HSC ICT

KAUSHIK SAHA

TEACHING ASST. AT ISLAMIC UNIVERSITY (IIUC)

Campus : STUDY ZONE , Gate NO: 07, Block:K,

Mobile: 01832221610

৫ম অধ্যায় প্রোগ্রামিং ভাষা

সি' প্রোগ্রামিং ভাষার প্রাথমিক ধারণা

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

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

' সি' প্রোগ্রামিং ভাষা একটি স্ট্রাকচার্ড এবং প্রোসিডিউর অরিয়েন্টেড প্রোগ্রামিং ভাষা

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

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

'সি' একটি মধ্যম-স্তরের প্রোগ্রামিং ভাষা

'সি' প্রোগ্রামিং ভাষায় নিম্নস্তরের ভাষার সুবিধা; যেমন: সিস্টেম সফটওয়্যার(Driver Software) এর মাধ্যমে হার্ডওয়্যার নিয়ন্ত্রণ এবং উচ্চস্তরের ভাষার সুবিধা যেমন- অ্যাপ্লিকেশন সফটওয়্যার(Adobe Photoshop) তৈরি করা যায়। অর্থাৎ উচ্চস্তরের ভাষার সুবিধা পাওয়া যায় আবার নিম্নস্তরের ভাষার সুবিধাও পাওয়া যায়। তাই সি প্রোগ্রামিং ভাষাকে মধ্যম-স্তরের প্রোগ্রামিং ভাষা বলা হয়।

'সি' প্রোগ্রামিং ভাষার বৈশিষ্ট্যঃ

১। সকল 'সি' প্রোগ্রামের কাজ main() ফাংশন থেকে শুরু হয় এবং এটি প্রতিটি প্রোগ্রামের জন্য অত্যবশ্যকীয়।

২। 'সি' প্রোগ্রামিং ভাষা একটি case sensitive ভাষা; অর্থাৎ uppercase letter এবং lowercase letter ভিন্ন অর্থ বহন করে।

৩। 'সি' প্রোগ্রামের প্রতিটি স্টেটমেন্ট এর শেষে সেমিকোলন(;) দিতে হয়।

৪। 'সি' প্রোগ্রামিং ভাষাকে মধ্যস্তরের প্রোগ্রামিং ভাষা বলা হয়।

৫। 'সি' প্রোগ্রামিং ভাষাকে General purpose language ও বলা হয়।

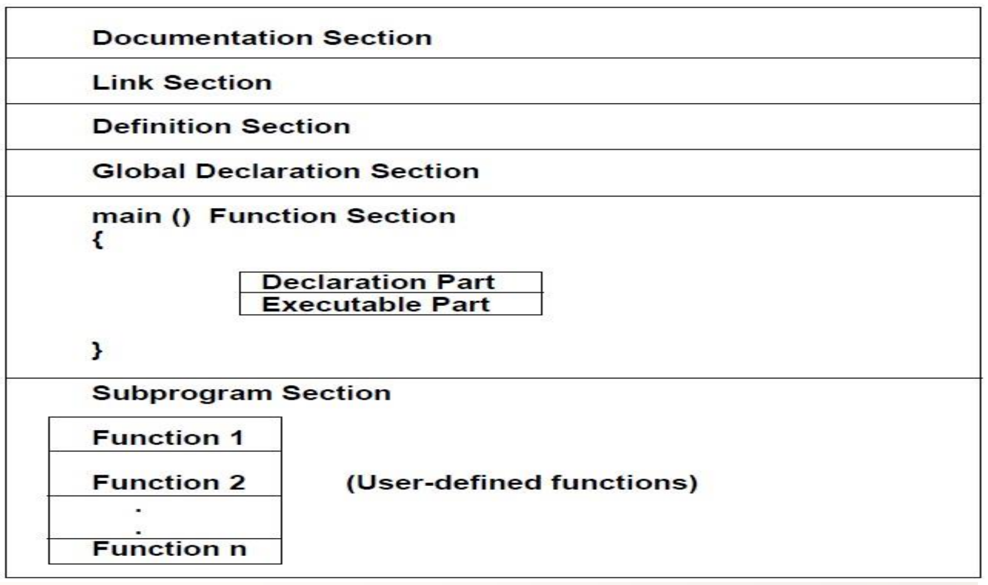
৬। 'সি' প্রোগ্রামিং ভাষাকে একটি স্ট্রাকচার্ড বা প্রোসিডিউর প্রোগ্রামিং ভাষা বলা হয়।

৭। 'সি' প্রোগ্রামিং ভাষায় পর্যাপ্ত সংখ্যাক লাইব্রেরি ফাংশন এবং পর্যাপ্ত সংখ্যাক অপারেটর রয়েছে যা যেকোনো জটিল প্রোগ্রাম লিখতে ব্যবহৃত হয়।

৮। 'সি' প্রোগ্রামিং ভাষায় লেখা প্রোগ্রাম যন্ত্র নির্ভরশীল নয়।

৯। 'সি' প্রোগ্রামিং ভাষার গুরুত্বপূর্ণ বৈশিষ্ট্য হল; এটি নিজেই নিজের বৈশিষ্ট্য বর্ধিত করতে পারে।

'সি' প্রোগ্রামিং ভাষায় লেখা একটি প্রোগ্রামের সাধারণ গঠনঃ



Documentation Section: এটি প্রোগ্রামের ঐচ্ছিক অংশ। এই অংশে প্রোগ্রামের নাম, বিষয়বস্তু, প্রোগ্রামারের নাম, ব্যবহারের নিয়ম ও প্রোগ্রামের উদ্দেশ্য কমেন্টস এর মাধ্যমে লেখা হয়। 'সি' প্রোগ্রামিং ভাষায় কমেন্ট লেখার জন্য দুইটি পদ্ধতি আছে।

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

সিঙ্গেল লাইনের ক্ষেত্রে- //..................

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

Definition Section: এই অংশে কনস্ট্যান্ট ঘোষণা করা হয়। কনস্ট্যান্ট ঘোষণা করার নিয়ম হল-

#define constant_name constant_value

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

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

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

'সি' প্রোগ্রামিং ভাষায় লেখা একটি প্রোগ্রামের বিভিন্ন অংশের বিশ্লেষণঃ

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

```
#include<stdio.h>

#include<conio.h>

main()

{
```

```
    int x, y, sum;

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

    sum = x + y;

    printf("Sum = %d", sum);

    getch();

}
```

প্রোগ্রাম বিশ্লেষণ:

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

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

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

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

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

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

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

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

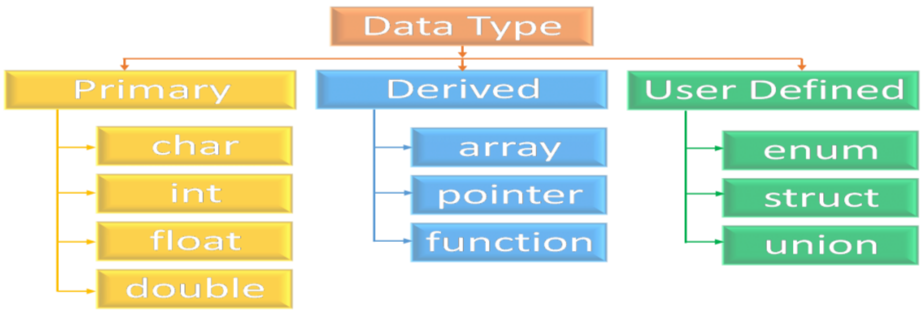
৯। getch( ) লাইব্রেরি ফাংশনটির কাজ হলো আউটপুট ব্যবহারকারী না সরানো পর্যন্ত ধরে রাখা।

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

ডেটা টাইপ, টোকেন, কি-ওয়ার্ড, কনস্ট্যান্ট ও ভেরিয়েবল।

ডেটা টাইপঃ ডেটা টাইপ ডেটার ধরনকে নির্দেশ করে; যেমন- পূর্ণসংখ্যা, ভগ্নাংশ, ক্যারেক্টার ইত্যাদি। প্রতিটি ডেটা টাইপের ভিন্ন ভিন্ন পরিমান মেমোরি প্রয়োজন হয় এবং প্রতিটি ডেটা টাইপের উপর নির্দিস্ট অপারেশন সম্পন্ন হয়।

‘সি’ প্রোগ্রামে নিমোক্ত ডেটা টাইপগুলো ব্যবহৃত হয়ঃ



Primary অথবা Basic অথবা Built-in ডেটা টাইপ:

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

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

বিভিন্ন ডেটা টাইপের মেমোরি পরিসর, ডেটা রেঞ্জ এবং ফরম্যাট স্পেসিফায়ারঃ

DATA TYPE	MEMORY (BYTES)	RANGE	FORMAT SPECIFIER
char	1	-128 to 127	%c
unsigned char	1	0 to 255	%c
int	2	-32,768 to 32,767	%d
unsigned int	2	0 to 65,535	%u
long int	4	-2,147,483,648 to 2,147,483,647	%ld
unsigned long int	4	0 to 4,294,967,295	%lu
float	4		%f
double	8		%lf
long double	10		%Lf

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

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

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

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

auto	double	int
break	else	long
case	enum	register
char	extern	return
const	float	short
continue	for	signed
default	goto	sizeof
do	if	static

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

চলক ঘোষণার ফরম্যাট –

Data_type variable_name;   যেমনঃ int number;

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

১। লোকাল ভেরিয়েবল

২। গ্লোবাল ভেরিয়েবল

১। লোকাল ভেরিয়েবল: কোনো ফাংশনের মধ্যে ভেরিয়েবল ডিক্লেয়ার করলে তাকে উক্ত ফাংশনের লোকাল ভেরিয়েবল বা স্থানীয় চলক বলা হয়। ফাংশনের মধ্যে ঘোষণাকৃত চলক উক্ত ফাংশনের বাইরে ব্যবহার করা যায় না। লোকাল ভেরিয়েবলের কর্মকান্ড শুধুমাত্র সংশ্লিষ্ট ফাংশনেই সীমাবদ্ধ থাকে। ভিন্ন ভিন্ন ফাংশনে একই নামের লোকাল ভেরিয়েবল থাকতে পারে।

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

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

'সি' প্রোগ্রামিং ভাষায় দুইভাবে কনস্ট্যান্ট ঘোষণা করা যায়। যথা-

১। const কীওয়ার্ড ব্যবহার করে

২। #define প্রিপ্রসেসর ব্যবহার করে

const কীওয়ার্ড ব্যবহার করে ধ্রুবক ঘোষণার ফরম্যাট হলো:

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

যেমনঃ **const float PI=3.1416;**

#define প্রিপ্রসেসর ব্যবহার করে ধ্রুবক ঘোষণার ফরম্যাট হলো:

```
#define ConstName ConstValue

#define PI 3.1416
```

স্ট্রিং(Strings): স্ট্রিং হলো কতগুলো ক্যারেক্টারের সমষ্টি যার শেষ উপাদান হলো null ক্যারেক্টার(\0)। এই null ক্যারেক্টার স্ট্রিং এর শেষ নির্দেশ করে। স্ট্রিং সবসময় ডাবল কোটেশনের (" ") সাহায্যে আবদ্ধ থাকে।

স্ট্রিং ডিক্লারেশন করার পদ্ধতিঃ

```
char string[20] = {‘s’,’t’,’u’,’d’,’y’, ‘\0’};

char string[20] = “demo”;

char string [] = “demo”;
```

Special symbols:

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

Brackets[]: অ্যারে এলিমেন্টের রেফারেন্স বুঝাতে ওপেনিং এবং ক্লোজিং ব্র্যাকেট ব্যবহৃত হয়। এটি সিঙ্গেল এবং মাল্টি-ডাইমেনশনাল সাবস্ক্রিপ্ট নির্দেশ করে।

Parentheses(): ফাংশন কল এবং ফাংশন প্যারামিটার নির্দেশ করতে এই বিশেষ চিহ্ন ব্যবহৃত হয়।

Braces{}: ওপেনিং এবং ক্লোজিং কার্লি ব্রেস যথাক্রমে একটি কোড ব্লকের শুরু ও শেষ নির্দেশ করে।

comma (,): একাধিক উপাদানকে পৃথক করতে এই চিহ্ন ব্যবহৃত হয়।

semi colon(;): একাধিক স্টেটমেন্টকে পৃথক করতে এই চিহ্ন ব্যবহৃত হয়।

asterick (*): পয়েন্টার ভেরিয়েবল তৈরি করতে এই বিশেষ চিহ্ন ব্যবহৃত হয়।

assignment operator: ভ্যালু অ্যাসাইন করতে এটি ব্যবহৃত হয়।

pre processor(#): প্রোগ্রাম ফাইল লিঙ্ক করতে কম্পাইলার অটোমেটিক্যালি এই চিহ্ন ব্যবহার করে।

'সি' প্রোগ্রামিং ভাষার অপারেটরসমূহ এবং রাশিমালা

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

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

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

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

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

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

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

কাজের প্রকৃতির উপর ভিত্তি করে 'সি' প্রোগ্রামিং ভাষার অপারেটর সমূহ:

১। গাণিতিক অপারেটর (Arithmetic Operators)

২। রিলেশনাল অপারেটর (Relational Operators)

৩। লজিক্যাল অপারেটর (Logical Operators)

৪। অ্যাসাইনমেন্ট অপারেটর (Assignment Operators)

৫। ইনক্রিমেন্ট এবং ডিক্রিমেন্ট অপারেটর (Increment and Decrement Operators)

৬। কন্ডিশনাল অপারেটর (Conditional Operators)

৭। বিট ওয়াইজ অপারেটর (Bitwise Operators)

৮। বিশেষ অপারেটর (Special Operator)

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

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

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

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

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

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

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

Simple assignment operator	Short hand assignment operator
a = a +	a + = 1
a = a –1	a – = 1
a = a * b	a * = b
a = a/b	a / = b
a = a%b	a% = b

ইনক্রিমেন্ট এবং ডিক্রিমেন্ট অপারেটর (Increment and Decrement Operators):

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

ইনক্রিমেন্ট অপারেটরের প্রকারভেদঃ

- pre-increment
- post-increment

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

```
#include<stdio.h>

#include<conio.h>

main()

{

  int x,i;

  i=10;

  x=++i;

  printf("x: %d",x);

  printf("i: %d",i);

  getch();

}
```

Output:

```
x: 11

i: 11
```

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

```
#include<stdio.h>

#include<conio.h>

void main()

{

  int x,i;

  i=10;

  x=i++;

  printf("x: %d",x);

  printf("i: %d",i);

  getch();

}
```

Output:

```
x: 10

i: 11
```

ডিক্রিমেন্ট অপারেটরের প্রকারভেদঃ

- pre-decrement
- post-decrement

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

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

```
void main()

{

  int x,i;

  i=10;

  x=--i;

  printf("x: %d",x);

  printf("i: %d",i);

}
```

Output:

```
x: 9

i: 9
```

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

```
#include<stdio.h>

#include<conio.h>

void main()

{

  int x,i;

  i=10;

  x=i--;

  printf("x: %d",x);

  printf("i: %d",i);

  getch();

}
```

Output:

```
x: 10

i: 9
```

কন্ডিশনাল অপারেটর (Conditional Operators):

'সি' প্রোগ্রামে শর্ত সাপেক্ষে কোন কাজ করার জন্য কন্ডিশনাল অপারেটর ব্যবহৃত হয়। কন্ডিশনাল অপারেটরের গঠন নিম্নরুপঃ

Syntax : (Condition? true_value: false_value); Example : (A < 0 ? Negative : Positive);

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

অপারেটর	**বর্ননা**
&	চলকের অ্যাড্রেস পেতে এই অপারেটর ব্যবহৃত হয়। উদাহরণ : &a, a এর অ্যাড্রেস দিবে।

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

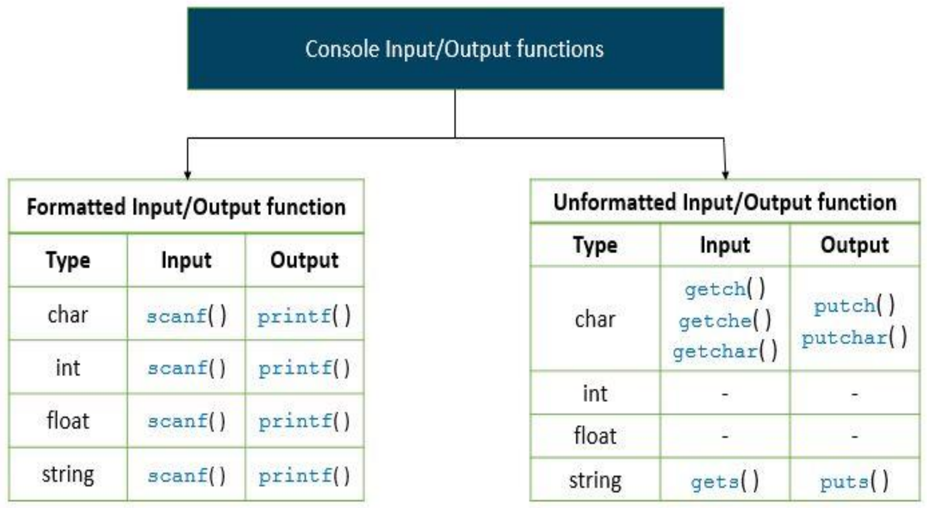
রাশিমালা (Expression): চলক, ধ্রুবক ও বিভিন্ন অপারেটরের সমন্বয়ে রাশিমালা বা Expression তৈরি হয়।

গাণিতিক Expression গুলো 'সি' প্রোগ্রামে নিম্নরুপে লেখা হয়-

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

'সি' প্রোগ্রামিং ভাষায় ইনপুট এবং আউটপুট ফাংশন সমূহ

ফাংশনসমূহঃ কোন প্রোগ্রামে ডেটা প্রক্রিয়া করার জন্য প্রথমে ডেটা ইনপুট নিতে হয়। প্রোগ্রামে ডেটা ইনপুট নেওয়ার জন্য ব্যবহৃত ফাংশনকে ইনপুট ফাংশন বলে।



Formatted Input/Output function

Type	Input	Output
char	scanf()	printf()
int	scanf()	printf()
float	scanf()	printf()
string	scanf()	printf()

Unformatted Input/Output function

Type	Input	Output
char	getch() getche() getchar()	putch() putchar()
int	-	-
float	-	-
string	gets()	puts()

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

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

ফরমেট	**ব্যবহার**	**উদাহরণ**
%c	char টাইপের ডেটা	scanf("%c",&a); printf("%c",a);
%d	int টাইপের	scanf("%d",&a); printf("%d",a);
%f	float টাইপের	scanf("%f",&a); printf("%f",a);
%lf	double টাইপের	scanf("%lf",&a); printf("%lf",a);
%ld	long int টাইপের	scanf("%ld",&a); printf("%ld",a);
%u	unsigned int	scanf("%u",&a); printf("%u",a);
%o	Octal ডেটা	scanf("%o",&a); printf("%o",a);
%x	Hexadecimal ডেটা	scanf("%x",&a); printf("%x",a);

scanf() ফাংশনের ব্যবহারঃ

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

scanf("format_specifier ", &variable_name);

উদাহরণঃ

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

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

scanf(” format_specifier1 format_specifier2….”, &variable_name1, &variable_name2…….);

উদাহরণঃ

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

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

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

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

printf() ফাংশনের ব্যবহারঃ

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

কোন কিছু হুবহু আউটপুটে দেখানোর জন্য printf() ফাংশনের ফরমেটঃ

আউটপুটে দেখানোর প্রয়োজনীয় টেক্সটি printf(” “); ফাংশনের ডাবল কোটেশনের মধ্যে লিখতে হয়। যেমন-

printf(” Output text should be here “);

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

printf(“format_specifier”, variable_name);

উদাহরণঃ

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

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

printf(“format_specifier1, format_specifier2….”, variable_name1, variable_name2…);

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

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

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

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

সাধারণ গাণিতিক সমস্যা সম্পর্কিত প্রোগ্রামসমূহ

১। দুইটি সংখ্যা ইনপুট নিয়ে যোগফল নির্নয় করার জন্য ‘সি’ প্রোগ্রামিং ভাষায় একটি প্রোগ্রাম।

```
#include<stdio.h>

main()

{

  int a, b, c;

  printf(“ Enter Two numbers: “);

  scanf(“%d %d”,&a,&b);

  c = a+b;

  printf("Summation = %d",c);

}
```

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

```
#include<stdio.h>

main()

{

    int a, b, c;

    printf(“ Enter Two numbers: “);

    scanf(“%d %d”,&a,&b);

    c = a-b;

    printf("Subtraction= %d",c);

}
```

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

```
#include<stdio.h>

main()

{

    int a, b, c;

    printf(“ Enter Two numbers: “);

    scanf(“%d %d”,&a,&b);

   c= a*b;

    printf("Multiplication= %d", c);            }
```

৫। সেলসিয়াস স্কেলের তাপমাত্রাকে ফারেনহাইট স্কেলের রুপান্তরের প্রোগ্রাম।

তাপমাত্রা পরিমাপের বিভিন্ন স্কেলের মধ্যে সম্পর্ক-

C/5 = F-32 / 9 = K-273 / 5

```
#include<stdio.h>

main()

{

    int c, f;

    printf("Enter celcious temperature :");

    scanf("%d",&c);

    f=9*c/5+32;

    printf("Ferhenheight temperature:%d”,f);

}
```

৬। ফারেনহাইট স্কেলের তাপমাত্রাকে সেলসিয়াস স্কেলের রুপান্তরের প্রোগ্রাম।

তাপমাত্রা পরিমাপের বিভিন্ন স্কেলের মধ্যে সম্পর্ক-

C/5 = F-32 / 9 = K-273 / 5

```
#include<stdio.h>

main()

{

    int c, f;

    printf("Enter Ferenheight temperature :");

    scanf("%d",&f);

    c=5*(f-32)/9;

    printf("Celcious temperature:%d”,c);

}
```

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

```
#include<stdio.h>

main()

{

    int b,h;

    float area;

    printf("Enter Base & Height:");

    scanf("%d %d",&b,&h);

    area=.5*b*h;

    printf("\nThe area is %.2f",area);

}
```

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

```
#include<stdio.h>

#include<math.h>

main()

{

    int a, b, c;

    float s, area;

    printf("Enter three 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();

}
```

৯। আয়তক্ষেত্রের ক্ষেত্রফল নির্নয় করার প্রোগ্রাম।

```
#include<stdio.h>

main()

{

    int l,w,area;

    printf("Enter the length & width: ");

    scanf("%d %d",&l,&w);

    area=l*w;

    printf("\nThe area is %d",area);

    getch();

}
```

১০। বৃত্তের ক্ষেত্রফল নির্নয় করার প্রোগ্রাম।

```
#include<stdio.h>

main ( )

{

    int r;

    float area;

    printf ("Enter integer value for radius:");

    scanf ("%d", &r) ;

    area = 3.1416*r*r;

    printf("\n Area of circle =%f", area);

    getch();

}
```

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

কন্ট্রোল স্টেটমেন্ট সমূহঃ

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

- ৩। জাম্পিং কন্ট্রোল স্টেটমেন্ট

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

'সি' প্রোগ্রামিং ভাষায় কন্ডিশনাল কন্ট্রোল স্টেটমেন্টসমূহঃ

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

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

```
if(condition)
{
  statement(s);
}
```

কীবোর্ড থেকে কোনো সংখ্যা ইনপুট দিয়ে দেখবে সংখ্যাটি ধনাত্মক কিনা?

```
#include<stdio.h>
main()
{
    int a;
    printf("Enter a number");
    scanf("%d",&a);
    if(a>0)
        printf("The given number is Positive");
}
```

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

```
if(condition)
{
  statement(s);
}
else
{
  statement(s);
}
```

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

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

else if স্টেটমেন্ট: প্রোগ্রামে যদি একাধিক কন্ডিশন যাচাই করতে হয় তাহলে প্রথম কন্ডিশন যাচাই করার জন্য if স্টেটমেন্ট ব্যবহার করা হয়। তারপরের কন্ডিশন গুলো যাচাই করার জন্য else if স্টেটমেন্ট ব্যবহার করা হয়। 'সি' প্রোগ্রামে "অন্যথায় যদি" অর্থে else if স্টেটমেন্ট ব্যবহার করা হয়। else if স্টেটমেন্ট ব্যবহারের ফরম্যাট হলো-

```
if(Condition1)
{
  statement(s)
}
else if(Condition2)
{
  statement(s)
}
  ...............
else
{
  statement(s)
}
```

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

```
#include<stdio.h>
main()
{
    int a;
    printf("Enter a number");
    scanf("%d",&a);
    if(a==0)
        printf("The given number is Zero");
    else if(a>0)
        printf("The given number is Positive");
    else
```

```
        printf("The given number is Negative");
}
```

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

```
if(Condition1)
{
  if(Condition2)
  {
     statement(s)
  }
}
```

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

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

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

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

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

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

অধিবর্ষ বা লিপ ইয়ার হচ্ছে প্রতি চার বছর পরপর ৩৬৬ দিনে বছর হিসাব করা হয়। গ্রেগরীয় বর্ষপঞ্জিমতে, প্রতি চার বছরে একবার ফেব্রুয়ারি মাসে ও বাংলা সনমতে ফাল্গুন মাসে এই অতিরিক্ত ১ দিন যোগ হয়। । যেমন: ২০১২ একটি অধিবর্ষ ও এর ফেব্রুয়ারি মাস হয়েছে ২৯ দিনে।

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

লিপ ইয়ার(অধিবর্ষ) নির্নয়ের প্রোগ্রাম।

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

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

```
#include<stdio.h>
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", a);

    else

        printf("Largest Number is: %d", b);

}
```

5। তিনটি সংখ্যার মধ্যে সবচেয়ে ছোট সংখ্যা নির্নয়ের জন্য সি একটি প্রোগ্রাম।

```
#include<stdio.h>

main()

{

    int a,b,c;

    printf("Enter three integer numbers:");

    scanf("%d  %d  %d", &a, &b, &c);

    if(a<b&&a<c)

    {

            printf("\n Smallest number is: %d", a);

}

   else if(b<a&&b<c)

            printf("\n Smallest number is: %d", b);

    }

    else

    {

           printf("\n Smallest number is: %d", c);

    }

}
```

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

```
#include<stdio.h>

main()

{

    int a,b,c;

    printf("Enter three integer numbers:");

    scanf("%d  %d  %d", &a, &b, &c);

    if(a>b&&a>c)

    {

            printf("\n largest number is: %d", a);

}

   else if(b>a&&b>c)

            printf("\n largest number is: %d", b);

    }

    else

    {

            printf("\n largest number is: %d", c);

    }

}
```

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

লুপের প্রকারভেদ:

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

•অসীম লুপ – যদি কোন লুপ অনবরত পুনরাবৃত্তি হতে থাকে, অর্থাৎ কখনো শেষ না হয় তবে তাকে অসীম লুপ বলে।

•মধ্যবর্তী লুপ – একটি লুপের মধ্যে যদি অপর একটি লুপ থাকে তাহলে তাকে মধ্যবর্তী বা নেস্টেড লুপ বলে।

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

লুপ কন্ট্রোল স্টেটমেন্ট সমূহঃ

১। for লুপ স্টেটমেন্ট

২। while লুপ স্টেটমেন্ট

৩। do...while লুপ স্টেটমেন্ট

প্রত্যেক লুপ কন্ট্রোল স্টেটমেন্টের দুটি অংশ থাকে। যথা-

১। লুপ ডিক্লারেশন

২। লুপ বডি

লুপ ডিক্লারেশন তিনটি প্রধান অংশ-

১। Initialization Statement- এই অংশে ভেরিয়েবলের প্রাথমিক মান নির্ধারন করা হয়।

২। Test Expression – এই অংশে শর্ত লেখা হয়। শর্ত মিথ্যা না হওয়া পর্যন্ত লুপ বডি পুনরাবৃত্তি হয়।

৩। Update Statement- প্রতিবার লুপ বডি পুনরাবৃত্তির পর ভেরিয়েবলের মান হ্রাস/বৃদ্ধি নির্ধারণ করা হয় এই অংশে।

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

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

এন্ট্রি কন্ট্রোল লুপ স্টেটমেন্ট

এক্সিট কন্ট্রোল লুপ স্টেটমেন্ট

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

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

for লুপ স্টেটমেন্টঃ 'সি' প্রোগ্রামে শর্ত সাপেক্ষে এক বা একাধিক স্টেটমেন্ট নির্দিষ্ট সংখ্যকবার নির্বাহ করতে for লুপ স্টেটমেন্ট ব্যবহার করা হয়। লুপ বডির কোড নির্বাহের

পূর্বে কন্ডিশন চেক করে। লুপ কতবার নির্বাহ হবে তা জানা থাকলেই কেবলমাত্র for লুপ ব্যবহার করা যায়। নিম্নে for লুপ স্টেটমেন্টের ফরম্যাট দেওয়া হলো-

```
for (initialization; condition; imcrement)

{

        Statement;

}
```

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

```
#include<stdio.h>

main()

{

    int i;

    for(i=1; i<=5; i++)

    {

        printf("Hello World\n");

    }

}
```

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

```
initialization;

while (condition)

{

        Statement;

        Increment;

}
```

এবারে আমরা While loop ব্যবহার করে 1+2+3=.........+100 প্রোগ্রাম তৈরি করব।

```
#include<stdio.h>

main()

{

    int i;

    i=1;

    while(i<=100)

    {

        s-s+I;

        i++;

    }

printf("The sum %d",s);

}
```

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

```
initialization;

do

{

            Statement;

        Increment;

}

while (condition) ;
```

1.১ থেকে ১০০ পর্যন্ত সংখ্যা গুলোর যোগফল দেখানোর প্রোগ্রাম। অথবা

১+২+৩+৪+ – – – – –+১০০ ধারার যোগফল দেখানোর প্রোগ্রাম।

/* for loop ব্যবহার করে প্রোগ্রাম */

```
#include<stdio.h>

main()

{

    int i,s=0;

    for(i=1;i<=100; i=i+1)

    {

        s=s+i;

    }

    printf("Sum=%d ",s);

}
```

/* while loop ব্যবহার করে প্রোগ্রাম */

```
#include<stdio.h>

main()

{

    int i,s=0;

    i=1;

    while(i<=100)

    {

        s=s+i;

        i=i+1;

    }

    printf("Sum=%d ",s);

}
```

/* do while loop ব্যবহার করে প্রোগ্রাম */

```
#include<stdio.h>

main()
```

```
{
    int i,s=0;
    i=1;
    do
    {
        s=s+i;
        i=i+1;
    } while(i<=100);
    printf("Sum=%d ",s);
}
```

৮। ১ থেকে n পর্যন্ত সংখ্যা গুলোর যোগফল দেখানোর প্রোগ্রাম। অথবা ১+২+৩+৪+ – – – – –+n ধারার যোগফল দেখানোর প্রোগ্রাম।

```
/* for loop ব্যবহার করে প্রোগ্রাম */
#include<stdio.h>
main()
{
    int i,n,s=0;
    printf("Enter Value of n: ");
    scanf("%d",&n);
    for(i=1;i<=n; i=i+1)
    {
        s=s+i;
    }
    printf("Sum=%d ",s);
}
```

```
/* while loop ব্যবহার করে প্রোগ্রাম */
#include<stdio.h>
main()
{
    int i,n,s=0;
    printf("Enter Value of n: ");
    scanf("%d",&n);
    i=1;
    while(i<=n)
    {
        s=s+i;
        i=i+1;
    }
    printf("Sum=%d ",s);
}
```

```
/* do while loop ব্যবহার করে প্রোগ্রাম */
#include<stdio.h>
main()
{
    int i,n,s=0;
    printf("Enter Value of n: ");
    scanf("%d",&n);
    i=1;
    do
    {
        s=s+i;
        i=i+1;
    } while(i<=n);
    printf("Sum=%d ",s);
}
```

2। ১ থেকে ১০০ এর মধ্যে অবস্থিত বিজোড় সংখ্যা গুলোর যোগফল দেখানোর প্রোগ্রাম। অথবা ১+৩+৫+ – – – – –+১০০ ধারার যোগফল দেখানোর প্রোগ্রাম।

```
/* for loop ব্যবহার করে প্রোগ্রাম */
#include<stdio.h>
main()
{
int i,s=0;
for(i=1; i<=100; i=i+2)
{
    s=s+i;
}
printf("Sum=%d ",s);
}
```

```
/* while loop ব্যবহার করে প্রোগ্রাম */
#include<stdio.h>
main()
{
int i,s=0;
i=1;
while(i<=100)
{
    s=s+i;
    i=i+2;
}
printf("Sum=%d ",s);
```

```
}

/* do while loop ব্যবহার করে প্রোগ্রাম */

#include<stdio.h>

main()

{

int i,s=0;

i=1;

do

{

    s=s+i;

    i=i+2;

} while(i<=100);

printf("Sum=%d ",s);          }
```

3. ১ থেকে ১০ এর মধ্যে অবস্থিত জোড় সংখ্যাগুলো দেখানোর প্রোগ্রাম। অথবা ২ ৪ ৬ ৮ ১০ ক্রম প্রদর্শনের প্রোগ্রাম।

```
/* for loop ব্যবহার করে প্রোগ্রাম */

#include<stdio.h>

main()

{

    int i;

    for(i=2;i<=10; i=i+2)

    {

        printf("%d\t ",i);

    }

    getch();

}
```

```
/* while loop ব্যবহার করে প্রোগ্রাম */

#include<stdio.h>

main()

{

    int i;

    i=2;

    while(i<=10)

    {

        printf("%d\t ",i);

        i=i+2;

    }

    getch();

}
```

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

ফাংশনের নাম– ফাংশনের নাম একটি আইডেন্টিফায়ার যা যেকোন নাম হতে পারে। তবে ফাংশনের নাম অর্থপূর্ণ হওয়া উচিৎ, যাতে নাম দেখেই ফাংশনের উদ্দেশ্য বুঝা যায়।

ফাংশনের নাম লেখার ক্ষেত্রে আইডেন্টিফায়ার এর নিয়ম অনুসরণ করে লেখতে হয়।

ফাংশন প্যারামিটার– প্যারামিটার অংশে ডেটা টাইপ সহ চলক থাকে, যা ফাংশনটি কী ধরণের কয়টি ডেটা ইনপুট নিবে তা নির্ধারণ করে।

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

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

১। লাইব্রেরি ফাংশন   ২। ইউজার-ডিফাইন্ড ফাংশন

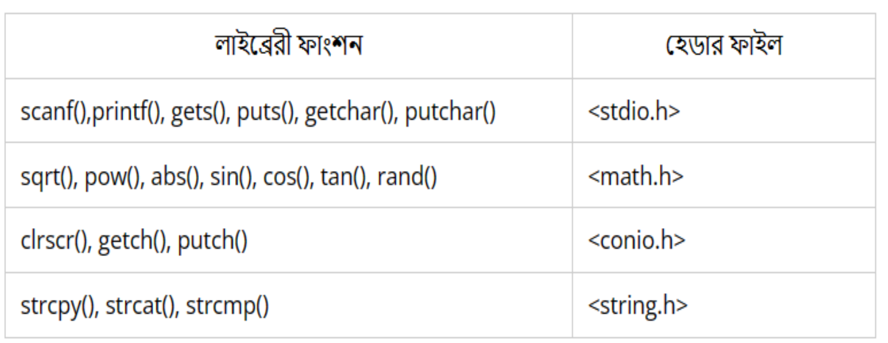
লাইব্রেরি ফাংশনঃ লাইব্রেরি ফাংশন হলো এমন একটা ফাংশন যার ডেফিনেশন প্রোগ্রামারকে লিখতে হয় না। যেগুলো বিশেষ কিছু কার্য সম্পাদনের জন্য 'সি' কম্পাইলারে বিল্ট-ইন থাকে।

শুধু প্রয়োজনে সেগুলোকে তাদের নিজস্ব ফরম্যাট অনুযায়ী main () ফাংশনে ব্যবহার বা কল করা হয়। যেমন- scanf(), printf(), gets(), puts(), getchar(), putchar(), abs(), pow(b,p), sqrt(), sin(), cos(), tan(), rand() ইথ্যাদি হচ্ছে লাইব্রেরি ফাংশন। এক্ষেত্রে প্রতিটি লাইব্রেরী ফাংশনের নিজস্ব হেডার ফাইল প্রোগ্রামের শুরুতে লিখে দিতে হয়। লাইব্রেরি ফাংশন দুই প্রকার।

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

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

বিভিন্ন লাইব্রেরী ফাংশন ও তাদের হেডার ফাইলঃ



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

ইউজার-ডিফাইন্ড ফাংশন: ইউজার-ডিফাইন্ড ফাংশন হলো এমন একটা ফাংশন যার ডেফিনেশন প্রোগ্রামারকে লিখতে হয়। যেমন – main() ফাংশন একটি ইউজার-ডিফাইন্ড ফাংশন। কারণ এর ডেফিনেশন প্রোগ্রামার লিখে থাকে।

ফাংশন চেনার সহজ উপায়ঃ ফাংশন চেনার সহজ উপায় হলো ফাংশনের নামের শেষে এক জোড়া প্রথম বন্ধনী '( )' থাকে, এই প্রথম বন্ধনীর মধ্যে অনেক কিছু থাকতে পারে, আবার নাও থাকতে পারে।

প্রতিটি ফাংশনের একটি নাম থাকে, যে নামে কম্পাইলার তাকে সনাক্ত করে।

Engr. Kaushik Saha
B.Sc in Electrical & Electronics Engineering (EEE)
Teaching asst. At Islamic University (IIUC)
Campus 1: SCIENCE CARE , Block:A, Halishahar.
Campus 2: STUDY ZONE , Gate NO: 07,Block:K, Halishahar.
Mobile: 01832221610