লজিক গেইটের অ আ ক খ

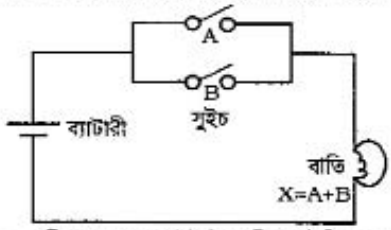
কমপিউটার জগৎ মার্চ '৯৫ সংখ্যায় বাইনারী গণনা নিয়ে আলাপকালে বলেছিলাম বিদ্যুৎ নির্ভর কমপিউটারে সমস্ত গাণিতিক হিসাব নিকাশ '০' এবং '১' দিয়ে প্রকাশিত। যোগ বিয়োগ গুণ ভাগসহ যাবতীয় গণনার কালে সার্কিট বা বর্তনীর কোনো এক বিন্দুতে দাঁড়ালে লক্ষ্য করা যাবে কেবল দুটো দশা-বিদ্যুতের অনুপস্থিতি (০) কিংবা উপস্থিতি (১)। কমপিউটারের ভেতরের মূল মস্তিষ্ক সিপিইউ বা সেন্ট্রাল প্রসেসিং ইউনিট। গোটা গাণিতিক, যৌক্তিক ও নিয়ন্ত্রণ সংক্রান্ত কর্মকান্ড সম্পাদনের দায়িত্ব এর উপরই ন্যস্ত। আবার রম (ROM), র‍্যাম (RAM) ইত্যাদি নামের স্মৃতি কক্ষ বা রেজিস্টারগুলোয় ডাটা বা তথ্য জমা থাকছে '০' এবং '১' এর সমাবেশে। আর কমপিউটারের এসব সিপিইউ, স্মৃতি ইত্যাদি জটিল অথচ গুরুত্বপূর্ণ অংশগুলো কিন্তু তৈরি করা হয় লক্ষ লক্ষ সরল ক্ষুদ্রাতিক্ষুদ্র বর্তনীর সমন্বয়ে। এসব ক্ষুদ্র সরল বৈদ্যুতিক বর্তনীর ইনপুট/আউটপুট ভোল্টেজ বা বিদ্যুৎ স্তরের মানগুলোকে কেবল ০ অথবা ১ ধরা যায়। এবং গুলোর কর্মকান্ডকে কতক সহজ যুক্তি দিয়েই বোঝানো সম্ভব। কমপিউটারের জটিল বৈদ্যুতিক বর্তনীর সংগঠনের প্রাথমিক উপাদান এই সহজ ক্ষুদ্রতম বর্তনীগুলোকেই বলা হয় মৌলিক যৌক্তিক বা লজিক গেইট। সাধারণতঃ একটি মৌলিক লজিক গেইটের ইনপুট প্রান্ত এক বা একাধিক হলেও আউটপুট প্রান্ত থাকে একটিই, যৌক্তিক কাজ কর্মের ধরন ধারণ বিচারে এরকম যৌক্তিক গেইটের সংখ্যা হাতে গোণা মাত্র কয়েকটি। দুই বা ততোধিক মৌলিক লজিক গেইট জুড়ে ক্রমান্বয়ে জটিল থেকে জটিলতর যৌগিক লজিক বর্তনী ইত্যাদি তৈরি করা হয়। আমরা সংক্ষেপে মৌলিক লজিক গেইটসমূহ এবং এসব খাটিয়ে আরো দু একটা যৌগিক লজিক গেইট নিয়ে আলোচনা করবো।

(ক) অর গেইট (OR Gate) ঃ দুই বা ততোধিক ইনপুট A, B, ...এবং একটি আউটপুট × বিশিষ্ট এই গেইট যৌক্তিক অর (OR) অপারেশন সম্পন্ন করে। অর যৌক্তিক অপারেশনটি এমন, ইনপুট A এবং অথবা B যদি অন '১' হয় তবে আউটপুট X অন '১' হবে। অর্থাৎ আউটপুটে (x=১) ভোল্টেজ পাওয়া যাবে কেবল তখনই যখন ইনপুটের কমপক্ষে একটি প্রান্তে ভোল্টেজ সরবরাহ চালু থাকবে। আবার একযোগে সবগুলো ইনপুট '০' অফ থাকলে একমাত্র সে ক্ষেত্রেই আউটপুট X=0 অফ পাওয়া যায়। বীজগণিতের ধরণে অর অপারেশনকে লেখা যায় এভাবে, A + B = X। পড়তে হবে A অর B সমান X। এখানে এই '+' যোগ চিহ্নটি যোগ নয় যৌক্তিক অর (OR) অপারেশন বোঝায়। প্রতীক দিয়ে বর্তনীতে অর গেইটের প্রকাশ এরকম-

দুটি ইনপুট A এবং B গ্রহণ করতে পারে এ রকম সম্ভাব্য চারটি বিন্যাস বা দশা (০০,০১,১০ এবং ১১) বিবেচনা করে আউটপুট X কেমন হবে তা নিচের একটি সত্য-মিথ্যা ছক বা Truth Table এ দেখানো হলো (চিত্র-১)। লক্ষ্য করুন, A অথবা B এর যে কোনো একটি কিংবা উভয়ই যদি অন '১' হয় তবে আউটপুট X=১ পাওয়া যায়। আর আউটপুট X=0 কেবল একটি ক্ষেত্রে যেখানে A, B দুটোই অফ (০) অবস্থায়।

A	B	X = A+B
০	০	০
০	১	১
১	০	১
১	১	১

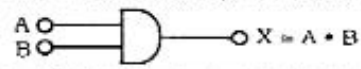
চিত্র-১ ঃ অর গেইটের সত্য-মিথ্যা ছক।



চিত্র-২ ঃ অর গেইট-বৈদ্যুতিক সুইচিং বর্তনী দিয়ে।

অর গেইটের কর্মকান্ড স্পষ্ট করতে উপরের চিত্র-২ এ একটি ব্যাটারী, দুটি সুইচ (A,B) আর একটি বাতি X সম্বলিত বৈদ্যুতিক বর্তনীর সহায়তা নেয়া হলো। সুইচ A এবং অথবা B বন্ধ (১) করলেই বাতি জ্বলছে X=১। দুটো সুইচ একযোগে খোলা থাকলে (০০) বাতি জ্বলবে না '০'। এটিই তাহলে যৌক্তিক অর গেইট হিসেবে কাজ করছে। জেনে রাখা ভালো, এই অর গেইট থেকেই অপরাপর প্রায় সব লজিক গেইটই রূপায়িত করা যায় বলে এটিকে ইউনিভার্সাল গেইটও বলা হয়ে থাকে।

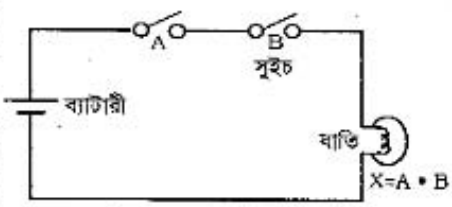
(খ) এন্ড গেইট (AND Gate) ঃ এন্ড গেইটের দুটি ইনপুটের কমপক্ষে একটি ইনপুট অফ '০' হলেই আউটপুট '০' হয়ে যাবে। একযোগে সবগুলো ইনপুট অন '১' হলেই কেবল আউটপুট অন '১' পাওয়া যায়। বীজ গাণিতিক প্রকাশে লেখা হয় এভাবে, A • B=X। উচ্চারণ, A এন্ড B সমান X। এখানে (•) ডট দিয়ে (গুণ নয়।) এন্ড (AND) অপারেশন বোঝানো হয়েছে। এন্ড লজিক গেইটের প্রতীক



সত্য-মিথ্যা ছকে এন্ড অপারেশনের ফলাফল দেখানো হলো (চিত্র-৩)। আগের মতোই, সমতুল্য একটি ব্যাটারী, দুটি সুইচ ও বাতি সম্বলিত বর্তনী দিয়ে এন্ড যৌক্তিক গেইটকে ব্যাখ্যা করা হয়েছে (চিত্র-৪)।

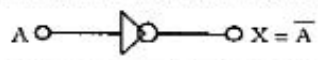
A	B	X = A • B
০	০	০
০	১	০
১	০	০
১	১	১

চিত্র-৩ ঃ এন্ড অপারেশনের সত্য-মিথ্যা ছক।



চিত্র-৪ ঃ এন্ড গেইট।

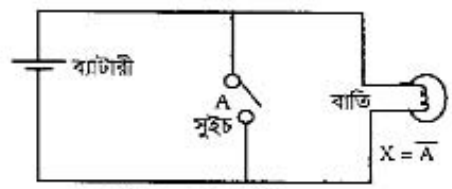
(গ) নট গেইট (Not Gate) ঃ একটি ইনপুট A এবং একটি আউটপুট X বিশিষ্ট এই যৌক্তিক নট গেইট মূলতঃ ইনপুটে আগত ডাটাকে উল্টে দেয়। ইনপুট অন '১' থাকলে আউটপুট হবে বিপরীত অফ '০'। ইনপুট '০' থাকলে আউটপুট হবে ১। বীজগণিতে লিখলে $X=\overline{A}$। উচ্চারণ X সমান নট A। আউটপুটে A এর উপরে রেখা দিয়ে বিপরীত মান বোঝায়। নট গেইটকে একারণে যৌক্তিক ইনভার্টারও বলে। আর অপারেশনটির নাম যৌক্তিক ইনভার্শন। নট গেইটের প্রতীক



সত্য-মিথ্যা ছকে (চিত্র-৫) এবং সমতুল্য বৈদ্যুতিক বর্তনীতে নিচে নট গেইটের কার্যপ্রণালী প্রকাশ করা হলো (চিত্র-৬)

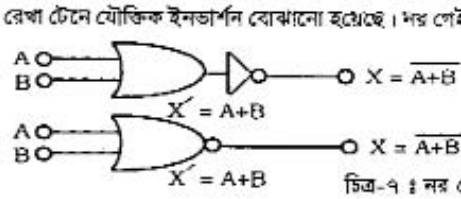
A	X = $\overline{A}$
০	১
১	০

চিত্র-৫ ঃ নট গেইটের সত্য-মিথ্যা ছক।



চিত্র-৫ ঃ নট গেইট।

(ঘ) নর গেইট (NOR Gate) ঃ একটি অর গেইটের পরপরই একটি নট গেইট জুড়ে এই যৌগিক নর গেইট বানানো হয়েছে। দুটো ইনপুটের মাঝে অর (OR) অপারেশনে প্রাপ্ত ফলাফলকে নট গেইট নট অপারেশনে উল্টে দিলে সবশেষে যে ফলাফল বেরিয়ে আসে তা-ই সামগ্রিক ভাবে NOR নর অপারেশনের ফলাফল। অর্থাৎ প্রথমে 'অর' পরে 'নট' ক্রিয়াশীল হলে দু'য়ে মিলে নর NOR অপারেশন সম্পন্ন হয়। বীজগণিতের আদলে দেখলে, $\overline{A+B}=X$। উচ্চারণ A নর B সমান X। A অর B এর উপরে রেখা টেনে যৌক্তিক ইনভার্শন বোঝানো হয়েছে। নর গেইটের প্রতীক

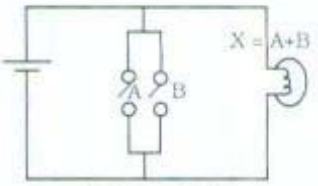


চিত্র-৭ ঃ নর গেইট

সত্যমিথ্যা ছকে (চিত্র-৮) নর অপারেশনের ফলাফল এবং বৈদ্যুতিক সমতুল্য নর গেইট বর্তনী (চিত্র-৯) দেখানো হলো।

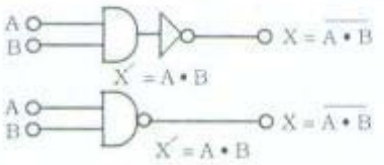
A	B	$X' = A+B$	$X = \overline{A+B}$
০	০	০	১
০	১	১	০
১	০	১	০
১	১	১	০

চিত্র-৮ ঃ নর অপারেশনের সত্য মিথ্যা ছক



চিত্র-৯ ঃ নরগেইট

(ঙ) ন্যান্ড গেইট (NAND Gate) ঃ এক্ষেত্রে এন্ড গেইটের সাথে একটি নট গেইট জুড়ে ন্যান্ড গেইট তৈরি করা যায়। চিত্র-১০ এ ন্যান্ড গেইটের প্রতীক দেখানো হলো।

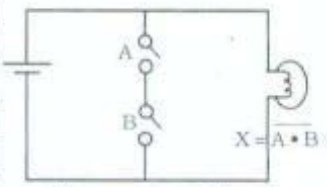


চিত্র-১০ ঃ ন্যান্ড গেইটের প্রতীক।

A এবং B ইনপুট দুটোর মাঝে (•) এন্ড অপারেশন থেকে প্রাপ্ত ফলাফল নট গেইট উল্টে দিয়ে শেষ অবধি ন্যান্ড অপারেশনের ফলাফল উগড়ে দেয়। বীজ গণিতে $\overline{A \cdot B} = X$। উচ্চারণ, A ন্যান্ড B সমান X। এক্ষেত্রে সত্যমিথ্যা ছকে (চিত্র-১১) এবং বৈদ্যুতিক সমতুল্য বর্তনীতে ন্যান্ড গেইট (চিত্র-১২) স্পষ্ট ব্যাখ্যা করা হয়েছে।

A	B	$X' = A \cdot B$	$X = \overline{A \cdot B}$
০	০	০	১
০	১	০	১
১	০	০	১
১	১	১	০

চিত্র -১১ ঃ ন্যান্ড অপারেশনের সত্যমিথ্যা ছক।



চিত্র-১২ ঃ ন্যান্ড গেইট।

(চ) এক্সক্লোসিভ অর গেইট (ExOR Gate) ঃ এক্সক্লোসিভ অর গেইটের দুটো ইনপুটে A এবং B এর একটি অপরটি থেকে ভিন্ন দশায় থাকলেই আউটপুট অন '১' পাওয়া যায়। দুটো ইনপুট একই দশায় থাকলে আউট পুট হয় অফ '০'। অর্থাৎ আউটপুট '১' পেতে চাইলে একটি ইনপুট অন '১' থাকলে অবশ্যই অপরটিকে অফ '০' দশায় থাকতে হবে। আবার দুটো ইনপুটই '০' কিংবা দুটোই '১' হলে আউটপুট হবে অফ '০'। বীজগাণিতিক প্রকাশে, $A \oplus B = X$। উচ্চারণ, A এক্স-অর B সমান X। প্রতীক (চিত্র-১৩) এবং সত্য-মিথ্যা ছক (চিত্র-১৪) দিয়ে এক্স-অর গেইট ও অপারেশন যথাক্রমে ব্যাখ্যা করা হয়েছে। বলা দরকার, এটি একটি যৌগিক গেট।

চিত্র-১৩ ঃ এক্সক্লোসিভ-অর গেইটের প্রতীক।

A	B	$X = A \oplus B$
০	০	০
০	১	১
১	০	১
১	১	০

চিত্র-১৪ ঃ এক্স-অর গেইটের সত্য-মিথ্যা ছক।

অর, এন্ড এবং নট এই তিনটি মৌলিক গেইট খাটিয়ে নানা কায়দায় এই এক্স-অর গেইট তৈরি করা সম্ভব। আগ্রহী পাঠক একটু চেষ্টা করেদেখুন, পেয়ে যাবেন এক্স-অর গেইট।

কমপিউটারের মূল যান্ত্রিক সংগঠন তথা হার্ডওয়্যার অধ্যয়নের প্রথম পর্বে অবশ্যই এসব মৌলিক কিংবা সহজ যৌগিক লজিক গেইটগুলোর ধারণা গ্রহণের বিকল্প নেই। এই সহজ সরল গেইটগুলোই কী চমৎকার ভঙ্গীমায় বিস্ময়কর কমপিউটারটিকে কর্মক্ষম ও সচল করে তোলে তা জানবার সুযোগ আমাদের আসবে। আমরা পরবর্তীতে দেখবো কেমন করে, কোন বীজগাণিতিক নিয়ম শৃংখলার আওতায় আরও বহুবিধ উচ্চতর জটিলতর বর্তনীগুলোকে সংগঠিত করা যায়।, আপাততঃ এ পর্যন্তই। ★

Now a very powerful combination!

digitek™

DIGITEK 386DX-40	DIGITEK 486DX-33	DIGITEK 486DX2-66
80386DX-40	INTEL 80486DX-33	INTEL 80486DX2-66
40 MHz	33 MHz	66 MHz
4 MB	4 MB	4 MB
128 KB	256 KB	256 KB
1.44 MB (3.5")	1.44 MB (3.5")	1.44 MB (3.5")
210 MB	340 MB	420 MB
SUPER MINI TOWER	SUPER MINI TOWER	SUPER MINI TOWER
101 KEYS KEYBOARD	101 KEYS KEYBOARD	101 KEYS KEYBOARD
3 BUTTON	3 BUTTON	3 BUTTON

SYSTEM COMES WITH SVGA MONO MONITOR/ SVGA COLOR MONITOR 28mm, LOW RADIATION, N/I

Please Call : 817564, 323927

Sole Distributor :

IPSITA COMPUTERS PTE LTD.

78, Kazi Nazrul Islam Avenue (3rd & 4th Floor)
Farmgate, Dhaka-1215, Bangladesh.
Tel : 817564, 323927, Fax : 880-2-817564

মুঃ তারেকুস মোমেন চৌধুরী

এসো 'সি' শিখি

(পূর্ব প্রকাশিতের পর)

আমরা এ পর্যন্ত সি এর বেশ কিছু মৌলিক বিষয় নিয়ে আলোচনা করেছি। ভেরিয়েবল ডিক্লেয়ার থেকে শুরু করে এর মান ইনপুট করা আউটপুট পাওয়া এমনকি এদের প্রিন্ট আউট নেয়ার কৌশল সম্পর্কে আলোকপাত করলাম। এবার আরো কিছু মৌলিক বিষয় নিয়ে আলোচনা করি। প্রথমেই দেখি এর ডাটার ধরণ কত রকমের হতে পারে। ডাটার ধরনের দিক থেকে সি ভাষা একটি স্বয়ংসম্পূর্ণ ভাষা। এর রয়েছে প্রচুর সুবিধা। তবে সি ভাষায় ডাটার ধরণ সম্পর্কে জানার আগে আমাদের কমপিউটারের স্মৃতি এবং এর পরিমাপের বিষয়ে অবগত হওয়া প্রয়োজন। তাই সংক্ষেপে এ বিষয়ে দু-একটি কথা বলে নিতে চাই।

আমরা যেভাবে সংখ্যা বা ক্যারেক্টার স্মৃতিতে সংরক্ষণ করে কমপিউটার কিন্তু সেভাবে করেনা। যদিও বাহ্যিক দিক থেকে আমরা দেখি কমপিউটারকে আমরা সংখ্যা বা ক্যারেক্টার যেভাবে দেই সেভাবেই সে আমাদেরকে দেখাচ্ছে। যেমন আমরা 123 লিখলে কমপিউটার আমাদেরকে ঠিক সেভাবেই দেখাচ্ছে।

আসলে কমপিউটারের স্মৃতির সবচাইতে ক্ষুদ্র একক হচ্ছে বিট (Bit)। প্রকৃত পক্ষে এই বিট হচ্ছে একটি সুইচ; যাকে অন আর অফ এভাবে ধরা হয়। অন এবং অফকে সাধারণভাবে 1 এবং 0 দ্বারা সনাক্ত করা হয়।। এবং 0 সংখ্যা দ্বারা স্মৃতির সমস্ত হিসাব করা হয়। এদেরকে বলা হয় বাইনারী সংখ্যা বা বাইনারী ডিজিট (Binary Digit)। এই বাইনারী ডিজিট দ্বারাই তৈরি হয়েছে Bit। 0 বা 1 এককভাবে কিছু গঠন করতে পারেনা। যে কোন কিছু গঠন করার জন্য একাধিক বিটের দরকার অর্থাৎ এক সেট বিট মিলে হয়তো কিছু তৈরি করবে। মোটামুটি সকল কমপিউটারের জন্য স্ট্যান্ডার্ড ভাবে কিছু তৈরি করার জন্য ৮ বিট একত্রে ব্যবহার করা হয়। অর্থাৎ কমপক্ষে ৮টি বিট একত্রে যে কোন সংখ্যা বা ক্যারেক্টার বা যে কোন চিহ্ন গঠন করে। মোট ৮টি বিট একত্রে করে কিছু তৈরি করলে সেটাকে হিসেব মতে বলা হয় বাইট। আর এই ৮বিট বা ১ বাইট ২৫৬ ধরনের বিট প্যাটার্নের যে কোনটিকে প্রকাশ করতে পারে। ০ এবং ১ এর প্যাটার্নকে সজ্জিত করে পাওয়া যাচ্ছে অক্ষর বা ক্যারেক্টার, সংখ্যা ইত্যাদি। এদেরকে বলা হয় ক্যারেক্টার কোড (যেমন এসকী কোড)।

আবার একাধিক বাইট ২টি বাইট নিজেরা গ্রুপ হয়ে তৈরি করে একটা বড় ইউনিট যাকে আমরা ওয়ার্ড বলি। এটি এমন যে, একাধিক বাইট একত্র করে ওয়ার্ড তৈরি করার অর্থ হল একাধিক বাইট কমপিউটারে সংরক্ষণ বা পুনরুদ্ধার করা।

এই বিট প্যাটার্ন দিয়ে নাম্বারকে কোডিং করা হয়। অর্থাৎ ইন্টিজারকে কোডিং করা হয়। যে কোন বিট প্যাটার্নের ইন্টিজারকে খুব সহজেই দেখানো সম্ভব যে এটি ধনাত্মক না ঋণাত্মক। আবার সিঙ্গেল বাইটের ওয়ার্ডকেও একই পদ্ধতিতে দেখানো যায়।

আমরা একটি বক্সে আট বিটের জন্য মান বসিয়ে যে কোন সংখ্যার মান খুবই সহজেই দেখাতে পারি। নিচের মানের বক্স দেখুন

—128	64	32	16	8	4	2	1

বক্সে দেখা যাচ্ছে মান দ্বিগুণ হারে বাড়ছে। নাম্বার ধনাত্মক বা ঋণাত্মক হতে পারে সেটি দেখানো হয়েছে সর্ববামের বক্সের নাম্বারটিতে এটি সর্বোচ্চ পরম মান (এ বক্সের জন্য) এবং ঋণাত্মক চিহ্ন বহন করছে।

আমরা এই বক্সে যে কোন বাইনারী সংখ্যা বসিয়ে তার ডেসিমাল মান দেখতে পারি। আর একটি কথা যে সমস্ত বাইনারী সংখ্যা 0 দিয়ে শুরু সেগুলোর ডেসিমাল মান ধনাত্মক আর 1 দিয়ে শুরু হলে তা হবে ঋণাত্মক।

যেমন ধরুন 10011011 এর মান কত হতে পারে তা বক্সে সাজিয়ে সহজেই বের করতে পারি।

—128	64	32	16	8	4	2	1
1	0	0	1	1	0	1	1

=—128+0+0+16+8+0+2+1

=— 101 (ডেসিমাল—১০১)

সাধারণত যে সমস্ত ডাটা মান ব্যবহার করা হয় সেগুলো ধ্রুব থাকে। এগুলো হচ্ছে আক্ষরিক বা অক্ষর ধ্রুব। এই আক্ষরিক সমূহ নিচের যে কোন একটি টাইপের হয়— যেমন,

(ক) পূর্ণ সংখ্যা
(খ) বাস্তব সংখ্যা (বা Floating Point)
(গ) ক্যারেক্টার
(ঘ) ক্যারেক্টার স্ট্রিং

পূর্ণ সংখ্যা আবার দশমিক, হেক্সাডেসিমাল বা অক্টাল নাম্বার হতে পারে। যেমন—

সংখ্যার পূর্বে কোন কিছু না থাকলে তা দশমিক সংখ্যা, পূর্বে Ox বা Ox থাকলে তা হেক্সাডেসিমাল আর পূর্বে 0 থাকলে তাই অক্টাল সংখ্যা।

বাস্তব সংখ্যাও দুই ভাবে বর্ণনা করা যায়। একটি দশমিক সংখ্যা আরেকটি সাইন্টিফিক বা এক্সপোনেনশিয়াল ফর্মে। মূল সংখ্যার সাথে E বা e সহযোগে তা প্রকাশ হয়। যেমন—

1.234E+2 বা 1.234e+2

এবং এটির মান হচ্ছে 123.4

আবার 1.234E–2 বা 1.234e–2 যার মান .01234

এই সাইন্টিফিক সংখ্যার প্রথমটি হচ্ছে mantissa এবং পরবর্তীটি হচ্ছে দশগুণানিপাত। অনেক বড় অংক বা অনেক ছোট অংকের সংখ্যাকে এর মাধ্যমে প্রকাশ করা ভাল।

কারেক্টার প্রকাশ করার সময় সিঙ্গেল কোড '' এবং ক্যারেক্টার স্ট্রিং প্রকাশ করার সময় ডাবল কোড " " ব্যবহার করতে হয়।

ডাটা টাইপ বা ডাটার ধরন

এবার আমরা ভেরিয়েবল কি কি ধরনের হতে পারে তা দেখব। আসলে ভেরিয়েবল হচ্ছে মেমরীর লোকেশন যেখানে ডাটা সংরক্ষণ হয়। আর এই ভেরিয়েবলকে নাম দিতে হয় এবং এর ধরন অনুযায়ী তা কিন্তু ডিক্লেয়ার করতে হয়। আমরা কিন্তু ডাটার ধরন অনুযায়ী কয়েকটি ভেরিয়েবল ডিক্লেয়ার করেছি। ডাটার আকৃতি, মান ইত্যাদির উপর ভিত্তি করে সি ভাষায় পাঁচটি বিভিন্ন ডাটার ধরন বা টাইপ রয়েছে। সারণী ১.১ এ তা উল্লেখ করা হলো।

double এবং void সম্পর্কে পরে বলছি। তার আগে আমরা একই সাথে বিভিন্ন টাইপের ডাটা একই এক্সপ্রেশনে ব্যবহার করে দেখি পারি কি না? পারবো, কারণ সাধারণভাবে চিন্তা করি একটি float এবং একটি integer যোগ করে কি পাবো? অবশ্যই float। নিচের প্রোগ্রামটি দেখুন

```
# include<stdio.h>
main()
{
int a=100;
float b=25.25;
float c;
c=a+b;
printf("\n%f",c);
}
```

ডাটার ধরন	আকৃতি (বিট)	রেঞ্জ
Char	8	–128 থেকে 127
int	16	–32768 থেকে 32767
float	32	3.4E–38 থেকে 3.4E + 38
double	64	1.7E–308 থেকে 1.7E+308

সারণী–১

প্রোগ্রাম ৩৯

আউটপুট হবে

125.250000

প্রোগ্রামটির ভেরিয়েবল a তে 100 অর্থাৎ integer সংখ্যা এবং b তে float সংখ্যা 25.25 দেয়া হয়েছে। ভেরিয়েবল c কে float হিসেবে ডিক্লেয়ার করা হয়েছে। কিন্তু c কে integer হিসেবে ডিক্লেয়ার করা হলে দশমিকের পরের অংকগুলো বাদ দিয়ে আউটপুট দিবে। এক্ষেত্রে ভুল ফলাফল দেয়ার সম্ভাবনাও থাকতে পারে। কাজেই এমন কাজ কখনো করা ঠিক হবে না। যদিও এ প্রোগ্রামটির ক্ষেত্রে দশমিক বিন্দুর পরের অংকগুলো বাদ দিয়ে মোটামুটি সঠিক ফলাফল দিবে। তাহলে দেখুন—

```
# include<stdio.h>
main()
{
int a=100;
float b=25.25;
int c;
c=a+b;
printf("\n%d",c);
}
```

প্রোগ্রাম ৪০

প্রোগ্রামটির আউটপুট হবে ঃ125

আমি আবারও বলছি এমন কাজ করা উচিত না। আরেকটি প্রোগ্রাম দেখুন—

```
# include<stdio.h>
main()
{
int a=32767;
float b=50;
int c;
c =a+b;
printf ("\n%d",c);
}
```

প্রোগ্রাম ঃ ৪১

আউটপুট হবে ঃ –32719

একটি ভুল ফলাফল দিল। এটি আসলে ঘটেছে রেঞ্জ এর সমস্যার কারণে। অর্থাৎ মেমরীতে এলোকেট করা হয়েছে এক রকম কিন্তু ভেরিয়েবল এনাইন করা হয়েছে অন্য রকম। পরবর্তীতে নাম্বারটি এমন আকৃতির হয়েছে যে, তার জন্য নির্ধারিত জায়গায় খাপ খাইয়ে বসতে সক্ষম হয়নি।

দুইটি পূর্ণ সংখ্যা নিয়ে একটিকে অপরটি দ্বারা ভাগ করলে পূর্ণসংখ্যা হতে পারে বা ভগ্নাংশও হতে পারে। এমন অবস্থায় আউটপুটটাকে ভগ্নাংশ হিসেবে ফলাফলটি দেয়ার জন্য বাধ্য করার চেষ্টা করব। তার জন্য একটি কাস্ট (cast) অপারেটর ব্যবহার করব। নিচের প্রোগ্রামে এটির ব্যবহার দেখানো হলো—

```
# include<stdio.h>
main()
{
```

```
int a, b;
a=3;
b=2;
printf ("\n%f", (float)a/b);
}
```

প্রোগ্রাম ঃ ৪২

আউটপুট হবে ঃ 1.500000

প্রোগ্রামটিতে এক্সপ্রেশনে float কথাটি লিখে ভাগফলটি ভগ্নাংশ হিসেবে দেয়ার জন্য বাধ্য করা হয়েছে। কোন ডাটা না হারিয়ে সঠিক ফলাফল দেয়ার জন্য এটি দেয়া হয়েছে। এটিকে বলা হয় Cast operator।

রেঞ্জ সমস্যার চাইতে কিছুটা ভিন্ন সমস্যা হল প্রিসিশন সমস্যা। অর্থাৎ আকারের সমস্যা। কারণ float কিন্তু সাত ডিজিট পর্যন্ত ডাটা দেখাবে, তার বেশি হলে ভুল করবে। ব্যাপারটির একটি উদাহরণ দিয়ে দেখা যাক। এখানে সাত ডিজিটের বেশি অংকের সংখ্যা ব্যবহার করে দেখি–

```
# include<stdio.h>
main()
{
float a;
a=123456789.0;
printf("%f",a);
}
```

প্রোগ্রাম ঃ ৪৩

আউটপুট পাবেন ঃ 123456792.000000

ব্যাপারটি ঘটেছে সাত ডিজিটের বেশি অংকের সংখ্যা দেয়ার কারণে, এখানে ব্যবহার করতে হবে double। double টাইপটি float টাইপের চেয়ে দ্বিগুণ হিসেবে ব্যবহৃত হয়। এটি ১৫ ডিজিট পর্যন্ত গ্রহণ করে। নিচের প্রোগ্রামটি দেখুন–

```
# include<stdio.h>
main()
{
double a;
a=123456789.0;
printf ("%f", a);
}
```

প্রোগ্রাম ঃ ৪৪

এবার / আউটপুট / পাবো: 123456789.000000

ডাটা টাইপ মডিফাইয়ার

আমরা আগেই দেখেছি পাঁচ ধরনের ডাটা উল্লেখ করা যায়। এই পাঁচ ধরনের ডাটার মাধ্যমে সব ভেরিয়েবল ডিক্লেয়ার সম্ভব নয়। কয়েকটি বিশেষ শব্দ সংযোজনের মাধ্যমে এদেরকে আরও ব্যাপকভাবে ব্যবহার করা যায়। এই বিশেষ শব্দগুলোকে বলা হয় ডাটা টাইপ মডিফাইয়ার। এর হল signed, unsigned, long এবং short। এই মডিফাইয়ার ব্যবহার করে ডাটা টাইপের ব্যাপকতা ১.২ নং সারণীতে দেখানো হল। তবে এই মডিফাইয়ারগুলো void এর সাথে ব্যবহার করা যায় না।

ডাটার ধরন	আকৃতি (বিট)	রেঞ্জ
Char	8	–128 থেকে 127
unsigned char	8	0 থেকে 225
singned char	8	–128 থেকে 127
int	16	–32768 থেকে 32767
short	16	–32768 থেকে 32767
short int	16	–32768 থেকে 32767
unsigned	16	0 থেকে 65535
unsigned int	16	0 থেকে 65535
unsigned short	16	0 থেকে 65535
signed	16	–32768 থেকে 32767
signed int	16	–32768 থেকে 32767
long	32	–2147483648 থেকে 2147483647
long int	32	–2147483648 থেকে 2147483647
unsigned long	32	0 থেকে 4294967295
signed long	32	–2147483648 থেকে 2147483647
float	32	3.4E–38 থেকে 3.4E+38
double	64	1.7E–308 থেকে 1.7E+308
long double	80	3.4E–4932 থেকে 1.1E+4932

সারণী– ১.২

আউটপুট ফরমাট ঃ

printf ফাংশনটির সাথে আমরা স্পেসিফাইয়ার ব্যবহার দেখেছি। ব্যবহারিক অর্থসহ সম্ভাব্য সকল কনভারশন স্পেসিফাইয়ার এর তালিকা ১.৩ নং সারণীতে দেয়া হল।

আমরা কিন্তু এসব স্পেসিফাইয়ারের ব্যবহার দেখেছি। তবু একটি প্রোগ্রামের মাধ্যমে পূর্ণ সংখ্যাকে হেক্সাডেসিমলে এবং অক্টাল নাম্বারে পরিবর্তন করা দেখি।

```
# include<stdio.h>
main()
{
int a=65;
printf ("\n Number is %d",a);
printf("\n Hexadecimal Number is %x",a);
printf ("\n Octal Number is %0",a);
}
```

প্রোগ্রাম ঃ ৪৫

আউটপুট পাবো–

Number is 65

Hexadecimal Number is 41

Octal Number is 101

%e ব্যবহার করে –100 সাইন্টিফিক সংখ্যা কেমন হবে দেখি নিচের প্রোগ্রামে–

```
# Include<stdio.h>
main()
{
float x =–100;
printf ("\n The number is %f", x);
printf("\n And Scientific notation is %e",x);
}
```

প্রোগ্রাম ঃ ৪৬

আউটপুট পাবো–

The number is –100

And Scientific notation is –1.000000e+02

কনভারশন স্পেসিফাইয়ার	আউটপুটের জন্য ব্যবহারিক অর্থ
%C	ক্যারেক্টার
%S	ক্যারেক্টার স্ট্রিং
%d	পূর্ণ সংখ্যা
%i	পূর্ণ সংখ্যা
%f	ভগ্নাংশ
%e বা %E	সাইন্টিফিক সংখ্যার জন্য
%g বা %G	সাইন্টিফিক সংখ্যার জন্য
%u	আনসাইনড পূর্ণ সংখ্যা
%o	অক্টাল সংখ্যা
%x	হেক্সাডেসিমাল সংখ্যা
%p	পয়েন্টার
%n	ইন্টিজার পয়েন্টার
%%	% চিহ্নটি প্রদর্শনের জন্য

সারণী– ১.৩

(চলবে)

pin point your choice

massive COMPUTERS Dial 862856

95/1 New Elephant Road, Zinnat Mansion, 1st floor, Dhaka 1205

massive PROFESSIONAL PC COMPUTERS

we deserve your desire...