[অনুগ্রহ করে পিডিএফটি সময় নিয়ে পড়বেন
এখানে বিভিন্ন অধ্যায়ের কিছু অংশ শেয়ার করা হলো]

NTRCA ICT Combo

Lecturer-Code(452), Assistant Teacher-Code(313) & Demonstrator-Code(325)

MD. SHAHRIAR NAZIM JOY

"বিসমিল্লাহির রাহমানির রাহিম"

সূচিপত্র

কোনো টপিকস সহজেই বুঝতে চাইলে সাবস্ক্রাইব করুন আমাদের ইউটিউব চ্যানেল।

Follow us to updates and any query
https://www.synchronousboi.com

‘আর এই যে, মানুষ তাই পায়, যা সে চেষ্টা করে।’ (সুরা: আন নাজম, আয়াত: ৩৯)

Chapter - 2
কম্পিউটারের বেসিক

১. কম্পিউটারের সংজ্ঞা

কম্পিউটার হলো এক ধরনের ইলেকট্রনিক যন্ত্র যা ইনপুটকৃত ডেটাকে প্রসেসিংয়ের মাধ্যমে তথ্য (Information) এ রূপান্তরিত করে। এটি নির্দেশ (Instruction) অনুসারে কাজ সম্পন্ন করে, ফলাফল আউটপুট হিসেবে প্রদর্শন করে এবং ভবিষ্যতের জন্য তথ্য সংরক্ষণ (Store) করতে পারে।

২. কম্পিউটারের বৈশিষ্ট্যসমূহ

- দ্রুতগতি: কম্পিউটার অতি অল্প সময়ে বিপুল পরিমাণ হিসাব-নিকাশ সম্পন্ন করতে পারে।
- নির্ভুলতা: নির্দেশ সঠিক থাকলে এটি শতভাগ নির্ভুল ফলাফল দেয়।
- সংগ্রহক্ষমতা: বিপুল পরিমাণ তথ্য সংরক্ষণ করতে সক্ষম।
- স্বয়ংক্রিয়তা: প্রোগ্রাম অনুযায়ী স্বয়ংক্রিয়ভাবে কাজ করে।
- বহুমুখিতা: শিক্ষা, চিকিৎসা, ব্যবসা, বিজ্ঞানসহ নানা ক্ষেত্রে ব্যবহৃত হয়।
- অবিরাম কাজের ক্ষমতা: ক্লান্তি ছাড়া ২৪ ঘণ্টা কাজ করতে পারে।

৩. কম্পিউটারের ইতিহাস (আবাকাস থেকে প্রথম বাণিজ্যিক কম্পিউটার পর্যন্ত)

- আবাকাস (Abacus): প্রাচীন যুগের গণনার যন্ত্র।
- প্যাসকেলাইন (Pascaline): 1642 সালে ব্লেইস প্যাসকেল উদ্ভাবন করেন।
- স্টেপড রেকনার: গটফ্রিড লিবনিজ তৈরি করেন।
- অ্যানালাইটিক্যাল ইঞ্জিন: চার্লস ব্যাবেজের উদ্ভাবিত যান্ত্রিক কম্পিউটার—“কম্পিউটারের জনক” বলা হয়।
- হোলারিথ ট্যাবুলেটিং মেশিন: হারমান হোলারিথ, 1890 সালে।
- ENIAC (Electronic Numerical Integrator and Computer): প্রথম ইলেকট্রনিক কম্পিউটার (1946)।
- UNIVAC-I: প্রথম বাণিজ্যিক কম্পিউটার (1951)।

কম্পিউটার সংজ্ঞা, বৈশিষ্ট্যসমূহ, কম্পিউটারের ইতিহাসের এমসিকিউ

১. কম্পিউটার হলো কী?
(A) একটি যান্ত্রিক যন্ত্র যা ডেটা সংরক্ষণ করে
(B) একটি ইলেকট্রনিক যন্ত্র যা ইনপুট ডেটাকে প্রসেসিং করে তথ্য তৈরি করে
(C) একটি শিক্ষামূলক যন্ত্র যা কেবল গণনা শেখায়
(D) একটি যন্ত্র যা শুধুমাত্র আউটপুট প্রদর্শন করে
উত্তর: B

২. কম্পিউটার কোন বৈশিষ্ট্য দ্বারা অত্যন্ত দ্রুত হিসাব-নিকাশ করতে পারে?
(A) নির্ভুলতা (B) অবিরাম কাজের ক্ষমতা
(C) দ্রুতগতি (D) বহুমুখিতা
উত্তর: C

৩. কম্পিউটার কতটা নির্ভুল ফলাফল দেয়?
(A) ৫০%
(B) ৭৫%
(C) শতভাগ, যদি নির্দেশ সঠিক হয়
(D) ৯০%
উত্তর: C

৪. কোন বৈশিষ্ট্যটি কম্পিউটারকে ক্লান্তি ছাড়া ২৪ ঘণ্টা কাজ করতে সক্ষম করে?
(A) সংগ্রহক্ষমতা (B) অবিরাম কাজের ক্ষমতা

কম্পিউটারের প্রকারভেদ ও শ্রেণীবিন্যাসের এমসিকিউ

১. অ্যানালগ কম্পিউটার কী ধরনের ডেটা প্রক্রিয়াজাত করে?
(A) বাইনারি সংখ্যা (0 ও 1)
(B) ধারাবাহিক (Continuous) মান
(C) শুধুমাত্র সংখ্যা
(D) অক্ষরের সংমিশ্রণ
উত্তর: B

২. অ্যানালগ কম্পিউটারের উদাহরণ কোনটি?
(A) ডেস্কটপ কম্পিউটার (B) থার্মোমিটার
(C) ল্যাপটপ (D) ব্যাংকের ATM
উত্তর: B

৩. ডিজিটাল কম্পিউটার কী ধরনের ডেটা ব্যবহার করে?
(A) ধারাবাহিক ডেটা (B) বাইনারি সংখ্যা (0 ও 1)
(C) গ্রাফ ও ডায়াল (D) সিমুলেশন ডেটা
উত্তর: B

৪. ডিজিটাল কম্পিউটারের প্রধান বৈশিষ্ট্য কী?
(A) নির্ভুল ও দ্রুতগতি সম্পন্ন
(B) শুধুমাত্র নির্দিষ্ট উদ্দেশ্য
(C) ফলাফল গ্রাফে প্রদর্শিত
(D) অতি দ্রুত গণনা করতে সক্ষম
উত্তর: A

৫. হাইব্রিড কম্পিউটার কীভাবে কাজ করে?
(A) শুধুমাত্র অ্যানালগ ডেটা ব্যবহার করে
(B) শুধুমাত্র ডিজিটাল ডেটা ব্যবহার করে
(C) অ্যানালগ ও ডিজিটাল উভয় ডেটা প্রক্রিয়াজাত করে
(D) কেবল গ্রাফিকাল আউটপুট দেয়
উত্তর: C

৬. মাইক্রো কম্পিউটার কোন উদ্দেশ্যে ব্যবহার হয়?
(A) ব্যক্তিগত ও ছোট অফিস কাজ
(B) মহাকাশ গবেষণা
(C) ব্যাংক ও বিমা
(D) রকেট লঞ্চ কন্ট্রোল
উত্তর: A

৭. মিনি কম্পিউটার কী বৈশিষ্ট্য রাখে?
(A) একাধিক ব্যবহারকারী একসঙ্গে কাজ করতে পারে
(B) শুধুমাত্র এক ব্যবহারকারী
(C) সর্বাধিক Floating Point Operations/sec
(D) শুধু বৈজ্ঞানিক গবেষণা
উত্তর: A

৮. মেইনফ্রেম কম্পিউটার প্রধানত কোন ক্ষেত্রে ব্যবহৃত হয়?
(A) ব্যক্তিগত কাজ (B) ব্যাংক, বিমা, সরকারি সংস্থা
(C) হোম এন্টারটেইনমেন্ট
(D) মোবাইল গেমস
উত্তর: B

৯. সুপার কম্পিউটারের উদাহরণ কোনটি?
(A) Intel 8086 (B) IBM Summit
(C) DEC PDP-11 (D) Desktop PC
উত্তর: B

১০. সুপার কম্পিউটার ব্যবহৃত হয় কোন ক্ষেত্রে?
(A) ইমেইল ও অফিস কাজ
(B) আবহাওয়ার পূর্বাভাস, পারমাণবিক পরীক্ষা
(C) ব্যাংক ট্রান্সজেশন
(D) মোবাইল অ্যাপস
উত্তর: B

১১. সাধারণ উদ্দেশ্য কম্পিউটার (General Purpose Computer) কীভাবে ব্যবহার হয়?
(A) এক ধরনের নির্দিষ্ট কাজের জন্য
(B) নানা ধরনের কাজ সম্পন্ন করতে পারে এবং প্রোগ্রাম পরিবর্তনযোগ্য
(C) শুধুমাত্র বৈজ্ঞানিক গবেষণার জন্য
(D) শুধুমাত্র ব্যাংকিং কাজে
উত্তর: B

Self Test

১. কম্পিউটার সিস্টেমের প্রধান দুটি অংশ কী?
A) ইনপুট ও আউটপুট
B) হার্ডওয়্যার ও সফটওয়্যার
C) সিপিইউ ও মনিটর
D) মেমরি ও প্রসেসর

২. হার্ডওয়্যার কী?
A) কম্পিউটারের অদৃশ্য অংশ
B) প্রোগ্রাম বা নির্দেশনা
C) দৃশ্যমান ও স্পর্শযোগ্য অংশ
D) শুধুমাত্র ইনপুট ডিভাইস

৩. নিচের কোনটি হার্ডওয়্যারের উদাহরণ নয়?
A) কিবোর্ড B) মাউস
C) মনিটর D) MS Word

৪. হার্ডওয়্যারের প্রধান বৈশিষ্ট্য কোনটি?
A) পরিবর্তনযোগ্য B) দৃশ্যমান নয়
C) ভেঙে যেতে পারে D) নির্দেশনার সমষ্টি

৫. হার্ডওয়্যার কাজ করার জন্য কী প্রয়োজন?
A) অপারেটিং সিস্টেম B) সফটওয়্যার
C) ইউটিলিটি প্রোগ্রাম D) ইন্টারনেট

৬. ইনপুট ডিভাইস কোনটি?
A) মনিটর B) প্রিন্টার
C) স্ক্যানার D) স্পিকার

৭. আউটপুট ডিভাইসের মধ্যে কোনটি পড়ে?
A) কিবোর্ড B) স্ক্যানার
C) মনিটর D) মাউস

৮. কোন সফটওয়্যারটি প্রেজেন্টেশন তৈরিতে ব্যবহৃত হয়?
A) Word B) Excel
C) PowerPoint D) Paint

৯. সফটওয়্যারকে স্পর্শ করা যায় না, তবে —
A) দেখা যায় না
B) দেখা যায় (ডিসপ্লেতে)
C) শোনা যায়
D) নষ্ট করা যায়

১০. সফটওয়্যারের প্রধান কাজ কী?
A) হার্ডওয়্যার তৈরি করা
B) হার্ডওয়্যারকে কার্যকর করা
C) বিদ্যুৎ সরবরাহ করা
D) ডেটা সংরক্ষণ করা

১১. সফটওয়্যার কয় ভাগে বিভক্ত?
A) দুই ভাগে B) তিন ভাগে
C) চার ভাগে D) পাঁচ ভাগে

১২. নিচের কোনটি অ্যাপ্লিকেশন সফটওয়্যার নয়?
A) MS Word B) Excel
C) Linux D) Photoshop

১৩. নিচের কোনটি সিস্টেম সফটওয়্যারের উদাহরণ?
A) MS Word
B) Adobe Photoshop
C) Windows
D) Chrome

১৪. ইউটিলিটি প্রোগ্রামের উদাহরণ কোনটি?
A) Antivirus B) PowerPoint
C) Excel D) Firefox

১৫. ওয়েব ব্রাউজারের কাজ কী?
A) ফাইল সংরক্ষণ B) ইন্টারনেট ব্যবহার
C) ভাইরাস প্রতিরোধ D) হার্ডওয়্যার সংযোগ

1. B	2. C	3. D	4. C	5. B	6. C	7. C	8. C
9. B	10. B	11. A	12. C	13. C	14. A	15. B	

Chapter - 3
সংখ্যা পদ্ধতি

সংখ্যা আবিষ্কারের ইতিহাস (History of inventing Numbers)

সভ্যতার সূচনালগ্ন থেকেই মানুষ হিসাব-নিকাশের প্রয়োজনীয়তা অনুভব করে। তখন গণনার জন্য নানা রকম উপকরণ যেমন- হাতের আঙুল, নুড়ি পাথর, কাঠি, ঝিনুক, রশির গিঁট, দেয়ালে দাগ কাটা ইত্যাদি ব্যবহার করা হতো। সময়ের বিবর্তনে গণনার ক্ষেত্রে বিভিন্ন চিহ্ন ও প্রতীক ব্যবহার শুরু হতে থাকে। খ্রিষ্টপূর্ব ৩৪০০ সালে হায়ারোগ্লিফিক্স সংখ্যা পদ্ধতির মাধ্যমে সর্বপ্রথম গণনার ক্ষেত্রে লিখিত সংখ্যা বা চিহ্নের ব্যবহার শুরু হয়। পরবর্তীতে পর্যায়ক্রমে মেয়ান, রোমান ও দশমিক সংখ্যা (বহুল ব্যবহৃত পদ্ধতি) পদ্ধতির ব্যবহার শুরু হয়। ভগ্নাংশের ধারণার প্রচলন ঘটে মিসরে।

শূন্য আবিষ্কারের ইতিহাসঃ গণিতে 'শূন্য' আবিষ্কার ভারতবর্ষে হয়েছিলো। ভারতবর্ষে সর্ব প্রথম শূন্য এর ব্যবহার দেখা যায়। ভারতের গণিতবিদ পিংগালাকে শূন্যের আবিষ্কারক বলা হয়। আর্য ভট্টকে ও শূন্যের আবিষ্কারক বলা হয়।

সংখ্যা পদ্ধতি আবিষ্কারের ইতিহাস গুরুত্বপূর্ন টিপস

১. ব্যবিলিয়নরা ব্যবহার করতো ৬০ ভিত্তিক সংখ্যা পদ্ধতি।
২. মেয়ানরা ২০ ভিত্তিক সংখ্যা এবং ৫ ভিত্তিক সংখ্যা পদ্ধতি ব্যবহার করতো।
৩. সংখ্যা গণনার পদ্ধতি হলো সংখ্যা পদ্ধতি।
৪. রোমান ও ইউরোপিয়ান সভ্যতায় শূন্যের ব্যবহার ছিল না।
৫. প্রাচীনকালের গণনায় ব্যবহৃত হতো প্রতীক বা চিহ্ন।
৬. মিশরীয়দের সংখ্যা পদ্ধতি ব্যবস্থা ছিল ১০ ভিত্তিক।
৭. গ্রিকদের সংখ্যা পদ্ধতি ছিল দশ ভিত্তিক।
৮. গণিতের যুগান্তকারী আবিষ্কার হলো শূন্য।
৯. দশ ভিত্তিক সংখ্যা পদ্ধতি গড়ে ওঠার কারণ হাতের দশ আঙুল ।
১০. খ্রিষ্টীয় শাসকরা শূন্যকে শয়তানের রূপ মনে করতো ।
১১. অঙ্ক বা প্রতীক হচ্ছে সংখ্যা প্রকাশের ক্ষুদ্রতম একক।

সংখ্যা পদ্ধতি (Number System):

কতগুলো চিহ্ন, বর্ণ, অঙ্ক ও বিশেষ চিহ্নের সাহায্য সংখ্যা প্রদর্শন বা গণনার পদ্ধতিকে সংখ্যা পদ্ধতি বলে।

সংখ্যা পদ্ধতি প্রধানত ২ প্রকারঃ-

১। পজিশনাল/ স্থানিক। ২। ননপজিশনাল/ অস্থানিক।

পজিশনাল সংখ্যা পদ্ধতিঃ যে সংখ্যা পদ্ধতির নির্দিষ্ট বা স্থানীয় মান আছে তাকে পজিশনাল সংখ্যা পদ্ধতি বলে।

নন-পজিশনাল সংখ্যা পদ্ধতিঃ যে সংখ্যা পদ্ধতির নির্দিষ্ট মান আছে কিন্তু স্থানীয় মান নেই তাকে নন-পজিশনাল সংখ্যা পদ্ধতি বলে। প্রাচীন কালে ব্যবহৃত হায়ারোগ্লিফিক্স (Hieroglyphics), মেয়ান ও রোমান সংখ্যা পদ্ধতি নন-পজিশনাল সংখ্যা পদ্ধতির উদাহরণ।

উদাহরণঃ "/\" এই চিহ্নটি প্রাচীনকালে ব্যবহার করা হত যার মান হলো ১০। নন-পজিশনাল সংখ্যা পদ্ধতিতে মূলত শূন্যের ব্যবহার নেই।

পজিশনাল সংখ্যা পদ্ধতি ৪ প্রকারঃ-

১। বাইনারি (binary) ।

২। অক্টাল (octal) ।

৩। ডেসিমেল (decimal) ।

৪। হেক্সাডেসিমেল (hexadecimal) ।

বাইনারিঃ

দুই অঙ্ক বা চিহ্ন বিশিষ্ট সংখ্যা পদ্ধতিকে বাইনারি সংখ্যা পদ্ধতি বলে। এর বেজ বা ভিত্তি ২। ইংল্যান্ডের গণিতবিদ জর্জ বুল বাইনারি সংখ্যা পদ্ধতি উদ্ভাবন করেন। বাইনারি সংখ্যা পদ্ধতি সবচেয়ে সরলতম সংখ্যা পদ্ধতি। চিহ্ন/ অংকঃ 0,1.

বাইনারি সংখ্যা পদ্ধতির কাজঃ

সকল ইলেক্ট্রনিক্স ডিভাইস শুধু দুটি অবস্থা অর্থাৎ বিদ্যুতের উপস্থিতি এবং অনুপস্থিতি বুজতে পারে। বিদ্যুতের উপস্থিতিকে ON/ HIGH/ TRUE কিংবা YES বলা হয় যা লজিক লেভেল ১ নির্দেশ করে এবং বিদ্যুতের অনুপস্থিতিকে OFF/LOW / FALSE কিংবা NO বলা হয় যা লজিক লেভেল ০ নির্দেশ করে। কম্পিউটার বা সকল ইলেক্ট্রনিক্স ডিভাইসে বাইনারি সংখ্যা পদ্ধতি ব্যবহৃত হয়।

অক্টালঃ

আট অঙ্ক বা চিহ্ন বিশিষ্ট সংখ্যা পদ্ধতিকে অক্টাল সংখ্যা পদ্ধতি বলে। এর বেজ বা ভিত্তি ৮।

চিহ্ন/ অংকঃ 0,1,2,3,4,5,6,7

অক্টাল সংখ্যা পদ্ধতির কাজঃ

ডিজিটাল সিস্টেমে বিভিন্ন ক্ষেত্রে বাইনারি সংখ্যাকে নির্ভুল ও সহজে উপস্থাপন করার জন্য অক্টাল সংখ্যা পদ্ধতি ব্যবহার করা হয়। অক্টাল পদ্ধতির উদ্ভাবক সপ্তম চার্লস।

ডেসিমেল(decimal):

দশ অঙ্ক বিশিষ্ট সংখ্যা পদ্ধতিকে ডেসিমেল সংখ্যা পদ্ধতি বলে। এর বেজ বা ভিত্তি ১০ ।

ডেসিমেল সংখ্যার চিহ্ন/ অংকঃ 0,1,2,3,4,5,6,7,8,9

ডেসিমেল (decimal) সংখ্যা পদ্ধতির কাজঃ ইউরোপে আরবরা এই সংখ্যা পদ্ধতির প্রচলন করায় অনেকে এটিকে আরবি সংখ্যা পদ্ধতি নামেও অভিহিত করেন। মানুষ সাধারণত গণনার কাজে ডেসিমেল সংখ্যা পদ্ধতি ব্যবহার করে।

হেক্সাডেসিমেল (Hexadecimal) :

ষোল অঙ্ক বা চিহ্ন বিশিষ্ট সংখ্যা পদ্ধতিকে হেক্সাডেসিমেল সংখ্যা পদ্ধতি বলে। এর বেজ বা ভিত্তি ১৬ ।

হেক্সাডেসিমেল সংখ্যার চিহ্ন/ অংকঃ 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F

হেক্সাডেসিমেল(decimal) সংখ্যা পদ্ধতির কাজঃ

ডিজিটাল সিস্টেমে বিভিন্ন ক্ষেত্রে বাইনারি সংখ্যাকে নির্ভুল ও সহজে উপস্থাপন করার জন্য হেক্সাডেসিমেল সংখ্যা পদ্ধতি ব্যবহার করা হয়। এছাড়া বিভিন্ন মেমোরি অ্যাড্রেস ও রং এর কোড হিসেবে হেক্সাডেসিমেল সংখ্যা পদ্ধতি ব্যবহার করা হয়।

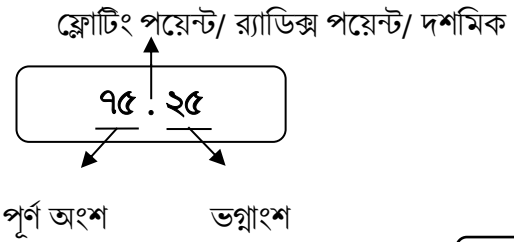
সংখ্যা পদ্ধতির বেজ (Base) বা ভিত্তিঃ

কোনো একটি সংখ্যা পদ্ধতিতে ব্যবহৃত মৌলিক চিহ্ন সমূহের সমষ্টিকে ঐ সংখ্যা পদ্ধতির বেজ (Base) বা ভিত্তি বলে।

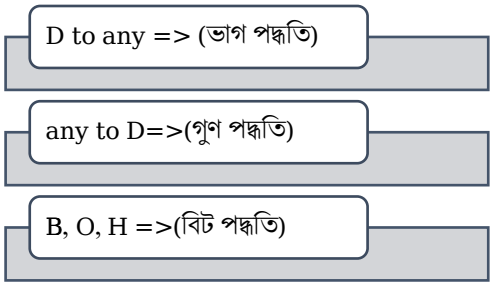
সংখ্যা পদ্ধতির রুপান্তর(Conversion of Numbers)

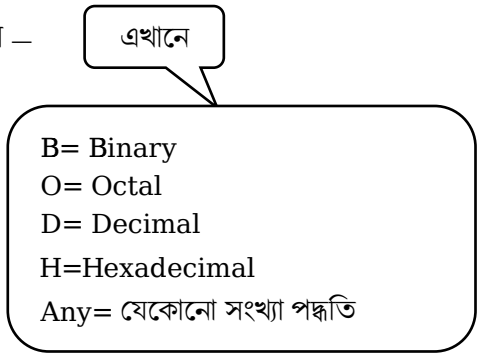
একটি সংখ্যার মূলত ২টি অংশ থাকে- ১। পূর্ণ অংশ। ২। ভগ্নাংশ।

এবং উপাদান সাধারণত থাকে ৩ টি।



পূর্ণ সংখ্যার পরিবর্তনঃ পূর্ণ সংখ্যার পরিবর্তন ৩ টি নিয়ম মেনে চলে –





D to any => (ভাগ পদ্ধতি)

D কে Any করতে হলে any এর বেজ দ্বারা D কে ভাগ করতে হবে। এবং ভাগশেষ সংরক্ষণ করতে হবে উত্তর লেখার সময় উক্ত ভাগশেষ সমূহ নীচ থেকে উপর দিকে লিখতে হবে।

Q-1. $(75)_{10} \longrightarrow (\ ?\ )_2$

2 | 75		LSB
2 | 37	→	1
2 | 18	→	1
2 | 9	→	0
2 | 4	→	1
2 | 2	→	0
2 | 1	→	0
0	→	1
		MSB

উত্তরঃ $(1001011)_2$

Q-2. $(175)_{10} \longrightarrow (\ ?\ )_8$

8 | 175		LSB
8 | 21	→	7
8 | 2	→	5
0	→	2
		MSB

উত্তরঃ $(257)_8$

Q-3. $(380)_{10} \longrightarrow (\ ?\ )_{16}$

16 | 380		LSB
16 | 23	→	12/C
16 | 1	→	7
0	→	1
		MSB

উত্তরঃ $(17C)_{16}$

8. বাইনারি যোগ

বাইনারি যোগ চারটি নিয়ম মেনে চলে-

1. 0 + 0=0
2. 0 + 1=1
3. 1 + 0=1
4. 1 + 1=0, carry 1

27.
```
  1 0 1 1 0
+ 1 0 0 1 1
-----------
1 0 1 0 0 1
```
উত্তরঃ $(101001)_2$

28.
```
  1 0 1 1 1
+ 1 0 0 1 1
-----------
1 0 1 0 1 0
```
উত্তরঃ $(101010)_2$

বাইনারি বিয়োগ

বাইনারি বিয়োগ চারটি নিয়ম মেনে চলে-

1. 0 - 0=0
2. 0 - 1=1, carry 1
3. 1 - 0=1
4. 1 - 1=0

29.
```
  1 0 1 1 1
- 1 0 0 0 1
-----------
0 0 1 1 0
```
উত্তরঃ $(110)_2$

30.
```
  1 0 1 1 0
- 1 0 0 1 1
-----------
0 0 0 1 1
```
উত্তরঃ $(11)_2$

বাইনারি গুণ

বাইনারি বিয়োগ চারটি নিয়ম মেনে চলে-

1. 0 × 0=0
2. 0 × 1=0
3. 1 × 0=0
4. 1 × 1=1

```
  1 0 1
× 1 1
-----
  1 0 1
1 0 1 0
-------
1 1 1 1
```

বাইনারি ভাগ

বাইনারি বিয়োগ চারটি নিয়ম মেনে চলে-

1. 0 ÷ 0= (Undefine)
2. 0 ÷ 1=0
3. 1 ÷ 0= (Undefine)
4. 1 ÷ 1=1

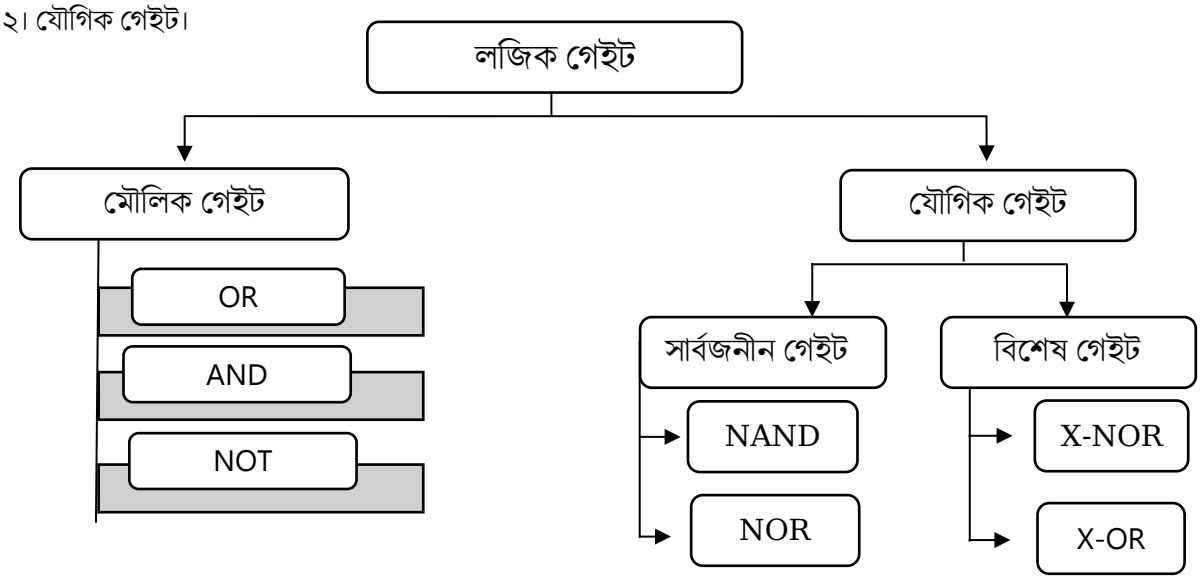
```
10 ) 1 0 1 0 (101
  1 0
  ---
    1 0
    ---
      0
```
Ans: $(101)_2$

লজিক গেইটঃ বুলিয়ান অ্যালজেবরায় মৌলিক কাজগুলো বাস্তবায়নের জন্য যে ডিজিটাল ইলেকট্রনিক সার্কিট বা বর্তনী ব্যবহার করা হয়, তাই লজিক গেইট।

লজিক গেইট ২ প্রকারঃ-

১। মৌলিক গেইট।

২। যৌগিক গেইট।



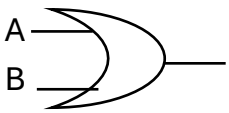
মৌলিক গেইটঃ যে গেইট বাস্তবায়ন করতে অন্য কোনো গেইটের প্রয়োজন হয় না তাকে মৌলিক গেইট বলে।

মৌলিক গেইট ৩ প্রকারঃ-

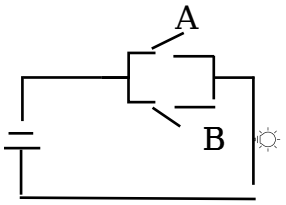
1. অর গেইট বা যৌক্তিক যোগের গেইটঃ

যে গেইট যৌক্তিক যোগের কাজ করে তাকে অর গেইট বলে।

লজিক চিত্রঃ সমীকরণঃ F=A+B সার্কিটঃ



A+B



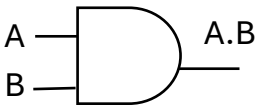
সত্যেক সারণি

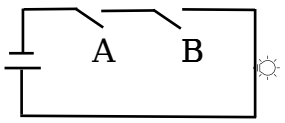
A	B	A+B
0	0	0
0	1	1
1	0	1
1	1	1

2. অ্যান্ড গেইট বা যৌক্তিক গুণনের গেইটঃ

যে গেইট যৌক্তিক গুণনের কাজ করে তাকে অ্যান্ড গেইট বলে।

লজিক চিত্রঃ সমীকরণঃ F=A.B সার্কিটঃ





সত্যেক সারণি

A	B	A.B
0	0	0
0	1	0
1	0	0
1	1	1

3. নট গেইট বা যৌক্তিক উল্টানো বা পূরকের গেইটঃ ইনপুট ও আউটপুটের সংখ্যা সমান থাকে।

যে গেইট যৌক্তিক পূরক/উল্টানোর কাজ করে তাকে নট গেইট বলে।

৩. অ্যালগোরিদম

কোনো একটি নির্দিষ্ট সমস্যা ধাপে ধাপে সমাধানের জন্য যুক্তিসম্মত সসীম সংখ্যক পর্যায়ক্রমিক ধারা বর্ণনাকে একত্রে অ্যালগরিদম বলা হয়। কোনো সমস্যাকে কম্পিউটার প্রোগ্রামিং দ্বারা সমাধান করার পূর্বে কাগজে-কলমে সমাধান করার জন্যই অ্যালগরিদম ব্যবহার করা হয়। আরব গণিতবিদ **'আল খারিজমী'** এর নাম অনুসারে অ্যালগরিদম নামকরণ করা হয়েছে। অ্যালগরিদমের প্রত্যেকটি ধাপসহ ইনপুট এবং আউটপুট স্পষ্টভাবে নির্ধারণ করতে হবে।

অ্যালগরিদম তৈরির সুবিধাঃ

১। এটি একটি ধাপভিত্তিক উপস্থাপনা ফলে সহজে প্রোগ্রামের উদ্দেশ্য বোঝা যায়।

২। একটি অ্যালগরিদম একটি নির্দিষ্ট পদ্ধতি ব্যবহার করে।

৩। এটি কোনও প্রোগ্রামিং ভাষার উপর নির্ভরশীল নয়, তাই প্রোগ্রামিং জ্ঞান ছাড়াই যেকারো পক্ষে এটি বোঝা সহজ।

৪। একটি অ্যালগরিদমের প্রতিটি ধাপের নিজস্ব লজিকাল ক্রম আছে তাই এটি ডিবাগ করা সহজ।

৫। প্রোগ্রাম পরিবর্তন ও পরিবর্ধনে সহায়তা করে।

৪. ফ্লোচার্ট বা প্রবাহ চিত্রঃ

যে চিত্রভিত্তিক পদ্ধতিতে বিশেষ কতকগুলো চিহ্নের সাহায্যে কোনো একটি নির্দিষ্ট সমস্যার সমাধান করা হয় তাকে ফ্লোচার্ট বলা হয়। অন্যভাবে বলা যায়, অ্যালগরিদমের চিত্ররূপই হল ফ্লোচার্ট।

ফ্লোচার্ট তৈরি করার নিয়মাবলিঃ

১। প্রতিটি ফ্লোচার্টের অবশ্যই একটি শুরু এবং শেষ থাকবে।

২। নিয়ন্ত্রণ প্রবাহ অবশ্যই টপ থেকে শুরু হবে।

৩। নিয়ন্ত্রণ প্রবাহ অবশ্যই বটম থেকে শেষ হবে।

৪। প্রচলিত চিহ্ন বা প্রতীক ব্যবহার করে ফ্লোচার্ট তৈরি করতে হবে।

৫। তীর চিহ্ন দিয়ে নিয়ন্ত্রণ প্রবাহ দেখাতে হবে।

৬। ফ্লোচার্টে কোন প্রোগ্রামিং ভাষা ব্যবহার করা যাবে না।

৭। চিহ্ন বা প্রতীক গুলো ছোট বড় হলে ক্ষতি নাই তবে আকৃতি ঠিক থাকতে হবে।

ফ্লোচার্টের সুবিধাঃ

১। একটি প্রোগ্রামের যুক্তির মধ্যে যোগাযোগের চমৎকার উপায় হলো ফ্লোচার্ট ।

২। ফ্লোচার্ট ব্যবহার করে সমস্যা বিশ্লেষণ করা সহজ।

৩। প্রোগ্রাম উন্নয়নের সময়, ফ্লোচার্ট একটি ব্লুপ্রিন্টের ভূমিকা পালন করে ।

৪। ফ্লোচার্ট এর সাহায্যে প্রোগ্রাম বা সিস্টেম রক্ষণাবেক্ষণ সহজ হয়।

৫। ফ্লোচার্টকে যেকোনো প্রোগ্রামিং ভাষার কোডে রূপান্তর করা সহজ।

ফ্লোচার্টের প্রকারভেদঃ ফ্লোচার্টকে প্রধানত দুইভাগে ভাগ করা যায়। যথা-

১। সিস্টেম ফ্লোচার্টঃ সিস্টেম ফ্লোচার্টে ডেটা গ্রহণ, প্রক্রিয়াকরণ, সংরক্ষণ এবং ফলাফল প্রদর্শনের প্রবাহ দেখানো হয়। কোন সিস্টেমের কার্যপ্রণালি বোঝাতে সিস্টেম ফ্লোচার্ট ব্যবহৃত হয়।

২। প্রোগ্রাম ফ্লোচার্টঃ প্রোগ্রাম ফ্লোচার্টে প্রোগ্রামের বিভিন্ন ধাপের বিস্তারিত বিবরণ দেওয়া হয়। এছাড়া প্রোগ্রামের ভুল নির্ণয় ও সংশোধনে প্রোগ্রাম ফ্লোচার্ট ব্যবহৃত হয়। ফ্লোচার্টের চার ধরনের স্ট্রাকচার আছে-

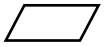
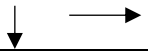
১। সরল অনুক্রম (Simple Sequence)

২। নির্বাচন বা সিলেকশন (Selection)

৩। পুনরাবৃত্তি বা লুপ (Loop)

৪। জাম্প (Jump)

ফ্লোচার্টে ব্যবহৃত প্রতীক সমূহ এবং এদের ব্যবহারঃ

প্রতীক	প্রতিকের নাম	ব্যবহার
	ডিম্বক (Oval)	শুরু এবং শেষ নির্দেশ করে ।
	সামন্তরিক	ইনপুট ও আউটপুট নির্দেশ করে।
	আয়তাকার	প্রক্রিয়াকরণ নির্দেশ করে।
	হীরক	এই প্রতীকের মধ্যে শর্ত / সিদ্ধান্ত লেখা হয়।
	বৃত্ত/কানেক্টর	একাধিক ফলাফল সংযোগ করে।
	তীর চিহ্ন	প্রবাহের দিক নির্দেশে ব্যবহৃত হয়।

সূডোকোড (Pseudo Code):

সূডো (Pseudo) একটি গ্রিক শব্দ যার অর্থ হচ্ছে ছদ্ম বা সত্য নয়। প্রোগ্রামিং করার পূর্বে মূলত সূডোকোড করা হয়। যা কমেন্ট আকারেও প্রোগ্রামে ব্যবহার করা হয়।

অ্যালগরিদম, ফ্লোচার্ট ও সূডোকোড MCQ

১. অ্যালগরিদম কী?
(A) প্রোগ্রামের কোড
(B) ধাপে ধাপে নির্দেশনার সসীম ধারা
(C) ডেটাবেসের নাম
(D) প্রোগ্রামের আউটপুট
উত্তর: B

২. অ্যালগরিদম নামকরণের উৎস কার নাম থেকে এসেছে?
(A) ডেনিস রিচি
(B) আল খারিজমী
(C) James Gosling
(D) Bjarne Stroustrup
উত্তর: B

৩. অ্যালগরিদমে কি থাকতে পারবে না?
(A) ধাপসমূহ (B) ইনপুট
(C) আউটপুট (D) কম্পিউটার কোড
উত্তর: D

৪. অ্যালগরিদমের সুবিধা হলো—
(A) শুধুমাত্র কম্পাইলার ব্যবহার করে
(B) প্রোগ্রাম পরিবর্তন ও ডিবাগিং সহজ হয়
(C) মেশিন ভাষা শিখতে হয়
(D) সমস্যা সমাধান করতে সময় লাগে বেশি
উত্তর: B

৫. দুটি সংখ্যার গড় নির্ণয়ের অ্যালগরিদমে প্রথম ধাপ কী?
(A) avg নির্ণয় করা (B) ইনপুট নেওয়া
(C) শুরু করা (D) ফলাফল দেখানো
উত্তর: C

৬. ফ্লোচার্ট কী?
(A) প্রোগ্রামের কোডের নাম
(B) চিত্রভিত্তিক উপায়ে সমস্যার সমাধান
(C) কম্পাইলার
(D) ডিবাগিং টুল
উত্তর: B

৭. ফ্লোচার্ট তৈরি করার জন্য কোন নিয়ম মেনে চলা হয় না?

Chapter - 8

Structured and Object-Oriented Programming (OOP) Concept

প্রোগ্রাম ডিজাইন মডেল

সহজ উপায়ে কার্যকরী প্রোগ্রাম তৈরির জন্য যে বিশেষ নীতিমালা বা পদ্ধতি অনুসরণ করা হয় তাকে প্রোগ্রাম ডিজাইন মডেল বলে। কয়েকটি জনপ্রিয় প্রোগ্রাম ডিজাইন মডেল-

১। স্ট্রাকচার্ড বা প্রসিডিউর প্রোগ্রামিং মডেল।

২। অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং(OOP) মডেল।

৩। ভিজ্যুয়াল প্রোগ্রামিং মডেল।

৪। ইভেন্ট ড্রাইভেন প্রোগ্রামিং মডেল।

স্ট্রাকচার্ড প্রোগ্রামিং

১৯৬৬ সালে স্ট্রাকচার্ড প্রোগ্রামিং এর প্রথম ধারণা দেন Corrado Bohm এবং Guiseppe Jacopini। এই দুই গণিতবিদ ব্যাখ্যা করেন যে, যেকোন প্রোগ্রাম শুধুমাত্র তিনটি স্ট্রাকচার যেমন- decisions, sequences, এবং loops এর সাহায্যে লেখা যায়। পরবর্তিতে ১৯৭০ সালে Edsger W.Dijkstra ব্যপকভাবে ব্যবহৃত স্ট্রাকচার্ড প্রোগ্রামিং পদ্ধতি উন্নয়ন করেন, যেখানে একটি সমস্যাকে বিভিন্ন ছোট ছোট মডিউল বা অংশে ভাগ করে একটি বড় সমস্যার সমাধান করা হয়। প্রতিটি মডিউলকে ফাংশন বলা হয়।

উদাহরণঃ সি, কোবল, প্যাসকেল, ফরট্রান, কিউবেসিক ইত্যাদি প্রোগ্রামিং ভাষায় স্ট্রাকচার্ড প্রোগ্রামিং ডিজাইন অনুসরণ করে প্রোগ্রাম লেখা যায় এজন্য এই প্রোগ্রামিং ভাষা গুলোকে স্ট্রাকচার্ড প্রোগ্রামিং ভাষা বলা হয়।

অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং

অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং হল এমন একটি প্রোগ্রামিং পদ্ধতি যা স্ট্রাকচার্ড প্রোগ্রামিং এর সুবিধার পাশাপাশি অতিরিক্ত বিশেষ কিছু সুবিধা যেমন- এনক্যাপ্সুলেশন, পলিমরফিজম ও ইনহেরিটেন্স প্রভৃতি ফিচার ব্যবহার করে প্রোগ্রাম লেখার সুবিধা প্রদান করে।

উদাহরণঃ জাভা, পাইথন, সি++ সি# ইত্যাদি হল অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং ভাষা। অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং ভাষার

বৈশিষ্ট্যসমূহ

অবজেক্টঃ অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং ভাষায় যেকোন ব্যাক্তি বা বস্তুকে অবজেক্ট বলা হয়। যেমন – একটি গাড়ি কে অবজেক্ট বলা যায়। প্রতিটি অবজেক্ট এর কিছু বৈশিষ্ট্য(attribute) ও আচরণ(behavior) থাকে। যেমন একটি গাড়ির কালার, মডেল ইত্যাদি হল বৈশিষ্ট্য, আবার গাড়িটি সামনে চলতে পারে এবং পিছনে চলতে পারে এগুলো হল আচরণ।

ক্লাসঃ অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং ভাষায় ক্লাস হল ভেরিয়েবল ও মেথডের সমন্বয়ে একটি টেম্পলেট বা ব্লু-প্রিন্ট যা কোন অবজেক্ট এর বৈশিষ্ট্য(attribute) ও আচরণ(behavior) উপস্থাপনের জন্য তৈরি করা হয়।

এনক্যাপ্সুলেশনঃ অবজেক্ট এর বৈশিষ্ট্য(attribute) ও আচরণ(behavior) কে একত্র করে ক্লাস তৈরি করাকে বলা হয় এনক্যাপ্সুলেশন।

পলিমরফিজমঃ পলিমরফিজম মানে হল বহুরূপ। একাদিক কোড মডিউলের নাম এক হলেও ভিন্ন ভিন্ন রূপ থাকতে পারে, এক্ষেত্রে কোন মডিউলটি কাজ করবে তা নির্ভর করে ডেটা পাঠানোর উপর।

ইনহেরিটেন্সঃ অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং ভাষায় ইনহেরিটেন্স এমন একটি ফিচার, যার কারণে একটি ক্লাসের বৈশিষ্ট্য অপর একটি ক্লাস ব্যবহার করতে পারে, একে বলা হয় ইনহেরিট করা। যে ক্লাস কে ইনহেরিট করা হয় তাকে বলে বেজ ক্লাস এবং যে ক্লাস অন্য ক্লাসকে ইনহেরিট করে তাকে বলে ডিরাইভড ক্লাস।

ভিজ্যুয়াল প্রোগ্রামিংঃ

ভিজ্যুয়াল প্রোগ্রামিং হল এমন একটি প্রোগ্রামিং পদ্ধতি যেখানে বিভিন্ন চিত্রভিত্তিক নির্দেশ বা কমান্ড ব্যবহার করে প্রোগ্রাম রচনা করা হয়। স্ট্রাকচার্ড বা অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং এর উপর ভিত্তি করেই ভিজ্যুয়াল প্রোগ্রামিং মডেল তৈরি। মাইক্রোসফট কোম্পানির তৈরিকৃত ভিজুয়্যাল বেসিক হল প্রথম ভিজ্যুয়াল প্রোগ্রামিং মডেল।

ইভেন্ট ড্রাইভেন প্রোগ্রামিংঃ

সকল ভিজ্যুয়াল প্রোগ্রাম হচ্ছে ইভেন্ট ড্রাইভেন প্রোগ্রাম। এই মডেলে কী-বোর্ডের কোন কী চাপ দেওয়া, কোন বিশেষ কন্ট্রোলের উপর মাউস ক্লিক করা ইত্যাদি কাজ গুলো হচ্ছে এক একটি ইভেন্ট। প্রতিটি ইভেন্টের জন্য পৃথক পৃথক কোড মডিউল থাকে। ব্যবহারকারী যখন কোন ইভেন্ট একটিভ করেন তখন ঐ ইভেন্টের জন্য নির্ধারিত কোড মডিউলটি নির্বাহ হয়।

প্রোগ্রাম ডিজাইন মডেল থেকে গুরুত্বপূর্ন টিপস

১. প্রোগ্রামের গঠন রীতিনীতিকে বলা হয়- প্রোগ্রাম মডেল ।

২. প্রথম চিত্রভিত্তিক প্রোগ্রামিং মডেল হলো- ভিজ্যুয়্যাল বেসিক ।

৩.OOP বা অবজেক্ট ওরিয়েন্টেড প্রোগ্রামিং এর বৈশিষ্ট্য- ইনহেরিটেন্স, এনক্যাপসুলেশন, পলিমরফিজম।

৪. পলিমরফিজম অর্থ হচ্ছে- বহুরূপ ।

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

৬. জলপ্রপাত মডেলে প্রোগ্রাম ডিজাইন ও কোডিং এ সময় ব্যয় হয়- 30 থেকে 40% সময় ।

প্রোগ্রামের ধারণাঃ

কত গুলো নির্দেশনার সমষ্টিকে প্রোগ্রাম বলা হয়। কম্পিউটার প্রোগ্রামও কিছু নির্দেশনার সমষ্টি।

প্রোগ্রামিং ভাষাঃ

কম্পিউটার প্রোগ্রাম তৈরির জন্য কিছু বর্ণ, চিহ্ন, অঙ্ক, বিশেষ চিহ্ন দিয়ে প্রোগ্রাম তৈরির কৌশলকেই প্রোগ্রামিং ভাষা বলে।

প্রোগ্রামিং ভাষার স্তরঃ কম্পিউটার প্রোগ্রামিং ভাষাকে ৫ টি স্তরে ভাগ করা হয়েছে-

১। প্রথম প্রজন্মঃ মেশিন/ যান্ত্রিক ভাষা (১৯৪৫)

২। দ্বিতীয় প্রজন্মঃ অ্যাসেম্বেলি ভাষা (১৯৫০)

৩। তৃতীয় প্রজন্মঃ উচ্চস্তরের ভাষা (১৯৬০)

৪। চতুর্থ প্রজন্মঃ অতি উচ্চস্তরের ভাষা (১৯৭০)

৫। পঞ্চম প্রজন্মঃ স্বাভাবিক বা ন্যাচারাল ভাষা (১৯৮০)

প্রোগ্রাম রচনার ভিত্তিতে ভাষাকে ২ ভাগে ভাগ করাঃ

১। নিম্ন স্তরের ভাষাঃ যান্ত্রিক ও অ্যাসেম্বেলি ভাষা ।

২। উচ্চস্তরের ভাষাঃ উচ্চস্তরের, অতি উচ্চস্তরের ও ন্যাচারাল ভাষা।

মেশিন বা যান্ত্রিক ভাষাঃ

যে ভাষায় 0 এবং 1 ব্যবহার করে প্রোগ্রাম বা কোড লেখা হয় তাকে মেশিন বা যান্ত্রিক ভাষা বলে। কম্পিউটারের নিজস্ব ভাষা হচ্ছে মেশিন ভাষা। মেশিন ভাষায় লেখা কোনো প্রোগ্রামকে অবজেক্ট বা বস্তু প্রোগ্রাম বলা হয়।

মেশিন ভাষার সুবিধা:

১। মেশিন ভাষার সবচেয়ে বড় সুবিধা হচ্ছে এই ভাষায় লেখা প্রোগ্রাম কম্পিউটার সরাসরি বুঝতে পারে।

২। কোনো প্রকার অনুবাদক প্রোগ্রামের প্রয়োজন হয় না।

৩। দ্রুত কাজ করে।

৪। মেশিন ভাষায় লেখা প্রোগ্রামে অতি অল্প মেমোরি প্রয়োজন হয়।

মেশিন ভাষার অসুবিধা:

১। শুধু ০ ও ১ ব্যবহার করা হয় বলে মেশিন ভাষা শেখা কষ্টকর ও প্রোগ্রাম লেখাও কষ্টসাধ্য।

২। প্রোগ্রাম লিখতে প্রচুর সময় লাগে।

৩। এই ভাষায় লেখা কোনো প্রোগ্রাম সহজে বোঝা যায় না।

৪। এই ভাষায় প্রোগ্রাম লিখলে ভুল হওয়ার সম্ভাবনা খুব বেশি থাকে।

৫। ভুল হলে তা বের করা এবং ভুল-ত্রুটি দূর করা অনেক কঠিন।

৬। এই ভাষা হলো যন্ত্র নির্ভর ভাষা যা এক ধরনের যন্ত্রে জন্য লেখা প্রোগ্রাম অন্য ধরনের যন্ত্রে ব্যবহার করা যায় না।

৭। এই ভাষায় প্রোগ্রাম রচনার ক্ষেত্রে কম্পিউটারের অভ্যন্তরীণ সংগঠন ভালোভাবে জানতে হয়।

অ্যাসেম্বলি ভাষাঃ

মেশিন ভাষায় নির্দেশনা দিতে বা প্রোগ্রাম লিখতে অনেক পরিশ্রম হত । ভুল সংশোধন করা কষ্টকর ছিল। এই সমস্যা সমাধানের জন্য সাংকেতিক চিহ্ন(ADD, SUM, DIV, MUL, INP, OUT, CLR ইত্যাদি) দিয়ে নতুন ভাষা তৈরি করা হয় । যে ভাষায় বিভিন্ন সংকেত বা নেমোনিক ব্যবহার করে প্রোগ্রাম লেখা হয় তাকে অ্যাসেম্বলি ভাষা বলে। অ্যাসেম্বলি ভাষায় প্রোগ্রাম লেখার জন্য ০ ও ১ ব্যবহার না করে বিভিন্ন সংকেত ব্যবহার করা হয়। এই সংকেতকে বলে সাংকেতিক কোড (Symbolic Code) বা নেমোনিক (Nemonic) কোড এবং এটি সর্বোচ্চ পাঁচটি লেটারের সমন্বয়ে হয় ।

MCQ

১. সহজ উপায়ে কার্যকরী প্রোগ্রাম তৈরির জন্য যে নীতিমালা অনুসরণ করা হয় তাকে কী বলে?
A. প্রোগ্রাম কোড B. প্রোগ্রাম ডিজাইন মডেল
C. অ্যালগরিদম D. ফ্লোচার্ট
Ans: B

২. নিচের কোনটি প্রোগ্রাম ডিজাইন মডেল নয়?
A. স্ট্রাকচার্ড B. অবজেক্ট ওরিয়েন্টেড
C. ভিজ্যুয়াল D. মেশিন ভাষা
Ans: D

৩. স্ট্রাকচার্ড প্রোগ্রামিং এর প্রথম ধারণা দেন কারা?
A. Dennis Ritchie ও Bjarne Stroustrup
B. Corrado Bohm ও Guiseppe Jacopini
C. James Gosling ও Guido van Rossum
D. Edsger W. Dijkstra ও Dennis Ritchie
Ans: B

৪. স্ট্রাকচার্ড প্রোগ্রামিং এ কয়টি মৌলিক স্ট্রাকচার রয়েছে?
A. 2 B. 3 C. 4 D. 5
Ans: B

৫. স্ট্রাকচার্ড প্রোগ্রামিং এর তিনটি স্ট্রাকচার হলো—
A. Class, Object, Method
B. Input, Output, Process
C. Sequence, Decision, Loop
D. Compile, Run, Debug
Ans: C

৬. স্ট্রাকচার্ড প্রোগ্রামিং পদ্ধতি জনপ্রিয় করেন—
A. Corrado Bohm B. James Gosling
C. Edsger W. Dijkstra D. Guido van Rossum
Ans: C

৭. স্ট্রাকচার্ড প্রোগ্রামিং এ সমস্যাকে ছোট অংশে ভাগ করে সমাধান করা হয়, একে বলে—
A. Encapsulation B. Modularization
C. Inheritance D. Polymorphism
Ans: B

৮. নিচের কোনটি স্ট্রাকচার্ড প্রোগ্রামিং ভাষা?
A. Java B. Python C. C D. C#
Ans: C

৯. OOP এর পূর্ণরূপ কী?
A. Object Oriented Process
B. Open Oriented Program
C. Open Object Programming
D. Object Oriented Programming
Ans: D

১০. নিচের কোনটি OOP এর বৈশিষ্ট্য নয়?
A. Encapsulation B. Polymorphism
C. Inheritance D. Compilation
Ans: D

১১. পলিমরফিজম শব্দের অর্থ কী?
A. একরূপ B. বহুরূপ C. দ্রুত কাজ D. পুনরাবৃত্তি
Ans: B

১২. অবজেক্ট বলতে কী বোঝায়?
A. শুধু ডেটা
B. শুধু ফাংশন
C. বাস্তব জগতের কোনো ব্যক্তি বা বস্তু
D. কোডের অংশ
Ans: C

১৩. ক্লাস হলো—
A. ভেরিয়েবল ও মেথডের ব্লু-প্রিন্ট
B. অবজেক্টের বাস্তব রূপ
C. শুধু ডেটা সংরক্ষণ
D. শুধু ফাংশনের সমষ্টি
Ans: A

১৪. এক ক্লাসের বৈশিষ্ট্য অন্য ক্লাস ব্যবহার করলে তাকে কী বলে?
A. Encapsulation B. Polymorphism
C. Inheritance D. Abstraction
Ans: C

১৫. OOP ভাষার উদাহরণ কোনটি?
A. COBOL B. FORTRAN C. Java D. Assembly
Ans: C

১৬. চিত্রভিত্তিক নির্দেশ ব্যবহার করে প্রোগ্রাম লেখাকে কী বলে?
A. Structured Programming
B. Visual Programming
C. Machine Programming
D. Procedural Programming
Ans: B

১৭. প্রথম ভিজ্যুয়াল প্রোগ্রামিং ভাষা কোনটি?
A. Java B. Python C. Visual Basic D. C++
Ans: C

১৮. সব ভিজ্যুয়াল প্রোগ্রাম মূলত কোন মডেলের উপর ভিত্তি করে?
A. Compiler Driven B. Event Driven
C. Procedure Driven D. Machine Driven
Ans: B

লাইব্রেরি ফাংশনঃ- হেডার ফাইল আলোচনাঃ

হেডার ফাইল	ধরণ	লাইব্রেরি ফাংশন
stdio.h	I/O functions	printf()
		scanf()
		getchar()
		putchar()
conio.h	I/O functions	getch()
		clrscr()
string.h	Strings functions	strcat()
		strcmp()
		strcpy()
math.h	Mathematics functions	asin()
		acos()
		atan()
		cos()
		exp()
		fabs()
		sqrt()

কলিং ফাংশন (Calling Function)

সি প্রোগ্রামিংয়ে কলিং ফাংশন হলো সেই ফাংশন যা মেইন ফাংশন বা অন্য ফাংশন থেকে ডাকা হয়। অর্থাৎ মূল প্রোগ্রাম বা অন্য ফাংশনের মাধ্যমে ইউজার-ডিফাইন্ড বা লাইব্রেরি ফাংশনকে কার্যকর করার প্রক্রিয়াই হলো কলিং। সহজভাবে বলা যায়: "যে ফাংশনকে আমরা ব্যবহার করতে চাই, তাকে ডাকা হয় কলিং ফাংশনের মাধ্যমে।"

কলিং ফাংশনের ধরন (Types of Function Call)- সি ভাষায় মূলত **দুটি ধরনের কলিং ফাংশন** ব্যবহার হয়—

১. ফাংশন কল বাই ভ্যালু (Call by Value)

- এই পদ্ধতিতে **ভ্যারিয়েবলের কপি** ফাংশনে পাঠানো হয়।
- ফাংশনের ভেতরে পরিবর্তন মূল ভ্যারিয়েবলে প্রভাব ফেলে না।
- সাধারণত ছোট বা প্রাথমিক ডেটার জন্য ব্যবহার হয়।

উদাহরণ:

```
#include <stdio.h>
void display(int x) {  // Call by Value
   x = x + 10;
   printf("Inside function: %d\n", x);
}
int main() {
   int num = 5;
   display(num);
   printf("In main: %d\n", num);  // মূল ভ্যারিয়েবল অপরিবর্তিত
   return 0;
}
```

২. ফাংশন কল বাই রেফারেন্স (Call by Reference)

- এই পদ্ধতিতে **ভ্যারিয়েবলের ঠিকানা (address)** ফাংশনে পাঠানো হয়।

- ফাংশনের ভেতরে পরিবর্তন মূল ভ্যারিয়েবলে প্রভাব ফেলে।
- সাধারণত বড় ডেটা বা অ্যারে/স্ট্রাকচার পরিবর্তনের জন্য ব্যবহৃত হয়।

উদাহরণ:

```
#include <stdio.h>
void increment(int *p) {  // Call by Reference
   *p = *p + 10;
}
int main() {
   int num = 5;
   increment(&num);
   printf("After function call: %d\n", num);  // মূল ভ্যারিয়েবল পরিবর্তিত
   return 0;
}
```

কলিং ফাংশনের বৈশিষ্ট্য (Characteristics of Calling Function)

1. কলিং ফাংশন **ফাংশনের নাম ব্যবহার করে ডাকা হয়।**
2. প্যারামিটার প্রেরণ করা যায় (অপশনাল)।
3. ফাংশনের রিটার্ন ভ্যালু মূল প্রোগ্রামে ব্যবহার করা যায়।
4. এটি প্রোগ্রামের পুনঃব্যবহারযোগ্যতা ও সংগঠন বাড়ায়।

কলিং ফাংশনের সুবিধা (Advantages)

1. প্রোগ্রাম সুসংগঠিত ও পাঠযোগ্য হয়।
2. কোড পুনঃব্যবহারযোগ্য হয়।
3. ত্রুটি (bugs) সনাক্ত করা সহজ হয়।
4. বড় প্রোগ্রামকে ছোট ছোট অংশে ভাগ করা যায়।

সংক্ষিপ্ত উদাহরণ

```
#include <stdio.h>
// ফাংশন ঘোষণাঃ
int add(int a, int b);
int main() {
   int sum;
   sum = add(10, 20);  // কলিং ফাংশন
   printf("Sum = %d", sum);
   return 0;
}
// ফাংশন সংজ্ঞাঃ
int add(int a, int b) {
   return a + b;
}
```

এখানে main() হলো কলিং ফাংশন এবং add() হলো কলড ফাংশন।

ফাংশন আর্গুমেন্ট

সি প্রোগ্রামিংয়ে ফাংশন আর্গুমেন্ট হলো সেই ডেটা বা মান যা কলিং ফাংশন থেকে কল করা ফাংশনে পাঠানো হয়। আর্গুমেন্টের মাধ্যমে আমরা ফাংশনে ইনপুট প্রদান করি, যাতে ফাংশন নির্দিষ্ট কাজ সম্পন্ন করতে পারে। সহজভাবে: "যে মান ফাংশনকে পাঠানো হয় কাজ করার জন্য, তাকে আর্গুমেন্ট বলে।"

ফাংশন আর্গুমেন্টের ধরন- সি ভাষায় ফাংশন আর্গুমেন্টকে প্রধানত চারটি ধরনে ভাগ করা যায়—

IMPORTANT ICT

1. C++ ভাষা টি উদ্ভাবন করেন কে ?
(A) ভেনিস রিসি (B) ডিগো ভান রসম
(C) জন স্ট্রাস্ট্রপ (D) অ্যাডা লাভলেস
Ans: C

2. সরাসরি কম্পিউটার এর সাথে সংযোগ স্থাপন করে কোন ভাষা ?
(A) মেশিন (B) অ্যাসেম্বলি
(C) উচ্চতর (D) অতিউচ্চতর
Ans: A

3. কোনটি মধ্যম স্তরের ভাষা ?
(A) বেসিক (B) কোবোল
(C) ফোরট্রান (D) সি
Ans: D

4. মেশিন লার্নিং অ্যাপ্লিকেশনে ব্যবহৃত কোনটি?
(A) c++ (B) Algol
(C) Python (D) Fortran
Ans: C

5. ইন্টারপ্রেটার এর কাজ কি?
(A) এক লাইন করে অনুবাদ করে
(B) এক সাথে পুরো প্রোগ্রাম অনুবাদ করে
(C) সব ভুল এক সাথে প্রদর্শন করে
(D) ডিবাগিং ও টেস্টিং ধীরগতিসম্পন্ন
Ans: A

6. উৎস প্রোগ্রাম > ? > বস্তু প্রোগ্রাম (?) চিহ্নিত স্থানে কি হবে
(A) কম্পাইলার (B) ইন্টারপ্রেটার
(C) অ্যাসেম্বলার (D) লিংকার
Ans: A

7. প্রোগ্রাম ডিজাইন মডেলে ২০% -৪০% সময় বায় হয় কোনটিতে?
(A) রক্ষণাবেক্ষণ (B) নিরীক্ষণ
(C) বিশ্লেষণ (D) ডিজাইন
Ans: D

8. নিচের কোনটি ডেটা টাইপ?
(A) main (B) double
(C) printf (D) scanf
Ans: B

9. নিচের কোনটি সঠিক চলক?
(A) 1num (B) 2_num
(C) char (D) num2
Ans: D

10. x এর মান 5 হলে এক্সপ্রেশন x+ = 15 এর মান কত?
(A) 5 (B) 10 (C) 15 (D) 20
Ans: D

11. দুটি স্ট্রিং এর মাঝে তুলনা করার জন্য c প্রোগ্রামিং ল্যাঙ্গুয়েজ এ কোন লাইব্রেরি ফাংশন ব্যবহৃত হয় ?
(A) Strcmp() (B) Strlen()
(C) Strin() (D) Strph()
Ans: A

12. কোন ভাষায় লিখিত প্রোগ্রামের জন্য অনুবাদকের প্রয়োজন হয় না ?
(A) Natural (B) Machine
(C) High Level (D) Assembly
Ans: B

13. মেশিন ভাষার সুবিধা কোনটি ?
(A) প্রোগ্রাম সহজে লেখা যায়
(B) সব ধরনের মেশিনে ব্যবহার উপযোগী
(C) প্রোগ্রাম সরাসরি ও দ্রুত কার্যকরী হয়
(D) প্রোগ্রামের ভুল সহজে শনাক্ত করা যায়
Ans: C

14. কোন ভাষায় লিখিত প্রোগ্রাম কম্পিউটার সরাসরি বুঝতে পারে ?
(A) মেশিন ভাষা (B) উচ্চস্তরের ভাষা
(C) অ্যাসেম্বলি ভাষা (D) চতুর্থ প্রজন্মের ভাষা
Ans: A

15. অ্যাসেম্বলি ভাষা কোন প্রজন্মের ভাষা?
(A) ১ম (B) ২য় (C) ৩য় (D) ৪র্থ
Ans: B

16. কোন ভাষায় হার্ডওয়্যার নিয়ন্ত্রণের পাশাপাশি উচ্চস্তরের ভাষার সুবিধা পাওয়া যায় ?
(A) PASCAL (B) COBOL
(C) C (D) FORTRAN
Ans: C

17. সাংকেতিক ভাষা কোনটি ?
(A) মেশিন ভাষা (B) অ্যাসেম্বলি
(C) উচ্চস্তরের ভাষা (D) অতি উচ্চস্তরের ভাষা
Ans: B

18. প্রোগ্রামের ভাষায় লেখা প্রোগামকে কী বলা হয় ?
(A) গন্তব্য প্রোগ্রাম (B) উৎস প্রোগ্রাম
(C) ভিজুয়্যাল প্রোগ্রাম (D) অনুবাদক প্রোগ্রাম
Ans: B

19. 4 GL বলতে বোঝায়-
(A) অতি উচ্চস্তরের ভাষা (B) উচ্চস্তরের ভাষা
(C) মধ্যম স্তরের ভাষা (D) নিম্নস্তরের ভাষা
Ans: A

SELF TEST

1. C প্রোগ্রামের এন্ট্রি পয়েন্ট কোনটি?
A. start() B. main()
C. init() D. program()
2. C ভাষায় কোনটি সঠিক data type?
A. real B. number
C. int D. digit
3. কোনটি loop statement?
A. if B. switch
C. goto D. for
4. কোন operator টি arithmetic operator?
A. && B. || C. + D. ==
5. C তে array index শুরু হয়—
A. 1 B. 0 C. -1 D. 2
6. String শেষ হয় কোন character দিয়ে?
A. '\n' B. '\t' C. '\0' D. ' '
7. Pointer কী সংরক্ষণ করে?
A. Value B. Address
C. Function D. Character
8. কোন symbol দিয়ে pointer declare করা হয়?
A. & B. * C. # D. %
9. কোনটি user-defined data type?
A. int B. float
C. struct D. char
10. File open করার function কোনটি?
A. fopen() B. fread()
C. fprintf() D. fclose()
11. Structure ব্যবহার করা হয়—
A. একই ধরনের data রাখতে
B. ভিন্ন ধরনের data রাখতে
C. শুধু integer রাখতে
D. শুধু string রাখতে
12. Union এ memory ব্যবহৃত হয়—
A. আলাদা আলাদা B. যোগফল অনুযায়ী
C. সর্বোচ্চ member অনুযায়ী D. শূন্য
13. OOP এর মূল ধারণা কয়টি?
A. 2 B. 3 C. 4 D. 5
14. Class কী?
A. Object B. Blueprint
C. Variable D. Function
15. Object হলো—
A. Class B. Variable
C. Class এর instance D. Method
16. Data hiding কে বলা হয়—
A. Polymorphism B. Inheritance
C. Abstraction D. Encapsulation
17. Inheritance মানে—
A. Code hiding B. Code reuse
C. Data loss D. Overloading
18. Method overriding সম্পর্কিত কোনটি?
A. Compile time B. Run time
C. Preprocessing D. Linking
19. Polymorphism মানে—
A. বহু রূপ B. এক রূপ
C. কোনো রূপ না D. কপি
20. কোনটি access specifier?
A. static B. public
C. void D. return
21. Which is correct C keyword?
A. define B. printf
C. auto D. main
22. Switch statement ব্যবহৃত হয়—
A. Loop এর জন্য
B. Multiple condition এর জন্য
C. File handling এর জন্য
D. Pointer এর জন্য
23. Which is not loop?
A. for B. while
C. do-while D. if
24. sizeof operator কী দেয়?
A. Value B. Address
C. Size D. Type
25. Function এর return type কোনটি?
A. int B. float
C. void D. সবগুলো
26. Recursion মানে—
A. Loop
B. Function নিজেকে কল করা
C. Pointer
D. File
27. Which is string function?
A. strlen() B. sqrt()
C. rand() D. abs()
28. fprintf() ব্যবহৃত হয়—
A. Keyboard input B. Screen output
C. File output D. Error
29. Interface মূলত কী?
A. Class
B. Object
C. Abstract method এর collection
D. Variable
30. Package ব্যবহৃত হয়—
A. Memory free করতে B. Code organize করতে
C. Loop চালাতে D. Data hide করতে
31. Abstraction মানে—
A. Implementation দেখানো
B. প্রয়োজনীয় তথ্য দেখানো

C. সব তথ্য দেখানো
D. Error

32. Constructor কাজ করে—
A. Destroy object B. Initialize object
C. Loop D. Input

33. Destructor ব্যবহৃত হয়—
A. Delete object B. Create object
C. Copy object D. Print

34. Virtual function সম্পর্কিত কোনটি?
A. Static binding B. Dynamic binding
C. Compile only D. Never used

35. Which is not OOP feature?
A. Encapsulation B. Inheritance
C. Polymorphism D. Compilation

36. Call by reference এ পাঠানো হয়—
A. Value B. Address
C. Constant D. Character

37. malloc() কাজ করে—
A. Compile time memory
B. Runtime memory
C. Static memory
D. ROM

38. free() ব্যবহৃত হয়—
A. Memory release B. Memory allocate
C. File close D. Pointer create

39. Which is conditional operator?
A. + B. ? :
C. == D. &&

40. Header file include করা হয়—
A. import B. using
C. #include D. define

41. Object-oriented language কোনটি?
A. C B. Assembly
C. C++ D. HTML

42. Pure OOP language কোনটি?
A. C B. Java
C. C++ D. Python

43. Method overloading ঘটে—
A. Compile time B. Run time
C. Linking time D. Execution end

44. Which supports multiple inheritance?
A. C B. Java
C. C++ D. HTML

45. Static member belongs to—
A. Object B. Class
C. Function D. File

46. this keyword নির্দেশ করে—
A. Previous object B. Current object
C. Parent class D. Child class

47. Default access specifier (Java) —
A. public B. private
C. protected D. default

48. Pointer to structure access করা হয়—
A. . B. -> C. * D. &

49. Which is correct union keyword?
A. union B. structure
C. class D. type

50. OOP তে code reusability আসে—
A. Polymorphism B. Inheritance
C. Abstraction D. Encapsulation

Answer Script

1.B	2.C	3.D	4.C	5.B	6.C	7.B	8.B	9.C	10.A
11.B	12.C	13.C	14.B	15.C	16.D	17.B	18.B	19.A	20.B
21.C	22.B	23.D	24.C	25.D	26.B	27.A	28.C	29.C	30.B
31.B	32.B	33.A	34.B	35.D	36.B	37.B	38.A	39.B	40.C
41.C	42.B	43.A	44.C	45.B	46.B	47.D	48.B	49.A	50.B

Chapter - 9

Introduction to Software Engineering

সফটওয়্যার ইঞ্জিনিয়ারিং (Software Engineering)

সফটওয়্যার ইঞ্জিনিয়ারিং হলো একটি সুশৃঙ্খল, গাণিতিক এবং প্রকৌশলগত পদ্ধতি যা ব্যবহার করে উচ্চমানের সফটওয়্যার তৈরি, পরিচালনা এবং রক্ষণাবেক্ষণ করা হয়। এটি কেবল প্রোগ্রামিং বা কোড করার মধ্যে সীমাবদ্ধ নয়, বরং একটি প্রজেক্টের পরিকল্পনা থেকে শুরু করে সেটি ব্যবহারকারীর হাতে পৌঁছানো পর্যন্ত প্রতিটি ধাপকে নিয়ন্ত্রণ করে। নিচে এর ইতিহাস, প্রকৃতি এবং অন্যান্য বিষয়ের সাথে সম্পর্কের বিস্তারিত আলোচনা করা হলো:

সফটওয়্যার ইঞ্জিনিয়ারিংয়ের ইতিহাস (History)- সফটওয়্যার ইঞ্জিনিয়ারিংয়ের ইতিহাস মূলত কম্পিউটিংয়ের বিবর্তনের সাথে জড়িত:

- **শুরুর দিকে (১৯৫০-১৯৬০):** শুরুতে কোনো নির্দিষ্ট নিয়ম ছিল না। প্রোগ্রামাররা ট্রায়াল অ্যান্ড এরর পদ্ধতিতে কাজ করতেন। হার্ডওয়্যার উন্নত হওয়ার সাথে সাথে সফটওয়্যার জটিল হতে থাকে।
- **সফটওয়্যার ক্রাইসিস (Software Crisis):** ১৯৬০-এর দশকের শেষের দিকে বড় বড় প্রজেক্টগুলো ব্যর্থ হতে শুরু করে। খরচ বেড়ে যাওয়া, সময়মতো ডেলিভারি না দেওয়া এবং অসংখ্য 'বাগ' বা ত্রুটির কারণে একটি বিশৃঙ্খল পরিস্থিতি তৈরি হয়।
- **জন্ম (১৯৬৮):** জার্মানির গারমিশ-পার্টেনকির্চেনে অনুষ্ঠিত NATO Software Engineering Conference-এ প্রথমবারের মতো "Software Engineering" শব্দটি আনুষ্ঠানিকভাবে ব্যবহার করা হয়। এর উদ্দেশ্য ছিল সফটওয়্যার তৈরিতেও সিভিল বা মেকানিক্যাল ইঞ্জিনিয়ারিংয়ের মতো সুনির্দিষ্ট নিয়ম প্রয়োগ করা।
- **আধুনিক যুগ:** ১৯৮০ এবং ৯০-এর দশকে অবজেক্ট অরিয়েন্টেড প্রোগ্রামিং (OOP) এবং বর্তমানে Agile, DevOps ও Cloud Computing-এর মাধ্যমে এটি একটি অত্যন্ত শক্তিশালী শিল্পে পরিণত হয়েছে।

সফটওয়্যার ইঞ্জিনিয়ারিংয়ের প্রকৃতি (Nature)- সফটওয়্যার ইঞ্জিনিয়ারিংয়ের প্রকৃতি বহুমুখী। এর প্রধান বৈশিষ্ট্যগুলো হলো:

- **ইঞ্জিনিয়ারিং ডিসিপ্লিন:** এটি প্রকৌশলবিদ্যার নিয়ম (Design, Development, Testing) মেনে চলে।
- **পদ্ধতিগত প্রক্রিয়া (Systematic Process):** এটি আন্দাজে কাজ না করে **SDLC** (Software Development Life Cycle) অনুসরণ করে।
- **মান নিয়ন্ত্রণ (Quality Control):** সফটওয়্যারটি কতটা নির্ভুল, নিরাপদ এবং ইউজার ফ্রেন্ডলি তা নিশ্চিত করাই এর মূল কাজ।
- **পরিবর্তনশীলতা (Evolutionary):** সফটওয়্যার কখনো শেষ হয় না; সময়ের সাথে সাথে একে আপডেট ও মেইনটেন্যান্স করতে হয়।

অন্যান্য বিষয়ের সাথে সম্পর্ক (Relation to Other Disciplines)- সফটওয়্যার ইঞ্জিনিয়ারিং একা চলে না; এটি অনেকগুলো বিষয়ের সংমিশ্রণ:

ক. কম্পিউটার সায়েন্স (Computer Science)- কম্পিউটার সায়েন্স থিওরি বা তত্ত্ব নিয়ে কাজ করে (যেমন: অ্যালগরিদম ও ডেটা স্ট্রাকচার)। সফটওয়্যার ইঞ্জিনিয়ারিং সেই তত্ত্বগুলোকে কাজে লাগিয়ে বাস্তব সমাধান বা প্রোডাক্ট তৈরি করে।

খ. ম্যানেজমেন্ট সায়েন্স (Management Science)- বড় প্রজেক্ট চালানোর জন্য বাজেট তৈরি, সময়সীমা নির্ধারণ (Planning) এবং লোকবল ব্যবস্থাপনা প্রয়োজন। সফটওয়্যার ইঞ্জিনিয়ারিং এখানে ম্যানেজমেন্টের নীতিগুলো ব্যবহার করে।

গ. সিস্টেম ইঞ্জিনিয়ারিং (System Engineering)- সিস্টেম ইঞ্জিনিয়ারিং হার্ডওয়্যার ও সফটওয়্যার মিলিয়ে পুরো সিস্টেম নিয়ে কাজ করে। সফটওয়্যার ইঞ্জিনিয়ারিং সেই সিস্টেমের একটি অবিচ্ছেদ্য অংশ হিসেবে সফটওয়্যার পার্টটি সামলায়।

ঘ. হিউম্যান ফ্যাক্টরস বা সাইকোলজি (Psychology/UI-UX)- সফটওয়্যার ব্যবহার করবে মানুষ। তাই একজন সফটওয়্যার ইঞ্জিনিয়ারকে ব্যবহারকারীর মনস্তত্ত্ব বুঝতে হয় যাতে ইন্টারফেসটি সহজ এবং কার্যকর হয় (User Experience)।

ঙ. ইকোনমিকস (Economics)- খরচ কমানো এবং লাভের হার নিশ্চিত করা সফটওয়্যার ইঞ্জিনিয়ারিংয়ের একটি বড় দিক। রিসোর্স ম্যানেজমেন্ট এবং কস্ট-বেনিফিট অ্যানালাইসিস এখানে গুরুত্বপূর্ণ।

Software Development Life Cycle (SDLC)

Software Development Life Cycle (SDLC) হলো একটি সুসংগঠিত প্রক্রিয়া যা উচ্চমানের সফটওয়্যার তৈরির জন্য ব্যবহৃত হয়। এটি সফটওয়্যার তৈরির পরিকল্পনা থেকে শুরু করে রক্ষণাবেক্ষণ পর্যন্ত প্রতিটি ধাপকে নির্দিষ্ট করে দেয়। এর মূল লক্ষ্য হলো কম সময়ে ও কম খরচে গ্রাহকের চাহিদা অনুযায়ী সেরা সফটওয়্যার প্রদান করা।

নিচে SDLC-এর ৭টি প্রধান ধাপ বিস্তারিত আলোচনা করা হলো:

Software এর প্রকৃতি (Nature of Software)

১. অস্পৃশ্যতা (Intangibility): সফটওয়্যার কোনো ভৌত বস্তু নয়। এটি মূলত কিছু লজিক্যাল নির্দেশাবলীর সমষ্টি। এটি দেখা বা ব্যবহার করা গেলেও হাত দিয়ে স্পর্শ করা যায় না।

২. ইঞ্জিনিয়ারড, ম্যানুফ্যাকচারড নয় (Engineered, not Manufactured): সাধারণ পণ্য (যেমন গাড়ি বা টিভি) কারখানায় উৎপাদন করা হয়। কিন্তু সফটওয়্যার কোনো কারখানায় তৈরি হয় না; এটি দক্ষ প্রকৌশলী বা প্রোগ্রামারদের দ্বারা ধাপে ধাপে ডিজাইন এবং ডেভেলপ করা হয়। মানুষের বুদ্ধি ও দক্ষতায় ডেভেলপ করা হয়।

৩. **ক্ষয়হীনতা (Does not Wear Out):** হার্ডওয়্যার যেমন ব্যবহারের ফলে পুরনো হয় বা নষ্ট হয়ে যায়, সফটওয়্যার তেমন হয় না। তবে সময়ের সাথে পরিবেশের পরিবর্তনের কারণে এটি কার্যকারিতা হারাতে পারে, যাকে সফটওয়্যার ডিটেরিওরাশন (Deterioration) বলা হয়। সময়ের সাথে আপডেট বা পরিবর্তনের প্রয়োজন হয় ।

৪. সফটওয়্যার কপি করা সহজ (Easily Replicable)

সফটওয়্যারের অন্যতম প্রধান প্রকৃতি হলো এটি একটি ডিজিটাল পণ্য। ভৌত বা ফিজিক্যাল পণ্যের তুলনায় এর উৎপাদন এবং বিতরণের প্রক্রিয়া একদম ভিন্ন। একটি সফটওয়্যার একবার তৈরি (Develop) করার পর এর হুবহু কপি বা প্রতিলিপি তৈরি করতে কোনো অতিরিক্ত শ্রমের প্রয়োজন হয় না।

৫. সফটওয়্যার নির্ভরশীল (Dependency)- হার্ডওয়্যার ও অপারেটিং সিস্টেমের উপর নির্ভরশীল। প্ল্যাটফর্ম পরিবর্তনে সফটওয়্যার পরিবর্তন লাগতে পারে

৬. পরিবর্তনশীলতা (Malleability): সফটওয়্যার অত্যন্ত নমনীয়। প্রয়োজন অনুযায়ী এর কোড পরিবর্তন করে নতুন ফিচার যোগ করা বা পুরনো ভুল সংশোধন করা খুব সহজ। ব্যবহারকারীর চাহিদা অনুযায়ী পরিবর্তন ও উন্নয়ন করা যায়

ব্যবহারিক গুণাবলী (Operational Characteristics)-

সফটওয়্যারটি চলার সময় কেমন কাজ করছে তা এর মাধ্যমে বোঝা যায়।

১. কার্যকারিতা (Functionality): এটি সফটওয়্যারের প্রাথমিক গুণ। অর্থাৎ, সফটওয়্যারটি যে উদ্দেশ্যে তৈরি করা হয়েছে, তা সে সঠিকভাবে করতে পারছে কি না।

২. নির্ভরযোগ্যতা (Reliability): একটি সফটওয়্যার যদি মাঝপথে বন্ধ হয়ে যায় বা ডেটা হারিয়ে ফেলে, তবে তার নির্ভরযোগ্যতা থাকে না। একটি আদর্শ সফটওয়্যার দীর্ঘক্ষণ নিরবচ্ছিন্নভাবে সেবা দেয়।

৩. দক্ষতা (Efficiency): একে Performance-ও বলা হয়। এটি কত দ্রুত রেসপন্স করছে এবং কত কম হার্ডওয়্যার রিসোর্স (RAM/CPU) ব্যবহার করছে, তা এখানে গুরুত্বপূর্ণ।

৪. ব্যবহারযোগ্যতা (Usability): সফটওয়্যারের ডিজাইন এমন হতে হবে যেন একজন নতুন ব্যবহারকারীও কোনো প্রশিক্ষণ ছাড়াই সেটি চালাতে পারেন (User-friendly UI/UX)।

৫. রক্ষণাবেক্ষণযোগ্যতা (Maintainability): সময়ের প্রয়োজনে সফটওয়্যারে পরিবর্তন আনতে হয়। কোড যদি গোছানো থাকে, তবে ডেভেলপাররা সহজে এর সমস্যা সমাধান বা আপডেট করতে পারেন।

৬. স্থানান্তরযোগ্যতা (Portability): এটি কি শুধু উইন্ডোজে চলে? নাকি লিনাক্স বা ম্যাক-এও চলে? যত বেশি প্ল্যাটফর্মে চলবে, তার পোর্টেবিলিটি তত বেশি।

৭. নিরাপত্তা (Security): বর্তমান যুগে এটি সবচেয়ে গুরুত্বপূর্ণ। সাইবার আক্রমণ থেকে ব্যবহারকারীর ব্যক্তিগত তথ্য এবং ডেটাবেস রক্ষা করা সফটওয়্যারের প্রধান গুণ।

৮. পুনঃব্যবহারযোগ্যতা (Reusability): ভালো মানের কোড বারবার লেখা লাগে না। একটি প্রজেক্টের মডিউল বা কোড অন্য প্রজেক্টে ব্যবহার করা গেলে সময় ও শ্রম দুটোই বাঁচে।

Software development model: waterfall

সফটওয়্যার ডেভেলপমেন্টের জগতে Waterfall Model হলো সবচেয়ে পুরনো এবং সহজ একটি মডেল। একে Linear-Sequential Life Cycle Model-ও বলা হয়। এই মডেলে সফটওয়্যার তৈরির প্রতিটি ধাপ ঝরনার পানির মতো ওপর থেকে নিচে ক্রমান্বয়ে নেমে আসে। একটি ধাপ সম্পূর্ণ শেষ না করে পরবর্তী ধাপে যাওয়া যায় না।

নিচে ওয়াটারফল মডেলের ধাপসমূহ এবং এর ভালো-মন্দ দিকগুলো বিস্তারিত আলোচনা করা হলো:

ওয়াটারফল মডেলের ধাপসমূহ (Stages of Waterfall)- এই মডেলে প্রতিটি ধাপের একটি সুনির্দিষ্ট আউটপুট থাকে যা পরবর্তী ধাপের ইনপুট হিসেবে কাজ করে:

Software development model: RDD

সফটওয়্যার ইঞ্জিনিয়ারিংয়ে RDD বলতে সাধারণত Responsibility-Driven Design-কে বোঝায়। এটি মূলত অবজেক্ট-ওরিয়েন্টেড ডিজাইন (Object-Oriented Design) করার একটি বিশেষ পদ্ধতি।

সাধারণত প্রথাগত ডিজাইন পদ্ধতিতে আমরা ডেটা বা তথ্যের ওপর বেশি জোর দেই (যেমন: এই অবজেক্টের কী কী ডেটা থাকবে)। কিন্তু RDD-তে আমরা ফোকাস করি "দায়িত্ব" বা Responsibility-র ওপর। অর্থাৎ, একটি অবজেক্ট কী কী কাজ করবে এবং অন্য অবজেক্টের সাথে কীভাবে যোগাযোগ করবে, সেটিই এখানে মূল বিষয়।

RDD-এর মূল ধারণা: Responsibility (দায়িত্ব)- রেসপন্সিবিলিটি ড্রাইভেন ডিজাইনে দায়িত্বকে দুই ভাগে ভাগ করা হয়:

- **Knowing (জানা):** একটি অবজেক্ট নিজের সম্পর্কে কী কী জানবে বা অন্য কোনো অবজেক্টের রেফারেন্স সম্পর্কে কী জানবে।
- **Doing (করা):** একটি অবজেক্ট নিজে কী কাজ করবে (যেমন: ক্যালকুলেশন) অথবা অন্য অবজেক্টকে দিয়ে কোনো কাজ করিয়ে নেবে।

RDD-এর প্রধান টুল: CRC Card- RDD পদ্ধতিতে ডিজাইন করার সময় CRC Card (Class, Responsibility, and Collaboration) ব্যবহার করা হয়। এটি একটি ফিজিক্যাল বা ডিজিটাল কার্ড যা ডিজাইনারদের চিন্তা করতে সাহায্য করে।

- **Class:** অবজেক্টের নাম।
- **Responsibilities:** অবজেক্টটি কী কী কাজ করবে তার তালিকা।
- **Collaborators:** কাজগুলো করার জন্য সে আর কোন কোন অবজেক্টের সাহায্য নেবে।

RDD পদ্ধতির ধাপসমূহ

১. **প্রয়োজনীয়তা বিশ্লেষণ:** সিস্টেমের মূল কাজগুলো চিহ্নিত করা।

২. **অবজেক্ট বা ক্লাস নির্ধারণ:** কোন কোন ইউনিট বা অবজেক্ট সিস্টেমে থাকবে তা ঠিক করা।

৩. **দায়িত্ব বণ্টন:** কোন অবজেক্ট কোন কাজের জন্য দায়ী থাকবে তা নির্ধারণ করা।

৪. **সহযোগিতা (Collaboration) নিশ্চিত করা:** একটি অবজেক্ট যখন নিজের কাজ নিজে করতে পারবে না, তখন সে কার সাহায্য নেবে তা ঠিক করা।

RDD-এর সুবিধা ও অসুবিধা

সুবিধা (Pros):

- **সহজবোধ্য:** কোড লেখার আগেই সিস্টেমের লজিক বোঝা সহজ হয়।
- **নমনীয়তা:** দায়িত্বগুলো আলাদা থাকায় পরবর্তীতে কোনো পরিবর্তন করা সহজ।
- **Loose Coupling:** অবজেক্টগুলো একে অপরের ওপর খুব বেশি নির্ভরশীল হয় না, ফলে কোড উন্নত মানের হয়।

অসুবিধা (Cons):

- **জটিলতা:** বড় সিস্টেমের ক্ষেত্রে সব দায়িত্ব সঠিকভাবে বণ্টন করা কঠিন হয়ে পড়ে।
- **অভিজ্ঞতা:** এই পদ্ধতিতে ডিজাইন করতে হলে অবজেক্ট-ওরিয়েন্টেড কনসেপ্টে অনেক দক্ষ হতে হয়।

Software development model: V model

V-Model বা Verification and Validation Model হলো Waterfall Model-এর একটি উন্নত ও প্রসারিত রূপ। এখানে, প্রতিটি ডেভেলপমেন্ট ধাপের বিপরীতে একটি নির্দিষ্ট টেস্টিং ধাপ থাকে। এর কাঠামোটি ইংরেজি **'V'** অক্ষরের মতো দেখায় বলে একে V-Model বলা হয়।

এই মডেলের মূল দর্শন হলো—"কোডিং করার আগেই টেস্টিংয়ের পরিকল্পনা করতে হবে।"

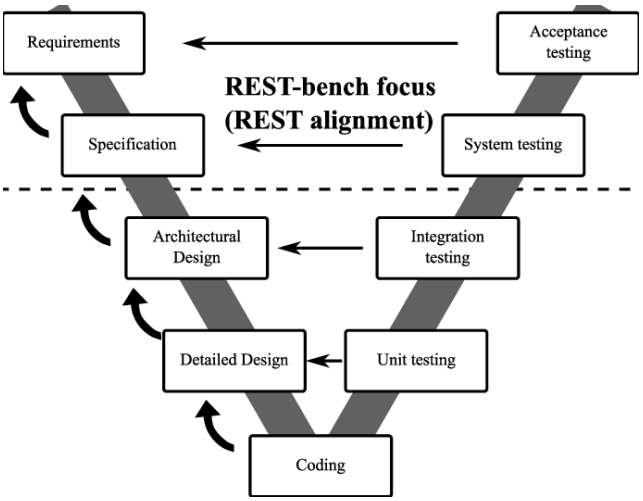
১. V-Model এর দুটি প্রধান দিক- এই মডেলটি মূলত দুটি সমান্তরাল পথের সমন্বয়ে গঠিত:

ক) বাম দিক (Verification Phase):- এটি সফটওয়্যার তৈরির পর্যায়। এখানে কোড লেখার আগে সিস্টেমের প্রয়োজনীয়তা এবং ডিজাইন যাচাই করা হয়।

1. **Requirement Analysis:** ব্যবহারকারীর চাহিদা বুঝে নিয়ে 'Acceptance Test' পরিকল্পনা করা হয়।
2. **System Design:** পুরো সিস্টেমের আর্কিটেকচার তৈরি এবং 'System Test' পরিকল্পনা।
3. **Architecture Design:** মডিউলগুলোর যোগাযোগ ঠিক করা এবং 'Integration Test' পরিকল্পনা।
4. **Module Design (Low Level):** বিস্তারিত লজিক তৈরি এবং 'Unit Test' পরিকল্পনা।

খ) ডান দিক (Validation Phase):- এটি সফটওয়্যার পরীক্ষার পর্যায়। ডেভেলপমেন্ট ধাপের পরিকল্পনাগুলো এখানে বাস্তবে টেস্ট করা হয়।

1. **Unit Testing:** কোডের ছোট ছোট অংশ বা মডিউল পরীক্ষা করা।
2. **Integration Testing:** মডিউলগুলো একত্রে ঠিকমতো কাজ করছে কি না তা দেখা।
3. **System Testing:** পুরো সফটওয়্যারটি একটি পূর্ণাঙ্গ সিস্টেম হিসেবে পরীক্ষা করা।
4. **Acceptance Testing:** ব্যবহারকারীর চাহিদা পূরণ হয়েছে কি না তা যাচাই করা।



V-Model এর সুবিধা (Pros)

- **উচ্চ গুণমান:** প্রতিটি ধাপে টেস্টিং পরিকল্পনা থাকে বলে ভুল হওয়ার সম্ভাবনা খুব কম।
- **শৃঙ্খলিত পদ্ধতি:** ছোট বা মাঝারি প্রজেক্ট যেখানে রিকয়ারমেন্ট একদম স্পষ্ট, সেখানে এটি দারুণ কার্যকর।
- **ত্রুটি আগে ধরা পড়ে:** কোডিং শুরু হওয়ার আগেই ডিজাইনের ভুলগুলো ধরা যায়।
- **সহজ ম্যানেজমেন্ট:** এর প্রতিটি ধাপের নির্দিষ্ট আউটপুট এবং রিভিউ প্রসেস থাকে।

V-Model এর অসুবিধা (Cons)

- **অপরিবর্তনশীল:** প্রজেক্টের মাঝপথে কোনো রিকয়ারমেন্ট পরিবর্তন করা অত্যন্ত কঠিন এবং ব্যয়বহুল।
- **উচ্চ ঝুঁকি:** ওয়াটারফল মডেলের মতোই, একদম শেষ পর্যায়ের আগে কোনো কার্যকরী সফটওয়্যার (Working Prototype) দেখা যায় না।
- **জটিল প্রজেক্টের জন্য অনুপযুক্ত:** বড় বা অবজেক্ট-ওরিয়েন্টেড প্রজেক্টের ক্ষেত্রে এটি খুব একটা সফল নয়।

COCOMO মডেল

COCOMO মডেলের পূর্ণরূপ হলো COnstructive COst MOdel। এটি সফটওয়্যার ইঞ্জিনিয়ারিংয়ের একটি অত্যন্ত জনপ্রিয় মডেল, যা কোনো সফটওয়্যার তৈরি করতে কতটুকু সময়, কতজন জনবল এবং কত টাকা খরচ হবে তার একটি সম্ভাব্য হিসাব (Estimation) করতে ব্যবহৃত হয়। ১৯৮১ সালে ব্যারি বোয়েম (Barry Boehm) এটি উদ্ভাবন করেন। এটি মূলত LOC (Lines of Code) বা সফটওয়্যারের কোড কত বড় হবে তার ওপর ভিত্তি করে কাজ করে।

কোকোমো মডেলের ৩টি শ্রেণি (Software Projects Types)- সফটওয়্যারের জটিলতা অনুযায়ী এই মডেল প্রজেক্টকে তিন ভাগে ভাগ করে:

১. **অর্গানিক (Organic):** ছোট এবং সহজ প্রজেক্ট। যেখানে অভিজ্ঞ টিম পরিচিত পরিবেশে কাজ করে (যেমন: ছোট ডাটাবেস বা সাধারণ ইনভেন্টরি সিস্টেম)।

২. **সেমি-ডিটাচড (Semi-detached):** মাঝারি মানের জটিল প্রজেক্ট। এখানে অভিজ্ঞ ও অনভিজ্ঞ কর্মীর মিশ্রণ থাকে এবং সফটওয়্যারের নিয়মগুলো একটু জটিল হয়।

৩. **এম্বেডেড (Embedded):** অত্যন্ত জটিল প্রজেক্ট। যেখানে কঠোর নিয়মকানুন থাকে (যেমন: এটিএম সফটওয়্যার বা এয়ার ট্রাফিক কন্ট্রোল সিস্টেম)।

Chapter - 10

Data Structure and Algorithm & Combinatorial Optimization

Data Structure

ডেটা স্ট্রাকচার (Data Structure) হলো ডেটা সংরক্ষণ, সাজানো এবং পরিচালনা করার একটি বিশেষ পদ্ধতি, যা ডেটাকে এমনভাবে সংগঠিত করে যাতে তা সহজে অ্যাক্সেস ও ব্যবহার করা যায়, যেমন অ্যারে, লিঙ্কড লিস্ট, স্ট্যাক, কিউ, ট্রি, গ্রাফ ইত্যাদি; এটি প্রোগ্রামিংয়ে অ্যালগরিদমের সাথে অপরিহার্য, যা ডেটা প্রক্রিয়া, অনুসন্ধান, সন্নিবেশ এবং মুছে ফেলার মতো কাজগুলোকে কার্যকরী করে তোলে। সহজ কথায় বলতে গেলে, Data Structure বা ডেটা স্ট্রাকচার হলো কম্পিউটারে ডেটা বা তথ্যকে সুসংগঠিতভাবে সাজিয়ে রাখার একটি পদ্ধতি। এটি ডেটার যৌক্তিক বা গাণিতিক উপস্থাপনা, যা ডেটার মান, তাদের মধ্যকার সম্পর্ক এবং ডেটার উপর প্রয়োগ করা যেতে পারে এমন ফাংশনগুলোকে একসাথে রাখে। ডেটা স্ট্রাকচার ডেটা সংগঠিত করে যাতে অ্যালগরিদমগুলো ডেটা সহজে প্রক্রিয়া করতে পারে, যেমন সার্চিং (খোঁজা) বা সর্টিং (সাজানো)।

প্রধান প্রকারভেদ

১. **আদিম (Primitive):** মৌলিক ডেটা টাইপ যেমন ইন্টিজার, ফ্লোট, বুলিয়ান, ক্যারেক্টার।

২. **অ-আদিম (Non-Primitive):** জটিল ডেটা স্ট্রাকচার যা একাধিক ডেটা সংরক্ষণ করে।

i. **লিনিয়ার (Linear):** ডেটা একটি অনুক্রমিক (sequential) বিন্যাসে থাকে (যেমন: অ্যারে, লিঙ্কড লিস্ট, স্ট্যাক, কিউ) ।

ii. **নন-লিনিয়ার (Non-Linear):** ডেটা অনুক্রমিক নয় (যেমন: ট্রি, গ্রাফ, হ্যাশ টেবিল) ।

ডেটা স্ট্রাকচার বলতে নন-প্রিমিটিভ কে বুঝায়। এদের উদাহরণ -

- **অ্যারে (Array):** একই ধরনের ডেটার একটি নির্দিষ্ট আকারের সংগ্রহ।
- **লিঙ্কড লিস্ট (Linked List):** ডেটা নোডের একটি চেইন, যেখানে প্রতিটি নোড পরবর্তী নোডের ঠিকানা ধারণ করে।
- **স্ট্যাক (Stack):** LIFO (Last-In, First-Out) নীতিতে কাজ করে (যেমন: প্লেটের স্তূপ) ।
- **কিউ (Queue):** FIFO (First-In, First-Out) নীতিতে কাজ করে (যেমন: লাইনে দাঁড়ানো) ।
- **ট্রি (Tree):** হায়ারার্কিক্যাল (hierarchical) বা গাছ-সদৃশ কাঠামো (যেমন: ফাইল সিস্টেম) ।
- **গ্রাফ (Graph):** নোড (vertex) এবং সংযোগকারী (edge) নিয়ে গঠিত (যেমন: সোশ্যাল নেটওয়ার্ক) ।

ডেটা স্ট্রাকচারের অপারেশন

ডেটা স্ট্রাকচারের ওপর সাধারণত যে কাজগুলো বা অপারেশনগুলো চালানো হয়, সেগুলোকে প্রধানত ৬টি ভাগে ভাগ করা যায়। নিচে বিস্তারিত আলোচনা করা হলো:

১. ট্রাভার্সিং (Traversing)- কোনো ডেটা স্ট্রাকচারের প্রতিটি উপাদানের (Element) মধ্য দিয়ে অন্তত একবার যাওয়াকে ট্রাভার্সিং বলে। এটি করা হয় যাতে প্রতিটি ডেটা চেক করা যায় বা প্রিন্ট করা যায়।

- **উদাহরণ:** একটি অ্যারের (Array) শুরু থেকে শেষ পর্যন্ত সব এলিমেন্ট দেখা বা প্রিন্ট করা।

২. ইনসারশন (Insertion)- বিদ্যমান ডেটা স্ট্রাকচারের মধ্যে নতুন কোনো ডেটা বা এলিমেন্ট যোগ করার প্রক্রিয়াকে ইনসারশন বলে। এটি স্ট্রাকচারের শুরুতে, মাঝখানে বা শেষে হতে পারে।

- **উদাহরণ:** একটি স্টুডেন্ট লিস্টে নতুন একজন ছাত্রের নাম যোগ করা।

৩. ডিলিশন (Deletion)- ডেটা স্ট্রাকচার থেকে কোনো অপ্রয়োজনীয় বা নির্দিষ্ট ডেটা সরিয়ে ফেলা বা মুছে ফেলার প্রক্রিয়াকে ডিলিশন বলে।

- **উদাহরণ:** একটি কন্টাক্ট লিস্ট থেকে কোনো পুরনো নম্বর ডিলিট করা।

৪. সার্চিং (Searching)- কোনো বড় ডেটা স্ট্রাকচারের ভেতর থেকে নির্দিষ্ট কোনো ডেটা বা এলিমেন্ট খুঁজে বের করার প্রক্রিয়াকে সার্চিং বলে।

- **উদাহরণ:** ডিকশনারি থেকে একটি নির্দিষ্ট শব্দের অর্থ খুঁজে বের করা। (যেমন: Linear Search বা Binary Search) ।

৫. সর্টিং (Sorting)- ডেটা স্ট্রাকচারের উপাদানগুলোকে একটি নির্দিষ্ট ক্রমানুসারে (যেমন: ছোট থেকে বড় বা বড় থেকে ছোট) সাজানোর প্রক্রিয়াকে সর্টিং বলে।

- **উদাহরণ:** পরীক্ষার রেজাল্ট অনুযায়ী রোল নম্বরগুলো সাজানো।

৬. মার্জিং (Merging)- দুটি ভিন্ন ডেটা স্ট্রাকচারকে (সাধারণত একই ধরণের) একত্রিত করে একটি নতুন ডেটা স্ট্রাকচার তৈরি করার প্রক্রিয়াকে মার্জিং বলে।

Hashing (হ্যাশিং)

ডেটা স্ট্রাকচারে Hashing হলো একটি অত্যন্ত কার্যকর পদ্ধতি যার মাধ্যমে বিশাল পরিমাণ ডেটা থেকে খুব দ্রুত (সাধারণত O(1) সময়ে) কোনো নির্দিষ্ট তথ্য খুঁজে বের করা যায়।

Hash Function (হ্যাশ ফাংশন)

হ্যাশ ফাংশন হলো একটি গাণিতিক অ্যালগরিদম যা যেকোনো সাইজের ডেটাকে (যাকে Key বলা হয়) একটি নির্দিষ্ট দৈর্ঘ্যের সংখ্যা বা ইনডেক্সে রূপান্তর করে। এটি আপনার ইনপুট ডেটাকে একটি ছোট ডিজিটাল ঠিকানায় ম্যাপ করে। অর্থ্যাৎ একটি নির্দিষ্ট ইনডেক্স বা ঠিকানা (Address) প্রদান করে। এই ঠিকানাতেই ডেটা সংরক্ষিত হয়।

একটি আদর্শ হ্যাশ ফাংশনের বৈশিষ্ট্য:

- **একই ইনপুটে একই আউটপুট:** যতবারই একই কি (Key) দেবেন, ততবারই একই ইনডেক্স পাওয়া যাবে।
- **দ্রুত হিসাব:** ফাংশনটি খুব দ্রুত ইনডেক্স বের করতে পারে।
- **ডেটা ডিস্ট্রিবিউশন:** এটি ডেটাগুলোকে মেমোরিতে ছড়িয়ে-ছিটিয়ে এমনভাবে রাখে যাতে কলিশন (একই স্থানে দুটি ডেটা পড়া) কম হয়।

উদাহরণ: সবচেয়ে সহজ হ্যাশ ফাংশন হলো Modulo Method:

$$h(k) = k \bmod m$$

(এখানে k হলো Key, m হলো টেবিলের সাইজ এবং h(k) হলো প্রাপ্ত ইনডেক্স)।

Hash Indices (হ্যাশ ইনডেক্স)

হ্যাশ ফাংশন ব্যবহার করে যে ইনডেক্স বা ঘর নম্বরটি পাওয়া যায়, তাকেই **Hash Index** বলে। এই ইনডেক্সগুলো একটি বড় টেবিলের (যাকে **Hash Table** বলা হয়) নির্দিষ্ট লোকেশন নির্দেশ করে। অর্থ্যাৎ হ্যাশ ফাংশন থেকে প্রাপ্ত যে ইনডেক্সে ডেটা জমা রাখা হয়, তাকেই হ্যাশ ইনডেক্স বলা হয়।

- **কীভাবে কাজ করে:** ধরুন আপনার কাছে ১০০টি ঘর আছে। আপনি ৯৫ নম্বর রোল নম্বরের ডেটা রাখতে চান। হ্যাশ ফাংশন হিসাব করে বের করল এর ইনডেক্স হলো ৫। তাহলে ওই শিক্ষার্থীর সব তথ্য ৫ নম্বর ইনডেক্সে বা ঘরে রাখা হবে।
- **সার্চিং:** যখন আপনার ওই শিক্ষার্থীর তথ্য দরকার হবে, কম্পিউটার আবার ৯৫-কে হ্যাশ ফাংশনে দেবে এবং সাথে সাথে ৫ নম্বর ইনডেক্সে চলে যাবে। ফলে পুরো ১০০টি ঘর খোঁজার প্রয়োজন হয় না।

উপকারিতা: অ্যারে বা লিঙ্কড লিস্টে কোনো তথ্য খুঁজতে গেলে শুরু থেকে সব চেক করতে হয় (Linear Search), যা অনেক সময় নেয়। কিন্তু হ্যাশ ইনডেক্স ব্যবহার করলে সরাসরি সঠিক ঠিকানায় পৌঁছানো যায়, যা সময় অনেক কমিয়ে দেয়।

Static and Dynamic Hashing

হ্যাশিং পদ্ধতিতে ডেটা কীভাবে স্টোর করা হবে এবং মেমোরির আকার পরিবর্তন করা যাবে কি না, তার ওপর ভিত্তি করে একে দুই ভাগে ভাগ করা হয়: **Static Hashing** এবং **Dynamic Hashing**।

১. Static Hashing (স্ট্যাটিক হ্যাশিং)

স্ট্যাটিক হ্যাশিংয়ে হ্যাশ টেবিলের আকার (Bucket সংখ্যা) সবসময় স্থির বা নির্দিষ্ট থাকে। আপনি যখন ডেটা ইনসার্ট করবেন, তখন টেবিলের সাইজ নিজে থেকে বাড়বে না।

- **কীভাবে কাজ করে:** হ্যাশ ফাংশন সবসময় একটি নির্দিষ্ট সীমার মধ্যে ইনডেক্স তৈরি করে। যেমন: যদি টেবিল সাইজ ১০ হয়, তবে ইনডেক্স সবসময় ০ থেকে ৯ এর মধ্যে হবে।
- **সমস্যা:** যদি ডেটার পরিমাণ অনেক বেড়ে যায়, তবে অনেক বেশি Collision (কলিশন) ঘটে। একসময় টেবিল পূর্ণ হয়ে গেলে নতুন ডেটা রাখার জন্য "Overflow" সমস্যা তৈরি হয়।
- **ব্যবহার:** যেখানে ডেটার পরিমাণ আগে থেকেই জানা থাকে এবং খুব একটা পরিবর্তন হয় না, সেখানে এটি কার্যকর।

২. Dynamic Hashing (ডায়নামিক হ্যাশিং)

ডায়নামিক হ্যাশিং (একে Extendible Hashing-ও বলা হয়) এমন একটি পদ্ধতি যেখানে ডেটার পরিমাণের ওপর ভিত্তি করে হ্যাশ টেবিলের আকার রান-টাইমে বাড়ানো বা কমানো যায়।

- **কীভাবে কাজ করে:** এখানে একটি ডিরেক্টরি (Directory) ব্যবহার করা হয়। যখন কোনো একটি বাকেট (Bucket) পূর্ণ হয়ে যায়, তখন পুরো টেবিল নতুন করে তৈরি না করে শুধু ওই নির্দিষ্ট বাকেটটিকে ভেঙে দুই ভাগ করা হয় এবং ডিরেক্টরির আকার বাড়িয়ে দেওয়া হয়।

MCQ

1. Huffman Encoding কোন ধরনের কম্প্রেশন অ্যালগরিদম?
A. Lossy B. Lossless
C. Hybrid D. Block-based
Ans: B

2. Huffman Encoding-এ কোন অক্ষরকে সবচেয়ে ছোট কোড দেওয়া হয়?
A. কম ব্যবহৃত B. র‍্যান্ডম
C. বেশি ব্যবহৃত D. সব অক্ষরকে সমান
Ans: C

3. আধুনিক ডাটাবেস (যেমন MySQL InnoDB) ব্যবহার করে—
A. BST B. AVL Tree
C. B-Tree D. B+ Tree
Ans: D

4. Huffman Tree তৈরিতে কোন ডেটা স্ট্রাকচার ব্যবহার করা হয়?
A. Stack B. Queue
C. Min Priority Queue D. Array
Ans: C

5. Huffman Tree-এর বাম এজে কোন বিট বরাদ্দ করা হয়?
A. 1 B. 0 C. 2 D. -1
Ans: B

6. Huffman Encoding-এর Prefix Property বোঝায়—
A. সব কোড সমান দৈর্ঘ্যের
B. একটি কোড অন্য কোডের প্রিফিক্স নয়
C. কোড র‍্যান্ডম
D. কোড ASCII ভিত্তিক
Ans: B

7. Huffman Encoding ব্যবহৃত হয়—
A. Sorting-এ
B. Searching-এ
C. File Compression-এ
D. Encryption-এ
Ans: C

8. AVL Tree কোন ধরনের ট্রি?
A. Binary Tree B. Heap
C. Self-balancing BST D. Complete Tree
Ans: C

9. AVL Tree-তে Balance Factor-এর সূত্র—
A. Left + Right B. Right − Left
C. Left – Right D. Root − Leaf
Ans: C

10. AVL Tree-তে একটি নোডের Balance Factor হতে পারে—
A. −2, −1, 0, 1, 2 B. −1, 0, 1
C. শুধু 0 D. শুধু 1
Ans: B

11. AVL Tree কেন প্রয়োজন?
A. Sorting-এর জন্য
B. Searching ধীর করার জন্য
C. Tree ব্যালেন্স রাখার জন্য
D. Memory কমানোর জন্য
Ans: C

12. LL Rotation কখন ঘটে?
A. ডান-ডান ক্ষেত্রে B. বাম-বাম ক্ষেত্রে
C. বাম-ডান ক্ষেত্রে D. ডান-বাম ক্ষেত্রে
Ans: B

13. LR Rotation হলো—
A. Single Rotation B. No Rotation
C. Double Rotation D. Circular Rotation
Ans: C

14. AVL Tree-এর সার্চিং টাইম কমপ্লেক্সিটি—
A. O(n) B. O(1)
C. O(log n) D. $O(n^2)$
Ans: C

15. B-Tree মূলত কোথায় ব্যবহৃত হয়?
A. Stack
B. Compiler
C. Database ও File System
D. Graph Traversal
Ans: C

16. B-Tree কোন ধরনের ট্রি?
A. Binary Tree
B. Multi-way Search Tree
C. Skewed Tree
D. Heap
Ans: B

17. B-Tree-এর সব Leaf Node থাকে—
A. ভিন্ন লেভেলে B. র‍্যান্ডম
C. একই লেভেলে D. Root-এ
Ans: C

18. Order m-এর B-Tree-তে সর্বোচ্চ কয়টি Key থাকতে পারে?
A. m B. m+1
C. m−1 D. 2m
Ans: C

19. B-Tree-তে নতুন ডাটা ইনসার্ট হয়—
A. Root-এ B. Internal Node-এ
C. Leaf Node-এ D. যেকোনো নোডে
Ans: C

20. B-Tree কেন ডিস্ক অ্যাক্সেস কমায়?
A. Height বেশি B. Height কম
C. Binary হওয়ায় D. In-memory হওয়ায়
Ans: B

১. IP-V4 এর উদাহরণঃ এটি ৩২-বিটের অ্যাড্রেস এবং সাধারণত চারটি সংখ্যা নিয়ে গঠিত যা ডট (.) দিয়ে আলাদা করা থাকে। প্রতিটি সংখ্যা ০ থেকে ২৫৫ এর মধ্যে হয়।

- **উদাহরণ ১:** 192.168.1.1
- **উদাহরণ ২:** 172.217.166.46 (এটি গুগলের একটি আইপি)
- **উদাহরণ ৩:** 8.8.8.8

২. IPv6 এর উদাহরণঃ এটি ১২৮-বিটের অ্যাড্রেস যা আটটি গ্রুপে বিভক্ত এবং কোলন (:) দিয়ে আলাদা করা থাকে। এখানে সংখ্যা এবং ইংরেজি অক্ষর (Hexadecimal) উভয়ই ব্যবহৃত হয়।

- **উদাহরণ ১:** 2001:0db8:85a3:0000:0000:8a2e:0370:7334
- **উদাহরণ ২:** 2404:6800:4003:c00::64 (সংক্ষিপ্ত রূপ)
- **উদাহরণ ৩:** fe80::1

প্রতিটি আইপি অ্যাড্রেসকে দুইভাগে ভাগ করা যায় –

১. নেটওয়ার্ক আইডি (Network ID)- এটি আইপি অ্যাড্রেসের প্রথম অংশ যা নির্দেশ করে যে ডিভাইসটি কোন নির্দিষ্ট নেটওয়ার্কের অন্তর্ভুক্ত।

- **কাজ:** এটি অনেকটা একটি এলাকার পোস্টাল কোড বা এলাকার নামের মতো। এটি ইন্টারনেট বা রাউটারকে জানায় যে তথ্যটি কোন নেটওয়ার্কে পাঠাতে হবে।

২. হোস্ট আইডি (Host ID)- এটি আইপি অ্যাড্রেসের শেষ অংশ যা ওই নির্দিষ্ট নেটওয়ার্কের ভেতরে থাকা কোনো নির্দিষ্ট ডিভাইসকে শনাক্ত করে।

- **কাজ:** এটি অনেকটা আপনার বাসার হোল্ডিং নম্বরের মতো। নেটওয়ার্কে পৌঁছানোর পর কোন নির্দিষ্ট মোবাইল বা কম্পিউটারে তথ্যটি যাবে, তা হোস্ট আইডি ঠিক করে।

এছাড়া আইপি এড্রেসকে ৫টি ক্লাসে ভাগ করা যায় – Class A, Class B, Class C, Class D, Class E.

১. Class A

- **রেঞ্জ:** ১.০.০.০ থেকে ১২৬.২৫৫.২৫৫.২৫৫ পর্যন্ত।
- **গঠন:** প্রথম ৮-বিট নেটওয়ার্ক আইডি এবং বাকি ২৪-বিট হোস্ট আইডি।
- **ব্যবহার:** এটি অনেক বড় নেটওয়ার্কের জন্য ব্যবহৃত হয় (যেমন: বড় টেক কোম্পানি বা সরকার)। এতে অনেক বেশি ডিভাইস (প্রায় ১ কোটি ৬৭ লাখ) যুক্ত করা যায়।

২. Class B

- **রেঞ্জ:** ১২৮.০.০.০ থেকে ১৯১.২৫৫.২৫৫.২৫৫ পর্যন্ত।
- **গঠন:** প্রথম ১৬-বিট নেটওয়ার্ক আইডি এবং বাকি ১৬-বিট হোস্ট আইডি।
- **ব্যবহার:** মাঝারি আকারের নেটওয়ার্কের জন্য ব্যবহৃত হয় (যেমন: বিশ্ববিদ্যালয় বা বড় কর্পোরেট অফিস)। এতে প্রায় ৬৫,৫৩৬টি ডিভাইস যুক্ত করা যায়।

৩. Class C

- **রেঞ্জ:** ১৯২.০.০.০ থেকে ২২৩.২৫৫.২৫৫.২৫৫ পর্যন্ত।
- **গঠন:** প্রথম ২৪-বিট নেটওয়ার্ক আইডি এবং শেষ ৮-বিট হোস্ট আইডি।
- **ব্যবহার:** ছোট আকারের নেটওয়ার্কের জন্য সবচেয়ে বেশি ব্যবহৃত হয় (যেমন: লোকাল অফিস বা আপনার বাসার রাউটার) । এতে সর্বোচ্চ ২৫৪টি ডিভাইস যুক্ত করা যায়।

৪. Class D

- **রেঞ্জ:** ২২৪.০.০.০ থেকে ২৩৯.২৫৫.২৫৫.২৫৫ পর্যন্ত।
- **ব্যবহার:** এটি সাধারণ কোনো হোস্ট বা ডিভাইসের জন্য ব্যবহার করা হয় না। এটি মূলত মাল্টিকাস্টিং (Multicasting) এর জন্য ব্যবহৃত হয় (যেমন: লাইভ ভিডিও স্ট্রিমিং বা কনফারেন্সিং) ।

৫. Class E

- **রেঞ্জ:** ২৪০.০.০.০ থেকে ২৫৫.২৫৫.২৫৫.২৫৫ পর্যন্ত।
- **ব্যবহার:** এটি সাধারণ মানুষের ব্যবহারের জন্য নয়। এটি ভবিষ্যতে ব্যবহারের জন্য সংরক্ষিত রাখা হয়েছে এবং বিভিন্ন গবেষণা বা পরীক্ষামূলক (Research & Development) কাজে ব্যবহৃত হয়।

Chapter - 12
Operating System

অপারেটিং সিস্টেম (OS)

অপারেটিং সিস্টেম (OS) হলো একটি কম্পিউটারের সবচেয়ে গুরুত্বপূর্ণ সফটওয়্যার। সহজ কথায়, এটি আপনার (ব্যবহারকারী) এবং কম্পিউটারের হার্ডওয়্যারের মধ্যে একটি সেতুবন্ধন হিসেবে কাজ করে। অপারেটিং সিস্টেম ছাড়া কম্পিউটার কেবল একগুচ্ছ ধাতব যন্ত্রাংশ ছাড়া আর কিছুই নয়।

অপারেটিং সিস্টেম হলো এমন একটি সিস্টেম সফটওয়্যার যা কম্পিউটারের মেমরি, প্রসেসর এবং অন্যান্য সমস্ত সফটওয়্যার ও হার্ডওয়্যার নিয়ন্ত্রণ করে। এটি কম্পিউটার চালু হওয়ার সাথে সাথেই মেমরিতে লোড হয় এবং কম্পিউটার বন্ধ না হওয়া পর্যন্ত সক্রিয় থাকে।

অপারেটিং সিস্টেমের কার্যাবলি (Functionalities)

অপারেটিং সিস্টেম একটি কম্পিউটারের সামগ্রিক ব্যবস্থাপকের মতো কাজ করে। এর প্রধান কাজগুলো হলো:

- **মেমরি ম্যানেজমেন্ট (Memory Management):** এটি প্রাইমারি মেমরি বা RAM নিয়ন্ত্রণ করে। কোন প্রোগ্রাম মেমরির কোথায় থাকবে এবং কখন মেমরি খালি করতে হবে, তা OS নির্ধারণ করে।
- **প্রসেসর ম্যানেজমেন্ট (Processor Management):** একটি কম্পিউটারে একসাথে অনেক কাজ চললে কোন কাজটিকে প্রসেসর (CPU) আগে সম্পন্ন করবে, তার সিডিউলিং (Scheduling) করে অপারেটিং সিস্টেম ।
- **ফাইল ম্যানেজমেন্ট (File Management):** হার্ডড্রাইভ বা এসএসডি-তে ডেটা কীভাবে জমা থাকবে, ফাইল কপি করা, মুছে ফেলা বা ডিরেক্টরি (Folder) সাজানোর কাজগুলো এটি করে।
- **ডিভাইস ম্যানেজমেন্ট (Device Management):** ইনপুট ও আউটপুট ডিভাইস (যেমন: মাউস, কিবোর্ড, প্রিন্টার) যাতে হার্ডওয়্যারের সাথে ঠিকমতো কাজ করতে পারে, সেজন্য ড্রাইভারের মাধ্যমে যোগাযোগ রক্ষা করে।
- **নিরাপত্তা রক্ষা (Security):** পাসওয়ার্ড সুরক্ষা প্রদান এবং অননুমোদিত ব্যবহারকারী বা ম্যালওয়্যার থেকে সিস্টেমকে রক্ষা করা এর বড় একটি কাজ।
- **ইউজার ইন্টারফেস প্রদান (User Interface):** ব্যবহারকারী যাতে সহজে কমান্ড দিতে পারে, সেজন্য গ্রাফিক্যাল (GUI) বা টেক্সট (CLI) ভিত্তিক পরিবেশ তৈরি করে।

অপারেটিং সিস্টেমের প্রধান বৈশিষ্ট্যসমূহ (Characteristics)

একটি আদর্শ অপারেটিং সিস্টেমের মধ্যে নিম্নলিখিত বৈশিষ্ট্যগুলো থাকে:

- **মাল্টি-টাস্কিং (Multi-tasking):** আধুনিক OS একই সময়ে একাধিক অ্যাপ্লিকেশন চালানোর ক্ষমতা রাখে। যেমন— গান শোনার পাশাপাশি ইন্টারনেটে ব্রাউজ করা।
- **থ্রুপুট (Throughput):** নির্দিষ্ট সময়ের মধ্যে অপারেটিং সিস্টেম কত বেশি কাজ সম্পন্ন করতে পারে, তাকে থ্রুপুট বলে। ভালো OS-এর থ্রুপুট সবসময় বেশি থাকে।
- **ভার্চুয়াল মেমরি (Virtual Memory):** যখন র‍্যামের জায়গা কমে যায়, তখন অপারেটিং সিস্টেম হার্ডডিস্কের কিছু অংশকে মেমরি হিসেবে ব্যবহার করে কাজ চালিয়ে নিতে পারে।
- **কনকারেন্সি (Concurrency):** একাধিক প্রসেস বা কাজের মধ্যে তালমিল বজায় রাখা, যাতে কোনো কাজ আটকে (Deadlock) না যায়।
- **রিসোর্স শেয়ারিং (Resource Sharing):** কম্পিউটারের বিভিন্ন সম্পদ (রিসোর্স) যেমন প্রিন্টার বা স্ক্যানার একাধিক ব্যবহারকারী বা সফটওয়্যারের মধ্যে সঠিকভাবে বণ্টন করে দেওয়া।
- **এরর ডিটেকশন (Error Detection):** কম্পিউটারের কোনো হার্ডওয়্যার বা সফটওয়্যারে সমস্যা হলে অপারেটিং সিস্টেম স্বয়ংক্রিয়ভাবে ব্যবহারকারীকে সতর্ক করে।

অপারেটিং সিস্টেমের প্রকারভেদ

১. ব্যাচ অপারেটিং সিস্টেম (Batch OS)- এটি কম্পিউটারের আদি যুগের সিস্টেম। এখানে ব্যবহারকারী সরাসরি কম্পিউটারের সাথে যোগাযোগ করে না।

- **কাজ:** একই ধরনের কাজগুলোকে (Jobs) পাঞ্চ কার্ডে অফলাইনে তৈরি করে একটি 'ব্যাচ' হিসেবে অপারেটরকে দেওয়া হতো। কম্পিউটার একে একে কাজগুলো শেষ করতো।
- **ব্যবহার:** বর্তমানে বেতন শিট (Payroll) বা বড় বিলিং সিস্টেমে এটি ব্যবহৃত হয়।

২. মাল্টিপ্রোগ্রামিং/মাল্টিটাস্কিং OS- একটি মাত্র প্রসেসরে যখন একই সময়ে একাধিক প্রোগ্রাম বা কাজ চলে, তখন তাকে মাল্টিটাস্কিং বলে।

- **কাজ:** আসলে প্রসেসর খুব দ্রুত একটি কাজ থেকে অন্য কাজে সুইচ করে, ফলে মনে হয় সব কাজ একসাথে চলছে।
- **উদাহরণ:** উইন্ডোজ বা অ্যান্ড্রয়েড (গান শোনার পাশাপাশি ব্রাউজ করা)।

৩. মাল্টিপ্রসেসর সিস্টেম- এখানে একটি কম্পিউটারে একাধিক CPU (প্রসেসর) থাকে যা একই মেমরি এবং বাস শেয়ার করে কাজ করে।

- **সুবিধা:** এর মাধ্যমে অত্যন্ত দ্রুত কাজ করা যায়। একে 'প্যারালাল সিস্টেম'ও বলা হয়।
- **উদাহরণ:** আধুনিক সার্ভার বা সুপার কম্পিউটার।

৪. ডিস্ট্রিবিউটেড অপারেটিং সিস্টেম- যখন অনেকগুলো আলাদা কম্পিউটার (Nodes) একটি নেটওয়ার্কের মাধ্যমে যুক্ত থাকে কিন্তু ব্যবহারকারীর কাছে মনে হয় সেটি একটিই কম্পিউটার।

- **কাজ:** কাজগুলো বিভিন্ন কম্পিউটারের প্রসেসরের মধ্যে ভাগ করে দেওয়া হয়।

৫. নেটওয়ার্ক অপারেটিং সিস্টেম (NOS)- এটি মূলত সার্ভারে চলে এবং নেটওয়ার্কে যুক্ত অন্যান্য কম্পিউটারের ডেটা, নিরাপত্তা এবং অ্যাপ্লিকেশন পরিচালনা করে।

- **উদাহরণ:** Microsoft Windows Server, Linux.

৬. রিয়েল-টাইম অপারেটিং সিস্টেম (RTOS)- যেখানে সময়ের বাধ্যবাধকতা অনেক বেশি। নির্দিষ্ট সময়ের মধ্যেই কাজ শেষ করতে হয়।

- **ব্যবহার:** মিসাইল কন্ট্রোল সিস্টেম, এটিএম মেশিন, স্যাটেলাইট নিয়ন্ত্রণ।

৭. এমবেডেড অপারেটিং সিস্টেম- এটি কোনো বড় সিস্টেমের অংশ হিসেবে নির্দিষ্ট একটি কাজ করার জন্য তৈরি করা হয়।

- **ব্যবহার:** ওয়াশিং মেশিন, মাইক্রোওয়েভ ওভেন, স্মার্ট ওয়াচ বা গাড়ির ড্যাশবোর্ড।

৮. সিঙ্গল-ইউজার/সিঙ্গল-টাস্কিং- এই সিস্টেমে একজন ব্যবহারকারী এক সময়ে মাত্র একটি কাজ করতে পারেন।

- **উদাহরণ:** পুরোনো MS-DOS অপারেটিং সিস্টেম।

৯. ইউজার ইন্টারফেস (UI) অনুযায়ী

এটি দুই প্রকার:

- **CLI (Command Line Interface):** লিখে কমান্ড দিতে হয় (যেমন: Linux Terminal)।
- **GUI (Graphical User Interface):** আইকন ও মেনু ব্যবহার করা যায় (যেমন: Windows, macOS)।

১০. টাইম শেয়ারিং অপারেটিং সিস্টেম

এটি মাল্টি-ইউজার সিস্টেমের একটি রূপ। এখানে প্রসেসরের সময়কে (Time Slice) ছোট ছোট ভাগে অনেক ব্যবহারকারীর মধ্যে ভাগ করে দেওয়া হয়।

- **সুবিধা:** প্রতিটি ব্যবহারকারী মনে করেন তারা একাই পুরো সিস্টেম ব্যবহার করছেন।

১১. ভার্চুয়াল স্টোরেজ অপারেটিং সিস্টেম- যখন কোনো বড় সফটওয়্যার চালানোর জন্য র‍্যামে (RAM) পর্যাপ্ত জায়গা থাকে না, তখন অপারেটিং সিস্টেম হার্ডড্রাইভের কিছু অংশকে মেমরি হিসেবে ব্যবহার করে। একেই ভার্চুয়াল মেমরি বা স্টোরেজ ম্যানেজমেন্ট বলে।

জনপ্রিয় কিছু অপারেটিং সিস্টেমের তালিকা

ডিভাইসের ধরন	জনপ্রিয় অপারেটিং সিস্টেম
ডেস্কটপ/ল্যাপটপ	Windows, macOS, Linux, ChromeOS
স্মার্টফোন	Android, iOS
সার্ভার	Linux, Windows Server, Unix

কেন এটি প্রয়োজনীয়তা

অপারেটিং সিস্টেম ছাড়া আপনি কোনো অ্যাপ্লিকেশন (যেমন: ব্রাউজার, গেম বা এমএস ওয়ার্ড) চালাতে পারবেন না। এটি কম্পিউটারের সুরক্ষা নিশ্চিত করে এবং হার্ডওয়্যারের জটিলতাগুলো ব্যবহারকারীর কাছ থেকে আড়ালে রেখে কম্পিউটার ব্যবহারকে সহজ করে তোলে।

বিসিএস প্রশ্ন ব্যাংক

47th BCS Question

১. একটি কম্পিউটারের প্রোসেসর ক্লক স্পিড ৪.০০ গিগা হার্জ হলে এর ক্লক মাইকেল টাইম কত?
ক) ২.৫ ন্যানো সেকেন্ড (ns)
খ) ২.৫ মাইক্রো সেকেন্ড (ms)
গ) 8 (ms)
ঘ) 8 (ns)

$f = 4.00\ \text{GHz} = 4.00 \times 10^9\ \text{Hz}$
$T = \frac{1}{f} = \frac{1}{4.00 \times 10^9}\ \text{seconds}$
$T = 0.25 \times 10^{-9}\ \text{seconds}$
$T = 2.5 \times 10^{-10}\ \text{seconds}$
$T = 0.25\ \text{nanoseconds (ns)}$

Ans: A

২. Precision Agriculture এ সাধারণত নিচের কোন প্রযুক্তি ব্যবহৃত হয়?
ক) ইনফ্রা রেড ইমেজিং
খ) আই.ও.টি (IoT), সেন্সর
গ) তার মাধ্যম সম্পন্ন নেটওয়ার্ক
ঘ) ও.এল.ই.ডি (OLED) ডিসপ্লে
Ans: B

৩. অপারেটিং সিস্টেমে ভার্চুয়াল মেমোরি ব্যবহার করা হয়-
ক) অনেক বেশি ডেটা সংরক্ষণের জন্য
খ) ক্লাউডে ডেটা সংরক্ষণের জন্য
গ) সেকেন্ডারি স্টোরেজ ব্যবহার করে RAM বাড়াতে
ঘ) এক্সটারনাল মেমোরি সংযোগের জন্য
Ans: C

৪. কম্পিউটার সিস্টেমের বেঞ্চমার্কিং করা হয় কী পরিমাপের জন্য?
ক) সিস্টেমের দাম
খ) সিস্টেমের কর্ম ক্ষমতা (Performance)
গ) শুধু বিদ্যুৎ শক্তি খরচের পরিমাণ
ঘ) স্টোরেজের ধারণ ক্ষমতা
Ans: B

৫. নিচের কোন ডিভাইসটি প্রধানত এম্বেডেড সিস্টেম ব্যবহৃত হয়?
ক) রাউটার
খ) সুপার কম্পিউটার
গ) হাই-অ্যান্ড সার্ভার
ঘ) মাইক্রোকন্ট্রোলার
Ans: D

৬. ই-কমার্সে সুরক্ষিত অনলাইন লেনদেনে প্রধানত কোন প্রটোকল ব্যবহৃত হয়?
ক) DHCP
খ) SMTP
গ) HTTPS
ঘ) ARP
Ans: C

৭. বিভিন্ন নেটওয়ার্কের মধ্যে যোগাযোগ স্থাপনের জন্য নিচের কোন ডিভাইসটি ব্যবহৃত হয়?
ক) রাউটার
খ) সুইচ
গ) ব্রিজ
ঘ) হাব
Ans: A

৮. ক্লাউড কম্পিউটিং এর কোন মডেলটি অ্যাপ্লিকেশন তৈরি করার জন্য প্রোগ্রামারদেরকে প্লাটফর্ম সরবরাহ করে?
ক) laaS
খ) SaaS
গ) PaaS
ঘ) DaaS
Ans: C

৯. একটি কম্পিউটার সিস্টেমে $(১১০০১০১১)_২$ বাইনারি সংখ্যাটির মান ডেসিমেল এ কত হবে?
ক) – ৫২
খ) – ৫৩
গ) ২০৩
ঘ) উপরের সবকটি হতে পারে
Ans: D

১০. কোন CPU আর্কিটেকচার স্মার্টফোনে বেশি ব্যবহৃত হয়?
ক) X86
খ) X64
গ) Qualcomm
ঘ) RISC
Ans: D

১১. কম্পিউটার টার্ন অন এর সময় সঠিক অর্ডার নিচের কোনটি?
ক) POST → Kernel → Bootloader
খ) Kernel → POST → Bootloader
গ) Kernel → Bootloader → POST
ঘ) POST → Bootloader → Kernel
Ans: D

১২. কোন ধরনের Storage Device সবচেয়ে দ্রুত গতি সম্পন্ন?
ক) HDD
খ) Floppy Disk
গ) SSD
ঘ) SSHD
Ans: C

১৩. ধরা যাক Algorithm A এর running time $O(n^2)$ এবং Algorithm B এর running time $O(n)$। তাহলে নিচের কোনটি সবচেয়ে সঠিক?
ক) Algorithm A, Algorithm B এর চেয়ে ধীর গতির
খ) Algorithm A, Algorithm B এর চেয়ে দ্রুত গতির
গ) Algorithm A, Algorithm B এর চেয়ে asymptotically ধীর গতির
ঘ) Algorithm B সর্বদা Algorithm A এর চেয়ে দ্রুত চলে
Ans: C

১৪. LLM চালানোর জন্য নিম্নোক্ত কম্পিউটারের কোন যন্ত্রাংশ সবচেয়ে বেশি গুরুত্বপূর্ণ?
ক) RAM
খ) Processor
গ) Graphics Card
ঘ) Storage Device
Ans: C

১৫. Quantum Computing এর জনক কাকে মনে করা হয়?
ক) David Deutsch
খ) Richard Feynman
গ) Paul Benloff
ঘ) Alexei Kitaev
Ans: A

Model test-03

1. C ভাষায় কোনটি storage class specifier?
A. auto B. register
C. static D. সবগুলো

2. Global variable-এর scope কোথায় থাকে?
A. শুধু ফাংশনের ভিতরে B. শুধু ব্লকের ভিতরে
C. পুরো প্রোগ্রামে D. শুধু main() এ

3. কোন অপারেটর দিয়ে pointer-এর value পাওয়া যায়?
A. & B. * C. -> D. .

4. NULL pointer মানে কী?
A. 1 B. -1 C. 0 D. Garbage

5. strcmp() ফাংশন ব্যবহৃত হয়—
A. String copy B. String compare
C. String concatenate D. String input

6. feof() ফাংশন কী চেক করে?
A. File open B. File close
C. End of file D. File error

7. Structure pointer মেম্বার অ্যাক্সেস অপারেটর কোনটি?
A. . B. * C. -> D. &

8. Inline function-এর সুবিধা কী?
A. Code বড় হয় B. Execution দ্রুত
C. Memory বেশি লাগে D. Security বাড়ে

9. Class variable কে আর কী বলা হয়?
A. Instance variable B. Static variable
C. Local variable D. Final variable

10. Method overriding হয়—
A. Compile time B. Run time
C. Link time D. Load time

11. Abstract class-এর বৈশিষ্ট্য কোনটি?
A. Object তৈরি হয়
B. Constructor নেই
C. Abstract method থাকতে পারে
D. Static হয়

12. Interface implement করতে কোন keyword ব্যবহৃত হয়?
A. extends B. implements
C. inherit D. override

13. Multiple inheritance সমস্যা তৈরি করে—
A. Runtime error B. Diamond problem
C. Syntax error D. Memory leak

14. Getter ও Setter ব্যবহৃত হয়—
A. Polymorphism B. Encapsulation
C. Inheritance D. Abstraction

15. Dynamic binding ঘটে—
A. Compile time B. Run time
C. Preprocessing D. Linking

16. Object-Oriented ভাষা নয় কোনটি?
A. Java B. C++
C. Python D. C

17. this keyword কী নির্দেশ করে?
A. Class B. Method
C. Current object D. Parent class

18. Final method—
A. Override করা যায় B. Override করা যায় না
C. Abstract হয় D. Private হয়

19. Access modifier নয় কোনটি?
A. public B. private
C. protected D. package

20. OOP-এ coupling কম হওয়া মানে—
A. Code জটিল B. Module স্বাধীন
C. Speed কম D. Error বেশি

21. Software Engineering-এর জনক কে?
A. Dennis Ritchie B. Barry Boehm
C. Winston Royce D. Alan Turing

22. SRS ডকুমেন্টে কোনটি থাকে?
A. Source code B. User requirement
C. Test result D. Bug list

23. Feasibility Study-এর ধরণ নয়—
A. Technical B. Economic
C. Operational D. Graphical

24. Prototyping model-এর সুবিধা—
A. Late feedback B. Early feedback
C. No testing D. High risk

25. Incremental model-এ সফটওয়্যার ডেলিভারি হয়—
A. একবারে B. ধাপে ধাপে
C. শেষে D. Random

26. Cohesion বেশি হলে—
A. Design ভালো B. Design খারাপ
C. Error বেশি D. Maintenance কঠিন

27. White-box testing সম্পর্কিত—
A. Internal structure B. User interface
C. Requirement D. Performance

28. Black-box testing ভিত্তি করে—
A. Code B. Design
C. Specification D. Algorithm

29. Debugging মানে—
A. Error তৈরি B. Error খোঁজা ও ঠিক করা
C. Testing D. Documentation

30. Acceptance testing করে—
A. Developer B. Tester
C. User D. Manager

31. Project scheduling tool নয়—
A. Gantt chart B. PERT
C. CPM D. DFD
32. Risk exposure = ?
A. Probability × Loss B. Time × Cost
C. Size × Effort D. Risk × Schedule
33. Reusability বাড়ে—
A. Structured design B. Modular design
C. Monolithic design D. Ad-hoc design
34. Software maintenance-এ সবচেয়ে বেশি খরচ হয়—
A. Corrective B. Adaptive
C. Perfective D. Preventive
35. UI Design-এর Golden Rule নয়—
A. Consistency B. Feedback
C. Complexity D. Error handling
36. Data Structure কী?
A. Data storage technique
B. Data organization method
C. Data processing
D. Data transmission
37. Asymptotic notation নয়—
A. Big-O B. Big- Ω
C. Big-θ D. Big-Δ
38. Selection Sort-এর বৈশিষ্ট্য—
A. Stable B. Unstable
C. Recursive D. Non-comparison
39. Best case Bubble Sort-এর সময়—
A. $O(n^2)$ B. O(log n)
C. O(n) D. O(n log n)
40. Tail recursion-এর সুবিধা—
A. Memory বেশি B. Optimization সহজ
C. Slow D. Complex
41. Doubly Linked List-এর সুবিধা—
A. একদিকে traversal B. দুইদিকে traversal
C. Fixed size D. Less memory
42. Stack-এর application নয়—
A. Function call
B. Expression evaluation
C. BFS
D. Undo operation
43. Deque মানে—
A. Single ended queue
B. Double ended queue
C. Priority queue
D. Circular queue
44. Hash table-এ Load factor = ?
A. n B. m
C. n/m D. m/n
45. Rehashing করা হয়—
A. Collision কমাতে B. Load factor বেশি হলে
C. Searching বাড়াতে D. Sorting করতে
46. Height-balanced tree হলো—
A. BST B. AVL
C. Heap D. B-tree
47. Heap property সম্পর্কিত—
A. Parent □ Child B. Sorted order
C. Height balance D. BST rule
48. Expression tree-এর leaf node থাকে—
A. Operator B. Operand
C. Function D. Pointer
49. Adjacency matrix-এর space complexity—
A. O(V) B. O(E) C. $O(V^2)$ D. O(V+E)
50. BFS ব্যবহার করা হয়—
A. Shortest path B. MST
C. Cycle detection D. Sorting
51. DFS-এর application—
A. Topological sort B. Level order
C. Shortest path D. Scheduling
52. Dijkstra Algorithm কাজ করে না—
A. Positive weight B. Zero weight
C. Negative weight D. Directed graph
53. Greedy Algorithm সবসময়—
A. Optimal solution দেয়
B. Approximate solution দেয়
C. Local choice নেয়
D. DP ব্যবহার করে
54. Matrix Chain Multiplication কোনটি?
A. Greedy B. DP
C. Backtracking D. Brute force
55. Principle of Optimality সম্পর্কিত—
A. Greedy
B. Divide & Conquer
C. Dynamic Programming
D. Backtracking
56. Backtracking-এ pruning মানে—
A. Branch বাড়ানো
B. অপ্রয়োজনীয় branch বাদ
C. Loop
D. Sorting
57. Hamiltonian path সমস্যা—
A. P problem B. NP-Complete
C. Polynomial D. Linear
58. Time complexity of Merge Sort—
A. $O(n^2)$ B. O(n log n)
C. O(n) D. O(log n)
59. Stable sort নয়—
A. Bubble B. Insertion
C. Merge D. Selection

91. NAT-এর কাজ—
A. Encryption B. IP translation
C. Routing D. Switching

92. CIDR ব্যবহৃত হয়—
A. Classful addressing
B. Classless addressing
C. Fixed routing
D. Static IP

93. ICMP ব্যবহৃত হয়—
A. Data transfer B. Error reporting
C. Routing D. Security

94. RIP metric—
A. Bandwidth B. Delay
C. Hop count D. Cost

95. Distance Vector algorithm-এর সমস্যা—
A. Flooding B. Count to infinity
C. Loop-free D. Fast convergence

96. Network congestion সমাধান—
A. Packet drop B. Traffic shaping
C. Routing loop D. Collision

97. QoS নিশ্চিত করে—
A. Reliability B. Delay, bandwidth
C. Security D. Encryption

98. Socket হলো—
A. Process
B. Interface
C. Communication endpoint
D. Protocol

99. Client request পাঠায়—
A. Response B. ACK
C. Request D. Segment

100. Computer Network-এর প্রধান লক্ষ্য—
A. Cost increase B. Resource sharing
C. Complexity D. Isolation

Answer Script

1.B	2.C	3.A	4.D	5.B	6.A	7.C	8.D	9.B	10.A
11.C	12.D	13.B	14.A	15.C	16.B	17.D	18.A	19.C	20.B
21.A	22.D	23.C	24.B	25.A	26.C	27.D	28.B	29.A	30.C
31.B	32.A	33.D	34.C	35.B	36.A	37.C	38.D	39.B	40.A
41.C	42.B	43.D	44.A	45.C	46.B	47.A	48.D	49.C	50.B
51.A	52.C	53.B	54.D	55.A	56.C	57.B	58.D	59.A	60.C
61.B	62.D	63.A	64.C	65.B	66.A	67.D	68.C	69.B	70.A
71.C	72.B	73.D	74.A	75.C	76.B	77.A	78.D	79.C	80.B
81.A	82.C	83.B	84.D	85.A	86.C	87.B	88.D	89.A	90.C
91.B	92.A	93.D	94.C	95.B	96.A	97.C	98. D	99.B	100.A