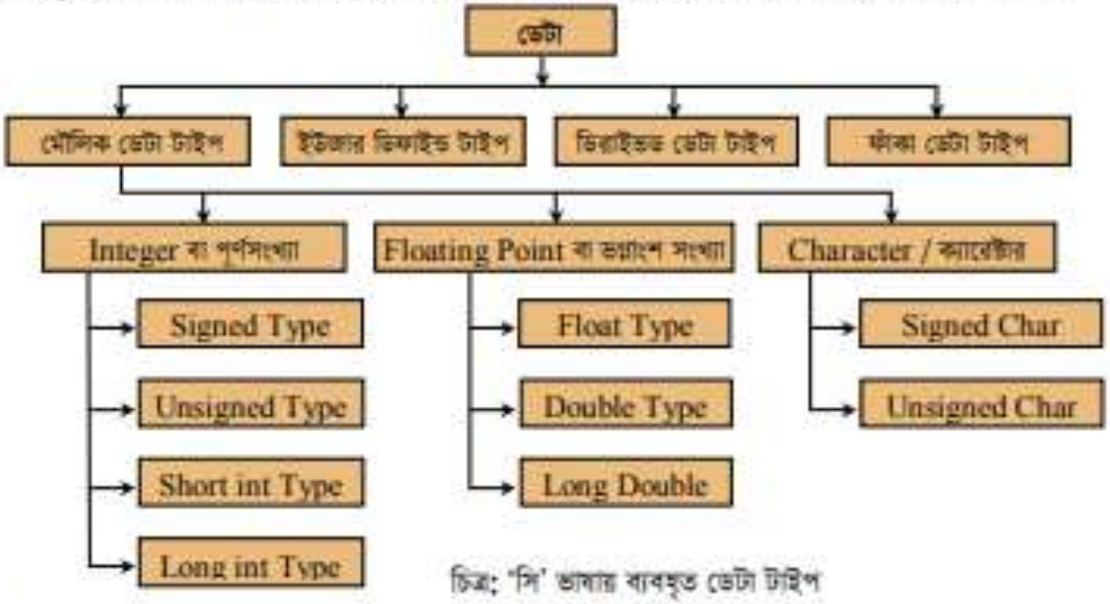
'সি' ভাষায় ব্যবহৃত ডেটা টাইপ (Data Type Uses in C Language)

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।



চিত্র: 'সি' ভাষায় ব্যবহৃত ডেটা টাইপ

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 নির্বাহকালে মেমোরিতে ch নামে এক বাইট জায়গা বরাদ্দ করে সেখানে 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,-৪৬৭, 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 নির্বাহকালে মেমোরিতে value নামে দুই বাইট জায়গা বরাদ্দ করে সেখানে 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

'সি' ভাষায় ব্যবহৃত চলক ও ধ্রুবক (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
- ধ্রুবক

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 পুনর্ব্যবহারের 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 সিনট্যাক্স বা ফরম্যাট 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, পুনর্ব্যবহার প্রভৃতি সুবিধা পাওয়া যায় না। উদাহরণ হিসেবে বলা যায়, কারো যদি লক্ষ লক্ষ বন্ধু থাকে, তবে যতই আন্তরিক হোক না কেন তারা কে কোন রুমে থাকে তা মনে রাখা সম্ভবপর নয়। কিন্তু তারা যদি তাদের নাম, রোল বা আইডি নাম্বারের অনুরূপ নামবিশিষ্ট রুমে থাকে তবে সহজেই তাদের খুঁজে বের করা সম্ভব হবে। মূলত প্রোগ্রামে ভেরিয়েবল ব্যবহারের মাধ্যমে মেমরিতে ভেরিয়েবলের নামবিশিষ্ট লোকেশনে ডেটা সংরক্ষণ করা হয়, ফলে পরবর্তী সময়ে সেগুলো খুঁজে পাওয়া সহজ হয়।

ভেরিয়েবলের স্কোপ 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 ঐ ভেরিয়েবলের স্কোপ 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 Varialable):** যখন কোনো ভেরিয়েবল কোন ফাংশনের মধ্যে অর্থাৎ ফাংশন বডিতে ঘোষণা করা হয় তখন সেই ভেরিয়েবলকে ঐ ফাংশনের সাপেক্ষে লোকাল ভেরিয়েবল বলা হয়। লোকাল ভেরিয়েবলের মান ও অস্তিত্ব শুধুমাত্র সংশ্লিষ্ট ফাংশনের মধ্যে সীমাবদ্ধ থাকে। এই মান অন্য ফাংশনে সরাসরি ব্যবহার করা যায় না। তবে লোকাল ভেরিয়েবল বিশিষ্ট কোনো ফাংশন অন্য কোনো ফাংশনে কল করে ব্যবহারকারী ফাংশনে পরোক্ষভাবে লোকাল ভেরিয়েবলের মান ব্যবহার 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 ব্লকের মধ্যেও সীমাবদ্ধ করে দেয়া যায়।
- **গ্লোবাল ভেরিয়েবল (Global Variable):** যখন কোন ভেরিয়েবল প্রোগ্রামের শুরুতে যেমন main() ফাংশনের পূর্বে ঘোষণা করা হয় তখন তাকে গ্লোবাল ভেরিয়েবল বলা হয়। গ্লোবাল ভেরিয়েবলের মান ও অস্তিত্ব কোনো নির্দিষ্ট ব্লক বা কোনো নির্দিষ্ট ফাংশনের মধ্যে সীমাবদ্ধ না থেকে পুরো প্রোগ্রামে বিস্তৃত থাকে। এ ধরনের ভেরিয়েবল ফাংশনের মধ্যে নয়; ফাংশনের উপরে ঘোষণা করা হয়। ফলে ভেরিয়েবলের মান, নাম ও ক্ষেত্র প্রোগ্রামে ব্যবহৃত সকল ফাংশনের জন্য সমানভাবে প্রযোজ্য।

লোকাল ভেরিয়েবল ও গ্লোবাল ভেরিয়েবল এর মধ্যে পার্থক্য নিম্নরূপ:

পার্থক্যের বিষয়	লোকাল ভেরিয়েবল	গ্লোবাল ভেরিয়েবল
১. সংজ্ঞা	১. কোনো ফাংশানের মধ্যে ভেরিয়েবল ডিক্লেয়ার করলে তাকে উক্ত ফাংশনের লোকাল ভেরিয়েবল বলা হয়।	১. সকল ফাংশনের বাহিরে প্রোগ্রামের শুরুতে ডিক্লেয়ার করা ভেরিয়েবলকে গ্লোবাল ভেরিয়েবল বলা হয়।
২. সীমাবদ্ধতা	২. কোনো ফাংশনের মধ্যে ডিক্লেয়ার করা লোকাল ভেরিয়েবল উক্ত ফাংশনের বাইরে ব্যবহার করা যায় না।	২. গ্লোবাল ভেরিয়েবলের কর্মকান্ড কোনো ফাংশনের মধ্যে সীমাবদ্ধ নয়।
৩. একই নাম	৩. ভিন্ন ভিন্ন ফাংশনে একই নামের লোকাল ভেরিয়েবল থাকতে পারে।	৩. একটি প্রোগ্রামের মধ্যে একই নামের একটি মাত্র গ্লোবাল 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, $3.4x10^{200}$ 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 এ্যাড্রেস 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 বন্ধনীর 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 ক্ষেত্র অতিক্রমকালে স্বয়ংক্রিয়ভাবে বরাদ্দকৃত মেমরি পরিসর খালি করে দেয়। এরূপ ভেরিয়েবলকে অটোমেটিক ভেরিয়েবল বলা হয়। অটোমেটিক ভেরিয়বল 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 দেয়া 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;
  //... ...
  }
  ```

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

ভেরিয়েবল 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 আলফাবেটিক ক্যারেক্টার (a, ... , z, A, ..., Z) হবে। ভেরিয়েবল নাম ডিজিট বা অংক দিয়ে শুরু হতে পারে না। যেমন- Roll_1 ও Roll_10 বৈধ ভেরিয়েবল; কিন্তু 1Roll 2_Roll অবৈধ।
- ভেরিয়েবলের মধ্যে স্পেশাল ক্যারেক্টার আন্ডারস্কোর (_) ও ডলার চিহ্ন ($) ব্যবহার করা যায়। আন্ডারস্কোর ও ডলার চিহ্ন ব্যতীত অন্য কোন স্পেশাল ক্যারেক্টার (যেমন !,@,#,%,*,+,- ইত্যাদি) ব্যবহার করা যায় না। যেমন, my_var, My$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 উত্তম। 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 ভেরিয়েবরের মানের রেঞ্জ অতিক্রম করলে প্রোগ্রামে ভুল ফলাফল আসতে পারে। তবে মজার ব্যাপার হলো এক্ষেত্রে কম্পাইলার কোনো সতর্ক বার্তা দেখায় না।

ধ্রুবক (Constant)

কনস্ট্যান্ট অর্থ স্থির বা ধ্রুবক 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 নির্বাহের সময় কোন অবস্থাতেই কনস্ট্যান্ট বা ধ্রুবকের 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 ধ্রুবক 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 আবশ্যক।

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

ধ্রুবক ঘোষণার জন্য সিনটেক্স (Syntax) হলো-

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

#define প্রিপ্রসেসর ব্যবহার করে ধ্রুবক 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 ধ্রুবক বা নিউমেরিক কনস্ট্যান্ট: এই ধরনের ধ্রুবক 0 থেকে 9 পর্যন্ত অংক বা S দ্বারা শুরু হয় এবং মান -2147483648 থেকে 294967295 এর মধ্যবর্তী যে কোন ঋণাত্মক বা ধনাত্মক হতে পারে। ধ্রুবকে কমা ব্যবহার করা যায় না; তবে প্রয়োজনে দশমিক চিহ্ন ব্যবহার করা যায়। সংখ্যাসূচক ধ্রুবক আবার নিম্নোক্ত পাঁচভাগে ভাগ করা যায় :

- ইন্টিজার কনস্ট্যান্ট : এ ধরনের ধ্রুবক ধনাত্মক বা ঋণাত্মক যে কোনো পূর্ণসংখ্যা হতে পারে। এ ধ্রুবকে দশমিক বিন্দু থাকে না। যেমন– 546, 45, 20000, -32768, +6532767 ইত্যাদি।
- ফ্লোটিং পয়েন্ট কনস্ট্যান্ট : এ ধরনের ধ্রুবক ধনাত্মক বা ঋণাত্মক পূর্ণ বা ভগ্নাংশবিশিষ্ট সংখ্যা পারে। পূর্ণসংখ্যার জন্য দশমিক চিহ্নের ব্যবহার আবশ্যক 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 ধ্রুবকও ধনাত্মক এবং ঋণাত্মক 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 ধনাত্মক হয় তাহলে E এর পরে যোগ (+) এবং ঋণাত্মক হলে বিয়োগ (-) ব্যবহার করা যায়। যেমন, 3.5E+3, 3.5E-5, ইত্যাদি।
- অক্টাল কনস্ট্যান্ট: এ ধরনের সংখ্যার পূর্বে একটি শূন্য (0) বসাতে হয়, যেমন- 0348, 01234 ইত্যাদি। তবে ফলাফলে এই অতিরিক্ত শূন্য অগ্রাহ্য হয়।
- হেক্সাডেসিমাল কনস্ট্যান্ট: এ ধরনের সংখ্যার পূর্বে 0x লিখতে হয়, যেমন- 0x12, 0xA2B ইত্যাদি। তবে ফলাফলে এই অতিরিক্ত 0x অগ্রাহ্য হয়।

২. অক্ষরসূচক ধ্রুবক বা স্ট্রিং কনস্ট্যান্ট: বর্ণ, অঙ্ক এবং অন্যান্য চিহ্ন সাজিয়ে এই ধ্রুবক গঠিত হয়। এই ধ্রুবককে 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 কতগুলো সুনির্দিষ্ট নিয়ম আছে। যেমন,

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

প্রোগ্রামে ধ্রুবক ব্যবহারের সুবিধা (Advantages of Constant)

নিম্নে প্রোগ্রামে ধ্রুবক ব্যবহারের কয়েকটি সুবিধা দেওয়া হলো -

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

সি ল্যাংগুয়েজে কনস্ট্যান্ট ও ভেরিয়্যাবল এর মধ্যে পার্থক্য নিম্নরূপ:

কনস্ট্যান্ট	ভেরিয়্যাবল
১. কন্সট্যান্ট অর্থ স্থির বা ধ্রুবক যা একটি নির্দিষ্ট মান ধারন করে। প্রোগ্রামে কোনো স্থির বা অপরিবর্তনশীল মান ব্যবহার করার জন্য তা কন্সট্যান্ট হিসেবে ঘোষণা করা হয়।	১. ভেরিয়্যাবল হলো একটা নাম, যে নামে কম্পাইলার নির্দিষ্ট ধরনের ডেটা রাখার জন্য মেমোরিতে জায়গা রাখে।
২. কন্সট্যান্ট 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 ভেরিয়েবল এর মান পরিবর্তন করা যায়।

রাশিমালা (Expression)

সি ভাষায় গাণিতিক এবং যৌক্তিক কাজ নিয়ন্ত্রণ করার জন্য কতগুলো বিশেষ সিম্বল (যেমন, +,-,*,/,++,--,<,>,>= ইত্যাদি) ব্যবহৃত হয়, এগুলোকে অপারেটর বলা হয়। আর যা ডেটা ধারণ করে তাকে অপারেন্ড বলা হয়। অপারেন্ড বা ডেটা ব্যবহার করে বিভিন্ন কর্ম সম্পাদনের জন্য অপারেটর ব্যবহৃত হয় এবং কতগুলো অপারেটর এবং কনস্ট্যান্টের অর্থবোধক ও সামঞ্জস্যপূর্ণ উপস্থাপনকে এক্সপ্রেশন বা বর্ণনা বলা হয়।
উদাহরণ হিসেবে বলা যায়, Average= (value1+value2) /2; একটি এক্সপ্রেশন। এখানে Average, value1, value2 অপারেন্ড; =, -, +, / অপারেটর এবং 2 কনস্ট্যান্ট।

সি-তে ব্যবহৃত অপারেটরসমূহ:

অপারেটরের সাথে সংযুক্ত অপারেন্ড বা কনস্ট্যান্টের সংখ্যার ভিত্তিতে সি প্রোগ্রামে ব্যবহৃত অপারেটর সমূহকে তিনটি প্রধান শ্রেণিতে ভাগ করা হয়। যথা:

- ইউনারি অপারেটর
- বাইনারি অপারেটর এবং
- টারনারী অপারেটর

ইউনারি অপারেটর: যে সকল অপারেটরের সাথে কেবল একটি করে অপারেন্ড 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) ;	অপারেন্ডের ধনাত্মক মান বুঝাতে ব্যবহৃত হয়।
- (ইউনারি মাইনাস)	v2= -v1; v3= -(v1+v2) ;	অপারেন্ডে ঋণাত্মক মান বুঝাতে ব্যবহৃত হয়।
++ (ইনক্রিমেন্টাল অপারেটর)	Counter ++; ++ Counter ;	অপারেন্ডের মানের সাথে ১ যোগ হয়।
-- (ডিক্রিমেন্টাল অপারেটর)	Counter --; -- Counter ;	অপারেন্ডের মান হতে ১ বিয়োগ হয়।
! (নট)	!not	শূণ্য বাদে অন্য কোনো অপারেন্ডের মান 0 করে দেয় কিন্তু শূণ্যের মান 1 করে দেয়।
~ (পূরক বা কমপ্লিমেন্ট)	~complement	অপারেন্ডের মান ১'এর পরিপূরকে রূপান্তর করে তা আবার দশমিকে রূপান্তর করে দেখায়।

ইউনারি অপারেটরগুলো পোস্টফিক্স বা প্রিফিক্স নোটেশনে কাজ করে। পোস্টফিক্স নোটেশন মানে হলো অপারেটরটা অপারেন্ডের পরে বসে। অন্যদিকে প্রিফিক্স নোটেশন মানে হলো অপারেটরটা অপারেন্ডের পূর্বে বসে। কোন কোন অপারেটরগুলো পোস্টফিক্স, প্রিফিক্স বা উভয় নোটেশনে কাজ করে তা নিচে দেওয়া হলো।

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

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

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

(condition) ? true result : false result;

এখানে condition সত্য হলে প্রোগ্রামে true result অংশ কাজ করবে অন্যথায় false result অংশ কাজ করবে।
যেমন;
x = (a > b) ? a : b, এখানে a, b এর চেয়ে বড় হলে x = a হবে, নতুবা x = b হবে। এই উদাহরণে a = 10; b = 20 হলে x এর মান হবে 20।

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 ঘোষণা
^	বিটওয়াইজ XOR	(,)	কমা
<<	শিফট লেফট	(&,*)	পয়েন্টার
>>	শিফট রাইট	(. এবং ->)	মেম্বার সিলেকশন
–	১'এর পরিপূরক		

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)

১. অ্যাসাইনমেন্ট স্টেটমেন্ট: ডেটার মান পরিবর্তন না হলে সরাসরি একটি চলকের মাধ্যমে কোনো ডেটাকে প্রকাশ করা যায়। যেমন: a = 25; b =5;

২. 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);
```