

অধ্যায় - ৫

## প্রোগ্রামিং ভাষা

### Programming Language

#### ৫.১ প্রোগ্রামের ধারণা (Concept of Program)

কম্পিউটার একটি প্রোগ্রাম নিয়ন্ত্রিত যন্ত্র। প্রোগ্রাম ছাড়া কম্পিউটার দিয়ে কোনো কাজ করা যায় না। প্রোগ্রাম বিহীন কম্পিউটার একটি কর্মহীন নিঝীর জড় বস্তু ছাড়া আর কিছুই নয়। কম্পিউটার দিয়ে কোনো কাজ করতে হলে সেই কাজের উপযোগী প্রোগ্রামের প্রয়োজন হয়।

কম্পিউটার একটি ইলেক্ট্রনিক যন্ত্র। প্রতিটি ইলেক্ট্রনিক যন্ত্রের মতো কম্পিউটার কেবল বিদ্যুতের উপস্থিতি আর অনুপস্থিতি বুঝতে পারে। বিদ্যুতের উপস্থিতিকে বাইনারি ১ আর অনুপস্থিতিকে বাইনারি ০ দ্বারা বোঝানো হয়। এক একগুচ্ছ বাইনারী ০ ও ১ দ্বারা এক একটি নির্দেশ গঠিত হয়, যা দ্বারা কম্পিউটার কোনো একটি নির্দিষ্ট কাজ করে থাকে। কম্পিউটারের বোধগম্য এক্লপ অসংখ্য নির্দেশ পরপর সাজিয়ে প্রোগ্রাম তৈরি করা হয়। প্রতিটি প্রোগ্রাম একটি নির্দিষ্ট ধরনের সমস্যার সমাধান করে তথা একটি নির্দিষ্ট ধরনের কাজ করে।

বাইনারি ০ ও ১ দ্বারা যে সকল প্রোগ্রাম লেখা হয় তা কম্পিউটার সরাসরি বুঝতে পারে। এজন্য এ ভাষাকে কম্পিউটারের নিজস্ব ভাষা বলা হয়। একে মেশিন ভাষা বা যান্ত্রিক ভাষা বলা হয়। বিজ্ঞানীদের অঙ্গুত্ব পরিশ্রমের ফলে মেশিন ভাষা বা যান্ত্রিক ভাষা ছাড়াও আরো অনেক কৃত্রিম ভাষার সৃষ্টি হয়েছে। এ সকল কৃত্রিম ভাষায় ০ বা ১ এর পরিবর্তে সাধারণ ভাষার (ইংরেজি ভাষার) শব্দাবলী প্রয়োগ করে কম্পিউটারে আদেশ নির্দেশ প্রদান করা যায় তথা প্রোগ্রাম রচনা করা যায়।

(কোনো সমস্যা সমাধানের জন্য তথা একটি নির্দিষ্ট কাজ করার জন্য সম্পাদনের অনুক্রমে সাজানো কম্পিউটারের বোধগম্য ভাষায় লেখা ধারাবাহিক নির্দেশের সমষ্টিকে বলা হয় প্রোগ্রাম। কম্পিউটারকে তার বোধগম্য ভাষায় নির্দিষ্ট নিয়ম অনুসারে বর্ণ, শব্দ, বাক্য বা বিশেষ চিহ্ন সাজিয়ে নির্দেশ প্রদান করা হয়। কম্পিউটারের বোধগম্য এক্লপ নির্দেশমালার সাহায্যে তৈরি হয় এক একটি প্রোগ্রাম। প্রোগ্রাম বা প্রোগ্রাম সমষ্টিকে বলা হয় সফটওয়্যার।)

#### ৫.২ প্রোগ্রামের ভাষা (Programming Language)

মানুষের ভাব প্রকাশের মাধ্যম হলো ভাষা। আমরা নিজেদের মধ্যে মনের ভাব প্রকাশ ও পারস্পরিক যোগাযোগের জন্য ভাষা ব্যবহার করি। ভাষার ব্যবহার কেবলমাত্র বুদ্ধিমান প্রাণিদের মধ্যে সীমাবদ্ধ। তবে বর্তমান কালে কম্পিউটারসহ অন্যান্য স্মার্ট ডিভাইজ দিয়ে নানা ধরনের কাজ পরিচালনার জন্য তৈরি করা হয়েছে যন্ত্রের উপযোগী ভাষা। এ সকল ভাষার মাধ্যমে মানুষের সাথে যন্ত্রের যোগাযোগ স্থাপিত হয়। যন্ত্রের উপযোগী এ সকল ভাষাকে প্রোগ্রামের ভাষা বলা হয়।

কোনো সমস্যা সমাধানের জন্য তথা কোনো একটি কাজ করার জন্য সম্পাদনের অনুক্রমে সাজানো কম্পিউটারের বোধগম্য ভাষায় লেখা ধারাবাহিক নির্দেশের সমষ্টিকে বলা হয় প্রোগ্রাম। কম্পিউটারকে তার বোধগম্য ভাষায় নির্দিষ্ট নিয়ম অনুসারে বর্ণ, শব্দ বাক্য বা বিশেষ চিহ্নকে সাজিয়ে নির্দেশ প্রদান করা হয়। কম্পিউটারের বোধগম্য এক্লপ অসংখ্য নির্দেশমালার সাহায্যে তৈরি হয় একটি প্রোগ্রাম। তাই বলা যায়, কম্পিউটারের বোধগম্য ভাষা যার সাহায্যে বর্ণ, শব্দ বা বিভিন্ন অক্ষর নির্দিষ্ট নিয়ম অনুসারে সাজিয়ে প্রোগ্রাম রচনা করা হয়, তাকে প্রোগ্রামের ভাষা

বলে। অর্থাৎ প্রোগ্রাম রচনায় ব্যবহৃত শব্দাবলী এবং নিয়ম কানুনের সমষ্টিকে প্রোগ্রামের ভাষা বলা হয়।  
সহজভাবে বলা যায়, কম্পিউটারে প্রোগ্রাম তৈরির জন্য ব্যবহৃত ভাষাসমূহকে প্রোগ্রামের ভাষা বলা হয়।

কম্পিউটার বিজ্ঞানীদের অক্লান্ত পরিশ্রমের ফলে অনেক কৃতিম ভাষার সৃষ্টি হয়েছে। যেমন- বেসিক(BASIC), ফোরট্রান(FORTRAN), কোবল(COBOL), সি(C), সি++(C++), অ্যালগল(Algol), প্যাসকেল(Pascal), ভিজুয়াল বেসিক(Visual Basic), জাভা(Java), ওরাকল(Oracle), পাইথন(Python) ইত্যাদি।

প্রোগ্রামের ভাষার প্রকারভেদঃ গঠন ও বৈশিষ্ট্য অনুসারে প্রোগ্রামের ভাষাকে ৫ ভাগে ভাগ করা হয়েছে। যেমন-  
ক) প্রথম প্রজন্মের ভাষা(১৯৪৫) : মেশিন ভাষা বা যান্ত্রিক ভাষা (Machine Language)

খ) দ্বিতীয় প্রজন্মের ভাষা(১৯৫০) : অ্যাসেম্বলি ভাষা (Assembly Language)

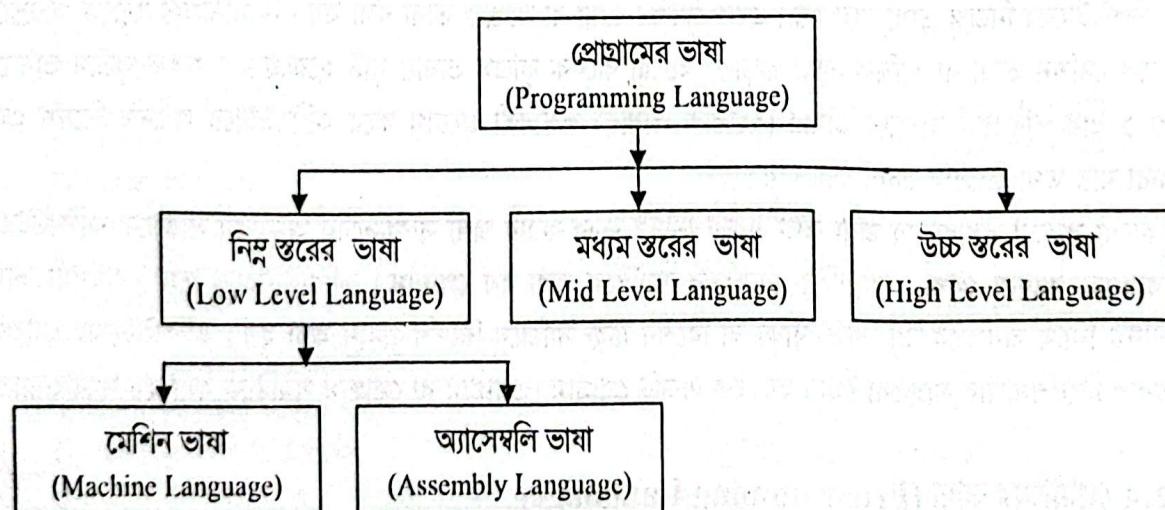
গ) ভূতীয় প্রজন্মের ভাষা(১৯৬০) : উচ্চতর ভাষা (High Level Language)

ସ) ମର୍ଗ୍ୟ ପ୍ରକଳ୍ପର ଭାଷା (୧୧୫୦) : EGL ବା ଅନ୍ତିମ ଭାଷା (Very High Level Language).

১৮) চতুর্থ প্রজন্মের ভাষা (১৯৭০) : 4GL বা আরও উচ্চতর ভাষা (Very High Level Language)

ঙ) পঞ্চম প্রজন্মের ভাষা(১৯৮০) : স্বাভাবিক ভাষা (Natural Language) →

মেশিন ভাষা এবং অ্যাসেম্বলী ভাষাকে নিম্নস্তরের ভাষা বলা হয়। এছাড়া মধ্যম স্তরের ভাষা নামক একটি ভাষা আছে। নিম্ন প্রেস্থামের ভাষার স্তর ভিত্তিক শ্রেণি বিভাগ দেখানো হলোঃ



## ৫.৩ মেশিন ভাষা (Machine Language)

যে প্রোগ্রামের ভাষায় নির্দেশগুলো বাইনারী ০ এবং ১ কিংবা হেক্সাডেসিম্যাল সংখ্যার সাহায্যে প্রোগ্রাম রচনা করা হয় তাকে মেশিন ভাষা বা যান্ত্রিক ভাষা বলা হয়। মেশিন ভাষা বা যান্ত্রিক ভাষা হলো প্রথম প্রজন্মের ভাষা। মেশিন ভাষা কম্পিউটারের নিজস্ব ভাষা। মেশিন ভাষার নির্দেশ কম্পিউটার সরাসরি বুঝতে পারে। মেশিন ভাষা ছাড়া অন্য ভাষার নির্দেশ কম্পিউটার সরাসরি বুঝতে পারে না। তাই অন্য ভাষায় লেখা প্রোগ্রাম নির্বাহের আগে কম্পিউটার উপর্যুক্ত অনুবাদক প্রোগ্রামের সাহায্যে এ সকল ভাষার প্রোগ্রামকে মেশিন ভাষায় পরিণত করে। এ সকল অনুবাদিত প্রোগ্রাম কম্পিউটারের বোধগম্য এবং নির্বাহযোগ্য হয়।

মেশিন ভাষায় সবচেয়ে কম লজিক ব্যবহার করে প্রোগ্রাম লেখা যায়। এ ভাষায় লেখা প্রোগ্রাম খুব কম মেমরিতে সংরক্ষণ করা যায় এবং প্রোগ্রাম নির্বাহের জন্য সবচেয়ে কম মেমরির প্রয়োজন হয়।

প্রথম প্রজন্মের প্রায় সকল কম্পিউটারেই মেশিন ভাষায় প্রোগ্রাম লেখা হতো। মেশিন ভাষার প্রোগ্রামকে বলা হয় অবজেক্ট কোড। মেশিন ভাষা ছাড়া অন্য ভাষার প্রোগ্রামকে বলা হয় সোর্স কোড।

**উদাহরণ-** নিচে দুটি সংখ্যা যোগ করার জন্য মেশিন ভাষার একটি নমুনা প্রোগ্রাম নিম্নে তুলে ধরা হলো-

নমুনা নির্দেশ	নির্দেশের কাজ
১০০০ ০০০০	(Start execution)
১০০০ ০০০১ ০০১১	(Read a).
১০০০ ০০১০ ০১০১	(Add b with a),
১০০০ ০০১১ ১০০০	(Store result in c, c=a+b)
১১১১ ১১১১	(Stop execution)

মেশিন ভাষার নির্দেশকে চার ভাগে ভাগ করা যায়। যেমন-

১. গাণিতিক নির্দেশ - যোগ, বিয়োগ, গুণ, ভাগ।
২. নিয়ন্ত্রণমূলক নির্দেশ - লোড, স্টোর ও জাম্প।
৩. ইনপুট-আউটপুট নির্দেশ - রিড ও রাইট।
৪. প্রত্যক্ষ ব্যবহারের নির্দেশ - স্টার্ট, হল্ট ও ইন্ড।

### মেশিন ভাষার সুবিধা

১. সবচেয়ে কম লজিক ব্যবহার করে প্রোগ্রাম লেখা যায়।
২. এ ভাষার প্রোগ্রাম চালানোর জন্য সবচেয়ে কম মেমরির প্রয়োজন হয়।
৩. মেশিন ভাষা কম্পিউটার সরাসরি বুঝতে পারে।
৪. প্রোগ্রাম রান করার জন্য অনুবাদক প্রোগ্রামের প্রয়োজন হয় না।
৫. এ ভাষার প্রোগ্রাম কম্পিউটার সবচেয়ে দ্রুত নির্বাহ করতে পারে।
৬. কম্পিউটারের অভ্যন্তরীণ গঠন জানতে মেশিনের ভাষা জানা আবশ্যিক।
৭. বৈদ্যুতিক বর্তনীর ভুল-ক্রটি নির্ণয়ে এ ভাষা ব্যবহার করা হয়।

### মেশিন ভাষার অসুবিধা

১. মেশিন ভাষায় প্রোগ্রাম লেখা প্রোগ্রামারদের জন্য কষ্টকর ও সময় সাপেক্ষ।
২. মেশিন ভাষায় প্রোগ্রাম লেখার জন্য নির্দেশ তালিকার সহায়তা নিতে হয়।
৩. কম্পিউটারের অভ্যন্তরীণ গঠন জানা না থাকলে এ ভাষায় প্রোগ্রাম রচনা করা যায় না।
৪. এ ভাষা মেশিন নির্ভর ভাষা। তাই এক কম্পিউটারের জন্য লেখা প্রোগ্রাম শুধুমাত্র ঐ কম্পিউটারে চলবে, অন্য কম্পিউটারে চালানো যাবে না।
৫. মেশিন ভাষায় প্রোগ্রাম লিখতে দক্ষ প্রোগ্রামার দরকার হয়।
৬. মেশিন ভাষায় লেখা প্রোগ্রাম পরিবর্তন করা কষ্টসাধ্য।

## ৫.৪ অ্যাসেম্বলি ভাষা (Assembly Language)

অ্যাসেম্বলি ভাষা হলো দ্঵িতীয় প্রজন্মের ভাষা। মেশিন ভাষা বা যান্ত্রিক ভাষার ০ ও ১-এর সাহায্যে দেওয়া বাইনারী নির্দেশকে কয়েকটি ইংরেজি বর্ণ বা শব্দ সংক্ষেপ দ্বারা প্রতিস্থাপিত করে তৈরি নতুন ভাষাকে অ্যাসেম্বলি ভাষা বলা হয়। অ্যাসেম্বলি ভাষার এ শব্দ সংক্ষেপসমূহকে নেমোনিক (Mnemonic) কোড বলা হয়। তাই বলা যায়, যে ভাষায় ইংরেজি ভাষার শব্দ সংক্ষেপ বা নেমোনিক কোড দ্বারা প্রোগ্রাম তৈরি করা হয় তাকে অ্যাসেম্বলি ভাষা বলা হয়। ধরা যাক, যান্ত্রিক ভাষায় ১০০১ ১১০১ বিটগুচ্ছ দ্বারা যোগের নির্দেশ বুঝানো হয়। অ্যাসেম্বলি ভাষায় এ বিটগুচ্ছকে ADD শব্দ দ্বারা প্রতিস্থাপিত করা হয়েছে। কিংবা যান্ত্রিক ভাষায় ১০০১ ১১১০ বিটগুচ্ছ দ্বারা বিয়োগের নির্দেশ বুঝানো হয়। অ্যাসেম্বলি ভাষায় এ বিটগুচ্ছকে SUB শব্দ দ্বারা প্রতিস্থাপিত করা হয়েছে। এভাবে যান্ত্রিক ভাষার ০ ও ১-এর সাহায্যে দেওয়া বাইনারী নির্দেশকে কয়েকটি ইংরেজি বর্ণ বা শব্দ সংক্ষেপ দ্বারা প্রতিস্থাপিত করা হয়েছে অ্যাসেম্বলি ভাষায়। অ্যাসেম্বলি ভাষার নির্দেশগুলো ০ ও ১ দিয়ে দেওয়া হয় না বলে কম্পিউটার এ ভাষার নির্দেশ সরাসরি বুঝতে পারে না। অ্যাসেম্বলি ভাষার নির্দেশকে মেশিনের ভাষায় রূপান্তরের জন্য অ্যাসেম্বলার নামক অনুবাদক প্রোগ্রামের প্রয়োজন হয়। মেশিন ভাষার মতো এ ভাষাও মেশিন নির্ভর। তাই এক কম্পিউটারের জন্য তৈরি অ্যাসেম্বলি ভাষার প্রোগ্রাম অন্য কম্পিউটারে ব্যবহার করা যায় না। প্রতিটি ভিন্ন ভিন্ন গঠনের কম্পিউটারে ব্যবহৃত অ্যাসেম্বলি ভাষার নির্দেশ আলাদা আলাদা হয়।

অ্যাসেম্বলি ভাষার নির্দেশে ৪টি অংশ থাকে। যথা-

১. লেবেল (Label)
২. অপকোড(Opcode)
৩. অপারেণ্ড(Operand) এবং
৪. মন্তব্য (Comments)।

**লেবেল (Label)** : এক স্থান হতে অন্য স্থানে প্রোগ্রামের নিয়ন্ত্রণ স্থানান্তরের জন্য লেবেল ব্যবহার করা হয়। আবার প্রোগ্রামের কোনো একটি স্থানকে চিহ্নিত করার জন্যও লেবেল ব্যবহার করা হয়। লেবেল ১টি বা ২টি অক্ষর নিয়ে গঠিত। ২টি অক্ষর নিয়ে গঠিত লেবেলের মাঝে কোনো ফাঁকা স্পেস ব্যবহার করা যাবে না।

**অপকোড (Opcode)** : অপকোড হলো অপারেশন কোড। অপকোড বা অপারেশন কোড কম্পিউটারকে নির্দিষ্ট কাজের নির্দেশনা প্রদান করে। অ্যাসেম্বলি ভাষার নেমোনিকগুলো হলো অপকোড। অপকোডের মাধ্যমে কম্পিউটার বুঝতে পারে তাকে কি কাজ করতে হবে।

**অপারেণ্ড (Operand)** : অপকোড-এর সাথে যুক্ত মেমরি বা রেজিস্টার অ্যাড্রেসগুলো হলো অপারেণ্ড। অপারেণ্ড অংশে প্রয়োজনীয় ডেটা বা ফলাফল জমা থাকে।

**মন্তব্য (Comments)** : প্রোগ্রামের কাজ কিংবা প্রোগ্রামের নির্দিষ্ট অংশের কাজ বোঝানোর জন্য প্রোগ্রামে মন্তব্য যোগ করতে হয়। এতে প্রোগ্রামের বিভিন্ন অংশের কাজ বুঝতে সুবিধা হয়। প্রোগ্রামে মন্তব্য না থাকলে নিজের লেখা প্রোগ্রামও একজন প্রোগ্রামারের বুঝতে কষ্ট হয়।

### অ্যাসেম্বলি ভাষার কিছু নেমোনিক ও এর ব্যবহার(Mnemonics & its uses)

অ্যাসেম্বলি ভাষা মেশিন নির্ভর ভাষা হওয়ায় এর নেমোনিক কম্পিউটার ভেদে ভিন্ন ভিন্ন হয়। তবে অধিকাংশ ক্ষেত্রে যে সকল নেমোনিক ব্যবহার হয়, সেগুলোর নামসহ ব্যবহার নিম্নে থেকে দেখানো হলোঃ

নেমোনিক	ব্যবহার
ADD	অ্যাকুমুলেটরে সংরক্ষিত সংখ্যাটির সাথে মেমরির নির্দিষ্ট অ্যাড্রেসের সংখ্যাটি যোগ কর।
SUB	অ্যাকুমুলেটরে সংরক্ষিত সংখ্যাটি থেকে মেমরির নির্দিষ্ট অ্যাড্রেসের সংখ্যাটি বিয়োগ কর।
MUL	অ্যাকুমুলেটরে সংরক্ষিত সংখ্যাটিকে মেমরির নির্দিষ্ট অ্যাড্রেসের সংখ্যাটি দিয়ে গুণ কর।
DIV	অ্যাকুমুলেটরে সংরক্ষিত সংখ্যাটিকে মেমরির নির্দিষ্ট অ্যাড্রেসের সংখ্যাটি দিয়ে ভাগ কর।
CLR	অ্যাকুমুলেটর পরিষ্কার কর।
STA	মেমরির নির্দিষ্ট অ্যাড্রেসের সংখ্যাটি অ্যাকুমুলেটরে স্থানান্তর কর।
STO	অ্যাকুমুলেটরের সংখ্যাটি মেমরির নির্দিষ্ট অ্যাড্রেসের সংরক্ষণ কর।
JMP	পরবর্তী নির্দেশের জন্য মেমরির নির্দিষ্ট অ্যাড্রেসে যাও(জাস্প কর)।
INP	ব্যবহারকারী কর্তৃক ইনপুটকৃত ডেটাটি মেমরির নির্দিষ্ট অ্যাড্রেসে জমা কর।
OUT	মেমরির নির্দিষ্ট অ্যাড্রেসে জমাকৃত ফলাফলটি আউটপুট অংশে প্রকাশ কর।
STP	প্রোগ্রামের নির্দেশের নির্বাহ থামাও।

উদাহরণ : অ্যাসেম্বলি ভাষায় দুটি সংখ্যার যোগফল নির্ণয়ের প্রোগ্রাম।

নির্দেশ	ব্যবহার
CLR	অ্যাকুমুলেটর পরিষ্কার কর।
INP:A	ব্যবহারকারী কর্তৃক ইনপুটকৃত ডেটাটি মেমরির A অ্যাড্রেসে জমা কর।
STA	A অ্যাড্রেসের সংখ্যাটি অ্যাকুমুলেটরে স্থানান্তর কর।
INP:B	ব্যবহারকারী কর্তৃক ইনপুটকৃত ডেটাটি মেমরির A অ্যাড্রেসে জমা কর।
ADD:B	অ্যাকুমুলেটরের সংখ্যাটির সাথে B অ্যাড্রেসের সংখ্যাটি যোগ কর।
STO:C	অ্যাকুমুলেটরের সংখ্যাটি মেমরির C অ্যাড্রেসে জমা কর।
OUT:C	C অ্যাড্রেসের ফলাফলটি আউটপুট অংশে প্রকাশ কর।
STP	প্রোগ্রামের নির্দেশের নির্বাহ থামাও।

### অ্যাসেম্বলি ভাষার সুবিধা (Advantages of Assembly Language)

- অ্যাসেম্বলি ভাষায় দক্ষ ও সংক্ষিপ্ত প্রোগ্রাম রচনা করা সম্ভব।
- প্রোগ্রাম রচনায় মেমরি অ্যাড্রেসের সরাসরি ব্যবহারের প্রয়োজন হয় না।
- মেশিন ভাষা অপেক্ষা এ ভাষায় প্রোগ্রাম রচনা করা সহজ ও শ্রম সাধ্যী।
- এ ভাষার প্রোগ্রামের ভুল সংশোধন করা মেশিন ভাষা অপেক্ষা সহজতর।

## উচ্চতরের প্রোগ্রামের ভাষার প্রকারভেদ:

ব্যবহার ব্যাপকতার উপর নির্ভর করে প্রোগ্রামের ভাষাকে দুইভাগে ভাগ করা যায়। যথা-

১. সাধারণ প্রয়োগের ভাষা (General Purpose Language)

২. বিশেষ প্রয়োগের ভাষা : (Special Purpose Language)

সাধারণ প্রয়োগের ভাষা : যে সকল প্রোগ্রামের ভাষার সাহায্যে সকল প্রকারের প্রোগ্রাম তৈরি করা যায় তাকে সাধারণ প্রয়োগের প্রোগ্রামের ভাষা বলে। এ সকল ভাষার সাহায্যে বিবিধ ক্ষেত্রের সমস্যা সমাধানের প্রোগ্রাম রচনা করা যায়। এ সকল প্রোগ্রামের ভাষা সিস্টেম সফটওয়্যার এবং অ্যাপ্লিকেশন সফটওয়্যার- উভয় ধরনের সফটওয়্যার ডিজাইনে সমান দক্ষতা সম্পন্ন।

উদাহরণ : সি, সি ++, প্যাসকেল, বেসিক, কোবল ইত্যাদি সাধারণ প্রয়োগের ভাষা।

বিশেষ প্রয়োগের ভাষা : যে সকল প্রোগ্রামের ভাষার সাহায্যে কেবলমাত্র একটি বিশেষ ক্ষেত্রে কাজের উপযোগী প্রোগ্রাম তৈরি করা যায় তাকে বিশেষ উদ্দেশ্যে প্রয়োগের ভাষা বলে। এ সকল প্রোগ্রামের ভাষায় একটি বিশেষ ব্যবহারিক কাজের দক্ষতা থাকে। বিশেষ উদ্দেশ্যে ব্যবহারের প্রোগ্রামের ভাষার ক্ষেত্রে দক্ষ কম্পাইলার ব্যবহার করা হয়।

### কয়েকটি বিশেষ প্রয়োগের ভাষা :

- i. টেক্স্ট প্রসেসিং ভাষা : Tex, Latex ইত্যাদি।
- ii. টেক্স্ট ডিসপ্লে : HTML
- iii. ডেটাবেজ তৈরি/মিথস্ক্রিপ্ট : SQL
- iv. সিদ্ধালিক ম্যাথ : Mathematica, Matlab
- v. পরিসংখ্যান : R, SPSS
- vi. কম্পিউটার গেইমস : Maya, Unreal Engine

ব্যবহার ক্ষেত্রের উপর ভিত্তি করে প্রোগ্রামের ভাষাকে আবার বিভিন্ন ভাগে ভাগ করা যায়। নিম্নে এরূপ কয়েকটি ব্যবহার ক্ষেত্রের দৃষ্টান্ত দেওয়া হলো :

**বাণিজ্যিক প্রয়োগের ভাষা :** বাণিজ্যিক কাজে ব্যবহৃত প্রোগ্রাম তৈরিতে এ ধরনের প্রোগ্রামের ভাষা ব্যবহার করা হয়। যেমন- COBOL একটি বাণিজ্যিক প্রয়োগের ভাষা। কোবল ভাষার সাহায্যে সহজে রেকর্ড এন্ট্রি ও রিপোর্ট প্রস্তুত করা যায়। বিভিন্নভাবে তথ্যকে সারণিতে প্রদর্শনের দক্ষতার কারণে বাণিজ্যিক প্রোগ্রাম তৈরিতে COBOL সমাদৃত।

**বৈজ্ঞানিক প্রয়োগের ভাষা :** গাণিতিক সমস্যা সমাধানের প্রোগ্রাম তৈরিতে এ সকল ভাষা ব্যবহার করা হয়। এ সকল ভাষার সাহায্যে বিভিন্ন প্রকার সমীকরণ সিরিজ, ভেষ্টর, মেট্রিক্স বহুবিধি গাণিতিক সমস্যার সমাধান দক্ষতার সাথে করা যায় এবং এ সংক্রান্ত তৈরি লাইব্রেরি ফাংশনের সুবিধা ও পাওয়া যায়। এ ধরনের কয়েকটি উল্লেখযোগ্য প্রোগ্রামের ভাষা হলো- FORTRAN, ALGOL ইত্যাদি।

## কয়েকটি উচ্চতর ভাষার পরিচিতি

৫.৬.১ সি (C) : সি একটি সিস্টেমস্ প্রোগ্রামিং ভাষা, যা ডেনিস রিচি ও Ken Thompson বেল ল্যাবরেটরিতে ১৯৬৯-১৯৭৩ সালের মধ্যে উন্নয়ন করেন। সি একটি জনপ্রিয় উচ্চতর প্রোগ্রামিং ভাষা। এটি একটি সাধারণ উদ্দেশ্যে ব্যবহৃত (General Purpose) স্ট্রাকচার্ড প্রোগ্রামিং ভাষা(Structured Programming Language)। ১৯৭০ সালে যুক্তরাষ্ট্রের এটিএ্যাভটি বেল ল্যাবরেটরি (AT&T Bell Laboratory)-তে ডেনিস রিচি (Dennis Ritchie) সি ভাষা(C Language) উভাবন করেন। তিনি DEC PDP -II নামক কম্পিউটারে ব্যবহারের উদ্দেশ্যে ইউনিক্স(Unix) অপারেটিং সিস্টেম তৈরির জন্য এ ভাষা উভাবন করেন। এটি মূলত মার্টিন রিচার্ডস (Martin Richards)-এর উভাবিত BCPL(Basic Combined Programming Language) নামক ভাষা উন্নতকরণের মাধ্যমে উভাবিত একটি নতুন প্রোগ্রামিং ভাষা। বেল ল্যাবরেটরিতে উভাবিত BCPL নামক ভাষাটি পরবর্তীতে B ভাষা হিসেবে পরিচিতি লাভ করে। এজন্য ডেনিস রিচি তার উভাবিত নতুন ভাষার নামকরণ করেন সি ভাষা, যা B ভাষার বর্ধিত পরিমার্জিত রূপ। ১৯৭৮ সাল পর্যন্ত এটি শুধুমাত্র বেল ল্যাবরেটরিতে ব্যবহৃত হয়েছিলো। ১৯৭৮ সালে ব্রেইন কার্নিংহান (Brain Kerninghan) ও ডেনিস রিচি(Dennis Ritche) এতে নানাবিধ ফিচার সংযুক্ত করেন এবং চূড়ান্ত রূপ দান করেন। মূলত ১৯৭৮ সালে ডেনিস রিচির লেখা “The C Programming Language” বইটি প্রকাশের পর এবং মাইক্রো কম্পিউটারের জনপ্রিয় হবার কারণে সি ভাষা ব্যবহার করে ব্যাপকভাবে প্রোগ্রাম তৈরি হতে থাকে।

১৯৮০ সালের মধ্য হতে এটি ব্যাপক জনপ্রিয়তা পায়। সি ভাষার সহজবোধ্যতা, কম্পাইলার প্রাণ্টির সুবিধা, ব্যবহারের ব্যাপকতা ইত্যাদি নানাবিধ সুবিধা সি ভাষাকে প্রোগ্রামিং ভাষা শিক্ষার একটি আদর্শ ভাষাতে পরিণত করেছে। সি ভাষার সাহায্যে যে সকল প্রোগ্রাম তৈরি করা যায় তার কয়েকটির নাম নিম্নে দেওয়া হলো :

১. অপারেটিং সিস্টেম (Operating System)
২. অ্যাসেম্বলার (Assembler)
৩. কম্পাইলার (Compiler)
৪. ইন্টারপ্রেটার (Interpreter)
৫. ডেটাবেজ ম্যানেজমেন্ট সিস্টেম (Database Management System)
৬. এডিটর (Editor)
৭. ভাইরাস এবং এন্টিভাইরাস (Virus and Antivirus) ইত্যাদি।

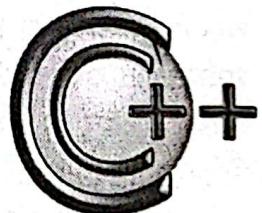
First Programming Languages
1945 - Machine Language
1950 - EDSAC Assembly Language
1951 - Regional Assembly Language
1952 - Autocode
1954 - IPL (Fortran to LISP)
1955 - Flow Matic (Led to COBOL)
1957 - FORTRAN (First compiler)
1958 - LISP, ALGOL 58
1959 - COBOL, RPG
1962 - APL, Simula
1964 - BASIC, PL/I
1967 - BCPL (forerunner to C)
Establishing Fundamental Paradigms
1968 - Logo
1970 - Pascal, Forth
1972 - C, Smalltalk, Prolog
1973 - ML
1975 - Scheme
1978 - SQL (A query language)
1980s: Consolidation, Modules, Performance
1980 - C++
1983 - Ada
1984 - Common Lisp, MATLAB
1985 - Eiffel
1986 - Erlang Objective C
1987 - Perl
1988 - Mathematica, TCL
1990s: The Internet Age :
1990 - Haskell
1991 - Python, Visual Basic
1993 - Ruby
1995 - Java, Delphi, JavaScript, PHP
1996 - WebDNA
1997 - Rebol
1999 - D
Current trends
2000 - ActionScript
2001 - C#, Visual Basic .NET
2003 - Groovy, Scala
2005 - F#
2006 - Windows Powershell
2007 - Clojure
2009 - Go
2011 - Dart
2012 - Rust
2014 - Swift
2015 - Rust

প্রোগ্রামের ভাষার পর্যায়ক্রমিক উভাবন

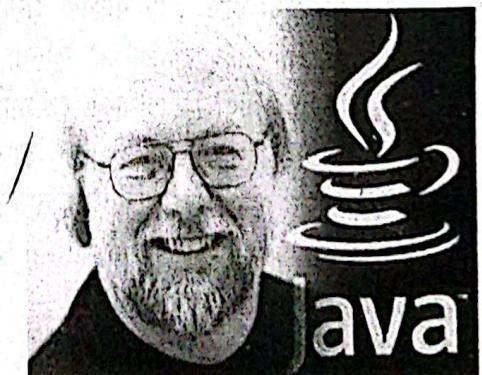
সি ভাষাই একমাত্র ভাষা যা মেশিন ভাষা ও উচ্চ স্তরের ভাষার সেতুবন্ধন রচনা করতে সক্ষম। সি ভাষার নমনীয়তা এবং ব্যাপক জনপ্রিয়তার কারণে অনেক প্রতিষ্ঠান সি ভাষাকে পরিবর্তিত ও নানা বৈশিষ্ট্য সমন্বিত করে এর নাম ভার্সন বাজারে আনে। এজন্য ব্যবহারিক সমস্যা সৃষ্টি হয়। এ সমস্যা সমাধানে ১৯৮৩ সালে ANSI (American National Standard Institute) নামক প্রতিষ্ঠান সি ভাষার একটি স্ট্যান্ডার্ড নির্ধারণ করেন, যা ANSI C নামে পরিচিত। ANSI C ছাড়াও সি এর কয়েকটি ভার্সন হচ্ছে- Turbo C, Quick C, C++, Borland C++ ইত্যাদি।

কম্পিউটারে ব্যবহৃত সিস্টেম সফটওয়্যার তৈরিতে ব্যবহৃত ভাষা হলো সিস্টেম প্রোগ্রামিং ভাষা। অ্যাপ্লিকেশন প্রোগ্রামিংয়ের কাজ হলো ব্যবহারকারীর সরাসরি ব্যবহারের জন্য প্রোগ্রাম তৈরি করা, যেখানে সিস্টেম প্রোগ্রামিংয়ের মাধ্যমে সে সকল সফটওয়্যার তৈরি করা হয় যা অ্যাপ্লিকেশন সফটওয়্যারে বিভিন্ন ধরনের সার্ভিস প্রদান করে এবং যত্রাংশসমূহ নিয়ন্ত্রণে একটি কমন কম্পিউটেশনাল প্লাটফর্ম তৈরি করে। সিস্টেম প্রোগ্রামিংয়ে হার্ডওয়্যার সচেতনতা এবং হার্ডওয়্যার পরিচালনাগত দক্ষতা অন্তর্ভুক্ত। এর মূল উদ্দেশ্য হলো হার্ডওয়্যার রিসোর্সগুলো দক্ষ ব্যবহার যাতে ব্যবহারকারীর প্রোগ্রাম পরিচালনার কাজ সহজ হয়।

**৫.৬.২ সি++(C++) :** সি++ একটি বহুল ব্যবহৃত অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং ভাষা। ১৯৮০ সালে যুক্তরাষ্ট্রে এটি এভ টি বেল ল্যাবরেটরিতে জর্ন স্ট্রাউস্ট্রপ (Bjarne Stroustrup) এ ভাষা উভাবন করেন। প্রথমে এর নাম ছিল 'সি উইথ ক্লাস'। পরবর্তী সময়ে আরও কিছু নতুন বৈশিষ্ট্য যোগ করে ১৯৮৩ সালে সি++ নামকরণ করা হয়। সি++ এ সি এর প্রায় সব বৈশিষ্ট্যসহ অতিরিক্ত আরও কিছু বৈশিষ্ট্য ও সুবিধা আছে। এজন্য সি++ কে সি-এর বর্ধিত সংস্করণ বা সুপারসেট বলা হয়। টেক্সট এডিটর তৈরি, কম্পাইলার ও ইন্টারপ্রেটার তৈরি, ডেটাবেজ হ্যান্ডেলিং, কমিউনিকেশন সিস্টেম ডিজাইন, ডিস্ট্রিবিউটেড সিস্টেম ডিজাইন, রিয়েল-টাইম সিস্টেম ডিজাইন ও উইঙ্গেজ ভিত্তিক এপ্লিকেশনসমূহ সি++ এর অনন্য অবদান।



**৫.৬.৩ জাভা (Java) :** বর্তমান সময়ে ব্যবহৃত ভাষাগুলোর মধ্যে জাভা একটি অত্যন্ত শক্তিশালী ও জনপ্রিয় প্রোগ্রামিং ভাষা। ১৯৯১ সালের শেষের দিকে জেমস গোসলিং (James Gasling)-এর নেতৃত্বে একদল কম্পিউটার বিজ্ঞানী জাভা ভাষা উভাবন করেন। সি++ ভাষায় নানাবিধ ফিচার যুক্ত করে জাভা তৈরি করা হয়েছে। জাভা ভাষা ব্যবহার করে থ্রুর ইন্টারনেট নির্ভর সফটওয়্যার তৈরি করা হচ্ছে। এজন্য জাভাকে সি++ এর ইন্টারনেট সংস্করণও বলা হয়।



জাভা ভাষার উভাবক জেমস গোসলিং

সি++ ভাষা হতে জাভার উত্তর হলেও এটি সি++ এর তুলনায় সহজ, নিরাপদ এবং প্লাটফর্ম স্বনির্ভর। জাভা প্লাটফর্ম স্বনির্ভর ভাষা বলে জাভায় তৈরী প্রোগ্রাম যে কোনো অপারেটিং সিস্টেমে চালানো যায়। এজন্য অপারেটিং সিস্টেমে কেবল মাত্র জাভা ভার্চুয়াল মেশিন (রান টাইম এনভাইরনমেন্ট) যুক্ত থাকতে হয়। অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং জাভার একটি গুরুত্বপূর্ণ দিক। অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং সুবিধার কারণে জাভায় অতি দীর্ঘ প্রোগ্রাম লেখা যায় এবং অনেক সহজে ক্রিটি মুক্ত করা যায়।

৭. চতুর্থ প্রজন্মের ভাষা মুক্ত প্রকৃতির ভাষা। তাই এক কম্পিউটারের জন্য তৈরি প্রোগ্রাম কোনোক্রম পরিবর্তন ছাড়াই অন্য কম্পিউটারে ব্যবহার করা যায়।
৮. এ সকল ভাষা ইন্টার্যাক্টিভ মোডে কাজ করতে সক্ষম। তাই কোনো একটি কম্বল লেখার সাথে সাথে তা কম্পাইল হয়ে যায়, পরবর্তীতে আর কম্পাইল করতে হয় না।
৯. চতুর্থ প্রজন্মের ভাষাকে স্বাভাবিক ভাষা বা নন-গ্রামার্থিক ভাষা বলা হয়।
১০. উচ্চস্তরের ভাষার মত উন্নত ও শক্তিশালী সফটওয়্যার এসকল ভাষা ব্যবহার করে তৈরি করা যায় না।
১১. এ ভাষা শিল্পমান সমর্থিত ভাষা নয়।
১২. এ ভাষা গুণগতভাবে উচ্চস্তরের ভাষার মত উন্নত নয়।

**কাজ**

চতুর্থ প্রজন্মের ভাষার বিভিন্নক্ষেত্রে প্রয়োগের তালিকা প্রস্তুত কর।

**পার্থক্য-** লো লেভেল ভাষা ও হাই লেভেল ভাষা : লো লেভেল ভাষা একটি মেশিন নির্ভর ভাষা। পক্ষান্তরে হাই লেভেল ভাষা মেশিন নির্ভর নয়। নিচে এদের মধ্যকার পার্থক্য তুলে ধরা হলো:

লো লেভেল ভাষা	হাই লেভেল ভাষা
১. লো লেভেল ভাষায় <u>প্রোগ্রাম লেখা</u> কষ্টকর ও সময় সাপেক্ষ।	১. হাই লেভেল ভাষায় <u>প্রোগ্রাম লেখা</u> অপেক্ষাকৃত সহজসাধ্য।
২. এ ভাষার <u>প্রোগ্রাম চালানোর জন্য</u> কম মেমরির প্রয়োজন হয়।	২. এ ভাষার <u>প্রোগ্রাম চালানোর জন্য</u> বেশি মেমরির প্রয়োজন হয়।
৩. এ ভাষার প্রোগ্রাম দ্রুত নির্বাহ হয়।	৩. এ ভাষার প্রোগ্রাম অপেক্ষাকৃত ধীর গতিতে নির্বাহ হয়।
৪. বৈদ্যুতিক বর্তনীর ভুল-ক্রটি নির্ণয়ে এ ভাষা ব্যবহার করা হয়।	৪. বিভিন্ন প্রকারের ব্যবহারিক প্রোগ্রাম তৈরিতে এ ভাষা ব্যবহার করা হয়।
৫. লো লেভেল ভাষায় লেখা <u>প্রোগ্রাম পরিবর্তন</u> করা কষ্টসাধ্য।	৫. হাই লেভেল ভাষায় লেখা <u>প্রোগ্রাম সহজেই</u> পরিবর্তন করা যায়।
৬. কম্পিউটারের <u>অভ্যন্তরীণ গঠন জানা</u> না থাকলে প্রোগ্রাম রচনা করা যায় না।	৬. কম্পিউটারের <u>অভ্যন্তরীণ গঠন জানার প্রয়োজন</u> হয় না।
৭. এ ভাষা <u>মেশিন নির্ভর</u> ভাষা। কোনো কম্পিউটারের জন্য লেখা প্রোগ্রাম শুধুমাত্র ঐ কম্পিউটারে চলবে, অন্য কম্পিউটারে চলবে না।	৭. এ ভাষা <u>মেশিন নির্ভর</u> নয়। কোনো কম্পিউটারের জন্য লেখা প্রোগ্রাম অন্য যে কোনো কম্পিউটারে চলবে।
৮. লো লেভেল ভাষায় প্রোগ্রাম লিখতে দক্ষ প্রোগ্রামার দরকার হয়।	৮. হাই লেভেল ভাষায় যে কোনো ইংরেজি জানা লোক প্রোগ্রাম লিখতে পারে।

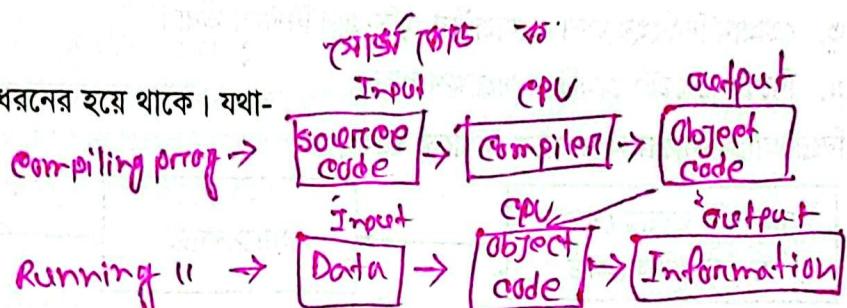
### ৫.৮ অনুবাদক প্রোগ্রাম (Translator Program)

যে সকল প্রোগ্রাম সোর্স প্রোগ্রামকে অবজেক্ট প্রোগ্রামে রূপান্তরিত করে, তাকে অনুবাদক প্রোগ্রাম বলা হয়।  
মেশিন ভাষার প্রোগ্রামকে বলা হয় অবজেক্ট প্রোগ্রাম। মেশিন ভাষা ব্যতিত অন্য ভাষার প্রোগ্রামকে বলা হয় সোর্স প্রোগ্রাম। তাই বলা যায়- যে সকল প্রোগ্রাম মেশিন ভাষা ব্যতিত অন্য (অ্যাসেম্বলি ও উচ্চতর) ভাষার প্রোগ্রামকে মেশিন ভাষায় রূপান্তরিত করে, তাকে অনুবাদক প্রোগ্রাম বলা হয়।

অনুবাদক প্রোগ্রামের প্রকারভেদ

অনুবাদক প্রোগ্রাম সাধারণত তিনি ধরনের হয়ে থাকে। যথা-

১. অ্যাসেম্বলির
২. কম্পাইলার
৩. ইন্টারপ্রেটার



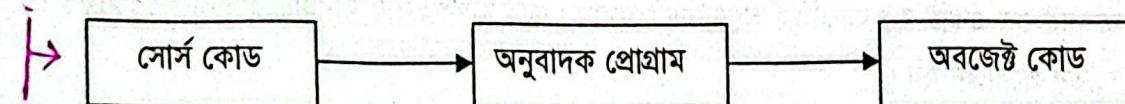
অনেক অনুবাদক প্রোগ্রাম রয়েছে যা এক ধরনের উচ্চতরের ভাষার প্রোগ্রামকে অন্য একটি উচ্চতরের ভাষায় অনুবাদ করতে পারে। এ ধরনের অনুবাদক প্রোগ্রামকে Language Converter বলা হয়। এ ধরনের কয়েকটি অনুবাদক প্রোগ্রামের নাম হলো:

- ক) FORTRAN-to-Ada Translator
- খ) Pascal-to-C Translator
- গ) CHILL-to-C++ Translator ইত্যাদি।

**অনুবাদক প্রোগ্রামের প্রয়োজনীয়তা :** কম্পিউটারের নিজস্ব ভাষা হলো মেশিন ভাষা। এ ভাষার প্রোগ্রাম কম্পিউটার সরাসরি বুঝতে পারে। অ্যাসেম্বলি ও উচ্চতর ভাষার প্রোগ্রাম কম্পিউটার সরাসরি বুঝতে পারে না। তাই অনুবাদক প্রোগ্রাম ব্যবহার করে অ্যাসেম্বলি ও উচ্চতর ভাষার প্রোগ্রামকে মেশিন ভাষায় রূপান্তরিত করা হয়।

নিম্নের ছকে অনুবাদক প্রোগ্রামের কাজ দেখানো হলো-

**অনুবাদক প্রোগ্রাম**



**অনুবাদক প্রোগ্রামের কার্যবলী**

১. সোর্স প্রোগ্রামকে অবজেক্ট প্রোগ্রামে রূপান্তরিত করা।
২. প্রোগ্রাম নির্বাহের জন্য প্রয়োজনীয় সূত্রিত পরিমাণ নির্ধারণ করা।
৩. প্রোগ্রামের ভূল-ক্রটি চিহ্নিত করা ও ভূল-ক্রটি সংশোধনে সহায়তা করা।
৪. ফাংশন বা সাবরুটিনের সাথে প্রোগ্রামের সংযোগ রক্ষা করা।
৫. প্রয়োজনে সোর্স প্রোগ্রামকে প্রিন্ট করা।

**মেশিন হার্ডওয়ার**

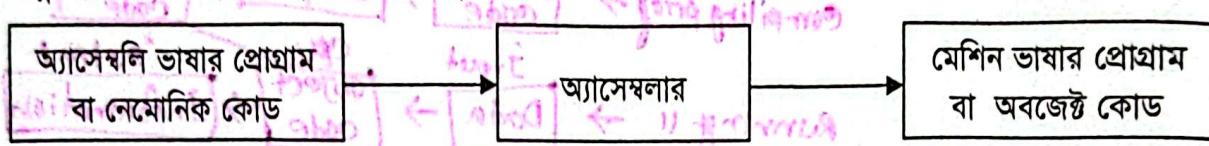
**স্মা�্ট টেক**

### ৫.৮.১ অ্যাসেম্বলার (Assembler)

অ্যাসেম্বলি ভাষার প্রোগ্রামকে অনুবাদের জন্য যে অনুবাদক প্রোগ্রাম ব্যবহৃত তাকে অ্যাসেম্বলার বলা হয়। অ্যাসেম্বলি ভাষার প্রোগ্রামকে মেশিনে রূপান্তরিত করে। অ্যাসেম্বলারের প্রধান প্রধান কাজ হলো-

১. অ্যাসেম্বলি ভাষার প্রোগ্রামকে তথা নেমোনিক কোডকে অবজেক্ট প্রোগ্রামে রূপান্তরিত করা।
২. প্রোগ্রামের নির্দেশ পরিক্ষা করা এবং ভুল সনাক্ত করা।
৩. প্রোগ্রাম নির্বাহের জন্য প্রয়োজনীয় স্থৃতি স্থান নির্ধারণ করা।
৪. নির্দেশ ও ডেটা মেমরিতে সংরক্ষণ করা।

নিম্নে অ্যাসেম্বলারের কাজ ছকে দেখানো হলো:



### ৫.৮.২ কম্পাইলার (Compiler)

কম্পাইলার একটি বহুল ব্যবহৃত অনুবাদক প্রোগ্রাম। বর্তমান সময়ে প্রায় সকল উচ্চতর প্রোগ্রামের ভাষায় কম্পাইলার ব্যবহৃত হয়। কম্পাইলার উচ্চতর ভাষার প্রোগ্রামকে মেশিন ভাষায় রূপান্তরিত করে। নিম্নে এর বৈশিষ্ট্যগুলো তুলে ধরা হলো-

#### কম্পাইলারের বৈশিষ্ট্য

১. কম্পাইলার সম্পূর্ণ প্রোগ্রামকে একসাথে অনুবাদ করে।
২. অনুবাদকৃত প্রোগ্রাম একসাথে মেমরিতে সংরক্ষণ করে।
৩. কম্পাইলারে প্রোগ্রাম নির্বাহের জন্য ইন্টারপ্রেটার অপেক্ষা কম সময় লাগে।
৪. কম্পাইলার প্রোগ্রামের সকল ভুলের তালিকা এক সাথে প্রকাশ করে।
৫. কম্পাইলারের সাহায্যে ডিবাগিং প্রক্রিয়া অপেক্ষাকৃত জটিল।
৬. কম্পাইলারের সাহায্যে ডিবাগিংয়ে বেশি সময়ের প্রয়োজন হয়।
৭. কম্পাইলার একটি বড় আকৃতির অনুবাদক প্রোগ্রাম। তাই কম্পাইলার ব্যবহারের জন্য অপেক্ষাকৃত বেশি স্থৃতির প্রয়োজন হয়।
৮. কম্পাইলার সহায়ক স্থৃতিতে অবজেক্ট ফাইল তৈরি করে।
৯. একবার অনুবাদকৃত প্রোগ্রাম পরবর্তীতে অনুবাদ ছাড়ায় ব্যবহার করা যায়।
১০. অনুবাদকৃত প্রোগ্রাম পূর্ণাঙ্গ মেশিন ভাষায় পরিণত হয়।

### ৫.৮.৩ ইন্টারপ্রেটার (Interpreter)

কম্পাইলারের মতো ইন্টারপ্রেটারও উচ্চতর প্রোগ্রামের ভাষায় ব্যবহৃত একটি অনুবাদক প্রোগ্রাম। বর্তমান সময়ে খুব কম সংখ্যক উচ্চতর প্রোগ্রামের ভাষায় ইন্টারপ্রেটার ব্যবহৃত হয়। ইন্টারপ্রেটার উচ্চতর ভাষার প্রোগ্রামকে মেশিন ভাষায় রূপান্তরিত করে। তবে এর অনুবাদ প্রক্রিয়া কম্পাইলার হতে ভিন্ন ধরনের।

### ইন্টারপ্রেটারের বৈশিষ্ট্য

১. ইন্টারপ্রেটার সম্পূর্ণ প্রোগ্রাম একসাথে অনুবাদ না করে এক লাইন করে অনুবাদ করে।
২. অনুবাদকৃত প্রোগ্রামের প্রতি লাইন আলাদাভাবে মেমরিতে সংরক্ষণ করে।
৩. প্রোগ্রাম নির্বাহের জন্য কম্পাইলার অপেক্ষা বেশি সময়ের প্রয়োজন হয়।
৪. একটি ভুল ধরা পড়লে প্রোগ্রামের নির্বাহ থেমে যায়।
৫. ইন্টারপ্রেটারের সাহায্যে ডিবাগিং প্রক্রিয়া অপেক্ষাকৃত সহজ।
৬. ডিবাগিংয়ে অপেক্ষাকৃত কম সময়ের প্রয়োজন হয়।
৭. ইন্টারপ্রেটার অপেক্ষাকৃত কম স্মৃতির কম্পিউটারে ব্যবহারযোগ্য।
৮. ইন্টারপ্রেটার সহায়ক স্মৃতিতে অবজেক্ট ফাইল তৈরি করে না।
৯. প্রোগ্রাম নির্বাহের সময় প্রতিবারই অনুবাদ করতে হয়।
১০. অনুবাদকৃত প্রোগ্রাম পূর্ণাঙ্গ মেশিন ভাষায় পরিণত হয় না।

### কম্পাইলার ও ইন্টারপ্রেটারের পার্থক্য(Difference between Compiler and Interpreter)

কম্পাইলার ও ইন্টারপ্রেটার উভয়ই উচ্চতরের ভাষায় ব্যবহৃত অনুবাদক প্রোগ্রাম। তবে অনুবাদ প্রক্রিয়ার ক্ষেত্রে উভয়ের মধ্যে যথেষ্ট পার্থক্য বিদ্যমান। নিচে এদের মধ্যকার পার্থক্য তুলে ধরা হলো :

কম্পাইলার	ইন্টারপ্রেটার
১. কম্পাইলার সম্পূর্ণ প্রোগ্রামকে একসাথে অনুবাদ করে।	১. ইন্টারপ্রেটার সম্পূর্ণ প্রোগ্রামকে একসাথে অনুবাদ করে না, একটি করে নির্দেশ অনুবাদ করে।
২. অনুবাদকৃত প্রোগ্রাম একসাথে মেমরিতে সংরক্ষিত হয়।	২. অনুবাদকৃত প্রোগ্রামের প্রতি লাইন আলাদাভাবে মেমরিতে সংরক্ষিত হয়।
৩. প্রোগ্রাম নির্বাহের জন্য ইন্টারপ্রেটার অপেক্ষা কম সময়ের প্রয়োজন হয়।	৩. প্রোগ্রাম নির্বাহের জন্য কম্পাইলার অপেক্ষা বেশি সময়ের প্রয়োজন হয়।
৪. কম্পাইলার সকল ভুলের তালিকা একত্রে প্রকাশ করে।	৪. একটি ভুল ধরা পড়লে প্রোগ্রামের নির্বাহ থেমে যায়।
৫. কম্পাইলারের সাহায্যে ডিবাগিংয়ে বেশি সময়ের প্রয়োজন হয়।	৫. ডিবাগিংয়ে অপেক্ষাকৃত কম সময়ের প্রয়োজন হয়।
৬. কম্পাইলার ব্যবহারের জন্য ইন্টারপ্রেটার অপেক্ষা বেশি স্মৃতির প্রয়োজন হয়।	৬. ইন্টারপ্রেটার অপেক্ষাকৃত কম স্মৃতির কম্পিউটারে ব্যবহার যোগ্য।
৭. কম্পাইলার সহায়ক স্মৃতিতে অবজেক্ট ফাইল তৈরি করে।	৭. ইন্টারপ্রেটার সহায়ক স্মৃতিতে অবজেক্ট ফাইল তৈরি করে না।
৮. একবার অনুবাদকৃত প্রোগ্রাম পরবর্তীতে অনুবাদ ছাড়া ব্যবহার করা যায়।	৮. প্রতিবার প্রোগ্রাম নির্বাহের সময় অনুবাদ করতে হয়।
৯. অনুবাদকৃত প্রোগ্রাম পূর্ণাঙ্গ মেশিন ভাষায় পরিণত হয়।	৯. অনুবাদকৃত প্রোগ্রাম পূর্ণাঙ্গ মেশিন ভাষায় পরিণত হয় না।
১০. কম্পাইলার ব্যবহার করা হয় এমন কয়েকটি প্রোগ্রামের ভাষা-সি, প্যাস্কেল, ভিজুয়াল বেসিক ইত্যাদি।	১০. ইন্টারপ্রেটার ব্যবহার করা হয় এমন কয়েকটি প্রোগ্রামের ভাষা-বেসিক, কিউবেসিক ইত্যাদি।

কাজ

কম্পাইলার ও ইন্টারপ্রেটারের মধ্যে কোনটি সুবিধাজনক বিশ্লেষণ কর।

শ্রেণীয়ের ভূলসমূহের বিবরণ।

শ্রেণীয়ে সাধারণত তিনি ধরনের ভূল থাকে।

- সিনটাক্স এরর(Syntax Error) বা টিক্সের ভূল
- লজিকাল এরর(Logical Error) বা যৌক্তিক ভূল
- এক্সিকিউশন এরর(Execution Error) বা নির্বাচিত ভূল

নিম্ন এদের সংক্ষিপ্ত বিবরণ দেওয়া হলো-

ক) **সিনটাক্স এরর(Syntax Error)** বা টিক্সের ভূল: প্রোগ্রাম লেখার সময় একটি নির্দিষ্ট প্রোগ্রামের ভাষা অনুসরণ করতে হয়। অনুসৃত ভাষায় ব্যবহৃত কীওয়াডের বানানগত ভূল কিংবা সঠিক চিহ্ন প্রয়োগে ভূল হলে এক্ষণ্ট ভূলকে সিনটাক্স এরর(Syntax Error) বা টিক্সের ভূল বলা হয়। যেমন- সি ভাষায় printf() এর স্থলে PRINTF( ) বা print লিখলে সিনট্যাক্স এরর হয়। আবার সি ভাষা প্রতিটি লাইনের শেষে একটি করে সোনিকোলন চিহ্ন (;) দিতে হয়। কোনো লাইনের শেষে সোনিকোলন চিহ্ন না দিলেও সিনট্যাক্স এরর হবে। সিনট্যাক্স এরর প্রোগ্রাম অনুবাদের সময় ধৰা পরে এবং অনুবাদক প্রোগ্রাম এ সংক্রান্ত Error Message প্রদান করে। সিনট্যাক্স এরর সংশোধন করা না হলে প্রোগ্রাম রান করে না।

খ) **লজিকাল এরর(Logical Error)** বা যৌক্তিক ভূল: কোনো গাণিতিক বা লজিকাল অপারেটরের ব্যবহার ব্যথাবথ না হলে কিংবা স্থূল প্রয়োগ সঠিক না হলে এক্ষণ্ট ভূলকে লজিকাল এরর(Logical Error) বা যৌক্তিক ভূল বলা হয়। যেমন-  $x>y$  এর স্থলে  $x< y$  লিখলে লজিকাল এরর হবে। লজিকাল এরর অনুবাদক প্রোগ্রাম কঙ্ক ধরা পরে না এবং কোনো এ সংক্রান্ত Error Message প্রদান করে না। লজিকাল এরর থাকলেও প্রোগ্রাম রান হবে। তবে প্রোগ্রাম হতে কাঁচিত ফলাফল পাওয়া যাবে না। অর্থাৎ ভূল ফলাফল পাওয়া যাবে।

গ) **এক্সিকিউশন এরর(Execution Error)** বা নির্বাচিত ভূল: প্রোগ্রাম নির্বাচের সময় কোনো অতিহচ্ছেয়োগ্য ডেটা ইনপুট দিলে কিংবা কোনো চলকের অতিহচ্ছেয়োগ্য মান হলে এক্ষণ্ট ভূলকে এক্সিকিউশন এরর(Execution Error) বা নির্বাচিত ভূল বলা হয়। এ ধরনের ভূলে প্রোগ্রাম নির্বাচের সময় হঠাত করে প্রোগ্রামের নির্বাচ বন্ধ হয়ে যায়। যেমন - ভাগ প্রক্রিয়ায় ব্যবহৃত চলকের মান ছোলা হলে প্রোগ্রামের নির্বাচ বন্ধ হবে এবং এর ফ্যাসেজ দেখাবে।

**ক্ষাজ**  **নির্ধারণ কর।**

একটি আদর্শ প্রোগ্রামের গুণাবলী

একটি আদর্শ প্রোগ্রামের যে সকল গুণাবলী থাকা আবশ্যিক, সেগুলো হলো

- প্রোগ্রামের শুরুতে এর উদ্দেশ্য, প্রস্তুত, চলক ইত্যাদির পরিচয় সন্তুষ্টিত করতে হবে, যাতে কোনো ব্যবহারকারী প্রোগ্রামের কার্যবলী সহজেই বুঝতে পারে।
- প্রোগ্রামে চলক হিসেবে প্রতিনিধিত্বকৰণ করতে হবে, যাতে চলকের ব্যবহার সম্পর্কে ধারণা করা যায়।
- প্রোগ্রামের অ্যালগরিদম, ফোর্মাট ও সূত্রকোড সরলভাবে প্রাপ্ত করাতে হবে, যাতে প্রোগ্রামের ধাপগুলো সহজে বোঝা যায়।
- প্রোগ্রামকে অকারণে দীর্ঘ না করা।
- প্রোগ্রামে অতিরিক্ত চক্রবর্ত বা লুপ ব্যবহার না করা।
- কোন নির্দিষ্ট কাজের জন্য উপযুক্ত প্রোগ্রামের ভাষা নির্বাচন করা।

### ৫.১০.১ অ্যালগরিদম (Algorithm)

আমরা আমাদের প্রাত্যহিক জীবনে কোনো কাজ করার সময় কাজটি একসাথে শেষ করতে পারি না। কাজটি ধাপে ধাপে পর্যায়ক্রমে সম্পাদন করতে হয়। উদাহরণস্বরূপ বলা যায়, একজন চা বিক্রেতা যখন চা প্রস্তুত করেন তখন তাকে কাজটি ধাপে ধাপে পর্যায়ক্রমিকভাবে শেষ করতে হয়। যেমন- প্রথমে তাকে চা তৈরির উপকরণ সংগ্ৰহ কৰতে হয়, কেটেলিতে পরিমাণ মতো চা পানি দিতে হয়, চূলা জুলাতে হয়, পানি ঝুটলো কিনা দেখতে হবে। পানি ঝুটলো তাতে পরিমাণ মতো চা পাতা, চিনি, দুধ ইত্যাদি উপকরণ মিশাতে হয় এবং পরিশেষে ছাকনি দিয়ে চায়ের কাপে চা ছেকে নিতে হয়। এসব ধাপ সার্টিকভাবে অনুসরণের মাধ্যমে চা প্রস্তুত সম্পন্ন হয়। একেবারে চা বিক্রেতা তার ইচ্ছে মতো কাজ করলে তার কাঞ্চিত আউটপুট (চা) প্রস্তুত করতে সক্ষম হবেন না। প্রোগ্রামিংয়ের অনুসরণের মাধ্যমেই কেবলমাত্র একজন প্রোগ্রামার তার পরিকল্পিত প্রোগ্রামটির সফল বাস্তবায়ন করতে পারেন।

অ্যালগরিদম শাখের অর্থ হলো ধাপে ধাপে সমস্যার সমাধান করা। একটি সমস্যা সমাধানের জন্য সমস্যাটিকে কর্যকর্তৃ ধাপে ধাপে প্রতেকটি ধাপ পর পর সমাধান করে সম্পূর্ণ সমস্যা সমাধান করার যে প্রক্রিয়া তার লিখিত বর্ণনাকে অ্যালগরিদম বলা হয়। এক কথায়, অ্যালগরিদম হলে কোন প্রোগ্রামিং সমস্যা সমাধানের যৌক্তিক ধারাবাহিক বিবরণ।

### ৫.১০.২ সূত্রে কোড (Psudo code)

সূত্রে একটি গ্ৰীক শব্দ। সূত্রে শব্দের অর্থ হচ্ছে ছুঁ বা সত্ত নয়। প্রোগ্রাম বনানৰ পূৰ্বে অনেকেই সূত্রে কোড প্ৰয়োজন কৰে থাকেন। সূত্রে কোডকে অ্যালগরিদম ও ফ্ৰোচাৰ্টেৰ বিকল্পও বলা যায়। সূত্রে কোড হলো কোনো প্ৰোগ্রামেৰ ধৰণ ও কাৰ্যাৰ্বলী সংবলিত বৰ্ণনা। ইংৰেজি ভাষায় কোনো প্ৰোগ্রামেৰ ধাপগুলোৱাৰ বৰ্ণনা কৰা হলে তাৰেক সূত্রে কোড বলা হয়।

প্ৰোগ্রাম লিখতে একজন প্ৰোগ্রামারকে একটি প্ৰোগ্রামেৰ তাৰা অনুসৰণ কৰতে হয়। অনুসূত তাৰাৰ নিয়ম মেলে নিৰ্দেশ সাজাতে হয়। বৰ্তমানে প্ৰায় ৪৫০টি প্ৰোগ্রামেৰ তাৰা রয়েছে। প্ৰতিটি তাৰাৰ নিজস্ব নিয়ম-কাৰ্যন, গঠনৰীতি ও শব্দ সংজ্ঞাৰ রয়েছে। একজন প্ৰোগ্রামারকে এ সকল নিয়ম কাৰ্যন মেলে প্ৰোগ্রাম লিখতে হয়। কাৰ্যামো ও ধাৰণাগত মিল থাকলেও প্ৰতিটি প্ৰোগ্রামেৰ তাৰাৰ গঠনগত অনেক অলিল রয়েছে। তাই একজন প্ৰোগ্রামারেৰ পক্ষে সব কয়টি প্ৰোগ্রামেৰ তাৰা শেখা সজ্জৰ নয়। এজন্য কোনো একটি প্ৰোগ্ৰামিং সমস্যা সমাধানেৰ উপায় নিৰ্দিষ্ট কোনো ভাষায় বিবৃত না কৰে সাধাৰণভাৱে সব ভাষার উপযোগী কৰে উপস্থাপন কৰা হয়। এতে কৰে একজন প্ৰোগ্রামার তাৰ জোন ভাষা প্ৰযোগ কৰে নিজেৰ প্ৰযোজনীয় প্ৰোগ্ৰামটি লিখতে পাৰেন। প্ৰোগ্রামেৰ তাৰ নিৰপেক্ষ এৱং প্ৰিমে বিবৰণই সূত্রে কোড।

**উদাহৰণ :** নিম্নে দুটি সংখ্যাৰ যোগফল ও গড় নিৰ্ণয়ৰ সূত্রে কোড দেখাবো হলো-

```
Input Number1, Number2
Sum=Number1+Number2
Average=Sum/2
Print Sum, Average
End
```

### ৫.১০.৩ ফ্লোচার্ট (Flow Chart)

ফ্লোচার্ট হলো অ্যালগরিদমের চিত্রভিত্তিক উপস্থাপনা। কোনো প্রোগ্রামের কার্যাবলী যে চিত্রের মাধ্যমে তুলে হয় তাকে ফ্লোচার্ট বলে। ফ্লোচার্ট হতে সমস্যা সমাধানের যৌক্তিক ধারাবাহিক ধাপগুলো সম্পর্কে সম্যক ধ্রুব করা যায়। অর্থাৎ কোন একটি প্রোগ্রাম কিভাবে কাজ করবে তা চিত্রের মাধ্যমে প্রকাশ করা হলে এটি হবে ফ্লোচার্ট। ফ্লোচার্টে কতকগুলি নির্দিষ্ট চিআংশ ব্যবহার করা হয়। প্রতিটি চিআংশ এক একটি ভিন্ন ধরণের ব্যবহার করা হয়। প্রোগ্রামের কাজ কিভাবে অগ্রসর হবে তা ফ্লোচার্ট হতে বোঝা যায়। ফ্লোচার্ট দুই প্রকার। যথা - প্রোগ্রাম ফ্লোচার্ট এবং সিস্টেম ফ্লোচার্ট।

**প্রোগ্রাম ফ্লোচার্ট :** যে ফ্লোচার্টে প্রোগ্রামের কাজ কিভাবে ধাপে ধাপে অগ্রসর হয় তা দেখানো হয় তাকে প্রোগ্রাম ফ্লোচার্ট বলে।

**সিস্টেম ফ্লোচার্ট :** যে ফ্লোচার্টে সিস্টেমের বিভিন্ন উপাদান ও এর পরিবর্তনগুলো তুলে ধরা হয় তাকে সিস্টেম ফ্লোচার্ট বলে।

#### ফ্লোচার্ট অংকনে বিবেচ্য বিষয়/ফ্লোচার্ট অংকনের নিয়মাবলী

ফ্লোচার্ট হলো কোনো প্রোগ্রামের মূল ভিত্তি। ফ্লোচার্ট অনুসরণ করে প্রোগ্রাম রচনা করা হয়। নিম্নে ফ্লোচার্ট অংকনে বিবেচ্য বিষয়গুলি তুলে ধরা হলো-

১. ফ্লোচার্ট অংকনে ব্যবহৃত চিহ্নগুলো পরপর হতে নিচে কিংবা ডান থেকে বাঁয়ে আঁকতে হবে।
২. প্রবাহরেখা দিয়ে চিহ্নগুলো ক্রমানুসারে সংযুক্ত করতে হবে।
৩. একাধিক প্রবাহরেখা দিয়ে যাতে পরস্পরকে ছেদ না করে সেদিকে নজর দেওয়া উচিত।
৪. ফ্লোচার্ট অংকনে ব্যবহৃত চিহ্নগুলো যথাযথ হতে হবে। অর্থাৎ চিহ্নগুলোর নির্দিষ্ট আকৃতি যেন ঠিক থাকে।
৫. ফ্লোচার্টের একটি নির্দিষ্ট নাম থাকবে।
৬. ফ্লোচার্ট অংকনে ব্যবহৃত একাধিক প্রবাহ রেখা পরস্পর একই সাথে মিলিত হলে সংযোগ প্রতীক (Connector) ব্যবহার করা উচিত।
৭. ফ্লোচার্ট কোন বিশেষ ভাষার জন্য অংকন করা উচিত নয়। অর্থাৎ ফ্লোচার্ট এমনভাবে আঁকতে হবে যাতে এটি অনুসরণ করে যে কোনো প্রোগ্রামিং ভাষায় প্রোগ্রাম রচনা করা যায়।
৮. ফ্লোচার্ট সংক্ষিপ্ত ও সহজবোধ্য হওয়া উচিত।
৯. ফ্লোচার্টের কোনো অংশের বিস্তারিত বর্ণনার জন্য সে অংশ পৃথক করে বিস্তারিত ভাবে লেখা উচিত।

কাজ

- ক)অ্যালগরিদম, ফ্লোচার্ট ও সূত্রকোড-এর সাদৃশ্য-বৈসাদৃশ্যের তুলনামূলক ছক প্রস্তুত কর
- খ) অ্যালগরিদম ও ফ্লোচার্ট প্রণয়নের প্রয়োজনীয়তা আলোচনা কর।

## ফ্লোচার্ট অংকনে ব্যবহৃত প্রতীকসমূহ এবং এদের ব্যবহার:

চিহ্নের নাম	প্রতীক	ব্যবহার
প্রাতিক চিহ্ন (Terminator)		এটি একটি উপবৃত্তাকার চিহ্ন। ফ্লোচার্টের শুরুতে ও শেষে এ চিহ্নটি একটি করে ব্যবহার করা হয়।
ইনপুট/আউটপুট চিহ্ন (I/O or Data)		এটি একটি সামান্যরিক আকৃতির চিহ্ন। প্রোগ্রাম ডেটা ইনপুট এবং প্রোগ্রাম হতে ফলাফল প্রদর্শনের ক্ষেত্রে এ চিহ্নটি ব্যবহার করা হয়।
প্রক্রিয়াকরণ চিহ্ন (Process)		এটি একটি আয়তাকার চিহ্ন। ডেটা প্রসেসিংয়ের ক্ষেত্রে এই চিহ্নটি ব্যবহার করা হয়। যেমন- কোনো চলকের মান নির্ধারণ বা মান নির্ণয়ের প্রয়োজনে এই চিহ্নটি ব্যবহার করা হয়।
সিদ্ধান্ত চিহ্ন (Decision)		এটি একটি রম্ভসাকৃতির চিহ্ন। যুক্তিমূলক বা সিদ্ধান্তমূলক কাজের ক্ষেত্রে এই চিহ্নটি ব্যবহার করা হয়। যেমন- কোনো দুটি রাশির মানের তুলনার প্রয়োজনে এই চিহ্নটি ব্যবহার করা হয়।
সংযুক্তি চিহ্ন (Connector)		এটি একটি বৃত্তাকার চিহ্ন। একাধিক প্রবাহের সংযোগস্থলে এ চিহ্নটি ব্যবহার করা হয়।
পূর্বনির্ধারিত প্রক্রিয়া (Predefined Process)		একাধিক প্রক্রিয়ার ক্ষেত্রে কোনো পূর্বনির্ধারিত প্রক্রিয়া বর্ণনার জন্য এ চিহ্নটি ব্যবহার করা হয়।
প্রবাহ রেখা (Flow Line)		এটি একটি দিক নির্দেশক সরলরেখা। ফ্লোচার্টের সকল চিহ্নকে এ রেখা দ্বারা যুক্ত করা হয়। এটি প্রবাহের দিক নির্দেশ করে।

## সিস্টেম ফ্লোচার্ট অংকনে ব্যবহৃত কয়েকটি প্রতীক/চিহ্ন/চিত্র:

নাম	প্রতীক	ইম	প্রতীক	নাম	প্রতীক
ডকুমেন্ট		প্রস্তুতি (প্রিপারেশন)		কোলেট বা সংযুক্তি	
মাল্টি ডকুমেন্ট		অফ পেজ কানেক্টর		স্ট বা সাজানো	
ডিসপ্লি বা প্রদর্শন		কার্ড		মার্জ বা এক্ত্রিকরণ	
ইন্টারনাল স্টোরেজ		পাঞ্চ টেপ		সংরক্ষিত (স্টোর্ড) ডেটা	
ম্যানুয়েল ইনপুট		ম্যানুয়েল অপারেশন		ডিলে	
ম্যাগনেটিক ডিস্ক		ডাইরেক্ট অ্যাক্সেস মেমরি		ম্যাগনেটিক টেপ	

### কয়েকটি প্রয়োজনীয় অ্যালগরিদম ও ফ্লোচার্ট

**অ্যালগরিদম ও ফ্লোচার্ট -১ :** তিনটি সংখ্যার যোগফল এবং গড় নির্ণয়ের অ্যালগরিদম লেখ এবং ফ্লোচার্ট আঁক।

নিচের ধাপসমূহ অনুসরণের মাধ্যমে তিনটি সংখ্যার যোগফল ও গড় নির্ণয় করা যাবে:

ধাপ-১ : কাজ শুরু।

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

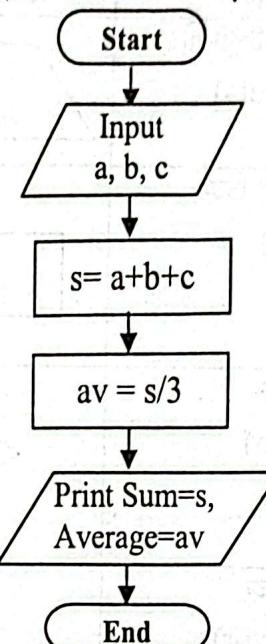
ধাপ-৩ : তিনটি সংখ্যা  $a, b$  এবং  $c$ -এর যোগফল  $s$  নির্ণয়। অর্থাৎ  $s=a+b+c$  নির্ণয়।

ধাপ-৪ : যোগফল  $s$  কে ৩ দিয়ে ভাগ করে গড়  $av$  নির্ণয়। অর্থাৎ  $av=s/3$  নির্ণয়।

ধাপ-৫ : যোগফল  $s$  এবং গড়  $av$ -এর মান প্রদর্শন।

ধাপ-৬ : কাজ শেষ।

নিচে তিনটি সংখ্যার যোগফল ও গড় নির্ণয়ের ফ্লোচার্ট দেখানো হলো:



**অ্যালগরিদম ও ফ্লোচার্ট-২ :** সেলসিয়াস তাপমাত্রাকে ফারেনহাইটে প্রকাশের অ্যালগরিদম লেখ এবং ফ্লোচার্ট আঁক।

নিচের ধাপসমূহ অনুসরণের মাধ্যমে সেলসিয়াস তাপমাত্রাকে ফারেনহাইটে প্রকাশ করা যাবে:

ধাপ-১ : কাজ শুরু।

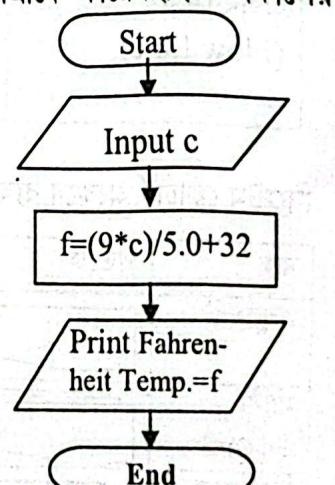
ধাপ-২ : সেলসিয়াস তাপমাত্রার মান  $c$  এর মান গ্রহণ।

ধাপ-৩ : ফারেনহাইট তাপমাত্রার মান  $f$  নির্ণয়।  
অর্থাৎ  $f=(9*c)/5.0+32$  নির্ণয়।

ধাপ-৪ : ফারেনহাইট তাপমাত্রা  $f$ -এর মান প্রদর্শন।

ধাপ-৬ : কাজ শেষ।

নিচে সেলসিয়াস তাপমাত্রাকে ফারেনহাইটে প্রকাশের ফ্লোচার্ট দেখানো হলো:



**স্বরূপীয় :** সেলসিয়াস তাপমাত্রাকে ফারেনহাইটে প্রকাশের জন্য প্রয়োজনীয় সূত্রটি হলো  $\frac{c}{5} = \frac{f-32}{9}$ । সূত্রটি

হতে ফারেনহাইট তাপমাত্রার মান নির্ণয়ের জন্য  $f = \frac{9c}{5} + 32$ । ৩২ সূত্রটি পাওয়া যায়। প্রাপ্ত সূত্রে সেলসিয়াস তাপমাত্রা  $c$ -এর যে কোনো মান বসিয়ে সরল করলে ফারেনহাইট তাপমাত্রা  $f$ -এর মান পাওয়া যায়।

কাজ →

ক) দুইটি সংখ্যার যোগফল এবং গড় নির্ণয়ের অ্যালগরিদম লেখ এবং ফ্লোচার্ট আঁক।

খ) ফারেনহাইট তাপমাত্রাকে সেলসিয়াসে প্রকাশের অ্যালগরিদম লেখ ও ফ্লোচার্ট আঁক।

অ্যালগরিদম ও ফ্রোচার্ট-৩ : ত্রিভুজের ভূমি ও উচ্চতা হতে এর ক্ষেত্রফল নির্ণয়ের অ্যালগরিদম লেখ এবং ফ্রোচার্ট আঁক।

নিচের ধাপসমূহ অনুসরণ করে একটি ত্রিভুজের ভূমি ও উচ্চতা হতে এর ক্ষেত্রফল নির্ণয় করা যাবে:

ধাপ-১ : কাজ শুরু।

ধাপ-২ : ত্রিভুজটির ভূমি  $b$  ও উচ্চতা  $h$  এর মান গ্রহণ।

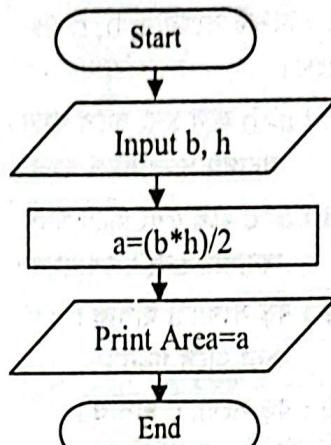
ধাপ-৩ : ভূমি  $b$  ও উচ্চতা  $h$ -এর গুণফলের অর্ধেক  $a$

নির্ণয়। অর্থাৎ  $a = (b * h) / 2$  নির্ণয়।

ধাপ-৪ : ত্রিভুজটির ক্ষেত্রফল  $a$ -এর মান প্রদর্শন।

ধাপ-৫ : কাজ শেষ।

নিচে একটি ত্রিভুজের ভূমি ও উচ্চতা হতে এর ক্ষেত্রফল নির্ণয়ের ফ্রোচার্ট দেখানো হলো:



কাজ

ত্রিভুজের তিন বাহুর দৈর্ঘ্যের হতে ক্ষেত্রফল নির্ণয়ের অ্যালগরিদম লেখ ও ফ্রোচার্ট আঁক।

অ্যালগরিদম ও ফ্রোচার্ট-৪ : ১টি পূর্ণসংখ্যা জোড় না বিজোড় তা নির্ধারনের অ্যালগরিদম লেখ এবং ফ্রোচার্ট আঁক।

নিচের ধাপসমূহ অনুসরণের মাধ্যমে একটি সংখ্যা জোড় না বিজোড় তা নির্ধারন করা যাবে:

ধাপ-১ : কাজ শুরু।

ধাপ-২ : পূর্ণ সংখ্যা  $n$  এর মান গ্রহণ।

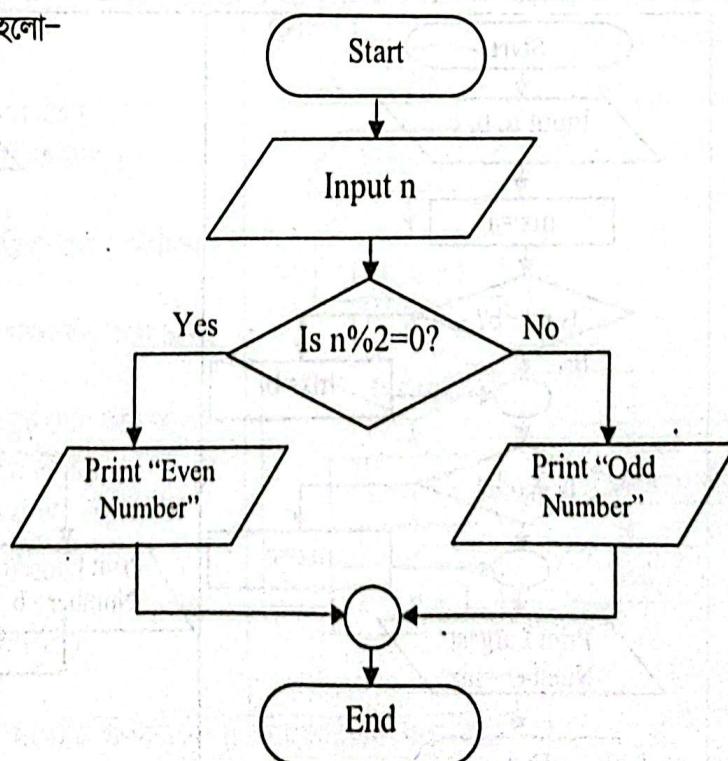
ধাপ-৩ :  $n/2$  -এর ভাগশেষ ০ হলে  
৪ নং ধাপে গমন। অন্যথায়  
৫ নং ধাপে গমন।

ধাপ-৪ :  $n$  একটি জোড় সংখ্যা  
ছাপাও। ৬ নং ধাপে গমন।

ধাপ-৫ :  $n$  একটি বিজোড় সংখ্যা  
ছাপাও। ৬ নং ধাপে গমন।

ধাপ-৬ : কাজ শেষ।

নিচে একটি পূর্ণ সংখ্যা জোড় না বিজোড় তা বের করার ফ্রোচার্টটি দেখানো হলো-



কাজ

১টি পূর্ণ সংখ্যা ধনাত্মক না খনাত্মক তা নির্ধারনের অ্যালগরিদম লেখ এবং ফ্রোচার্ট আঁক।

অ্যালগরিদম ও ফ্লোচার্ট -৯ : দুটি পূর্ণ সংখ্যার গ.সা.গ নির্ণয়ের অ্যালগরিদম এবং ফ্লোচার্ট আঁক।

#### অ্যালগরিদম:

ধাপ-১ : কাজ শুরু।

ধাপ-২ : দুটি পূর্ণ সংখ্যা  $a$  ও  $b$  এর মান গ্রহণ, যেখানে  $a > b$ ।

ধাপ-৩ :  $a$  কে  $b$  দিয়ে ভাগ করে ভাগশেষ  $d$  নির্ণয়। অর্থাৎ  $d = a \% b$  নির্ণয়।

ধাপ-৪ : ভাগশেষ  $d$ -এর মান ০ হলে ৮ নং ধাপে গমন। অন্যথায় ৫ নং ধাপে গমন।

ধাপ-৫ :  $a$ -এর মান  $b$ -এর সমান নির্ধারণ। অর্থাৎ  $a=b$  নির্ধারণ।

ধাপ-৬ :  $b$ -এর মান  $d$ -এর সমান নির্ধারণ। অর্থাৎ  $b=d$  নির্ধারণ।

ধাপ-৭ : ৩ নং ধাপে গমন।

ধাপ-৮ : গ.সা.গ  $b$  এর মান প্রদর্শন।

ধাপ-৯ : কাজ শেষ।

#### বিকল্প অ্যালগরিদম:

ধাপ-১ : কাজ শুরু।

ধাপ-২ : দুটি পূর্ণ সংখ্যা  $a$  ও  $b$  -এর মান গ্রহণ, যেখানে  $a > b$ ।

ধাপ-৩ :  $d$  কে ছোট সংখ্যা  $b$  -এর সমান নির্ধারণ।

ধাপ-৪ :  $a$  ও  $b$  কে  $d$  দিয়ে ভাগ করে ভাগশেষ ঘাচাই।

ধাপ-৫ : উভয় ভাগশেষের মান ০ হলে ৮নং ধাপে গমন। অন্যথায় ৬ নং ধাপে গমন।

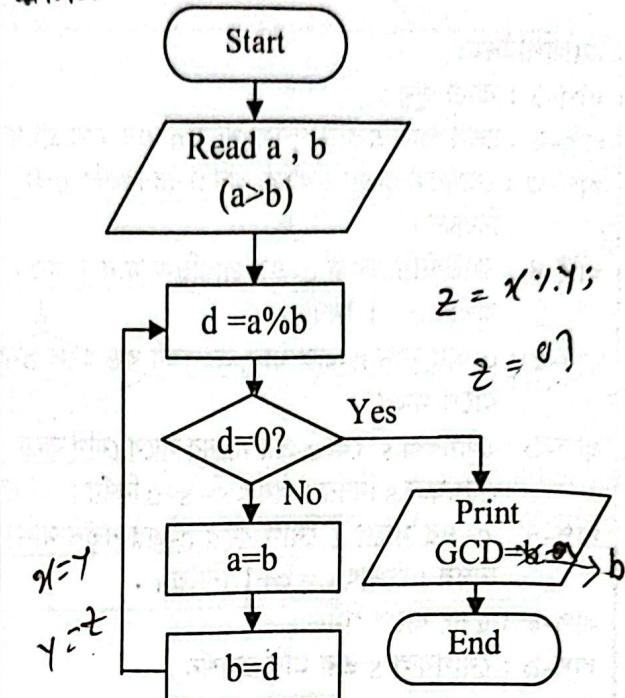
ধাপ-৬ :  $d$  -এর মান হতে ১ বিয়োগ করে  $d$  -এর নতুন মান নির্ধারণ। অর্থাৎ  $d = d - 1$  নির্ধারণ।

ধাপ-৭ : ৪ নং ধাপে গমন।

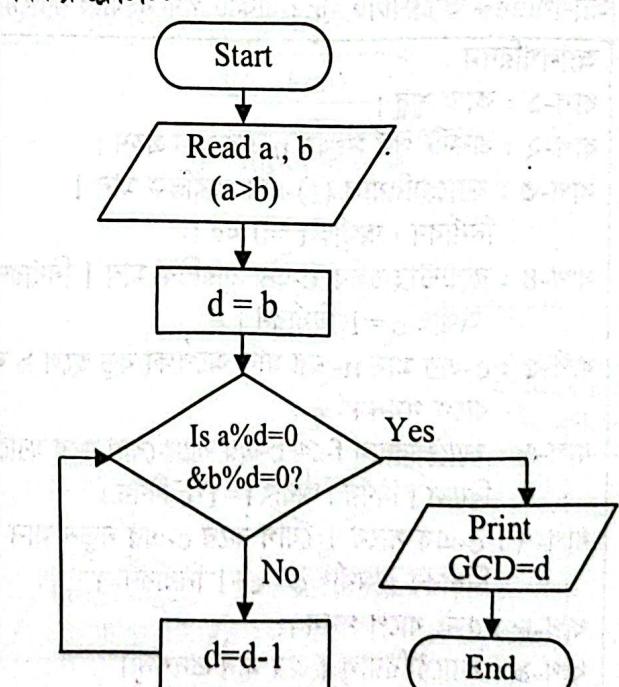
ধাপ-৮ : গ.সা.গ  $d$  -এর মান প্রদর্শন।

ধাপ-৯ : কাজ শেষ।

#### ফ্লোচার্ট:



#### বিকল্প ফ্লোচার্ট:



কাজ

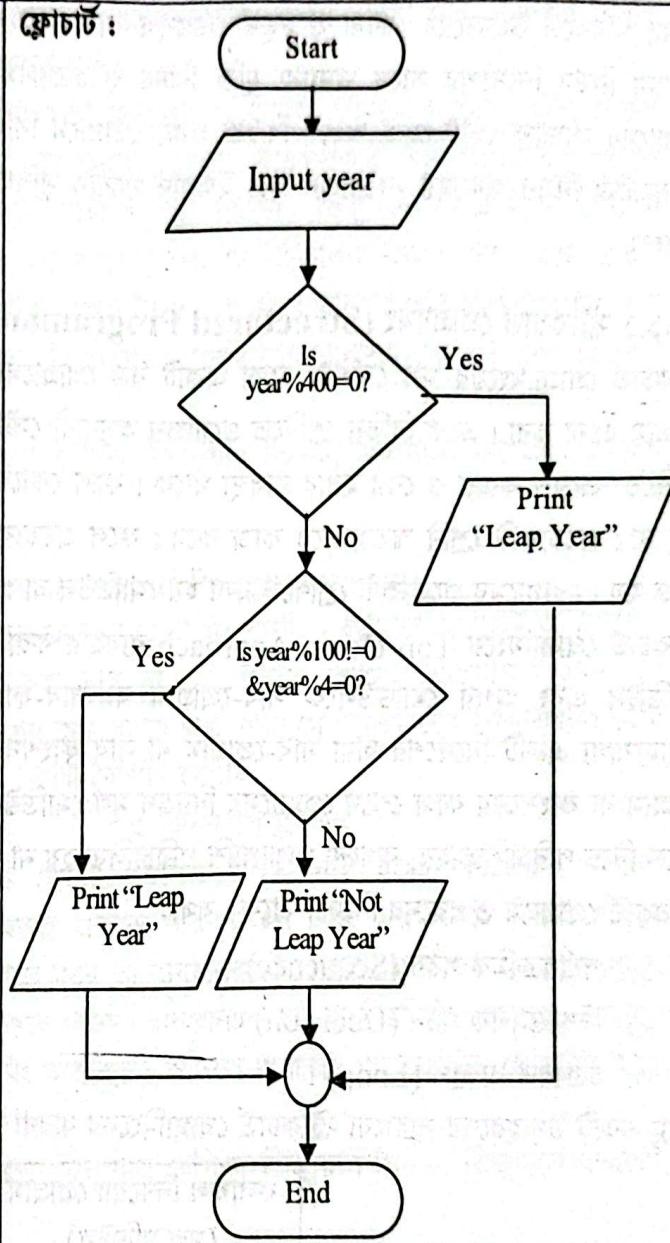
দুটি সংখ্যার গ.সা.গ নির্ণয়ের ফ্লোচার্ট আঁক ও অ্যালগরিদম লেখ।

ফ্রেচার্ট ও অ্যালগরিদম -১০ : একটি সাল/সন অধিবর্ষ বা লিপ ইয়ার (Leap Year) কি না তা নির্ণয়ের ফ্রেচার্ট আঁক ও অ্যালগরিদম লেখ:

#### অ্যালগরিদম :

- ধাপ-১ : কাজ শুরু।
- ধাপ-২ : একটি সন year -এর মান এর মান গ্রহণ।
- ধাপ-৩ : year কে 400 দিয়ে ভাগ করে ভাগশেষ যাচাই।
- ধাপ-৪ : ভাগশেষ এর মান 0 হলে year-টি Leap Year হাপাও এবং ৮ নং ধাপে গমন। অন্যথায় ৫ নং ধাপে গমন।
- ধাপ- ৫ : year কে 100 দিয়ে ভাগ করে ভাগশেষ 0 না হলে এবং ৪ দিয়ে ভাগ করে ভাগশেষ যাচাই।
- ধাপ- ৬ : ভাগশেষ এর মান 0 হলে year-টি Leap Year হাপাও এবং ৮ নং ধাপে গমন। অন্যথায় ৭ নং ধাপে গমন।
- ধাপ- ৭ : year-টি Leap Year নয়।
- ধাপ- ৮ : কাজ শেষ।

#### ফ্রেচার্ট :



**দ্রষ্টব্য :** সাধারণত কোনো একটি বর্ষ (সন/সাল) চার দ্বারা নিঃশেষে বিভাজ্য হলে Leap year হয়। তবে শতাব্দির ক্ষেত্রে, চার দ্বারা বিভাজ্য হলেও সকল শতাব্দি লিপ-ইয়ার নয়। যে সকল শতাব্দি 800 দ্বারা বিভাজ্য কেবলমাত্র সেই সকল শতাব্দিই লিপ ইয়ার। যেমন: 8 দ্বারা বিভাজ্য হলেও ১৭০০, ১৮০০, ১৯০০, ২১০০ সাল লিপ ইয়ার নয়। কারণ এ শতাব্দিগুলো 800 দ্বারা বিভাজ্য নয়। অপর পক্ষে, ১৬০০, ২০০০, ২৪০০ সাল লিপ ইয়ার।

কাজ

একটি মাত্র যুক্তিমূলক বাক্য ব্যবহার করে লিপ ইয়ার নির্ণয়ের অ্যালগরিদমটি লেখ।