp;n);     for(a=1;a&lt;=n;++a)     {         s +=a; // or, s=s+a;     }     printf("Sum=%d",s); }</pre> <p><b>আউটপুট:</b></p> <pre>Enter a last term: 10 Sum=55</pre>	<p><b>উদাহরণ-৪:</b> <math>1+2+4+8+\dots+n</math> ধারার যোগফল নির্ণয়ের জন্য প্রোগ্রাম।</p> <pre>#include&lt;stdio.h&gt; main() {     int s=0,n,a;     printf("Enter a last term: ");     scanf("%d",&amp;n);     for(a=1;a&lt;=n;a=a*2)     {         s=s+a;     }     printf("%d ",s); }</pre> <p><b>আউটপুট:</b></p> <pre>Enter a last term: 64 127</pre>

<p><b>উদাহরণ-৫:</b> 1 2 4 7 11 16 সংখ্যাগুলো প্রদর্শন করার প্রোগ্রাম লিখ।</p> <pre>#include&lt;stdio.h&gt; main() {     int i,j;     for(i=1,j=1;j&lt;=100;i++)     {         printf("%d\t",j);         j=j+i;     } }</pre> <p><b>আউটপুট:</b></p> <p>1      2      4      7      11      16</p>	<p><b>উদাহরণ-৬:</b> 1+2+4+7+11+16+.....+100 ধারার যোগফল নির্ণয়ের জন্য প্রোগ্রাম লিখ।</p> <pre>#include&lt;stdio.h&gt; main() {     int i,j,sum=0;     for(i=1,j=1;j&lt;=100;i++)     {         sum=sum+j;         j=j+i;     }     printf("SUM=%d",sum); }</pre> <p><b>আউটপুট:</b></p> <p>469</p>
--	--

<p><b>উদাহরণ-৭:</b> 3+7+11+ .....+n ধারাটির 30 টি পদের যোগফল নির্ণয়ের জন্য সি ভাষায় প্রোগ্রাম লিখ।</p> <pre>#include&lt;stdio.h&gt; main() {     int s=0,a,b;     b=1;     for(a=3;b&lt;=30;a+=4)     {         s=s+a;         b=b+1;     }     printf("%ld",s); }</pre> <p><b>আউটপুট:</b></p> <p>1830</p>	<p><b>উদাহরণ-৮:</b> 3+9+27+.....+n ধারাটির 30 টি পদের যোগফল নির্ণয়ের জন্য সি ভাষায় প্রোগ্রাম লিখ।</p> <pre>#include&lt;stdio.h&gt; main() {     int s=0,a,b;     for(a=3;a&lt;=100;a=a*3)     {         s=s+a;     }     printf("\nSum=%d",s); }</pre> <p><b>আউটপুট:</b></p> <p>Sum=120</p>
--	---

উপরের উদাহরণ গুলোতে for loop এর control variable এর মান কেবল int টাইপ ব্যবহার করা হয়েছে। তবে এখানে প্রয়োজন অনুসারে অন্য টাইপের ভেরিয়েবল নিয়েও কাজ করা যায়।

#### উদাহরণ-৬:

```
#include<stdio.h>
main()
{
    char ch;
    for( ch='a'; ch<='z';ch++ )
    {
        printf("%c ",ch);
    }
}
```

**ফলাফল :** a b c d e f g h i j k l m n o p q r s t u v w x y z

### নেস্টেড ফর লুপ (Nested for loop)

for loop কে অন্য for loop এর compound statement হিসাবেও ব্যবহার করা যায়। এ ধরনের for loop কে Nested for loop বলে। প্রোগ্রামে অনেক সময় এ ধরনের স্টেটমেন্ট প্রয়োজন হয়। এ ধরনের স্টেটমেন্ট ব্যবহারের নিয়ম হলো-

```
for (Initialization; Condition; Increment/Decrement)
{
    for(Initialization; Condition; Increment/Decrement)
    {
        Statements;
    }
}
```

<p><b>উদাহরণ :</b></p> <pre>#include&lt;stdio.h&gt; main() {     int i,j,n;     printf("Enter how many line you need to make pyramid = ");     scanf("%d",&amp;n);     for(i=1;i&lt;=n;i++)     {         for(j=1;j&lt;=i;j++)         printf("%d\t",j);         printf("\n");     } }</pre> <p><b>ফলাফল :</b></p> <pre>Enter how many line you need to make pyramid =4 1 1 2 1 2 3 1 2 3 4</pre>	<p><b>উদাহরণ :</b></p> <pre>#include&lt;stdio.h&gt; main() {     int i,j,n;     printf("Enter how many line you need to make pyramid = ");     scanf("%d",&amp;n);     for(i=1;i&lt;=n;i++)     {         for(j=1;j&lt;=i;j++)         printf("%d\t",i);         printf("\n");     } }</pre> <p><b>ফলাফল :</b></p> <pre>Enter how many line you need to make pyramid =4 1 2 2 3 3 3 4 4 4 4</pre>
---	---



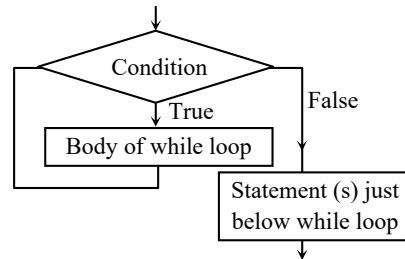
**কাজ:** নিচের প্রোগ্রামগুলোর আউটপুট লিখো:

<pre>main() {     int i,j;     for(i=1,j=2;i&lt;=5;i++)     printf("%d %d\n",i,j); }</pre>	<pre>main() {     int i,j;     for(i=1,j=2;i&lt;=5;i++,j++)     printf("%d %d\n",i,j); }</pre>	<pre>main() {     int i,j;     for(i=1,j=2;i&lt;=5;i++,j++)     printf("%d %d\n",i,j); }</pre>
--	--	--

### while স্টেটমেন্ট

‘সি’ প্রোগ্রামে শর্ত সাপেক্ষে দুই বা ততোধিকবার কোনো স্টেটমেন্ট সম্পাদনের জন্য while স্টেটমেন্ট ব্যবহার করা হয়। while লুপে প্রথমেই শর্তের মান পরীক্ষিত হয়, শর্ত সত্য হলে তবেই লুপ বডি সম্পাদিত হয়। এটি অনেকটা for স্টেটমেন্ট-এর বিকল্প হিসেবে ব্যবহার করা হয়। for স্টেটমেন্টের মতো পূর্বে ঘোষিত কোনো কাউন্টার ভেরিয়েবল ব্যবহার করে while স্টেটমেন্ট-এর আবর্তন সংখ্যা গণনা করা হয়। while স্টেটমেন্ট-এর ফরম্যাট হলো—

```
Counter Declaration;
Counter Initialization;
while (Condition)
{
    Statement(s);
    Increment/ Decrement;} Loop body
}
```



চিত্র : While স্টেটমেন্টের ফ্লোচার্ট

Counter Declaration অংশে উপযুক্ত ডেটাইপসহ ভেরিয়েবল ঘোষণা হয়, Counter Initialization অংশে কাউন্টার ভেরিয়েবলের প্রারম্ভিক মান দেওয়া হয়।

- while loop এর কাজ হলো প্রোগ্রামের Condition চেক করা, কন্ডিশনটি সত্য না মিথ্যা।
- যদি Condition সত্য হয় তাহলে এটা while loop এর বডির ভিতরের কোড গুলো এক্সিকিউট করবে। তারপর আবার এটি টেস্ট কন্ডিশন চেক করবে যে, এটি সত্য না মিথ্যা।
- এই প্রক্রিয়াটি ততক্ষণ চলবে টেস্ট কন্ডিশনটি যতক্ষণ পর্যন্ত মিথ্যা না হবে।

উদাহরণ-১: ০ থেকে ৩ পর্যন্ত সংখ্যা প্রদর্শনের জন্য প্রোগ্রাম লিখ।

```
#include<stdio.h>
main()
{
1   int x;
2   x=0;
3
4   while(x<4)
5   {
6       printf("%d\n",x);
7       x=x+1;
8   }
9 }
```

আউটপুট:

0  
1  
2  
3



উদাহরণ-২: দুটি সংখ্যার গ.সা.গু(GCD) নির্ণয়ের জন্য অ্যালগরিদম, ফ্লোচার্ট ও প্রোগ্রাম লিখ।

অ্যালগরিদম:

ধাপ-১: প্রোগ্রাম শুরু।

ধাপ-২: দুটি সংখ্যা L, S ( $L > S$ ) ইনপুট নিই।

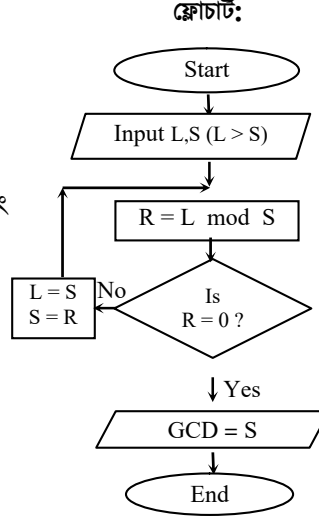
ধাপ-৩: ছোটো সংখ্যাটি (S) দিয়ে বড় সংখ্যাটিকে (L) ভাগ করে ভাগশেষ (R) নির্ণয় করি।

ধাপ-৪: ভাগশেষের মান (R) যদি 0 হয় তবে ৫ নং ধাপে গমন, অন্যথায়, নতুনভাবে  $L = S$  এবং  $S = R$  করে পুনরায় ৩নং ধাপে গমন।

ধাপ-৫: নির্ণেয় গ.সা.গু হবে ছোটো সংখ্যাটি (S)।

ধাপ-৬: প্রোগ্রাম শেষ।

ফ্লোচার্ট:



প্রোগ্রাম:

```

#include<stdio.h>
main()
{
    int l, s, r;
    scanf("%d %d", &l,&s);
    while(l%s!=0)
    {
        r = l%s;
        l = s;
        s = r;
    }
    printf("GCD=%d", s);
}
    
```

উদাহরণ-৩: দুটি সংখ্যার ল.সা.গু নির্ণয়ের জন্য অ্যালগরিদম, ফ্লোচার্ট ও প্রোগ্রাম।

অ্যালগরিদম:

ধাপ-১: প্রোগ্রাম শুরু

ধাপ-২: দুটি সংখ্যা L, S ( $L > S$ ) ইনপুট নিই।

ধাপ-৩:  $a=L$ ,  $b=S$  নিই।

ধাপ-৪: ছোটো সংখ্যাটি (S) দিয়ে বড় সংখ্যাটিকে (L) ভাগ করে ভাগশেষ (R) নির্ণয় করি।

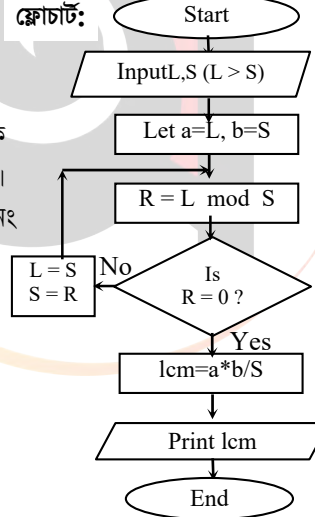
ধাপ-৫: ভাগশেষের মান (R) যদি 0 হয় তবে ৬ নং ধাপে গমন, অন্যথায়, নতুন ভাবে  $L = S$  এবং  $S = R$  করে পুনরায় ৩নং ধাপে গমন।

ধাপ-৬: নির্ণেয় ল.সা.গু  $lcm=a*b/s$  নির্ণয়।

ধাপ-৭: lcm এর মান প্রদর্শন।

ধাপ-৮: প্রোগ্রাম শেষ।

ফ্লোচার্ট:



প্রোগ্রাম:

```

#include<stdio.h>
main()
{
    int l, s, r, a, b, lcm;
    scanf("%d %d", &l,&s);
    a=l;
    b=s;
    while(l%s!=0)
    {
        r = l%s;
        l = s;
        s = r;
    }
    lcm=a*b/s;
    printf("LCM=%d", lcm);
}
    
```

[ বি.দ্র: ল.সা.গু = সংখ্যাদুটির গুণফল/ গ.সা.গু ]

তথ্য ও যোগাযোগ প্রযুক্তি (বোর্ড)-৩০ক

### do...while স্টেটমেন্ট

‘সি’ প্রোগ্রামে শর্ত সাপেক্ষে এক বা একাধিকবার কোনো স্টেটমেন্ট সম্পাদনের জন্য do...while স্টেটমেন্ট ব্যবহার করা হয়। while স্টেটমেন্টের মতো কোনো পূর্ব ঘোষিত কাউন্টার ভেরিয়েবল ব্যবহার করে do...while স্টেটমেন্টের আবর্তন সংখ্যা গণনা করে। নিচে do...while স্টেটমেন্টের ফরম্যাট দেয়া হলো—

Counter Declaration;

Counter Initialization;

do

{

Statement(s);

Increment/Decrement;

} while (Condition);

Counter Declaration অংশে উপযুক্ত ডেটাইপসহ ইনডেক্স ভেরিয়েবল ঘোষণা করে Counter Initialization অংশে তার প্রারম্ভিক মান দেওয়া হয়। Condition অংশে ইনডেক্স ভেরিয়েবলের চূড়ান্ত মান নির্ধারণের শর্ত দেওয়া হয়।

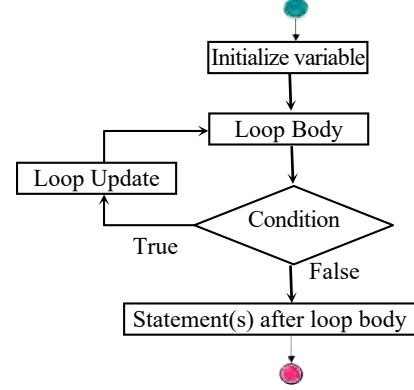
- দ্বিতীয় বন্ধনীর ভিতরের কোড ব্লক প্রথমে একবার সম্পাদিত(executed) হয়।
- তারপরে Condition নির্ণয় হয়। যদি Condition সত্য হয় তাহলে লুপের বডি পুনরায় সম্পাদিত হয়। Condition মিথ্যা না হওয়া পর্যন্ত এই প্রক্রিয়া চলতে থাকে।
- যখন Condition মিথ্যা হয়ে যায় বা মান 0(শূন্য) হয় তখন do...while লুপের সমাপ্তি ঘটে।

উদাহরণ-১: ০ থেকে ৩ পর্যন্ত সংখ্যা প্রদর্শনের জন্য প্রোগ্রাম লিখ।

```
#include<stdio.h>
main()
{
1   int x;
2   x=0;
3
4   do
5   {
6       printf("%d\n",x);
7       x=x+1;
8   } while(x<4);
9 }
```

আউটপুট:

0  
1  
2  
3



চিত্র: do-while স্টেটমেন্টের ফ্লোচার্ট

সি প্রোগ্রামিং-এ while এবং do..while লুপ একই রকম কাজ করে। এই দুইটি লুপের পার্থক্য হলো শুধু, while লুপ আগে টেস্ট কন্ডিশনটি চেক করে এবং তারপর কোড এক্সিকিউট করে অনুরূপ do..while লুপ আগে কোড এক্সিকিউট করে এবং তারপর টেস্ট কন্ডিশনটি চেক করে। সুতরাং do...while লুপ কমপক্ষে একবার এক্সিকিউশন হয়।

### অসীম লুপ (Infinite loop)

প্রোগ্রামে ইচ্ছা করলে লুপ সব সময়ের জন্য সত্য করে দেয়া যায় অর্থাৎ প্রোগ্রাম যতক্ষণ চলবে কম্পিউটার শুধু এই লুপ নিয়েই কাজ করবে। এ ধরনের লুপকে বলে অসীমলুপ (Infinite loop)। অসীমলুপ তৈরির জন্য for লুপ স্টেটমেন্টের ক্ষেত্রে *condition* অংশ বাদ দিতে হবে এবং while/ do-while লুপের ক্ষেত্রে *condition* অংশে কোনো এক্সপ্রেশন ব্যবহার না করে, 0 ছাড়া অন্য যে কোনো সংখ্যা ব্যবহার করতে হবে।

**উদাহরণ: for/ while / do-while loop ব্যবহার করে ১ থেকে অসীমপর্যন্ত সংখ্যা প্রিন্ট করার জন্য প্রোগ্রাম।**

for loop ব্যবহার করে	while loop ব্যবহার করে	do- while loop ব্যবহার করে
<pre>#include&lt;stdio.h&gt; main() { int a; for(a=1; ;a++) { printf("%d",a); } }</pre>	<pre>#include&lt;stdio.h&gt; main() { int a=1; while(1) { printf("%d",a); a++; } }</pre>	<pre>#include&lt;stdio.h&gt; main() { int a=1; do { printf("%d",a); a++; } while(1); }</pre>

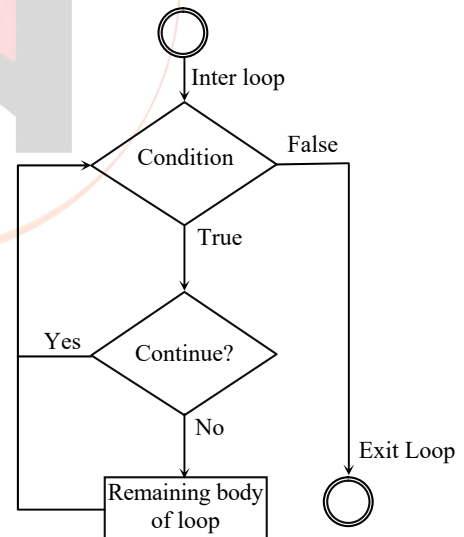
### জাম্প ও জাম্পিং স্টেটমেন্ট এর ব্যবহার (Jump and Uses of Jumping Statement)

প্রোগ্রামে সরল অনুক্রমকে ভঙ্গা করে প্রোগ্রামের এক লাইন থেকে পরবর্তী লাইনে না গিয়ে উপরে বা নিচে অন্য কোনো লাইন থেকে কাজ শুরু করলে তাকে জাম্প বলে। যে সকল স্টেটমেন্ট জাম্প এর কাজে ব্যবহৃত হয় তাকে জাম্পিং স্টেটমেন্ট বলে। জাম্পিং স্টেটমেন্ট গুলো হলো নিম্নরূপ:

- continue স্টেটমেন্ট
- break স্টেটমেন্ট
- goto স্টেটমেন্ট

### Continue স্টেটম্যান্ট

লুপের সাধারণ ফ্লো (flow) পরিবর্তন করার জন্য প্রোগ্রামিং-এ continue স্টেটমেন্ট ব্যবহৃত হয়। লুপের ভিতরের কিছু স্টেটমেন্টকে মাঝে মাঝে এড়িয়ে যাওয়ার প্রয়োজন হয়। এক্ষেত্রে continue স্টেটমেন্ট ব্যবহৃত হয়। অর্থাৎ সি-তে শর্তযুক্ত অথবা শর্তবিহীনভাবে কোনো স্টেটমেন্ট বা লুপের পুনরাবৃত্তি করার জন্য continue স্টেটমেন্ট ব্যবহৃত হয়। কোনো লুপের স্টেটমেন্ট অংশের যেখানে continue পাওয়া যাবে সেখান থেকে পরবর্তী ইনস্ট্রাকশনগুলো এক্সিকিউট হবে না এবং স্টেটমেন্ট অংশ আবার প্রথম থেকে কাজ শুরু করবে। অর্থাৎ continue স্টেটমেন্ট প্রোগ্রাম পয়েন্টারকে পূর্ববর্তী স্টেটমেন্ট বা লুপের প্রারম্ভে স্থানান্তর করে।



চিত্র : continue স্টেটমেন্টের ফ্লোচার্ট

**continue** স্টেটমেন্টের ফরম্যাট হলো:

```
continue;
```

বিস্তারিত ভাবে লেখা যায়,

```

→ While (test Expression)
{
    // codes
    if (condition for continue)
    {
        Continue;
    }
    // codes
}

→ for (initialization, condition, update)
{
    // codes
    if (condition for continue)
    {
        continue;
    }
    // codes
}

```

continue স্টেটমেন্ট if, else if, for, while ইত্যাদি ছাড়া কাজ করতে পারে না। তবে শর্তবিহীন continue স্টেটমেন্ট অসীম লুপের সৃষ্টি করে। এ জন্য সাধারণত if, else... if স্টেটমেন্টের সাথে সম্পর্কিত শর্ত সাপেক্ষে কোনো লুপের পুনরাবৃত্তি করার জন্য continue স্টেটমেন্ট ব্যবহৃত হয়। সেক্ষেত্রে শর্তের মান সত্য হলে continue স্টেটমেন্ট কার্যকরী হয়, অন্যথায় কম্পাইলার continue স্টেটমেন্ট উপেক্ষা করে পরবর্তী স্টেটমেন্ট নির্বাহ করে।

এখন একটি প্রোগ্রামের সাহায্যে continue স্টেটমেন্টের কাজ লক্ষ করি-

```

#include<stdio.h>
main()
{
    int a;
    for(a=1;a<=5;a++)
    {
        printf("\nThank");
        printf(" you");
    }
}

```

**ফলাফল:**

```

Thank you
Thank you
Thank you
Thank you
Thank you

```

কিন্তু যদি প্রোগ্রামে continue স্টেটমেন্ট ব্যবহার করে একটু পরিবর্তন করি,

```
#include<stdio.h>
```

```
main()
```

```
{
  int a;
```

```
  for(a=1;a<=5;a++)
```

```
  {
    printf("\nThank");
```

```
    if(a==2||a==4)
```

```
      continue;
```

```
    printf(" you");
```

```
  }
```

```
}
```

ফলাফল:

Thank you

Thank

Thank you

Thank

Thank you

← for a=1

← for a=2

← for a=3

← for a=4

← for a=5



কাজ:

```
#include<stdio.h>
```

```
main()
```

```
{
```

```
  int a, sum=0;
```

```
  for(a=1;a<=100;a++)
```

```
  {
```

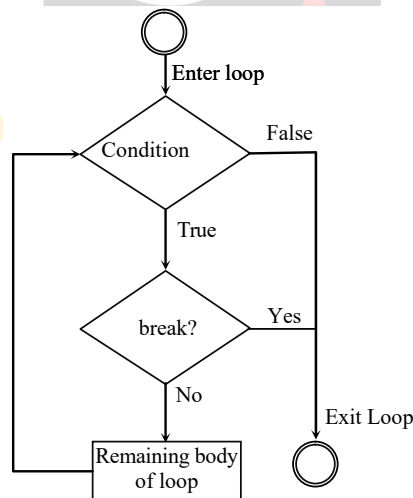
```
    sum=sum+a;
```

```
  }
```

প্রোগ্রামটির ৪ নং লাইনে কোনো রূপ পরিবর্তন না করে ধারাটির জোড় সংখ্যক / বিজোড় সংখ্যক পদের যোগফল নির্ণয়ের জন্য সি ভাষায় প্রোগ্রাম লেখ।

### break স্টেটমেন্ট

লুপের সাধারণ ফ্লো(flow) কে বন্ধ করে দেওয়ার জন্য প্রোগ্রামিং-এ break স্টেটমেন্ট ব্যবহৃত হয়। মাঝে মধ্যে টেস্ট এক্সপ্রেশনকে চেক করা ছাড়াই লুপকে তাৎক্ষণিক বন্ধ করে দেওয়ার প্রয়োজন হয়। এক্ষেত্রে break স্টেটমেন্ট ব্যবহৃত হয়। সি প্রোগ্রাম break স্টেটমেন্ট পাওয়া মাত্রই for, while ও do...while লুপকে তাৎক্ষণিক বন্ধ করে দেয়। কোনো লুপের নির্বাহ স্থগিত করে অর্থাৎ লুপ শেষ হবার আগেই লুপ থেকে বের হয়ে আসার জন্য break স্টেটমেন্ট ব্যবহৃত হয়। break স্টেটমেন্ট continue স্টেটমেন্টের বিপরীত কাজ করে।



চিত্র : break স্টেটমেন্টের ফ্লোচার্ট

**break** স্টেটমেন্টের ফরম্যাট হলো—

break;

সিন্দান্ত গ্রহণের জন্য continue স্টেটমেন্টের মতো break স্টেটমেন্টের জন্যও সাধারণত if, else if, for, while ইত্যাদি স্টেটমেন্ট ব্যবহৃত হয়।

এখন, প্রোগ্রামের সাহায্যে break স্টেটমেন্টের কাজ লক্ষ করি—

#include&lt;stdio.h&gt;

main()

```
{
    int a;
    for(a=1;a<=6;a++)
    {
        printf("%d\n",a);
    }
}
```

ফলাফল:

```
1
2
3
4
5
6
```

উক্ত প্রোগ্রামে ফলাফল 1 থেকে 6 পর্যন্ত সংখ্যা ছাপা হয়েছে কিন্তু break স্টেটমেন্ট ব্যবহার করে 6 পর্যন্ত ছাপানোর আগেই প্রোগ্রাম নির্বাহ বন্ধ করে দেয়া যায়।

#include&lt;stdio.h&gt;

main()

```
{
    int a;
    for(a=1;a<=6;a++)
    {
        printf("%d\n",a);
        if(a==3)
            break;
    }
}
```

অথবা #include&lt;stdio.h&gt;

main()

```
{
    int a;
    for(a=1;a<=6;a++)
    {
        printf("%d\n",a);
        if(a>2)
            break;
    }
}
```

ফলাফল:

```
1
2
3
```

উক্ত প্রোগ্রামে a এর মান ৩ বা ২ এর বেশি হলে break স্টেটমেন্ট ব্যবহার করে প্রোগ্রাম নির্বাহ বন্ধ করে দেয়া হয়েছে।

এখন continue ও break স্টেটমেন্ট ব্যবহার করে একটি প্রোগ্রাম দেয়া হলো—

#include&lt;stdio.h&gt;

main()

```
{
    int a;
    for(;;)
    {
        printf("Enter the positive number: ");
        scanf("%d",&a);
        if(a<0)
            continue;
        else
            break;
    }
    printf("You have entered %d",a);
}
```

ফলাফল:

```
Enter the positive number:-12
Enter the positive number:12
You have entered 12
```



## goto স্টেটমেন্ট

সি প্রোগ্রামের সাধারণ ধারাকে পরিবর্তন করার জন্য goto স্টেটমেন্ট ব্যবহার করা হয়। সাধারণত প্রোগ্রামের কোনো একটি অংশের কাজ বন্ধ রেখে অন্য একটি অংশকে সচল করাই goto statement-এর লক্ষ্য। অর্থাৎ শর্তযুক্ত বা শর্তবিহীনভাবে এক স্টেটমেন্ট থেকে উপরে বা নিচে অপর কোনো স্টেটমেন্টে নিয়ন্ত্রণ স্থানান্তর করার জন্য goto স্টেটমেন্ট ব্যবহার করা হয়। goto স্টেটমেন্টের ফরম্যাট হলো-

**LevelName:**

¶

**goto LevelName;**

এখানে, লেভেল (LevelName) একটা নাম, যা ভেরিয়েবলের নাম লেখার নিয়ম অনুসরণ করে নামের পর কোলন ( : ) ব্যবহার করতে হয়। একই প্রোগ্রাম বা ফাংশনে প্রয়োজনে ভিন্ন ভিন্ন নামে একাধিক লেভেল স্টেটমেন্টের উপরে বা নিচে ব্যবহার করা যেতে পারে। goto স্টেটমেন্টের উপরে লেভেল স্টেটমেন্টের ব্যবহার continue স্টেটমেন্ট ব্যবহারের অনুরূপ। continue স্টেটমেন্ট ব্যবহার করলে কেবল উপরের দিকে জাম্প করা যায় কিন্তু goto ব্যবহার করে প্রোগ্রামের সামনে কিংবা পিছনে যেকোনো স্থানে স্থানান্তর বা জাম্প করা যায়। goto স্টেটমেন্ট if, else if, for, while ইত্যাদি ছাড়া সরাসরি কাজ করতে পারে। তবে সাধারণত goto স্টেটমেন্ট ব্যবহার করে if, else if স্টেটমেন্টের সাথে সম্পর্কিত শর্ত সাপেক্ষে প্রোগ্রামের অপর কোন স্থানে জাম্প করা যায়। সেক্ষেত্রে শর্তের মান সত্য হলে goto স্টেটমেন্ট কার্যকরী হয়। অন্যথায় কম্পাইলার goto স্টেটমেন্ট উপেক্ষা করে পরবর্তী স্টেটমেন্ট সম্পাদন করে।



**জেনে রাখো**

একজন ভালো প্রোগ্রামের সবসময় goto স্টেটমেন্টের ব্যবহার এড়িয়ে চলা উচিত। কারণ goto স্টেটমেন্ট সমস্ত প্রোগ্রামে লেভেল খোঁজ করতে থাকে।

**উদাহরণ-১:** দুটি সংখ্যার গ.সা.গু নির্ণয়ের জন্য অ্যালগরিদম, ফ্লোচার্ট ও প্রোগ্রাম লিখ।

**অ্যালগরিদম**

ধাপ ১: শুরু

ধাপ ২: a, b এর মান গ্রহণ

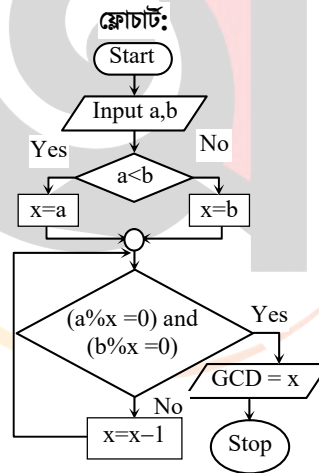
ধাপ ৩:  $a < b$  সত্য হলে  $x = a$  মিথ্যা  $x = b$

ধাপ ৪:  $(a \% x = 0)$  and  $(b \% x = 0)$  সত্য হলে ৬নং ধাপে যায়। নাহলে ৫ নং ধাপে যায়।

ধাপ ৫:  $x = x - 1$  নির্ণয় করি এবং ৪নং ধাপে যায়।

ধাপ ৬: ফলাফল x ছাপাই।

ধাপ ৭: শেষ



**প্রোগ্রাম:**

```

#include<stdio.h>
int main()
{
    int a, b, x;
    scanf("%d %d",&a,&b);
    x=(a<b)?a:b;
    again:
    if((a%x==0)&&(b%x==0))
        printf("GCD=%d",x);
    else
    {
        x=x-1;
        goto again;
    }
    return 0;
}
  
```

উদাহরণ-২. দুটি সংখ্যার ল.সা.গু নির্ণয়ের জন্য অ্যালগরিদম, ফ্লোচার্ট ও প্রোগ্রাম লিখ।

অ্যালগরিদম

ধাপ ১: শুরু

ধাপ ২: a, b এর মান গ্রহণ

ধাপ ৩:  $a > b$  সত্য হলে  $x=a$  মিথ্যা  $x=b$

ধাপ ৪:  $(x \% a = 0)$  and  $(x \% b = 0)$  সত্য হলে ৬নং ধাপে যায়।

নাহলে ৫ নং ধাপে যায়।

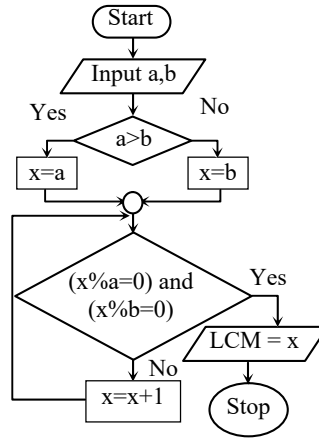
ধাপ ৫:  $x=x+1$  নির্ণয় করি

এবং ৪নং ধাপে যায়।

ধাপ ৬: ফলাফল x ছাপাই।

ধাপ ৭: শেষ

ফ্লোচার্ট:



প্রোগ্রাম:

```
#include<stdio.h>
int main()
{
    int a, b, x;
    scanf("%d %d",&a,&b);
    x=(a>b)?a:b;
    again:
    if((x%a==0)&&(x%b==0))
        printf("LCM=%d",x);
    else
    {
        x=x+1;
        goto again;
    }
    return 0;
}
```

উদাহরণ-৩. goto স্টেটমেন্ট ব্যবহার করে কোন সংখ্যার ফ্যাক্টোরিয়াল নির্ণয়ের জন্য প্রোগ্রাম লিখ।

```
#include<stdio.h>
```

```
main()
```

```
{
```

```
    int a,x;
```

```
    long fact=1;
```

```
    again:
```

```
        printf("\nType the positive integer: ");
```

```
        scanf("%d",&x);
```

```
    if(x<0)
```

```
    {
```

```
        printf("Negative number not allowed. ");
```

```
        goto again;
```

```
    }
```

```
    else if(x==0)
```

```
        printf("Factorial=1");
```

```
    else
```

```
    {
```

```
        for(a=2;a<=x;a++)
```

```
            fact=fact*a;
```

```
            printf("Factorial=%ld",fact);
```

```
    }
```

```
}
```

ফলাফল:

Type the positive integer: -12

Negative number not allowed.

Type the positive integer: 7

Factorial=5040

উদাহরণ-৪: goto স্টেটমেন্ট ব্যবহার করে ১ থেকে ১০ পর্যন্ত সংখ্যা প্রদর্শনের জন্য প্রোগ্রাম লিখ।

```
#include<stdio.h>

int main()
{
    int a=1;
    level:
    printf("%d\n",a);
    a=a+1;
    if (a<=10)
        goto level;
    return 0;
}
```

#### ধারা নির্ণয়

কোনো ধারা নির্ণয়ের জন্য চার ধরনের স্টেটমেন্ট ব্যবহার করা হয়। যথা: for,while, do-while, if-goto। ধরি কোনো ধারার প্রথমপদ a ও সাধারণ অন্তর d এবং শেষপদ n। বিভিন্ন ধারা নির্ণয়ের জন্য শুধুমাত্র পরিবর্তন হয় a, d এবং n-এর মান।

কোনো ধারার যোগফল নির্ণয়ের জন্য নিচে স্টেটমেন্ট গুলোর গঠন দেওয়া হলো।

<b>for স্টেটমেন্ট এর গঠন:</b> for (a = প্রথম পদ; a <= শেষপদ; a = a+ সাধারণ অন্তর) { printf(“%d”, a); } 	<b>while স্টেটমেন্ট এর গঠন:</b> a=প্রথম পদ; while(a <= শেষপদ) { printf(“%d”, a); a = a + সাধারণ অন্তর; } 
<b>do-while স্টেটমেন্ট এর গঠন:</b> a=প্রথম পদ; do { printf(“%d”, a); a=a+সাধারণ অন্তর; } while(a<=শেষপদ); 	<b>if-goto স্টেটমেন্ট এর গঠন:</b> a=প্রথম পদ; level: printf(“%d”, a); a=a+সাধারণ অন্তর; if(a<=শেষপদ) goto level; 

উদাহরণ-১: ১, ২, ৩ ..... n ধারাটি নির্ণয়ের জন্য অ্যালগরিদম, ফ্লোচার্ট ও প্রোগ্রাম লিখ।

**if-goto এবং do-while লুপ স্টেটমেন্ট ব্যবহার করে:**

অ্যালগরিদম:

ধাপ-১: শুরু করি।

ধাপ-২: ইনপুট হিসেবে n এর মান গ্রহণ করি।

ধাপ-৩: a = 1 ধরি।

ধাপ-৪: a এর মান ছাপাই।

ধাপ-৫: a = a + 1 নির্ণয় করি।

ধাপ-৬: যদি a ≤ n হয় তবে ৪ নং ধাপে যাই।

অন্যথায় ৭ নং ধাপে যাই।

ধাপ-৭: শেষ করি।

**if-goto ব্যবহার করে**

```
#include<stdio.h>
```

```
int main()
```

```
{
```

```
int a,n;
```

```
scanf("%d",&n);
```

```
a=1;
```

```
level:
```

```
printf("%d ",a);
```

```
a=a+1;
```

```
if(a<=n)
```

```
goto level;
```

```
return 0;
```

```
}
```

**do-while ব্যবহার করে**

```
#include<stdio.h>
```

```
int main()
```

```
{
```

```
int a,n;
```

```
scanf("%d",&n);
```

```
a=1;
```

```
do
```

```
{
```

```
printf("%d ",a);
```

```
a=a+1;
```

```
} while(a<=n);
```

```
return 0;
```

```
}
```

**for এবং while লুপ স্টেটমেন্ট ব্যবহার করে:**

অ্যালগরিদম:

ধাপ-১: শুরু করি।

ধাপ-২: ইনপুট হিসেবে n এর মান গ্রহণ করি।

ধাপ-৩: a = 1 ধরি।

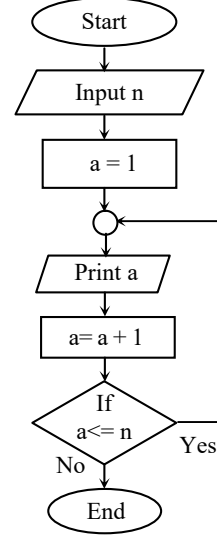
ধাপ-৪: যদি a ≤ n হয় তবে ৫ নং ধাপে যাই।

অন্যথায় ৬ নং ধাপে যাই।

ধাপ-৫: a = a + 1 নির্ণয় করি।

ধাপ-৬: a এর মান ছাপাই।

ফ্লোচার্ট:



ধাপ-৭: শেষ করি।

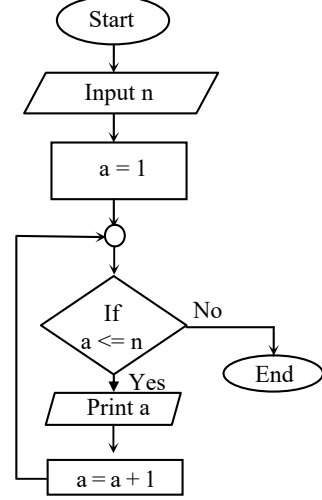
**while** ব্যবহার করে

```
#include<stdio.h>
int main()
{
    int a,n;
    scanf("%d",&n);
    a=1;
    while(a<= n)
    {
        printf("%d ",a);
        a=a+1;
    }
    return 0;
}
```

**for** ব্যবহার করে

```
#include<stdio.h>
main()
{
    int a,n;
    scanf("%d",&n);
    for(a=1;a<=n; a++)
    {
        printf("%d ",a);
    }
}
```

ফ্লোচার্ট:



কাজ:

১০,২০,৩০, ... .. , ১০০ ধারাটি প্রদর্শনের জন্য সি ভাষায় প্রোগ্রাম লেখ।

**ধারার যোগফল নির্ণয়**

অনুরূপ কোনো ধারার যোগফল নির্ণয়ের জন্য চার ধরনের স্টেটমেন্ট ব্যবহার করা হয়। যথা: for,while, do-while, if-goto। ধরি, কোনো ধারার যোগফল s, প্রথমপদ a ও সাধারণ অন্তর d এবং শেষপদ n। বিভিন্ন ধারার যোগফল নির্ণয়ের জন্য শুধুমাত্র পরিবর্তন হয় s, a, d এবং n-এর মান।

কোনো ধারার যোগফল নির্ণয়ের জন্য নিচে স্টেটমেন্ট গুলোর গঠন দেওয়া হলো।

<p><b>for স্টেটমেন্ট এর গঠন:</b></p> <pre>s=0; for(a=প্রথম পদ;a&lt;=শেষপদ;a=a+সাধারণ অন্তর) {     s=s+a; } printf("%d",s);</pre>	<p><b>while স্টেটমেন্ট এর গঠন:</b></p> <pre>s=0; a=প্রথম পদ; while(a&lt;=শেষপদ) {     s=s+a;     a=a+সাধারণ অন্তর; } printf("%d",s);</pre>
<p><b>do-while স্টেটমেন্ট এর গঠন:</b></p> <pre>s=0; a=প্রথম পদ; do {     s=s+a;     a=a+সাধারণ অন্তর; } while(a&lt;=শেষপদ); printf("%d",s);</pre>	<p><b>if-goto স্টেটমেন্ট এর গঠন:</b></p> <pre>s=0; a=প্রথম পদ; level: s=s+a;     a=a+সাধারণ অন্তর; if(a&lt;=শেষপদ) goto level; printf("%d",s);</pre>

উদাহরণ-১.  $1 + 2 + 3 + \dots + n$  ধারাটির যোগফল নির্ণয়ের জন্য অ্যালগরিদম, ফ্লোচার্ট ও প্রোগ্রাম লিখ।

**if-goto এবং do-while ব্যবহার করে:**

**অ্যালগরিদম:**

ধাপ-১: প্রোগ্রাম শুরু করি।

ধাপ-২: ইনপুট হিসেবে  $n$  এর মান গ্রহণ করি।

ধাপ-৩:  $s = 0$ ,  $a = 1$  ধরি।

ধাপ-৪:  $s = s + a$ ,  $a = a + 1$  নির্ণয় করি।

ধাপ-৫: যদি  $a \leq n$  হয় তবে ৪ নং ধাপে যাই।

অন্যথায় ৬ নং ধাপে যাই।

ধাপ-৬:  $s$  এর মান ছাপাই।

ধাপ-৭: প্রোগ্রাম শেষ করি।

**if-goto ব্যবহার করে**

```
#include<stdio.h>
int main()
{
    int a,s,n;
    scanf("%d",&n);
    s=0;
    a=1;
    level:
        s=s+a;
        a=a+1;
        if(a<=n) goto level;
    printf("%d ",s);
    return 0;
}
```

**do-while ব্যবহার করে**

```
#include<stdio.h>
int main()
{
    int a,s,n;
    scanf("%d",&n);
    s=0;
    a=1;
    do
    {
        s=s+a;
        a=a+1;
    } while(a<=n);
    printf("%d ",s);
    return 0;
}
```

**for এবং while লুপ স্টেটমেন্ট ব্যবহার করে:**

**অ্যালগরিদম:**

ধাপ-১: প্রোগ্রাম শুরু করি।

ধাপ-২: ইনপুট হিসেবে  $n$  এর মান গ্রহণ করি।

ধাপ-৩:  $s = 0$ ,  $a = 1$  ধরি।

ধাপ-৪: যদি  $a \leq n$  হয় তবে ৫ নং ধাপে যাই।

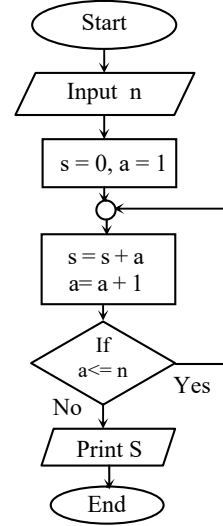
অন্যথায় ৬ নং ধাপে যাই।

ধাপ-৫:  $s = s + a$ ,  $a = a + 1$  নির্ণয় করি। ৪ নং ধাপে ফেরত যাই।

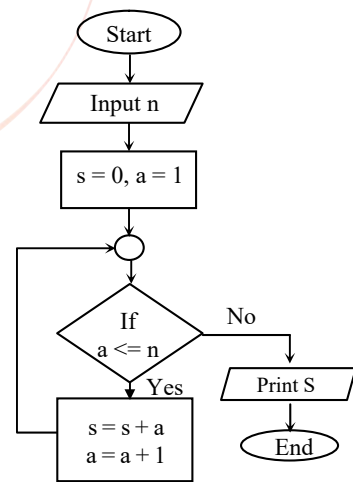
ধাপ-৬:  $s$  এর মান ছাপাই।

ধাপ-৭: প্রোগ্রাম শেষ করি।

**ফ্লোচার্ট:**



**ফ্লোচার্ট:**



### while ব্যবহার করে

```
#include<stdio.h>
int main()
{
    int a,s,n;
    scanf("%d",&n);
    s=0;
    a=1;
    while(a<=n)
    {
        s=s+a;
        a=a+1;
    }
    printf("%d ",s);
    return 0;
}
```

### for ব্যবহার করে

```
#include<stdio.h>
main()
{
    int a,s,n;
    scanf("%d",&n);
    s=0;
    for(a=1;a<=n; a++)
    {
        s=s+a;
    }
    printf("%d ",s);
}
```

উদাহরণ-২.  $1 + 3 + 5 + \dots + n$  ধারাটির যোগফল নির্ণয়ের জন্য অ্যালগরিদম, ফ্লোচার্ট ও প্রোগ্রাম লিখ।

### if-goto এবং do-while ব্যবহার করে:

#### অ্যালগরিদম:

- ধাপ-১: প্রোগ্রাম শুরু করি।  
 ধাপ-২: ইনপুট হিসেবে n এর মান গ্রহণ করি।  
 ধাপ-৩:  $s = 0$ ,  $a = 1$  ধরি।  
 ধাপ-৪:  $s = s + a$ ,  $a = a + 2$  নির্ণয় করি।  
 ধাপ-৫: যদি  $a \leq n$  হয় তবে ৪ নং ধাপে যাই।  
 অন্যথায় ৬ নং ধাপে যাই।  
 ধাপ-৬: s এর মান ছাপাই।  
 ধাপ-৭: প্রোগ্রাম শেষ করি।

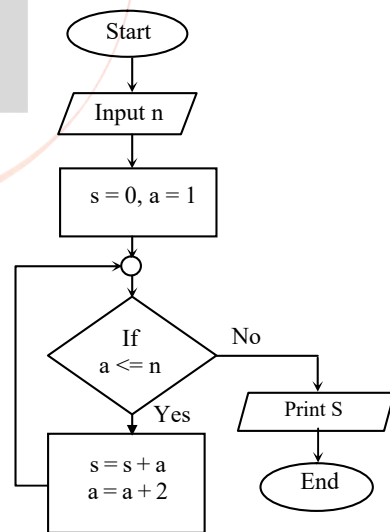
### if-goto ব্যবহার করে

```
#include<stdio.h>
int main()
{
    int a,s,n;
    scanf("%d",&n);
    s=0;
    a=1;
    level:
        s=s+a;
        a=a+2;
        if(a<=n) goto level;
    printf("%d ",s);
    return 0;
}
```

### do-while ব্যবহার করে

```
#include<stdio.h>
int main()
{
    int a,s,n;
    scanf("%d",&n);
    s=0;
    a=1;
    do
    {
        s=s+a;
        a=a+2;
    } while(a<=n);
    printf("%d ",s);
    return 0;
}
```

#### ফ্লোচার্ট:





**for এবং while লুপ স্টেটমেন্ট ব্যবহার করে:****অ্যালগরিদম:**

ধাপ-১: প্রোগ্রাম শুরু করি।

ধাপ-২: ইনপুট হিসেবে n এর মান গ্রহণ করি।

ধাপ-৩: s = 0, a = 1 ধরি।

ধাপ-৪: যদি a ≤ n হয় তবে ৫ নং ধাপে যাই।

অন্যথায় ৬ নং ধাপে যাই।

ধাপ-৫: s = s + a, a = a + 2 নির্ণয় করি। ৪ নং ধাপে ফেরত যাই।

ধাপ-৬: s এর মান ছাপাই।

ধাপ-৭: প্রোগ্রাম শেষ করি।

**for ব্যবহার করে**

```
#include<stdio.h>
main()
{
    int a,s,n;
    scanf("%d",&n);
    s=0;
    for(a=1;a<=n;a=a+2)
    {
        s=s+a;
    }
    printf("%d ",s);
    return 0;
}
```

**while ব্যবহার করে**

```
#include<stdio.h>
int main()
{
    int a,s,n;
    scanf("%d",&n);
    s=0;
    a=1;
    while(a<=n)
    {
        s=s+a;
        a=a+2;
    }
    printf("%d ",s);
    return 0;
}
```

উদাহরণ-৩. ২ + ৪ + ৬ + ..... + n ধারাটির যোগফল নির্ণয়ের জন্য অ্যালগরিদম, ফ্লোচার্ট ও প্রোগ্রাম লিখ।

**if-goto এবং do-while লুপ স্টেটমেন্ট ব্যবহার করে:****অ্যালগরিদম:**

ধাপ-১: প্রোগ্রাম শুরু করি।

ধাপ-২: ইনপুট হিসেবে n এর মান গ্রহণ করি।

ধাপ-৩: s = 0, a = 2 ধরি।

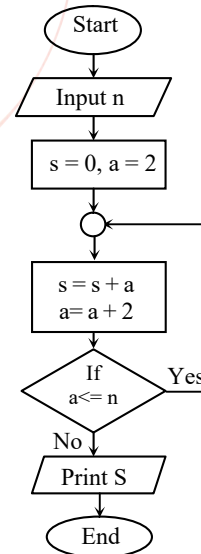
ধাপ-৪: s = s + a, a = a + 2 নির্ণয় করি।

ধাপ-৫: যদি a ≤ n হয় তবে ৪ নং ধাপে যাই।

অন্যথায় ৬ নং ধাপে যাই।

ধাপ-৬: s এর মান ছাপাই।

ধাপ-৭: প্রোগ্রাম শেষ করি।

**ফ্লোচার্ট:**

### if-goto ব্যবহার করে

```
#include<stdio.h>
int main()
{
    int a,s,n;
    scanf("%d",&n);
    s=0;
    a=2;
    level:
        s=s+a;
        a=a+2;
        if(a<=n) goto level;
    printf("%d ",s);
    return 0;
}
```

### do-while ব্যবহার করে

```
#include<stdio.h>
int main()
{
    int a,s,n;
    scanf("%d",&n);
    s=0;
    a=2;
    do
    {
        s=s+a;
        a=a+2;
    } while(a<=n);
    printf("%d ",s);
    return 0;
}
```

### for এবং while লুপ স্টেটমেন্ট ব্যবহার করে:

#### অ্যালগরিদম:

ধাপ-১: প্রোগ্রাম শুরু করি।

ধাপ-২: ইনপুট হিসেবে n এর মান গ্রহণ করি।

ধাপ-৩: s = 0, a = 2 ধরি।

ধাপ-৪: যদি a <= n হয় তবে ৫ নং ধাপে যাই।

অন্যথায় ৬ নং ধাপে যাই।

ধাপ-৫: s = s + a, a = a + 2 নির্ণয় করি। ৪ নং ধাপে ফেরত যাই।

ধাপ-৬: s এর মান ছাপাই।

ধাপ-৭: প্রোগ্রাম শেষ করি।

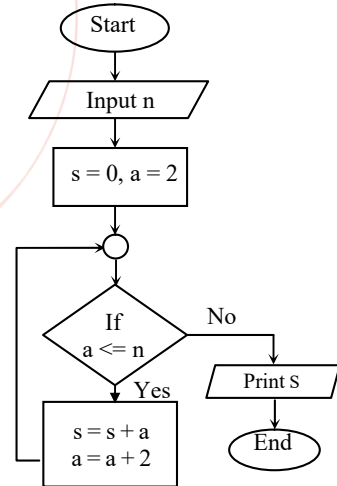
#### for ব্যবহার করে

```
#include<stdio.h>
main()
{
    int a,s,n;
    scanf("%d",&n);
    s=0;
    for(a=2;a<=n;a=a+2)
    {
        s=s+a;
    }
    printf("%d ",s);
}
```

#### while ব্যবহার করে

```
#include<stdio.h>
int main()
{
    int a,s,n;
    scanf("%d",&n);
    s=0;
    a=2;
    while(a<=n)
    {
        s=s+a;
        a=a+2;
    }
    printf("%d ",s);
    return 0;
}
```

#### ফ্লোচার্ট:



উদাহরণ-৪.  $1 + 2 + 3 + \dots + 100$  ধারাটির যোগফল নির্ণয়ের জন্য অ্যালগরিদম, ফ্লোচার্ট ও প্রোগ্রাম লিখ।

অ্যালগরিদম:

**if-goto এবং do-while** লুপ স্টেটমেন্ট ব্যবহার করে:

ধাপ-১: শুরু করি।

ধাপ-২:  $s = 0$ ,  $a = 1$  ধরি।

ধাপ-৩:  $s = s + a$ ,  $a = a + 1$  নির্ণয় করি।

ধাপ-৪: যদি  $a \leq 100$  হয় তবে ৩ নং ধাপে যাই।

ধাপ-৫:  $s$  এর মান ছাপাই।

ধাপ-৬: শেষ করি।

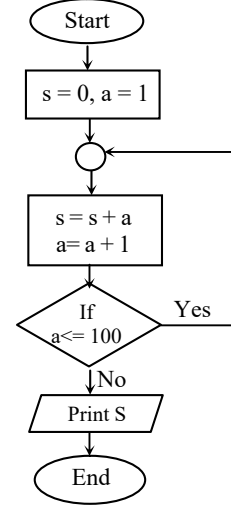
**if-goto ব্যবহার করে**

```
#include<stdio.h>
main()
{
    int a,s;
    s=0;
    a=1;
    level:
        s=s+a;
        a=a+1;
        if(a<=100) goto level;
        printf("%d ",s);
}
```

**do-while ব্যবহার করে**

```
#include<stdio.h>
main()
{
    int a,s;
    s=0;
    a=1;
    do
    {
        s=s+a;
        a=a+1;
    } while(a<=100);
    printf("%d ",s);
}
```

ফ্লোচার্ট:



**for এবং while** লুপ স্টেটমেন্ট ব্যবহার করে:

অ্যালগরিদম:

ধাপ-১: প্রোগ্রাম শুরু করি।

ধাপ-২:  $s = 0$ ,  $a = 1$  ধরি।

ধাপ-৩: যদি  $a \leq 100$  হয় তবে ৪ নং ধাপে যাই।

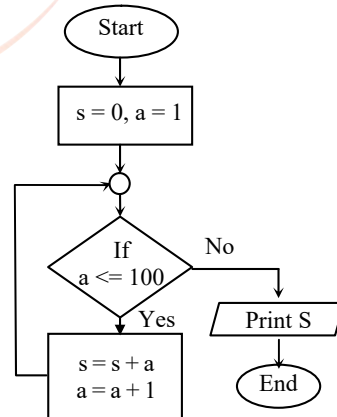
অন্যথায় ৫ নং ধাপে যাই।

ধাপ-৪:  $s = s + a$ ,  $a = a + 2$  নির্ণয় করি। ৩ নং ধাপে ফেরত যাই।

ধাপ-৫:  $s$  এর মান ছাপাই।

ধাপ-৬: প্রোগ্রাম শেষ করি।

ফ্লোচার্ট:



### for ব্যবহার করে

```
#include<stdio.h>
main()
{
    int a,s;
    s=0;
    for(a=1;a<=100; a=a+1)
    {
        s=s+a;
    }
    printf("%d ",s);
}
```

### while ব্যবহার করে

```
#include<stdio.h>
main()
{
    int a,s;
    s=0;
    a=1;
    while(a<=100)
    {
        s=s+a;
        a=a+1;
    }
    printf("%d ",s);
}
```

উদাহরণ-৫.  $1 + 3 + 5 + \dots + 100$  ধারাটির যোগফল নির্ণয়ের জন্য অ্যালগরিদম, ফ্লোচার্ট ও প্রোগ্রাম লিখ।

### if-goto এবং do-while লুপ স্টেটমেন্ট ব্যবহার করে:

অ্যালগরিদম:

ধাপ-১: শুরু করি।

ধাপ-২:  $s = 0$ ,  $a = 1$  ধরি।

ধাপ-৩:  $s = s + a$ ,  $a = a + 2$  নির্ণয় করি।

ধাপ-৪: যদি  $a \leq 100$  হয় তবে ৩ নং ধাপে যাই।

ধাপ-৫:  $s$  এর মান ছাপাই।

ধাপ-৬: শেষ করি।

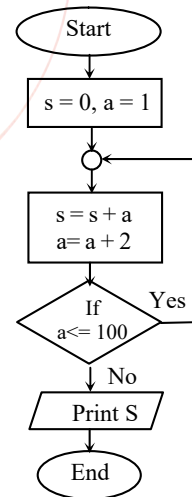
### if-goto ব্যবহার করে

```
#include<stdio.h>
main()
{
    int a,s;
    s=0;
    a=1;
    level:
        s=s+a;
        a=a+2;
        if(a<=100) goto level;
    printf("%d ",s);
}
```

### do-while ব্যবহার করে

```
#include<stdio.h>
main()
{
    int a,s;
    s=0;
    a=1;
    do
    {
        s=s+a;
        a=a+2;
    } while(a<=100);
    printf("%d ",s);
}
```

### ফ্লোচার্ট:



**for এবং while লুপ স্টেটমেন্ট ব্যবহার করে:****অ্যালগরিদম:**

ধাপ-১: শুরু করি।

ধাপ-২:  $s = 0$ ,  $a = 1$  ধরি।ধাপ-৩: যদি  $a \leq 100$  হয় তবে ৪ নং ধাপে যাই।

অন্যথায় ৫ নং ধাপে যাই।

ধাপ-৪:  $s = s + a$ ,  $a = a + 2$  নির্ণয় করি। ৩ নং ধাপে ফেরত যাই।ধাপ-৫:  $s$  এর মান ছাপাই।

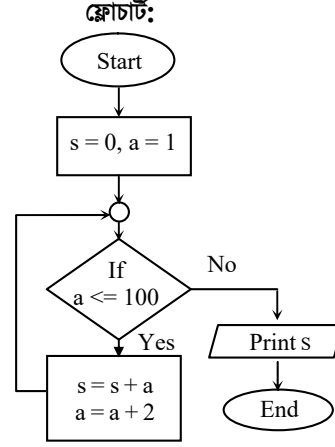
ধাপ-৬: শেষ করি।

**for ব্যবহার করে**

```
#include<stdio.h>
main()
{
    int a,s;
    s=0;
    for(a=1;a<=100;a=a+2)
    {
        s=s+a;
    }
    printf("%d ",s);
}
```

**while ব্যবহার করে**

```
#include<stdio.h>
main()
{
    int a,s;
    s=0;
    a=1;
    while(a<=100)
    {
        s=s+a;
        a=a+2;
    }
    printf("%d ",s);
}
```

উদাহরণ-৬.  $২ + ৪ + ৬ + \dots + 100$  ধারাটির যোগফল নির্ণয়ের জন্য অ্যালগরিদম, ফ্লোচার্ট ও প্রোগ্রাম।**if-goto এবং do-while লুপ স্টেটমেন্ট ব্যবহার করে:****অ্যালগরিদম:**

ধাপ-১: প্রোগ্রাম শুরু করি।

ধাপ-২:  $s = 0$ ,  $a = 2$  ধরি।ধাপ-৩:  $s = s + a$ ,  $a = a + 2$  নির্ণয় করি।ধাপ-৪: যদি  $a \leq 100$  হয় তবে ৩ নং ধাপে যাই।ধাপ-৫:  $s$  এর মান ছাপাই।

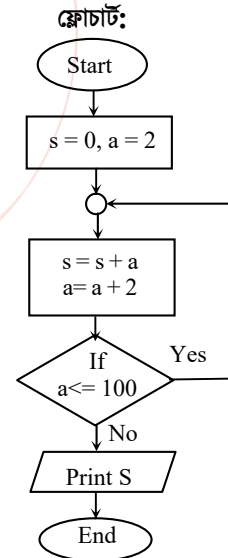
ধাপ-৬: প্রোগ্রাম শেষ করি।

**if-goto ব্যবহার করে**

```
#include<stdio.h>
main()
{
    int a,s;
    s=0;
    a=2;
```

**do-while ব্যবহার করে**

```
#include<stdio.h>
main()
{
    int a,s;
    s=0;
    a=2;
```



<pre>level:     s=s+a;     a=a+2;     if(a&lt;=100) goto level;     printf("%d ",s); }</pre>	<pre>do {     s=s+a;     a=a+2; }while(a&lt;=100); printf("%d ",s); }</pre>
--	---

#### for এবং while লুপ স্টেটমেন্ট ব্যবহার করে:

##### অ্যালগরিদম:

ধাপ-১: প্রোগ্রাম শুরু করি।

ধাপ-২:  $s = 0$ ,  $a = 2$  ধরি।

ধাপ-৩: যদি  $a \leq 100$  হয় তবে ৪ নং ধাপে যাই।

অন্যথায় ৫ নং ধাপে যাই।

ধাপ-৪:  $s = s + a$ ,  $a = a + 2$  নির্ণয় করি। ৩ নং ধাপে ফেরত পাই।

ধাপ-৫:  $s$  এর মান ছাপাই।

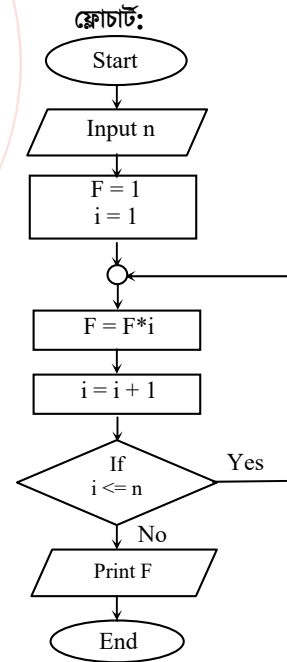
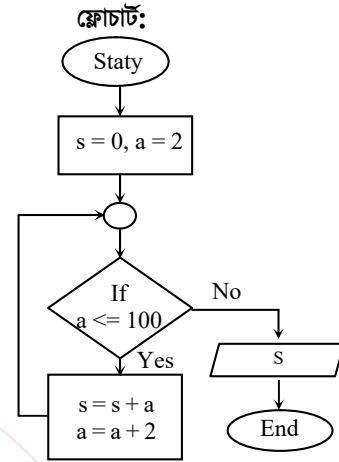
ধাপ-৬: প্রোগ্রাম শেষ করি।

##### for ব্যবহার করে

```
#include<stdio.h>
main()
{
    int a,s;
    s=0;
    for(a=2;a<=100; a=a+2)
    {
        s=s+a;
    }
    printf("%d ",s);
}
```

##### while ব্যবহার করে

```
#include<stdio.h>
main()
{
    int a,s;
    s=0;
    a=2;
    while(a<=100)
    {
        s=s+a;
        a=a+2;
    }
    printf("%d ",s);
}
```



উদাহরণ-৭. কোনো পূর্ণ সংখ্যার ফ্যাক্টোরিয়াল নির্ণয়ের জন্য অ্যালগরিদম, ফ্লোচার্ট ও প্রোগ্রাম লিখ।

#### if-goto এবং do-while লুপ স্টেটমেন্ট ব্যবহার করে:

##### অ্যালগরিদম:

ধাপ-১: প্রোগ্রাম শুরু করি।

ধাপ-২: ইনপুট হিসেবে  $n$  এর মান গ্রহণ করি।

ধাপ-৩:  $F = 1$ ,  $i = 1$  ধরি।

ধাপ-৪:  $F = F * i$ ,  $i = i + 1$  নির্ণয় করি।

ধাপ-৫: যদি  $i \leq n$  হয় তবে ৪নং ধাপে যাই।

ধাপ-৬:  $F$  এর মান ছাপাই।

ধাপ-৭: প্রোগ্রাম শেষ করি।

#### if-goto ব্যবহার করে

```
#include<stdio.h>
int main()
{
    int i,F,n;
    scanf("%d",&n);
    F=1;
    i=1;
    level:
        F=F*i;
        i=i+1;
        if(i<=n) goto level;
    printf("%d ",F);
    return 0;
}
```

#### do-while ব্যবহার করে

```
#include<stdio.h>
int main()
{
    int i,F,n;
    scanf("%d",&n);
    F=1;
    i=1;
    do
    {
        F=F*i;
        i=i+1;
    } while(i<=n);
    printf("%d ",F);
    return 0;
}
```

#### for এবং while লুপ স্টেটমেন্ট ব্যবহার করে:

ধাপ-১: প্রোগ্রাম শুরু করি।

ধাপ-২:  $n$  এর মান গ্রহণ করি।

ধাপ-৩:  $F = 1$ ,  $i = 1$  ধরি।

ধাপ-৪: যদি  $i \leq n$  না হয় তবে ৬নং ধাপে যাই।

ধাপ-৫:  $F = F * i$ ,  $i = i + 1$  নির্ণয় করি। ৪নং ধাপে ফেরত যাই।

ধাপ-৬:  $F$  এর মান ছাপাই।

ধাপ-৭: প্রোগ্রাম শেষ করি।

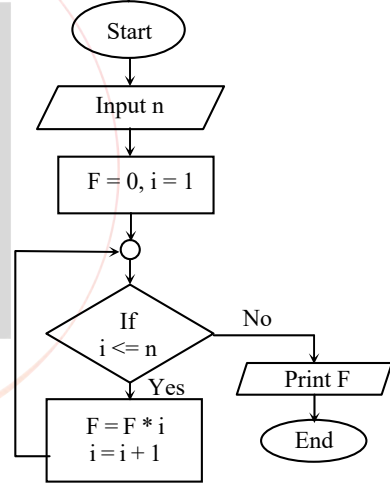
#### for ব্যবহার করে

```
#include<stdio.h>
main()
{
    int i,F,n;
    scanf("%d",&n);
    F=1;
    for(i=1;i<=n;i++)
    {
        F=F*i;
    }
    printf("%d ",F);
}
```

#### while ব্যবহার করে

```
#include<stdio.h>
int main()
{
    int i,F,n;
    scanf("%d",&n);
    F=1;
    i=1;
    while(i<=n)
    {
        F=F*i;
        i=i+1;
    }
    printf("%d ",F);
    return 0;
}
```

#### ফ্লোচার্ট:





for, while ও do- while লুপের তুলনামূলক প্রোগ্রাম দেখানো হলো:

for লুপ স্টেটমেন্ট ব্যবহার করে	while লুপ স্টেটমেন্ট ব্যবহার করে	do- while লুপ স্টেটমেন্ট ব্যবহার করে
<p>1 থেকে ১০ পর্যন্ত সংখ্যা দেখানোর জন্য প্রোগ্রাম।</p> <pre>#include&lt;stdio.h&gt; int main() {     int a;     for(a=1;a&lt;=10; a++)     {         printf("%d\t",a);     }     return 0; }</pre> <p>ফলাফল :</p> <pre>1 2 3 4 5 6 7 8 9 10</pre>	<p>1 থেকে ১০ পর্যন্ত সংখ্যা দেখানোর জন্য প্রোগ্রাম।</p> <pre>#include&lt;stdio.h&gt; int main() {     int a=1;     while(a&lt;=10)     {         printf("%d\t",a);         a++;     }     return 0; }</pre> <p>ফলাফল :</p> <pre>1 2 3 4 5 6 7 8 9 10</pre>	<p>1 থেকে ১০ পর্যন্ত সংখ্যা দেখানোর জন্য প্রোগ্রাম।</p> <pre>#include&lt;stdio.h&gt; int main() {     int a=1;     do     {         printf("%d\t",a);         a++;     } while(a&lt;=10);     return 0; }</pre> <p>ফলাফল :</p> <pre>1 2 3 4 5 6 7 8 9 10</pre>
<p>1 থেকে 15 পর্যন্ত বিজোড় সংখ্যা দেখানোর জন্য প্রোগ্রাম।</p> <pre>#include&lt;stdio.h&gt; int main() {     int a;     for(a=1;a&lt;=15; a=a+2)     {         printf("%d\t",a);     }     return 0; }</pre> <p>ফলাফল :</p> <pre>1 3 5 7 9 11 13 15</pre>	<p>1 থেকে 15 পর্যন্ত বিজোড় সংখ্যা দেখানোর জন্য প্রোগ্রাম।</p> <pre>#include&lt;stdio.h&gt; int main() {     int a=1;     while(a&lt;=15)     {         printf("%d\t",a);         a=a+2;     }     return 0; }</pre> <p>ফলাফল :</p> <pre>1 3 5 7 9 11 13 15</pre>	<p>1 থেকে 15 পর্যন্ত বিজোড় সংখ্যা দেখানোর জন্য প্রোগ্রাম।</p> <pre>#include&lt;stdio.h&gt; int main() {     int a=1;     do     {         printf("%d\t",a);         a=a+2;     } while(a&lt;=15);     return 0; }</pre> <p>ফলাফল :</p> <pre>1 3 5 7 9 11 13 15</pre>

for loop ব্যবহার করে	while ব্যবহার করে	do-while ব্যবহার করে
<p>1 থেকে 15 পর্যন্ত জোড় সংখ্যা দেখানোর জন্য প্রোগ্রাম।</p> <pre>#include&lt;stdio.h&gt; int main() {     int a;     for(a=2;a&lt;=15; a=a+2)     {         printf("%d\t",a);     }     return 0; }</pre> <p>ফলাফল :</p> <p>2 4 6 8 10 12 14</p>	<p>1 থেকে 15 পর্যন্ত জোড় সংখ্যা দেখানোর জন্য প্রোগ্রাম।</p> <pre>#include&lt;stdio.h&gt; int main() {     int a=2;     while(a&lt;=15)     {         printf("%d\t",a);         a=a+2;     }     return 0; }</pre> <p>ফলাফল :</p> <p>2 4 6 8 10 12 14</p>	<p>1 থেকে 15 পর্যন্ত জোড় সংখ্যা দেখানোর জন্য প্রোগ্রাম।</p> <pre>#include&lt;stdio.h&gt; int main() {     int a=2;     do     {         printf("%d\t",a);         a=a+2;     } while(a&lt;=15);     return 0; }</pre> <p>ফলাফল :</p> <p>2 4 6 8 10 12 14</p>
<p>1 থেকে n পর্যন্ত সংখ্যা দেখানোর জন্য প্রোগ্রাম।</p> <pre>#include&lt;stdio.h&gt; int main() {     int a, n;     printf("Value of n: ");     scanf("%d",&amp;n);     for(a=1;a&lt;=n; a=a+1)     {         printf("%d\t",a);     }     return 0; }</pre> <p>ফলাফল :</p> <p>Value of n:10</p> <p>1 2 3 4 5 6 7 8 9 10</p>	<p>1 থেকে n পর্যন্ত সংখ্যা দেখানোর জন্য প্রোগ্রাম।</p> <pre>#include&lt;stdio.h&gt; int main() {     int a, n;     printf("Value of n: ");     scanf("%d",&amp;n);     a=1;     while(a&lt;=n)     {         printf("%d\t",a);         a=a+1;     }     return 0; }</pre> <p>ফলাফল :</p> <p>Value of n:10</p> <p>1 2 3 4 5 6 7 8 9 10</p>	<p>1 থেকে n পর্যন্ত সংখ্যা দেখানোর জন্য প্রোগ্রাম।</p> <pre>#include&lt;stdio.h&gt; int main() {     int a, n;     printf("Value of n: ");     scanf("%d",&amp;n);     a=1;     do     {         printf("%d\t",a);         a=a+1;     } while(a&lt;=n);     return 0; }</pre> <p>ফলাফল :</p> <p>Value of n:10</p> <p>1 2 3 4 5 6 7 8 9 10</p>

for loop ব্যবহার করে	while loop ব্যবহার করে	do-while ব্যবহার করে
$1^2+2^2+3^2+.....+n^2$ ধারার যোগফল নির্ণয়ের জন্য প্রোগ্রাম <pre>#include&lt;stdio.h&gt; main() {     int a,s=0,n;     printf("Value of n: ");     scanf("%d",&amp;n);     for(a=1;a&lt;=n;a++)     {         s=s+a*a;     }     printf("Sum: %d",s);     return 0; }</pre> <b>ফলাফল :</b> Value of n:100 Sum:10670	$1^2+2^2+3^2+.....+n^2$ ধারার যোগফল নির্ণয়ের জন্য প্রোগ্রাম <pre>#include&lt;stdio.h&gt; int main() {     int a,s=0,n;     printf("Value of n: ");     scanf("%d",&amp;n);     a=1;     while(a&lt;=n)     {         s=s+a*a;         a=a+1;     }     printf("Sum=%d ",s);     return 0; }</pre> <b>ফলাফল :</b> Value of n:100 Sum of the series:10670	$1^2+2^2+3^2+.....+n^2$ ধারার যোগফল নির্ণয়ের জন্য প্রোগ্রাম <pre>#include&lt;stdio.h&gt; int main() {     int a,s=0,n;     printf("Value of n: ");     scanf("%d",&amp;n);     a=1;     do     {         s=s+a*a;         a=a+1;     } while(a&lt;=n);     printf("Sum=%d ",s);     return 0; }</pre> <b>ফলাফল :</b> Value of n:100 Sum:10670
এমন একটি ধারা নির্ণয় করতে হবে যার প্রথম পদ, প্রতি পদের বৃদ্ধি এবং শেষপদ কীবোর্ডের মাধ্যমে ইনপুট দিতে হবে। <pre>#include&lt;stdio.h&gt; main() {     int a,i,n,j;     printf("First term: ");     scanf("%d",&amp;a);     printf("Increment number: ");     scanf("%d",&amp;i);     printf("Last term: ");     scanf("%d",&amp;n);     printf("Series: ");     for(j=a;j&lt;=n;j=j+i)     {         printf("%d\t",j);     } }</pre>	এমন একটি ধারা নির্ণয় করতে হবে যার প্রথম পদ, প্রতি পদের বৃদ্ধি এবং শেষপদ কীবোর্ডের মাধ্যমে ইনপুট দিতে হবে। <pre>#include&lt;stdio.h&gt; main() {     int a,i,n,j;     printf("First term: ");     scanf("%d",&amp;a);     printf("Increment number: ");     scanf("%d",&amp;i);     printf("Last term: ");     scanf("%d",&amp;n);     printf("Series: ");     j=a;     while(j&lt;=n)     {         printf("%d\t",j);     } }</pre>	এমন একটি ধারা নির্ণয় করতে হবে যার প্রথম পদ, প্রতি পদের বৃদ্ধি এবং শেষপদ কীবোর্ডের মাধ্যমে ইনপুট দিতে হবে। <pre>#include&lt;stdio.h&gt; main() {     int a,i,n,j;     printf("First term: ");     scanf("%d",&amp;a);     printf("Increment number: ");     scanf("%d",&amp;i);     printf("Last term: ");     scanf("%d",&amp;n);     printf("Series: ");     j=a;     do     {         printf("%d\t",j);     } }</pre>

<pre> } }  ফলাফল : First term:2 Increment number:2 Last term:10 Series: 2 4 6 8 10 </pre>	<pre> printf("%d\t",j); j=j+i;  }  }  ফলাফল : First term:2 Increment number:2 Last term:10 Series: 2 4 6 8 10 </pre>	<pre> j=j+i; } while(j&lt;=n);  }  ফলাফল : First term:2 Increment number:2 Last term:10 Series: 2 4 6 8 10 </pre>
<b>for loop ব্যবহার করে</b>	<b>while loop ব্যবহার করে</b>	<b>do-while ব্যবহার করে</b>
<p>1.2+2.3+3.4+.....+n(n+1) ধারার যোগফল নির্ণয়ের জন্য প্রোগ্রাম</p> <pre> #include&lt;stdio.h&gt; main() {     int a,s=0,n;     printf("Value of n: ");     scanf("%d",&amp;n);     for(a=1;a&lt;=n;a++)     {         s=s+a*(a+1);     }     printf("Sum: %d",s); }  ফলাফল : Value of n:100 Sum:343400 </pre>	<p>1.2+2.3+3.4+.....+n(n+1) ধারার যোগফল নির্ণয়ের জন্য প্রোগ্রাম</p> <pre> #include&lt;stdio.h&gt; main() {     int a,s=0,n;     printf("Value of n: ");     scanf("%d",&amp;n);     a=1;     while(a&lt;=n)     {         s=s+a*(a+1);         a++;     }     printf("Sum: %d",s); }  ফলাফল : Value of n:100 Sum: 343400 </pre>	<p>1.2+2.3+3.4+.....+n(n+1) ধারার যোগফল নির্ণয়ের জন্য প্রোগ্রাম</p> <pre> #include&lt;stdio.h&gt; main() {     int a,s=0,n;     printf("Value of n: ");     scanf("%d",&amp;n);     a=1;     do     {         s=s+a*(a+1);         a++;     } while(a&lt;=n);     printf("Sum: %d",s); }  ফলাফল : Value of n: 100 Sum: 343400 </pre>
<p>কিবোর্ডের সাহায্যে গৃহীত দুটি পূর্ণ সংখ্যার গ.সা.গু নির্ণয়ের প্রোগ্রাম।</p> <pre> #include &lt;stdio.h&gt; int main() {     int l, s, i, gcd;     printf("Enter large value: "); </pre>	<p>কিবোর্ডের সাহায্যে গৃহীত দুইটি পূর্ণ সংখ্যার গ.সা.গু নির্ণয়ের প্রোগ্রাম।</p> <pre> #include&lt;stdio.h&gt; main() {     int l, s, r;     printf("Enter large value :"); </pre>	<p>কিবোর্ডের সাহায্যে গৃহীত দুইটি পূর্ণ সংখ্যার গ.সা.গু নির্ণয়ের প্রোগ্রাম।</p> <pre> #include&lt;stdio.h&gt; main() {     int l, s, r;     printf("Enter large value :"); </pre>

<pre>scanf("%d", &amp;l); printf("Enter small value: "); scanf("%d", &amp;s); for(i=1; i&lt;=l    i&lt;=s; ++i) {     if(l%i==0 &amp;&amp; s%i==0)         gcd=i; } printf("GCD=%d", gcd); return 0; }</pre> <p><b>ফলাফল :</b> Enter large value : 35 Enter small value :25 GCD = 5</p>	<pre>scanf("%d", &amp;l); printf("Enter small value :"); scanf("%d", &amp;s); while(l%s!=0) {     r = l %s;     l = s;     s = r; } printf("GCD=%d", s); }</pre> <p><b>ফলাফল :</b> Enter large value : 35 Enter small value :25 GCD = 5</p>	<pre>scanf("%d", &amp;l); printf("Enter small value :"); scanf("%d", &amp;s); do {     r = l %s;     l = s;     s = r; } while(l%s!=0); printf("GCD=%d", s); }</pre> <p><b>ফলাফল :</b> Enter large value : 35 Enter small value :25 GCD = 5</p>
---	---	---

#### লুপ সংক্রান্ত আরও কিছু প্রোগ্রাম

<p>তিনটি পূর্ণ সংখ্যার গ.সা.গু নির্ণয়ের প্রোগ্রাম।</p> <pre>#include&lt;stdio.h&gt; int main() {     int a, b,c, x, gcd;     printf("Enter three value: ");     scanf("%d %d %d", &amp;a,&amp;b,&amp;c);     x=(a&lt;b)?(a&lt;c)?a:c:(b&lt;c)?b:c;     for( ; x&gt;=1;x--)     {         if(a%x==0 &amp;&amp; b%x==0&amp;&amp; c%x==0)         {             gcd=x;             break;         }     }     printf("GCD=%d", gcd);     return 0; }</pre> <p><b>ফলাফল :</b> Enter three value: 15, 5 10 GCD=5</p>	<p>তিনটি পূর্ণ সংখ্যার ল.সা.গু নির্ণয়ের প্রোগ্রাম।</p> <pre>#include&lt;stdio.h&gt; main() {     int a,b,c,x,lcm;     printf("Type the three number:");     scanf("%d %d %d",&amp;a,&amp;b,&amp;c);     x=(a&gt;b)?(a&gt;c)?a:c:(b&gt;c)?b:c;     for(x&lt;=a*b*c;x++)     {         if((x%a==0)&amp;&amp;(x%b==0)&amp;&amp;(x%c==0))         {             lcm=x;             break;         }     }     printf("LCM is %d",lcm); }</pre> <p><b>ফলাফল :</b> Type the three number:3 9 12 LCM is 36</p>
--	--

কোনো সংখ্যা মৌলিক (Prime number) কিনা তা নির্ণয় করার প্রোগ্রাম লিখ।

```
#include<stdio.h>
main()
{
    int n, i, s;
    printf("Enter a number\n");
    scanf("%d", &n);
    for (i=2; i<=n-1; i++)
    {
        s=n%i;
        if (s==0)
        {
            printf("%d is not prime number",n);
            break;
        }
    }
    if (s!=0)
        printf("%d is prime number", n);
}
```

ফলাফল :

Enter a number  
13  
13 is prime number

কোন ফিবোনাচি সিরিজের মান নির্ণয় করার প্রোগ্রাম লিখ।

```
#include<stdio.h>
main()
{
    int n, i, a[100];
    printf("How many fibonacci number?");
    scanf("%d",&n);
    printf("Enter 1st & 2nd number : ");
    scanf("%d %d", &a[1], &a[2]),
    for(i=3; i<=n; i++)
    {
        a[i]=a[i-1]+a[i-2];
        printf("\n Fibonacci number %d", a[i]);
    }
}
```

ফলাফল :

How many fibonacci number?  
6  
Enter 1st & 2nd number :  
1 2  
Fibonacci number 3  
Fibonacci number 5  
Fibonacci number 8  
Fibonacci number 13



কাজ :

- কন্ডিশনাল কন্ট্রোল স্টেটমেন্ট এবং লুপ কন্ট্রোল স্টেটমেন্টের মধ্যে পার্থক্য লেখ।
- for loop ব্যবহার করে ইংরেজি বর্ণমালা প্রদর্শনের জন্য প্রোগ্রাম লেখ।
- for, while, do-while statement ব্যবহার করে নিম্নের মত আউটপুট পাওয়ার জন্য প্রোগ্রাম লেখ।  
Even number:  
8 16 24 32 40 48 56 64  
Odd Number:  
7 9 11 13 15 17 19 21 23 25
- if statement ব্যবহার করে নিম্নের মত আউটপুট পাওয়ার জন্য প্রোগ্রাম লেখ।  
Even number:  
2 4 6 8 10 12 14 16 18 20  
Odd Number:  
1 3 5 7 9 11 13 15 17 19
- while loop ও do-while loop এর মধ্যে পার্থক্য দেখাও।
- continue এবং goto স্টেটমেন্টের মধ্যে পার্থক্য দেখাও।
- for, while, do-while statement ব্যবহার করে নিম্নের ধারাটি প্রদর্শনের জন্য প্রোগ্রাম লেখ।  
z y x . . . a

### ৫.১২.১৬ অ্যারে (Array)

একটি চলক একক সময়ে শুধুমাত্র একটি ডেটা ধারণ করতে পারে। অনেক সময় প্রোগ্রামে একই ধরনের অনেক ডেটা নিয়ে কাজ করতে হয়। ফলে প্রত্যেকটি ডেটার জন্য আলাদা আলাদা চলক ঘোষণা করতে হয়। কিন্তু একই ধরনের অনেক ডেটার প্রত্যেকটির জন্য আলাদা আলাদা চলক বা ভ্যারিয়েবল নিয়ে কাজ করা বেশ সময় সাপেক্ষ এবং কষ্টেও কাজ যা মোটেই বাস্তব সম্মত নয়। এ সকল কাজে অ্যারে ব্যবহার করা হলে একটি অ্যারে চলক বা ভ্যারিয়েবলে অনেক ডেটা রাখা যায় এবং প্রসেস করা যায়।

একটি ভেরিয়েবলের নামের আওতায় মেমোরিতে পরপর সংরক্ষিত একই টাইপের কতগুলো ডেটার সমষ্টিকে অ্যারে বা বিন্যাস বলা হয়। অর্থাৎ একই ডেটা টাইপের কতগুলো ভেরিয়েবলের সেটকে অ্যারে বলা হয়। অ্যারে একটি ডিরাইভড ডেটা টাইপ। সাধারণ ভেরিয়েবল ঘোষণার মতো ব্যবহারের পূর্বে ডেটা টাইপসহ অ্যারে ভেরিয়েবল ঘোষণার প্রয়োজন হয়। অ্যারের উপাদানগুলো মেমোরিতে পাশাপাশি অবস্থান করে। ফলে একই টাইপের ডেটাগুলো মেমোরিতে একত্রে থাকে বিধায় প্রোগ্রাম নির্বাহ দ্রুত হয়। অ্যারে উচ্চ স্তরের ভাষার একটি অনন্য বৈশিষ্ট্য। অ্যারে ব্যবহার করে প্রোগ্রামের জটিলতা অনেকাংশে হ্রাস করা যায়। একটি অ্যারের প্রতিটি স্বতন্ত্র ভেরিয়েবলকে অ্যারে উপাদান (Array Element) বলা হয়।

যেমন, `int roll [5];`

এটি একটি একমাত্রিক অ্যারে। এখানে `int` হলো ডেটাটাইপ, `roll` হলো ভেরিয়েবলের নাম, এবং `[5]` হলো সাইজ। এই অ্যারের মধ্যে পাঁচটি উপাদান (0-4) রাখা যাবে। এই 5টা মান `roll [0]`, `roll [1]`, `roll [2]`, `roll [3]`, `roll [4]` এই পাঁচটি ঘরে থাকে। এক্ষেত্রে 0-4 সংখ্যাগুলোকে ইনডেক্স (index) বলে। কোন একটি মান পেতে হলে তার ইনডেক্স ব্যবহার করতে হয়। যেমন `roll [2]` অ্যারে তিন নম্বর ঘরের মান দেখাবে। উল্লেখ্য, অ্যারে উপাদানগুলোর সূচক শূন্য (0) থেকে শুরু হয়। এজন্য `roll` অ্যারেতে `roll [5]` নামক কোনো উপাদানের অস্তিত্ব নেই।

#### অ্যারে ভেরিয়েবলের মান নির্ধারণ

অ্যারে ভেরিয়েবলের মান নির্ধারণ বলতে অ্যারের উপাদানগুলোর জন্য মানকে বুঝায়। অ্যারের উপাদানের মান নির্ধারণের জন্য সাধারণত তিনটি পদ্ধতি অনুসরণ করা হয়। সেগুলো হলো—

- অ্যারে ঘোষণার শুরুতে
- অ্যারে ঘোষণার পরে
- প্রোগ্রাম নির্বাহের সময়ে।

**ঘোষণার সময়ে অ্যারে উপাদানের মান নির্ধারণ:** এ প্রক্রিয়ায় অ্যারে ঘোষণার সময় ভেরিয়েবলের ডেটা টাইপ অনুযায়ী দ্বিতীয় বন্ধনীর মধ্যে প্রতিটি অ্যারে উপাদানের জন্য আলাদাভাবে মান দেয়া হয়। প্রতিটি মানের মাঝে একটি করে পার্থক্যসূচক কমা বসে। এরূপে মান নির্ধারণের ফরম্যাট হলো—

**`DataType ArrayName[N] = { Value1, Value2, ....., ValueN};`**

উদাহরণ:

`char Name [6] = { 'R', 'A', 'H', 'M', 'A', 'N' };`

`char` টাইপ অ্যারের মান নিম্নলিখিতভাবেও করা যায়।

`char Name [6] = "RAHMAN";`



সাধারণত কোনো অ্যারে ঘোষণার সময়ে তার সাইজ নির্ধারণ করতে হয়। তবে এ পদ্ধতিতে কোনো অ্যারে ঘোষণার সাথে সাথে যদি দ্বিতীয় বন্ধনীর মধ্যে উপাদানগুলোর মান নির্ধারণ করা হয় তা হলে অ্যারে সাইজ না লিখলেও হয়। যেমন:

```
int Age[ ] = { 43,67,89,92,100};
```

**ঘোষণার পরে অ্যারে মান নির্ধারণ:** অ্যারে ঘোষণার পরে সাধারণ ভেরিয়েবলের মান নির্ধারণের নিয়মে ডেটা টাইপ অনুযায়ী প্রতিটি অ্যারে উপাদানের জন্য আলাদাভাবে মান দেয়া হয়। প্রতিটি মানের মাঝে একটি করে পার্থক্যসূচক সেমিকোলন বসে। এরূপে মান নির্ধারণের ফরম্যাট হলো—

```
DataType ArrayName[N];
ArrayName [0]=Value1;
ArrayName [2]=Value2;
ArrayName [3]=Value3;
... ..
ArrayName [N-1]=ValueN;
```

উদাহরণ:

```
int Age[5];
Age [0] = 20;
Age [1] = 21;
Age [2] =22;
Age [3] =23;
Age [4] =24;
```

### প্রোগ্রাম নির্বাহের সময় অ্যারে মান নির্ধারণ

প্রোগ্রাম নির্বাহের সময় scanf() ফাংশন ব্যবহার করে অ্যারে ভেরিয়েবলের ডেটা টাইপ অনুযায়ী প্রতিটি অ্যারে উপাদানের জন্য আলাদাভাবে মান দেওয়া হয়। প্রতিটি উপাদানের মান দেয়ার পর তা কার্যকরী করার জন্য এন্টার চাপতে হয় কিংবা ন্যূনতম একবার স্পেসবার চাপতে হয়। এরূপে মান নির্ধারণের ফরম্যাট হলো—

```
DataType ArrayName [N];
scanf ("FormatSpecifier",
&ArrayName[0]);
scanf ("FormatSpecifier",
&ArrayName[1]);
.....
scanf ("FormatSpecifier",
&ArrayName[N-1]);
```

উদাহরণ:

```
int Roll [5];
float Mark [5];
scanf ("%d", &Roll[0]);
scanf ("%f", &Mark[0]);
.....
scanf ("%d", &Roll[4]);
scanf ("%f", &Mark[4]);
```

অ্যারে উপাদানের মান নির্ধারণে প্রধান লক্ষ্যণীয় বিষয় হলো, প্রতিটি অ্যারে উপাদানের মান অবশ্যই অ্যারের ভেরিয়েবলের ডেটা টাইপ অনুযায়ী হতে হয়। যেমন, char টাইপ অ্যারের উপাদানের মান char টাইপ হয়, int টাইপ অ্যারের উপাদানের মান int টাইপ হয়, float টাইপ অ্যারের উপাদানের মান float টাইপ হয়, double টাইপ অ্যারের উপাদানের মান double টাইপ হয়, ইত্যাদি। তা না হলে প্রোগ্রামে ভুল আসতে পারে।

### অ্যারের বৈশিষ্ট্য:

- অ্যারের উপাদানগুলো সমগোত্রীয়।
- এটি একটি লিনিয়ার বা সরল ডেটা স্ট্রাকচার পদ্ধতি।
- অ্যারেতে একটি মাত্র ফিল্ড ব্যবহৃত হয়।
- এটির উপাদানের অ্যাড্রেস সাজানো থাকে।

### প্রোগ্রামে অ্যারে স্ট্রাকচারের সুবিধা:

- অ্যারে ব্যবহারের ফলে প্রোগ্রাম সহজ, সুন্দর ও ছোট হয়।
- সমজাতীয় অনেকগুলো ডেটাকে একটি মাত্র চলক দ্বারা প্রকাশ করা যায়।
- এটি প্রোগ্রামের জটিলতা কমায়।
- প্রোগ্রামকে সুন্দর করে।
- অ্যারে ব্যবহার করা সহজ।

### অ্যারে স্ট্রাকচারের অসুবিধা:

- এটিতে অনেক সময় মেমোরির অপচয় হয়।
- অ্যারের মধ্যস্থ কোনো ডেটা মুছতে হলে বা অ্যারের মধ্যে কোনো ডেটা সংযোজন করতে হলে অ্যারের অন্যান্য ডেটাগুলোকে স্থানান্তরের প্রয়োজন হয়।
- অ্যারেতে একই টাইপের ডেটা রাখতে হয়। ভিন্ন ডেটা টাইপের ডেটা একটি অ্যারেতে রাখা যায় না।
- প্রকৃত ডেটা অপেক্ষা অ্যারের সাইজ অনেক বেশি ঘোষণা করা হলে এক দিকে যেমন মেমোরির অপচয় হতে পারে, অপর দিকে প্রকৃত ডেটা অপেক্ষা অ্যারের সাইজ কম ঘোষণা করা হলে অ্যারেতে ডেটার পর্যাপ্ত স্থান সংকুলান হয় না।

### অ্যারের ডাইমেনশন বা মাত্রা

একটি অ্যারের যেকোনো উপাদানকে শনাক্ত করার জন্য যতগুলো সংখ্যা প্রয়োজন হয় তাকে ঐ অ্যারের ডাইমেনশন বা মাত্রা বলে। `int roll[50];` অ্যারের উপাদানের যেকোনো একটি শনাক্ত করার জন্য কেবল একটি সংখ্যা (০ থেকে ৪৯ এর মধ্যবর্তী) প্রয়োজন হবে। এ জন্য এই অ্যারের মাত্রা এক, অর্থাৎ এটি একটি একমাত্রিক অ্যারে। একমাত্রিক অ্যারের উপাদানগুলো টেবিলে কেবল একটি একক সারি বা কলাম আকারে উপস্থাপন করা যায়।

### চলক ও অ্যারের মধ্যে পার্থক্য:

চলক	অ্যারে
১. মেমোরি লোকেশনের নাম বা ঠিকানাকে চলক বলে।	১. একই ধরনের বা সম প্রকৃতির ডেটার সমাবেশকে অ্যারে বলে।
২. ডেটা চলকের নাম যেকোনো আকারের হতে পারে।	২. অ্যারের একটি নাম থাকে এবং এর সদস্য বা আইটেমসমূহকে বন্ধনীর মধ্যে রাখা সংখ্যা দিয়ে চিহ্নিত করা হয়।
৩. চলক একটি মুহূর্তে শুধু একটি মান ধারণ করতে পারে।	৩. অ্যারে একটি মুহূর্তে একের অধিক মান ধারণ করতে পারে।

### অ্যারের প্রকারভেদ

অ্যারে প্রধানত দু'প্রকার। যথা: ১। একমাত্রিক ও ২। বহুমাত্রিক

**একমাত্রিক অ্যারে (One dimensional array) :** অ্যারের অন্তর্ভুক্ত ডেটাগুলো যদি একটিমাত্র কলাম বা রো (সারি) আকারে থাকে তখন তাকে একমাত্রিক অ্যারে বলা হয়। একমাত্রিক অ্যারের গঠন নিম্নরূপ:

`Data_type Array_name [array_size];`

এখানে, `Data_type` যেকোনো বৈধ ডেটা টাইপ `Array_name` প্রোগ্রামার কর্তৃক দেয়া অ্যারে ভেরিয়েবলের যেকোনো বৈধ নাম। `Array_size` পূর্ণসংখ্যায় প্রকাশিত কোনো ধ্রুবক, যাকে অ্যারে সাইজ বা অ্যারে ইনডেক্স (index)

বলা হয়। অ্যারের নাম সংলগ্ন তৃতীয় বন্ধনীয় '[]' মধ্যে পূর্ণসংখ্যা দ্বারা প্রকাশিত অ্যারে সাইজ বা অ্যারে ইনডেক্স অ্যারের সাইজ নির্ধারণ করে, যা অ্যারে ভেরিয়েবলে সংরক্ষিত সর্বোচ্চ ডেটার সংখ্যা নির্দেশ করে।

উদাহরণ :

```
int Roll[10]; //int type array
Char Name [20]; //char type array
float marks[10]; //float type array
```

অ্যারেটিতে ডেটাগুলো নিম্নরূপে সজ্জিত থাকে।

6	3	4	5	9
roll[0]	roll[1]	roll[2]	roll[3]	roll[4]

উদাহরণ-১. একটি একমাত্রিক অ্যারেতে ৫টি সংখ্যা রেখে উক্ত ৫টি সংখ্যা প্রিন্ট করার প্রোগ্রাম লিখ।

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int j;
    int marks[5]={11, 23, 35, 42, 36};
    printf("One Dimensional Array Elements\n");
    for(j=0;j<5;j++)
    {
        printf("%d\t",marks[j]);
    }
    printf("\n") ;
    getch();
}
```

ফলাফল : One Dimensional Array Elements

11    23    35    42    36

উদাহরণ-২. n সংখ্যক সংখ্যার যোগফল নির্ণয়ের জন্য প্রোগ্রাম লিখ।

```
#include<stdio.h>
#include<conio.h>
main()
{
    int a[10],i,n,s=0;
    printf("How many number:");
    scanf("%d",&n);
    printf("Type the marks: ");
    for(i=1;i<=n;++i)
    {
        scanf("%d",&a[i]);
        s=s+a[i];
    }
    printf("SUM=%d",s);
    getch();
}
```

### ফলাফল

```
How many Number : 3
Type the marks : 8
Type the marks : 6
Type the marks : 10
Sum = 24
```

উদাহরণ-৩. n সংখ্যক সংখ্যা ইনপুট দিয়ে তার মধ্যে থেকে বড় সংখ্য নির্ণয়ের প্রোগ্রাম লিখ।

```
#include<stdio.h>
#include<conio.h>
main()
{
    int a[100],n,max,i;
    printf("Please type the total no of numbers: ");
    scanf("%d",&n);
    //Inserting data into the array.
    for(i=0;i<n;i++)
    {
        printf("\nEnter the number (%d of %d): ",i+1,n);
        scanf("%d",&a[i]);
    }
    //Finfding highest number.
    max=a[0];
    for(i=1;i<n;i++)
    {
        if(max<a[i])
            max=a[i];
    }
    printf("\nLargest number=%d",max);
    return 0;
    getch();
}
```

### ফলাফল :

```
Please type the total no of numbers:5
Enter the number (1 of 5):12
Enter the number (2 of 5):10
Enter the number (3 of 5):20
Enter the number (4 of 5):30
Enter the number (4 of 5):25
```

Largest number=30

উদাহরণ: int roll[4][3];

(1,1)	(1,2)	(1,3)	(1,4)
10	20	30	40
(2,1)	(2,2)	(2,3)	(2,4)
50	60	70	80
(3,1)	(3,2)	(3,3)	(3,4)
90	100	110	120

**বহুমাত্রিক অ্যারে:** যে অ্যারের কোনো উপাদানকে শনাক্ত করার জন্য দুই বা অধিক সংখ্যার প্রয়োজন হয় তাকে বহুমাত্রিক অ্যারে বলা হয়। বহুমাত্রিক অ্যারে দুই প্রকার। যথা: দ্বি-মাত্রিক ও ত্রি-মাত্রিক।

**দ্বি-মাত্রিক অ্যারে (Two dimensional array):** যে অ্যারেতে ডেটাগুলো একই সাথে কলাম ও রো আকারে উপস্থাপন করা হয়, তাকে দ্বি-মাত্রিক অ্যারে বলে। দ্বি-মাত্রিক অ্যারেতে দুটি সংখ্যা ব্যবহার করা হয়। প্রথম সংখ্যাটি রো এবং দ্বিতীয় সংখ্যাটি কলাম নির্দেশ করে। একটি দ্বি-মাত্রিক অ্যারেতে রো ও কলামের গুণফলের সমপরিমাণ ডেটা রাখা যাবে। Roll[20][15] একটি দ্বি-মাত্রিক অ্যারে যেখানে ৩০০টি ডেটা রাখা যাবে। দ্বি-মাত্রিক অ্যারের গঠন নিম্নরূপ:

Data\_type Array\_name[rowsize][columnsize];

যেমন- `int marks[2][3]`; যথা : `marks[0][0]`, `marks[0][1]`, `marks[0][2]`, `marks[1][0]`, `marks[1][1]`, কিংবা `marks[1][2]`, এই ৬টি উপাদানের যে কোনটি নির্দেশ করার জন্য মোট দুটি সংখ্যা যেমন- ০,০ বা ০,১ বা ০,২ বা ১,০ বা ১,১ বা ১, ২ প্রয়োজন হবে। প্রোগ্রামে মূলত ম্যাট্রিক্স অর্থাৎ সারি ও কলাম সম্পর্কিত কাজের জন্য দ্বিমাত্রিক অ্যারে ব্যবহার করা হয়। ফলে অনেক সহজে সমস্যার সমাধান করা যায়। যেমন, কোনো পরীক্ষায় পাঁচটি বিষয় আছে এবং প্রত্যেক বিষয়ের প্রাপ্ত নম্বর হিসেব করে ফলাফল নির্ণয় করার জন্য দ্বিমাত্রিক অ্যারে ব্যবহার করা যেতে পারে।

**উদাহরণ.** একটি দ্বিমাত্রিক অ্যারেতে ১২টি সংখ্যা (৩টি রো ও ৪টি কলামে) রেখে উক্ত ১২টি সংখ্যা প্রিন্ট করার প্রোগ্রাম।

```
#include<stdio.h>
#include<conio.h>
void main ( )
{
    int i, j;
    int marks[3][4]={ {1, 2, 3, 4}, {5, 6, 7, 8}, {9, 10, 11, 12}};
    printf("Two Dimensional Array Elements\n");
    for(i=0;i<3;i++)
    {
        for(j=0;j<4;j++)
        {
            printf("%d\t",marks[i][j]);
        }
        printf("\n");
    }
    getch ( );
}
```

**ফলাফল:** Two Dimensional Array Elements

1	2	3	4
5	6	7	8
9	10	11	12

**ত্রি-মাত্রিক অ্যারে (Three dimensional array):** যে অ্যারের কোনো উপাদানকে সনাক্ত করার জন্য দুই বা অধিক সংখ্যার প্রয়োজন হয় তাকে বহুমাত্রিক অ্যারে বলা হয়। এরূপ অ্যারেতে দুই বা ততোধিক সারি, কলাম ছাড়াও অপর এক বা একাধিক বৈশিষ্ট্য (মাত্রা) থাকতে পারে। এরূপ উচ্চতর মাত্রাবিশিষ্ট অ্যারেকে বহুমাত্রিক অ্যারে বলা হয়। কতগুলো বিশেষ কাজে এ ধরনের অ্যারে ব্যবহার করা হয়। যেমন, দৈর্ঘ্য, প্রস্থ ও উচ্চতাবিশিষ্ট কোনো বস্তুর আয়তন নির্ণয়ের জন্য ত্রিমাত্রিক অ্যারে ব্যবহার করা যেতে পারে। তবে, এরূপ অ্যারে ব্যবহার ও নিয়ন্ত্রণ একটু জটিল।

বহুমাত্রিক অ্যারের একটি উদাহরণ হলো : `int cule[2][4][5]; // 3-dim array`



**কাজ:**

১. অ্যারে ব্যবহার করে ফিবোনাকি সিরিজের মান নির্ণয়ের প্রোগ্রাম লেখ।
২. এমন একটি প্রোগ্রাম লেখ যেখানে ৩টি অ্যারে ব্যবহার করা হবে। ১মটিতে নাম, ২য়টিতে রোল ও ৩য়টিতে ঠিকানা ইনপুট দিতে হবে এবং পরে তা প্রদর্শন করতে হবে।

## পাঠ ৩১ ও ৩২

## ব্যবহারিক: ফাংশন ও ফাংশনের ব্যবহার

### ৫.১২.১৪ ফাংশন (Function)

প্রোগ্রামে নির্দিষ্ট কাজ সম্পাদনের জন্য কতগুলো স্টেটমেন্ট একটি ব্লকের মধ্যে যে নামে রাখা হয় তাকে ফাংশন বলা হয়। প্রতিটি সি প্রোগ্রাম এরূপ এক বা একাধিক ফাংশনের সমষ্টি। ফাংশন চেনার সহজ উপায় হলো ফাংশনের মানের শেষে এক জোড়া প্রথম বন্ধনী '()' থাকে, এই প্রথম বন্ধনীর মধ্যে অনেক কিছু থাকতে পারে, আবার নাও থাকতে পারে। ফাংশনের প্রথম বন্ধনীর মধ্যবর্তী মানকে আরগুমেন্ট (Argument) বা প্যারামিটার বলা হয়। প্রতিটি ফাংশনের একটি নাম থাকে, যে নামে কম্পাইলার তাকে শনাক্ত করে। প্রোগ্রাম নির্বাহের সময়ে কম্পাইলার যখন কোনো ফাংশন কল করে তখন মূল প্রোগ্রামের কাজ স্থগিত রেখে কল্ড ফাংশনে নির্বাহ শুরু করে। 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।

#### ফাংশনের প্রয়োজনীয়তা:

- প্রোগ্রামকে সংক্ষিপ্ত করে।
- প্রোগ্রাম ডিবাগিং সহজতর হয়।
- ব্যবহারকারী তার প্রয়োজনানুযায়ী ফাংশন তৈরি করে কার্য সম্পাদন করতে পারে।
- একই ফাংশন বিভিন্ন প্রোগ্রামে ব্যবহার করা যায়।
- প্রোগ্রামের দৈর্ঘ্য ছোট হয় ফলে মেমোরি স্পেস কম লাগে।
- প্রোগ্রামের পুনরাবৃত্তিমূলক নির্দেশনাকে ফাংশন ব্যবহারের মাধ্যমে সহজ করা যায়।
- প্রোগ্রাম রচনার ক্ষেত্রে অপেক্ষাকৃত কম সময় লাগে।
- প্রোগ্রাম সহজপাঠ্য হওয়ায় ভুল বের করা, সেগুলো পৃথক ও সংশোধন করা সহজ হয়।
- বড় প্রোগ্রামের ক্ষেত্রে ছোট ছোট প্রোগ্রাম মডিউলে বিভক্ত করা যায় বলে প্রোগ্রাম সহজবোধ্য হয়।
- একই ফাংশনকে ভিন্ন ভিন্ন ইনপুট ডেটা দিয়ে বারবার ব্যবহার করা যায়।

#### ফাংশনের প্রকারভেদ

‘সি’ প্রোগ্রামে ফাংশন সমূহকে প্রধান দুই ভাগে ভাগ করা যায়। যথা-

১. লাইব্রেরি ফাংশন (Library Function);
২. ইউজার-ডিফাইন্ড ফাংশন (User Defined Function)

**লাইব্রেরি ফাংশন (Library Function) :** সি কম্পাইলারে কতগুলো বিল্ট-ইন ফাংশন থাকে সেগুলোকে লাইব্রেরি ফাংশন বলা হয়। লাইব্রেরি ফাংশনগুলো তাদের নিজস্ব ফরম্যাট অনুযায়ী main() ফাংশনের মধ্যে ব্যবহার করা যায়। printf(), scanf(), getch(), abs(), sqrt(), clock(), time(), sin(), cos(), tan() ইত্যাদি বহুল ব্যবহৃত কয়েকটি লাইব্রেরি ফাংশনের উদাহরণ। প্রোগ্রামে বিভিন্ন ধরনের গাণিতিক, যৌক্তিক ও অন্যান্য কার্যক্রম সম্পাদনের জন্য লাইব্রেরি ফাংশন ব্যবহৃত হয়। লাইব্রেরি ফাংশনকে বিল্ট-ইন ফাংশনও বলা হয়। স্ট্রিং ভেরিয়েবলের ইনপুট অপারেশনে বহুল ব্যবহৃত লাইব্রেরি ফাংশনের মধ্যে scanf(), gets(), getchar(), getch(), getche() ইত্যাদি অন্যতম। আউটপুট অপারেশনে বহুল ব্যবহৃত লাইব্রেরি ফাংশনের মধ্যে printf(), puts(), putchar(), putch() ইত্যাদি অন্যতম।

তথ্য ও যোগাযোগ প্রযুক্তি (বোর্ড)-৩২ক

**প্রোগ্রামে কোনো লাইব্রেরি ফাংশন ব্যবহারের পদ্ধতি:** লাইব্রেরি ফাংশনগুলোর ব্যবহার সহজ হওয়ায় শুধুমাত্র ফাংশনের ব্যবহারবিধি ও ফরম্যাট জানা থাকলেই হয়। লাইব্রেরি ফাংশনগুলো ঘোষণা তাদের হেডার (.h) ফাইলে ও বিস্তারিত বর্ণনা সংশ্লিষ্ট লাইব্রেরি (.Lib) ফাইলে দেয়া থাকে। এজন্য সি প্রোগ্রামে কোনো লাইব্রেরি ফাংশন ব্যবহার করলে প্রোগ্রামের শুরুতেই #include ডিরেক্টিভ স্টেটমেন্টের সাহায্যে সংশ্লিষ্ট হেডার ফাইল সংযুক্ত করতে হয়। প্রোগ্রামে কোনো হেডার ফাইল সংযুক্ত করা হলে কম্পাইলার প্রোগ্রাম কম্পাইল করার সময় সংযুক্ত ফাইলের উপাদানগুলো সংশ্লিষ্ট লাইব্রেরি ফাইল থেকে কপি করে। কোনো লাইব্রেরি ফাংশনের হেডার ফাইল ও ব্যবহার বিধি কী তা জানার জন্য ঐ লাইব্রেরি ফাংশনের উপর কার্সর রেখে বা সিলেক্ট করে Alt+F1 বা Ctrl+F1 চাপলে হেল্প ফাইল থেকে তা জানা যায়।

**সি প্রোগ্রামে main () ফাংশনের গুরুত্ব:** প্রতিটি সি প্রোগ্রামে main () নামে একটি ইউজার-ডিফাইন্ড ফাংশন থাকে। প্রোগ্রাম নির্বাহের শুরুতে main () ফাংশন স্বয়ংক্রিয়ভাবে নিয়ন্ত্রিত হয়। প্রয়োজনে এক বা একাধিক ফাংশন নিয়ন্ত্রণ করে। প্রোগ্রামে একটিই main () ফাংশন থাকে। main () ফাংশন ছাড়া অন্যান্য ফাংশন যতবার প্রয়োজন কল করা যায়। সুতরাং সি প্রোগ্রামে যত ফাংশনই থাকুক না কেন main () ফাংশনকে ঘিরেই যাবতীয় কার্যক্রম পরিচালিত হয়।

**ইউজার ডিফাইন্ড ফাংশন (User Defined Function):** কম্পাইলারে বিল্ট-ইন বা লাইব্রেরি ফাংশন থাকা সত্ত্বেও প্রোগ্রাম রচনার সময় সব রকম ফাংশন পাওয়া যায় না। সেক্ষেত্রে প্রোগ্রামার তার নিজস্ব প্রয়োজন অনুযায়ী যে সকল ফাংশন তৈরি করে তাকে ইউজার-ডিফাইন্ড বা ব্যবহারকারী বর্ণিত ফাংশন বলা হয়। ইউজার-ডিফাইন্ড ফাংশন আকার-আকৃতি ও সমস্যার ধরন এবং সমাধানের কৌশলের ওপর নির্ভর করে। একটি নির্দিষ্ট কাজের জন্য ভিন্ন ভিন্ন প্রোগ্রামার কর্তৃক ব্যবহৃত ফাংশনগুলো নামে ও বর্ণনায় ভিন্ন ভিন্ন হতে পারে। ইউজার-ডিফাইন্ড ফাংশনের নাম একটি আইডেন্টিফায়ার। সুতরাং আইডেন্টিফায়ার নামকরণের নিয়ম অনুযায়ী ফাংশনের যেকোনো বৈধ নাম দেয়া যেতে পারে। একটি ইউজার ডিফাইন্ড ফাংশন কতকগুলো স্টেটমেন্ট নিয়ে গঠিত হয়। সামান্য কিছু ব্যতিক্রম ছাড়া প্রতিটি স্টেটমেন্ট সেমিকোলন দিয়ে শেষ হয়। ইউজার-ডিফাইন্ড ফাংশনের বর্ণনা main() ফাংশনের উপরে কিংবা নিচে থাকে কিন্তু ভিতরে নয়।

**ইউজার-ডিফাইন্ড ফাংশন নামকরণের নিয়মাবলী:**

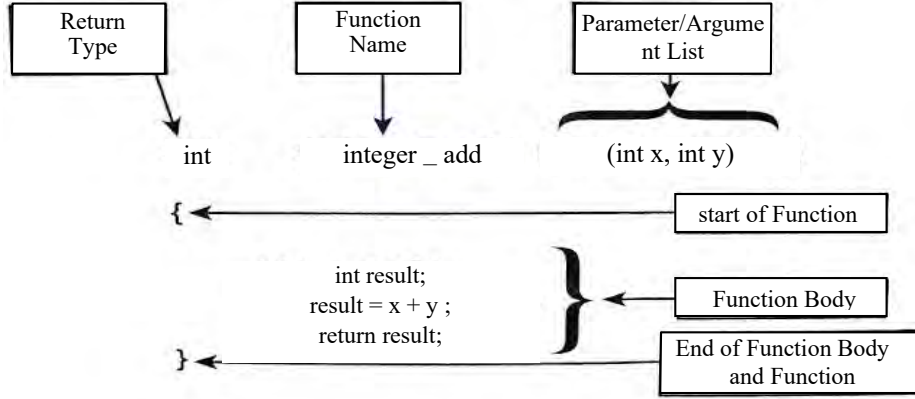
- প্রথম অক্ষর অবশ্যই অ্যালফাবেট হতে হয়।
- ফাংশনের নামের প্রথম অক্ষর কোনো সংখ্যা দেওয়া যায় না।
- কোনো বিশেষ চিহ্ন ব্যবহার করা যায় না।
- লাইব্রেরি ফাংশন বা কী-ওয়ার্ড এর নাম একই হতে পারে না।
- ফাংশনের নামের শেষে '()' দিতে হয়।
- ফাংশনের নাম তার কাজের সাথে মিল রেখে দেওয়া উচিত তবে এতে বাধ্যবাধকতা নেই।

সি প্রোগ্রামে ইউজার ডিফাইন্ড ফাংশনকে ব্যবহার করতে হলে চারটি বিষয়কে বিবেচনা করতে হয়। যাকে ফাংশনের মূল উপাদান বলা হয়। যথা—

- **ফাংশন ডেফিনেশন:** ফাংশন ডেফিনেশনকে ফাংশন বডিও বলা হয়। কোন ফাংশন কী কাজ করবে, তা ফাংশন ডেফিনেশনের মধ্যে বর্ণনা করতে হয়। অর্থাৎ ফাংশন ডেফিনেশনের মধ্যেই প্রোগ্রামের কোড লিখতে হয়। ইউজার-ডিফাইন্ড ফাংশন ঘোষণার ফরম্যাট হলো—

```
Return Type FunctionName (ArgumentList)
{
    // FunctionBody
    // ReturnStatement (Depends on Return Type)
}
```





উপরোক্ত গঠন থেকে আমরা দেখতে পাই, প্রতিটি ফাংশন ডেফিনেশনের প্রথম লাইনটি হলো ফাংশন হেডার। ফাংশন হেডার এর তিনটি অংশ থাকে।

১. **Return Type:** যে ডেটা ফাংশন রিটার্ন করে তা নির্দেশ করে।
  ২. **Function Name:** ইউজার ডিফাইন ফাংশনের নাম নির্দেশ করে।
  ৩. **Parameter/Argument List:** প্যারামিটার টাইপের সংখ্যা নির্দেশ করে। ফাংশন ডেফিনেশনে যে প্যারামিটার ব্যবহার করা হয় তাই হলো Formal প্যারামিটার। আর ফাংশন কলিং এর সময় যে আরগুমেন্ট বা প্যারামিটার দেওয়া হয় তা হলো Actual প্যারামিটার।
- **ফাংশন প্রোটোটাইপ:** কোনো প্রোগ্রামে ফাংশন ব্যবহার করা হলে, প্রোগ্রামের শুরুতেই সেই ফাংশন সম্পর্কে সংক্ষিপ্ত বর্ণনা দেয়া হয় যাতে কম্পাইলার বুঝতে পারে যে, এই নামের বা গঠনের ফাংশন পরে কোথাও ডিফাইন করা হয়েছে।
  - **ফাংশন কলিং:** call মানে ডাকা। যখন একটি ফাংশন অপর কোনো ফাংশন ব্যবহার করে তখন তাকে ব্যবহারকারী বা মূল ফাংশন বলে। যে ফাংশন ব্যবহার করা হয় তাকে ব্যবহৃত বা কল ফাংশন বলা হয়। প্রোগ্রামে যখন কোনো নির্দিষ্ট কাজ করার প্রয়োজন হয় তখন শুধুমাত্র ফাংশনের নাম লিখলেই সে কাজটি করে দেয়। এটাই হলো ফাংশন কলিং।
  - **ফাংশন রিটার্ন:** সি প্রোগ্রামের প্রতিটি ফাংশনই আলাদাভাবে ভেল্যু রিটার্ন করে। তবে void ফাংশনের শুরুতে উল্লেখ করা থাকলে আর কোনো ফাংশন ভেল্যু রিটার্ন করে না। মোটকথা কোনো ফাংশনের ভেল্যু তার কলিং ফাংশনে ফেরত যাওয়াকেই ফাংশন রিটার্ন বলে। ফাংশন রিটার্নের সাধারণ নিয়ম হলো-

return return\_value ; অথবা return ( return\_value ) ;

একটি ইউজার ডিফাইন্ড ফাংশনের উদাহরণ নিচে দেখানো হলো:

```
#include <stdio.h>
int add(int a, int b);           /*function prototype*/
main()
{
    int a,b;
    printf("Type the first number: ");
    scanf("%d",&a);
    printf("Type the second number: ");
    scanf("%d",&b);
    printf("Sum=%d\n",add(a,b));    /*function call*/
}
```

```

    }
    int add(int a, int b)          /*function call*/
    {
        int add;
        add=a+b;
        return add;
    }

```

প্রোগ্রামটিতে add() নামে একটি ইউজার ডিফাইন্ড ফাংশন তৈরি করা হয়েছে যা main() ফাংশনে call করে প্রয়োগ দেখানো হয়েছে।

ফলাফল : Type the first number:10↵  
Type the second number:6↵  
Sum=16

ফাংশন ব্যবহার করে দুটি সংখ্যার মধ্যে বড় সংখ্যা নির্ণয় করার জন্য প্রোগ্রাম লিখ।

```

#include<stdio.h>
int max(int x, int y)
{
    int z;
    z=(x>y)?x:y;
    return (z);
}
main()
{
    int a,b,c;
    printf("Type two number: ");
    scanf("%d %d",&a,&b);
    c=max(a,b);
    printf("Maximum=%d",max(a,b));
}

```

### রিকার্সিভ ফাংশন

একটি ফাংশন অন্য কোনো ফাংশনকে কল করতে পারে। সি-তে রিকার্সিভ নামে এক বিশেষ ধরনের ফাংশন ব্যবহৃত হয় যা প্রয়োজনে নিজেই নিজেকে কল করতে পারে। অর্থাৎ যখন কোনো ফাংশন নিজেই নিজেকে কল করে তখন তাকে রিকার্সিভ ফাংশন বলা হয়। ফাংশন কলকে বলে রিকার্সিভ কল। রিকার্সিভ ফাংশন একটি নির্দিষ্ট অবস্থা পর্যন্ত নিজেকে কল করতে পারে। ফলে একটি একটি শিকল এর সৃষ্টি হয়। রিকার্সিভ ফাংশনের মধ্যে এমন একটি ব্যবস্থা থাকতে হবে যাতে এক পর্যায়ে গিয়ে রিকার্সন শেষ হয়। অর্থাৎ আর রিকার্সন কল ঘটে না। এই ব্যবস্থাকে বলে রিকার্সনের টার্মিনেটিং কন্ডিশন।

### রিকার্সিভ ফাংশনের বৈশিষ্ট্য:

- রিকার্সিভ ফাংশন নিজেই নিজেকে কল করে।
- প্রতিটি রিকার্সিভ ফাংশনের একটি টার্মিনেটিং কন্ডিশন থাকতেই হবে; তা না হলে রিকার্সিভ কল চলতেই থাকে।
- রিকার্সিভ ফাংশনের নির্বাহের ফ্লো সর্বদা টার্মিনেটিং কন্ডিশনের দিকে অগ্রসর হয়।

রিকার্সিভ ফাংশন ব্যবহারের সুবিধা:

- প্রোগ্রামের জটিলতা অনেক কমে যায়।
- অনেক কম সংখ্যক ভেরিয়েবলের প্রয়োজন হয় এবং প্রোগ্রাম সহজ ও সুন্দর হয়।

রিকার্সিভ ফাংশন ব্যবহারের অসুবিধা:

- প্রোগ্রাম সম্পাদনের সময় অনেক বাড়িয়ে দেয়।
- রিকার্সিভ ফাংশন ব্যবহার করলে লুপের সংখ্যা বেড়ে যায়।

রিকার্সিভ ফাংশনের উদাহরণ নিচে দেখানো হলো:

```
#include<stdio.h>
long int factorial(int n);          /*function prototype*/
main()
{
    int n;
    printf("Type the desire value : ");
    scanf("%d",&n);
    printf("Factorial value is %ld\n",factorial(n));
}
long int factorial(int n)
{
    if (n<=1)
        return (1);
    else
        return (n*factorial(n-1));
}
```

ফলাফল : Type the desire value : 6↵  
Factorial value is 720



কাজ:

১. ফাংশন ব্যবহার করে তিনটি সংখ্যার মধ্যে বড় সংখ্যা নির্ণয়ের প্রোগ্রাম রচনা কর।
২. একটি প্রোগ্রাম লেখ যেখানে ২টি User-defined function ব্যবহার করতে হবে। একটি যোগের জন্য ও অন্যটি বিয়োগের জন্য।



## এ অধ্যায়ের প্রধান প্রধান শব্দভিত্তিক সারসংক্ষেপ

প্রোগ্রাম	কম্পিউটার বুঝতে পারে এমন নির্দেশমালাকে বলা হয় প্রোগ্রাম।
অনুবাদক প্রোগ্রাম	যে প্রোগ্রামের সাহায্যে সোর্স প্রোগ্রামকে অবজেক্ট প্রোগ্রামে পরিণত করা হয় তাকে বলা হয় অনুবাদক প্রোগ্রাম।
ইন্টারপ্রেটার	ইন্টারপ্রেটারও কম্পাইলারের মতো হাইলেভেল ভাষাকে মেশিন ভাষায় রূপান্তর করে। ইন্টারপ্রেটার লাইন নির্বাহ করে এবং তাৎক্ষণিক ফলাফল প্রদর্শন করে।
কম্পাইলার	কম্পাইলার হলো এক ধরনের অনুবাদক যা হাইলেভেল ভাষায় লিখিত প্রোগ্রামকে মেশিন ভাষায় রূপান্তর করে। অর্থাৎ সোর্স প্রোগ্রামকে অবজেক্ট প্রোগ্রামে রূপান্তর করে।
অ্যালগরিদম	কোনো একটি নির্দিষ্ট সমস্যা সমাধানের জন্য ধাপে ধাপে সমাধান করার যে পদ্ধতি তাকে অ্যালগরিদম বলে।
ফ্লোচার্ট	যে চিত্রভিত্তিক পদ্ধতিতে বিশেষ কতকগুলো চিহ্নের সাহায্যে কোনো একটি নির্দিষ্ট সমস্যার সমাধান করা হয় তাকে ফ্লোচার্ট বলা হয়।
ডেটা টাইপ	C প্রোগ্রামে অনেক ধরনের ডেটা নিয়ে কাজ করা হয়। যেমন— পূর্ণ সংখ্যা, ভগ্নাংশ, ক্যারেক্টার, স্ট্রিং ইত্যাদি। এদেরকে সাধারণভাবে ডেটাটাইপ বলে।
রাশিমালা	বিভিন্ন রাশির সমন্বিত অবস্থাকে রাশিমালা বা এক্সপ্রেশন বলে। একটি রাশিমালা বিভিন্ন চিহ্ন বা প্রতীক ব্যবহার করে তৈরি হয়। এই চিহ্ন বা প্রতীকগুলোকে বলা হয় অপারেটর।
কী-ওয়ার্ড	প্রত্যেক প্রোগ্রামিং ভাষার নিজস্ব কিছু সংরক্ষিত শব্দ আছে যা প্রোগ্রাম রচনার সময় ব্যবহার করা হয়। এই সংরক্ষিত শব্দগুলোকে কী-ওয়ার্ড বলা হয়। C প্রোগ্রামে ৩২টি সংরক্ষিত কী-ওয়ার্ড আছে।
চলক	ডেটাকে মেমোরিতে রাখার জন্য যে নাম ব্যবহার করা হয় তাকে চলক বলা হয়।
ধ্রুবক	যে রাশির মান পরিবর্তন হয় না তাকে ধ্রুবক বলে।
অপারেটর	রাশিমালায় ব্যবহৃত চিহ্ন বা প্রতীককে অপারেটর বলা হয়।
লুপিং	প্রোগ্রামের অংশ বিশেষ নির্দিষ্ট সংখ্যক বার কোনো শর্তে না পৌঁছা পর্যন্ত পুনরাবৃত্তি করাকে লুপিং বলে।
অ্যারে	অ্যারে হলো একই ধরনের ডেটার জন্য ব্যবহৃত চলকের একটি সিরিজ।



## অনুশীলনী

### ক. বহুনির্বাচনি প্রশ্ন

#### ▶ প্রোগ্রামের ধারণা ও ভাষা

##### সাধারণ বহুনির্বাচনি প্রশ্ন

১. গঠন বিচারে ও বৈশিষ্ট্য অনুযায়ী প্রোগ্রামের ভাষাকে কয় ভাগে ভাগ করা যায়?  
ক. ২ ভাগে                      খ. ৩ ভাগে  
গ. ৪ ভাগে                      ঘ. ৫ ভাগে
২. প্রোগ্রামের ভিত্তি কোনটি?  
ক. অ্যালগরিদম              ক. ডিবাগিং  
গ. টেস্টিং                      ঘ. সুডোকোড
৩. সফটওয়্যার তৈরির জন্য কী প্রয়োজন?  
ক. হার্ডওয়্যার              খ. প্রোগ্রাম  
গ. কম্পিউটার ভাষা      ঘ. ফার্মওয়্যার
৪. সমস্যা সমাধানের জন্য কম্পিউটারের ভাষায় ধারাবাহিকভাবে সাজানোর নির্দেশমালাকে কী বলে?  
ক. হার্ডওয়্যার              খ. ফার্মওয়্যার  
গ. প্রোগ্রাম                      ঘ. সফটওয়্যার
৫. কম্পিউটারের অভ্যন্তরে দুটি সংকেত কী কী?  
ক. ০ ও ১                      খ. ১ ও ২  
গ. ০ ও ২                      ঘ. ০ ও ৩
৬. কম্পিউটারে ব্যবহৃত প্রথম প্রজন্মের ভাষা সাল কত?  
ক. ১৯৪৫-১৯৪৯              খ. ১৯৫০-১৯৫৯  
গ. ১৯৬০-১৯৬৯              ঘ. ১৯৭০-১৯৭৯

##### বহুপদী সমাপ্তিসূচক প্রশ্ন

৭. কম্পিউটার বুঝতে পারে—  
i. ০  
ii. ১  
iii. ১০  
নিচের কোনটি সঠিক?  
ক. i ও ii                      খ. i ও iii  
গ. ii ও iii                      ঘ. i, ii ও iii
৮. একটি আদর্শ প্রোগ্রামের বৈশিষ্ট্য হচ্ছে —  
i. তথ্য প্রদানের ব্যবস্থা থাকতে হবে  
ii. তথ্য প্রক্রিয়াকরণের ব্যবস্থা থাকতে হবে  
iii. ফলাফল প্রাপ্তির সুবিধা থাকতে হবে  
নিচের কোনটি সঠিক?  
ক. i ও ii                      খ. i ও iii  
গ. ii ও iii                      ঘ. i, ii ও iii

#### ▶ মেশিন ভাষা ও অ্যাসেম্বলি ভাষা

##### সাধারণ বহুনির্বাচনি প্রশ্ন

৯. সরাসরি কোন ভাষা কম্পিউটারে ব্যবহার করা হয়?  
ক. অ্যাসেম্বলি ভাষা              খ. যান্ত্রিক ভাষা  
গ. উচ্চস্তরের ভাষা              ঘ. অতি উচ্চস্তরের ভাষা
১০. কম্পিউটারের মৌলিক ভাষা কোনটি?  
ক. মেশিন ভাষা                      খ. অ্যাসেম্বলি ভাষা  
গ. দ্বিতীয় প্রজন্মের ভাষা      ঘ. পঞ্চম প্রজন্মের ভাষা
১১. যান্ত্রিক ভাষার প্রধান উপকরণ নিচের কোনটি?  
ক. নিজস্ব ভাষা                      খ. অ্যাসেম্বলি ভাষা  
গ. মানুষের ভাষা                      ঘ. যন্ত্রের নিজস্ব ভাষা
১২. কোন ভাষায় লিখিত প্রোগ্রাম কম্পিউটার সরাসরি বুঝতে পারে?  
ক. মেশিন ভাষা                      খ. উচ্চস্তরের ভাষা  
গ. অ্যাসেম্বলি ভাষা                      ঘ. চতুর্থ প্রজন্মের ভাষা
১৩. অ্যাসেম্বলি ভাষার নির্দেশাবলীকে কত ভাগে ভাগ করা যায়?  
ক. ৩                                      খ. ৪  
গ. ৫                                      ঘ. ৬
১৪. কোন ভাষা মেশিন নির্ভরশীল?  
ক. Cobol Language  
খ. Assembly Language  
গ. Basic Language  
ঘ. C++ Language
১৫. কোন ভাষা ০ ও ১ নির্ভর?  
ক. মেশিন ভাষা                      খ. অ্যাসেম্বলি ভাষা  
গ. কৃত্রিম ভাষা                      ঘ. কম্পাইলার ভাষা
১৬. অ্যাসেম্বলি ভাষার লেবেলে কয়টি বর্ণ ব্যবহৃত হতে পারে?  
ক. ১ থেকে ২টি                      খ. ২ থেকে ৩টি  
গ. ৩ থেকে ৪টি                      ঘ. ৪ থেকে ৫টি
১৭. কোন ভাষা দিয়ে কম্পিউটারের মেমোরি অ্যাড্রেসের সঙ্গে সরাসরি সংযোগ সাধন সম্ভব?  
ক. মেশিন ভাষা                      খ. হাইলেভেল ভাষা  
গ. অ্যাসেম্বলি ভাষা                      ঘ. চতুর্থ প্রজন্মের ভাষা

##### বহুপদী সমাপ্তিসূচক প্রশ্ন

১৮. মেশিন ভাষা হলো —  
i. অন্যান্য ভাষা হতে দ্রুত নির্বাহ হয়  
ii. যন্ত্রের ওপর নির্ভরশীল থাকে  
iii. তাড়াতাড়ি প্রোগ্রাম লেখা যায়

নিচের কোনটি সঠিক?

- ক. i ও ii                      খ. i ও iii  
গ. ii ও iii                      ঘ. i, ii ও iii

১৯. অ্যাসেম্বলি ভাষায় ব্যবহৃত বিভিন্ন নির্দেশ নেমোনিক হলো—

- i. DIV                      ii. ADD  
iii. INT

নিচের কোনটি সঠিক?

- ক. i ও ii                      খ. i ও iii  
গ. ii ও iii                      ঘ. i, ii ও iii

#### অভিন্ন তথ্যভিত্তিক প্রশ্ন

নিচের অনুচ্ছেদটি পড়ো এবং ২০ ও ২১ নং প্রশ্নের উত্তর দাও।  
সামিহা তার বাসায় পুরাতন কম্পিউটারে ০ ও ১ ব্যবহার করে প্রোগ্রাম রচনা করছিল। তার বড় ভাই সজল সামিহাকে আরো দ্রুতগতিতে কাজ করার জন্য দ্বিতীয় প্রজন্মের ভাষা শিখিয়ে দিল।

২০. সজলের শিখানো ভাষার পরবর্তী প্রজন্মের ভাষা কোনটি?

- ক. মেশিন ভাষা                      খ. অ্যাসেম্বলি ভাষা  
গ. উচ্চস্তরের ভাষা                      ঘ. ন্যাচারাল ভাষা

২১. সামিহার ব্যবহৃত ভাষায়—

- i. প্রোগ্রাম রচনায় অধিক সময় লাগে  
ii. প্রোগ্রাম রচনা করা সহজ  
iii. ডিবাগ করা কষ্টসাধ্য

নিচের কোনটি সঠিক?

- ক. i ও ii                      খ. i ও iii  
গ. ii ও iii                      ঘ. i, ii ও iii

#### ▶ উচ্চস্তরের ভাষাসমূহ

##### সাধারণ বহুনির্বাচনি প্রশ্ন

২২. C এর আবিষ্কারক কে?

- ক. মার্টিন রিচার্ডস                      খ. লেডি এডা  
গ. ডেনিস রিচি                      ঘ. বব মাইনার

২৩. C প্রোগ্রামের নতুন সুবিধা সংযোজনের পরবর্তী সংস্করণ কোনটি?

- ক. ++C                      খ. C++  
গ. C --                      ঘ. --C

২৪. C++ এর আবিষ্কারক কে?

- ক. বিচি ডেনিস                      খ. বায়ার্ন স্ট্রাউসট্রুপ  
গ. ল্যারি এরিকসন                      ঘ. বব মাইনার

২৫. কোন ভাষাকে কম্পিউটার প্রোগ্রামিং ভাষার জনক বলা হয়?

- ক. C                      খ. C++  
গ. C-                      ঘ. ORACLE

২৬. কোন ভাষায় প্রোগ্রাম কম্পিউটার সংগঠনের নিয়ন্ত্রণের উদ্দেশ্য থাকে?

- ক. উচ্চস্তরের ভাষা                      খ. নিম্নস্তরের ভাষা  
গ. মেশিন ভাষা                      ঘ. কৃত্রিম ভাষা

২৭. উচ্চস্তরের ভাষা কোনটি?

- ক. Word Star                      খ. Visicalc  
গ. C++                      ঘ. Lotus 1-2-3

২৮. প্রকৃতপক্ষে উচ্চস্তরের ভাষা কোনটি?

- ক. ইংরেজি ও গাণিতিক চিহ্নের সমন্বয়  
খ. ইংরেজি ও বুলিয়ান অ্যালজেব্রার সমন্বয়  
গ. বুলিয়ান অ্যালজেব্রা ও মরগ্যানের উপপাদ্যের সমন্বয়  
ঘ. শুধুমাত্র গাণিতিক চিহ্নের সমন্বয়

২৯. হাই লেভেল ভাষাকে কত ভাগে ভাগ করা যায়?

- ক. ২                      খ. ৩  
গ. ৪                      ঘ. ৫

৩০. মেশিন নিয়ন্ত্রণ, সিমুলেশন, বৈজ্ঞানিক পরীক্ষা ইত্যাদি কাজে ব্যবহৃত হয় কোনটি?

- ক. Algol                      খ. CSL  
গ. Coral-66                      ঘ. QBE

৩১. প্রয়োগ বৈশিষ্ট্যের ভিত্তিতে কম্পিউটার ভাষাকে কত ভাগে ভাগ করা যায়?

- ক. ২                      খ. ৩  
গ. ৪                      ঘ. ৫

৩২. পাইথন ভাষা তৈরি করেন কে?

- ক. গুইডো ভান রোসাস                      খ. ল্যারি এরিকসন  
গ. বব লোরোসি                      ঘ. অগাস্টা

#### বহুপদী সমাপ্তিসূচক প্রশ্ন

৩৩. চতুর্থ প্রজন্মের ভাষাকে বলা হয়—

- i. অতি উচ্চতর ভাষা                      ii. মধ্যম স্তরের ভাষা  
iii. নিম্ন স্তরের ভাষা  
নিচের কোনটি সঠিক?

- ক. i                      খ. ii  
গ. ii ও iii                      ঘ. i, ii ও iii

#### ▶ চতুর্থ প্রজন্মের ভাষা ও অনুবাদক প্রোগ্রাম

##### সাধারণ বহুনির্বাচনি প্রশ্ন

৩৪. যে প্রোগ্রাম উৎস প্রোগ্রামকে যান্ত্রিক ভাষায় অনুবাদ করে বস্তু প্রোগ্রামে রূপান্তর করে তাকে কী বলে?

- ক. যান্ত্রিক প্রোগ্রাম  
খ. অনুবাদক প্রোগ্রাম  
গ. অবজেক্ট প্রোগ্রাম  
ঘ. উৎস প্রোগ্রাম

৩৫. কৃত্রিম বুদ্ধিমত্তার সাথে সম্পর্কিত কোন প্রজন্মের কম্পিউটার ভাষা?

- ক. দ্বিতীয়                      খ. তৃতীয়  
গ. চতুর্থ                      ঘ. পঞ্চম

৩৬. প্রসেসিং ক্ষমতা বেশি দরকার কোন ভাষায়?

- ক. চতুর্থ প্রজন্মের ভাষায়  
খ. মেশিন ভাষায়  
গ. অ্যাসেম্বলি ভাষায়  
ঘ. উচ্চস্তরের ভাষায়

৩৭. কোন প্রজন্মের ভাষাকে ননপ্রসিডিউর ভাষা বলা হয়?  
ক. ১ম প্রজন্ম খ. ২য় প্রজন্ম  
গ. ৩য় প্রজন্ম ঘ. ৪র্থ প্রজন্ম
৩৮. অ্যাসেম্বলার কী?  
ক. একটি মেশিন খ. ব্রাউজার  
গ. প্রিন্টার ঘ. সফটওয়্যার
৩৯. কম্পাইলার কোন মেমোরিতে থাকে?  
ক. গৌণ মেমোরি খ. সহায়ক মেমোরি  
গ. ক্যাশ মেমোরি ঘ. হার্ডডিস্ক


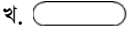


#### অভিন্ন তথ্যভিত্তিক প্রশ্ন

- নিচের উদ্দীপকটি পড়ো এবং ৪০ ও ৪১ নং প্রশ্নের উত্তর দাও।  
রাখি Computer fundamentals বই থেকে অনুবাদ প্রোগ্রাম সম্পর্কিত অধ্যায়টি পড়ছিল। সে এখান থেকে অনুবাদ প্রোগ্রামের প্রকারভেদ ও তাদের মধ্যে সাদৃশ্য ও বৈসাদৃশ্য সম্পর্কে জানতে পারে।
৪০. রাখি অনুবাদ প্রোগ্রাম সফটওয়্যার এর কতটি অনুবাদক সম্পর্কে জানতে পারে?  
ক. একটি খ. দুইটি  
গ. তিনটি ঘ. চারটি
৪১. রাখি কম্পাইলার ও ইন্টারপ্রেটার এর মধ্যে যে সাদৃশ্য দেখতে পায় তা হলো—  
i. হাই লেভেল ভাষায় লিখিত প্রোগ্রামকে মেশিন ভাষায় রূপান্তর করে  
ii. সোর্স প্রোগ্রামকে অবজেক্ট প্রোগ্রামে রূপান্তর করে  
iii. ইন্টারপ্রেটার ও কম্পাইলার মতো হাই লেভেল ভাষাকে মেশিন ভাষায় রূপান্তর করে  
নিচের কোনটি সঠিক?  
ক. i ও ii খ. i ও iii  
গ. ii ও iii ঘ. i, ii ও iii


#### ▶ প্রোগ্রামের সংগঠন, প্রোগ্রাম তৈরির ধাপসমূহ, অ্যালগরিদম ও ফ্লোচার্ট

##### সাধারণ বহুনির্বাচনি প্রশ্ন

৪২. যদি তুমি PRINT এর জায়গায় PRIMT টাইপ কর, তাহলে কোন ধরনের ভুল সংঘটিত হয়?  
ক. লজিক্যাল ভুল খ. সিনট্যাক্স ভুল  
গ. এক্সিকিউশন ভুল ঘ. রানটাইম ভুল
৪৩. ৫০ এর স্থানে ০৫ লেখা হলে এ ধরনের ভুলকে প্রোগ্রামের ক্ষেত্রে কী বলা হয়?  
ক. যুক্তিগত খ. সিনটেক্স  
গ. ডেটা ঘ. আউটপুট
৪৪. একটি প্রোগ্রামের কতটি অংশ থাকে?  
ক. ২টি খ. ৩টি  
গ. ৪টি ঘ. ৫টি

৪৫. প্রোগ্রাম তৈরির ধাপ কয়টি?  
ক. দুইটি খ. তিনটি  
গ. পাঁচটি ঘ. সাতটি
৪৬. প্রোগ্রাম টেস্টিং এর পূর্ববর্তী ধাপ কোনটি?  
ক. সমস্যা বিশ্লেষণ খ. কোডিং  
গ. প্রোগ্রাম উন্নয়ন ঘ. প্রোগ্রাম রক্ষণাবেক্ষণ
৪৭. প্রোগ্রাম কোন ধরনের ভুলের জন্য কম্পিউটারের বার্তা দেয়?  
ক. সিনট্যাক্স ভুল খ. লজিক্যাল ভুল  
গ. ডেটা ভুল ঘ. যেকোনো ভুল
৪৮. প্রোগ্রামের ভুলকে কয় ভাগে ভাগ করা হয়?  
ক. দুই ভাগে খ. তিন ভাগে  
গ. চার ভাগে ঘ. পাঁচ ভাগে
৪৯. অ্যালগরিদম কী?  
ক. পর্যায়ক্রম খ. সিদ্ধান্তক্রম  
গ. অনুক্রমিক ঘ. ফ্লোচার্ট
৫০. অ্যালগরিদমের চিত্ররূপকে কী বলে?  
ক. সিনট্যাক্স ভুল খ. ফ্লোচার্ট  
গ. অ্যালগরিদম ঘ. ডিবাগিং
৫১. ফ্লোচার্ট শুরু ও শেষ করতে কোন চিহ্নটি ব্যবহৃত হয়?  
ক.  খ.   
গ.  ঘ. 

#### বহুপদী সমাপ্তিসূচক প্রশ্ন

৫২. সি ভাষায় #include<stdio.h> হেডার ফাইলের অন্তর্গত বিভিন্ন ফাংশন হল—  
i. printf() ii. scanf()  
iii. gets ()  
নিচের কোনটি সঠিক?  
ক. i ও ii খ. i ও iii  
গ. ii ও iii ঘ. i, ii ও iii
৫৩. সিনট্যাক্স ভুল হলো—  
i. ব্রাকেট ঠিকমতো না দেওয়া  
ii. কমা না দেওয়া  
iii. বানান ভুল করা  
নিচের কোনটি সঠিক?  
ক. i ও ii খ. i ও iii  
গ. ii ও iii ঘ. i, ii ও iii
৫৪. ফ্লোচার্টে  চিহ্নটি ব্যবহার করা হয় —  
i. START  
ii. PROCESS  
iii. END  
নিচের কোনটি সঠিক?  
ক. i ও ii খ. i ও iii  
গ. ii ও iii ঘ. i, ii ও iii



## অভিন্ন তথ্যভিত্তিক প্রশ্ন

উদ্দীপকটি পড়ে ৫৫ ও ৫৬ নং প্রশ্নের উত্তর দাও:  
হাসান সাহেব একাউন্টিং সফটওয়্যার তৈরির সময় প্রতিটি মডিউল ছোট ছোট করে বিভক্ত করে সমাধানের ধাপ নির্ধারণ করেন। কিন্তু সফটওয়্যারটি তৈরির পর সেটি পরীক্ষা করে দেখা যায় প্রদত্ত ডেটার জন্য ফলাফল ভুল প্রদর্শিত হচ্ছে।

৫৫. উদ্দীপকে হাসান সাহেব কোন টুলগুলোকে নির্দেশ

করেছেন—

i. প্রোগ্রাম কোডিং ii. অ্যালগরিদম

iii. ফ্লোচার্ট

নিচের কোনটি সঠিক?

ক. i ও ii খ. i ও iii

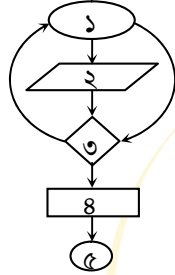
গ. ii ও iii ঘ. i, ii ও iii

৫৬. উদ্দীপকে প্রদর্শিত ভুলের কারণ কোনটি?

ক. কোডিং খ. ডিবাগিং

গ. অ্যালগরিদম ঘ. ফ্লোচার্ট

নিচের ফ্লোচার্টটি লক্ষ্য করো এবং ৫৭ ও ৫৮ নং প্রশ্নের উত্তর দাও:



৫৭. ১ নং ধাপ দ্বারা কী বোঝায়?

ক. প্রসেসিং খ. সিদ্ধান্ত

গ. শুরু ঘ. সংযুক্ত

৫৮. যে ধাপগুলোতে ভুল চিহ্ন প্রয়োগ করা হয়েছে তা হলো—

i. ধাপ-১ ii. ধাপ-২

iii. ধাপ-৩

নিচের কোনটি সঠিক?

ক. i ও ii খ. i ও iii

গ. ii ও iii ঘ. i, ii ও iii

## ▶ প্রোগ্রাম ডিজাইন মডেল

## সাধারণ বহুনির্বাচনি প্রশ্ন

৫৯. প্রোগ্রামের স্টেটমেন্টগুলো ভিন্ন ভিন্ন শ্রেণিতে বিভক্ত করে লেখা হয় কোন মডেলের সাহায্যে?

ক. ভিজুয়াল প্রোগ্রামিং

খ. স্ট্রাকচার প্রোগ্রামিং

গ. অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং

ঘ. ইভেন্ট ড্রাইভেন প্রোগ্রামিং

৬০. স্ট্রাকচার প্রোগ্রামের ধারণা দেন কে?

ক. বব রাইমান খ. এডগার ডি কসট্রো

গ. ল্যারি এরিকসন ঘ. মাইক্রোসফট কর্পোরেশন

৬১. কোন মডেলে পুরো সমস্যাকে বিভিন্ন অংশে ভাগ করা হয়?

ক. ভিজুয়াল প্রোগ্রামিং

খ. স্ট্রাকচার প্রোগ্রামিং

গ. অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং

ঘ. ইভেন্ট ড্রাইভেন প্রোগ্রামিং

## বহুপদী সমাপ্তিসূচক প্রশ্ন

৬২. অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং-এর বৈশিষ্ট্য হলো—

i. ইনহেরিটেন্স ii. পলিমরফিজম

iii. এনক্যাপসুলেশন

নিচের কোনটি সঠিক?

ক. i ও ii খ. i ও iii

গ. ii ও iii ঘ. i, ii ও iii

## ▶ 'সি' প্রোগ্রামের প্রাথমিক ধারণা, বৈশিষ্ট্য, কম্পাইলিং ও ভাষার গঠন

## সাধারণ বহুনির্বাচনি প্রশ্ন

৬৩. সি ভাষায় কোনো প্রোগ্রাম run করতে কী-বোর্ড কমান্ড কোনটি?

ক. shift + f9

খ. ctrl + f7

গ. ctrl + f9

ঘ. shift + f8

৬৪. সি শব্দটি এসেছে কোন ভাষা থেকে?

ক. CCPL

খ. BCPL

গ. LCPL

ঘ. ABCL

৬৫. IDE কী?

ক. Integrated Development Environment

খ. Internal Development Environment

গ. Internal Developing Environment

ঘ. Inturction Develop Environment

৬৬. সি ভাষা কোন ধরনের প্রোগ্রামিং মডেল অনুসরণ করে?

ক. স্ট্রাকচার

খ. অবজেক্ট ওরিয়েন্টেড

গ. ভিজুয়াল

ঘ. ইভেন্ট ড্রাইভেন

৬৭. C ভাষার প্রোগ্রাম-এ main function কয়টি থাকতে পারে?

ক. ১টি

খ. ২টি

গ. ৩টি

ঘ. ৪টি

৬৮. Header File সংযুক্ত করতে হয় কোন ভাষাতে?

ক. Basic

খ. Ada

গ. pascal

ঘ. C

৬৯. সি ভাষাতে প্রতিটি প্রোগ্রামের কাজ শুরু হয় কোন ফাংশন থেকে?

ক. Printf

খ. Scanf

গ. main

ঘ. Main

৭০. সি ভাষাতে একের অধিক লাইনের কমেন্টস লিখতে হয় কীভাবে?  
ক. দুটি স্লাস (//) দিয়ে খ. /\* \*/ দিয়ে  
গ. // // দিয়ে ঘ. /\* \*/ দিয়ে
৭১. সি প্রোগ্রামে প্রোগ্রাম রচনা করা হয় কোন বন্ধনীর মধ্যে?  
ক. () খ. {}  
গ. [] ঘ. <>
৭২. printf() কোন ধরনের ফাংশন?  
ক. main ফাংশন খ. header ফাংশন  
গ. Library ফাংশন ঘ. footer ফাংশন
৭৩. পলিমরফিজম নিচের কোন ভাষার বৈশিষ্ট্য?  
ক. সি খ. ভিজ্যুয়েল বেসিক  
গ. জাভা ঘ. ওরাকল
৭৪. Compiler derivative কোনটি?  
ক. ; খ. #  
গ. : ঘ. ()
৭৫. সি ভাষার সম্পূর্ণ অংশ কী দ্বারা আবদ্ধ থাকে?  
ক. () দ্বারা খ. {} দ্বারা  
গ. [] দ্বারা ঘ. “ ” দ্বারা
৭৬. main() ফাংশনের ঘোষণা অংশে কী ঘোষণা করতে হয়?  
ক. যেকোনো ভেরিয়েবল খ. গ্লোবাল ভেরিয়েবল  
গ. স্পেশাল ভেরিয়েবল ঘ. main() ফাংশন
৭৭. main() ফাংশনের নির্বাহ অংশে অন্তত কয়টি স্টেটমেন্ট থাকতে হয়?  
ক. ১টি খ. ২টি  
গ. ৩টি ঘ. ৪টি
৭৮. C প্রোগ্রাম রান করার কমান্ড কোনটি?  
ক. Ctrl + F6 খ. Ctrl + F9  
গ. Ctrl + F10 ঘ. Alt + Ctrl + F9

#### বহুপদী সমাপ্তিসূচক প্রশ্ন

৭৯. C ভাষার স্টেটমেন্ট কোনটি—  
i. getch ii. printf  
iii. input  
নিচের কোনটি সঠিক?  
ক. i ও ii খ. ii ও iii  
গ. i ও iii ঘ. সবগুলো
৮০. প্রোগ্রাম তৈরির ধাপে কোডিং—  
i. সমস্যার বিশ্লেষণের সাথে সম্পর্কিত  
ii. প্রোগ্রামিং ভাষার সাহায্যে করা  
iii. প্রোগ্রাম তৈরির পর ভুল খোঁজা  
নিচের কোনটি সঠিক?  
ক. i ও ii খ. i ও iii  
গ. ii ও iii ঘ. i, ii ও iii

৮১. main() ফাংশনের অংশসমূহ—  
i. ঘোষণা অংশ  
ii. নির্বাহ অংশ  
iii. লিংক অংশ  
নিচের কোনটি সঠিক?  
ক. i ও ii খ. i ও iii  
গ. ii ও iii ঘ. i, ii ও iii

#### অভিন্ন তথ্যভিত্তিক প্রশ্ন

উদ্দীপকটি পড়ে ৮২ ও ৮৩ নং প্রশ্নের উত্তর দাও:  
main ()

```
{
int n;
scanf("%d", &n);
printf("%d", sqrt (n));
}
```

৮২. উদ্দীপকে ব্যবহৃত ডেটা টাইপ কোনটি?  
ক. Primary খ. User defined  
গ. Derived ঘ. Empty
৮৩. উদ্দীপকে আবশ্যিক হেডার ফাইল কোনটি—  
i. stdio.h  
ii. conio.h  
iii. math.h  
নিচের কোনটি সঠিক?  
ক. i ও ii খ. i ও iii  
গ. ii ও iii ঘ. i, ii ও iii

#### ডেটা টাইপ, চলক, ধ্রুবক

#### সাধারণ বহুনির্বাচনি প্রশ্ন

৮৪. ইন্টিজার টাইপের ডেটার ফরম্যাট স্পেসিফাইয়ার কোনটি?  
ক. %c খ. %d  
গ. %F ঘ. %i
৮৫. প্রোগ্রামের যে অংশের জন্য কোনো ভেরিয়েবলের কার্যক্রম বিস্তৃত সেই অংশকে কী বলা হয়?  
ক. স্কোপ বা ক্ষেত্র খ. মডিফায়ার  
গ. কোয়ালিফায়ার ঘ. রাশিমালা
৮৬. সি-প্রোগ্রামে কতভাবে ধ্রুবক ব্যবহার করা যায়?  
ক. ২ খ. ৩  
গ. ৪ ঘ. ৫
৮৭. ডেটার ধরনকে কী বলে?  
ক. প্রোটোটাইপ খ. ডেটা টাইপ  
গ. ডেটা ঘ. তথ্য
৮৮. প্রাথমিক ডেটা টাইপ কত ধরনের?  
ক. ২ খ. ৩  
গ. ৪ ঘ. ৫

৮৯. উদ্দীপকটি লক্ষ্য কর:
- ```
main()
{
  const int k = 5;
  k++;
  printf("k is %d", k);
}
```
- নিচের কোনটি সঠিক?
- ক. k is 5  
খ. k is 6  
গ. Error, ধ্রুবক চলকের মান কেবল দুইবার পরিবর্তন হয়  
ঘ. Error, ধ্রুবক চলকের মান পরিবর্তন হয় না
৯০. নিচের কোনটি সঠিক?
- ক. #define pi 31.1416  
খ. #define pi=31.1416;  
গ. const int pi=3.1416  
ঘ. const pi=3.1416;
৯১. char টাইপ এর মডিফায়ার কোনটি?
- ক. short                      খ. sqrt  
গ. long                      ঘ. signed
৯২. Integer ডেটা টাইপের মেমোরি স্পেস কত?
- ক. 1 byte                      খ. 2 byte  
গ. 3 byte                      ঘ. 4 byte
৯৩. C প্রোগ্রামে ক্যারেক্টার ডেটা টাইপকে কী দ্বারা প্রকাশ করা হয়?
- ক. int                      খ. char  
গ. float                      ঘ. double
৯৪. C ভাষায় প্রোগ্রাম কম্পাইল করলে কোন ফাইল তৈরি হয়?
- ক. .CPP                      খ. .obj  
গ. .exe                      ঘ. .dba
৯৫. অবস্থানের ওপর ভিত্তি করে চলককে কত ভাগে ভাগ করা যায়?
- ক. ২ ভাগে                      খ. ৩ ভাগে  
গ. ৪ ভাগে                      ঘ. ৫ ভাগে
৯৬. float টাইপ ভেরিয়েবলের মান ইনপুট/আউটপুট করার জন্য কোনটি ব্যবহৃত হয়?
- ক. %d                      খ. %c  
গ. %f                      ঘ. %g
৯৭. long int টাইপের ভেরিয়েবলের মান ইনপুট/আউটপুট দেওয়ার জন্য কোনটি ব্যবহৃত হয়?
- ক. %d                      খ. %ld  
গ. %f                      ঘ. %lf
৯৮.  $a \times b \times c$  এর সি ভাষায় এক্সপ্রেশন কী হবে?
- ক.  $a \times b \times c$                       খ.  $a * b * c$   
গ.  $a \times b \times c$                       ঘ.  $ab \times c$

## বহুপদী সমাপ্তিসূচক প্রশ্ন

৯৯. সি প্রোগ্রামে কম্পাউন্ট ঘোষণা করা যায়—
- i. Const কিওয়ার্ড ব্যবহার করে  
ii. শুধু নাম ব্যবহার করে  
iii. #define প্রিপ্রসেসর ব্যবহার করে
- নিচের কোনটি সঠিক?
- ক. i ও ii                      খ. i ও iii  
গ. ii ও iii                      ঘ. i, ii ও iii
১০০. Empty ডেটা হিসেবে ব্যবহার করা হয়—
- i. integer                      ii. void  
iii. null
- নিচের কোনটি সঠিক?
- ক. i ও ii                      খ. i ও iii  
গ. ii ও iii                      ঘ. i, ii ও iii
১০১. সি ভাষার হেডার ফাইল হচ্ছে—
- i. প্রোগ্রামের আবশ্যিক অংশ  
ii. ডেটাটাইপ ধারণকারী ফাইল  
iii. ফাংশনের বর্ণনা ধারণকারী ফাইল
- নিচের কোনটি সঠিক?
- ক. i ও ii                      খ. i ও iii  
গ. ii ও iii                      ঘ. i, ii ও iii
১০২. ডেটা টাইপ মডিফায়ার—
- i. Short                      ii. Long  
iii. Signed
- নিচের কোনটি সঠিক?
- ক. i ও ii                      খ. i ও iii  
গ. ii ও iii                      ঘ. i, ii ও iii

▶ রাশিমালা, কি-ওয়ার্ড ও ইনপুট আউটপুট স্টেটমেন্ট

## সাধারণ বহুনির্বাচনি প্রশ্ন

১০৩.  $\text{pow}((15/3-9/3), (17\%5)) - 4 + 3 * 2 + 7$  হিসাবটির সি ভাষায় ফলাফল কত?
- ক. 1                      খ. -4  
গ. 13                      ঘ. 10
১০৪. সি ভাষায় গাণিতিক এবং যৌক্তিক কাজ নিয়ন্ত্রণ করার জন্য বিশেষ সিম্বলগুলোকে কী বলে?
- ক. রাশিমালা                      খ. অপারেন্ড  
গ. চলক                      ঘ. অপারেটর
১০৫. তিনটি পূর্ণ সংখ্যা ইনপুটের জন্য নিচের কোনটি সঠিক?
- ক. `scanf("%d%d%d",&a,&b,&c);`  
খ. `scanf("%d,%d,%d",&a,&b,&c);`  
গ. `scanf("%d %d %d",&a,&b,&c);`  
ঘ. `scanf("%d %d %d",&a,&b,&c);`



১২৬. প্রোগ্রামটির আউটপুট কত হবে?

- ক. ০ খ. ১  
গ. ২ ঘ. ৫

১২৭. A=3, B=4 হলে printf("%d", A>B); এর আউটপুট কত?

- ক. 0 খ. 1  
গ. 3 ঘ. 8

নিচের উদ্দীপকটি লক্ষ্য করো এবং প্রশ্ন দুটির উত্তর দাও:

```
main ()
{
  int a=5,b=4;
  a+=b;
  printf ("%d", sqrt(a));
}
```

১২৮. উদ্দীপকে কী ধরনের অপারেটর ব্যবহৃত হয়েছে?

- ক. অ্যাসাইনমেন্ট অপারেটর  
খ. টারনারি অপারেটর  
গ. বিটওয়াইজ অপারেটর  
ঘ. রিলেশনাল অপারেটর

১২৯. উদ্দীপকে আবশ্যিক হেডার ফাইল হলো—

- i. stdio.h ii. conio.h  
iii. math.h

নিচের কোনটি সঠিক?

- ক. i ও ii খ. i ও iii  
গ. ii ও iii ঘ. i, ii ও iii

উদ্দীপকটি পড়ে ১৩০ ও ১৩১ নং প্রশ্নের উত্তর দাও:

```
X = 100;
X = 5;
X = X% 10;
```

১৩০. X-এর মান কত?

- ক. 0 খ. 2  
গ. 10 ঘ. 20

১৩১. উদ্দীপকে ব্যবহৃত অপারেটর হচ্ছে —

- i. Arithmetic ii. Assignment iii. Logical

নিচের কোনটি সঠিক?

- ক. i ও ii খ. i ও iii  
গ. ii ও iii ঘ. i, ii ও iii

### ▶ স্টেটমেন্ট

#### সাধারণ বহুনির্বাচনি প্রশ্ন

১৩২. কোনো স্টেটমেন্ট লুপের সাধারণ ফ্লো পরিবর্তন করার জন্য ব্যবহৃত হয়?

- ক. if-else স্টেটমেন্ট খ. for স্টেটমেন্ট  
গ. continue স্টেটমেন্ট ঘ. while স্টেটমেন্ট

১৩৩. for লুপ স্টেটমেন্ট কয়টি অংশ নিয়ে কাজ করে?

- ক. ১ খ. ২  
গ. ৩ ঘ. ৪

১৩৪. প্রোগ্রামের এক লাইন থেকে পরবর্তী লাইনে না গিয়ে

উপরে বা নিচে অন্য কোনো লাইন থেকে কাজ শুরু করার জন্য যে স্টেটমেন্ট ব্যবহৃত হয় তাকে কী বলে?

- ক. কন্ডিশনাল স্টেটমেন্ট খ. লুপ স্টেটমেন্ট  
গ. জাম্পিং স্টেটমেন্ট ঘ. ইনপুট স্টেটমেন্ট

১৩৫. প্রোগ্রামের ১টি শর্তসাপেক্ষে কোনো স্টেটমেন্ট

সম্পাদনের জন্য কী ব্যবহৃত হয়?

- ক. if খ. if — else  
গ. if — else — else if ঘ. do .... while

১৩৬. প্রোগ্রামে অনেক শর্তের জন্য কোনটি ব্যবহার করা যুক্তিযুক্ত?

- ক. if — else — else if খ. switch  
গ. for ঘ. do — while

১৩৭. লুপ স্টেটমেন্টে লুপ বডি ও টেস্ট কন্ডিশনের অবস্থানের ভিত্তিতে লুপ স্টেটমেন্টকে কতভাগে ভাগ করা যায়?

- ক. ২ খ. ৩ গ. ৪ ঘ. ৫

১৩৮. লুপ কতবার হবে তা জানা থাকলেই কোন স্টেটমেন্ট ব্যবহার করা যাবে?

- ক. if খ. switch  
গ. for ঘ. if—else

১৩৯. কখন do while লুপ এর Condition পরীক্ষা করা হয়?

- ক. লুপের শেষে খ. লুপের শুরুতে  
গ. লুপের মাঝখানে ঘ. যেকোনো সময়

১৪০. for(i=1;i<=10;i=i+2)

```
{
  printf("%d",i);
}
```

স্টেটমেন্টটির ফলাফল কোনটি?

- ক. 13579 খ. 2 4 6 8 10  
গ. 1 2 3 4 5 6 7 8 9 ঘ. 1 2 3 4 5 6 7 8 9 10

#### বহুপদী সমাপ্তিসূচক প্রশ্ন

১৪১. জাম্পিং স্টেটমেন্ট হলো—

- i. continue ii. break  
iii. goto

নিচের কোনটি সঠিক?

- ক. i ও ii খ. i ও iii  
গ. ii ও iii ঘ. i, ii ও iii

১৪২. লুপিং স্টেটমেন্ট হলো—

- i. do ii. for  
iii. while

নিচের কোনটি সঠিক?

- ক. i ও ii খ. i ও iii  
গ. ii ও iii ঘ. i, ii ও iii

১৪৩. সি প্রোগ্রামে লুপিং স্টেটমেন্টগুলো হলো—

- i. for ii. do-while  
iii. while

নিচের কোনটি সঠিক?

- ক. i ও ii খ. i ও iii  
গ. ii ও iii ঘ. i, ii ও iii

১৪৪. লুপ স্টেটমেন্টসমূহের অংশগুলো হলো—  
 i. লুপ বডি ii. টেস্ট কন্ডিশন  
 iii. লুপ এক্সিকিউশন  
 নিচের কোনটি সঠিক?  
 ক. i ও ii খ. i ও iii  
 গ. ii ও iii ঘ. i, ii ও iii
১৪৫. সি প্রোগ্রামে ব্যবহৃত লাইব্রেরি ফাংশন হলো—  
 i. printf() ii. const()  
 iii. scanf()  
 নিচের কোনটি সঠিক?  
 ক. i ও ii খ. i ও iii  
 গ. ii ও iii ঘ. i, ii ও iii

#### অভিন্ন তথ্যভিত্তিক প্রশ্ন

নিচের উদ্দীপকটি পড় এবং ১৪৬ ও ১৪৭ নং প্রশ্নের উত্তর দাও:

```
#include <stdio.h>
main ()
{
    int a = 3, b;
    b = 2 * a;
    printf ("%d", b);
}
```

১৪৬. প্রোগ্রাম রান করলে b এর মান কত হবে?  
 ক. ৩ খ. ৪  
 গ. ৫ ঘ. ৬
১৪৭. প্রোগ্রাম রান করলে আউটপুট মান ৩ হবে, যখন—  
 i. b = a ++; ii. b = a --;  
 iii. b += a;  
 নিচের কোনটি সঠিক?  
 ক. i ও ii খ. i ও iii  
 গ. ii ও iii ঘ. i, ii ও iii

#### ▶ অ্যারে ও ফাংশন

##### সাধারণ বহুনির্বাচনি প্রশ্ন

১৪৮. অ্যারে কত প্রকার?  
 ক. ২ খ. ৩  
 গ. ৪ ঘ. ৫
১৪৯. a[0], a[1], a[2], a[3] একই অ্যারের অন্তর্গত হলে  
 সঠিক অ্যারে চলক ঘোষণা কোনটি?  
 ক. int a[0] খ. int a[1]  
 গ. int a[3] ঘ. int a[4]
১৫০. যে অ্যারের রো ও সারি উভয়ই থাকে তাকে কী বলে?  
 ক. একমাত্রিক অ্যারে খ. দ্বিমাত্রিক অ্যারে  
 গ. ত্রিমাত্রিক অ্যারে ঘ. বহুমাত্রিক অ্যারে
১৫১. দৈর্ঘ্য, প্রস্থ ও উচ্চতা বিশিষ্ট কোনো বস্তুর আয়তন  
 নির্ণয়ের জন্য কোন অ্যারে ব্যবহার করা হয়?  
 ক. একমাত্রিক খ. দ্বিমাত্রিক  
 গ. ত্রিমাত্রিক ঘ. চতুর্থ মাত্রিক









১৫২. কোনটি সি ভাষার ফাংশন?  
 ক. int() খ. stdio.h  
 গ. printf() ঘ. for()
১৫৩. ইউজার ডিফাইন্ড ফাংশনের নাম কী?  
 ক. ভেরিয়েবল খ. ধ্রুবক  
 গ. আইডেন্টিফায়ার ঘ. স্ট্রাকচার
১৫৪. ইউজার ডিফাইন্ড ফাংশনের বডি কোনটি?  
 ক. ফাংশনের বর্ণনা খ. ফাংশন কল  
 গ. ফাংশন প্রোটোটাইপ ঘ. রিটার্ন টাইপ
১৫৫. ইউজার ডিফাইন্ড ফাংশনে আর্গুমেন্ট ভেরিয়েবল না  
 থাকলে কী ব্যবহার করা হয়?  
 ক. return খ. void  
 গ. অন্য ভেরিয়েবল ঘ. অন্য ফাংশন
১৫৬. ফাংশন কল-এর শেষে থাকে কোনটি?  
 ক. (:) খ. (;)  
 গ. (,) ঘ. (())
১৫৭. সি প্রোগ্রামে ড্রাইভার ফাংশন কোনটি?  
 ক. লাইব্রেরি ফাংশন খ. ইউজার ডিফাইন্ড ফাংশন  
 গ. main() ফাংশন ঘ. বিল্ট-ইন ফাংশন
১৫৮. নিচের কোনটি গাণিতিক ফাংশন?  
 ক. stremp খ. sum  
 গ. printf ঘ. pow

##### বহুপদী সমাপ্তিসূচক প্রশ্ন

১৫৯. যে ফাংশন নিজেই নিজেকে কল করতে পারে তাকে বলে—  
 i. রিকার্সিভ  
 ii. লাইব্রেরি  
 iii. ইউজার ডিফাইন্ড  
 নিচের কোনটি সঠিক?  
 ক. i খ. ii  
 গ. i ও ii ঘ. i, ii ও iii
১৬০. অ্যারের প্রকারভেদ হলো—  
 i. একমাত্রিক  
 ii. বহুমাত্রিক  
 iii. শূন্যমাত্রিক  
 নিচের কোনটি সঠিক?  
 ক. i ও ii খ. i ও iii  
 গ. ii ও iii ঘ. i, ii ও iii
১৬১. বহুমাত্রিক অ্যারেগুলো হলো—  
 i. শূন্যমাত্রিক  
 ii. দ্বিমাত্রিক  
 iii. ত্রিমাত্রিক  
 নিচের কোনটি সঠিক?  
 ক. i ও ii খ. i ও iii  
 গ. ii ও iii ঘ. i, ii ও iii



## খ. বোর্ড পরীক্ষার বহুনির্বাচনি প্রশ্ন

১৬২. কোন ভাষায় লিখিত প্রোগ্রামের জন্য অনুবাদকের প্রয়োজন হয় না? /রা. বো. ১৯/  
ক. Natural খ. Machine  
গ. High Level ঘ. Assembly
১৬৩. মেশিন ভাষার সুবিধা কোনটি? /সি. বো. ২০১৭/  
ক. প্রোগ্রাম সহজে লেখা যায়  
খ. সবধরনের মেশিনে ব্যবহার উপযোগী  
গ. প্রোগ্রাম সরাসরি ও দ্রুত কার্যকর হয়  
ঘ. প্রোগ্রামের ভুল সহজে শনাক্ত করা যায়
১৬৪. কোন ভাষায় লিখিত প্রোগ্রাম কম্পিউটার সরাসরি বুঝতে পারে? /সি. বো. ১৬/  
ক. মেশিন ভাষা খ. উচ্চস্তরের ভাষা  
গ. অ্যাসেম্বলি ভাষা ঘ. চতুর্থ প্রজন্মের ভাষা
১৬৫. অ্যাসেম্বলি ভাষা কোন প্রজন্মের ভাষা? /দি. বো. ১৬/  
ক. ১ম খ. ২য়  
গ. ৩য় ঘ. ৪র্থ
১৬৬. সাংকেতিক ভাষা কোনটি? /সি. বো. ১৬/  
ক. মেশিন ভাষা খ. অ্যাসেম্বলি ভাষা  
গ. উচ্চস্তরের ভাষা ঘ. অতি উচ্চস্তরের ভাষা
১৬৭. কোন ভাষায় হার্ডওয়্যার নিয়ন্ত্রণের পাশাপাশি উচ্চস্তরের ভাষার সুবিধা পাওয়া যায়? /রা. বো. ১৯/  
ক. PASCAL খ. COBOL  
গ. C ঘ. FORTRAN
১৬৮. কোনটিতে কম মেমোরি ও রিসোর্স নিয়ে সহজে প্রোগ্রাম লেখা যায়? /ঘ. বো. ১৬/  
ক. এক্সেস খ. ওরাকল  
গ. সি ঘ. পাইথন
১৬৯. উৎস প্রোগ্রামকে একত্রে বস্তু প্রোগ্রামে রূপান্তর করে কোনটি? /দি. বো. ২০১৭/  
ক. কম্পাইলার খ. ইন্টারপ্রেটার  
গ. লিংকার ঘ. অ্যাসেম্বলার
১৭০. প্রোগ্রামিং ভাষায় লেখা প্রোগ্রামকে কী বলা হয়? /সি. বো. ১৬/  
ক. গন্তব্য প্রোগ্রাম খ. উৎস প্রোগ্রাম  
গ. ভিজুয়াল প্রোগ্রাম ঘ. অনুবাদক প্রোগ্রাম
১৭১. 4GL বলতে বুঝায় — /কু. বো. ১৬/  
ক. অতি উচ্চস্তরের ভাষা খ. উচ্চস্তরের ভাষা  
গ. মধ্যম স্তরের ভাষা ঘ. নিম্নস্তরের ভাষা
১৭২. কোনটি স্বাভাবিক ভাষা? /ঘ. বো. ২০১৭/  
ক. 4GL খ. 5GL  
গ. মেশিন ভাষা ঘ. অ্যাসেম্বলি ভাষা
১৭৩. কোনটি চতুর্থ প্রজন্মের ভাষা? /দি. বো. ২০১৭/  
ক. BASIC খ. PASCAL  
গ. INTELLECT ঘ. CSL
১৭৪. OPS5 কোন প্রজন্মের ভাষা? /কু. বো. ১৯/  
ক. পঞ্চম খ. চতুর্থ  
গ. তৃতীয় ঘ. দ্বিতীয়
১৭৫. অনুবাদক সফটওয়্যার কয় ধরনের? /চ. ঘ. বো. ১৬/  
ক. ২ খ. ৩  
গ. ৪ ঘ. ৫
১৭৬. প্রোগ্রাম তৈরিতে প্রোগ্রাম ডিজাইনের পরবর্তী ধাপ কোনটি? /সকল বোর্ড ২০১৮/  
ক. সমস্যা বিশ্লেষণ খ. প্রোগ্রাম কোডিং  
গ. প্রোগ্রাম বাস্তবায়ন ঘ. প্রোগ্রাম রক্ষণাবেক্ষণ
১৭৭.  $\boxed{\text{উৎস প্রোগ্রাম}} \rightarrow \boxed{?} \rightarrow \boxed{\text{বস্তু প্রোগ্রাম}}$  (?)  
চিহ্নিত স্থানে কী হবে? /কু. বো. ১৯/  
ক. কম্পাইলার খ. ইন্টারপ্রেটার  
গ. অ্যাসেম্বলার ঘ. লিংকার
১৭৮. C ভাষায় লেখা প্রোগ্রামকে কী কোড বলা হয়? /মাদ্রাসা- ১৬/  
ক. আসকি খ. সোর্স  
গ. অবজেক্ট ঘ. ইউনি
১৭৯. প্রোগ্রাম ফ্লোচার্টে প্রক্রিয়াকরণে কোন প্রতীকটি ব্যবহৃত হয়? /রা., কু., ব. বো. ১৬; দি., মাদ্রাসা বোর্ড ১৯/  
ক.  খ.   
গ.  ঘ. 
১৮০. ফ্লোচার্ট কত প্রকার? /সি. বো. ১৬/  
ক. ২ খ. ৪  
গ. ৬ ঘ. ৮
১৮১.  $\diamond$  প্রতীকটি কোন কাজে ব্যবহার হয়? /ব. বো. ১৯/  
ক. সিদ্ধান্ত গ্রহণ খ. প্রক্রিয়াকরণ  
গ. ডেটা ইনপুট ঘ. ডেটা আউটপুট
১৮২.  $\circ$  চিহ্ন দ্বারা C ভাষায় প্রোগ্রামিং-এ কী বোঝানো হয়? /দি. বো. ১৬; মাদ্রাসা বোর্ড ১৯/  
ক. সিদ্ধান্ত খ. ইনপুট  
গ. টীকা ঘ. সংযোগ
১৮৩. প্রোগ্রামিং এর ক্ষেত্রে ইনপুট বা আউটপুট চিহ্ন হিসেবে ব্যবহৃত হয় কোনটি? /মাদ্রাসা- ১৬/  
ক.  খ.   
গ.  ঘ. 
১৮৪. প্রোগ্রামে  $\square$  এই চিহ্নটি দ্বারা কী বুঝানো হয়? /মাদ্রাসা বোর্ড ২০১৮/  
ক. প্রবাহের দিক খ. সংযোগ  
গ. টিকা ঘ. শুরু



১৮৫. প্রোগ্রাম কোডিং এর পূর্ববর্তী ধাপ কোনটি? /দি. বো. ১৯/  
ক. সমস্যা বিশ্লেষণ খ. প্রোগ্রাম ডিজাইন  
গ. প্রোগ্রাম বাস্তবায়ন ঘ. প্রোগ্রাম রক্ষণাবেক্ষণ
১৮৬. প্রোগ্রামের ভুলত্রুটি খুঁজে বের করে তা সংশোধনের প্রক্রিয়াকে কি বলা হয়? /সি. বো. ১৭/  
ক. Encoding খ. Debugging  
গ. Coding ঘ. Decoding
১৮৭. প্রোগ্রাম তৈরিতে প্রোগ্রাম ডিজাইনের পরবর্তী ধাপ কোনটি? /সকল বোর্ড ১৮/  
ক. সমস্যা বিশ্লেষণ খ. প্রোগ্রাম কোডিং  
গ. প্রোগ্রাম বাস্তবায়ন ঘ. প্রোগ্রাম রক্ষণাবেক্ষণ
১৮৮. কোনটি অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ভাষা? /সি. বো. ১৯/  
ক. BASIC খ. HTML  
গ. C ঘ. Java
১৮৯. ইনহেরিটেন্স কোন প্রোগ্রামিং মডেল এর বৈশিষ্ট্য? /ঘ. বো. ১৯/  
ক. স্ট্রাকচার্ড প্রোগ্রামিং  
খ. অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং  
গ. ভিজুয়াল প্রোগ্রামিং  
ঘ. ইভেন্ট ড্রাইভেন প্রোগ্রামিং
১৯০. “সি” ভাষায় ইন্টিজার ডেটা টাইপ কত বিটের? /চ. বো. ১৯/  
ক. ৮ খ. ১৬  
গ. ৩২ ঘ. ৬৪
১৯১. সি ভাষায় float ডেটা টাইপ কত বিটের? /রা. বো. ১৯/  
ক. ১৬ খ. ৩২  
গ. ৪৮ ঘ. ৬৪
১৯২. float type চলকের জন্য মেমোরিতে কত বাইট জায়গার প্রয়োজন হয়? /রা. বো. -২০১৭/  
ক. ১ খ. ২  
গ. ৪ ঘ. ৮
১৯৩. C প্রোগ্রামিং ভাষায় long integer চলক মেমোরিতে কত বাইট জায়গা নেয়? /কু. বো. - ১৬/  
ক. ২ বাইট খ. ৪ বাইট  
গ. ৮ বাইট ঘ. ১৬ বাইট
১৯৪. সকল ধনাত্মক ও ঋণাত্মক পূর্ণসংখ্যাকে কী বলা হয়? /চ. বো. - ১৬/  
ক. ক্যারেক্টার খ. ইন্টিজার  
গ. রিয়াল ঘ. ডাবল
১৯৫. ফ্লোটিং ডেটার ফরমেট স্পেসিফায়ার কোনটি? /ঘ. বো. ১৯/  
ক. %d খ. %f  
গ. %c ঘ. %s

১৯৬. double ডেটা টাইপের জন্য ফরম্যাট স্পেসিফায়ার কোনটি? /রা. বো. -২০১৭/  
ক. %d খ. %f  
গ. %lf ঘ. %s
১৯৭. নিচের কোনটি সঠিক? /ঘ. বো. ১৯/  
ক. int number-1 খ. int number 1  
গ. int 1 number ঘ. int number\_1
১৯৮. printf() এর সাহায্যে ডেটা কোথায় পাঠানো হয়? /দি. বো. - ১৬/  
ক. ইনপুট মান ইনপুট মাধ্যমে  
খ. আউটপুট মান আউটপুট মাধ্যমে  
গ. ইনপুট মান আউটপুট মাধ্যমে  
ঘ. আউটপুট মান ইনপুট মাধ্যমে
১৯৯. নিচের কোনটি কী ওয়ার্ডের উদাহরণ? /কু. বো. ১৯/  
ক. long, int, scanf খ. short, cos, void  
গ. for, line, while ঘ. return, goto, break
২০০. সি ভাষায় রিলেশন অপারেটর কয় ধরনের? /চ. বো. ১৭/  
ক. ২ খ. ৩  
গ. ৫ ঘ. ৬
২০১. for (i = 1; i < 8; i++ = 2)  
printf ("%d", i);  
কোনটি উপরের স্টেটমেন্টের ফলাফল? /দি. বো. ১৬/  
ক. ১ ২ ৩ ৪ ৫ ৬ খ. ১ ৩ ৫ ৭  
গ. ২ ৪ ৬ ৮ ঘ. ১ ২ ৩ ৪ ৫ ৬ ৭ ৮
২০২. কোনটি সি-ভাষায় ব্যবহৃত কী-ওয়ার্ড? /রা. বো. ১৬/  
ক. img খ. for  
গ. select ঘ. href
২০৩. for (i = 1; i <= 5; i++)  
{ if (i == 3) continue;  
printf ("HSC Exam");  
}  
উদ্দীপকের প্রোগ্রামটিতে "HSC Exam" কতবার প্রদর্শিত হবে? /সকল বোর্ড ১৮/  
ক. ১ খ. ২  
গ. ৪ ঘ. ৫
২০৪. সি ভাষায় সমজাতীয় ডেটা সংরক্ষণের জন্য কোনটি ব্যবহার করা হয়? /সকল বোর্ড ২০১৮/  
ক. ফাংশন খ. পয়েন্টার  
গ. স্ট্রাকচার ঘ. অ্যারে
২০৫. নিচের কোনটি দ্বি-মাত্রিক অ্যারের উদাহরণ? /রা. বো. -২০১৭/  
ক. mark [5, 6] খ. mark (5, 6)  
গ. mark [5] [6] ঘ. mark (5) (6)

২০৬. বিট, বাইট, মেমরি অ্যাড্রেস নিয়ে কাজ করে—

[চ. বো. - ১৯/]

- i. মেশিন ভাষা
- ii. মধ্যস্তরের ভাষা
- iii. উচ্চস্তরের ভাষা
- নিচের কোনটি সঠিক?
- ক. i ও ii                      খ. i ও iii
- গ. ii ও iii                    ঘ. i, ii ও iii

২০৭. প্রোগ্রাম লিখতে মেশিন ভাষা ব্যবহার করা হলে—

[ব. বো. - ১৯/]

- i. প্রোগ্রাম পরিবর্তন করা কষ্টসাধ্য হয়
- ii. দক্ষ প্রোগ্রামার প্রয়োজন হয়
- iii. প্রোগ্রাম দ্রুত নির্বাহ হয়
- নিচের কোনটি সঠিক?
- ক. i ও ii                      খ. i ও iii
- গ. ii ও iii                    ঘ. i, ii ও iii

২০৮. মেশিন ভাষার প্রোগ্রাম —

[চ. বো. - ১৬/]

- i. সরাসরি ও দ্রুত কার্যকর হয়
- ii. কম্পিউটার সংগঠন বর্ণনা করে
- iii. লেখা সহজ ও সাধারণের ব্যবহার উপযোগী
- নিচের কোনটি সঠিক?
- ক. i ও ii                      খ. i ও iii
- গ. ii ও iii                    ঘ. i, ii ও iii

২০৯. বিট, বাইট, মেমরি অ্যাড্রেস নিয়ে কাজ করে— [চ. বো. - ১৯/]

- i. মেশিন ভাষা                      ii. মধ্যস্তরের ভাষা
- iii. উচ্চস্তরের ভাষা
- নিচের কোনটি সঠিক?
- ক. i ও ii                      খ. i ও iii
- গ. ii ও iii                    ঘ. i, ii ও iii

২১০. কম্পাইলারের সুবিধা হলো—

[মাদ্রাসা বোর্ড - ২০১৮/]

- i. সম্পূর্ণ প্রোগ্রামটি একবারে অনুবাদ করে
- ii. প্রোগ্রামে ডিবাগিং ও টেস্টিং দ্রুতগতি সম্পন্ন
- iii. ভুল থাকলে তা মনিটরে প্রদর্শন করে
- নিচের কোনটি সঠিক?
- ক. i ও ii                      খ. i ও iii
- গ. ii ও iii                    ঘ. i, ii ও iii

২১১. 'কম্পাইলার' ও 'ইন্টারপ্রেটার' এর মধ্যে পার্থক্য রয়েছে—

[দি. বো. - ১৬/]

- i. প্রোগ্রামটি অনুবাদের ক্ষেত্রে
- ii. কাজের গতির ক্ষেত্রে
- iii. ভুল প্রদর্শনের ক্ষেত্রে
- নিচের কোনটি সঠিক?
- ক. i ও ii                      খ. i ও iii
- গ. ii ও iii                    ঘ. i, ii ও iii

২১২. C প্রোগ্রামিং ভাষার জন্য কোন অনুবাদক প্রোগ্রাম ব্যবহৃত হয়?

[কু. বো. - ১৬/]

- i. কম্পাইলার
- ii. ইন্টারপ্রেটার
- iii. অ্যাসেম্বলার
- নিচের কোনটি সঠিক?
- ক. i                              খ. i ও ii
- গ. ii ও iii                    ঘ. i, ii ও iii

২১৩. প্রোগ্রাম তৈরির ধাপে কোডিং—

[চ. বো. - ১৬/]

- i. সমস্যার বিশ্লেষণের সাথে সম্পর্কিত
- ii. প্রোগ্রামিং ভাষার সাহায্যে করা
- iii. প্রোগ্রাম তৈরির পর ভুল খোঁজা
- নিচের কোনটি সঠিক?
- ক. i ও ii                      খ. i ও iii
- গ. ii ও iii                    ঘ. i, ii ও iii

২১৪. এই প্রতীকটির অর্থ হলো—

[মাদ্রাসা - ১৬/]

- i. ইনপুট
- ii. আউটপুট
- iii. প্রক্রিয়াকরণ
- নিচের কোনটি সঠিক?
- ক. i ও ii                      খ. i ও iii
- গ. ii ও iii                    ঘ. i, ii ও iii

২১৫. প্রোগ্রাম ডিজাইনের অন্তর্ভুক্ত কাজ হচ্ছে — [ব. বো. - ১৬/]

- i. অ্যালগরিদম প্রণয়ন
- ii. প্রবাহচিত্র তৈরি
- iii. সুডোকোড তৈরি
- নিচের কোনটি সঠিক?
- ক. i ও ii                      খ. i ও iii
- গ. ii ও iii                    ঘ. i, ii ও iii

২১৬. C ভাষায় লাইব্রেরি ফাংশন হলো— [চ. বো. - ১৯/]

- i. printf()
- ii. scanf()
- iii. add()
- নিচের কোনটি সঠিক?
- ক. i ও ii                      খ. i ও iii
- গ. ii ও iii                    ঘ. i, ii ও iii

২১৭. সি-ভাষার চলকগুলো লক্ষ করো—

[সকল বোর্ড - ২০১৮/]

- i. student\_name
- ii. student name
- iii. student@name
- নিচের কোনটি সঠিক?
- ক. i                              খ. iii
- গ. i ও iii                    ঘ. i, ii ও iii

২১৮. '%f' কাজ করে — /ফ. বো. ১৬/

- i. ইন্টিজার ii. ফ্লোট  
iii. রিয়েল

নিচের কোনটি সঠিক?

- ক. i ও ii খ. i ও iii  
গ. ii ও iii ঘ. i, ii ও iii

উদ্দীপকটি পড়ে ২১৯ ও ২২০ নং প্রশ্নের উত্তর:

জেরি সি-ভাষায় একটি প্রোগ্রাম রচনা করে যাতে দুইটি সংখ্যার যোগফল নির্ণয় করা যায়। প্রোগ্রামটি রান করার পর ২টি সংখ্যা প্রদান করলে ফলাফলে শুধু ২য় সংখ্যাটি প্রদর্শিত হয়। /সি. বো. ১৬/

২১৯. উদ্দীপকে উদ্ভূত সমস্যার কারণ কোনটি?

- ক. সঠিক হেডার ফাইল উল্লেখ না করা  
খ. ইনপুটে ভগ্নাংশ সংখ্যা প্রদান করা  
গ. আউটপুট ফাংশনে ভুল চলক ব্যবহার করা  
ঘ. প্রয়োজনীয় চলক ঘোষণা না করা

২২০. উদ্দীপকের ন্যায় প্রোগ্রাম তৈরির ক্ষেত্রে প্রয়োজন—

- i. বিশেষ ডেটাবেজ প্রোগ্রামিং ভাষা জানা থাকা  
ii. চলক ও ডেটা টাইপ সম্পর্কে ধারণা থাকা  
iii. ইনপুট আউটপুট সম্পর্কে সঠিক ধারণা থাকা

নিচের কোনটি সঠিক?

- ক. i ও ii খ. i ও iii  
গ. ii ও iii ঘ. i, ii ও iii

উদ্দীপকটি পড় এবং ২২১ ও ২২২ নং প্রশ্নের উত্তর দাও:

```
#include <stdio.h>
main ( )
{
    int i, s = 0;
    for (i = 1; i <= 6; i++)
    {
        s = s + i;
    }
    print f ("%d", s);
}
```

/সি. বো. ১৬/

২২১. প্রোগ্রামটির আউটপুট কত?

- ক. ৬ খ. ১৫  
গ. ১৯ ঘ. ২১

২২২. "i" এর মানের কী পরিবর্তনে আউটপুট ১২ হবে—

- ক.  $i = 0, i = i + 1$  খ.  $i = 1, i = i + 2$   
গ.  $i = 2, i = i + 1$  ঘ.  $i = 2, i = i + 2$

### উত্তরমালা

|     |   |     |   |     |   |     |   |     |   |     |   |     |   |     |   |     |   |     |   |     |   |     |   |
|-----|---|-----|---|-----|---|-----|---|-----|---|-----|---|-----|---|-----|---|-----|---|-----|---|-----|---|-----|---|
| ১   | ঘ | ২   | ক | ৩   | গ | ৪   | গ | ৫   | ক | ৬   | ক | ৭   | ঘ | ৮   | ঘ | ৯   | খ | ১০  | ক | ১১  | ঘ | ১২  | ক |
| ১৩  | খ | ১৪  | খ | ১৫  | ক | ১৬  | ক | ১৭  | ক | ১৮  | ক | ১৯  | ক | ২০  | গ | ২১  | খ | ২২  | গ | ২৩  | খ | ২৪  | খ |
| ২৫  | ক | ২৬  | ক | ২৭  | গ | ২৮  | ক | ২৯  | ক | ৩০  | খ | ৩১  | খ | ৩২  | ক | ৩৩  | ক | ৩৪  | খ | ৩৫  | ঘ | ৩৬  | ক |
| ৩৭  | ঘ | ৩৮  | ঘ | ৩৯  | খ | ৪০  | গ | ৪১  | খ | ৪২  | খ | ৪৩  | গ | ৪৪  | খ | ৪৫  | ঘ | ৪৬  | গ | ৪৭  | ক | ৪৮  | খ |
| ৪৯  | খ | ৫০  | খ | ৫১  | খ | ৫২  | ঘ | ৫৩  | ঘ | ৫৪  | খ | ৫৫  | ঘ | ৫৬  | ক | ৫৭  | গ | ৫৮  | খ | ৫৯  | খ | ৬০  | খ |
| ৬১  | খ | ৬২  | ঘ | ৬৩  | গ | ৬৪  | খ | ৬৫  | ক | ৬৬  | ক | ৬৭  | ক | ৬৮  | ঘ | ৬৯  | গ | ৭০  | খ | ৭১  | খ | ৭২  | গ |
| ৭৩  | গ | ৭৪  | খ | ৭৫  | খ | ৭৬  | ক | ৭৭  | ক | ৭৮  | খ | ৭৯  | ক | ৮০  | ক | ৮১  | ক | ৮২  | ক | ৮৩  | খ | ৮৪  | খ |
| ৮৫  | ক | ৮৬  | ক | ৮৭  | খ | ৮৮  | গ | ৮৯  | ঘ | ৯০  | ক | ৯১  | ঘ | ৯২  | খ | ৯৩  | খ | ৯৪  | খ | ৯৫  | ক | ৯৬  | গ |
| ৯৭  | খ | ৯৮  | খ | ৯৯  | খ | ১০০ | গ | ১০১ | খ | ১০২ | ঘ | ১০৩ | গ | ১০৪ | ঘ | ১০৫ | ক | ১০৬ | ক | ১০৭ | ক | ১০৮ | ক |
| ১০৯ | গ | ১১০ | খ | ১১১ | খ | ১১২ | খ | ১১৩ | ঘ | ১১৪ | গ | ১১৫ | খ | ১১৬ | খ | ১১৭ | ক | ১১৮ | গ | ১১৯ | ক | ১২০ | ক |
| ১২১ | খ | ১২২ | ঘ | ১২৩ | গ | ১২৪ | ক | ১২৫ | ঘ | ১২৬ | ক | ১২৭ | ক | ১২৮ | ক | ১২৯ | খ | ১৩০ | ক | ১৩১ | ক | ১৩২ | গ |
| ১৩৩ | ঘ | ১৩৪ | গ | ১৩৫ | ক | ১৩৬ | খ | ১৩৭ | ক | ১৩৮ | গ | ১৩৯ | ক | ১৪০ | ক | ১৪১ | ঘ | ১৪২ | গ | ১৪৩ | ঘ | ১৪৪ | ক |
| ১৪৫ | খ | ১৪৬ | ঘ | ১৪৭ | ক | ১৪৮ | ক | ১৪৯ | ঘ | ১৫০ | খ | ১৫১ | গ | ১৫২ | গ | ১৫৩ | গ | ১৫৪ | ক | ১৫৫ | খ | ১৫৬ | খ |
| ১৫৭ | গ | ১৫৮ | ঘ | ১৫৯ | ক | ১৬০ | ক | ১৬১ | গ | ১৬২ | খ | ১৬৩ | গ | ১৬৪ | ক | ১৬৫ | খ | ১৬৬ | খ | ১৬৭ | গ | ১৬৮ | গ |
| ১৬৯ | ক | ১৭০ | খ | ১৭১ | ক | ১৭২ | খ | ১৭৩ | গ | ১৭৪ | ক | ১৭৫ | খ | ১৭৬ | খ | ১৭৭ | ক | ১৭৮ | খ | ১৭৯ | খ | ১৮০ | ক |
| ১৮১ | ক | ১৮২ | ঘ | ১৮৩ | ঘ | ১৮৪ | গ | ১৮৫ | খ | ১৮৬ | খ | ১৮৭ | খ | ১৮৮ | ঘ | ১৮৯ | খ | ১৯০ | খ | ১৯১ | খ | ১৯২ | গ |
| ১৯৩ | খ | ১৯৪ | খ | ১৯৫ | খ | ১৯৬ | গ | ১৯৭ | ঘ | ১৯৮ | গ | ১৯৯ | ঘ | ২০০ | ঘ | ২০১ | খ | ২০২ | খ | ২০৩ | গ | ২০৪ | ঘ |
| ২০৫ | গ | ২০৬ | ক | ২০৭ | ঘ | ২০৮ | ক | ২০৯ | ক | ২১০ | খ | ২১১ | ঘ | ২১২ | খ | ২১৩ | ক | ২১৪ | ক | ২১৫ | ঘ | ২১৬ | ক |
| ২১৭ | ক | ২১৮ | গ | ২১৯ | গ | ২২০ | গ | ২২১ | ঘ | ২২২ | ঘ |     |   |     |   |     |   |     |   |     |   |     |   |

## গ. সংক্ষিপ্ত প্রশ্ন

- কোন ভাষার মাধ্যমে কম্পিউটারের সাথে সরাসরি যোগাযোগ করা যায়? বর্ণনা করো।
- মেশিন ভাষার সুবিধাসমূহ লিখো।
- যান্ত্রিক ও অ্যাসেম্বলি ভাষার মধ্যে তুলনামূলক পার্থক্য লিখো।
- অনুবাদক প্রোগ্রাম বলতে কী বোঝ? সংক্ষেপে বর্ণনা করো।
- কম্পাইলারের সুবিধাসমূহ লিখো।
- প্রোগ্রাম তৈরির ধাপসমূহ সংক্ষেপে লিখো।
- ফ্ল্যাচার্ট তৈরি করার নিয়মাবলী লিখো।
- অ্যালগরিদম ও ফ্লোচার্ট এর পার্থক্য লিখো।
- অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং বলতে কী বোঝ?
- প্রোগ্রামিং-এ হেডার ফাইল খুবই গুরুত্বপূর্ণ— বর্ণনা করো।
- লোকাল ও গ্লোবাল ভেরিয়েবলের মধ্যে তুলনামূলক বর্ণনা করো।
- সি প্রোগ্রামে পোস্টফিক্স ও প্রিফিক্স বলতে কী বোঝ?
- ফরম্যাট স্পেসিফায়ার বলতে কী বোঝ? %c, %d, %ld, %f, %lf সম্পর্কে লিখো।
- ইনক্রিমেন্ট ও ডিক্রিমেন্ট অপারেটর কাকে বলে? উদাহরণ দাও।
- প্রোগ্রামে অ্যারে স্ট্রাকচারের সুবিধা ও অসুবিধা লিখো।
- প্রোগ্রামে ফাংশনের প্রয়োজনীয়তা লিখো।

## ঘ. সৃজনশীল প্রশ্ন

- একটি কলেজের আইসিটি শিক্ষক শিক্ষার্থীদের প্রোগ্রামিং ভাষা ও প্রোগ্রাম রচনার বিভিন্ন ধাপ সম্পর্কে আলোচনা করছিলেন। এর মধ্যে কয়েকজন শিক্ষার্থী প্রবাহচিত্র সম্পর্কে বুঝতে না পারায় শিক্ষক বোর্ডে একটি প্রবাহচিত্র এঁকে তা বুঝিয়ে দিলেন এবং শিক্ষার্থীদের তিনটি সংখ্যা থেকে বৃহত্তম সংখ্যাটি নির্ণয়ের অ্যালগরিদম ও প্রবাহচিত্র তৈরি করতে বললেন।
  - টেস্টিং কী?
  - হাইলেবেল ভাষায় প্রোগ্রামিং করা সহজ— ব্যাখ্যা করো।
  - শিক্ষকের প্রদানকৃত অ্যালগরিদম ও প্রবাহচিত্রটি তৈরি করে দেখাও।
  - উদ্দীপকের সমস্যাটি সি-ভাষায় প্রোগ্রাম রচনা করো।
- কম্পিউটার সায়েন্সের ছাত্র পলাশ ‘সি’ ভাষায় কিছু উৎস কোড লিখল। ফাইলটি X নামে সংরক্ষণ করলো। এরপর সে ফাইলটিকে কম্পাইল করে উৎস কোডকে অবজেক্ট কোডে পরিণত করল এবং অবজেক্ট কোডকে Y নামে সংরক্ষণ করলো। কাজটি শেষ হলে পলাশের ছোট বোন চারু তাকে জিজ্ঞাসা করলো কেন তুমি উৎস কোডকে অবজেক্ট কোডে রূপান্তর করলে?
  - অনুবাদক প্রোগ্রাম কী?
  - কম্পাইলার ও ইন্টারপ্রেটারের মধ্যে দুটি পার্থক্য লেখ।
  - চারুর প্রশ্নের উত্তর তুমি কীভাবে দিবে? বর্ণনা করো।
  - X ও Y ফাইল দুটির মধ্যে কোনটি পলাশের জন্য অনুধাবন করা সহজ? উত্তরের স্বপক্ষে যুক্তি দাও।
- প্রোগ্রামটি লক্ষ্য কর এবং প্রশ্নগুলোর উত্তর দাও:
 

```
main()
{
    int a[10],i,s=0;
    for(i=1;i<=4;++i)
    {
        printf("Type the marks(%d): ",i);
        scanf("%d",&a[i]);
        s=s+a[i];
    }
    printf("SUM=%d",s);
}
```

  - অ্যালগরিদম কী?
  - মেশিন ভাষায় লিখিত প্রোগ্রাম দ্রুত নির্বাহ হয় কেন?
  - উদ্দীপকের প্রোগ্রামটির প্রবাহ চিত্র অংকন কর।
  - উদ্দীপকের “প্রোগ্রামটি লুপ কন্ট্রোল স্টেটমেন্ট ছাড়াও সমাধান সম্ভব”— কোডিংসহ ব্যাখ্যা কর।
- ধারা নির্ণয়ের প্রোগ্রাম দুটি লক্ষ্য কর:
 

```
main()
{
    int a, n;
    for(a=1;a<=10;a=a+1)
    {
        printf("%d\t",a);
    }
}
```

প্রোগ্রাম-১

```
main()
{
    int a,i,n,j;
    printf("First term: ");
    scanf("%d",&a);
    printf("Increment: ");
    scanf("%d",&i);
    printf("Last term: ");
    scanf("%d",&n);
    printf("Series: ");
    for(j=a;j<=n;j=j+i)
    {
        printf("%d\t",j);
    }
}
```

প্রোগ্রাম-২

  - ধুবক কী?
  - scanf (“%mf”, &a); স্টেটমেন্টটি ব্যাখ্যা কর।
  - প্রোগ্রাম-২ এর গতিধারা সহজে বুঝানোর উপায় দেখাও।
  - প্রোগ্রাম-১ এবং প্রোগ্রাম-২ এর মধ্যে কোনটিকে তুমি উত্তম বলে মনে কর — বিশ্লেষণ পূর্বক মতামত দাও।

৫. প্রোগ্রাম দুটি লক্ষ কর এবং প্রশ্নগুলোর উত্তর দাও।  
 1: scanf("%d",&n); 1: scanf("%d",&n);  
 2: s=0; 2: s=0;  
 3: do 3: while(n<=10)  
 4: { 4: {  
 5: s=s+n; 5: s=s+n;  
 6: n=n+1; 6: n=n+1;  
 7: }while(n<=10); 7: }  
 8: printf("%d",s); 8: printf("%d",s);

প্রোগ্রাম-১

প্রোগ্রাম-২

- ক. চলক কী?  
 খ. সি (C) একটি কেস সেনসিটিভ ভাষা— বুঝিয়ে লেখ।  
 গ. প্রোগ্রাম-২ এর অ্যালগরিদম লেখ এবং ব্যাখ্যা কর।  
 ঘ. n>10 এর জন্য প্রোগ্রাম-১ ও প্রোগ্রাম-২ এর আউটপুট একই হওয়ার জন্য প্রোগ্রামে কি ধরনের পরিবর্তন আনতে হবে- বিশ্লেষণ করো।

৬. #include <stdio.h>  
 #include <conio.h>  
 main()  
 {  
 int a, s;  
 s = 0;  
 for (a = 1; a <= 30; a = 2)  
 {  
 s = s + a;  
 }  
 printf("sum = %d", s);  
 getch();  
 }

[রা. বো. ১৯]

- ক. সংরক্ষিত শব্দ কী?  
 খ. K++ ও ++K ব্যাখ্যা করো।  
 গ. উদ্দীপকের প্রোগ্রামটির জন্য একটি প্রবাহচিত্র অঙ্কন করো।  
 ঘ. উদ্দীপকের প্রোগ্রামটি while লুপ ব্যবহার করে তৈরি করা সম্ভব কিনা— বিশ্লেষণ করো।

৭. বার্ষিক ক্রীড়া প্রতিযোগিতায় একাদশ শ্রেণীর শিক্ষার্থীদের A, B ও C দলে বিভক্ত করা হয়। রোল নম্বর 1 থেকে 30 পর্যন্ত A দলে, 31 থেকে 60 পর্যন্ত B দলে এবং 61 থেকে 100 পর্যন্ত C দলে অন্তর্ভুক্ত হবে। [ঢা., দি., সি., য. বো. ১৮]

- ক. অবজেক্ট প্রোগ্রাম কী?  
 খ. সি একটি কেস সেনসিটিভ ভাষা— বুঝিয়ে লেখ।  
 গ. উদ্দীপকে উল্লিখিত দল গঠনের জন্য অ্যালগরিদম লেখ।  
 ঘ. সি ভাষায় কন্ডিশনাল স্টেটমেন্ট ব্যবহার করে দল গঠনের জন্য একটি প্রোগ্রাম রচনা করো।

৮. দুটি সংখ্যার যোগফল নির্ণয়ের প্রোগ্রাম লক্ষ্য করো:  
 #include <stdio.h> #include <stdio.h>  
 main() main()

{  
 int a = 10, b = 15;  
 int c = a + b;  
 printf("%d", c);  
 }  
 {  
 int a, b, c;  
 scanf("%d %d", &a, &b);  
 c = a + b;  
 printf("%d", c);  
 }

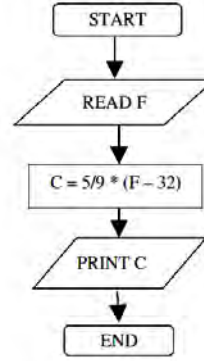
প্রোগ্রাম-১

প্রোগ্রাম-২

[কু. বো. ১৯]

- ক. ধ্রুবক কী?  
 খ. scanf("%f", &a); স্টেটমেন্টটি ব্যাখ্যা করো।  
 গ. প্রোগ্রাম-১ এর প্রবাহচিত্র অংকন কর।  
 ঘ. প্রোগ্রাম-১ ও প্রোগ্রাম-২ এর মধ্যে কোনটিকে তুমি উত্তম বলে মনে কর? বিশ্লেষণপূর্বক মতামত দাও।

৯.



[ঢা. বো. ১৬]

- ক. কম্পাইলার কী?  
 খ. অ্যালগরিদম কোডিং-এর পূর্বশর্ত— ব্যাখ্যা করো।  
 গ. উদ্দীপকের সমস্যাটির “সি” ভাষায় একটি প্রোগ্রাম লিখ।  
 ঘ. উদ্দীপকের ধারণা প্রোগ্রাম তৈরি ধাপের সাথে কিভাবে সম্পর্কিত? বিশ্লেষণ করো।

১০. প্রোগ্রামটি দেখ এবং নিচের প্রশ্নগুলোর উত্তর দাও:

```

#include<stdio.h>
#define a 3.1416
main()
{
    int r;
    float area;
    printf("Type the radius: ");
    scanf("%d",&r);
    area=a*r*r;
    printf("Area=%f",area);
}
    
```

- ক. প্রোগ্রামের ভিত্তি কী?  
 খ. উক্ত প্রোগ্রামে & ব্যবহৃত হয়েছে কেন?  
 গ. উক্ত প্রোগ্রামে r ব্যবহারের সুবিধা বর্ণনা করো।  
 ঘ. প্রোগ্রামে a এবং r এর ব্যবহারিক পার্থক্য নিরূপণ করো।

```

১১. #Include (stdio.h)
void main ( )
{
    int i, S=0;
    Print f ("Enter last number =")
    Scan f ("%d", n)
    I = 10;
    while (i < n)
    {
        S = S + i
        i = i + 10
    }
    Print f ("Sum = %d" s)
}

```

[য. বো. ১৯]

- ক. হেডার ফাইল কী?  
 খ. C ও C++ এর মধ্যে ভিন্নতা কী? ব্যাখ্যা কর।  
 গ. উদ্দীপকের প্রোগ্রামটি ডিবাগিং কর।  
 ঘ. উদ্দীপক প্রোগ্রামটি goto লুপ দিয়ে বাস্তবায়ন সম্ভব— দেখাও।

১২. প্রোগ্রামটি দেখ এবং নিচের প্রশ্নগুলোর উত্তর দাও:

```

main()
{
    int a;
    for(a=1; a<=10;a++)
    {
        printf("%d",a);
    }
}

```

- ক. ফরম্যাট স্পেসিফায়ার কী?  
 খ. \n এবং \r এর ব্যবহারিক তুলনামূলক পার্থক্য দেখাও।  
 গ. do-loop ব্যবহার করে উক্ত প্রোগ্রামটি লেখ।  
 ঘ. অসীম লুপ এর জন্য প্রোগ্রামটিতে কী পরিবর্তন আনতে হবে? বিশ্লেষণ করো।

```

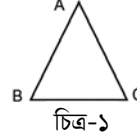
১৩. #include <stdio.h>
#include <conio.h>
int main ( )
{
    int i, Sum, n;
    clrscr ( );
    printf ("Enter the value of n = ");
    scanf ("%d", & n);
    Sum = 0;
    for ( i = 1; i <= n; i++)
    Sum = Sum + i;
    printf ("n\sum of all numbers from 1 to %d is = %d", n, Sum);
    getch ( );
}return 0;

```

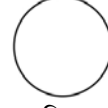
[রা., কু., চ., ব. বো. ১৮]

- ক. চলক কী?  
 খ. ডকুমেন্টেশন কেন করতে হয়?  
 গ. উদ্দীপকের প্রোগ্রামটির অ্যালগরিদম লিখ।  
 ঘ. উদ্দীপকের কোডে ব্যবহৃত লুপের পরিবর্তে do while লুপ ব্যবহার করে প্রোগ্রামটি লিখ।

১৪. চিত্র দুটি লক্ষ করো:



চিত্র-১



চিত্র-২

তন্মূলা কম্পিউটারে C প্রোগ্রাম ব্যবহার করে চিত্র-২ এ অংকিত বিষয়টির ক্ষেত্রফল নির্ণয় করল। ঐশী চিত্র-১ এর ক্ষেত্রফল ধাপে ধাপে ও চিত্রের সাহায্যে নির্ণয়ের ব্যবস্থা করল। [কু. বো. ১৬]

- ক. প্রোগ্রাম কী?  
 খ. অনুবাদক প্রোগ্রাম হিসেবে কম্পাইলার বেশি উপযোগী— ব্যাখ্যা করো।  
 গ. উদ্দীপকে ঐশীর চিত্র-১ এর ক্ষেত্রফল নির্ণয়ের ফ্লোচার্ট অংকন করো।  
 ঘ. তন্মূলা চিত্রটির ক্ষেত্রফল নির্ণয়ের প্রোগ্রাম লিখ। ব্যাসার্ধ ও এর ক্ষেত্রে ফলাফলের সত্যতা যাচাই করো।

১৫. নিচের ধারাটি লক্ষ করো:

$$7 + 14 + 21 + \dots + 100$$

- ক. 'সি' ভাষার জনক কে?  
 খ. 'সি' ভাষাকে কেন Mid Level ভাষা হয়? বুঝিয়ে লেখ।  
 গ. ধারাটির ১০ম পদ নির্ণয়ের প্রোগ্রাম লেখ।  
 ঘ. do-while লুপ ব্যবহার করে ধারাটির যোগফল নির্ণয়ের ক্ষেত্রে লুপটি কত বার ঘুরবে তা ধারাবাহিকভাবে বিশ্লেষণ করো।

১৬. (i)  $1+4+7+\dots+N$ । (ii) কোন সংখ্যার ফ্যাক্টোরিয়াল মান নির্ণয়ের ফর্মুলা:  $n!=n(n-1)(n-2)\dots\dots\dots 1$

- ক. মেশিন ভাষা কী?  
 খ. সি-ভাষায় চলকের নামকরণে কিছু নিয়মকানুন মেনে চলতে হয় কেন?  
 গ. উদ্দীপকে উল্লিখিত (i)-এর যোগফল নির্ণয়ের জন্য ফ্লোচার্ট আঁক।  
 ঘ. উদ্দীপকে উল্লিখিত (ii)-এর ফ্যাক্টোরিয়াল মান নির্ণয়ের 'সি'-ভাষায় প্রোগ্রাম লেখ।

১৭. রিয়েল তার ভাই রিয়াদকে বলল, প্রথম ১০০টি ধনাত্মক পূর্ণ সংখ্যার যোগফল কত? রিয়াদ তাকে For Loop ব্যবহার করে একটি 'সি' প্রোগ্রাম লিখে দিল এবং বলল এটি নির্বাহ করলে যোগফলটি পাওয়া যাবে।

- ক. অ্যাসেম্বলি ভাষা কী?  
 খ. কিওয়ার্ডকে ভেরিয়েবল হিসেবে ব্যবহার করা যায় না কেন?  
 গ. উদ্দীপকে উল্লিখিত সমস্যার 'সি' ভাষার প্রোগ্রাম কোড লিখ।  
 ঘ. উদ্দীপকে উল্লিখিত সমস্যার সমাধান আর কী কী ভাবে করা যেত, তুলনামূলক আলোচনা করো।



১৮. কলেজের আইসিটি শিক্ষক শিক্ষার্থীকে ১ থেকে ১০০ পর্যন্ত যোগ করে যোগফল নির্ণয়ের জন্য সি প্রোগ্রাম লিখতে বললেন। শিক্ষার্থী যে প্রোগ্রামটি লিখলেন তা নিম্নরূপঃ

```
#include<stdio.h>
#include<conio.h>
main ( )
{
int a, m, n, s = 0;
scanf ("%d %d", &m, &n);
for (a = m; a<=n; a++)
{
s = s + a ;
}
printf ("Sum = %d", s) ;
getch ( ) ;
}
```

ক. 4GL কী?

খ. সি-প্রোগ্রামিং-এ #include <stdio.h> আবশ্যিক কেন? ব্যাখ্যা কর।

গ. শিক্ষকের নির্দেশ মোতাবেক সিরিজটির যোগফল নির্ণয়ের জন্য অ্যালগরিদম লেখ।

ঘ. দুইটি প্রোগ্রামের মধ্যে কোনটি সুবিধাজনক বিশ্লেষণপূর্বক মতামত দাও।

১৯.  $1+3^0+.....+n^n$  এই সিরিজের যোগফল নির্ণয় করার জন্য সি ভাষায় প্রোগ্রাম লেখ।

```
#include <stdio.h>
void main( )
{
int n,c,s=0;
scanf ("%d", &n);
for(c=1;c<=n;c++)
{
s=s+c*c
printf ("%d",s);
}
```

ক. কম্পাইলার কী?

খ. for এবং do লুপ দুটির মধ্যে কোনটি ব্যবহার করা সহজ?

গ. উল্লিখিত সিরিজটির জন্য একটি ফ্লোচার্ট অংকন করো।

ঘ. সি প্রোগ্রামটিতে কী কী সমস্যা আছে তা বিশ্লেষণপূর্বক মতামত দাও।

২০.

```
#include < >
void main()
{
printf("enter two number");

do
{
t=i%S;
i=s;
s=t;
} While (s!=0)
printf("GCD is %d",i);
getch();
}
```

ক. ডিবাগিং কী?

খ. উৎস কোডকে কম্পাইল করার প্রয়োজন হয় কেন? ব্যাখ্যা করো।

গ. উদ্দীপকের সি-প্রোগ্রামটির ফ্লোচার্ট কী ধরনের হবে—বর্ণনা করো।

ঘ. সঠিক প্রোগ্রামে চলক i এবং s এর মান যদি ইনপুট হিসেবে 35 ও 20 দেওয়া হয় তবে লুপের প্রতিটি ধাপে কি ঘটবে—বিশ্লেষণ করো।

২১. নিচের উদ্দীপকটি লক্ষ করো এবং প্রশ্নগুলোর উত্তর দাও:

```
# include < stdio. h>
main ( )
{
int SUM, N;
printf ("Enter the last number");
scanf ("%d", &N);
SUM = 0
for (i = 1; i ≤ N; i = i + 3)
{
SUM = SUM + i;
}
printf("Result: % d", SUM);
}
```

/ব. বো. ১৭/

ক. সুডোকোড কী?

খ. অনুবাদক প্রোগ্রাম হিসেবে কম্পাইলার বেশি উপযোগী—ব্যাখ্যা করো।

গ. উপরের উদ্দীপকটির ফ্লোচার্ট অংকন করো।

ঘ. উপরের উপদ্দীপকটি do ..... while লুপের সাহায্যে করতে হলে কোডের কি পরিবর্তন করতে হবে—বিশ্লেষণ করো।

২২. (i)  $\frac{C}{5} = \frac{F-32}{9}$  (ii)  $1^3 + 2^3 + ..... + N^3$

ক. Syntax Error কাকে বলে?

খ. সি একটি উচ্চস্তরের ভাষার প্রোগ্রাম—ব্যাখ্যা করো।

গ. (i) নং উদ্দীপকের সেন্টিগ্রেডকে ফারেনহাইটের রূপান্তরের জন্য একটি ফ্লোচার্ট তৈরি করো।

ঘ. (ii) নং উদ্দীপকে উল্লিখিত সমস্যাটির 'সি' ভাষায় প্রোগ্রাম লিখ।

২৩. মাধবী কম্পিউটারে বসে নিম্নোক্ত প্রোগ্রামটি টাইপ করলো:

```
# include<stdio.h>
#include<conio.h>
main ( )
{
int i, sum=0 n;
printf ("Enter the value of n=");
Scanf ("%d", & n);
for (i=1; i<n; i++)
sum=sum+1;
printf("In total of series is % d", sum);
getch ( );
}
```



- ক. ন্যাচারাল ল্যাঙ্গুয়েজ কী?  
খ. Scanf (“%f”, &a) স্টেটমেন্টটি ব্যাখ্যা করো।  
গ. মাদারবীর প্রোগ্রামটির প্রবাহচিত্র লিখ।  
ঘ. উদ্দীপকের প্রোগ্রামটি do-while লুপ ব্যবহার করে লেখা যায় বর্ণনা করো।

২৪. আইসিটি বিষয়ের শিক্ষক ক্লাসে ছাত্রদের অপারেটর, চলক, ডেটা টাইপ, ও বিভিন্ন স্টেটমেন্ট সম্পর্কে পাঠদান করছিলেন এবং স্টেটমেন্ট ব্যবহার করে কীভাবে সি-ভাষায় (i) স্বাভাবিক সংখ্যার মধ্যে জোড় সংখ্যাগুলোর যোগফল ও (ii) GCD নির্ণয় করার জন্য প্রোগ্রাম লিখতে হয় তা ছাত্রদের বোঝালেন।

- ক. লুপ কী?  
খ. প্রোগ্রামে অপারেটরের গুরুত্ব লেখ।  
গ. উদ্দীপকে উল্লিখিত (i)-এর সি-ভাষায় প্রোগ্রাম লেখ।  
ঘ. উদ্দীপকে উল্লিখিত (ii)-এ কন্ট্রোল স্টেটমেন্ট ব্যবহার করে কীভাবে প্রোগ্রাম লিখতে হয়? মতামত দাও।

২৫. 

```
int i,sum=0
for (i=1 ; i<=n; i++)
{
    scanf(“%d”,&n);
    sum=sum+i;
}
```

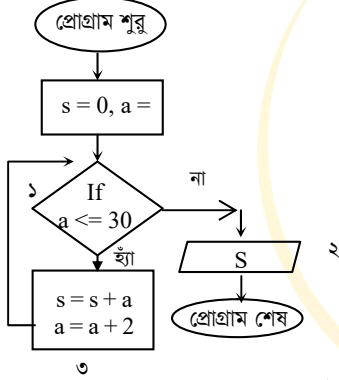
- ক. ডেটা টাইপ কী?  
খ. মেশিন ভাষায় কমান্ড-এর প্রয়োজন হয় না কেন?  
গ. উদ্দীপকের আলোকে for স্টেটমেন্টটির বর্ণনা দাও।  
ঘ. উদ্দীপকে কোন ভুল থাকলে তা সংশোধনপূর্বক প্রোগ্রামটি পরিপূর্ণ করতে কী পরিবর্তন আনতে হবে বিশ্লেষণ করো।

২৬.

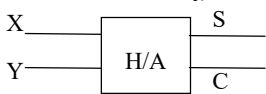
- ধাপ-১ঃ শুরু করো।  
ধাপ-২ঃ তিনটি সংখ্যা P, Q, R গ্রহণ করো।  
ধাপ-৩ঃ P কে Q দ্বারা গুণ কর এবং গুণফলকে K-এ রাখ।  
ধাপ-৪ঃ K কে R দিয়ে ভাগ কর এবং ভাগফল S-এ রাখ।  
ধাপ-৫ঃ K কে ছাপাও।  
ধাপ-৬ঃ S কে ছাপাও।  
ধাপ-৭ঃ শেষ কর।  
ক. ভেরিয়েবল কাকে বলে?  
খ. গ্লোবাল ভেরিয়েবল main() ফাংশনের উপরে ব্যবহার করা হয় কেন? ব্যাখ্যা করো।  
গ. উদ্দীপকে উল্লিখিত অ্যালগরিদমের জন্য একটি ফ্লোচার্ট তৈরি করো।  
ঘ. উদ্দীপকে উল্লিখিত অ্যালগরিদমটির জন্য ‘সি’ ভাষায় একটি প্রোগ্রাম তৈরি করো।

## ঙ. সমন্বিত অধ্যায়ের সৃজনশীল প্রশ্ন

১. উদ্দীপকটি পড় এবং প্রশ্নগুলোর উত্তর দাও।



- ক. এইচটিএমএল টেক্সট ফরম্যাটিং কী?  
খ. নিম্নস্বরের ভাষা বলতে কী বোঝ?  
গ. প্রবাহ চিত্রটি যে কার্যকারিতা প্রদর্শন করে এক্সিট কন্ট্রোল লুপ করে সি ভাষায় প্রোগ্রাম লিখ।  
ঘ. উদ্দীপকের ১ নং কে হোমপেইজ, ২ ও ৩ নং কে মূলধারার পেইজ ধরে লিনিয়ার ও মেনু স্ট্রাকচারের জন্য লেআউট এবং html কোড দেখাও।
২. উদ্দীপকটি পড় এবং প্রশ্নগুলোর উত্তর দাও।



- ক. দ্বৈত নীতি কী?  
খ. প্রোগ্রাম ডকুমেন্টেশন বলতে কী বোঝ?  
গ. শুধু NAND গেইট ব্যবহার করে সার্কিটটি বাস্তবায়ন কর।  
ঘ. সি ভাষা ব্যবহার করে সার্কিটটি বাস্তবায়ন সম্ভব কিনা - বিশ্লেষণের মাধ্যমে দেখাও।
৩. দেওয়া আছে,  
 $X=(13.5)_8, Y=(15.5)_{10}, Z=(17.5)_{16}$   
ক. বিট কী?  
খ. নিচের চলকগুলো কেন ভুল- ব্যাখ্যা কর।  
`int, main, private, number-1`  
গ. ২' এর পরিপূরক ব্যবহার করে প্রথম ও দ্বিতীয় সংখ্যার পূর্ণাংশের পার্থক্য নির্ণয় কর।  
ঘ. উদ্দীপকের সংখ্যাগুলো যাই থাকুক না কেন তাদের পূর্ণাংশকে দশমিক ধরে যে ধারা তৈরি হয় তার প্রথম n সংখ্যক পদের যোগফল নির্ণয়ের জন্য সি ভাষায় প্রোগ্রাম লেখ।
৪. রহিম দোকান থেকে  $(30)_8$  টাকার খাতা,  $(AB)_{16}$  টাকার কলম কিনে দোকানদারকে  $(500)_{10}$  টাকার ১টি নোট দিলেন।  
ক. লুপ কী?  
খ. ৪ বিটের কোন কোডগুলো BCD-তে ব্যবহৃত হয় না এবং কেন?  
গ. রহিম কত টাকা ফেরত পাবে তা হেক্সাডেসিমেল সংখ্যা পদ্ধতিতে দেখাও।  
ঘ. রহিম কত টাকা ফেরত পাবে তা নির্ণয়ের ফ্লোচার্টটি আঁক এবং এর স্বপক্ষে যুক্তি দাও।

## চ. সৃজনশীল জ্ঞান ও অনুধাবনমূলক প্রশ্ন

## ► জ্ঞানমূলক

## প্রোগ্রামের ধারণা

১. প্রোগ্রাম কী? /রা. বো. ১৭; দি. কু. বো. ১৬/
২. প্রোগ্রামিং কী?
৩. প্রোগ্রামের ভাষা কী?
৪. মেশিন ভাষা কী?
৫. অ্যাসেম্বলি ভাষা কী?
৬. মধ্যম স্তরের ভাষা কী?

## বিভিন্ন উচ্চস্তরের ভাষা সম্পর্কে আলোচনা

৭. উচ্চস্তরের ভাষা কী?
৮. 4GL কী? /য. বো. ১৬; ঢা. বো. ১৯/
৯. পঞ্চম প্রজন্মের ভাষা কী?
১০. ন্যাচারাল ল্যাঙ্গুয়েজ কী?

## অনুবাদক প্রোগ্রাম কম্পাইলার, অ্যাসেম্বলার, ইন্টারপ্রেটার

১১. অনুবাদক প্রোগ্রাম কী? /ব. বো. ১৯/
১২. অ্যাসেম্বলার কী?
১৩. ইন্টারপ্রেটার কী?
১৪. কম্পাইলার কী?
১৫. অবজেক্ট প্রোগ্রাম কী? /ঢা., দি., সি., য. বো. ১৮/

## প্রোগ্রামের সংগঠন

১৬. টেস্টিং কী?
১৭. ডিবাগিং কী?
১৮. বাগিং কী?
১৯. রান টাইম এরর কী? /সি. বো. ১৯/
২০. যৌক্তিক ত্রুটি কী?
২১. Syntax Error কী? /সি. বো. ১৬/

## অ্যালগরিদম ও ফ্লোচার্ট

২২. অ্যালগরিদম কাকে বলে? /দি. বো. ১৯/
২৩. ফ্লোচার্ট কী?
২৪. সুডোকোড কী? /ব. বো. ১৭; য. বো. ১৯/

## প্রোগ্রাম ডিজাইন মডেল

২৫. স্ট্রাকচার্ড প্রোগ্রামিং মডেল কাকে বলে?
২৬. OPP মডেল কাকে বলে?
২৭. ভিজুয়াল প্রোগ্রামিং মডেল কাকে বলে?
২৮. ইভেন্ট ড্রাইভেন প্রোগ্রামিং মডেল কাকে বলে?
২৯. পলিমরফিজম কী?

## ‘সি’ প্রোগ্রাম

৩০. ‘সি’ ভাষার জনক কে?
৩১. লিংকার কী?
৩২. হেডার ফাইল কী? /য. বো. ১৯/
৩৩. General Purpose প্রোগ্রামিং ভাষা কী?

## ‘সি’ ভাষায় ব্যবহৃত ডেটা টাইপ

৩৪. ডেটা টাইপ কী?

৩৫. ডেটা টাইপ মডিফায়ার বলতে কী বুঝায়?

৩৬. ডিরাইভড ডেটা টাইপ কী?

৩৭. ফাঁকা বা এম্পটি ডেটা সেট কী?

৩৮. ইউজার ডিফাইন ডেটা টাইপ কী?

## ‘সি’ ভাষায় ব্যবহৃত চলক ও ধ্রুবক

৩৯. ধ্রুবক কী? /ঢা. বো. ১৭; কু. বো. ১৯/

৪০. চলক কী? /রা., কু., চ., ব. বো. ১৮; ঢা., চ. বো. ১৯/

৪১. লোকাল ও গ্লোবাল ভেরিয়েবল কী?

৪২. বিন্ট ইন ভেরিয়েবল কী?

## রাশিমালা

৪৩. অ্যাসাইনমেন্ট অপারেটর কী? /দি. বো. ১৭/

৪৪. ইউনারি ও বাইনারি অপারেটর কী?

৪৫. টোকেন বলতে কী বোঝ?

৪৬. Presidency কী?

৪৭. কীওয়ার্ড কী? /চ. বো. ১৯/

## ইনপুট / আউটপুট স্টেটমেন্ট

৪৮. ইনপুট স্টেটমেন্ট কী?

৪৯. আউটপুট স্টেটমেন্ট কী?

৫০. ফরম্যাট স্পেসিফায়ার কী?

## কন্ট্রোল ও কন্ডিশনাল কন্ট্রোল স্টেটমেন্ট

৫১. কন্ট্রোল স্টেটমেন্ট কী?

৫২. জাম্পিং স্টেটমেন্ট কী?

## লুপ ও লুপের ব্যবহার

৫৩. লুপ কী?

৫৪. এন্ট্রি ও এক্সিট কন্ট্রোল লুপ কী?

## অ্যারে ও অ্যারের ব্যবহার

৫৫. অ্যারে কী?

৫৬. অ্যারে ডেটা টাইপ কিভাবে ডিক্লেয়ার করা যায়?

## ফাংশন ও ফাংশনের ব্যবহার

৫৭. ফাংশন কী?

## ► অনুধাবনমূলক

## প্রোগ্রামের ধারণা

১. মেশিন ভাষায় কমান্ড-এর প্রয়োজন হয় না কেন?
২. যান্ত্রিক ভাষাকে নিম্নস্তরের ভাষা বলা হয় কেন?
৩. ০, ১ দিয়ে লেখা ভাষা ব্যাখ্যা কর। /চ., য. বো. ১৬/
৪. মেশিন ভাষায় লিখিত প্রোগ্রাম দ্রুত নির্বাহ হয় কেন?
৫. মেশিন ভাষার সুবিধা ও অসুবিধা লেখ।
৬. অ্যাসেম্বলি ভাষা মেশিন ভাষার চেয়ে উত্তম কেন?
৭. অ্যাসেম্বলি ভাষাকে অনুবাদ করার জন্য কোন বিশেষ সফটওয়্যার ব্যবহার করা হয়?
৮. অ্যাসেম্বলি ভাষার সুবিধা ও অসুবিধা লেখ।
৯. যান্ত্রিক ভাষা ও অ্যাসেম্বলি ভাষার মধ্যে পার্থক্য দেখাও।

### বিভিন্ন উচ্চস্তরের ভাষা সম্পর্কে আলোচনা

১০. কোন ভাষায় প্রোগ্রাম লিখলে কম্পিউটার সরাসরি বুঝতে পারে না? ব্যাখ্যা কর।
১১. High level language-এর অসুবিধা ব্যাখ্যা করো।
১২. মেশিন ভাষা / নিম্নস্তরের ভাষা ও উচ্চস্তরের ভাষার মধ্যে পার্থক্য দেখাও।

### অনুবাদক প্রোগ্রাম- কম্পাইলার, অ্যাসেম্বলার, ইন্টারপ্রেটার

১৩. কম্পাইলার ও ইন্টারপ্রেটারের মধ্যে দুটি পার্থক্য লেখ।
১৪. উৎস কোডকে কম্পাইল করার প্রয়োজন হয় কেন? ব্যাখ্যা কর।
১৫. উচ্চতর ভাষায় লিখিত প্রোগ্রাম অনুবাদক প্রোগ্রাম প্রয়োজন কেন?
১৬. কম্পাইলার সুবিধাজনক কেন? ব্যাখ্যা কর। /দি.বো. ১৯/
১৭. ইন্টারপ্রেটারের তুলনায় কম্পাইলার সুবিধাজনক - ব্যাখ্যা কর। /চ.বো. ১৯/
১৮. অনুবাদক প্রোগ্রাম হিসেবে কম্পাইলার বেশি উপযোগী - ব্যাখ্যা কর। /কু.বো. ১৬/
১৯. প্রত্যেকবার প্রোগ্রাম নির্বাহের সময় কম্পাইল করা প্রয়োজন— ব্যাখ্যা কর।
২০. কম্পাইলার কীভাবে অনুবাদকের কাজ সম্পন্ন করে?

### প্রোগ্রামের সংগঠন

২১. ডকুমেন্টেশন কেন করতে হয়? /রা., কু., চ., ব.বো. ১৮/

### অ্যালগরিদম ও ফ্লোচার্ট

২২. অ্যালগরিদমের সুবিধাসমূহ লেখ।
২৩. প্রোগ্রাম কোডিং-এ অ্যালগরিদমের গুরুত্ব লেখ।
২৪. অ্যালগরিদম প্রোগ্রাম রচনার সহায়ক — ব্যাখ্যা কর।
২৫. অ্যালগরিদম কোডিং-এর পূর্বশর্ত— ব্যাখ্যা কর। /চ.বো. ১৬/
২৬. ফ্লোচার্ট হলো চিত্রভিত্তিক অ্যালগরিদম— ব্যাখ্যা কর।
২৭. অ্যালগরিদম ও ফ্লোচার্টের পার্থক্য লেখ।
২৮. সিস্টেম ফ্লোচার্টের চেয়ে প্রোগ্রাম ফ্লোচার্ট তৈরি করা বেশি প্রয়োজন হওয়ার কারণ ব্যাখ্যা কর।
২৯. সূডোকোডের চেয়ে ফ্লোচার্ট বেশি সর্বজনীন- ব্যাখ্যা কর।
৩০. সূডোকোড প্রোগ্রামিং ভাষা নির্ভর নয়-ব্যাখ্যা কর। /ব.বো. ১৯/

### ‘সি’ প্রোগ্রাম

৩১. সি প্রোগ্রামকে স্ট্রাকচার্ড প্রোগ্রামিং ভাষা বলা হয় কেন?
৩২. ‘সি’ একটি কেস সেনসিটিভ ভাষা? বুঝিয়ে লেখ। /সি.বো. ১৭; চা., দি., সি., য.বো. ১৮; চা.বো. ১৯/
৩৩. ‘সি’ কে মধ্যম স্তরের ভাষা বলা হয় কেন? /দি.বো. ১৬; চা.বো. ১৭/
৩৪. ‘সি’ কে স্ট্রাকচার্ড বা প্রোসিডিউর অরিয়েন্টেড প্রোগ্রামিং ল্যাঙ্গুয়েজ বলা হয় কেন? বুঝিয়ে লেখ।

৩৫. ‘সি’ কে একটি General Purpose প্রোগ্রামিং ভাষা বলা হয় কেন? বুঝিয়ে লেখ।

৩৬. সি-প্রোগ্রামিং-এ main() ফাংশন এর গুরুত্ব ব্যাখ্যা কর। /য.বো. ১৭/

৩৭. সি-প্রোগ্রামিং-এ #include<stdio.h> আবশ্যিক কেন? ব্যাখ্যা কর।

৩৮. সি ভাষার প্রোগ্রামে ব্যবহৃত ফাংশনের হেডার ফাইল উল্লেখ আবশ্যিক — ব্যাখ্যা কর।

৩৯. math.h ফাইলটি ব্যাখ্যা কর। /চ.বো. ১৯/

৪০. C ও C++ এর মধ্যে ভিন্নতা কী? ব্যাখ্যা কর। /য.বো. ১৯/

### ‘সি’ ভাষায় ব্যবহৃত ডেটা টাইপ

৪১. ইউজার ডিফাইন্ড ডেটা টাইপ বলতে কী বোঝ?
৪২. ডেটা টাইপ মডিফায়ার কী? বুঝিয়ে লেখ।
৪৩. Integer এর পরিবর্তে কখন Long Integer ব্যবহার করতে হয়?—বুঝিয়ে লিখ। /চ.বো. ১৭/

### ‘সি’ ভাষায় ব্যবহৃত চলক ও ধ্রুবক

৪৪. ভেরিয়েবল নামকরণের নিয়মাবলী লিখ।
৪৫. ‘সি’-ভাষায় চলকের নামকরণে কিছু নিয়মকানুন মেনে চলতে হয়।—ব্যাখ্যা কর। /কু.বো. ১৭/
৪৬. গ্লোবাল ভেরিয়েবল ও লোকাল ভেরিয়েবলের মধ্যে তুলনামূলক পার্থক্য লেখ।
৪৭. চলকের নামে আন্ডারস্কোর ব্যবহার করা যাবে-ব্যাখ্যা কর। /সি.বো. ১৯/
৪৮. ভেরিয়েবল নামকরণের সীমাবদ্ধতা কী?
৪৯. সি ভাষায় “1 number” সঠিক চলক নয়-ব্যাখ্যা কর। /চ.বো. ১৯/
৫০. চলক ও ধ্রুবক এর মধ্যে পার্থক্য লেখ।

### রাশিমালা ও কিওয়ার্ড

৫১. কখন ইউনারী অপারেটর ব্যবহার করা হয়? ব্যাখ্যা কর।
৫২. প্রোগ্রামে অপারেটরের গুরুত্ব লেখ।
৫৩. ++i এবং i++ এর মধ্যে পার্থক্য লেখ।
৫৪. কিওয়ার্ডকে ভেরিয়েবল হিসেবে ব্যবহার করা যায় না কেন?
৫৫. \n এবং \r এর ব্যবহারিক পার্থক্য ব্যাখ্যা কর।

### ইনপুট / আউটপুট স্টেটমেন্ট

৫৬. scanf(“%f”, &a) স্টেটমেন্টটি ব্যাখ্যা কর। /কু.বো. ১৯/
৫৭. scanf() ফাংশনে ‘&’ ব্যবহৃত হয় কেন?
৫৮. printf(“%d %x”, &a, &b); স্টেটমেন্টটি ব্যাখ্যা কর। /রা.বো. ১৭/
৫৯. আউটপুট ফাংশন বলতে কী বুঝায়? /ব.বো. ১৬/

### অ্যারে ও অ্যারের ব্যবহার

৬০. অ্যারে ও চলক এক নয়— ব্যাখ্যা কর।

### ফাংশন ও ফাংশনের ব্যবহার

৬১. ফাংশন নিজেই নিজেকে কল করে- ব্যাখ্যা কর।