

পঞ্চম অধ্যায়

প্রোগ্রামিং ভাষা

Programming Language



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।

প্রধান প্রধান শব্দ

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



এ অধ্যায়ের পাঠগুলো পড়ে যা যা শিখব

- প্রোগ্রামের ধারণা
- বিভিন্ন ভরের প্রোগ্রামিং ভাষা
- ব্যবহারিক : প্রোগ্রামের সংগঠন প্রদর্শন
- প্রোগ্রাম অ্যালগরিদম ও ফ্লোচার্ট প্রস্তুত করা
- ‘সি’ প্রোগ্রামিং ভাষা ব্যবহার করে প্রোগ্রাম প্রস্তুত করা

পাঠ পরিকল্পনা

- পাঠ-১ ও ২ : প্রোগ্রামের ধারণা ও প্রোগ্রামের ভাষা
- পাঠ-৩ ও ৪ : বিভিন্ন উচ্চস্তরের ভাষা সম্পর্কে আলোচনা
- পাঠ-৫ : চতুর্থ প্রজন্মের ভাষা বা 4GL
- পাঠ-৬ : প্রোগ্রামের সংগঠন ও প্রোগ্রাম তৈরির ধাপসমূহ
- পাঠ-৭-১০ : ব্যবহারিক: অ্যালগরিদম ও ফ্লোচার্ট
- পাঠ-১১ ও ১২ : প্রোগ্রাম ডিজাইন মডেল
- পাঠ-১৩ ও ১৪ : ‘সি’ প্রোগ্রাম
- পাঠ-১৫ ও ১৬ : ‘সি’ ভাষায় ব্যবহৃত ডেটা টাইপ
- পাঠ-১৭-১৯ : ‘সি’ ভাষায় ব্যবহৃত চলক ও ধূবক
- পাঠ-২০ : রাশিমালা ও কি-ওয়ার্ড
- পাঠ-২১-২৩ : ব্যবহারিক: ইনপুট / আউটপুট স্টেটমেন্ট
- পাঠ-২৪ : ব্যবহারিক: ব্যবহারিক নির্দেশাবলী ও কিছু প্রোগ্রাম প্রাকটিস
- পাঠ-২৫ ও ২৬ : ব্যবহারিক: কন্ট্রোল ও কন্ডিশনাল কন্ট্রোল স্টেটমেন্ট
- পাঠ-২৭ ও ২৮ : ব্যবহারিক: লুপ ও লুপের ব্যবহার
- পাঠ-২৯ ও ৩০ : ব্যবহারিক: অ্যারে ও অ্যারের ব্যবহার
- পাঠ-৩১ ও ৩২ : ব্যবহারিক: ফাংশন ও ফাংশনের ব্যবহার

পাঠ-১ ও ২

প্রোগ্রামের ধারণা (Concept of Program)

আজকের কম্পিউটার নানাবিধ কাজে ব্যবহৃত হলেও মূলত হিসাব নিকাশের যন্ত্র আবিষ্কারের ধারাবাহিকতার ফসল এটি। হিসাব নিকাশের জন্য বিভিন্ন পদ্ধতির ব্যবহার সেই প্রাচীনকাল থেকেই শুরু। হিসাব নিকাশের জন্য তখন নুড়ি পাথর, ঝিনুক, মার্বেল ইত্যাদি ব্যবহৃত হত। পরবর্তীতে খ্রিষ্টপূর্ব ২৫০০ অন্দে শুরু হয় এক ধরনের যন্ত্রের ব্যবহার যা অ্যাবাকাস নামে পরিচিত। অ্যাবাকাস হচ্ছে কাঠের তৈরি আয়তাকার কাঠামো। এতে কাঠের ফ্রেমের ভিতর আড়াআড়িভাবে সুতা বেঁধে বিভিন্ন রংয়ের গোলাকার বল সাজিয়ে রাখা হতো এবং এর সাহায্যে বিশেষ পদ্ধতিতে যোগ, বিয়োগ, গুণ, ভাগ ইত্যাদি গাণিতিক অপারেশন করা যেত। পরবর্তী বিভিন্ন সময়ে হিসাব নিকাশকে সহজ করার জন্য নানাবিধ যন্ত্র আবিষ্কৃত হতে থাকে। তবে দীর্ঘকাল ধরেই বিজ্ঞানীদের চেষ্টা ছিল যেন যন্ত্রের বাইরে থেকে নির্দেশ দিয়ে যন্ত্রকে পরিচালনা করা যায়। এই চেষ্টার অংশ হিসেবে ১৮২২ সালে ইংল্যান্ডের প্রকৌশলী ও গণিতবিদ চার্লস ব্যাবেজ (Charles Babbage) লগারিদমসহ গাণিতিক হিসাব নিকাশের কাজ অধিক সহজ করার জন্য ব্রিটিশ সরকারের সহায়তায় ডিফারেন্স ইঞ্জিন (Difference Engine) নামে একটি যন্ত্র আবিষ্কার করেন। এ যন্ত্রটি থেকে আশানুরূপ সাফল্য না পেয়ে তার প্রায় একযুগ পর ১৮৩৩ সালে নিজ উদ্যোগে অ্যানালাইটিক্যাল ইঞ্জিন (Analytical Engine) নামে অপর একটি যন্ত্র তৈরি করেন যা দুটি অংশে বিভক্ত ছিল। একটি অংশ ছিল প্রক্রিয়াকরণের জন্য, যা মিল (Mill) নামে পরিচিত ছিলো এবং অপর অংশ স্টোর (Store) নামে পরিচিত ছিল যা ডেটা সংরক্ষণ করতে পারত।

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



চিত্র : অ্যাডা লাভল্যাস

একটি আদর্শ প্রোগ্রামের বৈশিষ্ট্য

ভাষা জানা থাকলেও সবাই যেমন সাহিত্য রচনা করতে পারে না, তেমনি প্রোগ্রামিং ভাষা জানা থাকলেও ভালো প্রোগ্রাম রচনা করা সহজ নয়। প্রোগ্রামিং এর শুরুটা গাণিতিক মনে হলেও এর সাথে শিল্পরস জড়িত থাকে। আদর্শ প্রোগ্রাম বলতে যে প্রোগ্রামে কম্পিউটার প্রোগ্রামের যাবতীয় বৈশিষ্ট্য বা গুণাবলী বর্তমান সে ধরনের প্রোগ্রামকে বুঝায়। একটি আদর্শ প্রোগ্রামে সাধারণত পাঁচটি পর্ব থাকে।

যথা-১. পরিচয় পর্ব; ২. বর্ণনা; ৩. ইনপুট; ৪. প্রসেস; ৫. আউটপুট।

একটি প্রোগ্রাম আদর্শ হতে হলে তাকে নিম্নলিখিত গুণাবলীর অধিকারী হতে হয়।

- প্রোগ্রামের এলগরিদম সরলভাবে প্রণয়ন করা যেন প্রোগ্রামের ধাপগুলো সহজেই বুঝা যায়।
- প্রোগ্রামের প্রবাহচিত্র স্পষ্টভাবে উপস্থাপন করা যাতে প্রোগ্রাম নির্বাহের ধারা বুঝা যায়।
- প্রোগ্রামের শুরুতে প্রোগ্রামের উদ্দেশ্য, প্রোগ্রামারের নাম, ধূবক, চলক ইত্যাদির পরিচয় রাখা। এতে পরবর্তী প্রোগ্রামার সহজেই প্রোগ্রামের উদ্দেশ্য ও কাজ সম্পর্কে ধারণা পেতে পারে।
- চলক হিসাবে প্রতিনিধিত্বমূলক বা অর্থপূর্ণ শব্দ ব্যবহার করা যাতে চলকের উদ্দেশ্য বুঝাতে অসুবিধা না হয়।
- প্রোগ্রামকে অকারণে দীর্ঘ না করা।
- নির্দিষ্ট সমস্যার জন্য প্রয়োজনে উপযুক্ত প্রোগ্রামিং ভাষা নির্বাচন করা।
- প্রয়োজনের অতিরিক্ত চক্র বাদ দিতে হবে।

- ভবিষ্যৎ-এ প্রয়োজনে প্রোগ্রামের মধ্যে পরিবর্তন করা সহজ হবে।
- গুরুত্বপূর্ণ অংশ বা যে কোন ফাংশন লিখলে তার সাথে মন্তব্য দিতে হবে।
- তথ্য প্রদানের ব্যবস্থা থাকতে হবে।
- তথ্য প্রক্রিয়াকরণের ব্যবস্থা থাকতে হবে।
- অবশ্যই ফলাফল প্রাপ্তির সুবিধা থাকতে হবে।

প্রোগ্রামের ভাষা (Programming Language)

মনের ভাব প্রকাশের মাধ্যম হচ্ছে ভাষা। আবার বিভিন্ন জাতি ও দেশভেদে ভাষার ভিন্নতা আছে। প্রত্যেক ভাষার নিজস্ব বর্ণ ও ব্যাকরণ রয়েছে। মানুষের ভাষা কম্পিউটার বুঝতে পারে না, কম্পিউটার বোঝে কম্পিউটারের নিজস্ব ভাষা। আর আমরা জানি 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। মানুষের মধ্যে যেমন ভাষার ভিন্নতা আছে, তেমনি কম্পিউটারেও অনেক ধরনের ভাষা ব্যবহৃত হয়। তবে সে 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 কি ধরনের কাজ করবে, ডেটা কোথায় রাখবে, ফলাফল কি হবে ইত্যাদি নির্ধারণ করা। কম্পিউটারের ভাষা শুধু কম্পিউটারই বুঝে। প্রকৃতপক্ষে On (Yes) এবং Off (No) ছাড়া কম্পিউটার আর কিছু বোঝে না। একেই ০ এবং ১ দিয়ে বোঝানো হয় যাকে বলা হয় বাইনারি পদ্ধতি। এই পদ্ধতিতে ব্যবহৃত ০ (শূন্য) এবং ১ (এক) এদুটি সংখ্যাকে কম্পিউটারের ভাষায় বিট বলা হয়। এদেরকে ইলেকট্রনিক সার্কিটে দু'টি ভোল্টেজ লেভেল দ্বারা নির্দিষ্ট করা যায়।

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

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

১. মেশিন ভাষা বা যান্ত্রিক ভাষা (Machine Language)

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

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

- সুবিধা:**
১. মেশিন ভাষার সবচেয়ে বড় সুবিধা হচ্ছে সরাসরি কম্পিউটারের সাথে যোগাযোগ করা যায়।
 ২. মেশিন ভাষায় লেখা প্রোগ্রাম নির্বাহের জন্য কোনো প্রকার অনুবাদক প্রোগ্রামের প্রয়োজন হয় না। ফলে দুটি কাজ করে।
 ৩. মেশিন ভাষায় লিখিত প্রোগ্রামে অতি অল্প মেমোরি প্রয়োজন হয়।
 ৪. কম্পিউটারের ভেতরের 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। ভুল হলে তা বের করা এবং ভুল-ত্রুটি দূর করা খুব কঠিন।
৪. এ ভাষার সবচেয়ে বড় অসুবিধা হচ্ছে এক ধরনের কম্পিউটারের জন্য লিখিত প্রোগ্রাম অন্য ধরনের কম্পিউটারে ব্যবহার করা যায় না। মেশিন ভাষাকে নিম্নস্তরের ভাষাও বলা হয়।

২. অ্যাসেম্বলি ভাষা (Assembly language)

অ্যাসেম্বলি ভাষাকে সাংকেতিক (Symbolic) ভাষাও বলা হয়। এর প্রচলন শুরু হয় ১৯৫০ সাল থেকে। দ্বিতীয় প্রজন্মের কম্পিউটারে এই ভাষা ব্যাপকভাবে প্রচলিত ছিল। 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 এক হতে দুটি অ্যালফানিউমেরিক বর্ণ থাকে, এই বর্ণের মধ্যে কোন ফাঁক থাকে না। নির্দেশ নেমোনিক (যেমন LDA) ও রেজিস্টারের নাম লেবেল হিসেবে ব্যবহার করা যায় না। লেবেলের প্রথম বর্ণ সর্বদা কোন অক্ষরের হবে।

অপকোড: এতে নির্দেশ নেমোনিক থাকে। এ নেমোনিকগুলো বিভিন্ন কম্পিউটারে বিভিন্ন হতে পারে, তবে সাধারণত নিচের মত হয়।

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

অপারেন্ড: অপকোড যার উপর কাজ করে তাকে অপারেন্ড বলে। অপারেন্ডের অবস্থানের ঠিকানা বুকানোর জন্য এখানে সাধারণত অ্যালফানিউমেরিক 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 যাতে ভবিষ্যতে প্রোগ্রামার বা অন্য কেউ প্রোগ্রামের সঠিক অর্থ সহজে বুঝতে পারে। প্রোগ্রামের নিজের সুবিধার জন্য ব্যবহৃত হয়। অপারেন্ট ফিল্ডের পর কোলন (:) বা সেমিকোলন (;) দিয়ে মন্তব্য লেখা যায়।

উদাহরণ: A ও B যোগ করে C অবস্থানে রাখ। 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 হল।

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

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 ভাষার প্রোগ্রামকে মেশিন ভাষায় রূপান্তর করা হয় তাকে অ্যাসেম্বলার বলে। নিম্নের চিত্রে অ্যাসেম্বলি ভাষায় প্রোগ্রাম নির্বাহ প্রক্রিয়া দেখানো হল—



অ্যাসেম্বলি ভাষার সুবিধা:

- অ্যাসেম্বলি ভাষায় প্রোগ্রাম রচনা করা যান্ত্রিক ভাষার তুলনায় অনেক সহজ।
- প্রোগ্রাম রচনা করতে কম সময় লাগে।
- ভুল ত্রুটি বের করা কষ্টসাধ্য ব্যাপার।
- অনুবাদক প্রোগ্রামের প্রয়োজন হয়।
- প্রোগ্রাম পরিবর্তন করা সহজ।

অ্যাসেম্বলি ভাষার অসুবিধা:

- প্রোগ্রাম রচনার সময় প্রোগ্রামারকে মেশিন সম্পর্কে ধারণা থাকতে হয়।
- ভিন্ন ভিন্ন মেশিনে ভিন্ন ভিন্ন অ্যাসেম্বলি ভাষা ব্যবহার করতে হয়।
- ইহা যন্ত্র নির্ভর ভাষা।

যান্ত্রিক ভাষা ও অ্যাসেম্বলি ভাষার মধ্যে পার্থক্য:

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



কাজ: মেশিন ভাষা ও অ্যাসেম্বলি ভাষা ও উচ্চস্তরের ভাষার তুলনা একটি ছকের মাধ্যমে দেখাও।

পাঠ-৩ ও ৪

বিভিন্ন উচ্চস্তরের ভাষা সম্পর্কে আলোচনা (Description about Different Types of High Level Languages)

উচ্চতর ভাষা (High level language)

উচ্চতর ভাষা বা হাই লেভেল ভাষার সাথে মানুষের ভাষার (যেমন: ইংরেজি) মিল আছে। এই স্তরের ভাষায় লিখিত প্রোগ্রাম বিভিন্ন ধরনের মেশিনে ব্যবহার করা সম্ভব। অর্থাৎ, এই প্রোগ্রাম ভাষা কম্পিউটার সংগঠনের নিয়ন্ত্রণের উদ্দেশ্যে, এই জন্য এসব ভাষাকে উচ্চতর ভাষা বলা হয়। এটি মানুষের জন্য বুঝতে পারা খুব সহজ কিন্তু কম্পিউটার সরাসরি বুঝতে পারে না বলে অনুবাদক প্রোগ্রামের সাহায্যে একে মেশিন ভাষায় রূপান্তরিত করে নিতে হয়। উদাহরণ: Qbasic, Pascal, C/C++, JAVA ইত্যাদি।

উচ্চস্তরের ভাষার প্রকারভেদ (Classifications of High Level Language)

বিভিন্ন ভাষার প্রয়োগ ক্ষেত্র বিভিন্ন। প্রয়োগের ভিত্তিতে উচ্চস্তরের ভাষাকে সাধারণত নিম্নলিখিত দুই 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 কাজের উপযোগী করে তৈরি করা হয় তা বিশেষ কাজের ভাষা নামে পরিচিত। যেমন— COBOL, ALGOL, FORTRAN ইত্যাদি।

উচ্চস্তরের ভাষার ব্যবহার:

- বড় প্রোগ্রাম তৈরির কাজে।
- বৃহৎ ডেটা প্রসেসিং এর কাজে ব্যবহৃত প্রোগ্রাম তৈরি করতে।
- যেসব ক্ষেত্রে প্রচুর মেমরির প্রয়োজন সেসব ক্ষেত্রের সফটওয়্যার তৈরির কাজে।
- জাটিল গাণিতিক হিসাব নিকাশের সফটওয়্যার তৈরির কাজে।
- এ্যাপ্লিকেশন প্যাকেজ সফটওয়্যার তৈরির কাজে।
- বিভিন্ন ধরনের অটোমেটিক প্রসেস কন্ট্রোলের কাজে।

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

২. এতে ভুল হ্বার সম্ভবনা কম থাকে এবং প্রোগ্রামের ত্রুটি বের করে তা সংশোধন করা সহজ।

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

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

অসুবিধা: ১. উচ্চস্তরের ভাষার অসুবিধা হচ্ছে 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:

সি (C): ১৯৭০ সালে আমেরিকার বেল ল্যাবরেটরির গবেষক ডেনিস রিচি C ভাষা উদ্ভাবন

করেন। সিস্টেম প্রোগ্রাম তৈরির ক্ষেত্রে এ ভাষা বেশ জনপ্রিয়। C ভাষাকে কম্পিউটার

ভাষার জনক বলা হয়। এ ভাষার অনেক সংস্করণ রয়েছে। যেমন: C, ANSI C, Turbo

C, Visual C ইত্যাদি। পরবর্তী পাঠে আমরা C ভাষা নিয়ে বিস্তারিত আলোচনা করব।



সি++ (C++): C ভাষায় কিছু নতুন সুবিধা সংযোজন করায় এর পরবর্তী সংস্করণ যা দাঁড়ায় তা হচ্ছে C++ ভাষা। ১৯৮৩ সালে বেল ল্যাবরেটরিই আরেক ডেনিস গবেষক বিয়ান স্ট্রুটস্ট্রুপ (Bjarne Stroustrup) C++ ভাষার সূচনা 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। ১৯৮৫ সাল থেকে এর বাণিজ্যিক ব্যবহার শুরু হয়।



ভিজুয়্যাল বেসিক (Visual Basic): বিশ্বখ্যাত সফটওয়্যার নির্মাতা প্রতিষ্ঠান মাইক্রোসফট ১৯৯১ সালে কাস্টমার অ্যাপ্লিকেশন সফটওয়্যার তৈরির জন্য উইন্ডোজ অপারেটিং সিস্টেমের জন্য উপযোগী ভিজুয়্যাল বেসিক নির্মাণ করে। এটি মূলত বেসিক প্রোগ্রামিং ভাষার “গ্রাফিক্যাল ইউজার ইন্টারফেস” ভাসন। এটি একটি ভিজুয়্যাল প্রোগ্রামিং ভাষা। এতে ইন্ডেক্স ড্রাইভেন ফিচার সংযোজন করা হয়, ফলে যে কেউ ইচ্ছা করলে ভিজুয়্যাল বেসিক ব্যবহার করে অল্প সময়ে কাস্টমাইজড অ্যাপ্লিকেশন সফটওয়্যার তৈরি করতে পারে। পরবর্তীতে এর অনেক ভাসন তৈরি হয়। বর্তমানে VB.Net হচ্ছে এর সমসাময়িক ভাসন।



জাভা (Java): সান মাইক্রোসিস্টেমের তৈরি অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং ভাষা হচ্ছে জাভা। এটি মূলত 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 যা 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।



ওরাকল (Oracle): ডেটাবেজ সংক্রান্ত সফটওয়্যার তৈরির সবচেয়ে বড় প্রতিষ্ঠান ওরাকল কর্পোরেশন। এর তৈরি একটি RDBMS হচ্ছে ওরাকল। ১৯৭৭ সালে এড ওয়াটস ও বব মাইনারকে সাথে নিয়ে এটি তৈরি করেন ল্যারি এলিসন যারা বর্তমানে ওরাকল কর্পোরেশনের কর্ণধার। সিকিউরিটির দিক দিয়ে সবচেয়ে শক্তিশালী এবং সবচেয়ে বেশি ডেটা ধারণক্ষম RDBMS হচ্ছে ওরাকল।



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



ফোরট্রান (Fortran): উচ্চস্তরের ভাষার মধ্যে সবচেয়ে পুরোনো হচ্ছে ফোরট্রান (FORmula TRANslator) ভাষার। ১৯৫৩ সালে আইবিএম (IBM) এর গবেষক জন ব্যাকাস এটি তৈরি করেন। গাণিতিক জটিল হিসাব- নিকাশের সুবিধার জন্য মূলত এটি তৈরি করা হয়েছিল। এটি প্রকৌশল সংক্রান্ত গবেষণার কাজে এখনো বেশ জনপ্রিয়।



পাইথন (Python): পাইথন একটি মাল্টিপ্যারাডিজিম প্রোগ্রামিং ভাষা যা একই সাথে অবজেক্ট ওরিয়েন্টেড ফিচার ও স্ট্রাকচার্ট ফিচার সাপোর্ট করে। এটি একটি ডাইনামিক প্রোগ্রামিং ভাষা যেখানে প্রোগ্রাম রান করার পর চলক এবং কোনো মেথডকে স্বয়ংক্রিয়ভাবে চিনে নেয়। ১৯৮৯ সালে নেদারল্যান্ড-এর কম্পিউটার বিজ্ঞানী গুইডো ভ্যান রোসাম এই প্রোগ্রামিং ভাষা তৈরি করেন।



অ্যাসেম্বলি ভাষা ও উচ্চস্তরের ভাষার মধ্যে পার্থক্য

অ্যাসেম্বলি ভাষা	উচ্চস্তরের ভাষা
১. সংক্ষিপ্ত সাংকেতিক চিহ্ন বা সহায়ক নাম 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 লেখা প্রোগ্রাম যে কোন 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 অর্থাৎ এটি মেশিন নির্ভর।	১. কম্পিউটার এ ভাষা সরাসরি বুঝতে পারে না অর্থাৎ এটি মেশিন নির্ভর নয়।
২. এক মডেলের মেশিনের জন্য লিখিত প্রোগ্রাম অন্য কোনো মডেলের মেশিন বুঝতে পারে না।	২. যে কোন মডেলের জন্য লিখিত প্রোগ্রাম অন্য মডেলের মেশিন বুঝতে পারে।
৩. মেশিন ভাষার প্রোগ্রামারকে অনেক দক্ষ হতে হয়। বিশেষ করে কম্পিউটারের লজিক্যাল গঠন সম্পর্কে ভাল জ্ঞান থাকতে হয়।	৩. উচ্চস্তরের ভাষার প্রোগ্রামারকে পুরোপুরি দক্ষ না হলেও চলে। বিশেষ করে কম্পিউটারের লজিক্যাল গঠন সম্পর্কে জ্ঞান না থাকলেও হয়।
৪. মেশিন ভাষায় প্রোগ্রাম লেখা খুবই কঠিন ও সময় সাপেক্ষ।	৪. উচ্চস্তরের ভাষায় প্রোগ্রাম লেখা সহজ এবং কম সময় প্রয়োজন হয়।
৫. সরাসরি বাইনারি ভাষায় প্রোগ্রাম লেখা হয়ে থাকে। ফলে অনুবাদের প্রয়োজন হয় না।	৫. মানুষের বোধগম্য ভাষায় প্রোগ্রাম লেখা হয়ে থাকে। তবে অনুবাদক প্রোগ্রাম দ্বারা বাইনারি ভাষায় রূপান্তর করে নিতে হয়।
৬. কম পরিমাণ লজিক ও মেমরিতে এ ভাষায় লিখিত প্রোগ্রাম নির্বাচ করা যায়।	৬. এ ভাষায় লিখিত প্রোগ্রাম নির্বাচ করতে বেশি পরিমাণ লজিক ও মেমরি প্রয়োজন হয়।



কাজ: উপরোক্ষিত প্রোগ্রামিং ভাষাগুলোর মধ্যে বৈশিষ্ট্যগত পার্থক্য নিবৃত্ত করো।

পাঠ-৫

চতুর্থ প্রজন্মের ভাষা বা 4GL (4th Generation Language)

চতুর্থ প্রজন্মের ভাষা বা 4th Generation Language কে সংক্ষেপে 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, Visual Basic, Oracle ইত্যাদি।

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।

পঞ্চম প্রজন্মের ভাষা: 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।

অনুবাদক প্রোগ্রাম (Translator Program)

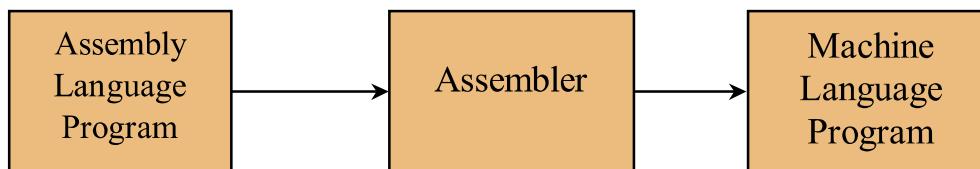
মেশিন ভাষায় লেখা প্রোগ্রামকে বলা হয় বস্তু প্রোগ্রাম (Object Program) এবং অন্য যেকোনো ভাষায় লেখা প্রোগ্রামকে বলা হয় উৎস প্রোগ্রাম (Source program)। যে প্রোগ্রামের সাহায্যে উৎস (Source) প্রোগ্রামকে বস্তু (Object) প্রোগ্রামে পরিণত করা হয় তাকে অনুবাদক প্রোগ্রাম বলে।



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/সফটওয়্যার তিনি প্রকার:

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

১. **অ্যাসেম্বলার (Assembler):** অ্যাসেম্বলি ভাষায় লিখিত প্রোগ্রামকে মেশিন ভাষায় অনুবাদ করার জন্য অ্যাসেম্বলার ব্যবহার করা হয়। এটি অ্যাসেম্বলি ভাষায় লিখিত প্রোগ্রামকে কোড বা যান্ত্রিক ভাষায় রূপান্তর করে অর্থাৎ, নেমোনিক কোডকে মেশিন ভাষায় অনুবাদ করে। প্রোগ্রামে কোনো ভুল থাকলে Error Message দেয়।



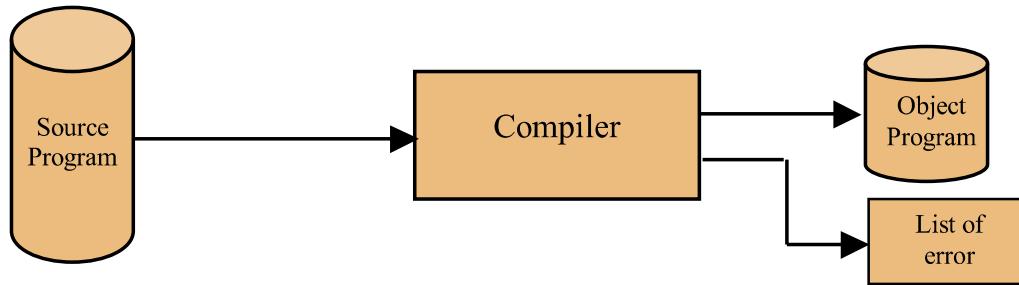
অ্যাসেম্বলারের প্রধান কাজসমূহ:

- ক. নেমোনিক কোডকে মেশিন ভাষায় অনুবাদ করা।
- খ. সাংকেতিক ঠিকানাকে মেশিন ভাষায় লেখার ঠিকানায় রূপান্তর করা।

গ. প্রোগ্রামে কোনো ভুল থাকলে Error Message দেওয়া।

ঘ. সব নির্দেশ ও ঠিকানা প্রধান মেমোরিতে রাখা।

- ২. কম্পাইলার (Compiler):** কম্পাইলার হলো এক ধরনের অনুবাদক যা হাইলেভেল ভাষায় লিখিত প্রোগ্রামকে মেশিন ভাষায় রূপান্তর করে। অর্থাৎ সোর্স প্রোগ্রামকে অবজেক্ট প্রোগ্রামে রূপান্তর করে। 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 উপাত্ত বা ডেটার ভিত্তিতে অবজেক্ট প্রোগ্রামকে নির্বাচ করানো হয় ফলাফল প্রদর্শনের জন্য। কম্পিউটার দিয়ে অনুবাদক প্রক্রিয়া চিত্রের মাধ্যমে দেখানো হলো।



চিত্র: কম্পাইলারের কাজ

কম্পাইলারের কাজ:

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

কম্পাইলারের সুবিধা:

- কম্পাইলার সম্পূর্ণ প্রোগ্রামটিকে একসাথে অনুবাদ করে ফলে প্রোগ্রাম নির্বাহের গতি দৃঢ় হয়।
- প্রোগ্রাম নির্বাহে কম সময় লাগে।
- কম্পাইলারের মাধ্যমে রূপান্তরিত প্রোগ্রাম সম্পূর্ণরূপে মেশিন প্রোগ্রামে রূপান্তরিত হয়।
- একবার প্রোগ্রাম কম্পাইল করা হলে পরবর্তীতে আর কম্পাইলের প্রয়োজন হয় না।
- প্রোগ্রামে কোন 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 মেমরিতে বেশি জায়গা লাগে।
- প্রোগ্রাম ডিবাগিং ও টেস্টিং এর কাজ ধীরগতি সম্পন্ন।

- ৩. ইন্টারপ্রেটার (Interpreter) :** ইন্টারপ্রেটারও কম্পাইলারের মতো হাইলেভেল ভাষাকে মেশিন ভাষায় রূপান্তর করে। তবে 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 নির্বাচ 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। তৃতীয় লাইনে প্রবেশ করার পর কম্পিউটার স্ক্রিনে B এর মধ্যে 8 প্রদর্শন করবে। এভাবে ইন্টারপ্রেটার প্রোগ্রামের লাইন অনুসারে প্রোগ্রামকে নির্বাহ করে।



ইন্টারপ্রেটারের সুবিধা:

- এটি ব্যবহারে প্রোগ্রামের ভুল সংশোধন করা এবং পরিবর্তন করা সহজ হয়।
- Interpreter Program আকারে ছোট হয় এবং মেমরি 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 ছোটে 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 : অ্যাসেম্বলার, কম্পাইলার ও ইন্টারপ্রেটারের কার্যবলীর তুলনামূলক ছক আঁক।

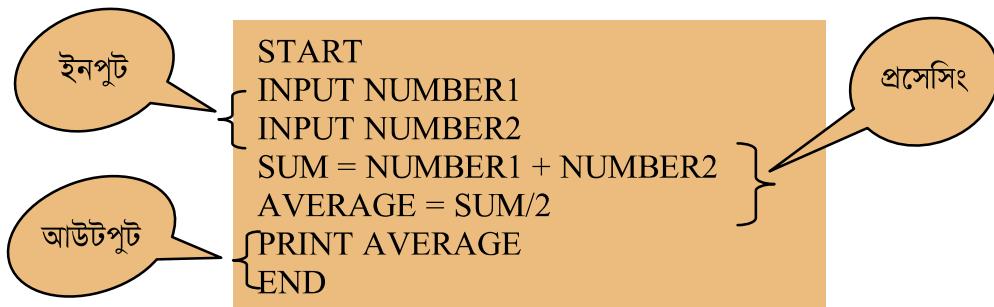
পাঠ-৬

প্রোগ্রামের সংগঠন (Program Organization)

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 কম্পিউটারকে যেসব ডেটা ও ইনস্ট্রুকশন দেওয়া হয় তা হলো ইনপুট। প্রদত্ত ডেটাকে ইনস্ট্রুকশন অনুযায়ী প্রক্রিয়াকরণ করা হয় যা হলো প্রসেসিং। প্রসেসিং থেকে প্রাপ্ত ফলাফল হলো আউটপুট। একটি উদাহরণের মাধ্যমে নিচে প্রোগ্রামের সংগঠন দেখানো হলো:



প্রোগ্রাম তৈরির ধাপ (Steps of Developing a Program)

প্রোগ্রাম সম্পর্কে আগের পাঠে আমরা জেনেছি। যেকোনো প্রোগ্রাম সুষ্ঠুভাবে তৈরি করতে কতকগুলো ধাপ অনুসরণ করতে হয়। এ ধাপগুলো অনুসরণ করে প্রোগ্রাম তৈরি করলে পরবর্তীতে কোনো সমস্যায় পড়তে হবে না। ধাপগুলো নিম্নরূপ-

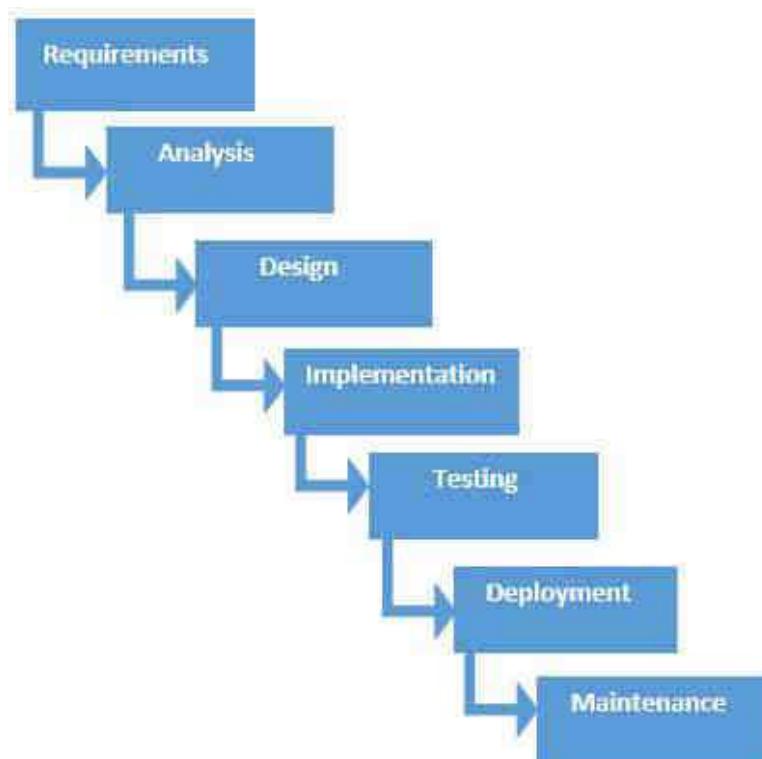
১. সমস্যা নির্দিষ্টকরণ (Problem specification)
 ২. সমস্যা বিশ্লেষণ (Problem Analysis)
 ৩. প্রোগ্রাম ডিজাইন (Program Design)
 ৪. প্রোগ্রাম উন্নয়ন (Program Development)
 ৫. প্রোগ্রাম বাস্তবায়ন (Program Implementation)
 ৬. ডকুমেন্টেশন (Documentation)
 ৭. প্রোগ্রাম রক্ষণাবেক্ষণ (Maintenance)।
- ১. সমস্যা নির্দিষ্টকরণ (Problem Specification):** সমস্যা সমাধানের পূর্বে তা অবশ্যই ভালোভাবে চিহ্নিত করতে হবে। সমস্যায় কোন ধরনের ইনপুট হবে এবং কোন ধরনের আউটপুট প্রয়োজন সে বিষয়ে সিদ্ধান্ত নেওয়া হবে। সমস্যা সমাধানের জন্য যে সকল তথ্য প্রয়োজন তা বিভিন্ন পদ্ধতি অবলম্বন করে সংগ্রহ করতে হবে। অর্থাৎ তথ্যানুসন্ধানের কাজটিও এ ধাপে সম্পন্ন করতে হবে।
- ২. সমস্যা বিশ্লেষণ (Problem Analysis):** সমস্যা নির্দিষ্টকরণের পর সমস্যাটিকে ক্ষুদ্র ক্ষুদ্র অংশে ভাগ করতে হবে। এই ধাপে নিম্নলিখিত বিষয়গুলোর ওপর গুরুত্ব দিতে হবে-
- ক. সমস্যার গাণিতিক মডেল তৈরি।
 - খ. সমস্যার সমাধানে কত সময় লাগবে তা নিরূপণ।

সমস্যা বিশ্লেষণের ক্ষেত্রে বিভিন্ন টুলস বা পদ্ধতি ব্যবহার করা হয়। যেমন— ডেটা ফ্লো ডায়াগ্রাম (DFD) ডিসিশন ট্রি, ডিসিশন টেবিল, স্টাকচার্ড ইংলিশ ইত্যাদি।

৩. প্রোগ্রাম ডিজাইন (Program Design): প্রোগ্রাম বিশ্লেষণ ধাপে যে ছোটো ছোটো ভাগগুলো করা হয়েছে তাদের পারস্পরিক সম্পর্ক ও সামগ্রিক সমাধান বের করতে হবে। 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 হবে।



চিত্র: প্রোগ্রাম তৈরির ধাপসমূহ

৪. প্রোগ্রাম উন্নয়ন (Program development): অ্যালগরিদম বা ফ্লোচার্টকে কোনো একটি উচ্চস্তরের প্রোগ্রামিং ভাষায় লিখতে হবে। একে বলা হয় কোডিং করা। যেমন: C/Pascal/QBasic ইত্যাদি ভাষায় কোডিং করা যেতে পারে।

৫. প্রোগ্রাম বাস্তবায়ন (Program implementation): হার্ডওয়্যার ক্রয় হতে শুরু করে সফটওয়্যার ইনস্টল এসব কাজ বাস্তবায়নের মধ্যে পড়ে। বাস্তবায়ন অংশের দুটি গুরুত্বপূর্ণ কাজ হচ্ছে:

১. টেস্টিং: ভুল-ভুটি পরীক্ষা করা ও ২. ডিবাগিং: ভুল সংশোধন করা।

ক. টেস্টিং: প্রোগ্রাম টেস্টিং হচ্ছে কোনো প্রোগ্রাম 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 এই ধাপে। এক্ষেত্রে যদি কোন অসঙ্গতি পাওয়া যায় তবে বুঝতে হবে প্রোগ্রাম কোডিংয়ের কোথাও ভুল হয়েছে। প্রোগ্রামে সাধারণত তিনি ধরনের ভুল পরিলক্ষিত হয়। যথা:

১. ব্যাকরণগত ভুল (Syntex Error)
২. যৌক্তিক ভুল (Logical Error)
৩. নির্বাহজনিত ভুল (Execution Error or Runtime Error)

১. **ব্যাকরণগত ভুল (Syntex Error):** যে ভাষায় প্রোগ্রাম লেখা হবে সেই ভাষার নিজস্ব কতগুলো নিয়ম থাকে। নিয়মবিহীন কোনো কোডিং হয়ে থাকলে তাকে ব্যাকরণগত ভুল হিসেবে বিবেচনা করা হয়। যেমন: C প্রোগ্রামিং ভাষায় কোন স্টেটমেন্টের পর সেমিকোলন (;) দিতে হয়। এক্ষেত্রে কোনো স্টেটমেন্টের পর সেমিকোলন (;) না দিলে তা হবে C প্রোগ্রামিং ভাষার ক্ষেত্রে ব্যাকরণগত ভুল। ব্যাকরণগত ভুল থাকলে কম্পিউটার Error Message দিবে এবং প্রোগ্রামের কোথায় কি ভুল হয়েছে তা জানিয়ে দিবে। ভুল সংশোধন করা সাপেক্ষে কম্পিউটার প্রোগ্রাম নির্বাহ করবে।
২. **যৌক্তিক ভুল (Logical Error):** প্রোগ্রামে কোনো লজিক লিখতে 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 ইনপুট দিতে ভুল হলে অর্থাৎ ভুল ডেটা ইনপুট দিলে আউটপুট বা ফলাফল ভুল আসবে। এধরনের ভুলকে নির্বাহজনিত ভুল বলা হয়।
৪. **ডিবাগিং:** আমরা উপরে প্রোগ্রাম টেস্টিং এর ক্ষেত্রে তিন ধরনের ভুল সম্পর্কে জেনেছি। প্রোগ্রামে যে কোনো ভুল চিহ্নিত করতে পারলে তাকে বলা হয় বাগ (Bug)। উক্ত বাগকে সমাধান করাকে বলা হয় ডিবাগ (Debug)। এক্ষেত্রে Syntex Error সমাধান করা সহজ। কিন্তু Logical Error সমাধান করা তুলনামূলক জটিল।
৫. **ডকুমেন্টেশন (Documentation):** প্রোগ্রাম ডেভেলপমেন্টের সময় ভবিষ্যতের কথা ভেবে বিভিন্ন বিষয় লিপিবদ্ধ করে রাখতে হয়। এই 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
৬. **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।

জেনে রাখো: প্রোগ্রাম থেকে ভুল-ত্রুটি খুঁজে বের করে তা সমাধান করাকে ডিবাগিং (debugging) বলা হয়।

‘Bug’ অর্থ পোকা। ডিবাগিং অর্থ পোকা দূর করা। ১৯৪৫ সালে মার্ক-১ কম্পিউটারে একটি মথ পোকা ঢুকে যাওয়ায় তা অচল হয়ে পড়ে। তখন থেকে ডিবাগিং শব্দটির উৎপত্তি। ডিবাগিং করার জন্য প্রোগ্রামটিকে প্রথমে শুরু থেকে শেষ পর্যন্ত ভালোভাবে পরীক্ষা করা হয়। প্রয়োজন হলে প্রোগ্রাম বিশেষজ্ঞের সাহায্য নেওয়া যেতে পারে।

কাজ : বাস্তবজীবনে প্রোগ্রাম তৈরির ধাপগুলোর প্রয়োগ দেখাও।



পাঠ-৭, ৮, ৯ ও ১০

ব্যবহারিক: অ্যালগরিদম ও ফ্লোচার্ট (Algorithm and Flow Chart)

অ্যালগরিদম (Algorithm): কোনো একটি নির্দিষ্ট সমস্যা সমাধানের জন্য যুক্তিসম্মত ও ধাপে ধাপে সমাধান করার যে পদ্ধতি, তাকে অ্যালগরিদম বলা হয়। কোনো সমস্যাকে কম্পিউটার প্রোগ্রামিং দ্বারা সমাধান করার পূর্বে কাগজে-কলমে সমাধান করার জন্যই অ্যালগরিদম ব্যবহার করা হয়। অবশ্যই নির্দিষ্ট সংখ্যক ধাপে সমস্যার সমাধান দেখাতে হবে। অ্যালগরিদমের শূন্য বা এক/একাধিক ইনপুট থাকতে হবে। অবশ্যই কমপক্ষে একটি আউটপুট থাকতে হবে। একে ব্যাপকভাবে প্রয়োগ করা সম্ভব।

অ্যালগরিদম তৈরির শর্ত বা নিয়ম-

১. অ্যালগরিদমটি সহজবোধ্য হতে হবে।
২. প্রত্যেকটি ধাপ স্পষ্ট হতে হবে যাতে সহজে বোঝা যায়।
৩. সঙ্গে সঙ্গে 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 পরিবর্ধনে সহায়তা করে।

ফ্লোচার্ট বা প্রবাহ চিত্র: যে চিত্রিভূতিক পদ্ধতিতে বিশেষ কতকগুলো চিহ্নের সাহায্যে কোনো একটি নির্দিষ্ট সমস্যার সমাধান করা হয় তাকে ফ্লোচার্ট বলা হয়। ফ্লোচার্টের সাহায্যে প্রোগ্রাম বোঝা সহজ হয় বলে এটি প্রোগ্রামার ও ব্যবহারকারীর মাঝে সংযোগ রক্ষার জন্য ব্যবহৃত হয়।

ফ্লোচার্ট ২ প্রকার : ক. প্রোগ্রাম ফ্লোচার্ট ও খ. সিস্টেম ফ্লোচার্ট।

ফ্লোচার্টের বৈশিষ্ট্য বা সুবিধাঃ

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 পরিবর্ধনে সহায়তা করে।
৫. সহজে ও সংক্ষেপে জটিল প্রোগ্রাম লেখা যায়।

ফ্লোচার্ট তৈরি করার নিয়মাবলী (Rules of drawing flowchart)

ফ্লোচার্ট তৈরি করার জন্য নিম্নবর্ণিত নিয়মাবলী অনুসরণ করা প্রয়োজন:

১. ফ্লোচার্ট তৈরি করার জন্য প্রচলিত প্রতীক ব্যবহার করা উচিত।
২. তার চিহ্ন দিয়ে উপর থেকে নিচে বা বাম থেকে ডান দিকে প্রবাহ দেখানো উচিত।
৩. ফ্লোচার্ট তৈরি করার সময় সংযোগ চিহ্ন যত কম হয় ততই ভাল।
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

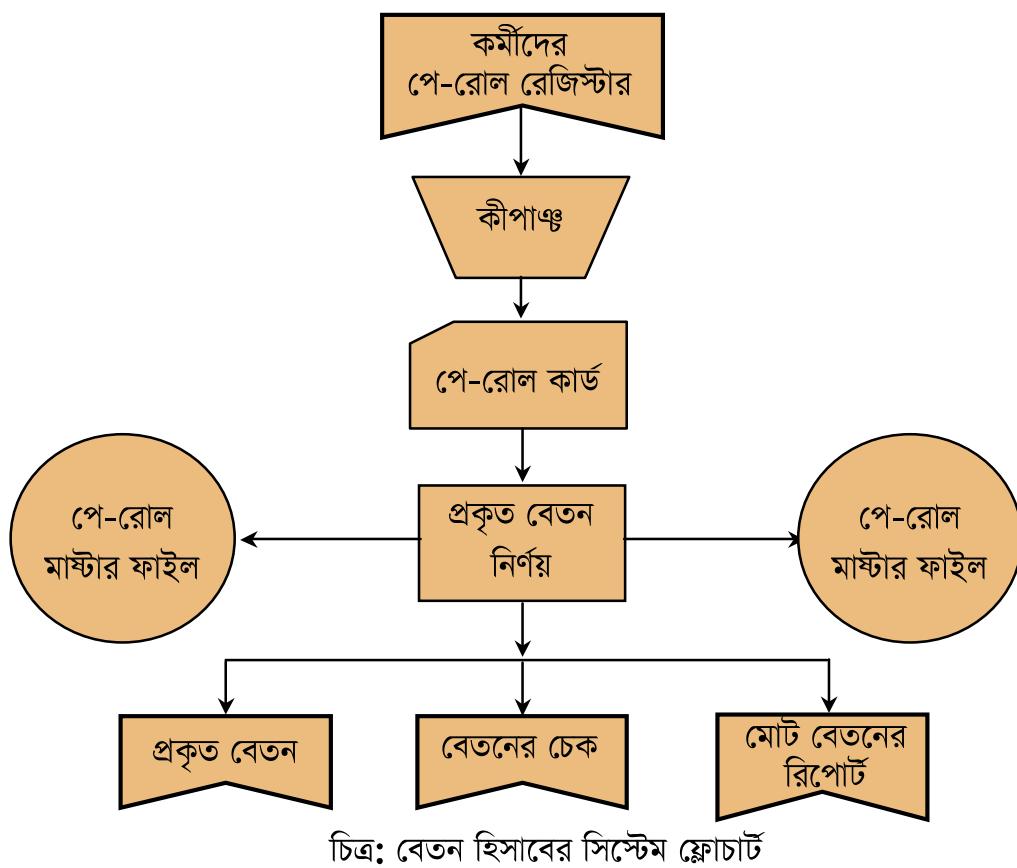
২. প্রোগ্রাম ফ্লোচার্ট

১. সিস্টেম ফ্লোচার্ট: সিস্টেম ফ্লোচার্টে উপাত্ত গ্রহণ, প্রক্রিয়াকরণ, স্মৃতিতে সংরক্ষণ ও ফলাফল প্রদর্শনের প্রবাহ দেখানো হয়। অর্থাৎ যে ফ্লোচার্টের মাধ্যমে কোন ব্যবস্থার সংগঠনকে সহজে তুলে ধরা যায় তাকে সিস্টেম ফ্লোচার্ট বলে।

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

চিত্র : সিস্টেম ফ্লোচার্টে ব্যবহৃত প্রতীক

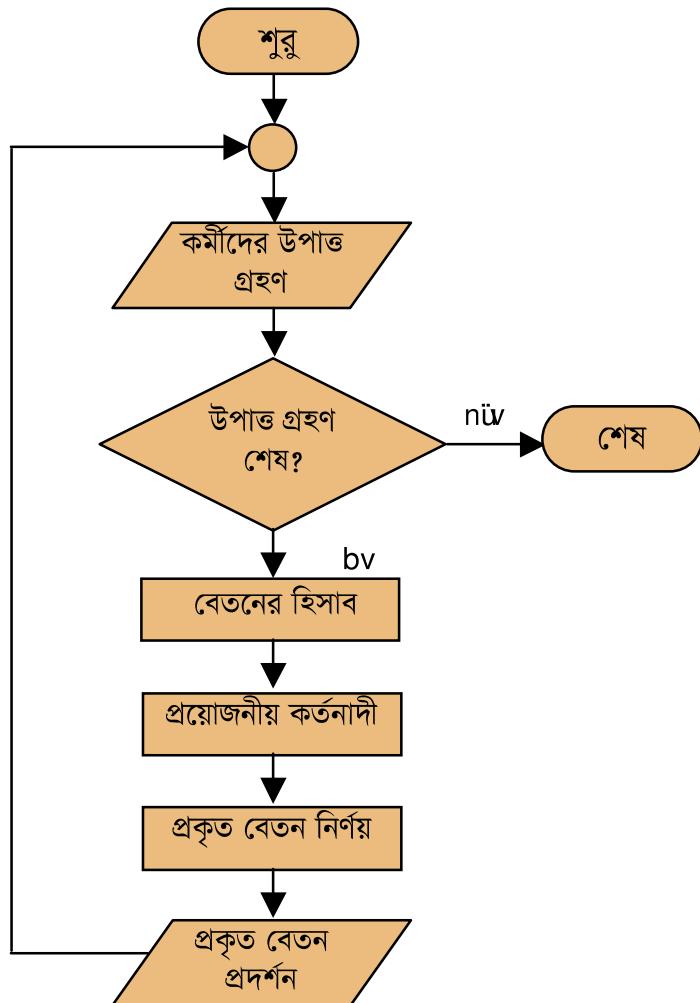
নিম্নে বেতন নির্ণয়ের জন্য একটি সিস্টেম ফ্লোচার্ট দেখান হয়েছে-



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

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

প্রতীক	অর্থ	প্রতীক	অর্থ
oval	Start/End	rectangle	Process
diamond	Decision	parallelogram	Predefined process
trapezoid	Input/Output	circle	Connector
arrow	Direction of flow	left arrow	Sort note



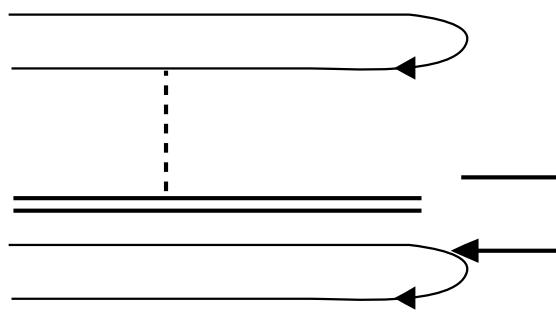
ফ্লোচার্টের মৌলিক গঠন:

1. সরল অনুক্রম (Simple sequence)
 2. নির্বাচন (Selection)
 3. লুপ বা চক্র (Repetition or loop)
 4. জাম্প (Jump)
১. **সরল অনুক্রম (Simple sequence):** এটি একটি সরল স্ট্রাকচার। এই স্ট্রাকচারে সকল নির্দেশগুলি নির্বাহের অনুক্রমে সাজানো থাকে।



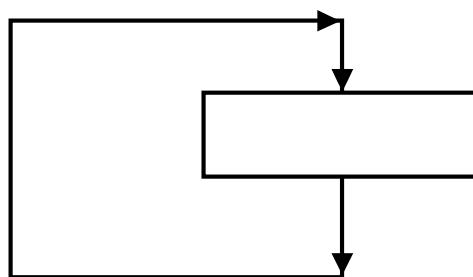
চিত্র: সরল অনুক্রম

২. **নির্বাচন (Selection):** যে সকল ক্ষেত্রে সিদ্ধান্তের প্রয়োজন হয় বা তুলনা করে কার্য নির্বাচ করতে হয় সেক্ষেত্রে এই স্ট্রাকচার ব্যবহৃত হয়।



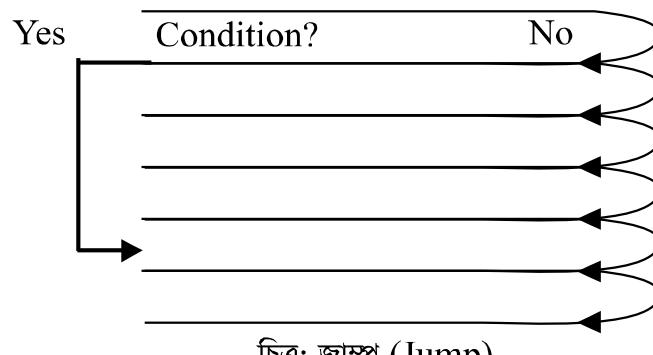
চিত্র: নির্বাচন

৩. লুপ বা চক্র (Repetition or loop): প্রোগ্রামে একই ধরনের কাজ বারবার করার প্রয়োজন হলে লুপ বা চক্র ব্যবহার করা হয়।



চিত্র: লুপ বা চক্র (Loop)

৪. জাম্প (Jump): প্রোগ্রামে সরল অনুক্রমকে ভঙ্গ করে প্রোগ্রামের মধ্যে এক লাইন থেকে পরবর্তী লাইনে না গিয়ে উপরে বা নিচে অন্য কোন লাইন থেকে কাজ শুরু করলে তাকে জাম্প বলে।



চিত্র: জাম্প (Jump)

সূত্রকোড (Pseudocode):

সূত্র (Pseudo) একটি গ্রীক শব্দ যার অর্থ ছদ্ম বা যা সত্য নয়। এটি প্রোগ্রাম ডিজাইনের ক্ষেত্রে ব্যবহৃত একটি পদ্ধতি। সূত্রকোডের মাধ্যমে সুন্দর ও সহজ ইংরেজি ভাষায় প্রোগ্রামের বিভিন্ন ধাপ বর্ণনা করা হয় যা দেখতে কোন প্রোগ্রামিং ভাষার কোডিংয়ের মতো মনে হয়। সূত্রকোড নির্দিষ্ট কোনো প্রোগ্রামিং ভাষার উপর নির্ভরশীল নয়। এ পদ্ধতিতে একটি প্রোগ্রামকে এমনভাবে উপস্থাপন করা হয় যেন সকলে তা সহজে বুঝতে পারে। সূত্রকোডকে অনেক সময় অ্যালগরিদমের বিকল্প হিসেবে বিবেচনা করা হয়। তিনটি সংখ্যার গড় নির্ণয়ের জন্য সূত্রকোড হচ্ছে—

```

Start
Input X,Y,Z
Total = X+Y+Z
Average = Total/3
Output Average
End
  
```

অ্যালগরিদম ও ফ্লোচার্টের পার্থক্য

অ্যালগরিদম	ফ্লোচার্ট
১. যে পদ্ধতিতে ধাপে ধাপে অগ্রসর হয়ে কোনো একটি নির্দিষ্ট সমস্যার সমাধান করা হয় তাকে বলা হয় অ্যালগরিদম।	১. যে পদ্ধতিতে চিত্রের সাহায্যে কতকগুলো চিহ্ন ব্যবহার করে সমস্যার ধারাবাহিক সমাধান করা হয় তাকে বলা হয় ফ্লোচার্ট।
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 ভুলত্রুটি দূর করা সহজ।

নিচে কিছু সমস্যার জন্য অ্যালগরিদম ও ফ্লোচার্ট দেখানো হলো:

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

অ্যালগরিদম:

ধাপ-১: প্রোগ্রাম শুরু করি।

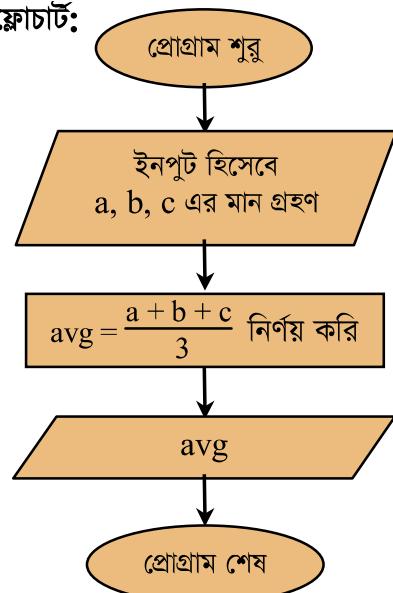
ধাপ-২: ইনপুট হিসেবে a, b এবং c এর মান গ্রহণ করি।

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. ত্রিভুজের ক্ষেত্রফল নির্ণয়ের অ্যালগরিদম ও ফ্লোচার্ট

(তিনি বাহুর ক্ষেত্রে)।

অ্যালগরিদম:

ধাপ-১: প্রোগ্রাম শুরু করি।

ধাপ-২: ইনপুট হিসেবে a, b এবং c (তিনি বাহুর
দৈর্ঘ্য) এর মান গ্রহণ করি।

ধাপ-৩: $S = (a + b + c) / 2$ ব্যবহার করে S এর
মান নির্ণয় করি।

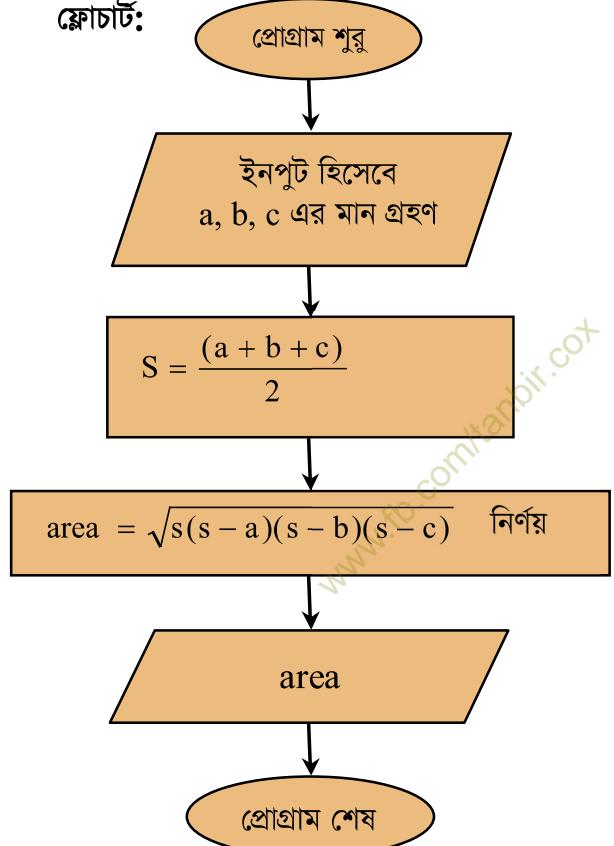
ধাপ-৪: $area = \sqrt{S(S - a)(S - b)(S - c)}$

ব্যবহার করে area এর মান নির্ণয় করি।

ধাপ-৫: area এর মান ছাপাই।

ধাপ-৬: প্রোগ্রাম শেষ করি।

ফ্লোচার্ট:



উদাহরণ-৩. ত্রিভুজের ক্ষেত্রফল নির্ণয়ের অ্যালগরিদম ও ফ্লোচার্ট (ভূমি ও উচ্চতার ক্ষেত্রে)।

অ্যালগরিদম:

ধাপ-১: প্রোগ্রাম শুরু করি।

ধাপ-২: ইনপুট হিসেবে b এবং h (ভূমি, উচ্চতা)
এর মান গ্রহণ করি।

ধাপ-৩: $area = (b * h) / 2$ ব্যবহার করে
area এর মান নির্ণয় করি।

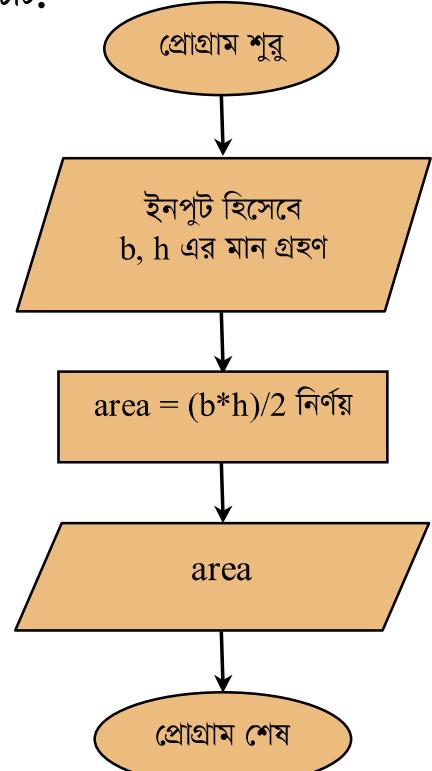
ধাপ-৪: area এর মান ছাপাই।

ধাপ-৫: প্রোগ্রাম শেষ করি।



জেনে রাখো: ত্রিভুজের ক্ষেত্রফল নির্ণয়ের জন্য দুটি পদ্ধতি
প্রচলিত। একটি হচ্ছে তিনি বাহুর দৈর্ঘ্য দেওয়া থাকলে ক্ষেত্রফল
নির্ণয়। অন্যটি, ভূমি ও উচ্চতার মান দেওয়া থাকলে ক্ষেত্রফল
নির্ণয়।

ফ্লোচার্ট:



উদাহরণ-৪. তিনটি 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: প্রথম সংখ্যাটি কি দ্বিতীয় ও
তৃতীয় সংখ্যার চেয়ে বড়?
ক. হ্যাঁ,
ক. না।

সংখ্যাটি অর্থাৎ a বড় এবং
৬নং ধাপে যাও।
খ. না।

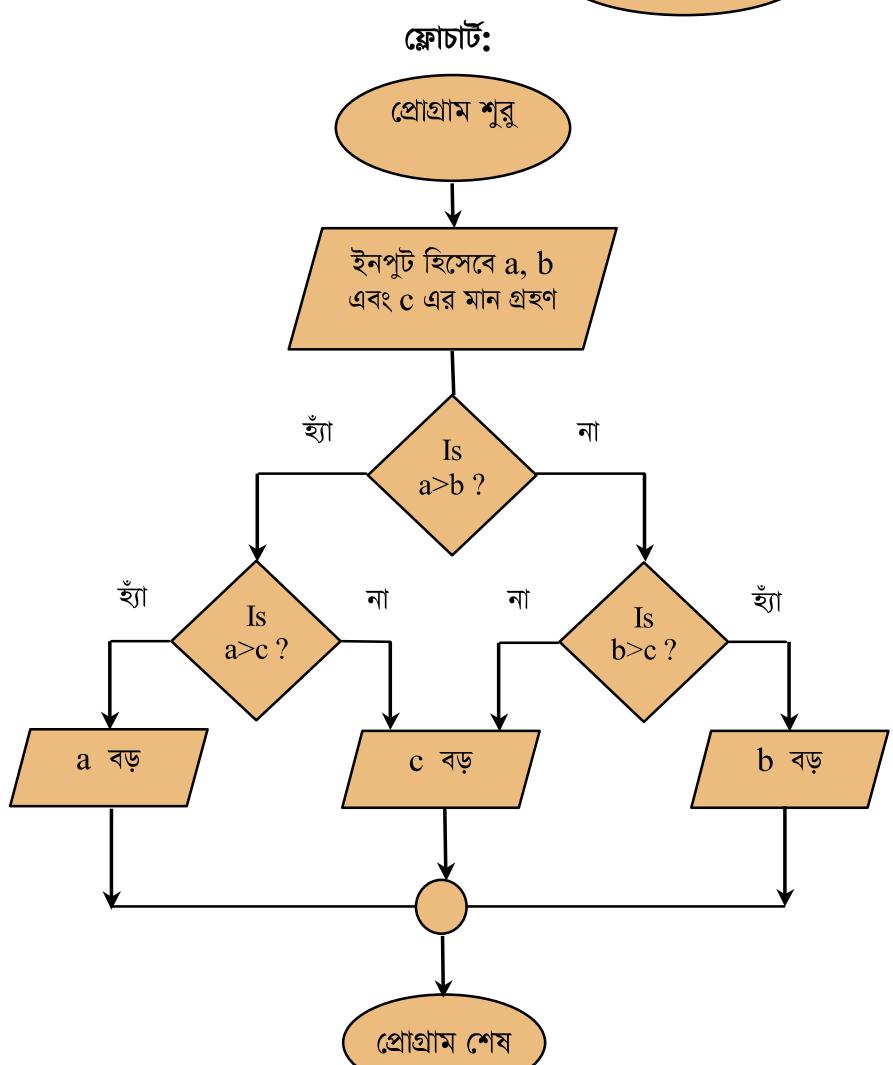
ধাপ-৪: দ্বিতীয় সংখ্যাটি কি তৃতীয়
সংখ্যার চেয়ে বড়?
ক. হ্যাঁ,
ক. না।

সংখ্যাটি অর্থাৎ b বড় এবং
৬নং ধাপে যাও।
খ. না।

ধাপ-৫: ফলাফল ছাপ, তৃতীয় সংখ্যাটি
অর্থাৎ c বড়।

ধাপ-৬: প্রোগ্রাম শেষ।

ফ্লোচার্ট:



উদাহরণ-৫. দুটি সংখ্যার গ.সা.গু নির্ণয়ের জন্য অ্যালগরিদম ও ফ্লোচার্ট।

অ্যালগরিদম:

ধাপ-১: প্রোগ্রাম শুরু।

ধাপ-২: দুটি সংখ্যা L , S ইনপুট হিসেবে নেওয়া হলো।
($L > S$)

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

ধাপ-৪: ভাগশেষের মান (R) যদি ০ হয় তবে ৫ নং ধাপে
গমন,

অন্যথায়, নতুনভাবে $L = S$ এবং $S = R$ করে
পুনরায় ৩ নং ধাপে গমন।

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

ধাপ-৬: প্রোগ্রাম শেষ।

উদাহরণ-৬. সেন্টিগ্রেড তাপমাত্রাকে ফারেনহাইট-এ

রূপান্তরের জন্য অ্যালগরিদম ও ফ্লোচার্ট নির্ণয়।

অ্যালগরিদম:

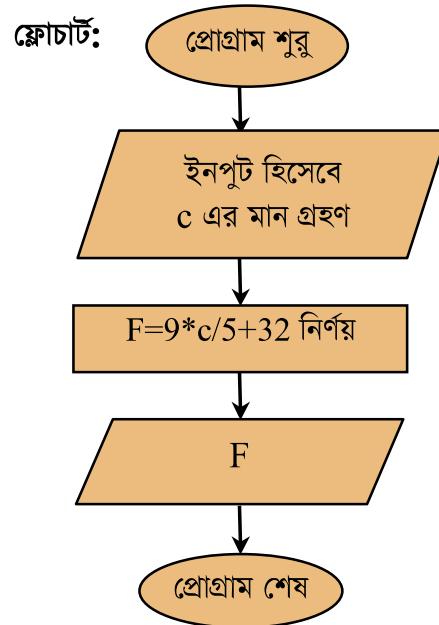
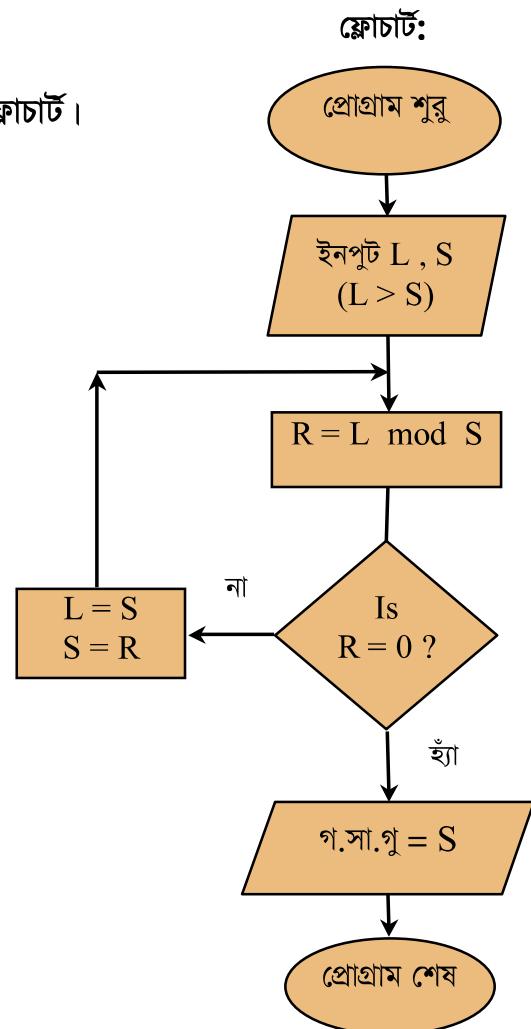
ধাপ-১: প্রোগ্রাম শুরু করি।

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।

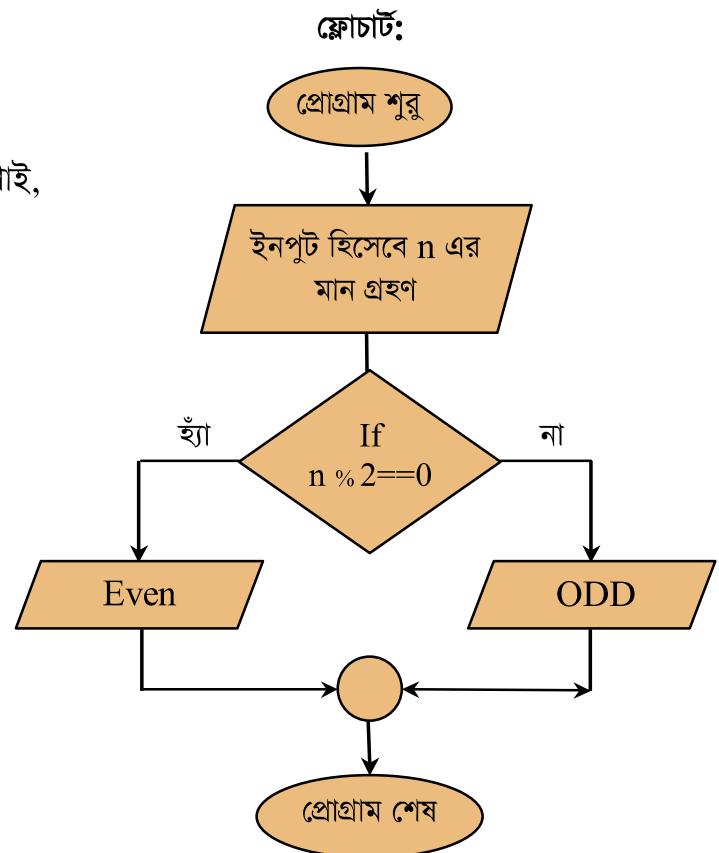
ধাপ-৫: প্রোগ্রাম শেষ করি।



উদাহরণ-৭. কোনো সংখ্যা জোড় না বিজোড় নির্ণয়ের জন্য অ্যালগরিদম ও ফ্লোচার্ট।

অ্যালগরিদম:

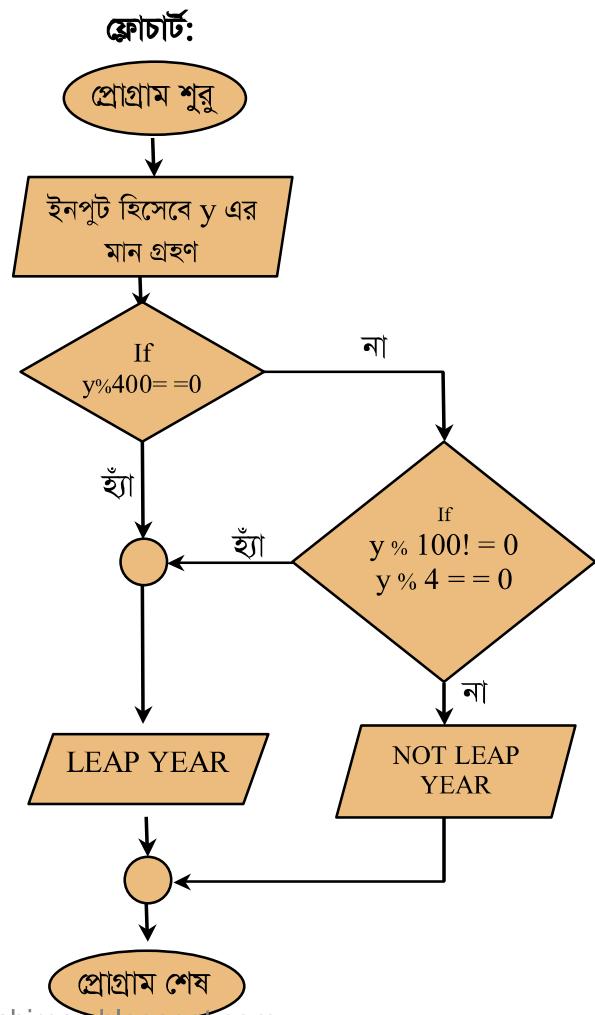
- ধাপ-১: প্রোগ্রাম শুরু করি।
- ধাপ-২: ইনপুট হিসেবে n এর মান গ্রহণ করি।
- ধাপ-৩: যদি ($n \% 2 == 0$) হয় তবে 'Even' ছাপাই,
নেং ধাপে যাই।
- ধাপ-৪: 'ODD' ছাপাই।
- ধাপ-৫: প্রোগ্রাম শেষ করি।



উদাহরণ-৮. কোনো বর্ষ অধিবর্ষ (Leap year) কি না তা নির্ণয়ের জন্য অ্যালগরিদম ও ফ্লোচার্ট।

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।
- ধাপ-৬: LEAP YEAR ছাপাই।
- ধাপ-৭: প্রোগ্রাম শেষ করি।



উদাহরণ-৯. $1 + 2 + 3 + \dots + n$ ধারাটির

যোগফল নির্ণয়ের জন্য অ্যালগরিদম ও ফ্লোচার্ট।

অ্যালগরিদম:

ধাপ-১: প্রোগ্রাম শুরু করি।

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

ধাপ-৩: $s = 0, i = 1$ ধরি।

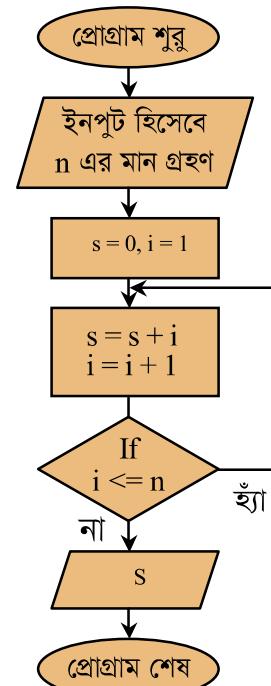
ধাপ-৪: $s = s + i, i = i + 1$ নির্ণয় করি।

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

ধাপ-৬: s এর মান ছাপাই।

ধাপ-৭: প্রোগ্রাম শেষ করি।

ফ্লোচার্ট:



উদাহরণ-১০. কোনো পূর্ণ সংখ্যার ফ্যাক্টরিয়াল নির্ণয়ের জন্য অ্যালগরিদম ও ফ্লোচার্ট।

অ্যালগরিদম:

ধাপ-১: প্রোগ্রাম শুরু করি।

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

ধাপ-৩: $F = 1, i = 1$ ধরি।

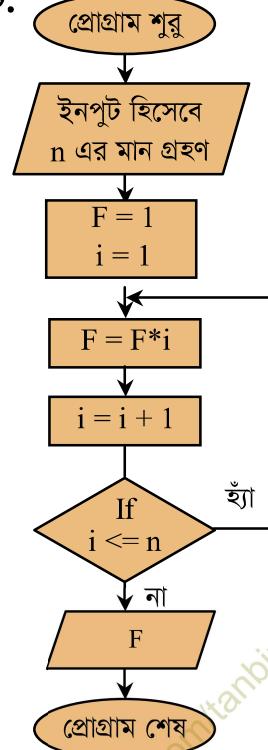
ধাপ-৪: $F = F * i, i = i + 1$ নির্ণয় করি।

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

ধাপ-৬: F এর মান ছাপাই।

ধাপ-৭: প্রোগ্রাম শেষ করি।

ফ্লোচার্ট:



লক্ষ করো: ফ্যাক্টরিয়াল নির্ণয় মূলত একটি গুণের ধারা। এটি অনেকটা যোগের ধারার মতো।



কাজ :

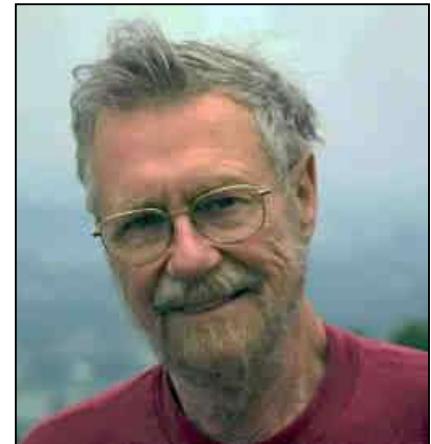
- আয়তক্ষেত্রের ক্ষেত্রফল নির্ণয়ের জন্য অ্যালগরিদম লেখ ও ফ্লোচার্ট অঙ্কন করো।
- তিনটি সংখ্যার মধ্যে ক্ষুদ্রতম সংখ্যাটি নির্ণয়ের অ্যালগরিদম লেখ ও ফ্লোচার্ট অঙ্কন করো।
- দুটি সংখ্যার মধ্যে ল. সা. গু নির্ণয়ের জন্য অ্যালগরিদম লেখ ও ফ্লোচার্ট অঙ্কন করো।
- ফারেনহাইট তাপমাত্রাকে সেন্টিগ্রেড তাপমাত্রায় রূপান্তর করার জন্য অ্যালগরিদম লেখ ও ফ্লোচার্ট অঙ্কন করো।
- ২০১২, ২০১৩ ও ২০১৪ সালের মধ্যে কোনটি লিপইয়ার এবং কোনটি লিপইয়ার নয় তা নির্ণয় করো।

পাঠ-১১ ও ১২

প্রোগ্রাম ডিজাইন মডেল (Program Design Model)

প্রোগ্রাম ডিজাইন মডেল: প্রোগ্রামের গঠনশৈলীকে বা প্রোগ্রামের গঠন রীতিনীতিকে প্রোগ্রামের মডেল বলা হয়। সুচারুভাবে 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:

স্ট্রাকচার্ড প্রোগ্রামিং: ডাচ কম্পিউটার বিজ্ঞানী এডগার ওয়েইবে ডেইকস্ট্রা (Edsger Wybe Dijkstra) প্রথম বড় আকারের প্রোগ্রাম উন্নয়নের উদ্দেশ্যে স্ট্রাকচার্ড প্রোগ্রামিং-এর ধারণা দেন। স্ট্রাকচার্ড মডেলে পুরো সমস্যাকে বিভিন্ন অংশ বা মডিউলে ভাগ করা হয়। প্রতিটি মডিউলকে ছোটো আকারের সমস্যা ভাবা 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। স্ট্রাকচার্ড প্রোগ্রামে তিন ধরনের কাঠামো ব্যবহৃত হয়ে থাকে—



চিত্র: এডগার ডেইকস্ট্রা

- **পর্যায়ক্রমিক কাঠামো:** এ কাঠামোতে প্রোগ্রামের বা মডিউলের একটির পর একটি নির্দেশ ধারাবাহিকভাবে নির্বাহ হয়। নির্দেশের ধারাবাহিকতা বা পর্যায় কখনো বিপ্লিত হয় না।
- **সিদ্ধান্তমূলক কাঠামো:** এ কাঠামো একটি নির্দিষ্ট শর্তের ওপর নির্ভর করে। শর্তটি সত্য হলে, একটি স্টেটমেন্ট বা নির্দেশ নির্বাহ হয়; আর শর্ত 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 শতহানভাবে নির্দিষ্ট সংখ্যক বার বা শর্তের অধীন অনিদিষ্ট সংখ্যক বার নির্বাহ করা যায়।

স্ট্রাকচার্ড প্রোগ্রামিং বা প্রোসিডিউর অরিয়েন্টেড প্রোগ্রামিং-এর বৈশিষ্ট্য:

স্ট্রাকচার্ড প্রোগ্রামিং বা প্রোসিডিউর অরিয়েন্টেড প্রোগ্রামিং-এর কিছু বৈশিষ্ট্য আছে যেগুলো নিম্নে দেয়া হল-

- মূল প্রোগ্রামকে ফাংশনের ভিত্তিতে ছোট ছোট অংশে ভাগ করা হয়।
- প্যারামিটার পাসিং বা অপারেটিং সিস্টেমের মাধ্যমে প্রোগ্রামের অংশগুলোর মধ্যে সংযোগ স্থাপিত হয়।
- প্রতিটি ফাংশনের ডেটা আলাদা থাকে।
- প্রোগ্রামে ডেটা অপেক্ষা ফাংশনের অধিক গুরুত্ব দেয়া হয়।
- বেশির ভাগ ফাংশন প্লেবাল ডেটা ব্যবহার করে।
- একই ডেটা এক ফাংশন থেকে অন্য ফাংশনে স্থানান্তরিত হতে পারে।
- ডেটা গোপন করার কার্যকরী কোন উপায় নেই।
- সাধারণ নিয়মে এক ফাংশন অন্য ফাংশনের সাথে সংযোগ রক্ষা করে।
- প্রয়োজনে যে কোন সময় প্রোগ্রামে অতিরিক্ত ডেটা বা ফাংশন যোগ করা যায় না। এ জন্য পুরো প্রোগ্রাম পরিবর্তন করতে হতে পারে।
- প্রোগ্রামে নতুন ডেটা যোগ করার সময় ব্যবহারকারীকে নিশ্চিত হতে হয় যে তা ফাংশন কর্তৃক অনুমোদিত।
- প্রোগ্রাম ডিজাইনে টপ-ডাউন পদ্ধতি অনুসরণ করা হয়।

- প্রোগ্রামিং-এ প্রোসিডিউর অরিয়েন্টেড প্রোগ্রাম ভাষা, যেমন- কোবল, প্যাসকেল, ফরট্রান, সি ইত্যাদি ব্যবহার করা হয়।

উদাহরণ: C, Pascal, Qbasic

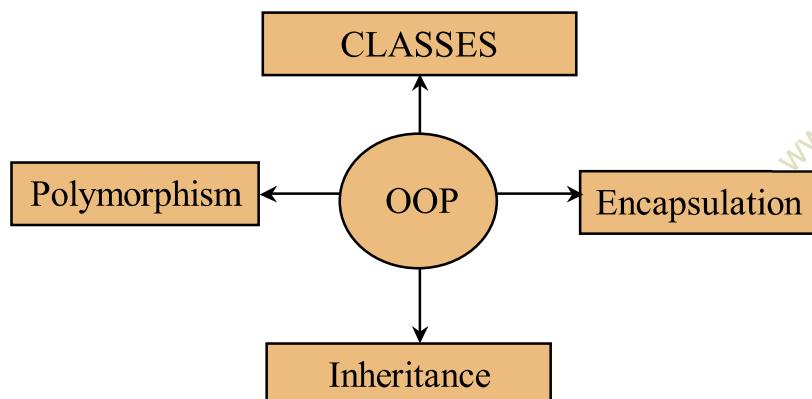
স্ট্রাকচার্ড প্রোগ্রামিং-এ প্রতিটি ফাংশনের ডেটা আলাদা থাকে। প্রোগ্রামে ডেটা অপেক্ষা ফাংশনের অধিক গুরুত্ব দেওয়া হয়। বেশির ভাগ ফাংশন প্লোবাল ডেটা ব্যবহার করে। একই ডেটা এক ফাংশন থেকে অন্য ফাংশনে স্থানান্তরিত হতে পারে। এ পদ্ধতিতে ডেটা গোপন করার কার্যকরী কোনো উপায় নেই। এতে সাধারণ নিয়মে এক ফাংশন অন্য ফাংশনের সাথে সংযোগ রক্ষা করে। এ ধরনের প্রোগ্রামিং-এ প্রয়োজনে যেকোনো সময় প্রোগ্রামে অতিরিক্ত ডেটা বা ফাংশন যোগ করা যায় না। এ জন্য পুরো প্রোগ্রাম পরিবর্তন করতে হতে পারে। **উদাহরণ:** C, Pascal, Qbasic ইত্যাদি।

অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং (OOP): অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং মডেলে ডেটা ও সংশ্লিষ্ট কোডকে একক হিসেবে বিবেচনা করা হয়। এ ধরনের একককে ক্লাস (Class) বলে। এক ক্লাসের ডেটা অন্য ক্লাসের কাছে অদ্ধ্য। ফলে অনিচ্ছাকৃতভাবে ভুল চলকের মান ব্যবহার করা সম্ভব নয়। এক একটি ক্লাস এক একটি ধরন বোঝায়। ক্লাসে কোনো ডেটা রেখে নির্বাহ করতে হলে নির্দিষ্ট ক্লাসের অবজেক্ট তৈরি করতে হয়। অবজেক্টের বিভিন্ন কোডকে নির্বাহ করানোর জন্য সংশ্লিষ্ট অবজেক্টকে বিশেষ বার্তা পাঠাতে হয়। কোনো প্রোগ্রাম উন্নয়নের সময় ক্লাসগুলো এমনভাবে নির্মাণ করা হয়, যাতে তা বাস্তব সমস্যাকে ভালোভাবে উপস্থাপন করতে পারে।

ছোটো 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 ছোটো থেকে বড় প্রোগ্রাম উন্নয়ন করা যায়। সকল প্রোগ্রামিং ভাষাই অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং সমর্থন করে না। কোনো প্রোগ্রামিং ভাষাকে পরিপূর্ণ অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং ভাষা হতে হলে কমপক্ষে তিনটি বৈশিষ্ট্য থাকতে হয়। এগুলো হলো-

- ইনহেরিটেন্স:** ইনহেরিটেন্স এমন একটি ক্ষমতা যার মাধ্যমে কোনো প্রচলিত ক্লাসের কোনো পরিবর্তন না করে পরিবর্ধিত নতুন ক্লাস তৈরি করা যায়। নতুন ক্লাস মূল ক্লাসের প্রয়োজনীয় বৈশিষ্ট্য ধারণ করে থাকে।
- এনক্যাপ্সুলেশন:** কোনো চলকের ডেটা এবং ইনস্ট্রাকশন একত্রিত অবস্থায় থাকাকে এনক্যাপ্সুলেশন বলে। যে চলকের জন্য যে ডেটা সেই চলকের বাইরে তার আর কোনো অস্তিত্ব নেই। ফলে ডেটার ওপর প্রোগ্রামের অন্য অংশের কোনো প্রভাব পড়ে না।
- পলিমরফিজম:** 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।

OOP প্রোগ্রামিং ভাষার উদাহরণ হলো C++, Java, C# ইত্যাদি।



ভিজুয়্যাল প্রোগ্রামিং: গ্রাফিক্যাল ইউজার ইন্টারফেস (GUI) সমৃদ্ধ পরিবেশকে বলা হয় ভিজুয়্যাল (Visual) বা দৃশ্যমান কাজের পরিবেশ। চিত্রভিত্তিক বা ভিজুয়্যাল পরিবেশে কাজ করা সহজ বলে চিত্রভিত্তিক প্রোগ্রামিং ভাষাও উত্তীর্ণ হয়েছে। চিত্রভিত্তিক পরিবেশ হলেও প্রোগ্রামিং-এর ক্ষেত্রে মূলত স্ট্রাকচার্ড প্রোগ্রামিং অথবা OOP মডেল ব্যবহৃত হয়। তবে প্রকৃত প্রোগ্রামিং মডেল প্রোগ্রাম থেকে আড়ালে থাকে বলে চিত্রভিত্তিক প্রোগ্রামিং বেশ জনপ্রিয়। মাইক্রোসফট কোম্পানির ভিজুয়্যাল বেসিক হলো প্রথম চিত্রভিত্তিক প্রোগ্রামিং মডেল। এর জনপ্রিয়তা দেখে বোরল্যান্ড নির্মাণ করে ডেলফি। বর্তমানে ভিজুয়্যাল বেসিক ও ডেলফি দুটিই জনপ্রিয় সফটওয়্যার। ভিজুয়্যাল প্রোগ্রামিং কে অনেকে প্রোগ্রামিং ভাষা বা মডেল বলতে চান না। কারণ, এগুলো 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।

ইভেন্ট ড্রাইভেন প্রোগ্রামিং: 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 উত্তীর্ণ 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 অলস 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 স্ফূর্তি হয়। সাথে সাথে ইভেন্টের সংশ্লিষ্ট কোড দ্বারা মডিউলটি নির্বাহ হয়। সব ভিজুয়্যাল প্রোগ্রামই হচ্ছে ইভেন্ট ড্রাইভেন প্রোগ্রাম।



কাজ: প্রত্যেক প্রোগ্রাম মডেলের তিনটি করে বৈশিষ্ট্য লেখ এবং প্রত্যেক মডেলের আওতাভুক্ত দুটি প্রোগ্রাম ভাষার নাম লেখ।

Id: www.facebook.com/tanbir.cox

Page: www.facebook.com/tanbir.ebooks

Web: www.tanbircox.blogspot.com

পাঠ-১৩ ও ১৪

‘সি’ প্রোগ্রাম (C Program)



চিত্র: ডেনিস রিচি



সি একটি স্ট্রাকচার্ড বা প্রোসিডিউর অরিয়েন্টেড প্রোগ্রামিং ল্যাঙ্গুয়েজ। বর্তমানে মিড লেভেল ল্যাঙ্গুয়েজ হিসেবে সি অত্যন্ত জনপ্রিয়। সি নামটা এসেছে মার্টিন রিচার্ডস (Martins Richards) এর উদ্ভাবিত বিসিপিএল (BCPL-Basic Combined Programming Language) ভাষা থেকে যা প্রাথমিকভাবে ক্যাম্বিজ বিশ্ববিদ্যালয়ে রিসার্স অরিয়েন্টেড কাজে ব্যবহৃত হত। BCPL সংক্ষেপে বি নামে পরিচিত ছিল। পরে বি এর উন্নয়নের ফলে সি এর বিকাশ ঘটে। ১৯৭০ সালে যুক্তরাষ্ট্রের টিএন্টি বেল ল্যাবোরেটরিতে (AT&T Bell Laboratory) ডেনিস রিচি (Danins Rictche) 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 হত। ১৯৭৮ সালে ডেনিস রিচির লেখা “দ্য সি প্রোগ্রামিং ল্যাঙ্গুয়েজ” বইটি প্রকাশের পর এবং মাইক্রোকম্পিউটারের জনপ্রিয়তা বাড়ার সাথে সাথে সি এর ব্যাপক প্রচলন শুরু হয়। সে সময় সি এর জনপ্রিয়তা বাড়ার কারণ ছিল এক কম্পিউটারে লেখা প্রোগ্রাম অন্য কম্পিউটারে ব্যবহারের সুবিধা। সি দিয়ে সহজে উচ্চ স্তরের এবং নিম্নস্তরের ভাষার মধ্যে সমন্বয় করা যায়। আবার উচ্চ স্তরের ভাষার (যেমন- ফরট্রোন) মতো বিট, বাইট, ও মেমরি 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 প্রতিটি অংশের জন্য আলাদাভাবে ভেরিয়েবল, স্ট্রাকচার, ফাংশন ইত্যাদি বর্ণনা করা যায় এবং প্রয়োজনে if, while, for, goto ইত্যাদি কন্ট্রোল স্টেটমেন্টের মাধ্যমে বিভিন্ন অংশের মধ্যে সমন্বয় সাধন করা যায়, কিংবা কোন ফাংশন বা স্ট্রাকচার পুনঃব্যবহার করা যায়। আনস্ট্রাকচার্ড ভাষায় (যেমন, বেসিক) এভাবে মূল সমস্যাকে একাধিক অংশে বিভক্ত করে প্রতিটি আলাদা অংশের জন্য আলাদাভাবে ফাংশন বর্ণনা করা যায় না। সি প্রোগ্রামিং ভাষাটি সব ধরনের কাজের জন্য ব্যবহৃত হয়। একজন প্রোগ্রামের যেসব সুবিধা দরকার, যেমন- বিভিন্ন ডেটা ব্যবহারের ব্যাপক স্বাধীনতা, স্বল্প সংখ্যক কী-ওয়ার্ড, দুট 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।

C প্রোগ্রাম ভাষার বৈশিষ্ট্য (Characteristics of ‘C’ language)

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

- সি একটি মধ্যস্তরের ভাষা। এ ভাষায় কম্পিউটারের বিট পর্যায়ের প্রোগ্রামিং এর মাধ্যমে হার্ডওয়্যার নিয়ন্ত্রণ ও সিস্টেম প্রোগ্রাম রচনা করা যায়।
- এ ভাষায় উচ্চস্তরের ভাষার সুবিধা পাওয়া যায় আবার নিম্নস্তরের ভাষা সমকক্ষ প্রোগ্রাম রচনা করা যায়।
- সি ল্যাংগুয়েজ দিয়ে সব ধরনের প্রোগ্রাম রচনা করা যায়। তাই একে General purpose language ও বলা হয়।
- সি ল্যাংগুয়েজে মূল সমস্যাকে ছোট ছোট ভাগে ভাগ করে প্রতিটি ভাগে প্রয়োজনীয় ভেরিয়েবল, ফাংশন, কন্ট্রোল ইত্যাদি ব্যবহার করা যায় এবং প্রয়োজনে ইত্যাদি কন্ট্রোল স্টেটমেন্টও ব্যবহার করা যায়। তাই সি-কে স্ট্রাকচার্ড ল্যাংগুয়েজও বলা হয়।

- সি প্রোগ্রামের ভাষা শুরু হয় একটি ফাংশন main() এর মাধ্যমে। প্রতিটি প্রোগ্রামের কাজ এই ফাংশন থেকে শুরু হয়।
- ফাংশনের মধ্যে যেসব Statement থাকে সেগুলোকে দ্বিতীয় বন্ধনী এর রাখতে হয়।
- প্রতিটি Statement-এর শেষে সেমিকোলন (;) দিতে হয়।
- প্রোগ্রামে কোন Comment ব্যবহার করতে হলে তার আগে /* চিহ্ন এবং শেষে */ চিহ্ন ব্যবহার করতে হয়।
- প্রোগ্রামের যে কোন স্থানে যতগুলো ইচ্ছা Comment দেখা যায়।
- পর্যাপ্ত সংখ্যাক লাইব্রেরি ফাংশন, ব্রাঞ্চিং স্টেটমেন্ট ও কন্ট্রোল স্টেটমেন্টের সুবিধা রয়েছে।
- সাধারণত সি ল্যাঙ্গুয়েজ দিয়ে লিখিত এক মেশিনের প্রোগ্রাম অন্য মেশিনে চালানো যায়।
- এতে অনেক পর্যাপ্ত সংখ্যক কম্পাউন্ড অপারেটর, যেমন- +, -, *= ইত্যাদি রয়েছে।

সি প্রোগ্রামিং ভাষার সুবিধা:

- এ ভাষার স্টেটমেন্ট গুলো ইংরেজি ভাষার মত বলে শেখা সহজ।
- প্রোগ্রাম রচনা করা সহজ।
- প্রোগ্রামের মধ্যে যেকোনো স্থানে কমেন্ট দেওয়া যায়।
- উচ্চ স্তরের ও মেশিন ভাষার প্রোগ্রামে লেখা যায়।
- অন্ন মেমরির 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
- দুটি প্রোগ্রাম নির্বাহ করা যায়।
- কোনো লাইনের নম্বর দিতে হয় না।

সি প্রোগ্রামিং ভাষার অসুবিধা:

- সি ভাষা case sensitive 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 সার্টিকভাবে চলক ঘোষণা করতে হয়।
- লাইব্রেরি ফাংশনের হেডার ফাইলগুলো ঠিকমত ডিক্লেয়ার করতে হয়।
- সি প্রোগ্রামিং ল্যাঙ্গুয়েজ অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ফিচারকে সমর্থন করে না।
- প্রোগ্রাম রান করার সময় চেকিং করা যায় না।

সি++ (C++) প্রোগ্রামিং ভাষা

সি++ একটি বহুল ব্যবহৃত অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ল্যাঙ্গুয়েজ। ১৯৮০ সালে যুক্তরাষ্ট্রের টি এন্টি টি বেল ল্যাবরেটরিতে বিয়ারনে স্ট্রাউস্ট্রুপ (Bjarne Stroustrup) সিমুলা ৬৭ এর ক্লাশ এবং সি প্রোগ্রামিং পদ্ধতির সমন্বয়ে অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং-এর বৈশিষ্ট সম্পর্ক "সি টেইথ ক্লাশ" নামে একটি নতুন ভাষা উভাবন করেন। পরবর্তী সময়ে স্ট্রাউস্ট্রুপের নীতির উপর ভিত্তি করে এতে অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং-এর অন্যান্য বৈশিষ্টসহ আরো কিছু নতুন বৈশিষ্ট যোগ করা হয় এবং ১৯৮৩ সালে এর নামকরণ করা হয় সি++। সি++ এর নামের সাথে সি এর ইনক্রিমেন্টাল অপারেটর (++) যুক্ত, তাই সাধারণত সি++ কে সি এর বর্ধিত



সংস্করণ বলা হয়। সি++-কে সি এর সুপারসেটও বলা হয়। কারণ সি এর প্রায় সকল বৈশিষ্টি সি++ এ বিদ্যমান। সামান্য কিছু ব্যতিক্রম ছাড়া সকল সি প্রোগ্রাম সি++ প্রোগ্রাম, কিন্তু সকল সি++ প্রোগ্রাম সি প্রোগ্রাম নয়। সি প্রোগ্রামের মত প্রতিটি সি++ প্রোগ্রাম এক বা একাধিক ফাংশনের সমষ্টি এবং প্রতিটি সি++ প্রোগ্রামে main() ফাংশন নামে একটি ইউজার-ডিফাইন্ড ফাংশন অবশ্যই থাকে। যেকোন সি/সি++ প্রোগ্রাম নির্বাহের সময় তা main() থেকেই শুরু হয়। main() ফাংশন অন্যান্য লাইব্রেরি এবং ইউজার-ডিফাইন্ড ফাংশন ব্যবহার করে।

C প্রোগ্রাম ভাষার গঠন

এক বা একাধিক ফাংশন নিয়ে 'সি' প্রোগ্রাম গঠিত হয়। তবে 'সি' প্রোগ্রামে main নামের ফাংশন অবশ্যই থাকতে হয়। একটি সি প্রোগ্রামে প্রধানত দুটি অংশ থাকে, হেডার ফাইল এবং main ফাংশন। তবে বড় প্রোগ্রামের ক্ষেত্রে main ফাংশনে ব্যবহৃত ফাংশনসমূহ আলাদা সোর্স ফাইলে বর্ণিত হয় এবং main ফাংশনকে পৃথক ফাইলে রাখা হয়। অতঃপর main ফাংশনে হেডার ফাইলসহ অন্যান্য ফাইলসমূহ সংযুক্ত করা হয়। চিত্রের মাধ্যমে সি-এর গঠন প্রণালি দেখানো হলো-

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

- Documentation Section: কোনো সি প্রোগ্রামের এটি ঐচ্ছিক অংশ। এ প্রোগ্রামের নাম, বিষয়বস্তু, প্রোগ্রামারের নাম, ব্যবহারের নিয়ম ও প্রোগ্রামের উদ্দেশ্য এখানে সংযুক্ত করা হয়। সাধারণত কমেন্টস এর মাধ্যমে এগুলো উল্লেখ করা হয়। সি ভাষায় কমেন্ট লেখার জন্য /*.....*/ এবং //....., ব্যবহার করা হয়।
- Link Section: এটি প্রোগ্রামের অত্যবশ্যিকীয় অংশ। প্রোগ্রামের বিভিন্ন স্থানে ব্যবহৃত ফাংশনগুলোর হেডার ফাইল এ অংশে সংযুক্ত করা হয়।
- Defination Section: এ অংশে কনস্ট্যান্ট ঘোষণা করা হয়।
- Global Declaration Section: এ অংশে গ্লোবাল চলক ঘোষণা করা হয়।
- main() ফাংশন: main() ফাংশন হলো প্রতিটি সি প্রোগ্রাম গঠনকারী প্রধান ফাংশন। এটি একটি ইউজার ডিফাইন্ড বা ব্যবহারকারী বর্ণিত ফাংশন, কারণ ব্যবহারকারী প্রোগ্রামার এর গঠন নির্ণয় করে থাকেন। সি প্রোগ্রামের মূল অংশ এই 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। একটি Declaration Part এবং অন্যটি Execution Part। Declaration Part-এ প্রয়োজনীয় চলক, অ্যারে, পয়েন্টার, ফাইল ইত্যাদি ঘোষণা করা হয় যা নির্বাহ অংশে ব্যবহার করা হবে এবং Execution Part-এ প্রোগ্রাম নির্বাহ হওয়ার জন্য কমপক্ষে একটি স্টেটমেন্ট থাকতে হয়। উভয় অংশের প্রত্যেক স্টেটমেন্টের শেষে সেমিকোলন(;) থাকতে হবে।

- Subprogram Section:** এক বা একাধিক ইউজার ডিফাইন্ড ফাংশন থাকে যা main() ফাংশনে কল করা হয়। ব্যবহারকারী কোনো ফাংশন তৈরি করে প্রোগ্রামে ব্যবহার করতে চাইলে তা প্রোগ্রামের main() ফাংশনের দ্বিতীয় বন্ধনীর বাইরে (উপরে বা নিচে) তৈরি করতে হয়।

হেডার ফাইল ও লাইব্রেরি ফাইল:

কম্পাইলারের যে সকল ফাইলের বর্ধিত নাম (.h) তাদেরকে হেডার ফাইল এবং যে সকল ফাইলের বর্ধিত নাম (.lib) সেগুলোকে লাইব্রেরি ফাইল বলা হয়। প্রতিটি কম্পাইলারের একটি শক্তিশালী লাইব্রেরি থাকে, যেখানে প্রোগ্রাম বাস্তবায়নে ব্যবহৃত বিভিন্ন ফাংশনের ঘোষণা এবং বিস্তারিত বর্ণনা দেয়া থাকে। একটি হেডার ফাইলে এক জাতীয় কতকগুলো লাইব্রেরি ফাংশন, বিল্ট-ইন ভেরিয়েবল, কনস্ট্যান্ট, স্ট্রাকচার ইত্যাদির প্রোটোটাইপ ঘোষণা করা থাকে এবং সংশ্লিষ্ট লাইব্রেরি ফাইলে সেগুলোর বিস্তারিত বর্ণনা দেয়া থাকে।

প্রিপ্রসেসর ডিরেক্টিভ:

যে সকল স্টেটমেন্ট # ক্যারেক্টার দিয়ে শুরু হয় সেগুলোকে প্রিপ্রসেসর ডিরেক্টিভ স্টেটমেন্ট বলা হয়। এগুলো ছোট প্রোগ্রাম বা প্রোগ্রামাংশ। সি প্রোগ্রামের মূল সোর্স ফাইলের সাথে প্রিপ্রসেসরগুলোতে দেয়া নির্দেশনা মোতাবেক অবজেক্ট ফাইল (.obj) তৈরি হয়। কিন্তু তাতে মূল সোর্স ফাইল পরিবর্তিত থাকে। পরবর্তীতে কম্পাইলার এই অবজেক্ট ফাইল থেকে এক্সিকিউটিবল ফাইল তৈরি করে। প্রিপ্রসেসর ব্যবহারে প্রোগ্রামের সৌন্দর্য বৃদ্ধি পায়, আকার ছোট হয় এবং প্রোগ্রাম বুঝতে সহজ হয়। প্রিপ্রসেসর ডিরেক্টিভগুলোর শেষে কোন সেমিকোলন থাকে না। প্রতিটি লাইনে কেবল একটি প্রিপ্রসেসর স্টেটমেন্ট থাকতে পারে।

বহুল ব্যবহৃত প্রিপ্রসেসর ডিরেক্টিভসমূহকে ৬টি গ্রুপে ভাগ করা হয়। যথা-

- ফাইল সংযুক্ত ডিরেক্টিভ
- কনস্ট্যান্ট ডিরেক্টিভ
- ম্যাক্রো ডিরেক্টিভ
- কম্পাইলার কন্ট্রোল ডিরেক্টিভ
- প্রাগমা ডিরেক্টিভ এবং
- স্পেশাল ডিরেক্টিভ

ফাইল সংযুক্তকারী ডিরেক্টিভ:

#include স্টেটমেন্টকে ফাইল সংযুক্তকারী ডিরেক্টিভ স্টেটমেন্ট বলা হয়। প্রোগ্রামে কোনো হেডার ফাইল বা সোর্স ফাইল সংযুক্ত করার জন্য এই ডিরেক্টিভ ব্যবহৃত হয়। এভাবে প্রোগ্রামে কোনো ফাইল যোগ করলে ফাইলের উপাদানসমূহ সংযুক্তকারী ফাইলে কপি হয়। সি প্রোগ্রামে ফাইল সংযুক্তকারী ডিরেক্টিভ মোট দুইভাবে ব্যবহৃত হয়। যথা-

প্রথম নিয়ম: #include<Filename>

দ্বিতীয় নিয়ম: #include “Filename”

#include ডিরেক্টিভ স্টেটমেন্টে ফাইলের নাম এংগেল বন্ধনীর (<>) মধ্যে এবং ডাবল কোটেশনের (“ ”) মধ্যে লেখার পার্থক্য:

#include ডিরেক্টিভ স্টেটমেন্টে ফাইলের নাম এংগেল বন্ধনীর(<>) মধ্যে লেখা হলে প্রোগ্রাম কম্পাইলেশনের সময় কম্পাইলার ঐ ফাইলকে প্রথমে Option মেনুর Directories সাবমেনুতে নির্দেশিত হেডার ডিরেক্টরিতে খোজ করে। যদি সেখানে তা পাওয়া যায় তবে প্রোগ্রামে তা ব্যবহার করে, অন্যথায় চলতি (কারেন্ট) ডিরেক্টরিতে খোজ করে; যেখানে প্রথম পাওয়া যায় সেখানে ব্যবহার করে। কিন্তু দ্বিতীয় নিয়মে, অর্থাৎ যখন ফাইলের নাম ডাবল কোটেশনের (“ ”) মধ্যে লেখা হয়, তখন কম্পাইলার ঐ ফাইলকে কেবল চলতি ডিরেক্টরিতে খোজ করে।

লাইব্রেরি ফাংশন:

সি কম্পাইলারে কতগুলো বিল্ট-ইন ফাংশন আছে সেগুলোকে লাইব্রেরি ফাংশন বলা হয়। লাইব্রেরি ফাংশনগুলো তাদের নিজস্ব ফরমেট অনুযায়ী main() ফাংশনের মধ্যে ব্যবহার করা হয়। printf(), scanf() ইত্যাদি বহুল ব্যবহৃত লাইব্রেরি ফাংশন।

নিম্নে কিছু লাইবেরি ফাংশন হেডার ফাইল সহ দেওয়া হলোঃ

হেডার ফাইল	ফাংশন এর নাম	ফাংশনের বিবরণ
stdio.h	scanf()	কীবোর্ড হতে যেকোনো ধরনের ডেটা ইনপুট দেওয়ার জন্য ব্যবহৃত হয়।
	printf()	যেকোনো ধরনের ডেটা আউটপুট পাওয়ার জন্য ব্যবহৃত হয়।
	getchar()	কীবোর্ড হতে একই সময়ে শুধু একটি ক্যারেক্টার ডেটা ইনপুট দেওয়ার জন্য ব্যবহৃত হয়।
	putchar()	একই সময়ে শুধু একটি ক্যারেক্টার আউটপুট পাওয়ার জন্য ব্যবহৃত হয়।
	gets()	কীবোর্ড হতে স্ট্রিং ডেটা ইনপুট দেওয়ার জন্য ব্যবহৃত হয়।
conio.h	puts()	স্ট্রিং আউটপুট পাওয়ার জন্য ব্যবহৃত হয়।
	clrscr()	মনিটরের পর্দা পরিস্কার করার জন্য ব্যবহৃত হয়।
	getch()	একটি ক্যারেক্টার ইনপুট দেওয়া যায় কিন্তু তা মনিটরে দেখায় না।

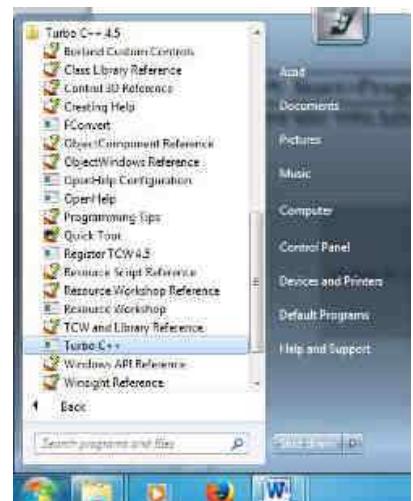
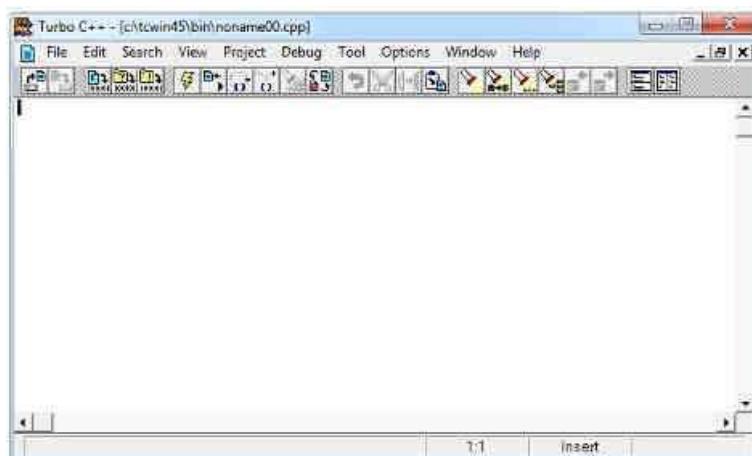
সি ভাষায় প্রোগ্রাম শুরু করার পদ্ধতি:

বেশিরভাগ সি কম্পাইলারে নিজস্ব সোর্স এডিটার আছে, যেখানে সহজে ও সুন্দরভাবে সি সোর্স ফাইল তেরি করা যায় এবং স্বয়ংক্রিয়ভাবে কিছু সুবিধা পাওয়া যায়। এরূপ একটি কম্পাইলার হচ্ছে টার্বো সি++৪.৫ (Turbo C++ 4.5)।

টার্বো সি++৪.৫ (Turbo C++ 4.5) খোলার কমান্ড নিম্নরূপ:

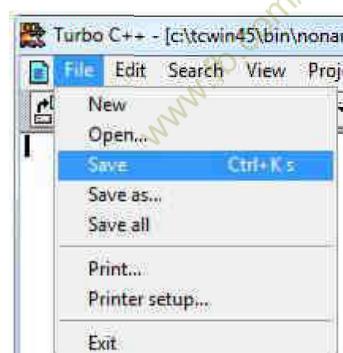
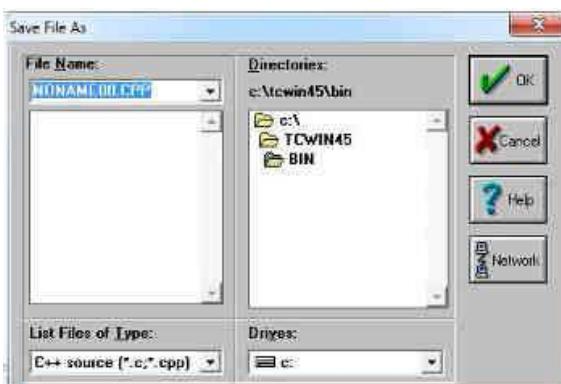
Start>Programs/All Programs>Turbo C++4.5> Turbo C++4.5

তাহলে Turbo C++ ৪.৫ এর এডিটর উইンドো দেখতে পাবো।



সি প্রোগ্রামে ফাইল সেভ বা সংরক্ষণ:

নতুন সোর্স ফাইল সেভ বা সংরক্ষণ করার জন্য File মেনু থেকে Save অবৰ্থা Save as সাবমেনু নির্বাচন করতে হয়।
তাহলে Save File As ডায়লগ বক্স প্রদর্শিত হয়।



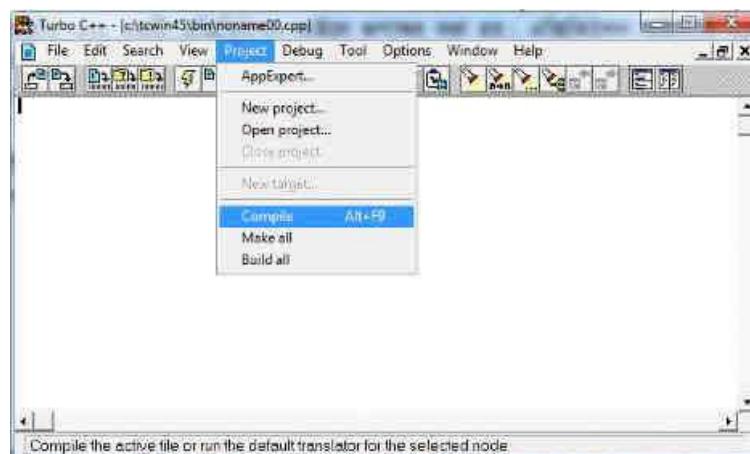
এই ডায়লগ বক্সের File Name এর টেক্সট বক্সে ফাইলের নাম লিখে এন্টার কী চাপতে হয়। পুর্বে সংরক্ষণ করা কোনো সোর্স ফাইল ভিন্ন নামে সংরক্ষণ করার জন্য Save as সাবমেনু নির্বাচন করতে হয়।

প্রোগ্রাম কম্পাইলিং এর গুরুত্ব (Importance of Program compiling)

কম্পাইলার সম্পর্কে আমরা আগেই জেনেছি। C ভাষায় লিখিত কোনো প্রোগ্রামকে সোর্স কোড বা সোর্স প্রোগ্রাম বলা হয়। সোর্স প্রোগ্রামকে কম্পাইলারের সাহায্যে একসাথে সম্পূর্ণরূপে মেশিন ভাষায় অনুবাদ করে একটি অবজেক্ট প্রোগ্রাম এবং একটি এক্সিকিউশন ফাইল রূপান্তর করা হয়। এক্সিকিউশন ফাইলটি হচ্ছে মূল ফাইল যার দ্বারা প্রোগ্রাম রান (Run) করানো হয়। আর এই প্রক্রিয়াকে বলে কম্পাইলিং। প্রোগ্রাম কম্পাইলিং এর মাধ্যমে প্রোগ্রামকে ত্রুটিমুক্ত করা হয়। সি ভাষাকে কম্পাইল করতে বিভিন্ন ধরনের কম্পাইলারের প্রচলন রয়েছে। যেমন- Turbo C/C++, ANSI C, Borland C/C++ ইত্যাদি।

সি প্রোগ্রাম কম্পাইল করার করার ধাপসমূহ:

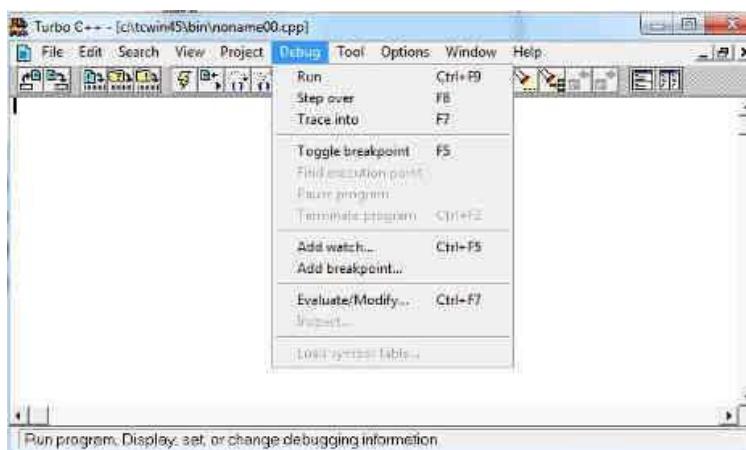
প্রোগ্রামটিকে কম্পাইল করার জন্য প্রথমে Project মেনুতে ক্লিক করতে হবে।



অতঃপর এর অধীনে Compile অপশনটি সিলেক্ট করতে হবে অবশ্য (Alt+F9) কী-দ্বয় চাপতে হবে। প্রোগ্রামটি কম্পাইল করার পর যদি এতে কোনো ভুল বা Error থাকে তাহলে তা ঠিক করতে হবে।

প্রোগ্রাম নির্বাহ করার পদ্ধতি:

প্রোগ্রামটিকে নির্বাহ করার জন্য প্রথমে Debug মেনুতে ক্লিক করতে হবে।



অতঃপর এর অধীনে Run অপশনটি সিলেক্ট করতে হবে অবশ্য (Ctrl+F9) কী-দ্বয় চাপতে হবে। প্রোগ্রামে কোনো ভুল না থাকলে প্রোগ্রামটি নির্বাহ হবে।



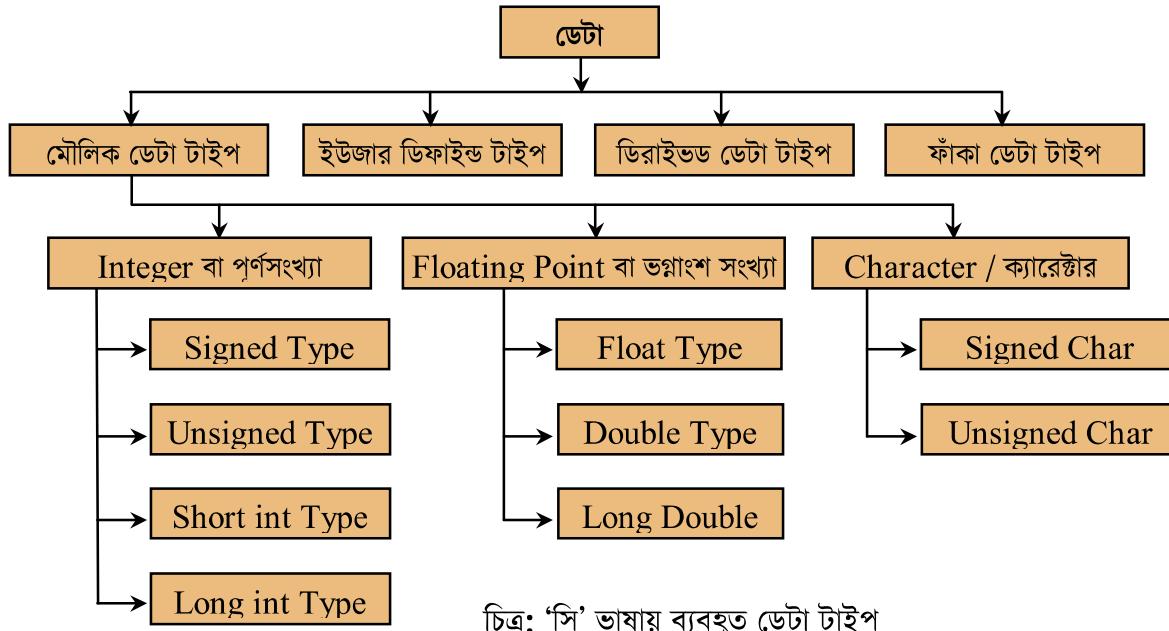
কাজ: সি প্রোগ্রামিং ভাষা কোন প্রোগ্রাম ডিজাইন মডেলের অন্তর্ভুক্ত? উক্ত মডেলের অন্তর্ভুক্তির কারণগুলো লেখ।

পার্ট-১৫ ও ১৬

‘সি’ ভাষায় ব্যবহৃত ডেটা টাইপ (Data Type Uses in C Language)

ডেটা টাইপ

‘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-char, int, float, double। এদেরকে বেসিক বা মৌলিক অথবা বিল্টইন ডেটা টাইপ বলা হয়। আবার প্রয়োজনে 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।



char টাইপ: সি প্রোগ্রামে ক্যারেক্টার টাইপ ডেটা নিয়ে কাজ করার জন্য char টাইপ ভেরিয়েবল ব্যবহার করা হয়। char টাইপ ভেরিয়েবল ঘোষণার জন্য char কিওয়ার্ড ব্যবহার করা হয়। প্রতিটি char টাইপ ভেরিয়েবলের জন্য কম্পাইলার ১ বাইট জায়গা সংরক্ষণ করে। অর্থাৎ প্রোগ্রামে char ch = 78; স্টেটমেন্টের মাধ্যমে ch নামে char টাইপের কোনো ভেরিয়েবল ঘোষণা করলে কম্পাইলার প্রোগ্রাম নির্বাচকালে মেমোরিতে ch নামে এক বাইট জায়গা বরাদ্দ করে স্থানে ৭৮ সংরক্ষণ করবে।

char টাইপ ডেটার জন্য সিনট্যাক্স (Syntax) হলো-

char <variable name>;

char ch = 'a';

Example: a, b, g, S, j

int টাইপ: সি প্রোগ্রামে পূর্ণসংখ্যা (যেমন, ২০,-৮৬৭, ৮৯০) ইত্যাদি নিয়ে কাজ করার জন্য int টাইপ ভেরিয়েবল ব্যবহার করা হয়। int টাইপ ভেরিয়েবল ঘোষণার জন্য int কিওয়ার্ড ব্যবহার করা হয়। প্রতিটি int টাইপ ভেরিয়েবলের জন্য কম্পাইলার ২ বাইট জায়গা সংরক্ষণ করে। অর্থাৎ প্রোগ্রামে int value = 90; স্টেটমেন্টের মাধ্যমে value নামে int টাইপের কোনো ভেরিয়েবল ঘোষণা করলে কম্পাইলার প্রোগ্রাম নির্বাচকালে মেমোরিতে value নামে দুই বাইট জায়গা বরাদ্দ করে স্থানে ৯০ সংরক্ষণ করবে।

int টাইপ ডেটার জন্য সিনট্যাক্স (Syntax) হলো-

int <variable name>;

int num1;

short int num2;

long int num3;

Example: 5, 6, 100, 2500

ডেটা টাইপ মডিফায়ার বা টাইপ কোয়ালিয়ার (Data type Modifier Or Type Qualifier)

ভিন্ন ভিন্ন ডেটা টাইপের প্রয়োজনে একই ভেরিয়েবলের জন্য সংরক্ষিত মেমরি পরিসর আলাদা হয়। যে সকল কীওয়ার্ড ব্যবহার করে ফ্লোট ছাড়া অন্যান্য মৌলিক বা প্রাথমিক ডেটা টাইপের ব্যাণ্ডি বা পরিসর এবং সংরক্ষণের জন্য মেমরি পরিমাণ বাঢ়ানো বা কমানো যায় এদেরকে ডেটা টাইপ মডিফায়ার বলে। সহজ কথায় বলা যায়, সি ভাষায় ব্যবহৃত মৌলিক ডেটা টাইপগুলোর জন্য সংরক্ষিত মেমরি 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 মডিফায়ার এবং float ও 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 মেমরি 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 - $2^{১৫}$ থেকে +(২ ^{১৫} - ১)
	Unsigned int	১৬	০ থেকে ৬৫,৩৩৬ বা ০ - ০ থেকে +(২ ^{১৬} - ১)
	Signed long int	৩২	-২১৪,৭৪,৮৩,৬৪৮ থেকে +২১৪,৭৪,৮৩,৬৪৭ বা - $2^{৩১}$ থেকে +(২ ^{৩১} - ১)
	Unsigned long int	৩২	০ থেকে ৪২৯,৪৯,৬৭,২৯৫ বা ০ - ০ থেকে +(২ ^{৩২} - ১)
ফ্লোটিং পয়েন্ট বা ভগ্নাংশ	Float	৩২	$3.8 \times E-৩৮$ থেকে $3.8 \times E+৩৮$
	Double	৬৪	$১.৭ \times E-৩০৮$ থেকে $১.৭ \times E+৩০৮$
	Long double	৮০	$৩.৪ \times E-৪৯৩২$ থেকে $৩.৪ \times E+৪৯৩২$
ক্যারেক্টার	Char or signed char	৮	-১২৮ থেকে +১২৭ বা - $2^৭$ থেকে +(২ ^৭ - ১)
	Unsigned char	৮	০ থেকে ২৫৫ বা ০ থেকে (২ ^৮ - ১)

চিত্র: 'সি' ডেটা টাইপ

float টাইপ: সি প্রোগ্রামে রিয়েল বা ভগ্নাংশসহ কোনো সংখ্যা (যেমন, ২০.৩৪, -৪৬.৮৭, ৮৯.৭০) ইত্যাদি নিয়ে কাজ করার জন্য float টাইপ ভেরিয়েবল ব্যবহার করা হয়। float টাইপ ভেরিয়েবল ঘোষণার জন্য float কিওয়ার্ড ব্যবহার করা হয়। প্রতিটি float টাইপ ভেরিয়েবলের জন্য কম্পাইলার ৪ বাইট বা ৩২ বিট জায়গা সংরক্ষণ করে। অর্থাৎ প্রোগ্রামে float value; স্টেটমেন্টের মাধ্যমে value নামে float টাইপের কোনো ভেরিয়েবল ঘোষণা করলে কম্পাইলার প্রোগ্রাম নির্বাচকালে মেমোরিতে value নামে ৪ বাইট বা ৩২ বিট জায়গা সংরক্ষণ করবে।

float টাইপ ডেটার জন্য সিনট্যাক্স (Syntax) হলো-

float <variable name>;

float num1;

double num2;

long double num3;

Example: 9.125, 3.1254

double টাইপ: সি প্রোগ্রামে float টাইপ ভেরিয়েবলের মতো রিয়েল বা ভগ্নাংশবিশিষ্ট সংখ্যা (যেমন, ২০,-৪৬৭, ৮৯০) ইত্যাদি নিয়ে কাজ করার জন্য double টাইপ ভেরিয়েবল ঘোষণা করা হয়, তবে float টাইপ ভেরিয়েবলের চেয়ে double টাইপ ভেরিয়েবলের রেঞ্জ বেশি। double টাইপ ভেরিয়েবল ঘোষণার জন্য double কিওয়ার্ড ব্যবহার করা হয়। প্রতিটি double টাইপ ভেরিয়েবলের জন্য কম্পাইলার মেমোরিতে ৮ বাইট জায়গা সংরক্ষণ করে। অর্থাৎ প্রোগ্রামে double value; স্টেটমেন্টের মাধ্যমে value নামে double টাইপের কোনো ভেরিয়েবল ঘোষণা করলে কম্পাইলার প্রোগ্রাম নির্বাচকালে মেমোরিতে value নামে ৮ বাইট জায়গা সংরক্ষণ করবে।

 কাজ: ১.

তোমাদের পিতার নাম, বয়স ও মাসিক আয়ের জন্য কোন কোন ডেটা টাইপ ব্যবহৃত হবে তা ছকের মাধ্যমে দেখাও।

পাঠ-১৭, ১৮ ও ১৯

‘সি’ ভাষায় ব্যবহৃত চলক ও ধূবক (Variable and Constant in C Language)

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-

- চলক ও
- ধূবক

চলক (Variable)

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

একটি ভেরিয়েবলের নিম্নলিখিত বৈশিষ্ট্য থাকতে হবে-

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

চলক ঘোষণার সিনটেক্স বা ফর্ম্যাট হলো-

Datatype VariableName; উদাহরণ: int number1;

অথবা Datatype VariableName=[value]; উদাহরণ: int number1=60;

ভেরিয়েবল ডিক্লেরেশনের স্থান:

ভেরিয়েবল মূলত তিনটি স্থানে ডিক্লেয়ার করা যায়। যথা-

- ফাংশনের মধ্যে
- ফাংশন প্যারামিটারে
- সমস্ত ফাংশনের বাইরে।

ভেরিয়েবল ব্যবহারের সুবিধা (Advantages of Variable)

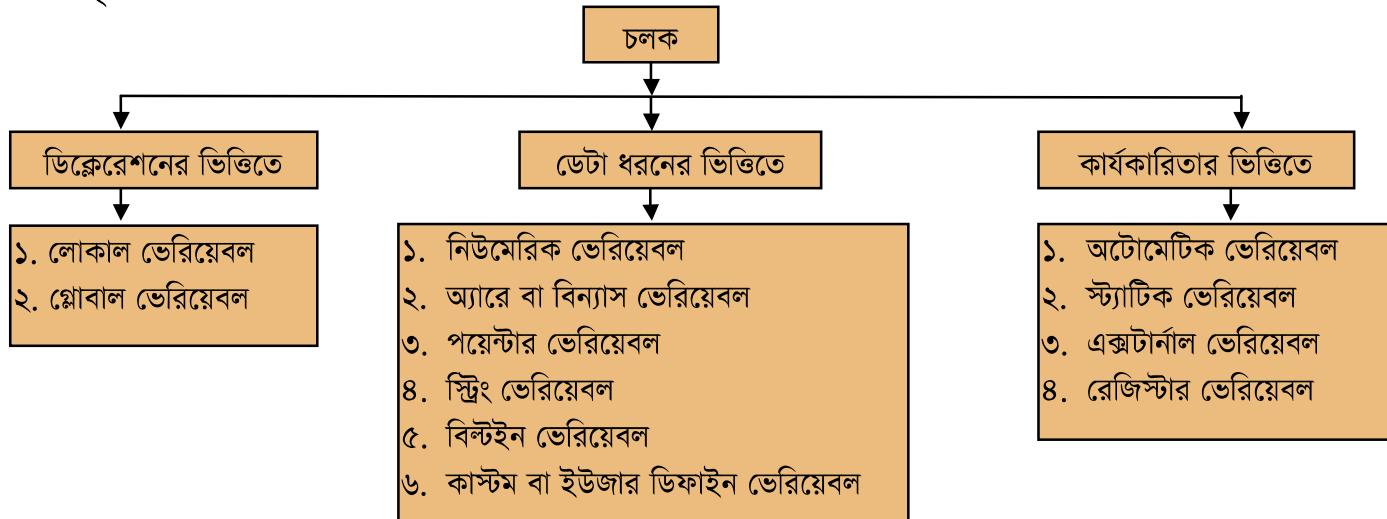
ভেরিয়েবল ব্যবহার না করেও প্রোগ্রামে বিভিন্ন ধরনের ডেটা নিয়ে কাজ করা যায়। তবে সেক্ষেত্রে ডেটার স্বয়ংক্রিয় মান নির্ধারণ, পুনব্যবহার প্রভৃতি সুবিধা পাওয়া যায় না। উদাহরণ হিসেবে বলা যায়, কারো যদি লক্ষ লক্ষ বন্ধু থাকে, তবে যতই আন্তরিক হোক না কেন তারা কে কোন রূমে থাকে তা মনে রাখা সম্ভবপর নয়। কিন্তু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

ভেরিয়েবলের স্কেপ বা ক্ষেত্র

ফাংশনে ব্যবহৃত ভেরিয়েবলগুলো ঘোষণার স্থান, প্রকৃতি, ডেটা টাইপ প্রভৃতি নিয়ামকের ওপর তাদের কার্যক্রম, বিস্তৃতি, স্থিতিশীলতা ইত্যাদি নির্ভর করে। প্রোগ্রামের যে অংশের জন্য কোনো ভেরিয়েবলের কার্যক্রম বিস্তৃত সেই অংশকে এই ভেরিয়েবলের স্কেপ বা ক্ষেত্র বলা হয়।

ভেরিয়েবল এর 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 কোন ফাংশনের মধ্যে অর্থাৎ ফাংশন বড়িতে 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 করা যায়। 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 মেমরি পরিসর খালি করে দেয়। ফলে দুই বা ততোধিক ফাংশনে একই নাম এবং ডাটা টাইপের লোকাল ভেরিয়েবল ব্যবহার করা যেতে পারে এবং তাতে কোন সমস্যা হয় না। প্রয়োজন লোকাল ভেরিয়েবলের কার্যক্রম ফাংশনের একটি নির্দিষ্ট রূপের মধ্যেও সীমাবদ্ধ করে দেয়া যায়।
- **ঘোবাল ভেরিয়েবল (Global Variable):** যখন কোন ভেরিয়েবল প্রোগ্রামের শুরুতে `main()` ফাংশনের পূর্বে ঘোষণা করা হয় তখন তাকে ঘোবাল ভেরিয়েবল বলা হয়। ঘোবাল ভেরিয়েবলের মান ও অস্তিত্ব কোনো নির্দিষ্ট রূপ 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 ঘোবাল ভেরিয়েবল এর মধ্যে পার্থক্য নিম্নরূপ:

পার্থক্যের বিষয়	লোকাল ভেরিয়েবল	ঘোবাল ভেরিয়েবল
১. সংজ্ঞা	১. কোনো ফাংশনের মধ্যে ভেরিয়েবল ডিক্লেয়ার করলে তাকে উক্ত ফাংশনের লোকাল ভেরিয়েবল বলা হয়।	১. সকল ফাংশনের বাহিরে প্রোগ্রামের শুরুতে ডিক্লেয়ার করা ভেরিয়েবলকে ঘোবাল ভেরিয়েবল বলা হয়।
২. সীমাবদ্ধতা	২. কোনো ফাংশনের মধ্যে ডিক্লেয়ার করা লোকাল ভেরিয়েবল উক্ত ফাংশনের বাইরে ব্যবহার করা যায় না।	২. ঘোবাল ভেরিয়েবলের কর্মকাণ্ড কোনো ফাংশনের মধ্যে সীমাবদ্ধ নয়।
৩. একই নাম	৩. ভিন্ন ফাংশনে একই নামের লোকাল ভেরিয়েবল থাকতে পারে।	৩. একটি প্রোগ্রামের মধ্যে একই নামের একটি মাত্র ঘোবাল ভেরিয়েবল থাকতে পারে।
৪. ডিক্লেয়ার স্থান	৪. ফাংশনের শুরুতে ডিক্লেয়ার করা হয়।	৪. সাধারণত প্রোগ্রামের শুরুতে ডিক্লেয়ার করা হয়।

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

1. সংখ্যাসূচক বা নিউমেরিক ভেরিয়েবল
 2. বিন্যাস বা অ্যারে ভেরিয়েবল
 3. নির্দেশক বা পয়েন্টার ভেরিয়েবল
 4. ব্যবহারকারী বর্ণিত বা কাস্টম ভেরিয়েবল
 5. স্ট্রিং বা অক্ষরমালা ভেরিয়েবল
 6. বিল্ট-ইন ভেরিয়েবল
- **নিউমেরিক ভেরিয়েবল:** যে চলক বা 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 কোন 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, 3.4×10^{20} অর্থাৎ 3.5E-203 ইত্যাদি) হতে পারে। প্রোগ্রামে এ ধরনের ভেরিয়েবলের ব্যবহারই অধিক।
 - **অ্যারে ভেরিয়েবল:** একই ধরনের কতগুলো ভেরিয়েবলের সমষ্টিকে অ্যারে ভেরিয়েবল বা বিন্যাস চলক বলা হয়। অ্যারে আবার একমাত্রিক, দ্বিমাত্রিক ও বহুমাত্রিক হতে পারে। যেমন: একমাত্রিক অ্যারের উদাহরণ হলো: A [2, 3] যা মোট 10টি ভেরিয়েবলের সমষ্টি নির্দেশ করে। আবার দ্বিমাত্রিক অ্যারের উদাহরণ হলো: A [2, 3] যা মোট ভেরিয়েবলের সমষ্টি নির্দেশ করে।
 - **পয়েন্টার ভেরিয়েবল:** পয়েন্টার এক প্রকার ভেরিয়েবল যা একই টাইপের অপর কোন ভেরিয়েবলকে নির্দেশ করে; অর্থাৎ একই টাইপের অপর কোনো ভেরিয়েবলের মেমরি এ্যড্রেস ধারণ করে। পয়েন্টার ভেরিয়েবল ব্যবহারের ফলে প্রোগ্রামের জটিলতা অনেকাংশে হ্রাস পায় এবং প্রোগ্রাম নির্বাহে অপেক্ষাকৃত কম সময় লাগে।
 - **কাস্টম ভেরিয়েবল:** অনেক সময় প্রোগ্রামের জটিলতা হ্রাস করার জন্য প্রোগ্রামার তার প্রয়োজনীয় বিভিন্ন টাইপ ভেরিয়েবলের সমন্বয়ে নিজস্ব ডেটা টাইপ তৈরি করে নেন এবং প্রোগ্রামের প্রয়োজনীয় স্থানে সেই টাইপের প্রয়োজনীয় সংখ্যক ভেরিয়েবল ঘোষণা ও ব্যবহার করেন। প্রোগ্রামার তথা প্রোগ্রাম ভাষা ব্যবহারকারী কর্তৃক তৈরি এরূপ ডেটা টাইপকে ইউজার- ডিফাইন্ড বা কাস্টম ডেটা টাইপ এবং এই টাইপ ভেরিয়েবলকে কাস্টম ভেরিয়েবল বলা হয়। সমস্যার ধরন অনুযায়ী নিজস্ব ডেটা টাইপ ভেরিয়েবল ঘোষণার মাধ্যমে বাস্তবতার খুব কাছাকাছি পৌঁছে অতি সহজে সমস্যার সমাধান করা যায়। এতে প্রোগ্রামের জটিলতা অনেকাংশে হ্রাস পায়। স্ট্রাকচার, ইউনিয়ন ও ইনুমারেশন সি-তে বহুল ব্যবহৃত কয়েকটি কাস্টম ডেটা টাইপ।
 - **স্ট্রিং ভেরিয়েবল:** যখন এক বা একাধিক ক্যারেক্টার বা বর্ণ দ্বিতীয় বর্ণনার মধ্যে আবদ্ধ করা হয় তখন তাকে স্ট্রিং বা অক্ষরমালা বলা হয়। যেমন, "Computer", "Programming in C", "University of Dhaka" ইত্যাদি। সি-তে স্ট্রিং ভেরিয়েবল ঘোষণার জন্য আলাদা কোন ডেটা টাইপ নেই, 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"
 - **বিল্ট ইন ভেরিয়েবল:** বিল্ট ইন ভেরিয়েবল হলো এমন একটি ভেরিয়েবল যা কতগুলো লাইব্রেরি ফাংশন কিংবা ডেটা আইটেমের মান সংরক্ষণে ব্যবহৃত হয়।
- কার্যকারিতার উপর নির্ভর করে ভেরিয়েবলকে চার ভাগে ভাগ করা যায়। যথা—**
- **অটোমেটিক ভেরিয়েবল:** প্রোগ্রামে বিল্ট-ইন এবং মোডিফাইড ডেটা টাইপের যেসকল লোকাল ভেরিয়েবল ঘোষণা করা হয় সেগুলো ট্রানজিয়েন্ট বা ক্ষণস্থায়ী প্রকৃতির। এসব ভেরিয়েবলের জন্য প্রোগ্রাম নির্বাহকালে কম্পাইলার প্রয়োজনীয় মেমরি পরিসর বরাদ্দ করে ফাংশন নির্বাহ শেষে এমনকি ঐ ভেরিয়েবলের ক্ষেত্রে অতিক্রমকালে স্বয়ংক্রিয়ভাবে বরাদ্দকৃত মেমরি পরিসর খালি করে দেয়। এরূপ ভেরিয়েবলকে অটোমেটিক ভেরিয়েবল বলা হয়। অটোমেটিক ভেরিয়েবল ঘোষণার জন্য ভেরিয়েবলের ডেটা টাইপের পূর্বে auto ব্যবহৃত হয়। এরূপ ঘোষণার ফরম্যাট হলো :

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

তবে কোনো লোকাল ভেরিয়েবল ঘোষণাকালে তার ডেটা টাইপের পূর্বে auto কীওয়ার্ড উল্লেখ না করলেও কম্পাইলার স্বয়ংক্রিয়ভাবে ভেরিয়েবল হিসেবে গণ্য করে। সুতরাং উপরের ঘোষণা নিচের ঘোষণার সমরূপ:

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

অর্থাৎ সি প্রোগ্রামে যে সকল লোকাল ভেরিয়েবল ব্যবহার করা হয় সেগুলোর সবই অটোমেটিক ভেরিয়েবল। অটোমেটিক ভেরিয়েবল বিশিষ্ট কোন ফাংশন কল করলে ঐ ভেরিয়েবলের জন্য মেমরিতে প্রয়োজনীয় পরিসর বরাদ্দ হয় এবং ফাংশন নির্বাহ শেষে স্বয়ংক্রিয়ভাবে বরাদ্দকৃত মেমরি পরিসর খালি হয়। ফলে দুই বা ততোধিক ফাংশনে বিভিন্ন মানবিশিষ্ট একই নামেও একই ডেটা টাইপের ভেরিয়েবল ব্যবহার করা যায়।

- **স্ট্যাটিক ভেরিয়েবল:** নিজস্ব ফাংশন এবং ব্যবহারকারী ফাংশনসহ পুরো প্রোগ্রামে কোনো ভেরিয়েবলের অর্জিত সর্বশেষ মান ব্যবহার করার জন্য তাকে স্ট্যাটিক ভেরিয়েবল হিসেবে ঘোষণা করা হয়। সি প্রোগ্রামে অটোমেটিক ভেরিয়েবল বিশিষ্ট কোন ফাংশন কল করা হলে প্রতিবার ফাংশন কলের জন্য ভেরিয়েবল গুলো 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।

এরূপ ঘোষণার ফরম্যাট হলো :

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

- **এক্সট্রান্যাল 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 মেমরিতে না রেখে দুটগতির রেজিস্টারে রাখা হয়। ফলে ডেটা প্রসেস সহজ হয়। এধরনের চলক ঘোষণা করতে register কী-ওয়ার্ড ব্যবহৃত হয়। যেমন - register int x;

ভেরিয়েবল ব্যবহারের বা ঘোষণার বা লেখার নিয়মাবলী

প্রোগ্রামার প্রোগ্রাম রচনার শুরুতে প্রয়োজনীয় সংখ্যক ভেরিয়েবল ঘোষণা করেন এবং প্রোগ্রামের পরবর্তী অংশে সেগুলো ব্যবহার করেন। সুতরাং তিনি তার ইচ্ছা অনুযায়ী ভেরিয়েবলের নামকরণ করতে পারেন না, কারণ ভেরিয়েবল ঘোষণা এবং নামকরণের মধ্যে কিছু মৌলিক সীমবদ্ধতা ও নিয়ম-কানুন রয়েছে। যেমন, অনেক সময় এক প্রোগ্রামারের রচিত প্রোগ্রাম অন্য প্রোগ্রামার কর্তৃক পরিবর্তন বা পরিবর্ধনের প্রয়োজন হতে পারে। সেক্ষেত্রে প্রোগ্রামে কোনো ভেরিয়েবল কোন কাজে ব্যবহৃত হয়েছে তা বুঝতে অসুবিধা হতে পারে। 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 কোন ভেরিয়েবলের নামকরণ প্রক্রিয়াকে ভেরিয়েবল ঘোষণা বলা হয়।

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

- ভেরিয়েবলের প্রথম অক্ষর অবশ্যই আলফা-বেটিক ক্যারেক্টার (a, ..., z, A, ..., Z) হবে। ভেরিয়েবল নাম জিডিট বা অংক দিয়ে শুরু হতে পারে না। যেমন- Roll_1 ও Roll_10 বৈধ ভেরিয়েবল; কিন্তু 1Roll 2_Roll অবৈধ।
- ভেরিয়েবলের মধ্যে স্পেশাল ক্যারেক্টার আভারস্কেপ্ট () ও ডলার চিহ্ন (\$) ব্যবহার করা যায়। আভারস্কেপ্ট ও ডলার চিহ্ন ব্যতীত অন্য কোন স্পেশাল ক্যারেক্টার (যেমন !,@,#,%,*,+,- ইত্যাদি) ব্যবহার করা যায় না। যেমন, my_var, My\$Roll 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 কোন ভেরিয়েবল ঘোষণা করা যেতে পারে।
- ভেরিয়েবল নামের মধ্যে কোন ফাঁকা স্থান থাকতে পারে না। যেমন, RollNo, Roll, MyRoll ইত্যাদি বৈধ ভেরিয়েবল। কিন্তু Roll N ও Roll 1, My Roll অবৈধ।

- সি প্রোগ্রামে বড় হাতের এবং 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।
- কোন কীওয়ার্ডের নাম ভেরিয়েবল হিসেবে ব্যবহার করা যায় না এবং main কোন কীওয়ার্ড না হলেও ভেরিয়েবল নাম হিসেবে main ব্যবহার করা যায় না। অবশ্য কীওয়ার্ড-সমূহের নামের এক বা একাধিক বর্ণ বড় হরফে লিখে আইডেন্টিফায়ারের নাম হিসেবে ব্যবহার করা যায়। তবে এরূপ না করাই উচ্চম। যেমন, int, char man Main Main ইত্যাদি বৈধ ভেরিয়েবল। কিন্তু int, main ইত্যাদি অবৈধ।
- ভেরিয়েবল নামকরণে যে কোন সংখ্যক ক্যারেক্টার ব্যবহার করা যায়। তবে ANST নিয়ম অনুযায়ী ভেরিয়েবল নামকরণে ৩১টি ক্যারেক্টারের বেশি ব্যবহার না করাই ভাল।

ভেরিয়েবলের মান নির্ধারণের ক্ষেত্রে সাধারণত ভুল

ভেরিয়েবলের মান নির্ধারণের ক্ষেত্রে সাধারণত ভেরিয়েবলের ডেটা টাইপ এবং 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।

ধূরক (Constant)

কনস্ট্যান্ট অর্থ স্থির বা ধূরক যা একটি নির্দিষ্ট মান ধারণ করে। অনেক সময় প্রোগ্রামে একটি স্থির বা অপরিবর্তনশীল মান ব্যবহৃত হয়। সেক্ষেত্রে প্রোগ্রামে ঐ মানকে কনস্ট্যান্ট হিসেবে ঘোষণা করা হয়। প্রোগ্রাম নির্বাচের সময় কোন অবস্থাতেই কনস্ট্যান্ট বা ধূরকের মান পরিবর্তন করা যায় না, অর্থাৎ 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:

- ১। Const কীওয়ার্ড ব্যবহার করে
- ২। #define প্রিপ্রসেসর ব্যবহার করে।

const কীওয়ার্ড ব্যবহার ধূরক ঘোষণার ফরম্যাট হলো:

```
const ConstType ConstName = ConstValue;
```

এখানে ConstType বলতে একটি মৌলিক বা মডিফাইড ডেটা টাইপ (যেমন, char, int, float, double, long ইত্যাদি) বুঝায়। তবে ConstType উল্লেখ না থাকলে কম্পাইলার তাকে int টাইপ হিসেবে ধরে নেয়। আর ConstName হলো প্রোগ্রামারের দেয়া কোন বৈধ নাম। উল্লেখ্য, এরূপ ঘোষণার জন্য কনস্ট্যান্ট-এর প্রারম্ভিক মান এবং শেষে সেমিকোলন দেয়া আবশ্যিক।

```
const int Max = 50;
const char Ch = 'a';
```

ধূরক ঘোষণার জন্য সিনটেক্স (Syntax) হলো-

```
int main()
{
    const float PI = 3.14;
    char = 'A';
    return 0;
}
```

#define প্রিপ্রসেসর ব্যবহার করে ধূরক ঘোষণার ফরম্যাট হলো:

```
#define ConstName ConstValue
```

যেমন:

#define	Max	50
#define	TRUE	1
#define	FALSE	0
#define	PI	3.141592

এরূপে ঘোষণার জন্য কেবল কনস্ট্যান্টের নাম ও প্রারম্ভিক মান দিতে হয়। তবে কোনো টাইপ উল্লেখ করতে হয় না এবং মাঝে সমান চিহ্ন ও শেষে সেমিকোলন বসে না।

কনস্ট্যান্ট প্রধানত দুই ধরনের, যথা :

১. সংখ্যাসূচক ধূবক বা নিউমেরিক কনস্ট্যান্ট: এই ধরনের ধূবক ০ থেকে 9 পর্যন্ত অংক বা \$ দ্বারা শুরু হয় এবং মান - 2147483648 থেকে 294967295 এর মধ্যবর্তী যে কোন ঝগাইক বা ধণাইক হতে পারে। ধূবকে কমা ব্যবহার করা যায় না; তবে প্রয়োজনে দশমিক চিহ্ন ব্যবহার করা যায়। সংখ্যাসূচক ধূবক আবার নিম্নোক্ত পাঁচভাগে ভাগ করা যায় :

- ইন্টিজার কনস্ট্যান্ট : এ ধরনের ধূবক ধনাইক বা ঝগাইক যে কোনো পূর্ণসংখ্যা হতে পারে। এ ধূবকে দশমিক বিন্দু থাকে না। যেমন- 546, 45, 20000, -32768, +6532767 ইত্যাদি।
 - ফ্লোটিং পয়েন্ট কনস্ট্যান্ট : এ ধরনের ধূবক ধনাইক বা ঝগাইক পূর্ণ বা ডগাংশবিশিষ্ট সংখ্যা পারে। পূর্ণসংখ্যার জন্য দশমিক চিহ্নের ব্যবহার আবশ্যিক নয়, তবে আংশিক মানের জন্য এর বিকল্প নেই। যেমন- 56, 56, 0, 55, 50, 3.141592, -45.678, +65.32767 ইত্যাদি।
 - এক্সপোনেনশিয়াল কনস্ট্যান্ট : এ ধরনের ধূবকও ধনাইক এবং ঝগাইক উভয় ধরনের হতে পারে। এ রকম সংখ্যা 10 এর সূচক বা ঘাত (Power) হিসাবে লেখা হয় এবং E অক্ষর দিয়ে বোঝান হয়। এখানে E দিয়ে 10 এর সাথে যে সূচক সংখ্যাটি থাকে তার মান লেখা হয়। সূচক সংখ্যাটি যদি ধনাইক হয় তাহলে E এর পরে যোগ (+) এবং ঝগাইক হলে বিয়োগ (-) ব্যবহার করা যায়। যেমন, 3.5E+3, 3.5E-5, ইত্যাদি।
 - অষ্টাল 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 ধূবক বা স্ট্রিং কনস্ট্যান্ট: বর্ণ, অঙ্ক এবং অন্যান্য চিহ্ন সাজিয়ে এই ধূবক গঠিত হয়। এই ধূবককে সিঙ্গেল অথবা ডাবল কোটেশন দ্বারা নির্দিষ্ট করা হয়। স্ট্রিং কনস্ট্যান্টে 0 হতে 225 টি অক্ষর থাকতে পারে। যেমন, "A", "Dhaka", "Bangladesh" ইত্যাদি।

কনস্ট্যান্ট ব্যবহারের নিয়ম: কনস্ট্যান্ট ব্যবহারের ক্ষেত্রে 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 স্পেসিফিকার ব্যবহৃত হয়, ইত্যাদি।

প্রোগ্রামে ধূবক ব্যবহারের সুবিধা (Advantages of Constant)

নিম্ন প্রোগ্রামে ধূবক ব্যবহারের কয়েকটি সুবিধা দেওয়া হলো -

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

সিল্যাংগুয়েজে কনস্ট্যান্ট ও ভেরিয়্যাবল এর মধ্যে পার্থক্য নিম্নরূপ:

কনস্ট্যান্ট	ভেরিয়্যাবল
১. কনস্ট্যান্ট অর্থ স্থির বা ধূবক যা একটি নির্দিষ্ট মান ধারণ করে। প্রোগ্রামে কোনো স্থির বা অপরিবর্তনশীল মান ব্যবহার করার জন্য তা কনস্ট্যান্ট হিসেবে ঘোষণা করা হয়।	১. ভেরিয়্যাবল হলো একটা নাম, যে নামে কম্পাইলার নির্দিষ্ট ধরনের ডেটা রাখার জন্য মেমোরিতে জায়গা রাখে।
২. কনস্ট্যান্ট কমা ব্যবহার করা যায় না তবে প্রয়োজনে দশমিক ব্যবহার করা যায়।	২. ভেরিয়্যাবলের মান নির্ধারণ করার সময় সংখ্যার মধ্যে কমা ব্যবহার করা যাবে।
৩. প্রোগ্রাম চালানোর সময় কোনভাবেই কনস্ট্যান্ট এর মান পরিবর্তন করা যায় না।	৩. প্রোগ্রাম চালানোর সময় যখন প্রয়োজন ইচ্ছেমত ভেরিয়্যাবল এর মান পরিবর্তন করা যায়।



- কাজ:**
১. প্রোগ্রামে চলক ব্যবহারের গুরুত্ব লেখ।
 ২. চলক ও ধূবক প্রোগ্রামে কীভাবে ব্যবহার করতে হয় উদাহরণ দিয়ে দেখাও।
 ৩. নিচের চলকগুলো কেন অবৈধ তা ব্যাখ্যা কর।
 - i. int 9x ii. int "id" iii. int main iv. float marks 50 v. char fa-name;

পাঠ-২০

রাশিমালা (Expression)

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

উদাহরণ হিসেবে বলা যায়, $\text{Average} = (\text{value1} + \text{value2}) / 2$; একটি এক্সপ্রেশন। এখানে Average , value1 , value2 অপারয়ান্ড; $=$, $-$, $+$, $/$ অপারেটর এবং 2 কনস্ট্যান্ট।

সি-তে ব্যবহৃত অপারেটরসমূহ:

অপারেটরের সাথে সংযুক্ত অপারয়ান্ড বা কনস্ট্যান্টের সংখ্যার ভিত্তিতে সি প্রোগ্রামে ব্যবহৃত অপারেটর সমূহকে তিনটি প্রধান শ্রেণিতে ভাগ করা হয়। যথাঃ

- ইউনারি অপারেটর
- বাইনারি অপারেটর এবং
- টারনারী অপারেটর

ইউনারি অপারেটরঃ যে সকল অপারেটরের সাথে কেবল একটি করে অপারয়ান্ড বা কনস্ট্যান্ট সংযুক্ত থাকে তাদেরকে ইউনারি অপারেটর বলা হয়। নিম্নে একটি ছকে বহুল ব্যবহৃত কয়েকটি ইউনারি অপারেটর এবং তাদের ব্যবহার উল্লেখ করা হলো।

অপারেটর	উদাহরণ	ব্যবহার
+ (ইউনারি প্লাস)	$v2 = +v1;$ $v3 = +(v1 - v2);$	অপারয়ান্ডের ধনাত্মক মান বৃক্ষাতে ব্যবহৃত হয়।
- (ইউনারি মাইনাস)	$v2 = -v1;$ $v3 = -(v1 + v2);$	অপারয়ান্ডে ঋণাত্মক মান বৃক্ষাতে ব্যবহৃত হয়।
++ (ইনক্রিমেন্টাল অপারেটর)	Counter ++; ++ Counter ;	অপারয়ান্ডের মানের সাথে ১ যোগ হয়।
-- (ডিক্রিমেন্টাল অপারেটর)	Counter --; -- Counter ;	অপারয়ান্ডের মান হতে ১ বিয়োগ হয়।
! (নট)	!not	শূণ্য বাদে অন্য কোনো অপারয়ান্ডের মান 0 করে দেয় কিন্তু শূণ্যের মান 1 করে দেয়।
~ (পূরক বা কমপ্লিমেন্ট)	~complement	অপারয়ান্ডের মান 1'এর পরিপূরকে রূপান্তর করে তা আবার দশমিকে রূপান্তর করে দেখায়।

ইউনারি অপারেটরগুলো পোষ্টফিল্ড বা প্রিফিল্ড নোটেশনে কাজ করে। পোষ্টফিল্ড নোটেশন মানে হলো অপারেটরটা অপারয়ান্ডের পরে বসে। অন্যদিকে প্রিফিল্ড নোটেশন মানে হলো অপারেটরটা অপারয়ান্ডের পূর্বে বসে। কোন কোন অপারেটরগুলো পোষ্টফিল্ড, প্রিফিল্ড বা উভয় নোটেশনে কাজ করে তা নিচে দেওয়া হলো।

অপারেটর	নোটেশন	অপারেটর	নোটেশন	অপারেটর	নোটেশন
+	প্রিফিল্ড	++	উভয়	!	প্রিফিল্ড
-	প্রিফিল্ড	--	উভয়	~	প্রিফিল্ড

বাইনারি অপারেটর: যে সকল অপারেটরের সাথে দুইটি করে অপারয়েন্ট বা কনস্ট্যান্ট সংযুক্ত থাকে তাদেরকে বাইনারি অপারেটর বলা হয়। যেমন, সি প্রোগ্রামে ইউনারি অপারেটর অপেক্ষা বাইনারি অপারেটরের আধিক্য দেখা যায়। উপরের তালিকায় দেখা ইউনারি আপারেটরগুলো এবং নীচের টারনারি অপারেটরটা ছাড়া বাকী সব অপারেটরগুলোই বাইনারি অপারেটর। বাইনারি অপারেটর ইনফিক্স (infix) নোটেশনে কাজ করে। অর্থাৎ অপারেটরগুলো দুটো অপারয়েন্টের মাঝখানে ব্যবহৃত হয়।

টারনারি অপারেটর: যে সকল অপারেটর এক সাথে তিনটি অপারয়েন্ট 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।

যেমনঃ

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

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

গাণিতিক অপারেটর		লজিক্যাল অপারেটর	
+	যোগ করার জন্য	&&	লজিক্যাল AND
-	বিয়োগ করার জন্য		লজিক্যাল OR
*	গুণ করার জন্য	!	লজিক্যাল NOT
/	ভাগ করার জন্য		
%	ভাগশেষ নির্ণয়ের জন্য		

অ্যাসাইনমেন্ট অপারেটর		রিলেশনাল অপারেটর	
$a = a + 1$	$a += 1$	==	সমান কি না
$a = a - 1$	$a -= 1$!=	সমান নয়
$a = a * (n+1)$	$a *= n=1$	<	ছোটো তুলনা করতে
$a = a / (n+1)$	$a /= n=1$	>	বড় তুলনা করতে
$a = a \% b$	$a \% = b$	<=	ছোটো বা সমান তুলনা করতে
		>=	বড় বা সমান তুলনা করতে

বিটওয়াইজ অপারেটর		বিশেষ অপারেটর	
&	বিটওয়াইজ AND	()	ফাংশন কল
	বিটওয়াইজ OR	[]	অ্যারে ইনডেক্স ধোষণা
^	বিটওয়াইজ XOR	(,)	কমা
<<	শিফট লেফট	(&, *)	পয়েন্টার
>>	শিফট রাইট	(. এবং ->)	মেম্বার সিলেকশন
~	১'এর পরিপূরক		

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 ছোটো হাতের অক্ষরে লিখতে হয়। নিচের টেবিলে C 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

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 এর ভাষায় লেখ।

$$1. ax^2 + bx + c = 0$$

$$2. \frac{ax^2}{bx} + C = 0$$

পাঠ-২১, ২২ ও ২৩

ব্যবহারিক: ইনপুট / আউটপুট স্টেটমেন্ট (Input / Output Statement)

প্রোগ্রামের মাধ্যমে কম্পিউটারকে কোনো তথ্য দেয়ার জন্য ডেটা সরবরাহ করতে হয়। C প্রোগ্রামে তিনটি পদ্ধতিতে ইনপুট দেয়ার ব্যবস্থা আছে। যথা— ১. অ্যাসাইনমেন্ট স্ট্যাটমেন্ট, ২. এক বা একাধিক বর্ণ পড়া (getchar কমান্ড, gets কমান্ড) ৩. ফরমেটেড ইনপুট (scanf কমান্ড)

- অ্যাসাইনমেন্ট স্টেটমেন্ট:** ডেটার মান পরিবর্তন না হলে সরাসরি একটি চলকের মাধ্যমে কোনো ডেটাকে প্রকাশ করা যায়। যেমন: $a = 25; b = 5;$
- এক বা একাধিক বর্ণ পড়া:** একটি বর্ণ পড়ার জন্য C প্রোগ্রামে getchar () কমান্ড ব্যবহৃত হয়। একাধিক বর্ণ পড়ার জন্য C প্রোগ্রামে gets () কমান্ড ব্যবহৃত হয়।

যে সমস্ত ফাংশন ব্যবহার করে একটি ক্যারেক্টার ইনপুট দেওয়া যায় তা নিচে দেওয়া হলো-

ফাংশন	সিনটেক্স	বৈশিষ্ট্য
scanf()	scanf("%c",&c)	একটি ক্যারেক্টার ইনপুট দেওয়া যায় এবং তা মনিটরে দেখায়
getch()	c=getch()	একটি ক্যারেক্টার ইনপুট দেওয়া যায় কিন্তু তা মনিটরে দেখায় না
getche()	c=getche()	একটি ক্যারেক্টার ইনপুট দেওয়া যায় এবং তা মনিটরে দেখায়
getchar()	c=getchar()	একটি ক্যারেক্টার ইনপুট দেওয়া যায় এবং তা মনিটরে দেখায়

যে সমস্ত ফাংশন ব্যবহার করে স্ট্রিং ইনপুট দেওয়া যায় তা নিচে দেওয়া হলো-

ফাংশন	সিনটেক্স	বৈশিষ্ট্য
scanf()	scanf("%s",&ch)	scanf("%s",&ch) কোনো ফাঁকা স্পেস অনুমোদন করে না।
scanf()	scanf("%[^n]",ch)	scanf("%[^n]",ch) ফাঁকা স্পেস অনুমোদন করে।
gets()	gets(ch)	gets(ch) ফাঁকা স্পেস অনুমোদন করে।

এক্ষেত্রে gets ফাংশন এবং scanf() ফাংশনের মধ্যে কিছু পার্থক্য আছে। gets ফাংশনের মাধ্যমে একই সময়ে শুধু মাত্র একটি স্ট্রিং (ফাঁকা স্থান সহ) ইনপুট দেওয়া যায়। একই সময়ে যত ক্যারেক্টারই ইনপুট দেওয়া হোক না কেন gets ফাংশন তাকে একটি স্ট্রিং হিসেবে বিবেচনা করে। অপরদিকে scanf() ফাংশন একই সময়ে একাধিক স্ট্রিং ইনপুটের অনুমতি প্রদান করে। এখানে একই লাইনের প্রতিটি ফাঁকা স্থান আলাদা আলাদা ডেটা আইটেম নির্দেশ করে।

যদি আমরা তিনটি স্ট্রিং ইনপুট করতে চাই তাহলে,

gets ফাংশন এর জন্য :

```
gets(n1);
gets(n2);
gets(n3);
```

scanf() ফাংশন এর জন্য :

```
scanf("%s %s %s",&n1,&n2,&n3);
```

যদি এবারে My College, Dhaka ইনপুট দেওয়া হয় তাহলে, gets ফাংশন এর জন্য n1 এর ইনপুট মান ধরবে My College, Dhaka এবং n2 এবং n3 আলাদা মান ইনপুট দিতে হবে। কিন্তু scanf() ফাংশন এর জন্য n1 এর মান ধরবে My এবং n2-এর মান ধরবে College, এবং এই সমস্যা দূর করার জন্য নিম্নের সিনটেক্সটি ব্যবহার করতে হবে।

```
scanf("%[^n]",ch);
```

একটি ফাংশন ব্যবহার করে তিনটি স্ট্রিং ইনপুট দেওয়ার জন্য নিম্নরূপ কোড লিখতে হবে।

```
scanf("%[^n] %[^n] %[^n] ",n1,n2, n3)
```

৩. ফরমেটেড ইনপুট (scanf কমান্ড): অপারেটর বা ব্যবহারকারীর নিকট থেকে প্রোগ্রাম চলাকালীন যেকোনো ধরনের ডেটা ইনপুট নিতে হলে C প্রোগ্রামে scanf() কমান্ড ব্যবহৃত হয়। এ ডেটা চলক হিসেবে মেমরিতে সংরক্ষিত থাকে। এক্ষেত্রে বিভিন্ন ধরনের ডেটার জন্য চলকের নামের পূর্বে কিছু কোড যুক্ত করতে হয়। scanf() কমান্ড ব্যবহার করে কোনো ভেরিয়েবলের মান নির্ধারণের ফরমেট হলো—

```
scanf("F_S", &variablename);
```

এখানে F_S হলো ফরমেট স্পেসিফিয়ার যা ভেরিয়েবল এর আউটপুট অপারেশনে ব্যবহৃত ফরমেট স্পেসিফিয়ারের অনুরূপ। variablename হলো পূর্ব ঘোষিত কোনো ভেরিয়েবল এর নাম এবং & হলো অ্যাড্রেস অপারেটর যা ভেরিয়েবল এর জন্য গৃহীত মান মেমরির ঐ লোকেশনে সংরক্ষণ করে। এড্রেস অপারেটর(&) ব্যবহার না করলে গৃহীত মান মেমরিতে ভেরিয়েবলের সঠিক লোকেশনে প্রেরিত হয় না বিধায় প্রোগ্রাম নির্বাহে সঠিক ফলাফল নাও আসতে পারে। আর এজন্য কম্পাইলার কোন বার্তা প্রদর্শন নাও করতে পারে।

আউটপুট স্টেটমেন্ট (Output statement)

কম্পিউটারের মনিটরে কোনো প্রোগ্রামের ফলাফল দেখার জন্য যে সকল স্টেটমেন্ট ব্যবহার করা হয়, সেগুলো হচ্ছে আউটপুট স্টেটমেন্ট। C প্রোগ্রামে দু'টি পদ্ধতিতে ফলাফল প্রদর্শনের ব্যবস্থা আছে।

১. এক বা একাধিক বর্ণ লেখা: একটি বর্ণ লেখার জন্য C প্রোগ্রামে putchar() কমান্ড ব্যবহৃত হয়। একাধিক বর্ণ লেখার জন্য C প্রোগ্রামে puts() কমান্ড ব্যবহৃত হয়।

যে সমস্ত ফাংশন ব্যবহার করে একটি ক্যারেক্টার আউটপুট পাওয়া যায় তা নিচে দেওয়া হলো-

ফাংশন	সিনটেক্স	বৈশিষ্ট্য
printf()	printf("%c",a)	একটি ক্যারেক্টার মনিটরে দেখায়
putc()	put(a)	একটি ক্যারেক্টার মনিটরে দেখায়
putchar()	putchar(a)	একটি ক্যারেক্টার মনিটরে দেখায়

যে সমস্ত ফাংশন ব্যবহার করে স্ট্রিং আউটপুট পাওয়া যায় তা নিচে দেওয়া হলো—

ফাংশন	সিনটেক্স	বৈশিষ্ট্য
printf()	printf("%s",ch)	ফাঁকা স্পেস স্ট্রিং মনিটরে প্রদর্শন করে
puts()	puts(ch)	ফাঁকা স্পেস স্ট্রিং মনিটরে প্রদর্শন করে

এক্ষেত্রে puts ফাংশন এবং printf() ফাংশনের মধ্যে কিছু পার্থক্য আছে। puts() ফাংশনের মাধ্যমে একই সময়ে শুধু মাত্র একটি স্ট্রিং (ফাঁকা স্থান সহ) আউটপুট পাওয়া যায়। অপরদিকে printf() ফাংশন একই সময়ে একাধিক স্ট্রিং আউটপুটের অনুমতি প্রদান করে।

puts ফাংশন এর জন্য :

```
puts (n1);
puts (n2);
puts (n3);
```

printf() ফাংশন এর জন্য :

```
printf("%s %s %s", n1, n2, n3);
```

২. ফরমেটেড আউটপুট (printf() কমান্ড): বিভিন্ন ডেটা টাইপের পরিবর্তনশীল ডেটা ফলাফল হিসেবে প্রদর্শন করতে printf() কমান্ড ব্যবহৃত হয়।

সি প্রোগ্রামে ফরম্যাটেড (কাঞ্চিত আকারে) ভেরিয়েবলের মান গ্রহণ এবং প্রদর্শনের জন্য সে সকল ক্যারেক্টার সেট বা ক্যারেক্টারগুচ্ছ ব্যবহৃত হয় তাদেরকে ফরম্যাট স্পেসিফিয়ার বলা হয়।

নিচের টেবিলে কোনো ডেটার জন্য চলকের পূর্বে কোন কোড ব্যবহার করতে হবে, তা দেখানো হলো:

ফরমেট 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 নোটেশনে ইনপুট / আউটপুট করার জন্য	scanf("%e",&data); printf("%e",data);
%E	float টাইপ মান এক্সপোনেনসিয়াল E নোটেশনে 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 নোটেশনে ইনপুট / আউটপুট করার জন্য	scanf("%g",&data); printf("%g",data);
%G	float টাইপ মান %f অথবা %E নোটেশনে ইনপুট / আউটপুট করার জন্য	scanf("%G",&data); printf("%G",data);
%hd	Read a short integer value	scanf("%hd",&data); printf("%hd",data);
%o	অষ্টাল 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);
%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);

বিশেষ কিছু ক্যারেক্টার আছে (যেমন, /, ", \n, \r, \t ইত্যাদি) যেগুলো printf() ফাংশনের ডাবল কোটেশনের ("")
মধ্যে যেভাবে ব্যবহার করা হয় ফলাফলে সেরূপ প্রদর্শিত হয় না। printf() বা এরূপ কোন ফাংশন দ্বারা এসব
ক্যারেক্টার প্রদর্শনের জন্য এই ক্যারেক্টারগুলোর 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	আউটপুট নিচের লাইনে প্রদর্শনের জন্য	printf("This is \n Dhaka College");	This is Dhaka College
\t	আউটপুট পূর্ববর্তী লাইনের শুরুতে একই কলাম বরাবর প্রদর্শনের জন্য	printf("This is \t Dhaka College");	This is Dhaka College
\r	আউটপুট পরবর্তী (নতুন) লাইনের শুরুতে প্রদর্শনের জন্য	printf("This is \r Dhaka College");	Dhaka College
		printf("\t\tThis is \r Dhaka College");	Dhaka College This is
\a	সংকেত (Alarm) দানের জন্য	printf("\a\This is Dhaka College");	you will hear an alarm
\b	আউটপুট পেছনে (বামে) একঘর সরানোর জন্য	printf("This is \b Dhaka College");	This is Dhaka College
\``	ডাবল কোটেশন (') ক্যারেক্টার প্রদর্শনের জন্য	printf("This is \`` Dhaka College");	This is ``Dhaka College
\`	ডাবল 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
<hr/>			
\?	প্রশ্নবোধক চিহ্ন প্রদর্শনের জন্য	printf("This is \? Dhaka College");	This is ? Dhaka College

ছক : কয়েকটি ব্যাকস্লাশ ক্যারেক্টার ও তাদের ব্যবহার



কাজ: ১. getch() ও getchar() এর মধ্যে পার্থক্য দেখাও।

২. scanf("%s" & ch) ও scanf("%[^\\n]", ch) এর মধ্যে পার্থক্য দেখাও।

৩. scanf() ও gets এর মধ্যে পার্থক্য দেখাও।

৪. printf() ও puts এর মধ্যে পার্থক্য দেখাও।

পাঠ-২৪

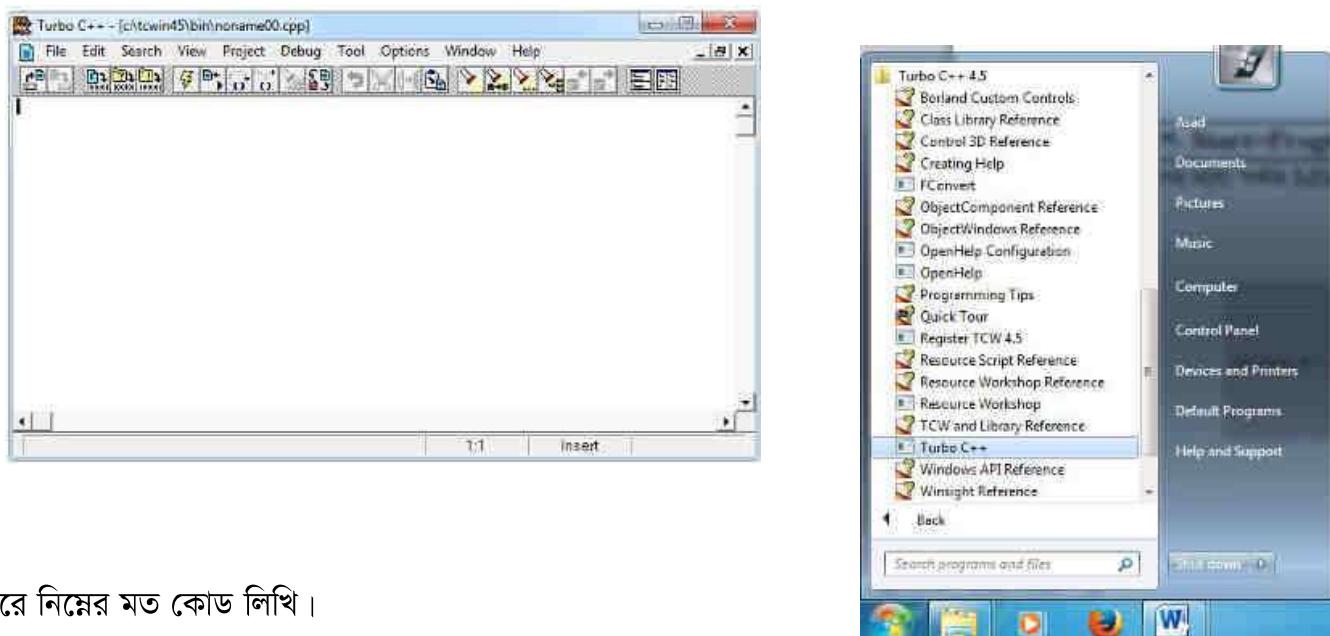
ব্যবহারিক: ব্যবহারিক নির্দেশাবলী ও কিছু প্রোগ্রাম প্র্যাকটিস

এ পর্যন্ত আমরা শুধুমাত্র সি ভাষার বিভিন্ন নিয়মাবলী শিখেছি। এখন শিখব কিভাবে প্রোগ্রাম কোড লেখা যায় এবং তা কম্পাইল ও নির্বাহ করা যায়।

প্রথমেই আমরা এমন একটি প্রোগ্রাম কোড লিখতে চাই যার আউটপুটে শুধু নিজের নাম দেখাবে। এইজন্য প্রথমে আমরা টার্বো সি++৪.৫ (Turbo C++ 4.5) প্রোগ্রামটি খুলি। প্রোগ্রামটি খোলার ক্ষমতা নিম্নরূপ:

Start>Programs/All Programs>Turbo C++4.5> Turbo C++4.5

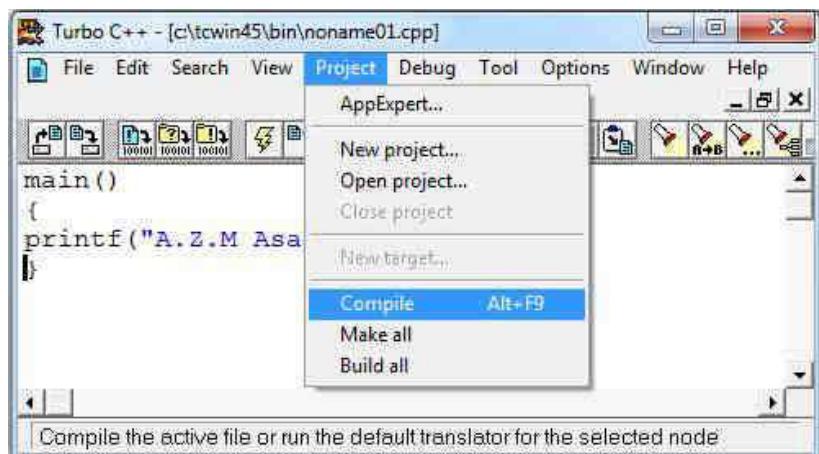
তাহলে Turbo C++ ৪.৫ এর এডিটর উইন্ডো দেখতে পাবো।



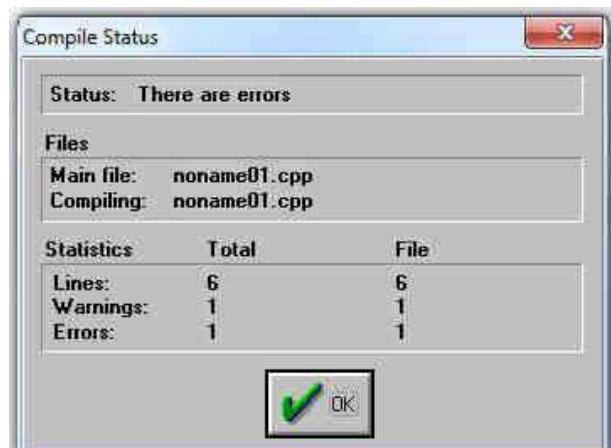
এবারে নিম্নের মত কোড লিখি।

```
main()
{
printf("A.Z.M Asaduzzaman");
}
```

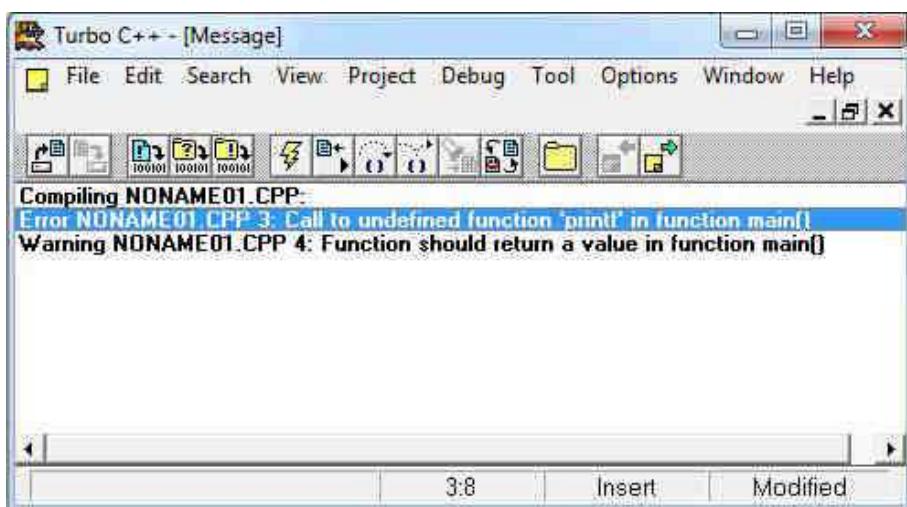
এবারে Project মেনুর Compile এর উপর ক্লিক করে অথবা Alt+F9 চেপে কম্পাইল করি।



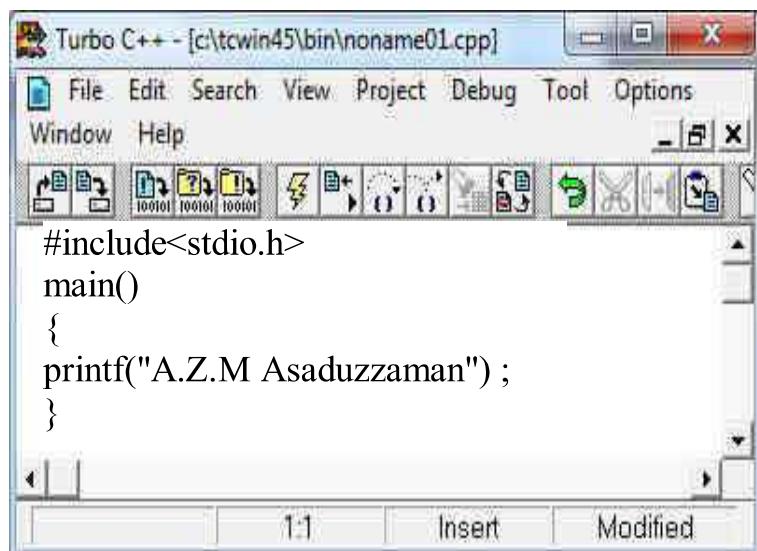
তাহলে নিম্নের মত ইরর মেসেজ আসবে।



এবারে OK এর উপর ক্লিক করি তাহলে নিম্নের মত ইরর দেখাবে।

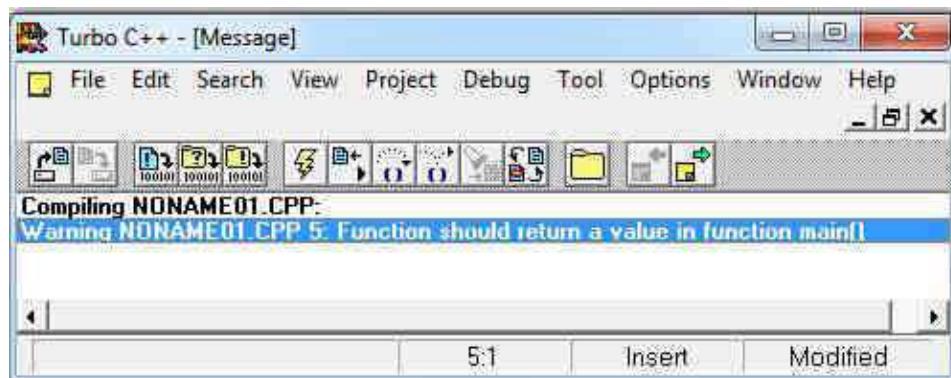


এখানে প্রথম ইরর মেসেজ দ্বারা বোঝাচ্ছে printf() ফাংশন undefined। কারণ printf() ফাংশনের হেডার ফাইল main() ফাংশনের উপর লিঙ্ক করা হয়নি। তাই প্রোগ্রাম কোডে কিছু পরিবর্তন করতে হবে অর্থাৎ main() ফাংশনের উপরে printf() ফাংশনের হেডার ফাইল সংযুক্ত করতে হবে। তাহলে প্রোগ্রাম কোড হবে নিম্নরূপ:

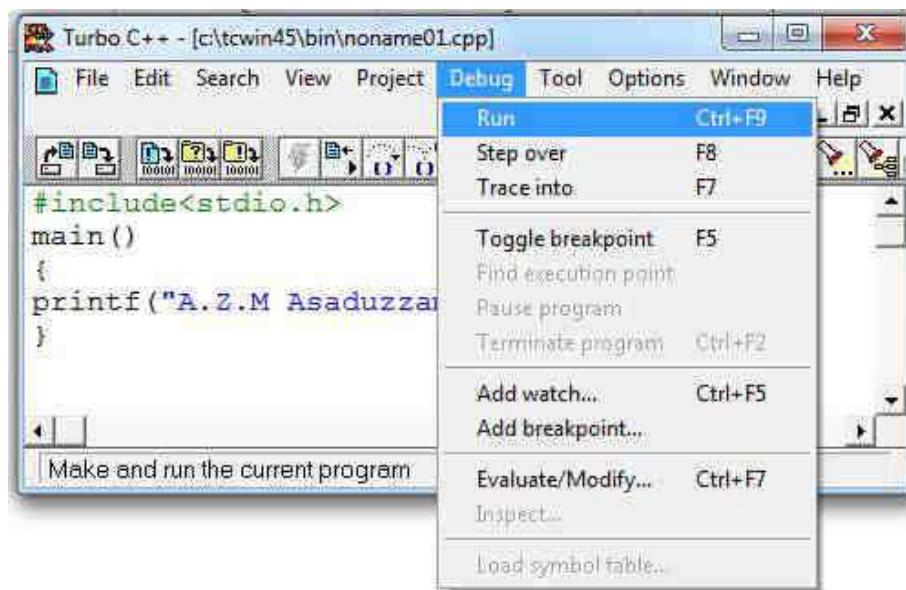


```
#include<stdio.h>
main()
{
printf("A.Z.M Asaduzzaman");
}
```

এবারে কম্পাইল তাহলে নিম্নের মত মেসেজ দেখাবে।



উক্ত সতর্কতা মেসেজ দেখালেও প্রোগ্রাম নির্বাহ হবে। এবারে প্রথমে আমরা প্রোগ্রামটি নির্বাহ করি পরে উক্ত সমস্যা দূর করা যাবে। প্রোগ্রামটিকে নির্বাহ করার জন্য প্রথমে Debug মেনুতে ক্লিক করতে হবে। অতঃপর এর অধীনে Run অপশনটি সিলেক্ট করতে হবে অথবা (Ctrl+F9) কী-দ্বয় চাপতে হবে। তাহলে নিম্নের মত আউটপুট দেখাবে।



The screenshot shows the Turbo C++ IDE interface. The main window displays the following C code:

```
#include<stdio.h>
main()
{
    printf("A.Z.M Asaduzzaman");
}
```

Below the code editor, the status bar shows "Program running" and the current time as 3:29. A separate window titled "(Inactive C:\TCWIN45\BIN\NONAME01.EXE)" shows the output of the program: "A.Z.M Asaduzzaman".

উক্ত সমস্যা থেকে দুই ভাবে সমাধান করা যায়। প্রথমত main() ফাংশনের আগে void কীওয়াডটি ব্যবহার করে। তাহলে প্রোগ্রামটি হবে নিম্নরূপ:

The screenshot shows the Turbo C++ IDE interface with the following modified C code:

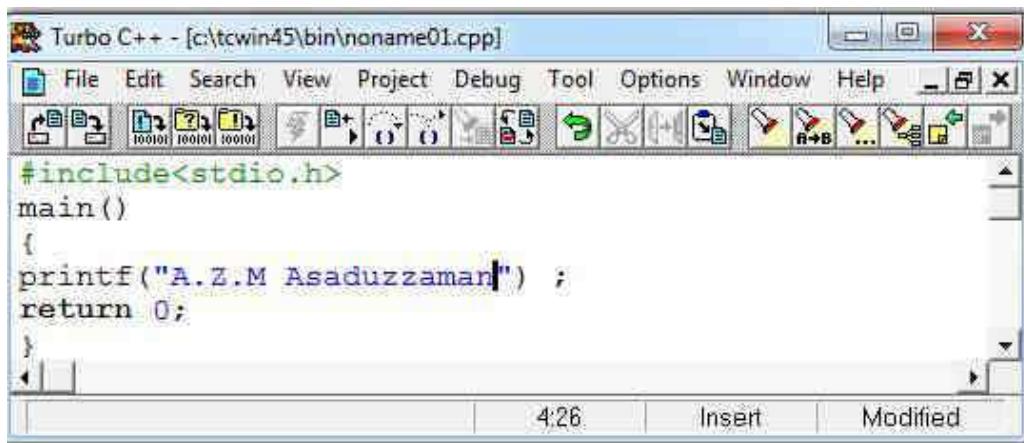
```
#include<stdio.h>
void main()
{
    printf("A.Z.M Asaduzzaman");
}
```

এবারে প্রোগ্রামটি কম্পাইল করি। কম্পাইল কমান্ড দিলে নিচের মেসেজ বক্সটি আসবে।



লক্ষ করলে দেখতে পাচ্ছি এখানে কোনো সতর্কতা বা ইরর নাই।

দ্বিতীয়ত উপায়টি হলো সবার শেষে return 0; স্টেমেন্টটি ব্যবহার করা। তাতে প্রোগ্রামটি হবে নিম্নরূপ:



```
#include<stdio.h>
main()
{
    printf("A.Z.M Asaduzzaman");
    return 0;
}
```

এবারে প্রোগ্রামটি কম্পাইল করি। কম্পাইল করান্ত দিলে নিচের মেসেজ বক্সটি আসবে।



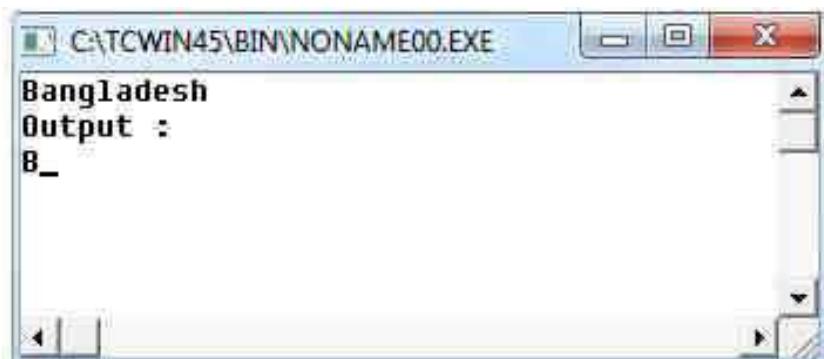
লক্ষ্য করলে দেখতে পাচ্ছ এখানে কোনো সতর্কতা বা ইরর নাই।

সুতরাং আমরা শিখে ফেলেছি কিভাবে প্রোগ্রাম কোড লিখতে হয় এবং কিভাবে প্রোগ্রাম কম্পাইল ও নির্বাহ করতে হয়।
এবারে আমরা কিছু ক্যারেক্টার বা স্ট্রিং নিয়ে প্রোগ্রাম দেখব।

উদাহরণ-১. শুধুমাত্র একটি ক্যারেক্টার ইনপুট দিয়ে তা আউটপুট দেখানোর জন্য প্রোগ্রাম [`getchar()` ও `putchar()` ফাংশন এর ব্যবহার]

```
#include<stdio.h>

#include<conio.h>
main()
{
    int c;
    c=getchar();
    printf("Output :\n");
    putchar(c);
    getch();
}
```



```
C:\TCWIN45\BIN\NONAME00.EXE
Bangladesh
Output :
B_
```

প্রোগ্রাম কম্পাইল: Alt+F9 কী-ব্রে একত্রে চেপে প্রোগ্রাম কম্পাইল করি।

প্রোগ্রাম নির্বাহ করার পদ্ধতি: Ctrl+F কী-ব্রে একত্রে চেপে প্রোগ্রাম নির্বাহ করি। ফলে পাশের চিত্রের মত একটি খালি বা ফাঁকা আউটপুট স্ক্রিন আসবে। সেখানে এক বা একাধিক লাইন টাইপ করা যাবে। টাইপ করা শেষে Enter press করলে আউটপুট হিসেবে শুধুমাত্র প্রথম অক্ষরটি দেখাবে। যেমন : খালি বা ফাঁকা আউটপুট স্ক্রিন আসবে সেখানে যদি টাইপ করা হয় Bangladesh

এবারে Enter চাপলে আউটপুট হিসেবে শুধুমাত্র প্রথম অক্ষরটি অর্থাৎ B দেখাবে এবং দেখতে নিম্নের মত দেখাবে।

উদাহরণ-২. একাধিক শব্দবিশিষ্ট স্ট্রিং এর ইনপুট অপারেশনে gets() ফাংশন এবং আউটপুট অপারেশনে puts() ফাংশন ব্যবহার করে একটি প্রোগ্রাম দেয়া হলো:

```
#include<stdio.h>
#include<conio.h>
void main()
{
    char a[25];
    gets(a);
    puts(a);
    getch();
}
```

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

এই প্রোগ্রামটি রান করলে একটি খালি বা ফাঁকা আউটপুট স্ক্রিন আসবে সেখানে এক বা একাধিক লাইন টাইপ করা যাবে। যেমন: I Love my Bangladesh. টাইপ করা শেষে Enter press করলে মনিটরে নিম্নের আউটপুটটি দেখাবে।

যেমন :

I Love my Bangladesh ↵

I Love my Bangladesh

উপরোক্ত প্রোগ্রামটিকে scanf() এবং printf() ব্যবহার নিম্নোক্ত ভাবেও লেখা যায়।

```
#include<stdio.h>
int main()
{
    char name[30];
    printf("Enter your name:");
    scanf("%[^\\n]",name);
    printf("%s Thank you",name);
    getch();
    return 0:
}
```

এই প্রোগ্রামে gets() ফাংশনের কাজটি scanf(" %[\\n]",data); printf("%s",data); ব্যবহার করা হয়েছে।

ফলাফল: Enter your name: Adnan Jaami ↵
Adnan Jaami Thank you

উদাহরণ-৩. ছোট হাতের অক্ষরকে বড় হাতের অক্ষরে
রূপান্তর।

```
#include<stdio.h>
#include<ctype.h>
main()
{
char x;
printf("Enter a small letter: ");
x=getchar();
printf("Capital Letter: ");
putchar(toupper(x));
}
```

ফলাফল:

Enter a small letter:a ↴
Capital Letter:A

উদাহরণ-৪. বড় হাতের অক্ষরকে ছোট হাতের অক্ষরে
রূপান্তর।

```
#include<stdio.h>
#include<ctype.h>
main()
{
char x;
printf("Enter a Capital character: ");
x=getchar();
printf("Small Letter: ");
putchar(tolower(x));
}
```

ফলাফল: Enter a Capital character: A ↴

Small Letter:a

**উক্ত প্রোগ্রামসময়ে toupper(), tolower() ফাংশনসময় ব্যবহার করা হয়েছে যাদের হেডার ফাইল ctype.h। তাই
প্রোগ্রামের main() ফাংশনের উপরে #include<ctype.h> লিখে হেডার ফাইল যুক্ত করা হয়েছে।**

উক্ত প্রোগ্রামসময়কে আমরা অন্যভাবেও লিখতে পারি। প্রোগ্রামটি লেখার আগে আমরা অ্যাসকি (ASCII) কোড সম্পর্কে একটু মনে করে নিই। অ্যাসকি কোডে A এর দশমিক মান হচ্ছে 65 অন্য দিকে a দশমিক মান হচ্ছে 97। ফলে দেখা যাচ্ছে উভয়েরই দশমিক মানের পার্থক্য হচ্ছে 32। এভাবে লক্ষ করলে দেখা যাবে বড় হাতের (Capital Letter) প্রতিটি অক্ষরের দশমিক মানের সাথে ছোট হাতের (Small Letter) প্রতিটি অক্ষরের দশমিক মানের অন্তর 32।

সুতরাং বড় হাতের প্রতিটি অক্ষরের সাথে 32 যোগ করলে অক্ষরটি ছোট হাতের অক্ষরে পরিণত হবে। আর ছোট হাতের অক্ষরের হতে 32 বিয়োগ করলে অক্ষরটি বড় হাতের অক্ষরে পরিণত হবে।

উদাহরণ-৫. ছোট হাতের অক্ষরকে বড় হাতের অক্ষরে
রূপান্তর।

```
#include<stdio.h>
#include<conio.h>
void main()
{
char n;
printf("Enter any lower case character:");
scanf("%c",&n);
printf("You entered : %c",n-32);
getch();
}
```

ফলাফল: Enter any lower case character: a ↴
You entered: A

উদাহরণ-৬. বড় হাতের অক্ষরকে ছোট হাতের অক্ষরে
রূপান্তর।

```
#include<stdio.h>
#include<conio.h>
void main()
{
char n;
printf("Enter any uppercase character:");
scanf("%c",&n);
printf("You entered: %c",n+32);
getch();
}
```

ফলাফল: Enter any uppercase character: A ↴
You entered: a

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

<p>উদাহরণ-১ঃ কীবোর্ড থেকে একটি ডেসিমেল বা দশমিক সংখ্যা ইনপুট দিয়ে অক্টাল ও হেক্সাডেসিমেল সংখ্যায় বৃপ্তির করার জন্য সি ভাষায় প্রোগ্রাম।</p> <pre>#include<stdio.h> #include<conio.h> main() { int a; printf("Decimal number: "); scanf("%d",&a); printf("\nOctal number: %o",a); printf("\nHexadecimal number: %x",a); getch(); }</pre> <p>ফলাফল:</p> <p>Decimal number: 10 ↴ Octal number: 12 Hexadecimal number: a</p>	<p>উদাহরণ-২ঃ কীবোর্ড থেকে একটি অক্টাল সংখ্যা ইনপুট দিয়ে ডেসিমেল ও হেক্সাডেসিমেল সংখ্যায় বৃপ্তির করার জন্য সি ভাষায় প্রোগ্রাম।</p> <pre>#include<stdio.h> #include<conio.h> main() { int a; printf("Octal number: "); scanf("%o",&a); printf("\nDecimal number: %d",a); printf("\nHexadecimal number: %x",a); getch(); }</pre> <p>ফলাফল:</p> <p>Octal number: 177 ↴ Decimal number: 127 Hexadecimal number: 7f</p>	<p>উদাহরণ-৩ঃ কীবোর্ড থেকে একটি হেক্সাডেসিমেল সংখ্যা ইনপুট দিয়ে অক্টাল ও ডেসিমেল সংখ্যায় বৃপ্তির করার জন্য সি ভাষায় প্রোগ্রাম।</p> <pre>#include<stdio.h> #include<conio.h> main() { int a; printf("Hexadecimal number: "); scanf("%x",&a); printf("\nDecimal number: %d",a); printf("\nOctal number: %o",a); getch(); }</pre> <p>ফলাফল:</p> <p>Hexadecimal number: a ↴ Decimal number: 10 Octal number: 12</p>
--	--	---

printf() ফাংশন ব্যবহারকালে সাধারণত char টাইপ ভেরিয়েবল এর মান বাম এলাইনমেন্ট এবং int, float, double টাইপ ভেরিয়েবলের মান ডান এলাইনমেন্ট হিসেবে প্রদর্শিত হয়। তাই হিসেবের অসুবিধার জন্য অথবা ফলাফলের সৌন্দর্য বাড়াতে অনেক সময় ইন্টিজার টাইপ ভেরিয়েবলে মান প্রদর্শনে %2d, %3d, %5d ইত্যাদি ফরমেট স্পেসিফিয়ার ব্যবহৃত হয়। মূলত এলাইনমেন্ট আউটপুট প্রদর্শনে কিংবা প্রদর্শিতব্য ডেটা বা ভেরিয়েবলের বামে প্রয়োজনীয় সংখ্যক ফাঁকা স্থান তৈরি করতে এরূপ ফরম্যাট ব্যবহৃত হয়। অনুরূপে %c, %f, %ld প্রভৃতি স্পেসিফিয়ার এরূপে ব্যবহার করা যায়।

নিম্নের ছকে কয়েকটি ফরমেট স্পেসিফিয়ার ও তাদের ব্যবহার উল্লেখ করা হলো।

ফরমেট স্পেসিফিয়ার	ব্যবহার
%c	একটি ক্যারেক্টার প্রদর্শনে ১ ঘর জায়গা নেবে।
%2c	একটি ক্যারেক্টার প্রদর্শনে ২ ঘর জায়গা নেবে।
%nc	একটি ক্যারেক্টার প্রদর্শনে n ঘর জায়গা নেবে যার বামে n-1 ঘর ফাঁকা রাখবে।।
%d	প্রয়োজনীয় সংখ্যক ঘর নিয়ে একটি পূর্ণমান প্রদর্শন করবে।
%nd	একটি পূর্ণমান প্রদর্শনে n ঘর জায়গা নেবে যার বামে অবশিষ্ট ঘর ফাঁকা রাখবে।
%f	একটি দশমিক মান প্রদর্শনে দশমিক চিহ্নের পরে সর্বোচ্চ ৬ ঘর দেখাবে।
%nf	একটি দশমিক মান প্রদর্শনে দশমিক চিহ্নের পরে সর্বোচ্চ n ঘর দেখাবে।
%m.nf	m ঘর জায়গা নিয়ে একটি দশমিক মান প্রদর্শন করবে, যেখানে দশমিক চিহ্নের পরে সর্বোচ্চ (m-n-1) ঘর দেখাবে(দশমিকের জন্য ১ ঘর)
%ns	n ঘর জায়গা নিয়ে একটি স্ট্রিং প্রদর্শন করবে যেখানে স্ট্রিং এর বাম দিকে (স্ট্রিং দৈর্ঘ্য-n ঘর জায়গা) ফাঁকা রাখবে।
%no	অক্টাল মান প্রদর্শনে n ঘর জায়গা নেবে যার বামে অবশিষ্ট ঘর ফাঁকা রাখবে।
%nx	হেক্সাডেসিমেল মান প্রদর্শনে n ঘর জায়গা নেবে যার বামে অবশিষ্ট ঘর ফাঁকা রাখবে।

আমরা জানি, 265 একটি int টাইপ সংখ্যা, এই সংখ্যাটি প্রদর্শনের বেলায় %d ব্যবহার করলে আউটপুট মোট তিন ঘর জায়গা নিবে। কিন্তু %4d ব্যবহার করলে আউটপুট চার ঘর জায়গা নিবে, সুতরাং বাম দিকের এক ঘর ফাঁকা থাকবে। অনুরূপভাবে, %5d ব্যবহারে আউটপুট মোট পাঁচ ঘর জায়গা নিবে এবং বাম দিকের দুই ঘর ফাঁকা থাকবে, ইত্যাদি।

উদাহরণ-১

```
#include<stdio.h>
#include<conio.h>
main()
{
int x=265;
printf("\nx=%d",x);
printf("\nx=%4d",x);
printf("\nx=%5d",x);
getch();
}
```

ফলাফল:
x=265
x= 265
x= 265

আবার কোন float টাইপ মান প্রদর্শনের জন্য ভেরিয়েবলের মান যথাক্রমে দুই, তিন, চার ও ছয় দশমিক পর্যন্ত প্রদর্শন করার জন্য যথাক্রমে %.2f, %.3f, %.5f ও %f ফরম্যাট স্পেসিফিয়ার ব্যবহৃত হয়। নিম্নে সি প্রোগ্রামের মাধ্যমে দেখানো হলো।

উদাহরণ-২

```
#include<stdio.h>
#include<conio.h>
main()
{
float x=255;
printf("\nx=%f ",x);
printf("\nx=%10.0f ",x);
printf("\nx=%10.1f ",x);
printf("\nx=%10.2f ",x);
printf("\nx=%10.3f ",x);
printf("\nx=%10.4f ",x);
printf("\n");
printf("\nx=%f ",x);
printf("\nx=%0.0f ",x);
printf("\nx=%%.1f ",x);
```

ফলাফল:
x=255.000000
x= 255
x= 255.0
x= 255.00
x= 255.000
x= 255.0000

x=255.000000
x=255
x=255.0
x=255.00
x=255.000
x=255.0000

```

printf("\nx=%.2f ",x);
printf("\nx=%.3f ",x);
printf("\nx=%.4f ",x);
}

```

এবারে ধারাবাহিকভাবে আমরা বিভিন্ন অপারেটর নিয়ে প্রোগ্রাম লিখব।

অ্যারিথমেটিক অপারেটর: 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	$5/5=1$
-	বিয়োগ করার জন্য	$7-5=2$	%	ভাগশেষ নির্ণয়ের জন্য	$7\%5=2$
*	গুণ করার জন্য	$7*5=35$			

এখানে + এবং - ছাড়া বাকীগুলো শুধুমাত্র বাইনারি অপারেটর হিসাবে কাজ করে। অর্থাৎ, এরা দু'টো অপারেন্ট নিয়ে কাজ করে। + এবং - কে ইউনারী কিংবা বাইনারি অপারেটর হিসেবে ব্যবহার করা যায়।

প্রোগ্রামে ভাগের কাজ করার জন্য (/) অপারেটর ব্যবহার করে এক্সপ্রেশন তৈরি করার সময় নীচের বিষয়গুলো খেয়াল রাখতে হবে,

- ১) দ্বিতীয় অপারেন্টের মান অবশ্যই শূন্য হতে পারবে না।
- ২) উভয় অপারেন্টের ডেটা টাইপ যদি int হয়, তাহলে ভাগফলও int টাইপের হবে।

এখানে % অপারেটরটা ছাড়া অন্য অপারেটরগুলোর অপারেন্ট হিসেবে যেকোনো ডেটা টাইপ (int, float, double, char) ব্যবহার করা যায়। কিন্তু % অপারেটরের অপারেন্ট হিসাবে অবশ্যই int টাইপের ডেটাটাইপ ব্যবহার করতে হবে।

উদাহরণ-১: কীবোর্ড থেকে দুইটি সংখ্যা ইনপুট দিয়ে যোগফল নির্ণয় করার জন্য সি ভাষায় প্রোগ্রাম। <pre>#include<stdio.h> #include<conio.h> main() { int a, b, sum; printf(" Type the first number: "); scanf("%d",&a); printf(" Type the second number: "); scanf("%d",&b); sum = a+b; printf("Summation = %d",sum); getch(); }</pre> <p>ফলাফল:</p> <p>Type the first number: 100 ↴ Type the second number:27 ↴ Summation= 127</p>	উদাহরণ-২: কীবোর্ড থেকে দুইটি সংখ্যা ইনপুট দিয়ে বিয়োগফল নির্ণয় করার জন্য সি ভাষায় প্রোগ্রাম। <pre>#include<stdio.h> #include<conio.h> main() { int a, b, sub; printf(" Type the first number: "); scanf("%d",&a); printf(" Type the second number: "); scanf("%d",&b); sub = a-b; printf("Subtraction= %d",sub); getch(); }</pre> <p>ফলাফল:</p> <p>Type the first number: 100 ↴ Type the second number:27 ↴ Subtraction= 73</p>	উদাহরণ-৩: কীবোর্ড থেকে দুইটি সংখ্যা ইনপুট দিয়ে গুনফল নির্ণয় করার জন্য সি ভাষায় প্রোগ্রাম। <pre>#include<stdio.h> #include<conio.h> main() { int a, b, mul; printf(" Type the first number: "); scanf("%d",&a); printf(" Type the second number: "); scanf("%d",&b); mul = a*b; printf("Multiplication= %d", mul); getch(); }</pre> <p>ফলাফল:</p> <p>Type the first number: 100 ↴ Type the second number:27 ↴ Multiplication = 2700</p>
--	---	--

উপরোক্ত প্রোগ্রামে প্রথম printf() ফাংশন আমাদের জন্য জরুরী ছিল না। কিন্তু উক্ত ফাংশনটি ব্যবহৃত না হলে ইউজার কি করবে তা বুঝতে পারতো না। তাই ফাংশন ব্যবহার করে ইউজারকে বলে দেয়া হচ্ছে তাকে কি করতে হবে।

এই প্রোগ্রামের scanf() ফাংশনে দুটি প্যারামিটার %d এবং &a ব্যবহৃত হয়েছে। %d কম্পাইলারকে বলে দেয় যে, ইউজার একটি ইন্টিজার সংখ্যা ইনপুট দিবে এবং &a কম্পাইলারকে বলে দেয় যে, ইউজার কর্তৃক ইনপুটকৃত ইন্টিজার সংখ্যাটি a চলকের মধ্যে রাখতে হবে।

উদাহরণ-৪: কীবোর্ড থেকে দুইটি সংখ্যা ইনপুট দিয়ে ভাগফল নির্ণয় করার জন্য সি ভাষায় প্রোগ্রাম।

```
#include<stdio.h>
#include<conio.h>
main()
{
int a, b;
float div;
printf("Type the first number: ");
scanf("%d",&a);
printf("Type the second number: ");
scanf("%d",&b);
div = a/b;
printf("Division= %f",div);
getch();
}
```

ফলাফল:

Type the first number: 5 ↴
Type the second number:2 ↴
Division= 2.000000

সি ভাষা যেমন কেস সেনসিটিভ তেমনি ডেটা টাইপ সেনসিটিভও বটে। তাই ভিন্ন ভিন্ন ডেটা টাইপ ভেরিয়েবল বিশিষ্ট এক্সপ্রেশন নিয়ে কাজ করার সময় ভেরিয়েবলের টাইপ অনুযায়ী সঠিক ডেটা ব্যবহার করা হয়। এসাইনমেন্ট অপারেটরের সাহায্যে স্বয়ংক্রিয়ভাবে ডেটার টাইপ পরিবর্তন করা যায়। সেক্ষেত্রে এসাইনমেন্ট অপারেটরের ডান দিকের ভেরিয়েবলের ডেটা টাইপ বাম দিকের ভেরিয়েবলের ডেটা টাইপ পরিবর্তন হয়।

যেমন,
float y=2.25;
int x=y;

কাষ্ট অপারেশনের ক্ষেত্রে স্বল্পকালীন সময়ের জন্য (অনেকটা জোর করে) এক টাইপ ডেটা বা ভেরিয়েবলকে অন্য টাইপের ডেটা বা ভেরিয়েবলে রূপান্তর করা যায়।

এক্ষেত্রে x এর মান 5 নির্ধারিত হয়।

আবার কোন এক্সপ্রেশনে একই সাথে কনস্ট্যান্ট এবং বিল্টইন ডেটা টাইপ ভেরিয়েবল থাকলে কম্পাইলার নিজস্ব নিয়মে স্বয়ংক্রিয়ভাবে টাইপ পরিবর্তন করে নেয়। তবে কোনো কোনো এক্সপ্রেশনের ক্ষেত্রে কম্পাইলার স্বয়ংক্রিয়ভাবে ডেটা টাইপ পরিবর্তন করে না। যেমন- উপরোক্ত প্রোগ্রামে a=5,b=2 হলে a/b এর সঠিক মান 2.50 হয়, কিন্তু এখানে ডিভিশন অপারেটরের দ্বিতীয় অপারেন্ড পূর্ণসংখ্যা হওয়ায় ভাগফল পূর্ণসংখ্যা (2.000000) পাওয়া যাবে। সুতরাং a/b এর সঠিক মান পাওয়ার জন্য a ও b উভয়ই অথবা অন্তত যেকোনো একটি float টাইপ হওয়া আবশ্যিক। কিন্তু a ও b প্রোগ্রামে float হিসেবে ঘোষণা করা নাও থাকতে পারে। এসব ক্ষেত্রে কাষ্ট অপারেটর ব্যবহার করে টাইপ পরিবর্তন করা যেতে পারে। তবে এরূপ পরিবর্তন কেবল এই স্টেটমেন্টের জন্য কার্যকর থাকে, পরবর্তী স্টেটমেন্ট বা স্টেটমেন্ট সমূহের জন্য নয়।

সুতরাং উপরের প্রোগ্রামটি নিম্নের মত লিখলে সঠিক ফলাফল পাওয়া যাবে।

```
#include<stdio.h>
#include<conio.h>
main()
{
int a, b;
```

উদাহরণ-৫. সেলসিয়াস স্কেলের তাপমাত্রাকে ফারেনহাইট তাপমাত্রায় রূপান্তরের প্রোগ্রাম।

```
#include<stdio.h>
#include<conio.h>
main()
{
int c, f;
```

উদাহরণ-৬. ফারেনহাইট স্কেলের তাপমাত্রাকে সেলসিয়াস তাপমাত্রায় রূপান্তরের প্রোগ্রাম।

```
#include<stdio.h>
#include<conio.h>
main()
{
int c, f;
```

<pre> float div; printf("Type the first number: "); scanf("%d",&a); printf("Type the second number: "); scanf("%d",&b); div = a/(float)b; printf("Division of %d by %d is %f",a,b, div); getch(); } </pre> <p>ফলাফল:</p> <p>Type the first number: 5 ↴ Type the second number:2.↵ Division of 5 by 2 is = 2.500000</p>	<pre> printf("Enter celcious temperature :"); scanf("%d",&c); f=9*c/5+32; printf("Ferhenheight temperature:%d",f); getch(); } </pre> <p>ফলাফল :</p> <p>Enter celcious temperature : 100↵ Fahrenheight temperature: 212</p>	<pre> printf("Enter Ferenheight temperature :"); scanf("%d",&f); c=5*(f-32)/9; printf("Celcious temperature:%d",c); getch(); } </pre> <p>ফলাফল :</p> <p>Enter Ferenheight temperature : 100↵ Farhenheight temperature of 100 celcious is : 212</p>
<p>উদাহরণ-৭. ত্রিভুজের ভূমি ও উচ্চতা দেয়া, এর ক্ষেত্রফল নির্ণয় করার প্রোগ্রাম।</p> <pre> #include<stdio.h> #include<conio.h> main() { int b,h; float area; printf("Enter the Base:"); scanf("%d",&b); printf("Enter the Height:"); scanf("%d", &h); area=.5*b*h; printf("\nThe area is %.2f",area); getch(); } </pre> <p>ফলাফল:</p> <p>Enter the Base:5 ↴ Enter the Height:6↵ The area is 15.00</p>	<p>উদাহরণ-৮. ত্রিভুজের তিনটি বাহুর দৈর্ঘ্য যথাক্রমে a, b ও c দেয়া আছে। ত্রিভুজের ক্ষেত্রফল ও নির্ণয় করার প্রোগ্রাম।</p> <pre> #include<stdio.h> #include<conio.h> #include<math.h> main() { int a, b, c; float s, area; printf("Enter 3 integer values :"); scanf("%d %d %d", &a,&b,&c); s = (a + b + c)/2; area = sqrt(s*(s-a)*(s-b)*(s-c)); printf("Area of triangle is = %f", area); getch(); } </pre> <p>ফলাফল:</p> <p>Enter 3 integer values : 4 5 6 ↴ Area of triangle is = 6.480741</p>	<p>উদাহরণ-৯. আয়তক্ষেত্রের ক্ষেত্রফল নির্ণয় করার প্রোগ্রাম।</p> <pre> #include<stdio.h> #include<conio.h> main() { int a,b,x; printf("Enter the length & width: "); scanf("%d %d",&a,&b); x=a*b; printf("\nThe area is %d",x); getch(); } </pre> <p>ফলাফল:</p> <p>Enter the length & width:6 3↵ The area is 18</p>
<p>উদাহরণ-১০. বৃত্তের ক্ষেত্রফল নির্ণয়ের প্রোগ্রাম।</p> <pre> #include<stdio.h> #include<conio.h> main() { int r; float area; printf ("Enter integer value for radius:"); scanf ("%d", &r); </pre>	<p>উদাহরণ-১১. কোন পরিমাপ ফুটকে মিটারে প্রকাশ করার প্রোগ্রাম।</p> <p>[সূত্র : ১ মিটার = ৩.২৮ ফুট]</p> <pre> #include<stdio.h> #include<conio.h> main() { float m,f; clrscr(); printf("Enter feet: "); scanf("%f", &f); </pre>	<p>উদাহরণ-১২. কোন পরিমাপ ইঞ্চিকে সেন্টিমিটারে প্রকাশ করার প্রোগ্রাম।</p> <p>[সূত্র : ১ ইঞ্চি = ২.৫৪ সেন্টিমিটার]</p> <pre> #include<stdio.h> #include<conio.h> main() { int inch; float cm; </pre>

<pre>area = 3.14*r*r; printf("\n Area of circle =%f", area); getch(); } ফলাফল: Enter integer value for radius : 3 ↴ Area of circle = 28.260000</pre>	<pre>m= f/3.28; printf("\n Meter is %.2f",m); getch(); } ফলাফল: Enter feet: 6.56 ↴ Meter is 2.00</pre>	<pre>printf ("Integer value for inches:"); scanf ("%d", &inch) ; cm=inch*2.54; printf ("\nCentimeter=% .2f", cm); getch(); } ফলাফল: Integer value for inches: 5 ↴ Centimeter = 12.70</pre>
--	--	--



- কাজ:** ১. একটি মাঠের চারপাশে ৩ মিটার চওড়া একটি রাস্তা আছে। যদি মাঠের দৈর্ঘ্য x মিটার হয় তবে রাস্তার ক্ষেত্রফল নির্ণয়ের জন্য সি ভাষায় একটি প্রোগ্রাম লেখ।
 ২. একটি গরু x মিটার দৈর্ঘ্যের একটি রশি বা দড়ি দিয়ে একটি খুঁটির সাথে এমনভাবে বাঁধা আছে যাতে গরুটি অবাধে চলাচল করতে পারে। গরুটি ১ পাঁকে সর্বোচ্চ যে পথ পরিভ্রমণ করতে পারে তার দৈর্ঘ্য এবং উক্ত পথ দ্বারা তৈরিকৃত ক্ষেত্রের ক্ষেত্রফল নির্ণয়ের জন্য প্রোগ্রাম লেখ।

অ্যাসাইনমেন্ট অপারেটর :

অ্যাসাইনমেন্ট অপারেটর	উদাহরণ	ব্যবহার
=	a=b+5;	a এর মান b+5 হবে।
+=	a+=5;	a এর মান a+5 হবে, যা $x=x+5$ এর অনুরূপ।
-=	a-=5;	a এর মান a-5 হবে, যা $x=x-5$ এর অনুরূপ।
=	a=5;	a এর মান a*5 হবে, যা $x=x*5$ এর অনুরূপ।
/=	a/=5;	a এর মান a/5 হবে, যা $x=x/5$ এর অনুরূপ।
%=	a%=5;	a এর মান a%5 হবে, যা $x=x \% 5$ এর অনুরূপ।

উদাহরণ-১: <pre>#include<stdio.h> #include<conio.h> main() { int a=4,b=5,c=6,e=7,h=8,t=9,g=10; printf("The initial value a=4,b=5,c=6,e=7,h=8,t=9,g=10"); printf("\nThe value of a=t+5 is %d ",t+5); printf("\nThe value of b+=5 is %d ",b+=5); printf("\nThe value of c-=5 is %d ",c-=5); printf("\nThe value of e*=5 is %d ",e*=5); printf("\nThe value of g/=4 is %d ",g/=4); getch(); } ফলাফল: The initial value a = 4, b = 5, c=6, e=7, h=8, t=9,g=10 The value of a=t+5 is 14</pre>	উদাহরণ-২: <pre>#include<stdio.h> #include<conio.h> main() { int a=2,b=3,c=4; a=a+b+c; b=a+b+c; c=a+b+c; printf("The last value of a,b,c is %d,%d and %d",a,b,c); getch(); } ফলাফল: The last value of a, b, c is 9, 16 and 29</pre>
--	---

The value of b+=5 is 10	
The value of c-=5 is 1	
The value of e*=5 is 35	
The value of g/=4 is 12	

রিলেশনাল অপারেটর:

সি প্রোগ্রামে দুটো অপারেন্টের মধ্যে তুলনামূলক সম্পর্ক (ছোট, ছোট বা সমান, বড়, বড় বা সমান,সমান, সমান নয়) তৈরি করার জন্য রিলেশনাল অপারেটর ব্যবহার করা হয়। রিলেশনাল অপারেটর হলো বাইনারি অপারেটর। রিলেশনাল অপারেটরের বাম দিকে একটি অপারয়ান্ড এবং ডানদিকে আরেকটি অপারয়ান্ড থাকে। উভয় অপারয়ান্ড ভেরিয়েবল বা এক্সপ্রেশন হতে পারে আবার একটি ভেরিয়েবল বা এক্সপ্রেশন হলে অন্যটি কনস্ট্যান্ট হতে পারে। নিচে রিলেশনাল অপারেটর এবং এদের ব্যবহার ছকের মাধ্যমে দেওয়া হলো।

অপারেটর	ব্যবহার	উদাহরণ
< (ছোট)	ডান দিকের অপারয়ান্ডের চেয়ে বাম দিকের অপারয়ান্ড ছোট কিনা তা যাচাইয়ের জন্য।	a<5; a<(b-10);
<= (ছোট বা সমান)	ডান দিকের অপারয়ান্ডের চেয়ে বাম দিকের অপারয়ান্ড ছোট বা সমান কিনা তা যাচাইয়ের জন্য।	a<=5; a<=(b-c);
> (বড়)	ডান দিকের অপারয়ান্ডের চেয়ে বাম দিকের অপারয়ান্ড বড় কিনা তা যাচাইয়ের জন্য।	a>5; a>(b+c);
>= (বড় বা সমান)	ডান দিকের অপারয়ান্ডের চেয়ে বাম দিকের অপারয়ান্ড বড় বা সমান কিনা তা যাচাইয়ের জন্য।	a>=5; a>=(b+c);
== (সমান)	ডান দিকের অপারয়ান্ড এবং বাম দিকের অপারয়ান্ড সমান কিনা তা যাচাইয়ের জন্য।	a==10; a==(b+c)
!= (অসমান)	ডান দিকের অপারয়ান্ড এবং বাম দিকের অপারয়ান্ড অসমান কিনা তা যাচাইয়ের জন্য।	a!=15; a!=(b+c)

এখানে, a,b,c যে কোন বৈধ ভেরিয়েবল। রিলেশনাল অপারেটরের সাথে এক বা একাধিক ভেরিয়েবল অথবা অ্যারেথমেটিক এক্সপ্রেশন সহযোগে রিলেশনাল এক্সপ্রেশন তৈরি করা হয়। একটি রিলেশনাল এক্সপ্রেশনে যতগুলো অপারেন্ট কিংবা রিলেশনাল অপারেটর থাকুক না কেন এর মান কেবলমাত্র 0 অথবা 1 ছাড়া অন্য কিছু হবে না। যদি রিলেশনাল এক্সপ্রেশনটি সত্য হয় তাহলে এক্সপ্রেশনের মান হবে 1, কিন্তু রিলেশনাল এক্সপ্রেশনটি মিথ্যা হলে এক্সপ্রেশনের মান হবে 0। রিলেশনাল অপারেটরের ক্রমধারা বাম থেকে ডান দিকে হয়।

উদাহরণ-১ঃ

```
#include<stdio.h>
main()
{
    int a=5,b=0,c=10;
    int result1,result2;
    result1=a>b;
    result2=b>c;
    printf("Result1=%d",result1);
    printf("\nResult2=%d",result2);
}
```

ফলাফল:

Result1=1
Result2=0

লজিক্যাল অপারেটর:

সি প্রোগ্রামে বিভিন্ন ধরনের লজিক্যাল অপারেটর(যেমন- লজিক্যাল অর, লজিক্যাল অ্যান্ড, লজিক্যাল নট) সম্পর্ক করার জন্য লজিক্যাল অপারেটর ব্যবহার করা হয়। লজিক্যাল অর এন্ড লজিক্যাল অ্যান্ড বাইনারি অপারেটর হলেও লজিক্যাল নট ইউনারী অপারেটর। অপারেটর গুলো int টাইপের ডেটা নিয়ে কাজ করে। লজিক্যাল অপারেটরের সাথে এক বা একাধিক ভেরিয়েবল অথবা অ্যারেথমেটিক এক্সপ্রেশন সহযোগে লজিক্যাল এক্সপ্রেশন তৈরি করা হয় এবং লজিক্যাল এক্সপ্রেশনের মান শূন্য (0) অথবা এক(1) ছাড়া অন্যকিছু হতে পারে না। লজিক্যাল এক্সপ্রেশনে ব্যবহৃত কোন অপারেন্ট বা এক্সপ্রেশনের মান শূন্য ব্যতীত অন্য যে কোনো সংখ্যা হলে তার মান এক(1) ধরা হয়; অন্যথায় এর মান শূন্য ধরা হয়।

লজিক্যাল অর (logical OR) এক্সপ্রেশনের যেকোনো একটি অপারেন্ট true বা 1 হলে এক্সপ্রেশনের মান 1 হবে অন্যথায় এক্সপ্রেশনের মান 0 হবে।

লজিক্যাল অ্যান্ড (logical AND) এক্সপ্রেশনের সকল অপারেন্ট true বা 1 হলে এক্সপ্রেশনের মান 1 হবে অন্যথায় এক্সপ্রেশনের মান 0 হবে।

লজিক্যাল নট (logical NOT) এক্সপ্রেশন শুধু একটি অপারেন্ট নিয়ে কাজ করে বিধায় অপারেন্টটি true বা 1 হলে এক্সপ্রেশনের মান 0 হবে এবং অপারেন্টটি false বা 0 হলে এক্সপ্রেশনের মান 1 হবে।

কোনো লজিক্যাল অ্যান্ড এক্সপ্রেশনের মান বের করার জন্য কম্পাইলার প্রথমে প্রথম অপারেন্ট নিয়ে কাজ করে এবং এই অপারেন্টের মান 1 হলে কম্পাইলার দ্বিতীয় অপারেন্ট নিয়ে কাজ করে। যদি দ্বিতীয় অপারেন্টের মান 1 হয় তাহলে এক্সপ্রেশনের মানও 1 হবে। অন্যথায় মান 0 হবে। কিন্তু প্রথম অপারেন্টের মান যদি 0 হয় তাহলে দ্বিতীয় অপারেন্ট নিয়ে কাজ করে না। কেননা $\&\&$ এক্সপ্রেশনে একটি অপারেন্ট 0 হলেই পুরো এক্সপ্রেশনটির মান 0 হয়। তাই বাকী অপারেন্ট গুলো নিয়ে কাজ করার প্রয়োজন হয়।

কোনো লজিক্যাল অর এক্সপ্রেশনের মান বের করার জন্য কম্পাইলার প্রথমে প্রথম অপারেন্ট নিয়ে কাজ করে এবং এই অপারেন্টের মান 0 হলে কম্পাইলার দ্বিতীয় অপারেন্ট নিয়ে কাজ করে। যদি দ্বিতীয় অপারেন্টের মান 0 হয় তাহলে এক্সপ্রেশনের মানও 0 হবে। অন্যথায় মান 1 হবে। কিন্তু প্রথম অপারেন্টের মান যদি 1 হয় তাহলে দ্বিতীয় অপারেন্ট নিয়ে কাজ করে না। কেননা || এক্সপ্রেশনে একটি অপারেন্ট 1 হলেই পুরো এক্সপ্রেশনটির মান 1 হয়। তাই বাকী অপারেন্ট গুলো নিয়ে কাজ করার প্রয়োজন হয়।

নিচে উকৰে মাধ্যমে কয়েকটি লজিক্যাল অপারেশনের ফলাফল দেখানো হলো।

অপারেন্ট			OR অপারেশন	AND অপারেশন	NOT অপারেশন		
A	B	C	এক্সপ্রেশন	ফলাফল	এক্সপ্রেশন	ফলাফল	
3	4	5	A B	3 4 =1 1 =1	A&&B	3&&4 =1&&1 =1	!A !B
1	0	5	A B	1 0 =1	A&&B	1&&0 =0	
1	2	3	A C	1 2 =1 1 =1	A&&C	1&&2 =1&&1 =1	
4	5	0	(A&&B) C	(4&&5) 0 =(1&&1) 0 =1 0 =1	(A B)&&C	(4 5)&&0 =(1 1)&&0 =1&&0 =0	

মূলত বিভিন্ন প্রকার শর্ত পরীক্ষণে লজিক্যাল অপারেটরসমূহের বাস্তব প্রয়োগ দেখা যায়। নিচে কয়েকটি লজিক্যাল অপারেশনের ফলাফল পরীক্ষা করার জন্য প্রোগ্রাম দেয়া হলো।

উদাহরণ-১:

```
#include<stdio.h>
main()
{
    int a=2,b=3,c=0;
    int result1,result2,result3;
    result1=((a&&b)||c);
    result2=((a||b)&&c);
    printf("Result1=%d",result1);
    printf("\nResult2=%d",result2);
}
```

ফলাফল:

Result1=1

Result2=0

লজিক্যাল অপারেটর দ্বারা আমরা ডিজিটাল লজিক গেইট বাস্তবায়ন করতে পারি। লজিক্যাল অপারেটর দ্বারা গেইটকে নিম্নলিখিতভাবে প্রকাশ করতে পারি। লজিক্যাল অর, লজিক্যাল অ্যান্ড এবং লজিক্যাল নট অপারেশনের ফলাফল যথাক্রমে লজিক্যাল অর গেট, অ্যান্ড গেট এবং নট গেটের অনুরূপ।

লজিক গেইট	অপারেটর	লজিক্যাল এক্সপ্রেশন	সি ভাষায় প্রকাশ
OR	\parallel	$A+B$	$A\parallel B$
AND	$\&\&$	AB	$A\&\&B$
NOT	!	\bar{A}	\bar{A}
NOR		$\overline{A+B}$	$!(A\parallel B)$
NAND		$\overline{A.B}$	$!(A\&\&B)$

উদাহরণ-১: দুটি ইনপুট A ও B হলে OR Gate, AND Gate, NOT Gate, NOR Gate, NAND Gate বাস্তবায়ন করার জন্য সি ভাষায় প্রোগ্রাম লেখ।

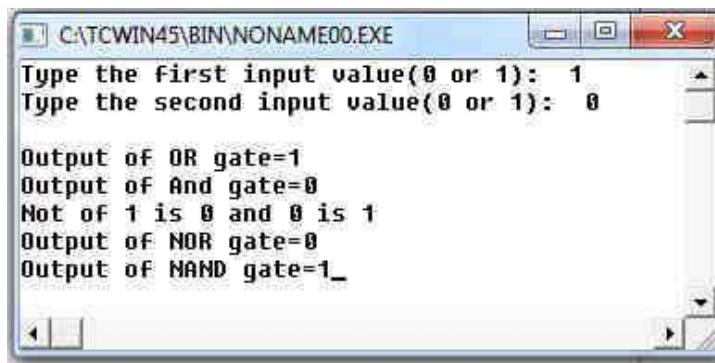
```
#include<stdio.h>
#include<conio.h>
main()
{
    int a,b,or,and,not1, not2,nor,nand;
    printf("Type the first input value : ");
    scanf("%d",&a);
    printf("Type the second input value : ");
    scanf("%d",&b);
    or=a||b;
    and=a&&b;
    not1=!a;
    not2=!b;
    nor=!(a||b);
    nand=!(a&&b);
    printf("\nOutput of OR gate=%d",or);
    printf("\nOutput of And gate=%d",and);
```

```

printf("\nNot of %d is %d and %d is %d",a,not1,b,not2);
printf("\nOutput of NOR gate=%d",nor);
printf("\nOutput of NAND gate=%d",nand);
getch();
}

```

আউটপুট: আমরা শুধু একটি ইনপুট দিয়ে নিম্নের মত আউটপুট দেখতে পাচ্ছি।



গেইটগুলোর সত্যক সারণি অনুযায়ী অন্যান্য মানগুলো ইনপুট দিয়ে তোমরা দেখতে পারো ঠিকমত রেজাল্ট আসছে কিনা।

বিটওয়াইজ অপারেটর:

প্রোগ্রামের কার্যক্ষমতা বাড়ানোর জন্য প্রোগ্রামে অনেক সময় বাইনারি ডেটা বা বিট নিয়ে কাজ করতে হয়। সি প্রোগ্রামে এই বিট পর্যায়ে কাজ করার জন্য যে সকল অপারেটর ব্যবহার করা হয় তাদেরকে বিটওয়াইজ অপারেটর বলে। নিচে ছকের মাধ্যমে বিটওয়াইজ অপারেটরসমূহের তালিকা এবং ব্যবহার উল্লেখ করা হলো।

বিটওয়াইজ অপারেটর	
& (বিটওয়াইজ AND)	দুইটি অপারেন্ডের বিটসমূহের মধ্যে জোড়ায় জোড়ায় অ্যান্ড (AND) অপারেশন সম্পন্ন করে।
 (বিটওয়াইজ OR)	দুইটি অপারেন্ডের বিটসমূহের মধ্যে জোড়ায় জোড়ায় অর (OR) অপারেশন সম্পন্ন করে।
^ (বিটওয়াইজ XOR)	দুইটি অপারেন্ডের বিটসমূহের মধ্যে জোড়ায় জোড়ায় এক্সঅর (ক্যারি বাদে বাইনারি যোগ) অপারেশন সম্পন্ন করে।
<< (শিফট লেফট)	কোন অপারেন্ডের বাইনারি বিট সমূহকে এক বা একাধিক বার বামদিকে সরানোর জন্য ব্যবহৃত হয়।
>> (শিফট রাইট)	কোন অপারেন্ডের বাইনারি বিট সমূহকে এক বা একাধিক বার ডানদিকে সরানোর জন্য ব্যবহৃত হয়।
~ (১' এর পরিপূরক করে তা দশমিকে দেখায়)	কোন অপারেন্ডের বাইনারি বিট সমূহকে বিপরীত করে অর্থাৎ 0 থাকলে 1 এবং 1 থাকলে 0 করে ফেলে।

যেমন:

A এবং B এর মান		ফলাফল			
A	B	A&B	A B	A^B	~A
0	0	0	0	0	1
0	1	0	1	1	1
1	0	0	1	1	0
1	1	1	1	0	0

বিটওয়াইজ নট হলো ইউনারি অপারেটর, অন্যগুলো বাইনারি অপারেটর। বিটওয়াইজ অপারেটর কেবল int টাইপের ডেটা নিয়ে কাজ করে। বিটওয়াইজ অপারেশনের পূর্বে কম্পাইলার উভয় অপারেশনের মান বাইনারিতে পরিবর্তন করে এবং অপারেন্ট দুইটির ডান দিক থেকে জোড়ায় জোড়ায় (বিট বাই বিট) অপারেশন সম্পন্ন করে। অতপর অ্যান্ডকৃত সংখ্যাটিকে পুনরায় দশমিকে রূপান্তর করে আউটপুট দেখায়। যেমন: A=4 এবং B=5 হলে A&B=4 হবে। নিচে ব্যাখ্যা করা হলো।

বিটওয়াইজ অ্যান্ড (&) অপারেশনের পূর্বে কম্পাইলার সকল অপারেন্টের মান বাইনারিতে পরিবর্তন করে এবং অপারেন্ট গুলোর ডান দিক থেকে জোড়ায় জোড়ায় (বিট বাই বিট) অ্যান্ড অপারেশন সম্পন্ন করে। অতপর অ্যান্ডকৃত সংখ্যাটিকে পুনরায় দশমিকে রূপান্তর করে আউটপুট দেখায়। যেমন: A=4 এবং B=5 হলে A&B=4 হবে। নিচে ব্যাখ্যা করা হলো।

$$A=(4)_{10} \Rightarrow A=(0\ 0\ 0\ 0\ 0\ 1\ 0\ 0)_2$$

$$B=(5)_{10} \Rightarrow B=(0\ 0\ 0\ 0\ 0\ 1\ 0\ 1)_2$$

$$A\&B = (0\ 0\ 0\ 0\ 0\ 1\ 0\ 0)_2 \text{ যা দশমিক } 4 \text{ এর সমান}$$

∴ A&B=4

বিটওয়াইজ অর (|) অপারেশনের পূর্বে কম্পাইলার সকল অপারেন্টের মান বাইনারিতে পরিবর্তন করে এবং অপারেন্ট গুলোর ডান দিকে থেকে জোড়ায় জোড়ায় (বিট বাই বিট) অর অপারেশন সম্পন্ন করে। অতপর অরকৃত সংখ্যাটিকে পুনরায় দশমিকে রূপান্তর করে আউটপুট দেখায়। যেমন: A=4 এবং B=5 হলে A|B=5 হবে। নিচে ব্যাখ্যা করা হলো।

$$A=(4)_{10} \Rightarrow A=(0\ 0\ 0\ 0\ 0\ 1\ 0\ 0)_2$$

$$B=(5)_{10} \Rightarrow B=(0\ 0\ 0\ 0\ 0\ 1\ 0\ 1)_2$$

$$A|B = (0\ 0\ 0\ 0\ 0\ 1\ 0\ 1)_2 \text{ যা দশমিক } 5 \text{ এর সমান}$$

∴ A|B = 5

বিটওয়াইজ এক্সঅর(^) অপারেশনের পূর্বে কম্পাইলার সকল অপারেন্টের মান বাইনারিতে পরিবর্তন করে এবং অপারেন্টগুলোর ডান দিক থেকে জোড়ায় জোড়ায় (বিট বাই বিট) এক্সঅর অপারেশন সম্পন্ন করে। অতপর এক্সঅরকৃত সংখ্যাটিকে পুনরায় দশমিকে রূপান্তর করে আউটপুট দেখায়। যেমন: A=4 এবং B=5 হলে A^B=5 হবে। নিচে ব্যাখ্যা করা হলো।

$$A=(4)_{10} \Rightarrow A=(0\ 0\ 0\ 0\ 0\ 1\ 0\ 0)_2$$

$$B=(5)_{10} \Rightarrow B=(0\ 0\ 0\ 0\ 0\ 1\ 0\ 1)_2$$

$$A^B = (0\ 0\ 0\ 0\ 0\ 0\ 0\ 1)_2 \text{ যা দশমিক } 1 \text{ এর সমান}$$

∴ A^B=1

আবার লেফট শিফট অপারেশনের পূর্বে কম্পাইলার << এর বাম দিকের অপারেন্টের মান বাইনারিতে পরিবর্তন করে অতপর ডান দিকের অপারেন্টের মান অনুযায়ী বিটগুলো এক এক করে বামদিকে সরিয়ে ডান দিকের ফাঁকা স্থান শূন্য দ্বারা পূর্ণ করে। অর্থাৎ প্রতিবার লেফট শিফট অপারেশনে কোন সংখ্যার মান দ্বিগুণ হয়। নিচে b=2 এর জন্য লেফট শিফট অপারেশনের মান দেখানো হলো।

এক্সপ্রেশন		এক্সপ্রেশনের বাইনারি মান	এক্সপ্রেশনের দশমিক মান
b	শিফটেড নয়	00000010	2
b<<1	১ম লেফট শিফট	00000100	4
b<<2	২য় লেফট শিফট	00001000	8
b<<3	৩য় লেফট শিফট	00010000	16

রাইট শিফট অপারেশনের পূর্বে কম্পাইলার >> এর মান বাম দিকের অপারেন্ডের বাইনারিতে পরিবর্তন করে অতপর ডানদিকের অপারেন্ডের মান অনুযায়ী বিটগুলো এক এক করে ডানদিকে সরিয়ে বামদিকের ফাঁকা স্থান শূন্য(0) দ্বারা পূর্ণ করে। অর্থাৎ প্রতিবার রাইট শিফট অপারেশনে কোনো সংখ্যার মান অর্ধেক হয়।

নিচে কয়েকটি সংখ্যার লেফট শিফট ও রাইট শিফট অপারেশনের ফলাফল দেখানো হলো।

এক্সপ্রেশন		এক্সপ্রেশনের বাইনারি মান	এক্সপ্রেশনের দশমিক মান
b	শিফটেড নয়	00010000	16
b>>1	১ম রাইট শিফট	00001000	8
b>>2	২য় রাইট শিফট	00000100	4
b>>3	৩য় রাইট শিফট	00000010	2

উদাহরণ-১: বিটওয়াইজ অর, বিটওয়াইজ অ্যান্ড এবং বিটওয়াইজ এক্সঅর অপারেটরের ব্যবহার।

```
#include<stdio.h>
#include<conio.h>
main()
{
    int a=4,b=5;
    printf("Bitwise OR=%d",a|b);
    printf("\nLogical OR=%d",a||b);
    printf("\nBitwise AND=%d",a&b);
    printf("\nLogical AND=%d",a&&b);
    getch();
}
```

ফলাফল:

Bitwise OR=5

Logical OR=1

Bitwise AND=4

Logical AND=1

উদাহরণ-২. বিটওয়াইজ লেফট শিফট, বিটওয়াইজ রাইট শিফট অপারেটরের ব্যবহার।

```
#include<stdio.h>
#include<conio.h>
main()
{
    int a=4;
    printf("1st Shift
Right=%d",a>>1);
    printf("\n2nd Shift
Right=%d",a>>2);
    printf("\n1st Shift
Left=%d",a<<1);
    printf("\n2nd Shift Left
=%d",a<<2);
    getch();
}
```

ফলাফল:

1st Shift Right=2

2nd Shift Right=1

1st Shift Left=8

2nd Shift Left =16

গেইট	লজিক্যাল এক্সপ্রেশন	সি ভাষায় প্রকাশ
XOR Gate	$A \oplus B$	A^B
XNOR Gate	$\overline{A \oplus B}$	$!(A^B)$

এবাবে আমরা বিটওয়াইজ অপারেটর এর ব্যবহার দেখব। নিচে বিটওয়াইজ ব্যবহার করে একটি সি ভাষায় প্রোগ্রাম দেখানো হলো।

উদাহরণ-৩: দুটি ইনপুট A ও B হলে XOR Gate, XNOR বাস্তবায়ন করার জন্য সি ভাষায় প্রোগ্রাম লেখ।

```
#include<stdio.h>
#include<conio.h>
main()
{
    int a,b,xor, xnor;
    printf("Type the first input value: ");
    scanf("%d",&a);
    printf("Type the second input value: ");
    scanf("%d",&b);
    xor=a^b;
    xnor=!( a^b);
    printf("\nOutput of XOR gate=%d",xor);
    printf("\nOutput of XNOR gate=%d",xnor);
    getch();
}
```

ফলাফল: Type the first input value:10

Type the Second input value:7

Output of XOR gate = 13

Output of XNOR gate = 0

বিটওয়াজ নট অপারেশন এবং লজিক্যাল নট অপারেশনের মধ্যে ভিন্নতা আছে। বিটওয়াইজ নট অপারেশনের আগে কম্পাইলার অপারেন্ডের মান বাইনারিতে পরিবর্তন করে এবং প্রতিটি বিটের জন্য বাইনারি নট অপারেশন সম্পন্ন করে। অতঃপর দশমিক সংখ্যায় পরিণত করে। কিন্তু লজিক্যাল নট অপারেশনের ব্যবহৃত অপারেন্ডের মান শূন্য (0) হলে কম্পাইলার এক্সপ্রেশনের ফলাফল এক(1) ধরে, আর অপারেন্ডের মান শূন্য না হলে এক্সপ্রেশনের ফলাফল শূন্য(0) ধরে।

Binary Number	Value(n)	One's Complement	Two's Complement	Value(-n)
00000001	1	11111110	11111111	-1
00000010	2	11111101	11111110	-2
00000011	3	11111100	11111101	-3
.....
00001000	8	11110111	11111000	-8
00001001	9	11110110	11110111	-9

উদাহরণ-৪: দুইয়ের পরিপূরক নির্ণয়ের জন্য প্রোগ্রাম।

```
#include<stdio.h>
#include<conio.h>
main()
{
    int a=4,b=5;
    printf("Two's complement of %d is %d",a,~a+1);
    printf("\nTwo's complement of %d is %d",b,~b+1);
    getch();
}
```

ফলাফল:

Two's complement of 4 is -4

Two's complement of 5 is -5

ইনক্রিমেন্ট/ ডিক্রিমেন্ট অপারেটর()

++ এবং -- কে যথাক্রমে ইনক্রিমেন্টাল ও ডিক্রিমেন্টাল অপারেটর বলা হয়। কোন অপারয়ান্ডের মান 1 বৃদ্ধি বা হ্রাস করতে যথাক্রমে ইনক্রিমেন্টাল ও ডিক্রিমেন্টাল ব্যবহৃত হয়। যেমন, Counter++ এবং Counter--; এখানে Counter একটি int টাইপ ভেরিয়েবল। Counter++ স্টেটমেন্টের মাধ্যমে Counter = Counter + 1; এবং Counter-- স্টেটমেন্টের মাধ্যমে Counter = Counter - 1; হয়। সাধারণত for এবং while লুপে ইনক্রিমেন্টাল এবং ডিক্রিমেন্টাল অপারেটর বেশি ব্যবহৃত হয়।

ইনক্রিমেন্টাল এবং ডিক্রিমেন্টাল অপারেটরের প্রিফিক্স এবং পোস্টফিক্স নোটেশন এবং এই দুইয়ের পার্থক্য:

ইনক্রিমেন্টাল বা ডিক্রিমেন্টাল অপারেটর এর অপারয়ান্ডে সংলগ্ন বাম দিকে বা আগে থাকলে তাকে প্রিফিক্স নোটেশন বলা হয়; যেমন, ++ Counter বা, -- Counter। আর ইনক্রিমেন্টাল বা ডিক্রিমেন্টাল অপারেটর ভেরিয়েবল সংলগ্ন এর অপারয়ান্ডে সংলগ্ন ডান দিকে বা পরে থাকলে তাকে পোস্টফিক্স নোটেশন বলা হয়; যেমন, Counter++ বা, Counter--; এখানে Counter একটি int টাইপ ভেরিয়েবল।

ইনক্রিমেন্টাল এবং ডিক্রিমেন্টাল অপারেটরের প্রিফিক্স এবং পোস্টফিক্স নোটেশন প্রায় একই রকম কাজ করে, তবে এদের মাঝে সামান্য পার্থক্য আছে। ইনক্রিমেন্টাল বা ডিক্রিমেন্টাল অপারেটরের প্রিফিক্স নোটেশনের ক্ষেত্রে কম্পাইলার প্রথমে ভেরিয়েবলের প্রারম্ভিক মানের সাথে যথাক্রমে এক ঘোগ বা বিয়োগ করে, অতপর প্রোগ্রামের একই স্টেটমেন্ট এই বৰ্ধিত মান ব্যবহার করে। কিন্তু ইনক্রিমেন্টাল বা ডিক্রিমেন্টাল অপারেটরের পোস্টফিক্স নোটেশনের ক্ষেত্রে কম্পাইলার প্রথমে প্রোগ্রামে ভেরিয়েবলের পুরাতন মান ব্যবহার করে, অতপর ভেরিয়েবলের মানের সাথে যথাক্রমে এক ঘোগ বা বিয়োগ করে। এই নতুন মান পরবর্তী স্টেটমেন্ট ধাপ থেকে কার্যকর হয়।

কন্ডিশনাল অপারেটর (Conditional Operator)

সি প্রোগ্রামে শর্ত সাপেক্ষে কোন ভেরিয়েবল বা এক্সপ্রেশনের মান অন্য কোন ভেরিয়েবল বা এক্সপ্রেশনের মান হিসাবে নির্ধারণ করার জন্য যে সকল অপারেটর ব্যবহার করা হয় তাকে কন্ডিশনাল অপারেটর বলে।

উদাহরণ: দুটি সংখ্যার মধ্যে বড় সংখ্যা নির্ণয়ের জন্য

কন্ডিশনাল অপারেটর ব্যবহার করে সি ভাষায় প্রোগ্রাম।

```
#include <stdio.h>
#include <conio.h>
main()
{
    int x,y;
    printf("Enter the first number: ");
    scanf("%d",&x);
    printf("Enter the second number: ");
    scanf("%d",&y);
    printf("Maximum number:%d", (x>y)?x:y);
    getch();
}
```

ফলাফল: Enter the first number:5

Enter the second number:6

Maximum number:6

```
#include<stdio.h>
#include<conio.h>
main()
{
    int a=4,b=5;
    printf("a++=%d",a++);
    printf("\n++b=%d",++b);
    getch();
}
```

অর্থবা

```
#include <stdio.h>
#include <conio.h>
main( )
{
    int x,y,a;
    printf("Enter the first number: ");
    scanf("%d",&x);
    printf("Enter the second number: ");
    scanf("%d",&y);
    a=(x>y)?x:y;
    printf("Maximum number:%d",a);
    getch();
}
```

ফলাফল: Enter the first number:5

Enter the second number:6

Maximum number:6

উদাহরণ: তিনটি সংখ্যার মধ্যে বড় সংখ্যা নির্ণয়ের জন্য কম্পিউটার অপারেটর ব্যবহার করে সি ভাষায় প্রোগ্রাম।

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int x,y,z,m;
    printf("Enter 1st number = ");
    scanf("%d",&x);
    printf("Enter 2nd number = ");
    scanf("%d",&y);
    printf("Enter 3rd number = ");
    scanf("%d",&z);
    m=(x>y)? (x>z)? x:z: (y>z)? y:z;
    printf("Maximum number = %d",m);
    getch();
}
```

ফলাফল: Enter 1st number =12
Enter 2nd number =13
Enter 3rd number =14
Maximum number =14

উদাহরণ: তিনটি সংখ্যার মধ্যে মধ্যম সংখ্যা নির্ণয়ের জন্য কম্পিউটার অপারেটর ব্যবহার করে সি ভাষায় প্রোগ্রাম।

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int x,y,z,m;
    printf("Enter 1st number = ");
    scanf("%d",&x);
    printf("Enter 2nd number = ");
    scanf("%d",&y);
    printf("Enter 3rd number = ");
    scanf("%d",&z);
    m=(x>y)? (x>z)? (y>z)? y:z: x:z:(x>z)? x:z:y;
    printf("Middle number = %d",m);
    getch();
}
```

উদাহরণ: কোনো একটি সাল লিপ ইয়ার(অধিবর্ষ) কি-না তা নির্ণয়ের জন্য সি ভাষায় প্রোগ্রাম

```
#include <stdio.h>
#include <conio.h>
main( )
{
    int year;
    printf(" Enter the year(4 digit) to check: ");
    scanf("%d",&year);
    ((year%400==0)|(year % 100!=0)&&(year%4==0))?printf("year %d is a leap year",year):
    printf("year %d is not a leap year",year);
    getch( );
}
```

উদাহরণ: তিনটি সংখ্যার মধ্যে ছোট সংখ্যা নির্ণয়ের জন্য কম্পিউটার অপারেটর ব্যবহার করে সি ভাষায় প্রোগ্রাম।

```
#include<stdio.h>
#include<conio.h>
void main()
{
    int x,y,z,m;
    printf("Enter 1st number = ");
    scanf("%d",&x);
    printf("Enter 2nd number = ");
    scanf("%d",&y);
    printf("Enter 3rd number = ");
    scanf("%d",&z);
    m=(x<y)? (x<z)? x:z: (y<z)? y:z;
    printf("Minimum number = %d",m);
    getch();
}
```

ফলাফল: Enter 1st number =12
Enter 2nd number =13
Enter 3rd number =14
Minimum number=12

ফলাফল: Enter 1st number =12
Enter 2nd number =13
Enter 3rd number =14
Middle number =13

ফলাফল: Enter the year(4 digit) to check:: 2012
year 2012 is a Leap year
অথবা, Enter a year: 2013
year 2013 is not a Leap year

পাঠ-২৫ ও ২৬

ব্যবহারিক : কন্ট্রোল ও কন্ডিশনাল কন্ট্রোল স্টেটমেন্ট (Control and Conditional Control Statement)

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

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

```
x=y+7;
printf("This is simple statement");
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 কোন সেমিকোলন (;) দিতে হয় না। কম্পাউন্ড স্টেটমেন্টকে আবার রুক স্টেটমেন্টও বলা হয়। একটি কম্পাউন্ড স্টেটমেন্ট আবার অন্য কোন কম্পাউন্ড স্টেটমেন্টকে ধারণ করতে পারে। যেমন:
- ```
{
 x=0;
 printf("This is compound statement");
 ++x;
}
```

### কন্ট্রোল স্ট্রাকচার (Control Structure)

'সি' প্রোগ্রামের স্টেটমেন্ট সমূহ সাধারণত স্বয়ংক্রিয়ভাবে ও পর্যায়ক্রমে একবার করে সম্পাদিত হয়। কিন্তু যদি দুই বা ততোধিকবার সম্পাদনের প্রয়োজন হয়, কিংবা কোনো স্টেটমেন্ট কোনো শর্ত সাপেক্ষে অথবা অপর কোনো স্টেটমেন্টের ফলাফলের ভিত্তিতে সম্পাদনের প্রয়োজন হয় অথবা কোনো স্টেটমেন্ট হতে প্রোগ্রামের নিয়ন্ত্রণ অন্য কোনো স্টেটমেন্টে স্থানান্তরের প্রয়োজন হয়, সেসব ক্ষেত্রে স্টেটমেন্ট-সমূহের নির্বাহ প্রোগ্রামার নিয়ন্ত্রণ করে। প্রোগ্রামে এমন স্টেটমেন্ট-সমূহের নির্বাহ নিয়ন্ত্রণের জন্য কন্ট্রোল স্ট্রাকচার ব্যবহার করা হয়। '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)

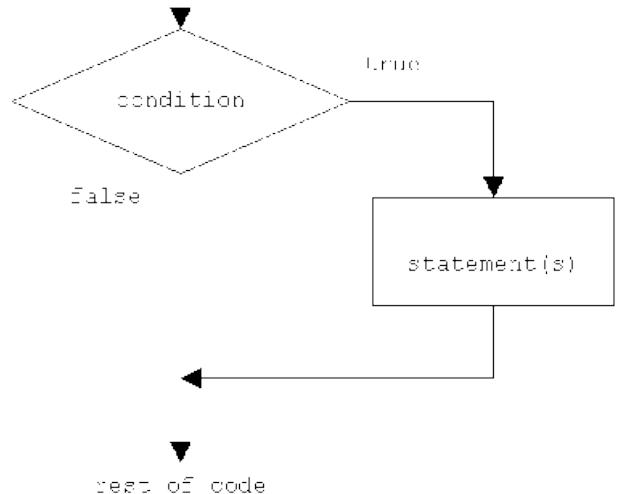
**কন্ডিশনাল কন্ট্রোল স্টেটমেন্ট (Conditional Control Statement):** 'সি' প্রোগ্রামে শর্তসাপেক্ষে কোনো স্টেটমেন্ট সম্পাদনের জন্য কন্ডিশনাল কন্ট্রোল ব্যবহৃত হয়। কন্ডিশনাল কন্ট্রোলে ব্যবহৃত শর্ত সত্য হলে প্রোগ্রামে এক ধরনের ফলাফল পাওয়া যায় এবং মিথ্যা হলে অন্য ধরনের ফলাফল পাওয়া যায়। অন্যতম কন্ডিশনাল কন্ট্রোল স্টেটমেন্টগুলো হচ্ছে:

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

**if স্টেটমেন্ট:** প্রোগ্রামে শর্ত সাপেক্ষে কোনো স্টেটমেন্ট সম্পাদনের জন্য if স্টেটমেন্ট ব্যবহার করা হয়। if স্টেটমেন্ট ব্যবহারের ফরম্যাট হলো- if (Condition)

```
{
 Action1;
}
```

if স্টেটমেন্টে শর্ত সাধারণত এক বা একাধিক লজিক্যাল বা রিলেশনাল এক্সপ্রেশন হয় যা if পরবর্তী প্রথম বন্ধনীর মধ্যে লেখা হয়। এই শর্তের মান যদি সত্য হয় তবে Action1 সম্পাদিত হয়। তা না হলে Action1 সম্পাদিত হয় না।



চিত্র : if স্টেটমেন্টের ফোচার্ট

**উদাহরণ-১:** প্রোগ্রামে if statement যেভাবে কাজ করে তা একাটি উদাহরণের সাহায্যে দেখানো হলো।

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

**ফলাফল:** Enter a value : 15  
15 is a positive number.

**উদাহরণ-২:** তিনটি সংখ্যার মধ্যে মাঝারি সংখ্যাটি নির্ণয়ের জন্য প্রোগ্রাম।

```
#include<stdio.h>
#include<conio.h>
main()
{
int a,b,c;
printf("Enter 3 integer values:");
scanf("%d %d %d", &a, &b, &c);
if(b>a && a>c || c>a && a>b)
{
 printf("%d is middle number",a);
}
if(a>b && b>c || c>b && b>a)
{
 printf("%d is middle number",b);
}
if(a>c && c>b || b>c && c>a)
{
 printf("%d is middle number",c);
}
```

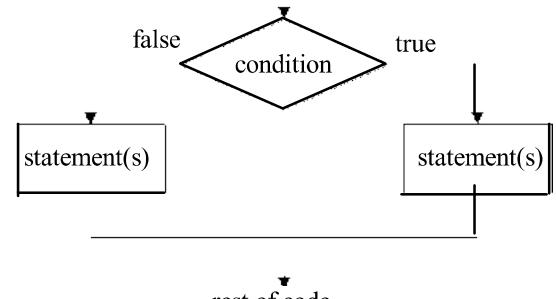
**ফলাফল:** Enter 3 integer values: 4 9 5  
5 is middle number

**if.....else স্টেটমেন্ট :** ‘সি’ প্রোগ্রামে ‘অন্যথায়’ অর্থে if স্টেটমেন্টের সাথে else স্টেটমেন্ট ব্যবহৃত হয়। if....else স্টেটমেন্ট ব্যবহারের ফরম্যাট হলো-

```

if (Condition)
{
 Action1;
}
else {
 Action 2;
}

```



চিত্র : if ..... else স্ট্রিমেন্টের ফ্লোচার্ট

if.... else কট্টোলে ব্যবহৃত শর্ত (condition) সাধারণত এক বা একাধিক লজিক্যাল বা রিলেশনাল এক্সপ্রেশন হয় যা if এর পরে প্রথম বন্ধনীর মধ্যে লেখা হয়। এই শর্তের মান যদি সত্য হয় তবে Action 1 সম্পাদিত হয়। তা নাহলে Action 1 সম্পাদিত না হয়ে Action 2 সম্পাদিত হয়।

**উদাহরণ-১:** কোন সংখ্যা জোড়/বিজোড় তা নির্ণয় করার প্রোগ্রাম।

```

#include<stdio.h>
#include<conio.h>
main()
{
 int a;
 printf("Enter a number: ");
 scanf("%d", &a);
 if (a%2==0)
 printf("\nThe number %d is even.",a);
 else
 printf("\nThe number %d is odd.",a);
 getch();
}

```

ফলাফল : Enter a number:12 ↴

The number 12 is even.

**উদাহরণ-৩:** দুইটি পূর্ণ সংখ্যা ইনপুট দিতে হবে এবং এদের মধ্যে বড় সংখ্যাটি প্রিন্ট করার জন্য if-else ব্যবহার করে প্রোগ্রাম।

```

#include<stdio.h>
#include<conio.h>
void main()
{
 int a, b;
 printf("Enter 1st value :");
 scanf("%d",&a);
 printf("Enter 2nd value :");
 scanf("%d",&b);
 if (a>b)
 printf("Largest Number is : %d",

```

**উদাহরণ-২:** কোন সংখ্যা ধনাত্মক/খনাত্মক তা নির্ণয় করার প্রোগ্রাম।

```

#include<stdio.h>
#include<conio.h>
main()
{
 int a;
 printf("Enter a number: ");
 scanf("%d", &a);
 if (a>=0)
 printf("\nThe number %d is positive.",a);
 else
 printf("\nThe number %d is Negative.",a);
 getch();
}

```

ফলাফল : Enter a Number: 12

The number 12 is positive.

**উদাহরণ-৪:** একটি সাল ইনপুট দিতে হবে এবং সালটি Leap Year কিনা তা নির্ণয় করার জন্য প্রোগ্রাম।

```

#include<stdio.h>
#include<conio.h>
void 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",

```

a);
else
 printf("Largest Number is: %d", b);
getch();
}

```

ফলাফল : Enter 1st value: 6  
Enter 2nd value: 9

Largest Number is: 9

**উদাহরণ-৫:** দুইটি পূর্ণ সংখ্যা ইনপুট দিয়ে গ.সা.গু

নির্ণয়ের জন্য প্রোগ্রাম।

```

#include<stdio.h>
#include<conio.h>
main()
{
int a,b,x;
printf("Type the two number: ");
scanf("%d %d",&a,&b);
x=(a<b)?a:b;
again: if(a%x==0 && b%x==0)
printf("GCD of %d and %d is
%d",a,b,x);
else
{
 x=x-1;
 goto again;
}
getch();
}

```

ফলাফল : Type the two number: 12 8  
GCD of 12 and 8 is 4

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

```

if (Condition1)
{
 if(condition2)
 Action1;
 else
 Action2;
}
else
{
 if(condition3)
 Action3;
 else
 Action4;
}

```

```

y);
getch();
}

```

ফলাফল: Enter a year: 2012  
2012 is a Leap year

অথবা, Enter a year: 2013  
2013 is not a Leap year

**উদাহরণ-৬:** দুইটি পূর্ণ সংখ্যা ইনপুট দিয়ে ল.সা.গু

নির্ণয়ের জন্য প্রোগ্রাম।

```

#include<stdio.h>
#include<conio.h>
main()
{
int a,b,x;
printf("Type the two number: ");
scanf("%d %d",&a,&b);
x=(a>b)?a:b;
again: if(x%a==0 && x%b==0)
printf("LCM of %d and %d is %d",a,b,x);
else
{
 x=x+1;
 goto again;
}
getch();
}

```

ফলাফল: Type the two number: 12 8  
LCM of 12 and 8 is 24

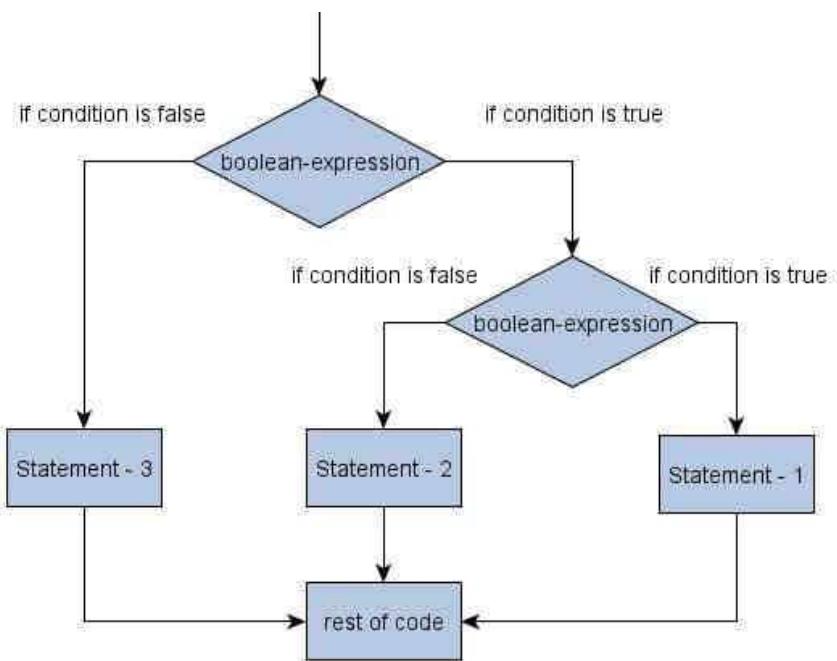
**উদাহরণ-৭:** তিনটি সংখ্যার মধ্যে বড় সংখ্যাটি নির্ণয়ের জন্য প্রোগ্রাম।

```
#include<stdio.h>
#include<conio.h>
main()
{
 int a,b,c;
 printf("Enter 3 integer values:");
 scanf("%d %d %d", &a, &b, &c);
 if (a > b)
 {
 if(a > c)
 printf("\n Largest Value is: %d", a);
 else
 printf("\n Largest Value is: %d", c);
 }
 else
 {
 if(b > c)
 printf("\n Largest Value is: %d", b);
 else
 printf("\n Largest Value is: %d", c);
 }
 getch();
}
```

**ফলাফল:** Enter 3 integer values: 4 9 5  
Largest Value is: 9

**else if স্টেটমেন্ট (বা nested if স্টেটমেন্ট):** প্রোগ্রামে একাধিক শর্ত যাচাই করার জন্য else if ব্যবহার করা হয়। সি' প্রোগ্রামে “অন্যথায় যদি” অর্থে if...else স্টেটমেন্টের সাথে else if স্টেটমেন্ট ব্যবহার করা হয়। else if স্টেটমেন্ট if...else স্টেটমেন্টের if এবং else স্টেটমেন্টের মাঝে বসে। else if স্টেটমেন্টে else এবং if এর মাঝে ফাঁকা স্থান থাকে। else if স্টেটমেন্ট ব্যবহারের ফরম্যাট হলো—

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



**উদাহরণ-১:** তিনটি সংখ্যার মধ্যে বড় সংখ্যাটি নির্ণয়ের জন্য প্রোগ্রাম।

```
#include<stdio.h>
#include<conio.h>
main()
{
 int a,b,c;
 printf("Enter 3 integer values:");
 scanf("%d %d %d", &a, &b, &c);
 if ((a > b) && (a > c))
 printf("\n Largest Value is: %d", a);
 else if ((b > a) && (b > c))
 printf("\n Largest Value is: %d", b);
 else
 printf("\n Largest Value is: %d", c);
 getch();
}
```

**ফলাফল:** Enter 3 integer values: 4 9 5  
Largest Value is: 9

**উদাহরণ-৩:** তিনটি সংখ্যার মধ্যে মাঝারি সংখ্যাটি নির্ণয়ের জন্য প্রোগ্রাম।

```
#include<stdio.h>
#include<conio.h>
main()
{
 int a,b,c;
 printf("Enter 3 integer values:");
 scanf("%d %d %d", &a, &b, &c);
 if(a>b)
 {
 if(b>c)
 {
 printf("%d is middle one",b);
 }
 else if(c>a)
 {
 printf("%d is middle one",a);
 }
 else
 {
 printf("%d is middle one",c);
 }
 }
 else
 {
 if(b<c)
 {
```

**উদাহরণ-২:** একটি সাল ইনপুট দিতে হবে এবং সালটি Leap Year কিনা তা নির্ণয় করার জন্য প্রোগ্রাম।

```
#include<stdio.h>
#include<conio.h>
main()
{
 int y;
 printf("Enter a year:");
 scanf("%d",&y);
 if (y % 400== 0)
 printf("%d is a Leap year", y);
 else if ((y % 100 != 0) && (y % 4==0))
 printf("%d is a Leap year", y);
 else
 printf("%d is not a Leap year", y);
 getch();
}
```

**ফলাফল:** Enter a year: 2012  
2012 is a Leap year  
অথবা, Enter a year: 2013  
2013 is not a Leap year

```

 printf("%d is middle one",b);
 }
 else if(c<a)
 {
 printf("%d is middle one",a);
 }
 else
 {
 printf("%d is middle one",c);
 }
}
}

```

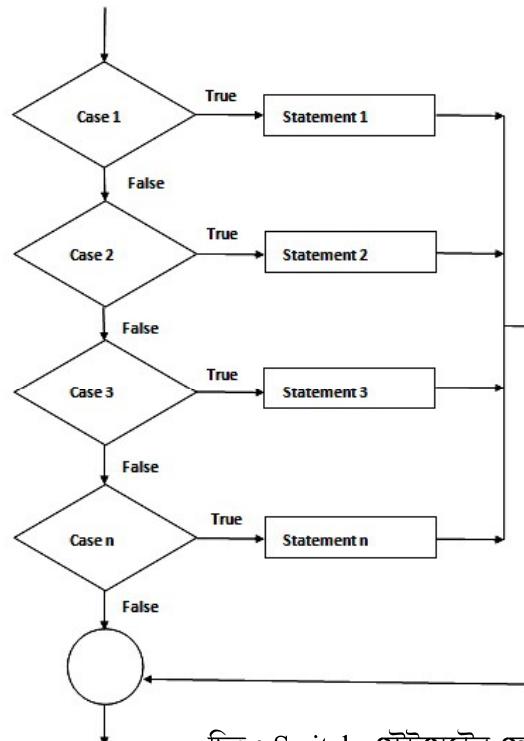
**ফলাফল:** Enter 3 integer values: 4 9 5  
5 is middle one

**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 ব্যবহার করা হয়। switch স্টেটমেন্ট-এর সাথে অতিরিক্ত case, break ও default স্টেটমেন্ট ব্যবহৃত 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 নির্বাচন করা হয়। switch স্টেটমেন্ট-এর ফরম্যাট হলো-

```

Data Type IndexVariable;
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;
}

```



চিত্র : Switch স্টেটমেন্টের ফ্লোচার্ট

**উদাহরণ-১: switch statement ব্যবহার করে বিভিন্ন গেট বাস্তবায়ন করার প্রোগ্রাম।**

```

#include<stdio.h>
#include<conio.h>>
main()
{
int a,b, andgate, orgate, notgate, nandgate, norgate, xor gate, xnorgate;

```

```

int choice;
printf("1: AND Gate\n");
printf("2: OR Gate\n");
printf("3: NOT Gate\n");
printf("4: NAND Gate\n");
printf("5: NOR Gate\n");
printf("6: XOR Gate\n");
printf("7: XNOR Gate\n");
printf("8: Exit\n");

printf("Type the choice(1 to 7 for above listed gate): ");
scanf("%d",&choice);
printf("\n");
switch(choice)
{
case 1:
 printf(".....For AND Gate.....\n");
 printf("\n");
 printf("Enter the input value1 (0 or 1 only):");
 scanf("%d",&a);
 printf("Enter the input value2(0 or 1 only):");
 scanf("%d",&b);
 andgate=a&&b;
 printf("AND gate Output=%d",andgate);
 break;
case 2:
 printf(".....For OR Gate.....\n");
 printf("\n");
 printf("Enter the input value1(0 or 1 only):");
 scanf("%d",&a);
 printf("Enter the input value2(0 or 1 only):");
 scanf("%d",&b);
 orgate=a||b;
 printf("OR Gate Output=%d",orgate);
 break;
case 3:
 printf(".....For NOT Gate.....\n");
 printf("\n");
 printf("Enter the input value(0 or 1 only):");
 scanf("%d",&a);
 notgate=!a;
 printf("NOT Gate Output=%d",notgate);
 break;
case 4:
 printf(".....For NAND Gate.....\n");
 printf("\n");
 printf("Enter the input value1(0 or 1 only):");
 scanf("%d",&a);
 printf("Enter the input value2(0 or 1 only):");
 scanf("%d",&b);
 nandgate=!(a&&b);
 printf("NAND Gate Output=%d",nandgate);
 break;
case 5:

```

```

printf(".....For NOR Gate.....\n");
printf("\n");
printf("Enter the input value1(0 or 1 only):");
scanf("%d",&a);
printf("Enter the input value2(0 or 1 only):");
scanf("%d",&b);
norgate=!(a||b);
printf("NOR Gate Output=%d",norgate);
break;

```

case 6:

```

printf(".....For XOR Gate.....\n");
printf("\n");
printf("Enter the input value1(0 or 1 only):");
scanf("%d",&a);
printf("Enter the input value2(0 or 1 only):");
scanf("%d",&b);
xorgate=a^b;
printf("XOR Gate Output=%d",xorgate);
break;

```

case 7:

```

printf(".....For XNOR Gate.....\n");
printf("\n");
printf("Enter the input value1(0 or 1 only):");
scanf("%d",&a);
printf("Enter the input value2(0 or 1 only):");
scanf("%d",&b);
xnorgate=!(a^b);
printf("XNOR Gate Output=%d",xnorgate);
break;

```

case 8:

break;

default:

printf("Invalid choice.");

}

getch();

}

**ফলাফল :**

- 1: AND Gate
- 2: OR Gate
- 3: NOT Gate
- 4: NAND Gate
- 5: NOR Gate
- 6: XOR Gate
- 7: XNOR Gate
- 8: Exit

Type the choice (1 to 7 for above listed gate): 1

..... For AND Gate.....

Enter the input value1 (0 or 1 only):1

Enter the input value2 (0 or 1 only):1

AND gate output=1



**কাজ:** ১. একটি সালের ফেব্রুয়ারি মাস কত দিনে তা নির্ণয়ের জন্য প্রোগ্রাম লেখ।  
২. কোনো বিষয়ের প্রাপ্ত নম্বর থেকে গ্রেড নির্ণয়ের জন্য প্রোগ্রাম লেখ।

## পাঠ-২৭ ও ২৮

### ব্যবহারিক : লুপ ও লুপের ব্যবহার (Loop and Uses of Loop)

প্রোগ্রামের অংশ বিশেষ নির্দিষ্ট সংখ্যক বার কোনো শর্তে না পৌছা পর্যন্ত পুনরাবৃত্তি করাকে লুপিং বা চক্র নিয়ন্ত্রণ বলা হয়। লুপকে তিন ভাগে ভাগ করা হয়। যথা—

- অসীম লুপ (Endless Loop):** যদি কোনো লুপ অনবরত আবর্তন হতে থাকে, কখনো শেষ না হয় তবে তাকে অসীম লুপ বলে।
- সসীম লুপ (Finite Loop):** নির্দিষ্ট সংখ্যক আবর্তনের পর যে লুপ শেষ হয় তাকে সসীম লুপ বলে।
- 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 কন্ট্রোল স্টেটমেন্ট ব্যবহৃত হয় তাকে লুপ কন্ট্রোল স্টেটমেন্ট বলে। লুপ স্টেটমেন্টসমূহে সাধারণত দুইটি অংশ থাকে। যথাঃ

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

লুপ স্টেটমেন্টের লুপ বডি এবং 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 স্টেটমেন্ট, while 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

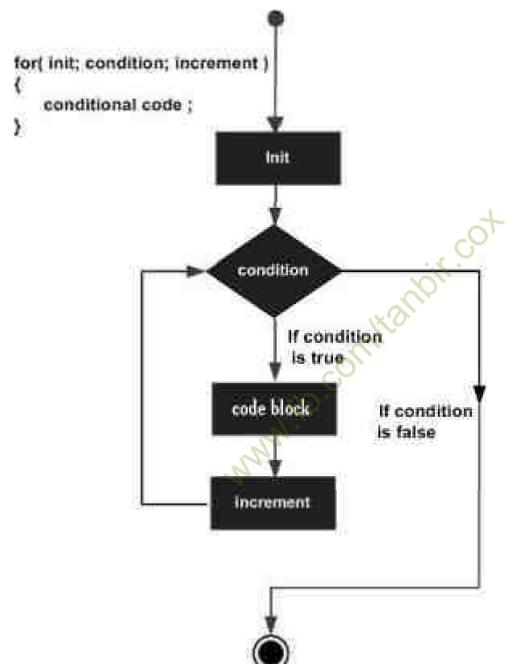
সি প্রোগ্রামে লুপ নির্বাহের জন্য ব্যবহৃত অন্যতম লুপ কন্ট্রোল স্টেটমেন্ট সমূহ হলঃ

- for লুপ স্টেটমেন্ট
- while লুপ স্টেটমেন্ট
- do...while লুপ স্টেটমেন্ট
- continue স্টেটমেন্ট
- goto স্টেটমেন্ট

লুপ স্টেটমেন্ট নির্বাহের জন্য প্রধান বিবেচ্য বিষয়সমূহ হলঃ

- 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 করা হয়। এরূপ ভেরিয়েবলকে কাউন্টার ভেরিয়েবল বলে। for স্টেটমেন্ট-এর ফরম্যাট দেখানো হলো—



চিত্র : for স্টেটমেন্টের ফ্লোচার্ট

```
Counter Declaration;
for (initial value; condition; decrement/increment)
{
 statement;
}
```

**Counter Declaration** অংশে উপযুক্ত ডেটা টাইপসহ কাউন্টার ভেরিয়েবল ঘোষণা করা হয়, **initial value** অংশে কাউন্টার ভেরিয়েবলের প্রারম্ভিক মান দেওয়া হয়, **test** অংশে কাউন্টার ভেরিয়েবলের চূড়ান্ত মান কিংবা চূড়ান্ত মান নির্ধারণের শর্ত দেয়া হয় এবং **decrement/increment** অংশে প্রতিবার আবর্তনে কাউন্টার ভেরিয়েবলের ছাস/বৃদ্ধির মান নির্ধারণ করা হয়। কাউন্টার ভেরিয়েবল চূড়ান্ত মানে না পৌঁছা পর্যন্ত কিংবা শর্ত সত্য থাকা পর্যন্ত for লুপের সাথে সংশ্লিষ্ট স্টেটমেন্ট সম্পাদিত হতে থাকে।

**উদাহরণ-১. for loop** ব্যবহার করে Bangladesh লেখাটি ১০ বার প্রিন্ট করার জন্য প্রোগ্রাম।

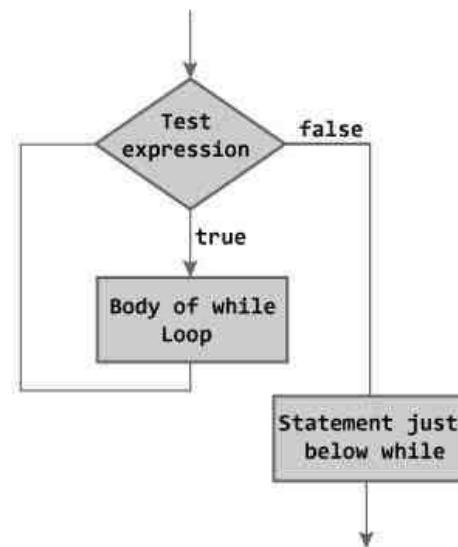
```
#include<stdio.h>
#include<conio.h>
int main()
{
 int a;
 for(a=1;a<=10; a++)
 {
 printf("\n Bangladesh");
 }
 getch();
 return 0;
}
```

ফলাফল : Bangladesh  
Bangladesh

**while স্টেটমেন্ট:** ‘সি’ প্রোগ্রামে শর্ত সাপেক্ষে দুই বা ততোধিকবার কোনো স্টেটমেন্ট সম্পাদনের জন্য while স্টেটমেন্ট ব্যবহার করা 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—

```
CounterDeclaration;
Counter Initialization;
while (Condition is true)
{
 statement;
 increment/ decrement;
}
```

CounterDeclaration অংশে উপযুক্ত ডেটা টাইপসহ কাউন্টার ভেরিয়েবল ঘোষণা হয়, Counter Initialization অংশে কাউন্টার ভেরিয়েবলের প্রারম্ভিক মান দেওয়া হয়। Condition অংশে কাউন্টার ভেরিয়েবলের চূড়ান্ত মান কিংবা চূড়ান্ত মান নির্ধারণের শর্ত দেওয়া হয়। Condition সত্য থাকা পর্যন্ত while লুপের সাথে সংশ্লিষ্ট স্টেটমেন্ট সম্পাদিত হতে থাকে।



চিত্র : While স্টেটমেন্টের ফ্লোচার্ট

**Bangladesh লেখাটি ১০ বার প্রিন্ট করার জন্য প্রোগ্রাম।**

```
#include<stdio.h>
#include<conio.h>
int main()
{
int a=1;
while(a<=10);
{
 printf("Bangladesh\n");
 a++;
}
getch();
return 0;
}
```

ফলাফল : Bangladesh  
Bangladesh

**do...while স্টেটমেন্ট:** ‘সি’ প্রোগ্রামে শর্ত সাপেক্ষে এক বা একাধিকবার কোনো স্টেটমেন্ট সম্পাদনের জন্য do স্টেটমেন্ট ব্যবহার করা হয়। while স্টেটমেন্টের মতো কোনো পূর্ব ঘোষিত কাউন্টার ভেরিয়েবল ব্যবহার করে do....while স্টেটমেন্টের আবর্তন সংখ্যা গণনা করা হয় এবং সেই অনুযায়ী সিদ্ধান্ত নেওয়া হয়। পাশে do....while স্টেটমেন্টের ফরম্যাট দেয়া হলো—

```
CounterDeclaration;
Counter Initialization;
do
{
 statement;
 increment/decrement;
}while (Condition is true);
```

Counter Declaration অংশে উপযুক্ত ডেটা টাইপসহ ইনডেক্স ভেরিয়েবল ঘোষণা করা হয় এবং Counter Initialization অংশে তার প্রারম্ভিক মান দেওয়া হয়। Condition অংশে ইনডেক্স ভেরিয়েবলের চূড়ান্ত মান কিংবা চূড়ান্ত মান নির্ধারণের শর্ত দেওয়া হয়।

**Bangladesh লেখাটি ১০ বার প্রিন্ট করার জন্য প্রোগ্রাম।**

```
#include<stdio.h>
#include<conio.h>
int main()
{
int a=1;
do
{
 printf("Bangladesh\n");
 a++;
} while(a<=10);
getch();
return 0;
}
```

ফলাফল : Bangladesh  
Bangladesh

এবারে নিম্নে কিছু প্রোগ্রামের কোড উপরোক্ত তিনটি লুপ দিয়ে তুলনামূলকভাবে করা হলো:

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

| 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; #include&lt;conio.h&gt;</pre> | <p>1 থেকে 15 পর্যন্ত জোড় সংখ্যা দেখানোর জন্য প্রোগ্রাম।</p> <pre>#include&lt;stdio.h&gt; #include&lt;conio.h&gt;</pre> | <p>1 থেকে 15 পর্যন্ত জোড় সংখ্যা দেখানোর জন্য প্রোগ্রাম।</p> <pre>#include&lt;stdio.h&gt; #include&lt;conio.h&gt;</pre> |

|                                                                                                                                                                                                                                                                                                               |                                                                                                                                                                                                                                                                                                                      |                                                                                                                                                                                                                                                                                                                          |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|--------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre> int main() { int a; for(a=2;a&lt;=15; a=a+2) {     printf("%d\t",a); } getch(); } ফলাফল : 2 4 6 8 10 12 14 </pre>                                                                                                                                                                                       | <pre> int main() { int a=2; while(a&lt;=15) {     printf("%d\t",a);     a=a+2; } getch(); } ফলাফল : 2 4 6 8 10 12 14 </pre>                                                                                                                                                                                          | <pre> int main() { int a=2; do {     printf("%d\t",a);     a=a+2; } while(a&lt;=15); getch(); } ফলাফল : 2 4 6 8 10 12 14 </pre>                                                                                                                                                                                          |
| <p><b>1</b> থেকে <b>n</b> পর্যন্ত সংখ্যা দেখানোর জন্য প্রোগ্রাম।</p> <pre> #include&lt;stdio.h&gt; #include&lt;conio.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); } getch(); }  ফলাফল : Value of n:10 1 2 3 4 5 6 7 8 9 10 </pre> | <p><b>1</b> থেকে <b>n</b> পর্যন্ত সংখ্যা দেখানোর জন্য প্রোগ্রাম।</p> <pre> #include&lt;stdio.h&gt; #include&lt;conio.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; } getch(); }  ফলাফল : Value of n:10 1 2 3 4 5 6 7 8 9 10 </pre> | <p><b>1</b> থেকে <b>n</b> পর্যন্ত সংখ্যা দেখানোর জন্য প্রোগ্রাম।</p> <pre> #include&lt;stdio.h&gt; #include&lt;conio.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); getch(); }  ফলাফল : Value of n:10 1 2 3 4 5 6 7 8 9 10 </pre> |
| <p><b>1</b> থেকে <b>100</b> পর্যন্ত সংখ্যার যোগফল দেখানোর জন্য প্রোগ্রাম।<br/>অথবা<br/><b>1+2+3+.... . . . . . +100</b><br/>ধারার যোগফল দেখানোর জন্য প্রোগ্রাম।</p> <pre> #include&lt;stdio.h&gt; #include&lt;conio.h&gt; int main() </pre>                                                                   | <p><b>1</b> থেকে <b>100</b> পর্যন্ত সংখ্যার যোগফল দেখানোর জন্য প্রোগ্রাম।<br/>অথবা<br/><b>1+2+3+.... . . . . . +100</b><br/>ধারার যোগফল দেখানোর জন্য প্রোগ্রাম।</p> <pre> #include&lt;stdio.h&gt; #include&lt;conio.h&gt; int main() </pre>                                                                          | <p><b>1</b> থেকে <b>100</b> পর্যন্ত সংখ্যার যোগফল দেখানোর জন্য প্রোগ্রাম।<br/>অথবা<br/><b>1+2+3+.... . . . . . +100</b><br/>ধারার যোগফল দেখানোর জন্য প্রোগ্রাম।</p> <pre> #include&lt;stdio.h&gt; #include&lt;conio.h&gt; int main() </pre>                                                                              |

|                                                                                                                   |                                                                                                                       |                                                                                                                           |
|-------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------|
| <pre>{ int a,s=0; for(a=1;a&lt;=100; a=a+1) {     s=s+a; } printf("Sum=%d ",s); getch(); } ফলাফল : Sum=5050</pre> | <pre>{ int a=1,s=0; while(a&lt;=100) {     s=s+a;     a=a+1; } printf("Sum=%d ",s); getch(); } ফলাফল : Sum=5050</pre> | <pre>{ int a=1,s=0; do {     s=s+a;     a=a+1; } while(a&lt;=100); printf("Sum=%d ",s); getch(); } ফলাফল : Sum=5050</pre> |
|-------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------|

| for loop ব্যবহার করে                                                                                                                                                                                                                                                                                                                                       | while ব্যবহার করে                                                                                                                                                                                                                                                                                                                                                   | do-while ব্যবহার করে                                                                                                                                                                                                                                                                                                                                                    |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p><b>1 থেকে 100 পর্যন্ত বিজোড় সংখ্যার যোগফল দেখানোর জন্য প্রোগ্রাম।</b></p> <p>অথবা</p> <p><b>১+৩+৫+..... . . . . . +১০০</b></p> <p>ধারার যোগফল দেখানোর জন্য প্রোগ্রাম।</p> <pre>#include&lt;stdio.h&gt; #include&lt;conio.h&gt; int main() { int a,s=0; for(a=1;a&lt;=100; a=a+2) {     s=s+a; } printf("Sum=%d ",s); getch(); } ফলাফল : Sum=2500</pre> | <p><b>1 থেকে 100 পর্যন্ত বিজোড় সংখ্যার যোগফল দেখানোর জন্য প্রোগ্রাম।</b></p> <p>অথবা</p> <p><b>১+৩+৫+..... . . . . . +১০০</b></p> <p>ধারার যোগফল দেখানোর জন্য প্রোগ্রাম।</p> <pre>#include&lt;stdio.h&gt; #include&lt;conio.h&gt; int main() { int a=1,s=0; while(a&lt;=100) {     s=s+a;     a=a+2; } printf("Sum=%d ",s); getch(); } ফলাফল : Sum= Sum=2500</pre> | <p><b>1 থেকে 100 পর্যন্ত বিজোড় সংখ্যার যোগফল দেখানোর জন্য প্রোগ্রাম।</b></p> <p>অথবা</p> <p><b>১+৩+৫+..... . . . . . +১০০</b></p> <p>ধারার যোগফল দেখানোর জন্য প্রোগ্রাম।</p> <pre>#include&lt;stdio.h&gt; #include&lt;conio.h&gt; int main() { int a=1,s=0; do {     s=s+a;     a=a+2; } while(a&lt;=100); printf("Sum=%d ",s); getch(); } ফলাফল : Sum= Sum=2500</pre> |
| <p><b>1 থেকে 100 পর্যন্ত জোড় সংখ্যার যোগফল দেখানোর জন্য প্রোগ্রাম।</b></p> <p>অথবা</p> <p><b>২+৪+৬+..... . . . . . +১০০</b></p> <p>ধারার যোগফল দেখানোর জন্য প্রোগ্রাম।</p> <pre>#include&lt;stdio.h&gt; #include&lt;conio.h&gt; int main() { int a,s=0;</pre>                                                                                             | <p><b>1 থেকে 100 পর্যন্ত জোড় সংখ্যার যোগফল দেখানোর জন্য প্রোগ্রাম।</b></p> <p>অথবা</p> <p><b>২+৪+৬+..... . . . . . +১০০</b></p> <p>ধারার যোগফল দেখানোর জন্য প্রোগ্রাম।</p> <pre>#include&lt;stdio.h&gt; #include&lt;conio.h&gt; int main() { int a=2,s=0;</pre>                                                                                                    | <p><b>1 থেকে 100 পর্যন্ত জোড় সংখ্যার যোগফল দেখানোর জন্য প্রোগ্রাম।</b></p> <p>অথবা</p> <p><b>২+৪+৬+..... . . . . . +১০০</b></p> <p>ধারার যোগফল দেখানোর জন্য প্রোগ্রাম।</p> <pre>#include&lt;stdio.h&gt; #include&lt;conio.h&gt; int main() { int a=2,s=0;</pre>                                                                                                        |

|                                                                                                                                                                                                                                                                                                                                                                 |                                                                                                                                                                                                                                                                                                                                                                         |                                                                                                                                                                                                                                                                                                                                                                             |
|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre> for(a=2;a&lt;=100; a=a+2) {     s=s+a; } printf("Sum=%d ",s); getch(); } ফলাফল : Sum=2550 </pre>                                                                                                                                                                                                                                                          | <pre> while(a&lt;=100) {     s=s+a;     a=a+2; } printf("Sum=%d ",s); getch(); } ফলাফল : Sum=2550 </pre>                                                                                                                                                                                                                                                                | <pre> do {     s=s+a;     a=a+2; } while(a&lt;=100); printf("Sum=%d ",s); getch(); } ফলাফল : Sum=2550 </pre>                                                                                                                                                                                                                                                                |
| <p><b>1 থেকে n পর্যন্ত জোড় সংখ্যার যোগফল দেখানোর জন্য প্রোগ্রাম।</b><br/>অথবা<br/><b>১+২+৩+.... ... ... +n ধারার যোগফল দেখানোর জন্য প্রোগ্রাম।</b></p> <pre> #include&lt;stdio.h&gt; int main() { int a,s=0,n; printf("Value of n: "); scanf("%d",&amp;n); for(a=1;a&lt;=n; a++) {     s=s+a; } printf("Sum=%d ",s); } ফলাফল : Value of n: 100 Sum=5050 </pre> | <p><b>1 থেকে n পর্যন্ত জোড় সংখ্যার যোগফল দেখানোর জন্য প্রোগ্রাম।</b><br/>অথবা<br/><b>১+২+৩+.... ... ... +n ধারার যোগফল দেখানোর জন্য প্রোগ্রাম</b></p> <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+1; } printf("Sum=%d ",s); } ফলাফল : Value of n: 100 Sum=5050 </pre> | <p><b>1 থেকে n পর্যন্ত জোড় সংখ্যার যোগফল দেখানোর জন্য প্রোগ্রাম।</b><br/>অথবা<br/><b>১+২+৩+.... ... ... +n ধারার যোগফল দেখানোর জন্য প্রোগ্রাম</b></p> <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+1; } while(a&lt;=n); printf("Sum=%d ",s); } ফলাফল : Value of n: 100 Sum=5050 </pre> |
| <p><b>for loop ব্যবহার করে</b></p> <p><b><math>1^2+2^2+3^2+.....+n^2</math> ধারার যোগফল নির্ণয়ের জন্য প্রোগ্রাম লেখ।</b></p> <pre> #include&lt;stdio.h&gt; #include&lt;conio.h&gt; main() { int a,s=0,n; printf("Value of n: "); scanf("%d",&amp;n); </pre>                                                                                                    | <p><b>while loop ব্যবহার করে</b></p> <p><b><math>1^2+2^2+3^2+.....+n^2</math> ধারার যোগফল নির্ণয়ের জন্য প্রোগ্রাম লেখ।</b></p> <pre> #include&lt;stdio.h&gt; #include&lt;conio.h&gt; int main() { int a,s=0,n; printf("Value of n: "); scanf("%d",&amp;n); a=1; </pre>                                                                                                 | <p><b>do-while ব্যবহার করে</b></p> <p><b><math>1^2+2^2+3^2+.....+n^2</math> ধারার যোগফল নির্ণয়ের জন্য প্রোগ্রাম লেখ।</b></p> <pre> #include&lt;stdio.h&gt; #include&lt;conio.h&gt; int main() { int a,s=0,n; printf("Value of n: "); scanf("%d",&amp;n); </pre>                                                                                                            |

|                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                       |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                                           |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre>for(a=1;a&lt;=n;a++) {     s=s+a*a; } printf("Sum: %d",s); getch(); }</pre> <p><b>ফলাফল :</b><br/>Value of n:100<br/>Sum:10670</p>                                                                                                                                                                                                                                                                                                                                                                                                        | <pre>while(a&lt;=n) {     s=s+a*a;     a=a+1; } printf("Sum=%d ",s);</pre> <p><b>ফলাফল :</b><br/>Value of n:100<br/>Sum of the series:10670</p>                                                                                                                                                                                                                                                                                                                                                                                                       | <pre>a=1; do {     s=s+a*a;     a=a+1; } while(a&lt;=n); printf("Sum=%d ",s);</pre> <p><b>ফলাফল :</b><br/>Value of n:100<br/>Sum:10670</p>                                                                                                                                                                                                                                                                                                                                                                                                                |
| <p>এমন একটি ধারা নির্ণয় করতে হবে<br/>যার প্রথম পদ, প্রতি পদের বৃদ্ধি<br/>এবং শেষপদ কীভোর্ডের মাধ্যমে<br/>ইনপুট দিতে হবে।</p> <pre>#include&lt;stdio.h&gt; #include&lt;conio.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); } getch(); }</pre> <p><b>ফলাফল :</b><br/>First term:2<br/>Increment number:2<br/>Last term:10<br/>Series: 2 4 6 8 10</p> | <p>এমন একটি ধারা নির্ণয় করতে হবে<br/>যার প্রথম পদ, প্রতি পদের বৃদ্ধি<br/>এবং শেষপদ কীভোর্ডের মাধ্যমে<br/>ইনপুট দিতে হবে।</p> <pre>#include&lt;stdio.h&gt; #include&lt;conio.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);     j=j+i; } getch(); }</pre> <p><b>ফলাফল :</b><br/>First term:2<br/>Increment number:2<br/>Last term:10<br/>Series: 2 4 6 8 10</p> | <p>এমন একটি ধারা নির্ণয় করতে হবে<br/>যার প্রথম পদ, প্রতি পদের বৃদ্ধি<br/>এবং শেষপদ কীভোর্ডের মাধ্যমে<br/>ইনপুট দিতে হবে।</p> <pre>#include&lt;stdio.h&gt; #include&lt;conio.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);     j=j+i; } while(j&lt;=n); getch(); }</pre> <p><b>ফলাফল :</b><br/>First term:2<br/>Increment number:2<br/>Last term:10<br/>Series: 2 4 6 8 10</p> |

| <b>for loop</b> ব্যবহার করে                                                                                                                                                                                                                                                                                                                             | <b>while loop</b> ব্যবহার করে                                                                                                                                                                                                                                                                                                                                    | <b>do-while</b> ব্যবহার করে                                                                                                                                                                                                                                                                                                                                           |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p><b>1.2+2.3+3.4+.....+n(n+1)</b><br/>ধারার যোগফল নির্ণয়ের জন্য<br/>প্রোগ্রাম লেখ।</p> <pre>#include&lt;stdio.h&gt; #include&lt;conio.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); getch(); }</pre> <p><b>ফলাফল :</b><br/>Value of n:100<br/>Sum:343400</p> | <p><b>1.2+2.3+3.4+.....+n(n+1)</b><br/>ধারার যোগফল নির্ণয়ের জন্য প্রোগ্রাম<br/>লেখ।</p> <pre>#include&lt;stdio.h&gt; #include&lt;conio.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); getch(); }</pre> <p><b>ফলাফল :</b><br/>Value of n:100<br/>Sum: 343400</p> | <p><b>1.2+2.3+3.4+.....+n(n+1)</b><br/>ধারার যোগফল নির্ণয়ের জন্য<br/>প্রোগ্রাম লেখ।</p> <pre>#include&lt;stdio.h&gt; #include&lt;conio.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); getch(); }</pre> <p><b>ফলাফল :</b><br/>Value of n: 100<br/>Sum: 343400</p> |
| <p><b>কোন সংখ্যার ফ্যাকটোরিয়াল নির্ণয় করার প্রোগ্রাম।</b></p> <pre>#include&lt;stdio.h&gt; #include&lt;conio.h&gt; main() { int n, i; long f=1; printf("Enter a number: "); scanf("%d", &amp;n); for(i=1; i&lt;=n; i++) {     f=f*i; } printf("Factorial is %ld", f); getch(); }</pre> <p><b>ফলাফল :</b> Enter a number : 6<br/>Factorial is 720</p>  | <p><b>কোন সংখ্যার ফ্যাকটোরিয়াল নির্ণয় করার প্রোগ্রাম।</b></p> <pre>#include&lt;stdio.h&gt; #include&lt;conio.h&gt; main() { int n, i=1; long f=1; printf("Enter a number: "); scanf("%d", &amp;n); while(i&lt;=n){     f=f*i;     i++; } printf("Factorial is %ld", f); getch(); }</pre> <p><b>ফলাফল :</b> Enter a number : 6<br/>Factorial is 720</p>         | <p><b>কোন সংখ্যার ফ্যাকটোরিয়াল নির্ণয় করার প্রোগ্রাম।</b></p> <pre>#include&lt;stdio.h&gt; #include&lt;conio.h&gt; main() { int n, i=1; long f=1; printf("Enter a number: "); scanf("%d", &amp;n); do {     f=f*i;     i++; } while(i&lt;=n); printf("Factorial is %ld", f); getch(); }</pre> <p><b>ফলাফল :</b> Enter a number : 6<br/>Factorial is 720</p>         |

| for loop ব্যবহার করে                                                                                                                                                                                                                                                                                                                                                                                                                                                               | while loop ব্যবহার করে                                                                                                                                                                                                                                                                                                                                                                                                        | do-while ব্যবহার করে                                                                                                                                                                                                                                                                                                                                                                                                              |
|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <p>কিরোড়ের সাহায্যে গৃহীত দুটি পূর্ণ সংখ্যার গ.সা.গু নির্ণয়ের প্রোগ্রাম</p> <pre>#include &lt;stdio.h&gt; #include&lt;conio.h&gt; int main() { int l, s, i, gcd; printf("Enter lange value: "); 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; getch(); }</pre> <p>ফলাফল : Enter large value : 35<br/>Enter small value :25<br/>GCD = 5</p> | <p>কিরোড়ের সাহায্যে গৃহীত দুইটি পূর্ণ সংখ্যার গ.সা.গু নির্ণয়ের প্রোগ্রাম।</p> <pre>#include&lt;stdio.h&gt; #include&lt;conio.h&gt; main() { int l, s, r; printf("Enter large value :"); 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); getch(); }</pre> <p>ফলাফল : Enter large value : 35<br/>Enter small value :25<br/>GCD = 5</p> | <p>কিরোড়ের সাহায্যে গৃহীত দুইটি পূর্ণ সংখ্যার গ.সা.গু নির্ণয়ের প্রোগ্রাম।</p> <pre>#include&lt;stdio.h&gt; #include&lt;conio.h&gt; main() { int l, s, r; printf("Enter large value :"); 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); getch(); }</pre> <p>ফলাফল : Enter large value : 35<br/>Enter small value :25<br/>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 num1, num2,num3, i, hcf; printf("Enter first number integers: "); scanf("%d", &amp;num1); printf("Enter second integers: "); scanf("%d", &amp;num2); printf("Enter third integers: "); scanf("%d",&amp;num3); for(i=1; i&lt;=num1    i&lt;=num2   i&lt;=num3; ++i) { if(num1%i==0 &amp;&amp; num2%i==0 &amp;&amp; num3%i==0)}</pre> | <p>দুটি পূর্ণ সংখ্যার ল.সা.গু নির্ণয়ের প্রোগ্রাম</p> <pre>#include&lt;stdio.h&gt; #include&lt;conio.h&gt; main() { int a,b,i,flag=0; printf("Type the two number:"); scanf("%d %d",&amp;a,&amp;b); for(i=1;i&lt;=a*b &amp;&amp; flag==0;i++) { if(i%a == 0 &amp;&amp; i%b == 0 ) { flag++; printf("LCM is %d",i); } }</pre> |
|-------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|

|                                                                                                                                                                                                                                                                                                                                                                                                                                                               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |
|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| <pre> hcf=i; } printf("H.C.F of %d ,%d and %d is %d",        num1,num2,num3,hcf); return 0; }  ফলাফল : Enter first number integers: 15 Enter second integers: 10 Enter third integers:5 H.C.F of 15 , 10 and 5 is 5 </pre>                                                                                                                                                                                                                                    | <pre> getch(); }  ফলাফল : Type the two number: 7 3 LCM is 21 </pre>                                                                                                                                                                                                                                                                                                                                                                                                                     |
| <p>তিনটি পূর্ণ সংখ্যার ল.সা.গু নির্ণয়ের প্রোগ্রাম</p> <pre> #include&lt;stdio.h&gt; #include&lt;conio.h&gt; main() { int a,b,c,i,flag=0; printf("Type the three number:"); scanf("%d %d %d",&amp;a,&amp;b,&amp;c); for(i=1;i&lt;=a*b*c &amp;&amp; flag==0;i++) {     if(i%a == 0 &amp;&amp; i%b == 0 &amp;&amp; i%c     == 0)     {         flag++;         printf("LCM is %d",i);     } } getch(); }  ফলাফল : Type the three number:3 9 12 LCM is 36 </pre> | <p>কোন সংখ্যা মৌলিক (Prime number) কিনা তা নির্ণয় করার প্রোগ্রাম।</p> <pre> #include&lt;stdio.h&gt; #include&lt;conio.h&gt; main() {     int n, i, s;     printf("Enter a number\n");     scanf("%d", &amp;n);     for (i=2; i&lt;=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 </pre> |
| <p>কোন ফিবোনাকি সিরিজের মান নির্ণয় করার প্রোগ্রাম।</p> <pre> #include&lt;stdio.h&gt; #include&lt;conio.h&gt; main() { int n, i, a[100];     printf("How many fibonacci number? ");     scanf("%d",&amp;n);     printf("Enter 1st &amp; 2nd number : "); </pre>                                                                                                                                                                                               |                                                                                                                                                                                                                                                                                                                                                                                                                                                                                         |

```

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

for loop এর সিনটেক্সে আমরা দেখেছিলাম for loop চারটা অংশ নিয়ে কাজ করে, initial value, condition, decrement/increment, statement। তবে এখানে প্রথম তিনটি অংশ ব্যবহার করা না করা অনেক সময় প্রোগ্রামের প্রয়োজনীয়তার উপর নির্ভর করে। যেমন control variable(count)এর মান যদি প্রোগ্রামের কোথাও আগে থেকে নির্ধারণ করা থাকে, তাহলে for loop এর প্রথম অংশ বা initial value অংশ বাদ দেয়া যায়। সেক্ষেত্রে সিনটেক্স হবে নিম্নরূপ:

counterdeclaration;

counterinitialization;

for( ;condition; increment/decrement)

{

Statement;

}

**উদাহরণ :**

```

#include<stdio.h>
#include<conio.h>
int main()
{
 int a=1; /* count এর মান for loop এর বাইরে নির্ধারণ করা হয়েছে */
 for(;a<=10; a++) /* তাই for loop এর প্রথম অংশ বাদ দেওয়া হয়েছে */
 {
 printf("%d\t",a);
 }
 getch();
}

```

**ফলাফল :** 1 2 3 4 5 6 7 8 9 10

একইভাবে প্রয়োজনে increment অংশও বাদ দেয়া যায়। এখানে উল্লেখ্য যে, লুপ এর কোনো অংশ বাদ দিলেও সেমিকোলন(;) ঠিকই ব্যবহার করতে হবে। সেক্ষেত্রে সিনটেক্স হবে নিম্নরূপ:

counter declaration;

counter initialization;

```
for(;condition;)
{
 Statement;
 increment/decrement;
}
```

উদাহরণ:

```
#include<stdio.h>
#include<conio.h>
int main()
{
 int a=1; /* initialization করা হয়েছে */
 for(; a<=10 ;) /* সেমিকোলন ব্যবহার করা হচ্ছে */
 {
 printf("%d\t",a++); /* increment অংশের কাজ এখানে হচ্ছে */
 }
 getch();
}
```

ফলাফল : 1 2 3 4 5 6 7 8 9 10

একইভাবে প্রয়োজনে *condition* অংশও বাদ দেয়া যায়। এখানে উল্লেখ্য যে, লুপ এর কোনো অংশ বাদ দিলেও সেমিকোলন (;) ঠিকই ব্যবহার করতে হবে। এক্ষেত্রে loop শুধু চলতেই থাকবে। একে অসীম (Endless Loop) লুপ বলে। সেক্ষেত্রে সিনটেক্স হবে নিম্নরূপ:

```
counter declaration;
for(counter initialization; ; increment/decrement)
{
 Statement;
}
```

উদাহরণ :

```
#include<stdio.h>
main()
{
 int a;
 for(a=1;;a++) /* মাঝের অংশ অর্থাৎ শর্ত নেই */
 {
 printf("%d",a);
 }
}
```

আবার ইচ্ছ করলে তিনটা অংশও বাদ দেয়া যায়। সেক্ষেত্রে সিনটেক্স হবে,

```
for(; ;)
{
 statement;
}
```

এক্ষেত্রে loop শুধু চলতেই থাকবে। একে অসীম (Endless Loop) লুপ বলে। যদিও for loop এই কাজটা সমর্থন করে, তবু প্রোগ্রামে এটা ব্যবহার না করাই ভালো। যদি কোন অংশই ব্যবহার না করতে হয়, তাহলে এক্ষেত্রে for loop ব্যবহার না করে অন্য loop যেমন, while loop ব্যবহার করাই ভালো। তাহলে এবারে অসীম লুপের ব্যবহার দেখি।

উদাহরণ :

```
#include<stdio.h>
#include<conio.h>
int main()
{
 int a=1; /* initialization করা হয়েছে */
 for(; ;) /* তিনটি অংশই নেই */
 {
 printf("%d\n ",a++); /* increment অংশের কাজ এখানে হচ্ছে */
 }
 getch();
}
```

**উদাহরণ :**

```
#include <stdio.h>
int main () {
 for(; ;) {
 printf("This loop will run forever.\n");
 }
 return 0;
}
```

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

**উদাহরণ :**

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

**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

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

```
for(initialization; condition; increment/decrement)
{
 for(initialization; condition; increment/decrement)
 {
```

```

 statements;
}

উদাহরণ :
#include<stdio.h>
#include<conio.h>
main()
{
int i,j,n;
printf("Enter how many line you need to make pyramid = ");
scanf("%d",&n);
for(i=1;i<=n;i++)
{
 {
 for(j=1;j<=i;j++)
 printf("%d",j);
 printf("\n");
 }
}
getch();
}

```

ফলাফল : Enter how many line you need to make pyramid =4

```

1
1 2
1 2 3
1 2 3 4

```

উদাহরণ :

```

#include<stdio.h>
#include<conio.h>
main()
{
int i,j,n;
printf("Enter how many line you need to make pyramid = ");
scanf("%d",&n);
for(i=1;i<=n;i++)
{
 {
 for(j=1;j<=i;j++)
 printf("% d",i);
 printf("\n");
 }
}
getch();
}

```

ফলাফল : Enter how many line you need to make pyramid =4

```

1
2 2
3 3 3
4 4 4 4

```

while loop এর কন্ডিশন মিথ্যা না হওয়া পর্যন্ত প্রোগ্রামে লুপ স্টেটমেন্ট কাজ করতে থাকে। তবে প্রোগ্রামে ইচ্ছা করলে লুপটা সব সময়ের জন্য সত্য করে দেয়া যায় অর্থাৎ প্রোগ্রাম যতক্ষণ চলবে কম্পিউটার শুধু এই লুপ নিয়েই কাজ করবে। একে বলে looping forever এবং এধারণের লুপকে বলে infinite loop। এক্ষেত্রে কন্ডিশন অংশে কোন এক্সপ্রেশন ব্যবহার না করে, ০ ছাড়া অন্য যে কোন সংখ্যা ব্যবহার করতে পারে।

**উদাহরণ-8:** while loop ব্যবহার করে ১ থেকে অসীম পর্যন্ত সংখ্যা প্রিন্ট করার জন্য প্রোগ্রাম।

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

```
main()
```

```
{
```

```
 int a=1;
```

```
 while(1)
```

```
{
```

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

```
 a++;
```

```
}
```

```
}
```

## Continue স্টেটমেন্ট

সি-তে শর্তযুক্ত অথবা শর্তবিহীনভাবে কোন স্টেটমেন্ট বা লুপের পুনরাবৃত্তি করার জন্য continue স্টেটমেন্ট ব্যবহৃত হয়। কোন 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।

continue স্টেটমেন্টের ফরমেট হলোঃ

```
continue;
```

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 কোন লুপের পুনরাবৃত্তি করার জন্য continue স্টেটমেন্ট ব্যবহৃত হয়। সেক্ষেত্রে শর্তের মান সত্য হলে continue স্টেটমেন্ট কার্যকরী হয়, অন্যথায় কম্পাইলর �continue স্টেটমেন্ট উপেক্ষা করে পরবর্তী স্টেটমেন্ট নির্বাহ করে। এবারে প্রোগ্রামের সাহায্যে continue স্টেটমেন্টের কাজ লক্ষ করি-

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

```
main()
```

```
{
```

```
int a;
```

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

```
{
```

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

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

```
}
```

```
}
```

কিন্তু প্রোগ্রামে continue স্টেটমেন্ট ব্যবহার করে একটু পরিবর্তন করি

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

```
main()
```

```
{
```

**ফলাফলঃ**

Thank you

Thank you

Thank you

Thank you

Thank you

```

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

### break স্টেটমেন্ট

কোনো লুপের নির্বাহ স্থগিত করে অর্থাৎ লুপ শেষ হবার আগেই লুপ থেকে বের হয়ে আসার জন্য break স্টেটমেন্ট ব্যবহৃত হয়। break স্টেটমেন্ট মোটামুটি continue স্টেটমেন্টের বিপরীত কাজ করে। break স্টেটমেন্টের ফরমেট হলো—

```
break;
```

continue স্টেটমেন্টের মত break স্টেটমেন্টের জন্যও সাধারণত if, else if, for, while ইত্যাদি স্টেটমেন্ট ব্যবহৃত হয়।

এবারে প্রোগ্রামের সাহায্যে break স্টেটমেন্টের কাজ লক্ষ করি-

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

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

অথবা

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

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

**ফলাফল:**

1  
2

উক্ত প্রোগ্রামে a এর মান 3 বা 2 এর বেশী হলে break স্টেটমেন্ট ব্যবহার করে প্রোগ্রাম নির্বাহ বন্ধ করে দেয়া হয়েছে।

এবাবে continue এবং break স্টেটমেন্ট ব্যবহার করে একটি প্রোগ্রাম দেয়া হলো-

```
#include<stdio.h>
main()
{
int a;
for(; ;)
{
printf("Enter the positive number: ");
scanf("%d",&a);
if(a<0)
 continue;
else
 break;
}
printf("\nYou have entered %d",a);
}
```

#### ফলাফল:

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

### goto স্টেটমেন্ট

সাধারণত প্রোগ্রামের এক জায়গার কাজ আপাততঃ বন্ধ রেখে, অন্য জায়গার কোন কাজ করানোর জন্য অর্থাৎ শর্তযুক্ত বা শর্তবিহীনভাবে এক স্টেটমেন্ট থেকে উপরে বা নিচে অপর কোন স্টেটমেন্টে নিয়ন্ত্রণ স্থানান্তর করার জন্য goto স্টেটমেন্ট ব্যবহার করা হয়। goto স্টেটমেন্টের ফরমেট হলো-

Levelname:

goto levelname;

এখানে, লেভেল (Levelname) একটা নাম, যা ভেরিয়েবলের নাম লেখার নিয়ম অনুসরণ করে লেখা হয় এবং এই নামের পর কোলন( : ) ব্যবহার করতে হয়। একই প্রোগ্রাম বা ফাংশনে প্রয়োজনে ভিন্ন ভিন্ন নামে একাধিক লেভেল স্টেটমেন্ট ব্যবহার করা যেতে পারে। লেভেল স্টেটমেন্ট goto স্টেটমেন্টের উপরে বা নিচে ব্যবহার করা যেতে পারে। goto স্টেটমেন্টের উপরে লেভেল স্টেটমেন্টের ব্যবহার ফলাফল continue স্টেটমেন্ট ব্যবহারের অনুরূপ। continue স্টেটমেন্ট ব্যবহার করলে কেবল উপরের দিকে জাম্প করা যায় কিন্তু goto ব্যবহার করে প্রোগ্রামের সামনে কিংবা পিছনে যে কোন স্থানে স্থানান্তর বা জাম্প করা যায়। এমনকি goto ব্যবহার করে প্রোগ্রামে ব্যবহৃত কোন ফাংশনে কিংবা অপর কোন লুপের মধ্যে অবস্থিত লেভেল স্টেটমেন্ট নির্দেশিত 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

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

প্রোগ্রামের অপর কোন স্থানে জাম্প করা যায়। সেক্ষেত্রে শর্তের মান সত্য হলে goto স্টেটমেন্ট কার্যকরী হয়। অন্যথায় কম্পাইলার goto স্টেটমেন্ট উপেক্ষে করে পরবর্তী স্টেটমেন্ট সম্পাদন করে।

**উদাহরণ:** goto স্টেটমেন্ট ব্যবহার করে কোন সংখ্যার ফ্যাক্টোরিয়াল নির্ণয়ের জন্য প্রোগ্রাম।

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

```
#include<conio.h>
```

```
main()
```

```
{
```

```
int a,x;
```

```
long fact=1;
```

```
xx:
```

#### ফলাফল:

Type the positive integer: -12  
Negative number not allowed.  
Type the positive integer:7  
Factorial=5040

```

printf("\nType the positive integer: ");
scanf("%d",&x);
if(x<0)
{
 printf("Negative number not allowed. ");
 goto xx;
}
else if(x==0)
 printf("Factorial=1");
else
{
 for(a=2;a<=x;a++)
 fact=fact*a;
 printf("\nFactorial=%ld",fact);
}
getch();
}

```



### কাজ :

- কন্ডিশনাল কন্ট্রোল স্টেটমেন্ট এবং লুপ কন্ট্রোল স্টেটমেন্টের মধ্যে পার্থক্য লেখ।
- for, while, do-while 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 |
|---|---|---|---|---|----|----|----|----|----|

- 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

## পাঠ-২৯ ও ৩০

### ব্যবহারিক : অ্যারে ও অ্যারের ব্যবহার (Array and uses of Array)

#### অ্যারে (Array)

একটি সাধারণ ভেরিয়েবলের নামের আওতায় মেমরিতে পরপর সংরক্ষিত একই টাইপের কতগুলো ডেটার সমষ্টিকে অ্যারে বা বিন্যাস বলা হয়। অন্য কথায়, একই ডেটা টাইপের কতগুলো ভেরিয়েবলের সেটকে অ্যারে বলা হয়। অ্যারে একটি ডিইভিড ডেটা টাইপ। সাধারণ ভেরিয়েবল ঘোষণার মত ব্যবহারের পূর্বে ডেটা টাইপসহ অ্যারে ভেরিয়েবল ঘোষণার প্রয়োজন হয়।

অ্যারের উপাদানগুলো মেমরিতে পাশাপাশি অবস্থান করে, ফলে একই টাইপের ডেটাগুলো মেমরিতে একত্রে থাকে বিধায় সেগুলো নিয়ে কাজ করার সময় 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) বা সার্ক্সিপেটেড ভেরিয়েবল বলা হয়। উল্লেখ্য, অ্যারের উপাদান বা সার্ক্সিপেটেড ভেরিয়েবলগুলো মেমরিতে পাশাপাশি অবস্থান করে।

যেমন, int roll [5];

এখানে roll একটি int টাইপ অ্যারে। এই অ্যারেতে মোট পাঁচটি উপাদান আছে সেগুলো হলো:

roll[0], roll[1], roll [2], roll[3] এবং roll[4]।

উল্লেখ্য, অ্যারে উপাদানগুলোর সূচক শূন্য (0) থেকে শুরু হয়। এজন্য roll অ্যারেতে roll[5] নামক কোন উপাদানের অস্তিত্ব নেই।

#### অ্যারে ভেরিয়েবলের মান নির্ধারণ

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

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

#### ঘোষণার সময়ে অ্যারে উপাদানের মান নির্ধারণ

এ প্রক্রিয়ায় অ্যারে ঘোষণার সময় অ্যারে 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 হল—

DataType ArrayName[N]= { Value1,Vale2,....., ValueN};

উদাহরণ:

char Name [6]= { ‘R’,’A’,’H’,’M’,’A’,’N’};

char টাইপ অ্যারের মান নিম্নলিখিতভাবেও করা যায়।

char Name [6] = “RAHMAN”;

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

যেমন:

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

### ঘোষণার পরে অ্যারে উপাদানের মান নির্ধারণ

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

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

#### উদাহরণ:

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

### প্রোগ্রাম নির্বাহের সময় অ্যারে উপাদানের মান নির্ধারণ

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

```
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 হল, প্রতিটি এ্যারে 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 বেশ খানিকটা অংশ সরাতে হয়।
- অ্যারেতে একই টাইপের ডেটা রাখতে হয়। ভিন্ন ডেটা টাইপের ডেটা একটি অ্যারেতে রাখা যায় না।
- এতে প্রকৃত ডেটা অপেক্ষা অ্যারের সাইজ অনেক বেশি ঘোষণা করা হলে এক দিকে যেমন মেমরির 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 যেকোন উপাদানকে সনাক্ত করার জন্য যত বা যতগুলো সংখ্যা প্রয়োজন হয় তাকে ঐ অ্যারের ডাইমেনশন বা মাত্রা বলে। int roll[50]; অ্যারের পাঁচটি উপাদানের যে কোন একটি শনাক্ত করার জন্য কেবল একটি সংখ্যা (১ 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. উচ্চ পর্যায়ের ভাষায় একজন প্রোগ্রামার কাজ করার সুবিধার জন্য মেমোরিতে ডেটা সংরক্ষণ করতে সেলের অ্যাড্রেস ব্যবহার না করে সেলে একটি নাম দেওয়া এবং ঐ নামের অধীনেই ডেটা রাখা যায়, একে চলক বলে। | ১. একই ধরণের 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 হয়। |
| ৩. চলক একটি মুহূর্তে শুধু একটি মান ধারণ করতে পারে।                                                                                                                                             | ৩. অ্যারে একটি মুহূর্তে একর 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 যে কোন বৈধ ডেটা টাইপ Array\_name প্রোগ্রামার কর্তৃক দেয়া অ্যারে ভিরিয়েবলের যে কোন বৈধ নাম এবং Array\_size পূর্ণসংখ্যায় প্রাকাশিত কোন ধূবক, 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 ভিরিয়েবলে 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 ডেটা দুভাবে রাখা যাবে। রো আকারে অথবা, কলাম আকারে।
```

|         |         |         |         |         |
|---------|---------|---------|---------|---------|
| ৬       | ৩       | ৮       | ৫       | ৯       |
| roll[0] | roll[1] | roll[2] | roll[3] | roll[4] |

|   |         |
|---|---------|
| ৬ | roll[0] |
| ৩ | roll[1] |
| ৮ | roll[2] |
| ৫ | roll[3] |
| ৯ | roll[4] |

**উদাহরণ-১:** একটি একমাত্রিক অ্যারেতে ৫টি সংখ্যা রেখে উক্ত ৫টি সংখ্যা প্রিন্ট করার প্রোগ্রাম।

```
#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।

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

```
#include<stdio.h>
int main(){
 long int decimalNumber,remainder,quotient;
 int binaryNumber[100],i=1,j;
 printf("Enter any decimal number: ");
 scanf("%ld",&decimalNumber);
 quotient = decimalNumber;
 while(quotient!=0){
 binaryNumber[i++]= quotient % 2;
 quotient = quotient / 2;
 }
 printf("Equivalent binary value of decimal number %d: ",decimalNumber);
 for(j = i - 1 ;j> 0;j--)
 printf("%d",binaryNumber[j]);
 return 0;
}
```

**ফলাফল:** Enter any decimal Number : 15

Equivalent binary value of decimal Number 15 : 101

**উদাহরণ-৪. দশমিক থেকে অষ্টালে রূপান্তর**

```
#include<stdio.h>
int main(){
 long int decimalNumber,remainder,quotient;
 int octalNumber[100],i=1,j;
 printf("Enter any decimal number: ");
 scanf("%ld",&decimalNumber);
 quotient = decimalNumber;
 while(quotient!=0){
 octalNumber[i++]= quotient % 8;
 quotient = quotient / 8;
 }
 printf("Enter any decimal number : 15
15 : 17 %d: ",decimalNumber);
```

```

for(j = i -1 ;j> 0;j--)
 printf("%d",octalNumber[j]);
return 0;
}

```

**ফলাফল** Enter any decimal number : 15

Equivalent octal value to decimal number 15 : 17

**উদাহরণ-৫.** দশমিক থেকে হেক্সাডেসিমেলে রূপান্তর

```

#include<stdio.h>
int main(){
 long int decimalNumber,remainder,quotient;
 int i=1,j,temp;
 char hexadecimalNumber[100];
 printf("Enter any decimal number: ");
 scanf("%ld",&decimalNumber);
 quotient = decimalNumber;
 while(quotient!=0){
 temp = quotient % 16;
 //To convert integer into character
 if(temp < 10)
 temp =temp + 48;
 else
 temp = temp + 55;
 hexadecimalNumber[i++]= temp;
 quotient = quotient / 16;
 }
 printf("Enter any decimal number : 52 %d: ",decimalNumber);
 52 : 34
 for(j = i -1 ;j> 0;j--)
 printf("%c",hexadecimalNumber[j]);
 return 0;
}

```

**ফলাফল:** Enter any decimal number : 52

Equivalent hexadecimal value of decimal Number 52 : 34

**উদাহরণ-৬.** বাইনারি থেকে দশমিকে রূপান্তর

```

#include<stdio.h>
int main(){
 long int binaryNumber,decimalNumber=0,j=1,remainder;
 printf("Enter any number any binary number: ");
 scanf("%ld",&binaryNumber);
 while(binaryNumber!=0){
 remainder=binaryNumber%10;
 decimalNumber=decimalNumber+remainder*j;
 binaryNumber=binaryNumber/10;
 j=j*2;
 }
 printf("Equivalent decimal value of binary number : %d",decimalNumber);
}

```

```

j=j*2;
binaryNumber=binaryNumber/10;
}
printf("Equivalent decimal value: %ld",decimalNumber);
return 0;
}

```

**ফলাফল:** Enter any number any binary number: 1101  
Equivalent decimal value: 13

#### উদাহরণ-৭. অকটাল থেকে দশমিকে রূপান্তর

```

#include<stdio.h>
#include<math.h>
int main(){
 long int octal,decimal =0;
 int i=0;
 printf("Enter any octal number: ");
 scanf("%ld",&octal);
 while(octal!=0){
 decimal = decimal + (octal % 10) * pow(8,i++);
 octal = octal/10;
 }
 printf("Equivalent decimal value: %ld",decimal);
 return 0;
}

```

**ফলাফল:** Enter any octal number: 346  
Equivalent decimal value: 230

#### উদাহরণ-৮: অকটাল থেকে বাইনারিতে রূপান্তর

```

#include<stdio.h>
#define MAX 1000
int main(){
 char octalNumber[MAX];
 long int i=0;
 printf("Enter any octal number: ");
 scanf("%s",octalNumber);
 printf("Equivalent binary value: ");
 while(octalNumber[i]){
 switch(octalNumber[i]){
 case '0': printf("000"); break;
 case '1': printf("001"); break;
 case '2': printf("010"); break;
 case '3': printf("011"); break;
 case '4': printf("100"); break;

```

```

 case '5': printf("101"); break;
 case '6': printf("110"); break;
 case '7': printf("111"); break;
 default: printf("\nInvalid octal digit %c ",octalNumber[i]); return 0;
 }
 i++;
}
return 0;
}

```

**ফলাফল:** Enter any octal number: 123

Equivalent binary value: 001010011

**উদাহরণ-৯.** বাইনারি থেকে অক্টালে রূপান্তর

```

#include<stdio.h>
int main(){
 long int binaryNumber,octal Number=0,j=1,remainder;
 printf("Enter any number any binary number: ");
 scanf("%ld",&binaryNumber);
 while(binaryNumber!=0){
 remainder=binaryNumber%10;
 octal Number=octal Number+remainder*j;
 j=j*2;
 binaryNumber=binaryNumber/10;
 }
 printf("Equivalent octal value: %ld",octalNumber);
 return 0;
}

```

**ফলাফল:** Enter any number any binary number: 1101

Equivalent octal value: 15

**উদাহরণ ১০.** হেক্সাডিসিমেল থেকে বাইনারিতে রূপান্তর

```

#include<stdio.h>
#define MAX 1000
int main(){
 char binaryNumber[MAX],hexaDecimal[MAX];
 long int i=0;
 printf("Enter any hexadecimal number: ");
 scanf("%s",hexaDecimal);
 printf("\nEquivalent binary value: ");
 while(hexaDecimal[i]){
 switch(hexaDecimal[i]){
 case '0': printf("0000"); break;
 case '1': printf("0001"); break;
 case '2': printf("0010"); break;

```

```

 case '3': printf("0011"); break;
 case '4': printf("0100"); break;
 case '5': printf("0101"); break;
 case '6': printf("0110"); break;
 case '7': printf("0111"); break;
 case '8': printf("1000"); break;
 case '9': printf("1001"); break;
 case 'A': printf("1010"); break;
 case 'B': printf("1011"); break;
 case 'C': printf("1100"); break;
 case 'D': printf("1101"); break;
 case 'E': printf("1110"); break;
 case 'F': printf("1111"); break;
 case 'a': printf("1010"); break;
 case 'b': printf("1011"); break;
 case 'c': printf("1100"); break;
 case 'd': printf("1101"); break;
 case 'e': printf("1110"); break;
 case 'f': printf("1111"); break;
 default: printf("\nInvalid hexadecimal digit %c ",hexaDecimal[i]); return 0;
 }
 i++;
}
return 0;
}

```

**ফলাফল:** Enter any hexadecimal number: 2AD5  
 Equivalent binary value: 0010101011010101

### উদাহরণ ১১. বাইনারি থেকে হেক্সাডেসিমেলে রূপান্তর

```

#include<stdio.h>
int main(){
 long int binaryNumber,hexadecimalNumber=0,j=1,remainder;
 printf("Enter any binary number: ");
 scanf("%ld",&binaryNumber);
 while(binaryNumber!=0){
 remainder=binaryNumber%10;
 hexadecimalNumber=hexadecimalNumber+remainder*j;
 j=j*2;
 binaryNumber=binaryNumber/10;
 }
 printf("Equivalent hexadecimal value: %lX",hexadecimalNumber);
 return 0;
}

```

**ফলাফল:** Enter any number any binary number: 1101  
 Equivalent hexadecimal value: D

### উদাহরণ ১২: দুটি বাইনারি সংখ্যার যোগফল

```
#include<stdio.h>
int main(){
 long int binary1,binary2;
 int i=0,remainder = 0,sum[20];
 printf("Enter any first binary number: ");
 scanf("%ld",&binary1);
 printf("Enter any second binary number: ");
 scanf("%ld",&binary2);
 while(binary1!=0||binary2!=0){
 sum[i++] = (binary1 %10 + binary2 %10 + remainder) % 2;
 remainder = (binary1 %10 + binary2 %10 + remainder) / 2;
 binary1 = binary1/10;
 binary2 = binary2/10;
 }
 if(remainder!=0)
 sum[i++] = remainder;
 --i;
 printf("Sum of two binary numbers: ");
 while(i>=0)
 printf("%d",sum[i--]);
 return 0;
}
```

**ফলাফল:** Enter any first binary number: 1100011

Enter any second binary number: 1101

Sum of two binary numbers: 1110000

**বহুমাত্রিক অ্যারে:** যে অ্যারের কোন উপাদানকে সনাক্ত করার জন্য দুই বা এর অধিক সংখ্যার প্রয়োজন হয় তাকে বহুমাত্রিক অ্যারে বলা হয়। বহুমাত্রিক অ্যারে আবার 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 এবং কলামের

গুণফলের সমপরিমাণ ডেটা রাখা যাবে। Roll[20][15] একটি দ্বি-মাত্রিক অ্যারে যেখানে 300টি ডেটা রাখা যাবে।  
**দ্বি-মাত্রিক অ্যারের গঠন নিম্নরূপ:**

**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   |

Data\_type Array\_name[rowsize][columnszie];  
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- 0,0 বা 0,1 বা 0,2 বা 1,0 বা 1,1 বা 1, 2 প্রয়োজন হবে। প্রোগ্রামে মূলত মেট্রিক্স 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 বিমাত্রিক 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 এর 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 কোন 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:** 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।



Id: [www.facebook.com/tanbir.cox](https://www.facebook.com/tanbir.cox)



Page: [www.facebook.com/tanbir.ebooks](https://www.facebook.com/tanbir.ebooks)



Web: [www.tanbircox.blogspot.com](https://www.tanbircox.blogspot.com)

## পাঠ-৩১ ও ৩২

### ব্যবহারিক : ফাংশন ও ফাংশনের ব্যবহার (Function and uses of Function)

#### ফাংশন (Function)

যখন কোনো নির্দিষ্ট কাজ সম্পাদনের জন্য কতগুলো স্টেটমেন্ট কোনো নামে একটি রুকের মধ্যে রাখা হয় তখন তাকে ফাংশন বলা হয়। যে চলক রাশির মান অন্য কোনো চলক রাশির ওপর নির্ভরশীল তাকে ফাংশন বলে। 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 বা প্যারামিটার বলা হয়। প্রতিটি ফাংশনের একটি নাম থাকে, যে নামে কম্পাইলার তাকে সনাক্ত করে। প্রোগ্রাম নির্বাহের সময়ে কম্পাইলার যখন কোন ফাংশন কল পায় তখন মূল প্রোগ্রামের কাজ স্থগিত রেখে কল্প ফাংশনে নির্বাহ শুরু করে এবং নির্বাহ শেষে মূল ফাংশনে প্রত্যাবর্তন পূর্বক পরবর্তী লাইন থেকে নির্বাহ চালিয়ে যায়। তবে এই প্রক্রিয়ায় অতিরিক্ত কিছুটা সময় ব্যয় হয়। তাই ছোট কোন 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 ভূল বের করা, সেগুলো পৃথক করা এবং 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-ডিফাইনড ফাংশন (User-Defined Function)

**লাইব্রেরি ফাংশন (Library Function) :** সি কম্পাইলারে কতগুলো বিল্ট-ইন ফাংশন আছে 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(), gethe() ইত্যাদি অন্যতম। আউটপুট অপারেশনে বহুল ব্যবহৃত লাইব্রেরি ফাংশনের মধ্যে printf(), puts(), putchar(), putch() ইত্যাদি অন্যতম।

#### প্রোগ্রামে কোন লাইব্রেরি ফাংশন ব্যবহারের পদ্ধতি

লাইব্রেরি ফাংশনগুলোর ব্যবহার সহজ, এজন্য কেবল ঐ ফাংশনের ব্যবহারবিধি এবং ফরম্যাট জানলেই চলে। লাইব্রেরি ফাংশনগুলো ঘোষণা তাদের হেডার (.h) ফাইলে এবং বিস্তারিত বর্ণনা সংশ্লিষ্ট লাইব্রেরি (.Lib) ফাইলে দেয়া থাকে। এজন্য সি প্রোগ্রামে কোন 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। এরূপ সংযোগের ফলে কম্পাইলারের এমন মনে হয় যে, সংযুক্ত ফাইল যেন সংযোগকারী ফাইলেরই অংশবিশেষ। কম্পাইলারের হেল্প ফাইল থেকে এসব ফাংশনের বিস্তারিত ব্যবহারবিধি এবং নমুনা প্রোগ্রাম দেখা যায় এবং সেখান থেকে কপি করে নিজস্ব প্রোগ্রামে ব্যবহার করা যায়। কোনো লাইব্রেরি ফাংশনের হেডার ফাইল এবং ব্যবহার বিধি কি তা জানার জন্য এ 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 থেকে। প্রোগ্রাম নির্বাহের শুরুতে main () ফাংশন স্বয়ংক্রিয়ভাবে নিয়ন্ত্রিত হয় এবং প্রয়োজনে এক বা একাধিক ফাংশন নিয়ন্ত্রণ করে। প্রোগ্রামে একটিই main () ফাংশন থাকে এবং 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 চাহিদা অনুযায়ী সব রকম ফাংশন পাওয়া যায় না। সেক্ষেত্রে প্রোগ্রামার তার নিজস্ব প্রয়োজন এবং প্রজ্ঞা অনুযায়ী যে সকল ফাংশন তৈরি করে প্রোগ্রাম তথা main() ফাংশন ব্যবহার করেন। প্রোগ্রামার কর্তৃক তৈরী এরূপ ফাংশনকে 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 এবং 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 ওপরে কিংবা নিচে থাকে কিন্তু ভেতরে নয়।

### ইউজার-ডিফাইন্ড ফাংশন নামকরণের নিয়মাবলী:

- প্রথম অক্ষর অবশ্যই অ্যালফাবেট হতে হবে। ফাংশনের নামের প্রথম অক্ষর কোনো সংখ্যা দেওয়া যাবে না।
- কোনো বিশেষ চিহ্ন ব্যবহার করা যাবে না।
- লাইব্রেরী ফাংশন বা কী-ওয়ার্ড এর সাথে নাম মিলতে পারবে না।
- ফাংশনের নামের শেষে ‘()’ দিতে হবে।
- ফাংশনের নাম তার কাজের সাথে মিল রেখে দেওয়া উচিত তবে এতে বাধ্যবাধকতা নেই।।

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

- ফাংশন ডেফিনিশন:** ফাংশন ডেফিনিশনকে ফাংশন বডিও বলা হয়। কোনো ফাংশন কী কাজ করবে, তা ফাংশন ডেফিনিশনের মধ্যে বর্ণনা করতে হয়। অর্থাৎ ফাংশন ডেফিনিশনের মধ্যেই প্রোগ্রামের কোড লিখতে হয়। ইউজার-ডিফাইন্ড ফাংশন ঘোষণার ফরম্যাট হলো

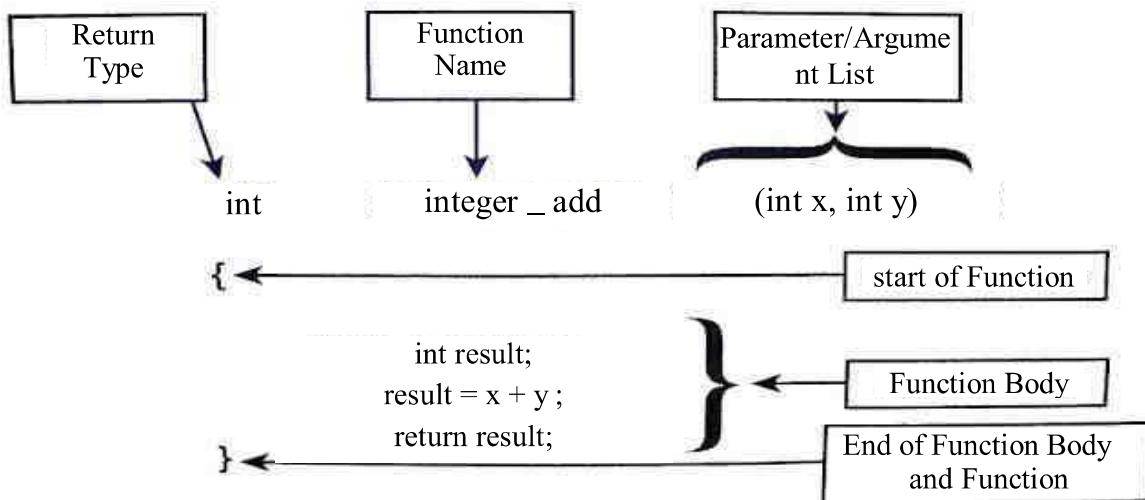
```
ReturnType FunctionName (ArgumentList)
```

```
{
```

```
 // FunctionBody
```

```
 // ReturnStatement (Depends on ReturnType)
```

```
}
```



উপরোক্ত গঠন থেকে আমরা দেখতে পাই, প্রাতাট ফাংশন ডোফনেশনের প্রথম লাইনট হলো ফাংশন হেডার। ফাংশন হেডার এর তিনটি অংশ থাকে।

- Return Type:** যে ডেটা ফাংশন রিটার্ন করে তা নির্দেশ করে।
  - Function Name:** ইউজার ডিফাইন ফাংশনের নাম নির্দেশ করে।
  - 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 তাহলো 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। সি ভাষা ও call কে প্রায় একই অর্থে ব্যবহার করা হয়েছে। প্রোগ্রামার কোনো ফাংশনকে ডিফাইন করলে তার একটা নাম দিয়ে থাকে এবং প্রতিটি ফাংশনই কোনো নির্দিষ্ট কাজ করার জন্য ডিফাইন করা 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 ফাংশনটি সি প্রোগ্রামের 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 ) ;  
নিচে একটি উদাহরণের মাধ্যমে বিষয়গুলো বোঝার চেষ্টা করি।

```

#include<stdio.h>
#include<conio.h> → ফাংশন প্রটোটাইপ
show();
main() → ফাংশন কল
{
 clrscr();
 show();
 return(0);
 getch(); → ফাংশন ডেফিনেশন
}

```

একটি ইউজার ডিফাইন্ড ফাংশনের উদাহরণ নিচে দেখানো হলো:

```
#include <stdio.h>
#include <conio.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*/
 getch();
}
int add(int a, int b) /*function call*/
{
 int add;
 add=a+b;
 return add;
}
```

```
#include <stdio.h>
int convert(int);
int main()
{
 int dec, bin;
 printf("Enter a decimal number: ");
 scanf("%d", &dec);
 bin = convert(dec);
 printf("The binary equivalent of %d is %d.\n", dec, bin);
 return 0;
}
int convert(int dec)
{
 if (dec == 0)
 {
 return 0;
 }
 else
 {
 return (dec % 2 + 10 * convert(dec / 2));
 }
}
```

### 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 এবং এই প্রক্রিয়াকে রিকার্সন বলা হয় এবং এইরূপ ফাংশন কলকে বলে রিকার্সিভ কল। রিকার্সিভ ফাংশন একটি নির্দিষ্ট অবস্থা পর্যন্ত নিজেকে কল করতে পারে এবং এর 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 না ঘটে। এই ব্যবস্থাকে বলে রিকার্সনের টার্মিনেটিং কন্ডিশন।

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

### রিকার্সিভ ফাংশনের বৈশিষ্ট্য:

- রিকার্সিভ ফাংশন নিজেই নিজেকে কল করে।
- প্রতিটি রিকার্সিভ ফাংশনের একটি টার্মিনেটিং কন্ডিশন থাকতেই হবে; তা না হলে রিকার্সিভ কল চলতেই থাকে।
- মূলত রিকার্সিভ ফরমুলা বা সিরিজ সহজে রিকার্সিভ ফাংশন দিয়ে সমাধান করা যায়।

### 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:** একটি প্রোগ্রাম লেখ যেখানে ২টি User-defined function ব্যবহার করতে হবে। একটি যোগের কাজের জন্য ও অন্যটি বিয়োগের জন্য।



## এ অধ্যায়ের প্রধান শব্দভিত্তিক সারসংক্ষেপ

- |                   |                                                                                                                                              |
|-------------------|----------------------------------------------------------------------------------------------------------------------------------------------|
| প্রোগ্রাম         | : কম্পিউটার বুবাতে পারে এমন নির্দেশমালাকে বলা হয় প্রোগ্রাম।                                                                                 |
| অনুবাদক প্রোগ্রাম | : যে প্রোগ্রামের সাহায্যে সোর্স প্রোগ্রামকে অবজেক্ট প্রোগ্রামে পরিণত করা হয় তাকে বলা হয় অনুবাদক প্রোগ্রাম।                                 |
| 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।                 |
| C প্রোগ্রাম       | : C একটি স্ট্রাকচার্ড বা প্রসিডিউর ওরিয়েন্টেড প্রোগ্রামিং ল্যাঙ্গুয়েজ।                                                                     |
| ডেটা টাইপ         | : C প্রোগ্রামে অনেক ধরনের ডেটা নিয়ে কাজ করা হয়। যেমন— পূর্ণ সংখ্যা, ভগ্নাংশ, ক্যারেক্টার, স্ট্রিং ইত্যাদি। এদেরকে সাধারণভাবে ডেটাটাইপ বলে। |
| চলক               | : ডেটাকে মেমোরিতে রাখার জন্য যে নাম ব্যবহার করা হয় তাকে চলক বলা হয়।                                                                        |
| ধূবক              | : যে রাশির মান পরিবর্তন হয় না তাকে ধূবক বলে।                                                                                                |
| অপারেটর           | : রাশিমালায় ব্যবহৃত চিহ্ন বা প্রতীককে অপারেটর বলা হয়।                                                                                      |
| অ্যারে            | : অ্যারে হলো একই ধরনের ডেটার জন্য ব্যবহৃত চলকের একটি সিরিজ।                                                                                  |
| লুপিং             | : প্রোগ্রামের অংশ বিশেষ নির্দিষ্ট সংখ্যক বার কোনো শর্তে না পৌঁছা পর্যন্ত পুনরাবৃত্তি করাকে লুপিং বলে।                                        |



## ଅନୁଶୀଳନୀ

କ. ବୃନ୍ଦାଚଳ ପ୍ରଶ୍ନ

২৫. কম্পিউটারের কৃত্রিম বুদ্ধিমত্তা ব্যবহার করা হয় কোন প্রজন্মের কম্পিউটারের ভাষায়?  
 ক. প্রথম খ. দ্বিতীয়  
 গ. তৃতীয় ঘ. পঞ্চম
২৬. কম্পিউটারের মৌলিক ভাষা কোনটি?  
 ক. মেশিন ভাষা খ. অ্যাসেম্বলি ভাষা  
 গ. দ্বিতীয় প্রজন্মের ভাষা  
 ঘ. পঞ্চম প্রজন্মের ভাষা
২৭. কোন ভাষা সবচেয়ে দুর?  
 ক. মেশিন ভাষা খ. অ্যাসেম্বলি ভাষা  
 গ. কৃত্রিম ভাষা ঘ. কম্পাইলার ভাষা
২৮. কম্পিউটারের যান্ত্রিক ভাষার বর্ণ কোনগুলো?  
 ক. ১, ২ খ. ১, ৪  
 গ. ০, ১ ঘ. ০, ২
২৯. সাংকেতিক ভাষা কোনটি?  
 ক. মেশিন ভাষা খ. যান্ত্রিক ভাষা  
 গ. অ্যাসেম্বলি ভাষা ঘ. এডা ভাষা
৩০. অ্যাসেম্বলি ভাষাকে অনুবাদ করার জন্য কোনটি ব্যবহৃত হয়?  
 ক. ইন্টারপ্রেটার খ. অ্যাসেম্বলার  
 গ. কম্পাইলার ঘ. ডিবাগার
৩১. কোন ভাষায় প্রোগ্রাম কম্পিউটারের সংগঠনের নিয়ন্ত্রণের উর্ধ্বে থাকে?  
 ক. উচ্চস্তরের ভাষা খ. নিম্নস্তরের ভাষা  
 গ. মেশিন ভাষা ঘ. কৃত্রিম ভাষা
৩২. উচ্চ স্তরের ভাষা কোনটি?  
 ক. Word Star খ. Visicalc  
 গ. C++ ঘ. Lotus 1-2-3
৩৩. মধ্যস্তরের ভাষা কোনটি?  
 ক. Word Star খ. Visicalc  
 গ. C ঘ. Lotus 1-2-3
৩৪. গাণিতিক কাজের ক্ষমতা, গাণিতিক ফাংশন তৈরির জন্য ব্যবহৃত হয় কোনটি?  
 ক. Fortran খ. SQL  
 গ. Coral-66 ঘ. QUEL
৩৫. C এর আবিষ্কারক কে?  
 ক. মার্টিন রিচার্ডস খ. লেভি এডা  
 গ. ডেনিস রিচি ঘ. বব মাইনার
৩৬. C আবিষ্কার হয় কত সালে ?  
 ক. ১৯৭০ খ. ১৯৭১  
 গ. ১৯৭২ ঘ. ১৯৭৩
৩৭. ওরাকল কী?  
 ক. DBMS খ. RDBMS  
 গ. OAK ঘ. OOP
৩৮. Algol এর পূর্ণরূপ কী?  
 ক. ALGOrithmic Language  
 খ. Algorithm Language  
 গ. Algonitre Language  
 ঘ. Algo Language
৩৯. কোন ভাষাকে কম্পিউটার প্রোগ্রামিং ভাষার জনক বলা হয়?  
 ক. C খ. C++  
 গ. Java ঘ. ORACLE
৪০. 4GL এর পূর্ণরূপ কী?  
 ক. Forth Generation Language  
 খ. Four Generation Language  
 গ. Forth General Language  
 ঘ. Four General Language
৪১. কোনটি ৪র্থ প্রজন্মের ভাষা?  
 ক. Basic খ. Pascal  
 গ. Intellect ঘ. CSL
৪২. উৎস প্রোগ্রামকে বস্তু প্রোগ্রামে পরিণত করতে যে সফটওয়্যার প্রয়োজন তাকে কী বলে?  
 ক. অপারেটিং সিস্টেম খ. অনুবাদক প্রোগ্রাম  
 গ. অ্যাসেম্বলার ঘ. অ্যাপ্লিকেশন প্রোগ্রাম
৪৩. অনুবাদক প্রোগ্রাম কত ধরনের?  
 ক. ২ ধরনের খ. ৩ ধরনের  
 গ. ৪ ধরনের ঘ. ৫ ধরনের
৪৪. হাইলেভেল ভাষার উৎস প্রোগ্রামকে বস্তু প্রোগ্রামে অনুবাদ করে কোনটি?  
 ক. কম্পাইলার খ. অ্যাসেম্বলার  
 গ. ইন্টারপ্রেটার ঘ. ডিবাগার
৪৫. কোনটি প্রোগ্রামের সব ভুলগুলো একসাথে প্রদর্শন করে?  
 ক. অ্যাসেম্বলার খ. ইন্টারপ্রেটার  
 গ. কম্পাইলার ঘ. ডিবাগার
৪৬. কোন অনুবাদক প্রোগ্রামে একবার অনুবাদের পর আর অনুবাদ করার প্রয়োজন হয় না?  
 ক. অ্যাসেম্বলার খ. ইন্টারপ্রেটার  
 গ. কম্পাইলার ঘ. ডিবাগার
৪৭. একটি প্রোগ্রামের কতটি অংশ থাকে?  
 ক. ২টি খ. ৩টি  
 গ. ৪টি ঘ. ৫টি
৪৮. ডেটা বা নির্দেশ কম্পিউটারে প্রবেশ করানো কী?  
 ক. ইনপুট খ. প্রসেস  
 গ. আউটপুট ঘ. ডিবাগিং
৪৯. প্রোগ্রামের একটি নির্দিষ্ট সমস্যা ধাপে ধাপে সমাধান করার পদ্ধতিকে কী বলে?  
 ক. অ্যালগরিদম খ. অ্যালজেব্রা  
 গ. ফ্লোচার্ট ঘ. অ্যালগল
৫০. সমস্যা সমাধানের জন্য প্রোগ্রাম তৈরির ধাপ কয়টি ?  
 ক. দুইটি খ. তিনটি  
 গ. চারটি ঘ. পাঁচটি
৫১. প্রোগ্রামের ভুলকে কী বলে ?  
 ক. বাগ খ. ফ্লোচার্ট  
 গ. সুড়োকোড ঘ. লজিক
৫২. প্রোগ্রামের ভুল সংশোধনকে কী বলে ?  
 ক. বাগ খ. ফ্লোচার্ট  
 গ. সুড়োকোড ঘ. ডিবাগিং





৯৩. সি ভাষায় #include<stdio.h> হেডার ফাইলের অন্তর্গত বিভিন্ন ফাংশন হল—

i. printf()      ii. scanf()

iii. gest()

নিচের কোনটি সঠিক?

ক. i ও ii      খ. i ও iii

গ. ii ও iii      ঘ. i, ii ও iii

৯৪. সি ভাষায় ব্যবহৃত বিভিন্ন হেডার ফাইল হলো —

i. #include<scan.h>

ii. #include<conio.h>

iii. #include<stdio.h>

নিচের কোনটি সঠিক?

ক. i ও ii      খ. i ও iii

গ. ii ও iii      ঘ. i, ii ও iii

৯৫. প্রোগ্রাম তৈরির ধাপগুলো হলো—

i. প্রোগ্রাম ডিজাইন ii. প্রোগ্রাম বাস্তবায়ন

iii. প্রোগ্রাম রক্ষণাবেক্ষণ

নিচের কোনটি সঠিক?

ক. i ও ii      খ. i ও iii

গ. ii ও iii      ঘ. i, ii ও iii

#### ► অভিন্ন তথ্যভিত্তিক প্রশ্ন ও উত্তর

নিচের প্রোগ্রামটি দেখ এবং পরবর্তী দুটি প্রশ্নের উত্তর দাও।

```
main()
{
float a=10.7;
float b=11.5;
int c;
c=a+b;
printf("sum=%d");
}
```

৯৬. স্টেটমেন্টটির ফলাফল কোনটি?

ক. 24      খ. 23

গ. 22.2      ঘ. 22

৯৭. প্রোগ্রামে সঠিক ফলাফল না আসার কারণ-

i. ইস্পেক সিকুয়েন্স ii. ফরমেট স্পেসিফিকেশন

iii. ফাংশন

নিচের কোনটি সঠিক?

ক. i      খ. ii

গ. i ও ii      ঘ. i, ii ও iii

নিচের উদ্দীপকটি পড়ো এবং ৯৮ ও ৯৯ নং প্রশ্নের উত্তর দাও:

রিয়া টিচার্স ট্রেনিং কলেজের কম্পিউটার শিক্ষকের কাছ থেকে উচ্চস্তরের ভাষা সম্পর্কে কিছু তথ্য লাভ করে। সে উচ্চস্তরের ভাষার সুবিধা অসুবিধা সম্পর্কে তথ্য জানতে পারে এবং উচ্চস্তরের বিভিন্ন ভাষা সম্পর্কেও জ্ঞান লাভ করে।

৯৮. রিয়া উচ্চস্তরের কোন ভাষাটি সম্পর্কে জ্ঞান লাভ করে?

ক. ইন্টারপ্রেটার      খ. অ্যাসেম্বলার

গ. ভিজুয়াল বেসিক      ঘ. মেশিন ভাষা

৯৯. উচ্চস্তরের ভাষার সুবিধা হলো—

i. প্রোগ্রাম লিখতে সহজ হয় এবং লিখতে কম সময় লাগে

ii. সরাসরি কম্পিউটারের সাথে যোগাযোগ করা যায়

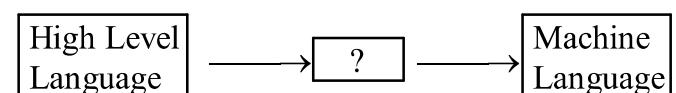
iii. ভুল হবার সম্ভাবনা কম থাকে এবং প্রোগ্রামের ত্রুটি বের করে তা সংশোধন করা সহজ হয়

নিচের কোনটি সঠিক?

ক. i ও ii      খ. i ও iii

গ. ii ও iii      ঘ. i, ii ও iii

নিচের উদ্দীপকটি পড়ো এবং ১০০ ও ১০১ নং প্রশ্নের উত্তর দাও:



১০০. উদ্দীপকে ? চিহ্নিত অংশে নিচের কোনটি ব্যবহার করা হয়?

ক. HTML      খ. ASSEMBLY

গ. C      ঘ. TRANSLATOR

১০১. Natural Language এর জন্য উদ্দীপকে ? চিহ্নিত অংশে নিচের কোনটি ব্যবহার করা উচিত?

ক. Adder      খ. Assembler

গ. Compiler      ঘ. Intelligent Compiler

নিচের উদ্দীপকটি পড়ো এবং ১০২ ও ১০৩ নং প্রশ্নের উত্তর দাও।

আরিফ তার কম্পিউটার 'সি' প্রোগ্রামের সময় অনেকগুলো ডেটা নিয়ে কাজ করে যেমন : Char, int, float, double। আবার প্রয়োজনে সে নিজস্ব ডেটা টাইপ তৈরি করে নেয়।

১০২. আরিফের তৈরি করা ডেটা টাইপকে কী বলে?

ক. মৌলিক ডেটা টাইপ

খ. ইউজার ডিফাইনড ডেটা টাইপ

গ. ডিরাইভড ডেটা টাইপ

ঘ. ফাঁকা ডেটা টাইপ

১০৩. আরিফ যে চারটি ডেটা নিয়ে কাজ করে এদেরকে বলা হয়—

ি. বেসিক      ii. মৌলিক

iii. বিল্ট-ইন ডেটা

নিচের কোনটি সঠিক?

ক. i ও ii      খ. i ও iii

গ. ii ও iii      ঘ. i, ii ও iii

নিচের অনুচ্ছেদটি পড়ো এবং ১০৪ ও ১০৫ নং প্রশ্নের উত্তর দাও:

সুন্মিতা একজন কম্পিউটার সায়েন্সের ছাত্রী। সে ১ থেকে ২০ পর্যন্ত সংখ্যাগুলো প্রিন্ট করার জন্য While loop ব্যবহার করে। তার শিক্ষক তাকে এর বিকল্প হিসেবে অন্য একটি স্টেটমেন্ট ব্যবহারের কথা বলেন।

১০৪. শিক্ষক সুন্মিতাকে বিকল্প কোন স্টেটমেন্ট ব্যবহারের কথা বলেন?

ক. do while      খ. if

গ. for      ঘ. do for

১০৫. সুন্মিতার ব্যবহৃত স্টেটমেন্টের ফরম্যাট কোনটি?

ক. While (initial value; test; decrement/increment)

খ. 2 statements counter Initialization; while (Condition is true){ }

গ. do { statement}

While (condition is true)

ঘ. For (initial value; test, decrement/ increment) {Statement}

## খ. সৃজনশীল প্রশ্ন

১. কম্পিউটার সায়েন্সের ছাত্রী অনন্না ‘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 অনন্নার ছোট বোন তামান্না তাকে জিজ্ঞাসা করলো কেন তুমি উৎস কোডকে অবজেক্ট কোডে রূপান্তর করলে?

ক. অনুবাদক প্রোগ্রাম কী?

খ. কম্পাইলার ও ইন্টারপ্রিটারের মধ্যে দুটি পার্থক্য লেখ।

গ. তামান্নার প্রশ্নের উত্তর তুমি কীভাবে দিবে? বর্ণনা করো।

ঘ. X ও Y ফাইল দুটির মধ্যে কোনটি অনন্নার জন্য অনুধাবন করা সহজ? উত্তরের স্বপক্ষে যুক্তি দাও।

২. রিফাত তার বই-এর দোকানে সেসব বই-এর মূল্য ১০০ থেকে ৪০০ টাকার মধ্যে সেগুলোর ওপর ১০% কমিশন দেয়, যে সব বইয়ের মূল্য ৫০০ থেকে ১০০০ টাকার মধ্যে সেগুলোর ওপর ২০% কমিশন দেয় এবং যেসব বইয়ের মূল্য ১০০০ টাকার বেশি তার ওপর ৩০% কমিশন দেয়। রিফাত তার বন্ধু সাকিবকে তার বইয়ের বিক্রিত মূল্য বের করার জন্য প্রয়োজনীয় প্রবাহচিত্র লিখে দিতে বললো।

ক. ইন্টারপ্রিটার কী?

খ. উৎস কোডকে কম্পাইল করার প্রয়োজন হয় কেন? ব্যাখ্যা কর।

গ. যদি রিফাত সমস্ত কমিশন তুলে নিয়ে যেকোনো বইয়ের ওপর ১৫% কমিশন দেয় তাহলে রিফাতের বইয়ের বিক্রিত মূল্য নির্ণয়ের জন্য প্রোগ্রাম লেখ।

ঘ. সাকিব কীভাবে কাজটি করলো তা দেখাও।

৩. 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. প্রোগ্রাম কী?

খ. উক্ত প্রোগ্রামে & ব্যবহৃত হয়েছে কেন?

গ. উক্ত প্রোগ্রামে r ব্যবহারের সুবিধা বর্ণনা কর।

ঘ. প্রোগ্রামে a এবং r এর ব্যবহারিক পার্থক্য নিরূপণ কর।

৪. প্রোগ্রামটি দেখ এবং নিচের প্রশ্নগুলোর উত্তর দাও:

```
#include<stdio.h>
longint factorial(int n);
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));
}
```

ক. চলক কী?

খ. কখন ইউনারী অপারেটর ব্যবহার করা হয়? ব্যাখ্যা কর।

গ. উদ্দীপকে factorial যে অর্থে ব্যবহৃত হয়েছে তার বৈশিষ্ট্য ও সুবিধা লেখ।

ঘ. প্রোগ্রামে যে সব ইনপুট আউটপুট ফাংশন ব্যবহৃত হয়েছে তাদেরকে ফরমেটেড ইনপুট আউটপুট ফাংশন বলা যেতে পারে কি? উত্তরের স্বপক্ষে যুক্তি দাও।

৫. প্রোগ্রামটি দেখ এবং নিচের প্রশ্নগুলোর উত্তর দাও:

```
main()
```

```
{
 int a;
```

```
for(a=1; a<=10;a++)
```

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

ক. ফরমেট স্পেসিফিয়ার 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।

৬. আদনান জামি দুটি সংখ্যা L, S ( $L > S$ ) এর গ.স.গু নির্ণয়ের জন্য ‘সি’ ভাষা প্রোগ্রাম করতে চাচ্ছে। কিন্তু সে প্রোগ্রামটির লজিক কিছুই বুঝতে পারছে না। অবশেষে সে তার আইসিটি শিক্ষকের স্মরণাপন হলেন। তার শিক্ষক তাকে সমস্যাটি কয়েকটি ধাপে ভেঙে প্রত্যেকটি ধাপের চিত্রসহকারে উপস্থাপন করে তাকে বুঝিয়ে দিলেন। এখন আদনান জামির আর কোনো সমস্যা রাখল না।

ক. প্রোগ্রামিং কী?

খ. প্রোগ্রামারগন কোনো বড় প্রোগ্রামকে ছোট ছোট ভাগে ভাগ করে কি সুবিধা পান? বুঝিয়ে বল।

গ. শিক্ষক হিসেবে তুমি সমস্যাটির সমাধান দাও।

ঘ. L= 8 এবং S=3 হলে উক্ত ধাপগুলো কীভাবে কাজ করবে পর্যায়ক্রমে দেখাও।

- |     |                                                                                                                                                                                                                     |                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                |
|-----|---------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|-----------------------------|------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------------|
| ৭.  | নিচের ধারাটি লক্ষ্য কর:                                                                                                                                                                                             | $7 + 14 + 21 + \dots + 100$ | ১১. $1+3^0+\dots+n^n$ এই সিরিজের যোগফল নির্ণয় করার জন্য সি ভাষায় প্রোগ্রাম লেখ।                                                                                                                                                                                                                                                                                                                                                              |
| ক.  | 'সি' ভাষার জনক কে?                                                                                                                                                                                                  |                             | #include <stdio.h><br>void main()<br>{<br>int n,c,s=();<br>scanf("%d", &n);<br>for(c=1;c<n;c++)<br>{s=s+c*c<br>printf("%d",s);                                                                                                                                                                                                                                                                                                                 |
| খ.  | 'সি' ভাষাকে কেন Mid Level ভাষা হয়? বুঝিয়ে লেখ।                                                                                                                                                                    |                             | ক. কম্পাইলার কী?<br>খ. for এবং do লুপ দুটির মধ্যে কোনটি ব্যবহার করা সহজ?<br>গ. উন্নিখিত সিরিজটির জন্য একটি ফ্লোচার্ট অংকন কর।                                                                                                                                                                                                                                                                                                                  |
| গ.  | ধারাটির ১০ম পদ নির্ণয়ের প্রোগ্রাম লেখ।                                                                                                                                                                             |                             | ঘ. সি প্রোগ্রামটিতে কী কী সমস্যা আছে তা বিশ্লেষণপূর্বক মতামত দাও।                                                                                                                                                                                                                                                                                                                                                                              |
| ঘ.  | do-while লুপ ব্যবহার করে ধারাটির যোগফল নির্ণয়ের ক্ষেত্রে লুপটি কত বার ঘুরবে তা ধারাবাহিকভাবে বিশ্লেষণ করো।                                                                                                         |                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| ৮.  | (i) $1+4+7+\dots+N$   (ii) কোন সংখ্যার ফ্যাক্টরিয়াল মান নির্ণয়ের ফরমুলা: $n!=n(n-1)(n-2)\dots$                                                                                                                    | 1                           | ১২.                                                                                                                                                                                                                                                                                                                                                                                                                                            |
| ক.  | মেশিন ভাষা কী?                                                                                                                                                                                                      |                             | #include < ><br>void main()<br>{<br>printf("enter two number");<br><br>do<br>{<br>t=i%S;<br>i=s;<br>s=t;<br>} While (s!=0)<br>printf("GCD is %d",i);<br>getch();<br>}<br>ক. ডিবাগিং কী?<br>খ. উৎস কোডকে কম্পাইল করার প্রয়োজন হয় কেন? ব্যাখ্যা কর।<br>গ. উন্দীপকের সি-প্রোগ্রামটির ফ্লোচার্ট কী ধরনের হবে—বর্ণনা কর।<br>ঘ. সঠিক প্রোগ্রামে চলক i এবং s এর মান যদি ইনপুট হিসেবে 35 ও 20 দেওয়া হয় তবে লুপের প্রতিটি ধাপে কি ঘটবে—বিশ্লেষণ কর। |
| খ.  | সি-ভাষায় চলকের নামকরণে কিছু নিয়মকানুন মেনে চলতে হয় কেন?                                                                                                                                                          |                             | ১৩. একটি কলেজের আইসিটি শিক্ষক শিক্ষার্থীকে ১ থেকে ১০০ পর্যন্ত যোগ করে যোগফল নির্ণয়ের জন্য সি প্রোগ্রাম লিখতে বললেন। শিক্ষার্থী যে প্রোগ্রামটি লিখলেন তা নিম্নরূপঃ                                                                                                                                                                                                                                                                             |
| গ.  | উন্দীপকে উন্নিখিত সমস্যার সি ভাষার প্রোগ্রাম কোড লিখ।                                                                                                                                                               |                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| ঘ.  | উন্দীপকে উন্নিখিত সমস্যার সমাধান আর কী কী ভাবে করা যেত, তুলনামূলক আলোচনা কর।                                                                                                                                        |                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| ৯.  | রিফাত তার ভাই রিয়াদকে বলল, প্রথম ১০০টি ধনাত্মক পূর্ণ সংখ্যার যোগফল কর? রিয়াদ তাকে For Loop ব্যবহার করে একটি সি প্রোগ্রাম লিখে দিল এবং বলল এটি নির্বাহ করলে যোগফলটি পাওয়া যাবে।                                   |                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| ক.  | অ্যাসেম্বলি ভাষা কী?                                                                                                                                                                                                |                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| খ.  | কী ওয়ার্ডকে ভেরিয়েবল হিসেবে ব্যবহার করা যায় না কেন?                                                                                                                                                              |                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| গ.  | উন্দীপকে উন্নিখিত সমস্যার সি ভাষার প্রোগ্রাম কোড লিখ।                                                                                                                                                               |                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| ঘ.  | উন্দীপকে উন্নিখিত সমস্যার সমাধান আর কী কী ভাবে করা যেত, তুলনামূলক আলোচনা কর।                                                                                                                                        |                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| ১০. | 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><br>#include<conio.h><br>main ()<br>{<br>int a, m, n, s = 0;<br>scanf ("%d%d", &m, &n);<br><br>for (a = m; a<=n;<br>a++)<br>{<br>s = s + a ;<br>}<br>printf ("Sum =<br>%d", s);<br>getch () ;<br>} |                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                |
| 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।                                                                                                                                                   |                             |                                                                                                                                                                                                                                                                                                                                                                                                                                                |

১৮.

```
#include<stdio.h>
#include<conio.h>
void main()
{
int i, s;
s=0;
for(i=2;i<=100;i=i+2)
{
s=s+i;
}
printf("The sum is=%d",s);
getch();
}
```

ক. ফ্লোচার্ট কী?

খ.  $i++$  এবং  $++$  ব্যাখ্যা কর।

গ. উদ্দীপকটির একটি ফ্লোচার্ট অংকন কর।

ঘ. উদ্দীপকটিতে for এর পরিবর্তে do...while ব্যবহার করলে প্রোগ্রামটির কী পরিবর্তন হবে তা বিশ্লেষণ কর যেখানে  $i$ -এর সর্বোচ্চ মান ব্যবহারকারী কর্তৃক নির্ধারিত হবে।

১৫.

$$(i) \frac{C}{5} = \frac{F - 32}{9}$$

$$(ii) 1^3 + 2^3 + \dots + N^3$$

ক. Syntex Error কাকে বলে?

খ. সি একটি উচ্চস্তরের ভাষার প্রোগ্রাম— ব্যাখ্যা কর।

গ. (i) নং উদ্দীপকের সেন্টিগ্রেডকে ফারেনহাইটের রূপান্তরের জন্য একটি ফ্লোচার্ট তৈরী কর।

ঘ. (ii) নং উদ্দীপকে উল্লিখিত সমস্যাটির ‘সি ভাষায়’ প্রোগ্রাম লিখ।

১৬. মাধবী কম্পিউটারে বসে নিম্নোক্ত প্রোগ্রামটি টাইপ করলোঁ:

```
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", &amp; a) স্টেটমেন্টটি ব্যাখ্যা কর।

গ. মাধবীর প্রোগ্রামটির প্রবাহচিত্র লিখ।

ঘ. উদ্দীপকের প্রোগ্রামটি do-while লুপ ব্যবহার করে লিখা যায় বর্ণনা কর।

১৭. একটি সরকারি বিশ্ববিদ্যালয়ের ছাত্রী তানিয়া। সে কম্পিউটার ইঞ্জিনিয়ারিং পড়ছে। তার ছোট বোন আনিকা এবার জেএসসি পরীক্ষার্থী। একদিন তানিয়া দেখল আনিকা তার গণিতের  $1+3+5+ \dots +99 = ?$  ধারাটির যোগফল নির্ণয়ের চেষ্টা করছে। কিন্তু পারছিল না। তানিয়া একটি সফটওয়্যার ব্যবহার করে সমস্যাটি সমাধান করে দিল।

ক. ফাংশন কী?

খ. প্রোগ্রাম কোডিং-এ অ্যালগরিদমের গুরুত্ব লেখ।

গ. উদ্দীপকের ধারাটি সমাধানে একটি ফ্লোচার্ট আঁক।

ঘ. উদ্দীপকের ধারায় 1, 3, 5,  $\dots$  99 এর স্থলে 2, 4, 6,  $\dots$  100 হলে ধারাটির যোগফল নির্ণয়ের সি প্রোগ্রাম লিখ।

১৮. আইসিটি বিষয়ের শিক্ষক ক্লাসে ছাত্রদের অপারেটর, চলক, ডেটা টাইপ, ও বিভিন্ন স্টেটমেন্ট সম্পর্কে পাঠ্যদান 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।

১৯. 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 কর।

২০. ধাপ-১ং শুরু কর।

ধাপ-২ং তিনটি সংখ্যা P, Q, R গ্রহণ কর।

ধাপ-৩ং P কে Q দ্বারা গুণ কর এবং গুণফলকে K-এ রাখ।

ধাপ-৪ং K কে R দিয়ে ভাগ কর এবং ভাগফল S-এ রাখ।

ধাপ-৫ং K কে ছাপাও।

ধাপ-৬ং S কে ছাপাও।

ধাপ-৭ং শেষ কর।

ক. ভেরিয়েবল কাকে বলে?

খ. গ্লোবাল ভেরিয়েবল main() ফাংশনের উপরে ব্যবহার করা হয় কেন? ব্যাখ্যা কর।

গ. উদ্দীপকে উল্লিখিত অ্যালগরিদমের জন্য একটি ফ্লোচার্ট তৈরি কর।

ঘ. উদ্দীপকে উল্লিখিত অ্যালগরিদমটির জন্য সি ভাষায় একটি প্রোগ্রাম তৈরি কর।